

Д. Бертсекас

Р. Галлагер

СЕТИ

ПЕРЕДАЧИ ДАННЫХ



Издательство «Мир»

DATA NETWORKS

Dimitri Bertsekas

Massachusetts Institute of Technology

Robert Gallager

Massachusetts Institute of Technology

Prentice-Hall International, Inc.

Д. Бертсекас
Р. Галлагер

СЕТИ ПЕРЕДАЧИ ДАННЫХ

*Перевод с английского
кандидатов техн. наук*

Н. Б. ЛИХАНОВА, В. А. МИХАЙЛОВА и
С. П. ФЕДОРЦОВА

под редакцией

д-ра техн. наук Б. С. ЦЫБАКОВА



Москва «Мир» 1989

ББК 32.97
Б52
УДК 681.3

Бертсекас Д., Галлагер Р.

Б52 Сети передачи данных: Пер. с англ. — М.: Мир, 1989. —
544 с., ил.

ISBN 5-03-000639-7

В книге известных американских ученых рассматриваются архитектура сетей, методы передачи данных по линиям связи, модели сетей, а также вопросы маршрутизации и управления потоками данных. Приводятся примеры реализации сетей.

Для специалистов в области информатики, связи и вычислительной техники, а также студентов старших курсов соответствующих специальностей вузов.

Б 2404070000-447 148-90
041 (01)-89

ББК 32.97

Редакция литературы по информатике и робототехнике

ISBN 5-03-000639-7 (русск.)
ISBN 0-13-196825-4 (англ.)

© 1987 by Prentice-Hall, Inc.
© перевод на русский язык, «Мир», 1989

Предисловие редактора перевода

Сети передачи данных — новое направление науки и техники, которое бурно развивается. Создание сетей потребовало использования в системах связи новой технической базы, новых принципов передачи и коммутации, значительно расширило сферу применений и виды коммуникационных услуг.

Книга, предлагаемая вниманию читателей, посвящена сетям передачи данных. В ней рассматриваются основные принципы построения и функционирования сетей. Представлена семиуровневая модель сети, рекомендованная Международной организацией по стандартам. Подробно обсуждаются стратегии подтверждения правильной передачи пакетов и сообщений, а также способы восстановления нормальной работы сети после возникновения ошибок и неисправностей. Дается описание способов идентификации и адресации в сетях. Много внимания уделяется множественному доступу в локальных, спутниковых и радиосетях. Представлены все известные подходы к задачам маршрутизации и управления потоками в сетях.

В книге удачно сочетаются описательный и аналитический материалы. Используемые математические модели отражают работу реальных сетей. Для упрощения анализа рассматриваются независимые пуассоновские входные потоки, а в качестве основной характеристики — средняя задержка пакета.

Особый интерес представляют те разделы книги, в которых приводятся результаты, полученные авторами. К ним относятся разделы о маршрутизации, случайном множественном доступе и проверке корректности протоколов автоматического запроса повторной передачи.

Следует отметить, что не все вопросы, рассматриваемые в этой книге, можно считать полностью изученными. Так, лишь на начальной стадии находятся исследования по пакетным радиосетям. Далеко неудовлетворительными являются методы расчета вероятностных характеристик длин очередей пакетов в узлах сетей. Еще много усилий ученых и разработчиков потребует рассмотрение проблем оптимальной маршрутизации и управления потоками. Необходима дальнейшая разработка целого ряда теоретических и практических вопросов случайного множественного доступа.

При работе над переводом возникали определенные терминологические трудности. Насколько возможно, мы придерживались уже принятых терминов. В тех случаях, когда появлялись неизвестные или малораспространенные термины, для них брались русские эквиваленты или оставлялась английская аббревиатура. При переводе были исправлены замеченные опечатки, имевшиеся в английском издании. Большую помощь в этом оказали авторы книги.

Книга представляет несомненный интерес для специалистов в области техники и теории связи, вычислительной техники и информатики, а также для студентов соответствующих специальностей.

Б. С. Цыбаков

ЛИТЕРАТУРА

Поскольку в книге мало цитируются работы советских авторов, укажем некоторые хорошо известные книги.

Авен О. И., Гурин М. Н., Коган Я. А. Оценка качества и оптимизация вычислительных систем. — М.: Наука, 1981.

- Богуславский Л. Б. Управление потоками данных в сетях ЭВМ. — М.: Энергоатомиздат, 1984.
- Бойченко Е. В., Кальфа В., Овчинников В. В. Локальные вычислительные сети. — М.: Радио и связь, 1985.
- Захаров Г. П. Методы исследования сетей передачи данных. — М.: Радио и связь, 1982.
- Лазарев В. Г. Сетевые протоколы и управление в распределенных вычислительных системах. — М.: Наука, 1986.
- Мизин И. А., Богатырев В. А., Кулешов А. П. Сети коммутации пакетов. — М.: Радио и связь, 1986.
- Прангишвили И. В., Подлазов В. С., Стецюра Г. Г. Локальные микропроцессорные вычислительные сети. — М.: Наука, 1985.
- Суздалев А. В., Чугреев О. С. Передача данных в локальных сетях связи. — М.: Радио и связь, 1987.
- Самойленко С. И. Сети ЭВМ. — М.: Наука, 1986.

Предисловие

За прошедшие пятнадцать лет сети передачи данных прошли путь от состояния, когда они проектировались весьма нестандартными методами, сильно зависящими от имеющихся технических средств, до состояния, когда появилось определенное концептуальное понимание многих вопросов, лежащих в их основе. Главная цель авторов данной книги состоит в том, чтобы это концептуальное понимание передать читателю.

Предыдущие книги по этой тематике можно разделить на две большие группы. Первую из них составляют книги, имеющие в основном описательный характер, причем наибольшее внимание в них уделяется существующим сетям и некоторым аспектам их работы. К этой группе относятся, например, книги Таненбаума [227] и Сталлингса [220]. Вторую группу составляют книги, посвященные в первую очередь анализу характеристик работы сетей. К этой группе относятся, например, книги Клейнрока [144], Хейза [115], Стака и Артурза [221]. В данной книге в отличие от предыдущих описание и анализ сбалансированы. Описательный материал используется для иллюстрации основных концепций, а аналитический материал дается для обеспечения более глубокого понимания этих концепций. По-видимому, в области сетей передачи данных продолжение разделения описания и анализа становится неразумным из-за того, что основные концепции уже разработаны. Наибольшее понимание достигается при внимательном изучении этих концепций.

Книга предназначена для использования на разных уровнях — от факультативных занятий на старших курсах вузов до курса лекций для аспирантов первого года обучения и более глубокого курса для аспирантов последних лет обучения. Она может использоваться также как справочное пособие при проектировании и проведении исследований. Материал был включен в ряд курсов для аспирантов Массачусетского технологического института (МТИ) и несколько коротких курсов различных уровней. Предполагается, что читатель знаком с основами элементарной теории вероятностей и имеет некоторую подготовку в области электротехники и вычислительной техники; этого достаточно для понимания излагаемого материала.

На протяжении всей книги основные концепции и принципы сначала объясняются с помощью простых нематематических средств, а затем рассматриваются вопросы построения математической модели и проводится математический анализ. После этого обсуждаются свойства и характеристики, выявленные при анализе, и приводятся примеры, поясняющие более тонкие детали. Для иллюстрации идей в книге широко используются рисунки. В начальные курсы математический анализ можно не включать; это даст возможность начинающим уловить основные идеи. Включение математического анализа для более подготовленных студентов позволит обеспечить более глубокое понимание материала и приобрести способность выполнять исследования в этой области.

В гл. 1 дается общее введение в рассматриваемую область, а также излагается концепция уровневой сети. Уровневая структура позволяет рассматривать различные вопросы сетей передачи данных довольно независимо друг от друга, что делает возможным чтение каждой из последующих глав с любой желаемой степенью проникновения (в том числе выборочное чтение); при этом почти не уменьшается способность понимания материала других глав.

В гл. 2 рассматриваются два нижних уровня уровневой структуры. Первый снизу, или физический уровень, связан с передачей битов по физической среде

передачи. Мы даем краткое введение, которое будет полезным, но не обязательным для понимания остальной части книги. Следующий уровень — уровень управления линией передачи данных; он связан с надежной передачей пакетов по линии связи. Разд. 2.4, в котором рассматриваются стратегии повторения передачи, должен, вероятно, входить в любой курс, поскольку в нем выявляются тонкости распределенных алгоритмов (протоколов) в наиболее простом случае.

В гл. 3 развивается теория массового обслуживания, используемая для анализа методов множественного доступа (гл. 4) и в меньшей степени алгоритмов маршрутизации (гл. 5). В курсы для слушателей, мало интересующихся математическим анализом, не войдет, вероятно, большая часть этой главы; однако рекомендуется включить в них результаты. Тем не менее теорему Литтла и определение пуассоновского процесса опускать не следует, поскольку они значительно способствуют пониманию последующих глав. Эта глава содержит множество результатов, которые выводятся в ней намного проще, чем в литературе по теории массового обслуживания, за счет рассмотрения только стационарного режима и иногда за счет ослабления строгости ради ясности и понимания. Читатели с хорошей математической подготовкой смогут сами восстановить отсутствующие детали, обеспечивающие полную строгость; для большинства же читателей эти детали могут затушевывать ход рассуждений.

В гл. 4 рассматривается связь в сетях с множественным доступом; ими могут быть локальные сети, спутниковые сети и радиосети. В курсы с меньшей теоретической направленностью, по-видимому, не войдет вторая половина разд. 4.2, весь разд. 4.3 и большая часть разд. 4.4, что обеспечит быстрый переход к локальным и спутниковым сетям разд. 4.5. В концептуальном плане эта глава позволит более глубоко понять природу распределенных алгоритмов.

В гл. 5 рассматриваются методы маршрутизации. Материал дается в порядке возрастания сложности и глубины, так что читатели могут продвигаться до тех пор, пока это будет удаваться без особого труда. Наряду с маршрутизацией, которая рассматривается здесь более основательно по сравнению с имеющейся литературой, в этой главе также более глубоко описываются распределенные алгоритмы. Кроме того, уделяется внимание топологическому проектированию и восстановлению сети после отказов из-за выхода из строя линии связи.

Глава 6 посвящена управлению потоками (или контролю за перегрузками, как это иногда называют). Первые три раздела имеют в основном описательный характер; в них описываются, во-первых, цели и трудности управления потоками, во-вторых, некоторые общие подходы и, наконец, способы осуществления управления потоками в нескольких существующих сетях. Последний раздел содержит более развитый аналитический материал о последних работах в этой области.

В книге совершенно не рассматриваются протоколы высоких уровней, а именно различные процессы, которые происходят в вычислительных машинах и устройствах, использующих сеть для обмена информацией, при условии, что более нижние уровни обеспечивают возможность надежной передачи. Эта тема отлична по своей природе от тех, которые затрагиваются; при основательном рассмотрении она увеличила бы объем книги вдвое.

Мы заранее извиняемся за большое число аббревиатур и жаргонных выражений. Часто используемые аббревиатуры были включены для того, чтобы читатели смогли общаться со специалистами в этой области, а также для справочных целей, т. е. для разъяснения смысла этих аббревиатур.

В конце каждой главы, за исключением первой, дается большое число задач, как простых, позволяющих лучше понять основные концепции и методы, так и более сложных, решения которых дополняют результаты, полученные в главе. Решения задач содержатся в руководстве, которое можно получить в издательстве Prentice-Hall.

Каждая глава имеет также краткий перечень источников и рекомендации по дальнейшему изучению литературы. Мы приносим извинения многим авторам, чей вклад не был упомянут. Литература по сетям передачи данных обширна, и мы

ограничились ссылками, которые, на наш взгляд, наиболее полезны и содержат материал, дополняющий текст.

Стимулирующие преподавательская работа и атмосфера исследований, характерные для МТИ, идеально способствовали написанию этой книги. В частности, мы благодарны многим студентам, которые использовали этот материал. Их замечания в значительной мере помогли добиться ясности в изложении. В равной степени мы благодарны многим коллегам и аспирантам за детальную критику различных глав. Особую признательность выражаем нашему коллеге Пьеру Умбле, его советы, знания и глубокое понимание неоценимы. Кроме того, полезную помощь оказали Ердал Арикан, Давид Кастанон, Роберт Купер, Тони Эфремидес, Ели Гафни, Мариан Гарднер, Пол Грин, Елен Хан, Брус Гаек, Роберт Кеннеди, Джон Спинелли и Джон Цициклис. Мы также признательны Ненси Янг за перепечатку многих исправлений и переработок и Эми Хендриксон за компьютерный набор книги с помощью системы TEX. Наши редакторы из Prentice-Hall также сделали очень полезный вклад в подготовку окончательного текста при весьма напряженном графике работ. В заключение мы хотим выразить благодарность за финансовую поддержку в проведении исследования Управлению перспективного планирования научно-исследовательских работ министерства обороны, предоставившего субсидию ONR-N00014-84-K-0357, Национальному научному фонду за субсидии ECS-8310698 и EC-8217668 и управлению ARO за субсидию DAAC-29-84-K-000.

*Димитри Бертсекас
Роберт Галлагер*

1. Введение

Многоуровневая архитектура сети

1.1. Исторический обзор

Примитивные формы сетей передачи данных возникли в далеком прошлом; к ним относятся дымовые сигналы, которыми пользовались первобытные общества, и, несомненно, телеграфия XIX века. В этих системах сообщения сначала вручную кодировались в последовательности, как правило, двоичных символов и затем вручную передавались и принимались. Там, где это было необходимо, сообщения вручную ретранслировались далее на промежуточных пунктах.

Значительным прогрессом явилось применение в начале 50-х годов линий связи, которые соединили центральные ЭВМ с удаленными терминалами и другими периферийными устройствами, такими как принтеры и пункты дистанционного ввода заданий (ДВЗ) (рис. 1.1). Число таких периферийных устройств быстро увеличивалось в 60-е годы, когда появились вычислительные системы с разделением времени и возросла производительность центральных ЭВМ. В связи с этим стало невыгодным выделять отдельную протяженную линию связи для каждого периферийного устройства. Были разработаны удаленные мультиплексоры или концентраторы, которые собирали весь трафик от множества близлежащих периферийных устройств и передавали его по одной линии центральному процессору. Наконец, чтобы освободить центральный процессор от управления связью, были созданы специальные процессоры, называемые *фронтальными*, для управления связью с периферийными устройствами. Это привело к более сложной структуре, показанной на рис. 1.2. В таких системах связь автоматизирована, в противоположность, например, телеграфии, но управление связью централизовано и осуществляется фронтальным процессором. Хотя полностью обосновано и широко принято называть такую систему сетью передачи данных, однако проще считать ее вычислительной машиной с удаленными периферийными устройствами. Многие интересные проблемы, связанные с сетями передачи данных, такие как распределенное управление системой, транзитная передача сообщений по множеству линий связи, совместное использование линий связи многими пользователями и процессами, в таких централизованных системах не возникают.

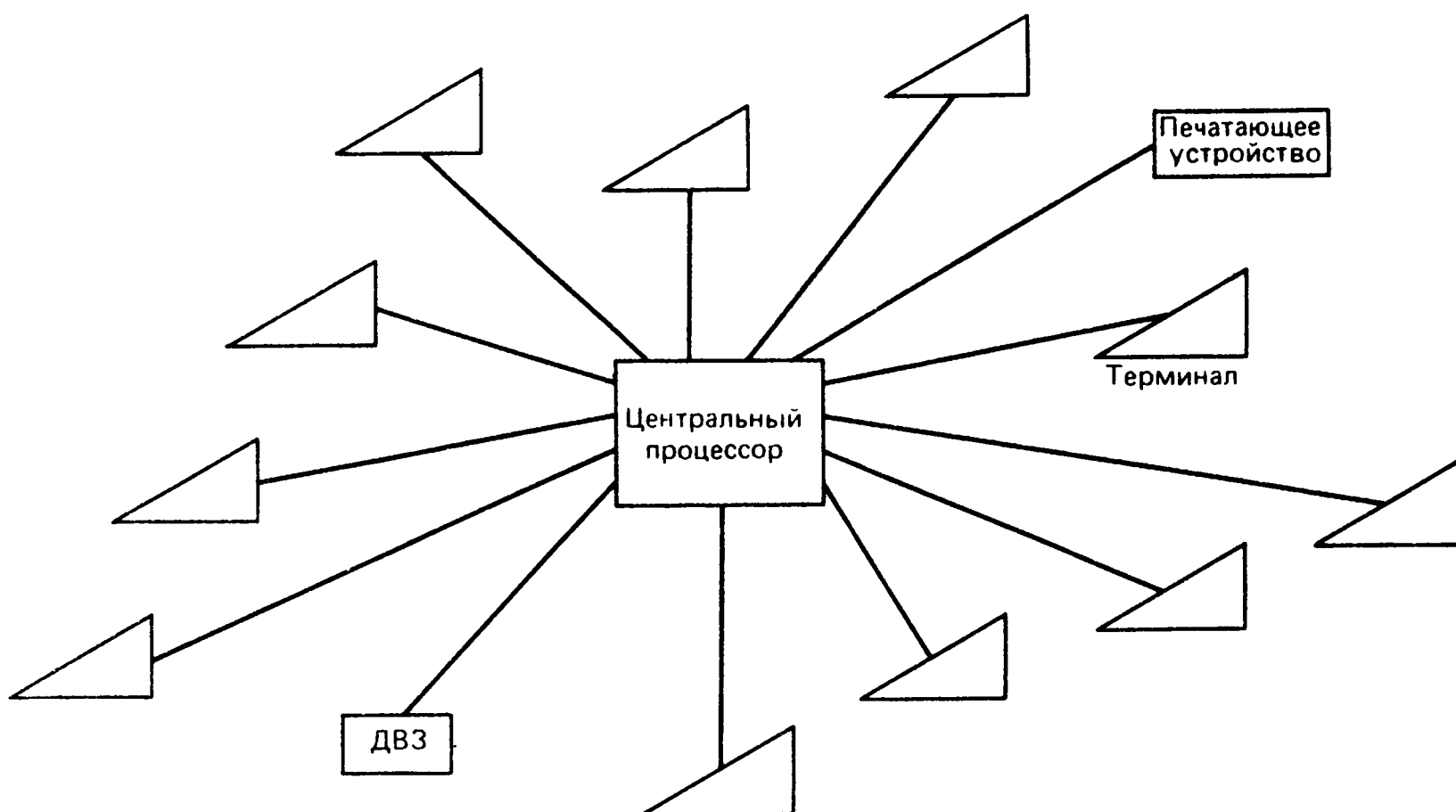


Рис. 1.1. Сеть с одним центральным процессором и отдельной линией связи к каждому устройству.

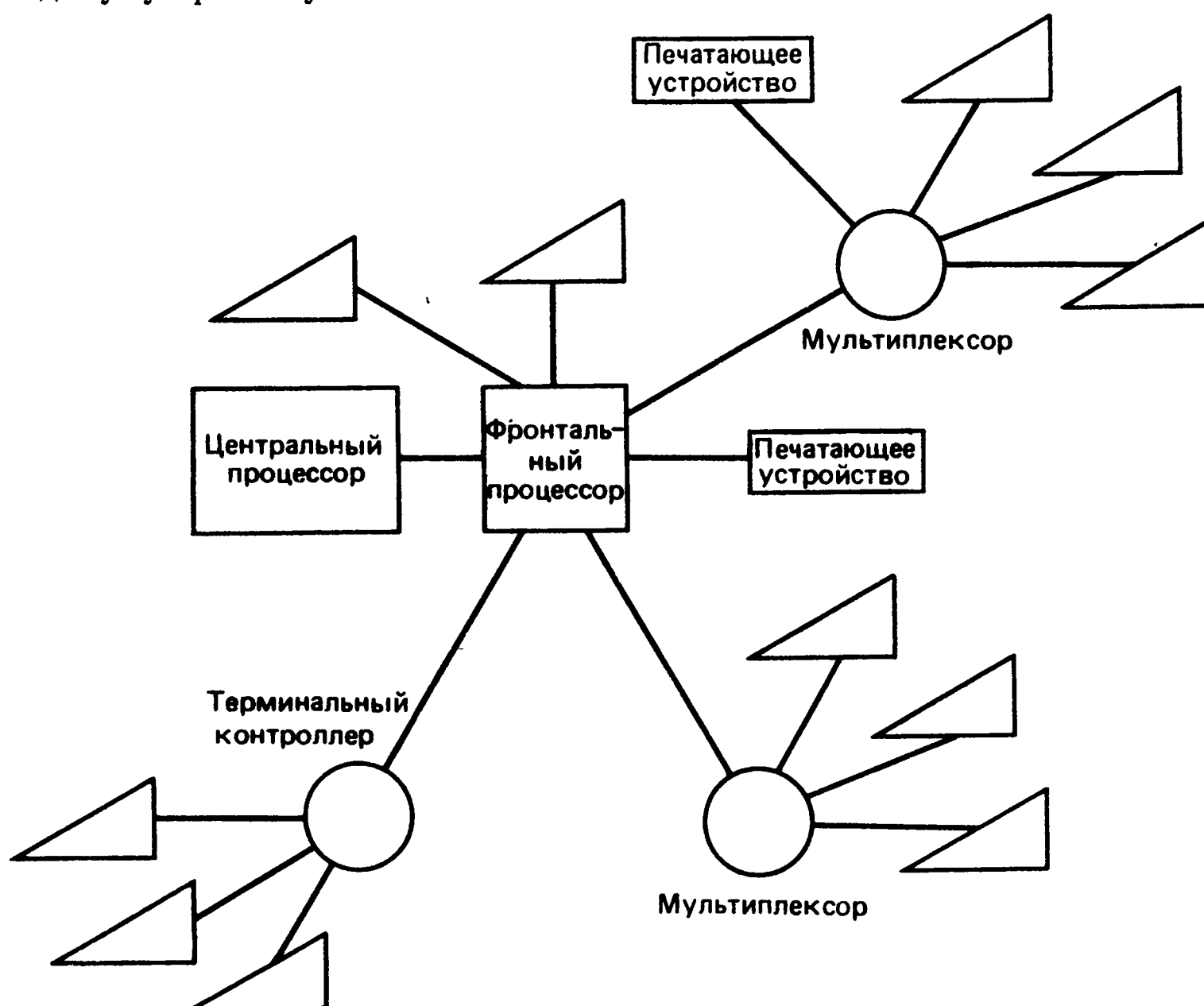


Рис. 1.2. Сеть с одним центральным процессором, но с линиями связи, которые совместно используются устройствами.

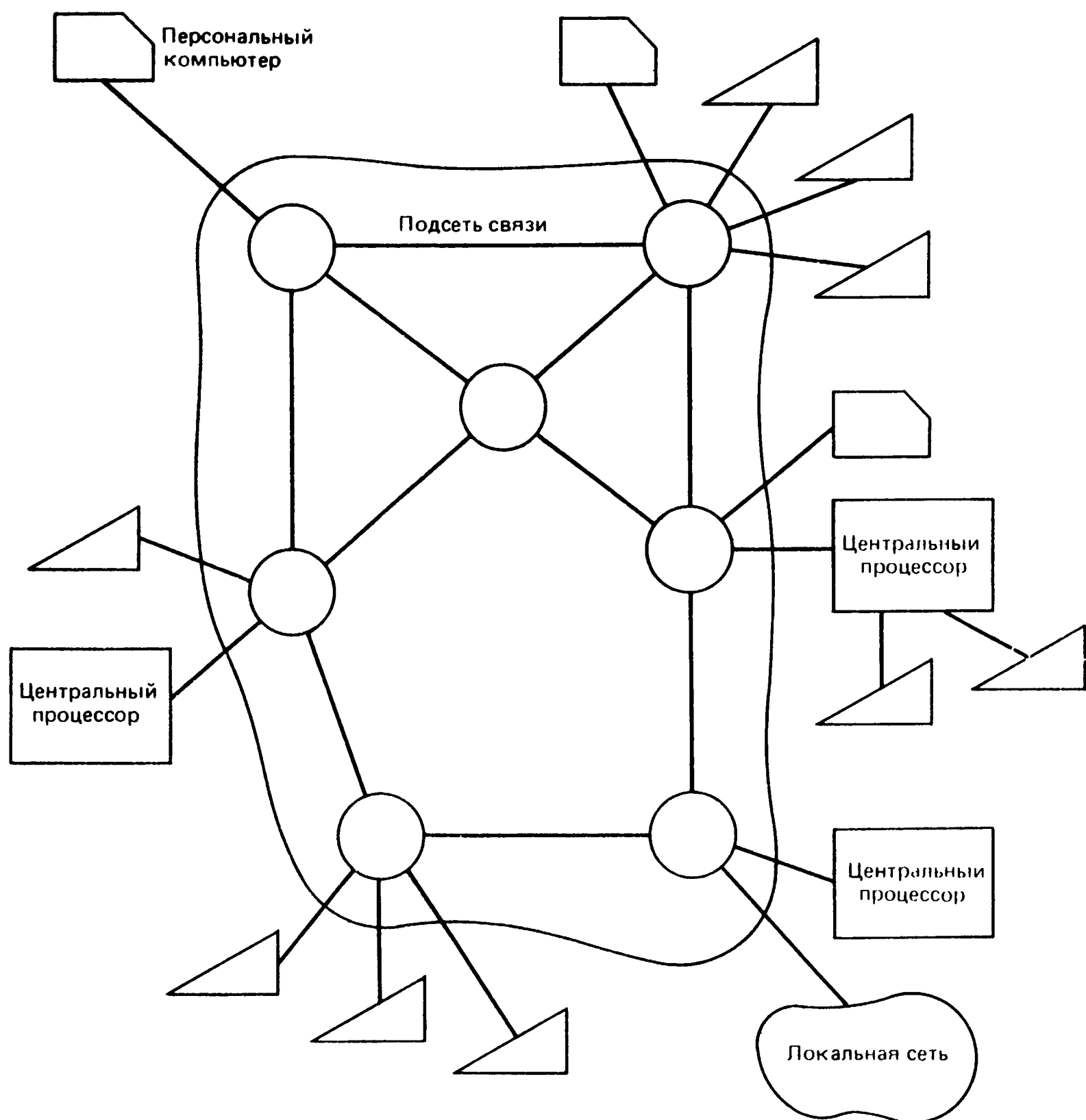


Рис. 1.3. Общая схема сети, содержащей подсеть из линий связи и узлов. Внешние устройства соединены с подсетью посредством линий, идущих к узлам подсети.

Сети ARPANET и TYMNET, появившиеся примерно в 1970 г., были первыми широкомасштабными сетями передачи данных общего назначения, которые соединили географически распределенные вычислительные системы, периферийные устройства и пользователей. На рис. 1.3 показана схема таких сетей. В подсети связи находится множество узлов, различные пары которых соединены линиями связи. Вне подсети находятся различные ЭВМ, базы данных, терминалы и т. д., которые соединяются с помощью этой подсети. Сообщения, возникающие в этих внешних устройствах, входят в подсеть связи, проходят от узла к узлу по линиям связи и, наконец, выходят к внешнему получателю. Основная задача узлов подсети, которые по существу являются ЭВМ, заключается в том, чтобы направлять сообщения по маршрутам,

проходящим через подсеть. Эти узлы иногда называются процессорами связи (ПС), а иногда — коммутаторами. В отдельных сетях (например, DECNET) некоторые узлы подсети могут быть включены в ЭВМ, использующие сеть. Полезно, однако, считать, что узлы подсети логически отделены от ЭВМ и ЭВМ находятся вне подсети.

Следует отметить, что на рис. 1.1 и 1.2 центром сети является вычислительная система, а на рис. 1.3 — подсеть (т. е. коммуникационная часть сети). Если запомнить эту схему, в которой внешние устройства окружают подсеть связи, то это облегчит в дальнейшем понимание уровневой архитектуры сети, рассматриваемой далее в этой главе, и распределенного управления сетью, рассматриваемого на всем протяжении книги.

После 1970 г. произошло стремительное увеличение числа сетей передачи данных. Эти сети, так же как и родоначальные ARPANET и Tymnet, описываются позже, а сейчас мы будем использовать общую модель сетей передачи данных, показанную на рис. 1.3.

1.1.1. Технические и экономические предпосылки

Прежде чем перейти к рассмотрению сетей передачи данных, дадим краткий обзор, во-первых, технических и экономических факторов, которые привели к появлению сетей, и, во-вторых, различных приложений, для которых требуются сети. Основной движущей силой быстрого развития вычислительной техники, техники связи и сетей передачи данных стала интегральная техника, в частности разработка микросхем с очень высокой степенью интеграции (СБИС).

В вычислительной технике это привело к появлению более быстродействующих процессоров меньшей стоимости, более дешевых оперативных и внешних запоминающих устройств с большими быстродействием и емкостью. В результате стоимость вычислений каждые два года понижалась примерно в два раза.

Вместе с тем с появлением все более и более мощных микропроцессоров небольшие персональные компьютеры и компьютеризированные рабочие станции стали более эффективными по сравнению с большими вычислительными средствами, работающими в режиме разделения времени. Таким образом, с одной стороны, возрастающее число приложений может быть компьютеризовано с незначительными затратами, а с другой — наблюдается быстрое увеличение числа вычислительных систем, а не производительности малого числа сверхбольших вычислительных систем.

Предыдущее обсуждение затрат на компьютеризацию не учитывает стоимости программного обеспечения. Создание про-

граммного обеспечения значительно усовершенствовалось, но зато увеличилась оплата труда хороших инженеров-программистов. Однако, если программное обеспечение может копироваться, цена копий уменьшается обратно пропорционально их числу. Таким образом, хотя затраты на программное обеспечение составляют основную часть при создании новой вычислительной системы, с увеличением рынка уменьшается цена одного экземпляра. Каждое достижение в интегральной технике уменьшает стоимость и улучшает характеристики вычислительных систем; это ведет к увеличению рынка, что порождает падение цен на программное обеспечение и в свою очередь приводит к дальнейшему увеличению рынка. Каждое новое применение требует, однако, нового специального программного обеспечения, которое первоначально является дорогостоящим (до тех пор пока не сформируется рынок) и требует повышения квалификации пользователя. Таким образом, точное прогнозирование увеличения рынка вычислительной техники, так же как и рынка сетей передачи данных, представляет довольно трудную задачу.

1.1.2. Техника связи

Стоимость передачи данных по линии связи от одного пункта к другому падала, но значительно медленнее, чем стоимость вычислений. Стоимость линии передачи данных увеличивается с ростом скорости передачи данных, но медленнее, чем по линейному закону. Таким образом, стоимость передачи одного двоичного символа уменьшается при увеличении скорости передачи данных по линии связи. Часто бывает экономически выгоднее использовать одну высокоскоростную линию связи для многих пользователей, чем отдельную линию связи для каждого пользователя (этот эффект можно заметить, рассматривая рис. 1.2, где используются мультиплексоры и концентраторы, позволяющие разделить затраты на передачу).

Одно из следствий совместного использования высокоскоростных линий связи состоит в том, что стоимость передачи данных от одного пункта к другому увеличивается медленнее, чем линейно, при увеличении расстояния между пунктами. Это объясняется тем, что путь, по которому проходят данные, может состоять из короткой линии, ведущей к совместно используемой протяженной высокоскоростной линии, и еще одной короткой линии, ведущей к адресату.

Оценка стоимости средств связи чрезвычайно специфична и сложна. Стоимость линии связи зависит от того, находятся ли средства в собственном пользовании или арендуются; при аренде стоимость зависит от текущей конкурентной и биржевой ситуации. Далее подробности, связанные со стоимостью передачи, не

будут рассматриваться; здесь укажем два общих факта, касающихся стоимостей, которые важны.

Во-первых, для глобальных сетей (т. е. сетей, покрывающих площадь, которая больше площади города) стоимость сети в настоящее время определяется в основном затратами на передачу. Следовательно, желательно использовать линии связи эффективно, возможно, при дополнительных компьютеризационных затратах. Как будет показано в разд. 1.3, пульсирующий характер типичной передачи данных и высокая стоимость незагруженных линий связи привели к созданию пакетных сетей передачи данных, которые позволяют увеличить использование линий.

Во-вторых, для локальных сетей стоимость сети не определяется затратами на передачу. На малой площади относительно высокоскоростная связь за умеренную цену может быть достигнута при использовании коаксиального кабеля или даже витой пары проводов. Желание использовать такую среду и избежать относительно дорогих устройств коммутации привело к появлению локальных сетей, в которых высокоскоростная среда используется на основе множественного доступа. Тип структуры таких сетей рассматривается в гл. 4.

Существуют две важные незавершенные разработки, которые могли бы радикально снизить важность эффективного использования средств связи в глобальных сетях передачи данных. Первая из них — это применение высокоскоростных оптических волокон. Трудно обеспечить эффективность использования средств связи, которые могут передавать данные со скоростью гигабит в секунду. В этом случае более важными становятся вопросы доступа, обработки и коммутации на концах линий.

Второй является создание стандартов для цифровых сетей интегрального обслуживания (ЦСИО). Существующие телефонные сети, предназначенные для передачи речи, будут в конце концов заменены полностью цифровыми сетями, обеспечивающими как телефонное обслуживание, так и высокоскоростную цифровую передачу данных для тех, кто сейчас имеет телефоны. Когда будут работать такие сети и какова будет их стоимость, сейчас неизвестно, но они могли бы значительно уменьшить стоимость передачи данных, сделав менее важным эффективное использование средств связи.

1.1.3. Применения сетей передачи данных

С распространением ЭВМ, о чем говорилось выше, нетрудно предсказать рост потребности в передаче данных. Краткое описание нескольких приложений, которые нуждаются в системах связи, поможет понять основные проблемы, которые связаны с сетями связи.

Начнем с того, что существует много приложений, требующих удаленного доступа к базам данных. Простыми примерами являются информационные и финансовые службы, доступные пользователям персональных ЭВМ. Более сложными примерами, требующими неоднократного взаимодействия удаленного пункта с базой данных и связанными с ней программами, являются дистанционная медицинская диагностика с помощью ЭВМ и дистанционное компьютерное обучение. В некоторых из этих приложений нужно учитывать стоимость введения базы данных везде, где она могла бы потребоваться, и стоимость необходимой передачи при удаленном доступе к ней. В других примерах, в которых база данных быстро меняется, нет альтернативы, которая исключала бы связь удаленных пунктов с центральной базой данных.

Далее, существует много приложений, требующих дистанционного обновления баз данных, которое может сочетаться с доступом к данным. Система резервирования авиабилетов, аппараты автоматического подсчета голосов, системы управления инвентаризацией, автоматические системы упорядочения входа и работа группы географически распределенных авторов над общим текстом являются такими примерами. Вообще в приложениях подобного типа имеются множество географически распределенных пунктов, в которых данные входят в систему, и, как правило, множество географически распределенных пунктов, в которых требуются выходные данные. Как в случае, когда входные данные обрабатываются и запоминаются в одном пункте (как на рис. 1.1 и 1.2), так и в случае, когда они обрабатываются и запоминаются во многих пунктах (как на рис. 1.3), необходимо, чтобы сеть собирала и распределяла выходные данные. Необходимо также соблюдать значительные меры предосторожности, чтобы гарантировать отсутствие искажения выходных данных из-за задержек в системе.

Еще одним широко известным приложением является электронная почта для людей, пользующихся сетью. Такую почту можно читать, заносить в файл, направлять другим пользователям, дополняя, быть может, комментариями, или читать, находясь в различных пунктах сети. Очевидно, что такая служба имеет много преимуществ по сравнению с традиционной почтой с точки зрения скорости доставки и гибкости. Кроме того, оно по сравнению с речевой телефонной службой обеспечивает возможность получения записи разговора, уменьшения стоимости (для дальних вызовов) и устранение необходимости того, чтобы оба пользователя связывались в одно и то же время. Речевая служба с устройствами-автоответчиками может в принципе обладать гибкостью электронной почты, но она много дороже для дальних вызовов. К тому же многие люди не любят записывать речевые сообщения на автоответчиках.

Наконец, имеется приложение, когда требуется использовать удаленную вычислительную систему для решения некоторой вычислительной задачи. Это может быть при необходимости распределения нагрузки, когда локальная ЭВМ перегружена, при отсутствии или неисправности локальной ЭВМ или когда удаленная ЭВМ более приспособлена для данной задачи. Существуют также вычислительные задачи реального времени, в которых задержка ответа вычислительной системы не должна превышать некоторого максимального значения. Если такая задача не под силу одной ЭВМ, то она может быть распределена между удаленными вычислительными системами.

Следует заметить, что во всех вышеупомянутых приложениях можно было бы использовать сеть с централизованными вычислительными средствами, такую, как показана на рис. 1.1 или 1.2. Чтобы увидеть это, просто мысленно представьте, что все ЭВМ в сети, такой, как на рис. 1.3, стянуты в одну центральную область. Вычислительные машины были бы тогда соединены не длинными, а короткими линиями связи, а вся сеть, за исключением некоторого изменения задержек передачи, оставалась бы неизменной. Такая структура допускала бы как совместное использование памяти вычислительными машинами, так и централизованное техническое обслуживание. Почему тогда так быстро увеличилось значение сетей передачи данных с географически распределенными вычислительными средствами? Одной из основных причин является стоимость передачи. Когда ЭВМ распределены, многие вычислительные задачи могут быть выполнены на месте. Даже для указанных выше приложений затраты на передачу могут быть значительно снижены посредством локальной обработки. Другая причина состоит в том, что организации часто приобретают ЭВМ для задач локальной автоматизации и только после того, как эта автоматизация осуществлена, возникает потребность в дистанционных взаимодействиях. Наконец, организации часто желают сохранять контроль над своими собственными вычислительными системами, а не быть чрезмерно зависимыми от политики цен, изменений в программном обеспечении и потенциальных нарушений сохранности вычислительных средств, совместно используемых многими организациями.

Другим часто упоминаемым преимуществом сети с распределенными вычислительными средствами является более высокая надежность. Для централизованной системы на рис. 1.2 выход из строя линии связи может привести к тому, что ряд пунктов полностью лишится доступа к вычислительным средствам. Для сети на рис. 1.3 выход из строя линии связи является менее разрушительным, особенно если существует несколько различных путей между каждой парой узлов, и вопрос о надежности становится более сложным. Если все ЭВМ сети сосредоточены в одном

центре, то сеть может выйти из строя при аварии в этом центре. Однако вычислительный центр может быть более тщательно защищен и восстановления в нем могут быстрее и легче выполняться, чем при распределенных вычислительных средствах. Кроме этих факторов, по-видимому, не существует других причин, из-за которых географически распределенные вычислительные средства более или менее надежны по сравнению с географически централизованными (но логически распределенными) средствами. Во всяком случае, основное внимание далее будет уделяться таким сетям, как на рис. 1.3, у которых подсеть связи следует рассматривать как центр всей сети.

1.2. Сообщения и коммутация

1.2.1. Сообщения и пакеты

Под сообщением в сети передачи данных подразумевается примерно то же, что и при обычном использовании этого слова. Например, в системе резервирования авиабилетов мы можем рассматривать запрос на резервирование, включающий дату, номер рейса, фамилии пассажиров и т. д., как сообщение. В системе электронной почты сообщением может считаться один документ, пересылаемый от одного пользователя к другому. Если этот же документ рассылается затем нескольким другим пользователям, то в зависимости от контекста иногда можно рассматривать это как рассылку нескольких новых сообщений, а иногда — как рассылку одного и того же сообщения. В системе передачи файлов сообщением, как правило, является файл. В системе передачи изображений (т. е. картин, рисунков, диаграмм и т. д.) сообщением считается изображение. В приложении, требующем интерактивной связи между двумя или более пользователями, сообщением может быть единичный акт связи между двумя пользователями. При интерактивном акте пользователь 1 может послать сообщение пользователю 2, пользователь 2 может ответить сообщением первому, который может затем послать другое сообщение второму и так далее до тех пор, пока разговор не завершится полностью.

Важной особенностью сообщения является то, что с точки зрения пользователей сети оно является единым целым при передаче. Если адресат получает только часть сообщения, то она является бесполезной.

Иногда необходимо делать различие между сообщением и представлением сообщения. Как в подсети связи, так и в ЭВМ сообщение обычно представляется последовательностью двоичных символов 0 и 1. Для краткости двоичный символ будет называться *битом*. Когда сообщение движется от отправителя к полу-

чителю, над последовательностью битов, использованных для представления сообщения, часто производится ряд преобразований. Такие преобразования иногда желательны для сжатия данных, а иногда — для облегчения передачи сообщения по сети. Ниже дается краткое описание этих вопросов.

Цель сжатия данных заключается в том, чтобы уменьшить длину последовательности битов, представляющей сообщение. С точки зрения теории информации сообщение рассматривается как элемент множества возможных сообщений, на котором задано вероятностное распределение, показывающее частоту появления отдельных сообщений. Для вероятностей сообщений можно получить только грубые априорные или адаптивные оценки. Идея состоит в том, чтобы приписать более короткие последовательности битов более вероятным сообщениям, а более длинные последовательности битов — менее вероятным сообщениям, уменьшая таким образом среднюю длину представления. Например, чтобы получить битовое представление текста, можно приписать часто используемым буквам алфавита (или часто используемым словам в словаре) последовательность из малого числа битов, а редко используемым буквам или словам — из большого числа битов. Другим примером является система резервирования авиабилетов, в которой часто используемые сообщения имеют жестко заданный формат (дата, номер рейса, фамилии и т. д.) и поэтому могут быть представлены очень компактно; при этом более длинные последовательности битов приписываются редким ситуациям. Сжатие данных более детально описано в гл. 2 при обсуждении сжатия служебной информации. Этот вопрос не будет рассматриваться в общем виде, так как он не относится к области сетей передачи данных. Сжатие данных применяется как при хранении данных в памяти, так и при передаче данных от одного пункта к другому.

Вместе с тем преобразование представлений сообщений с целью облегчения их передачи является основным вопросом сетей передачи данных. В последующих главах приводится ряд примеров, в которых различные виды служебной информации должны быть добавлены к сообщениям для того, чтобы обеспечить надежную связь, доставить сообщение требуемому адресату, управлять перегрузкой и т. д. Будет также показано, что передача по подсети связи длинных сообщений без их разбивки является нецелесообразной, в частности из-за задержки, необходимости больших буферов и введения контроля за перегрузками. Поэтому сообщения, представляемые длинными последовательностями битов, обычно разбиваются на более короткие последовательности битов, которые называются *пакетами*. Эти пакеты можно затем передать по подсети как независимые объекты и собрать из них сообщения в конечном пункте.

Назначение подсети связи при этом состоит в том, чтобы принять пакеты из пунктов, находящихся вне подсети, затем передать эти пакеты по некоторому пути, образованному из линий связи и других узлов, и, наконец, доставить их в пункты назначения. Подсеть должна каким-либо образом получать информацию о том, куда пакет движется, что же касается смысла соответствующего ему сообщения, то оно не имеет никакого значения для подсети. По отношению к подсети пакет является просто последовательностью битов, которая должна быть надежно и быстро передана по подсети связи.

1.2.2. Сеансы

Когда два пользователя сети передачи данных желают обменяться сообщениями, они сначала устанавливают сеанс аналогично тому, как это делается в телефонных сетях. Вопрос о том, почему необходима специальная подготовка перед началом обмена сообщениями между двумя пользователями, обсуждается в дальнейшем. При сеансе, например, между пользователями 1 и 2 некоторая последовательность сообщений передается от пользователя 1 к пользователю 2, а некоторая последовательность сообщений идет от пользователя 2 к пользователю 1. С точки зрения пользователей эти сообщения обычно обусловлены конкретными событиями. Однако с точки зрения подсети связи эти сообщения возникают в произвольные и непредсказуемые моменты.

Для анализа подсети разумно использовать случайный процесс в качестве модели последовательности моментов поступления сообщений или пакетов в заданном сеансе. Для упрощения будет рассматриваться модель, в которой поступления сообщений происходят в случайные моменты времени независимо друг от друга и от моментов поступления в других сеансах. Этот тип процесса поступления сообщений называется *пуассоновским* и будет определен и рассмотрен в разд. 3.3. Подобная модель является вполне реальной для многих типов сеансов и не учитывает взаимодействия между сообщениями, движущимися в двух направлениях во время сеанса. Однако такие простые модели позволяют понять основные обменные соотношения, которые приходится учитывать при проектировании сетей; в более реальных и сложных моделях эти обменные соотношения часто не удается выявить.

Чтобы перевести вопрос о выборе модели процесса поступления сообщений в сеансах в более практическую плоскость, заметим, что сети, особенно такие, как на рис. 1.3, обычно предназначены для многих применений. Так как проектирование и построение подсети связи являются длительным процессом и при-

менения быстро изменяются, а число их увеличивается, то подсеть следует проектировать так, чтобы удовлетворить требованиям разнообразных применений. Любая сложная модель поступления сообщений в сеансах станет, вероятно, неверной по мере использования сети. Эта точка зрения, состоящая в том, что подсеть необходимо проектировать так, чтобы ее работа не зависела от тонких деталей применений, обсуждается далее в разд. 1.3 при рассмотрении уровней сети.

Мы имеем следующую модель функционирования подсети. Подсеть должна обеспечивать связь для медленно меняющегося множества сеансов; в каждом сеансе сообщения, случайная длина которых имеет некоторое распределение, поступают в случайные моменты в соответствии с некоторым случайным процессом. Поскольку в основном не будет учитываться взаимодействие между двумя встречными потоками сообщений в сеансе, то мы будем моделировать сеанс с двумя направлениями посредством двух однонаправленных сеансов, один из которых соответствует потоку сообщений в одном направлении, а другой — в противоположном направлении. В дальнейшем мы используем слово *сеанс* для таких однонаправленных сеансов. В таких же вопросах, как инициирование сеанса и квитирование от конца до конца, делается различие между сеансами с двумя и с одним направлением.

Хотя детальные характеристики различных видов применений рассматриваться не будут, однако некоторые характеристики сеансов нужно иметь в виду. Ниже перечислены наиболее важные из них.

1. *Скорость поступления сообщений и интервалы между моментами поступления.* Типичные скорости поступления сообщений для сеансов находятся в пределах от сотни поступлений сообщений в секунду до одного поступления сообщения в течение многих минут. Простыми распределениями промежутка времени между соседними поступлениями сообщений являются пуассоновское, детерминированное (т. е. фиксированный временной интервал между двумя соседними сообщениями) и равномерное (т. е. длина интервала времени между последовательными сообщениями равномерно распределена между некоторым минимумом и максимумом).
2. *Длительность сеанса.* Иногда (как в случае электронной почты) сеанс иницируется ради одного сообщения. Другие сеансы длятся рабочий день или даже постоянно.
3. *Средняя длина сообщения и распределение длины.* Типичная длина сообщения принадлежит интервалу примерно от нескольких битов до 10^8 бит, причем на верхней границе интервала находятся применения, связанные с передачей фай-

лов, а на нижней — интерактивные сеансы от терминала к ЭВМ. Простыми примерами распределения длины являются экспоненциально убывающая плотность распределения вероятности, равномерное распределение между некоторым минимумом и максимумом и фиксированная длина.

4. *Допустимая задержка.* Допустимая средняя задержка лежит в диапазоне от 10 мс для некоторых приложений, требующих управления в режиме реального времени, до 1 с или менее для приложений, требующих взаимодействия терминала с ЭВМ, или до нескольких минут или более для некоторых приложений, связанных с передачей файлов. В ряде приложений существует максимально допустимая задержка (в отличие от средней задержки). Например, при передаче пакетированной речи источник кодирует сегменты речи фиксированной длины в пакеты. Получатель должен провести обратное преобразование пакетов в речевые сегменты, не допуская превышения некоторой фиксированной полной задержки; любой пакет, не пришедший в течение этого времени, просто игнорируется.
5. *Надежность.* В некоторых приложениях требуется, чтобы все доставленные сообщения не содержали ошибок. Например, в банковских приложениях, при передаче программ для ЭВМ или файлов ошибка в единственном бите сообщения может иметь серьезные последствия. В других приложениях, таких как электронная почта, все сообщения должны быть доставлены, но случайный ошибочный бит в сообщении обычно может быть исправлен читателем визуально. Наконец, имеются приложения, в которых как случайная ошибка в одном бите, так и случайная потеря целых пакетов или сообщений являются допустимыми. Например, в распределенных измерительных системах передаваемые сообщения уже содержат шум, а случайная потеря сообщений вскоре восполняется более новыми сообщениями. При передаче пакетированной речи случайная потеря (или поздняя доставка) пакета или случайное искажение бита просто увеличивает зашумленность принятого речевого сигнала. Следует отметить, однако, что применение методов сжатия данных при передаче пакетированной речи, а также в других приложениях сильно повышает необходимость безошибочной передачи.
6. *Порядок сообщений и пакетов.* Пакеты, составляющие сообщения, или должны поддерживаться в правильном порядке при следовании по сети, или правильный порядок должен восстанавливаться в некотором пункте. Во многих применениях (таких как обновление баз данных) сообщения также должны доставляться в правильном порядке,

тогда как в других приложениях порядок сообщений не имеет значения. Вопрос о том, где рассматривать надежность и порядок сообщений (т. е. во внешних пунктах или внутри подсети связи, или же в обоих случаях), является важным вопросом проектирования. Это будет обсуждаться в разд. 2.8.

В свете этих характеристик полезно обратить внимание на три вида приложений, которые занимают в некоторой степени крайние позиции и предъявляют разные требования к подсети связи. Одним из них являются сеансы между интерактивным терминалом и ЭВМ; сообщения в этом случае короткие, поток сообщений слабый, требования к задержке умеренно строгие, а потребность в надежности высокая. Другим приложением являются сеансы с передачей файлов; сообщения в этом случае очень длинные, скорость поступления сообщений обыкновенно низкая, требования к задержке очень слабые, а потребность в надежности очень высокая. Третьим приложением является передача пакетированной речи. В этом случае понятие сообщения не очень полезно, кроме того, пакеты короткие, скорость поступления пакетов высокая, требование ограниченности задержки строгое и потребность в надежности достаточно низкая. Сеть, которая может работать во всех этих случаях одновременно, вероятно, без большого труда может быть использована в любом другом интересном приложении.

1.2.3. Коммутация каналов и передача с промежуточным накоплением

Существуют два общих подхода, известных как коммутация каналов и передача с промежуточным накоплением, которые могут использоваться в подсетях связи для передачи трафика различных сеансов. Ниже дается краткий обзор коммутации каналов и рассматриваются причины, из-за которых этот подход приводит к неэффективному использованию каналов связи для многих типов сеансов. Затем описывается передача с промежуточным накоплением, причем показывается, как при этом подходе преодолевается упомянутая неэффективность.

При коммутации каналов для каждого сеанса задается определенная скорость передачи r_s , выраженная в битах в секунду (она может быть различна для двух направлений двустороннего сеанса, но здесь рассматриваются только односторонние сеансы). Затем по подсети прокладывается путь от передающего пункта до пункта назначения. Каждая линия связи, принадлежащая этому пути, выделяет затем часть r_s ее полной пропускной способности в заданном направлении для этого сеанса. Распределение пропускной способности линии связи между различными сеансами обычно

выполняется посредством временного уплотнения (ВУ) или частотного уплотнения (ЧУ); более подробно эти методы будут описаны позже. Важно, что сумма скоростей передач всех сеансов, использующих линию, не может превышать полную скорость передачи этой линии. Таким образом, если линия связи полностью распределена между существующими сеансами, новый сеанс не может использовать эту линию. Когда нельзя найти никакого пути, проходящего по линиям, имеющим неиспользованный резерв скорости, по крайней мере равный r_s бит в секунду, новый сеанс отвергается (т. е. дается сигнал занятости). Вместе с тем если сеанс успешно инициирован, то он имеет гарантированную скорость передачи r_s по сети. Узлы при этом просто принимают входящий поток битов данного сеанса от входящей линии и направляют его в отведенную ему часть выходящей линии. Это не так просто, как кажется, но соответствующая техника давно известна и хорошо разработана.

Коммутация каналов почти всегда используется в телефонных сетях, но она редко применяется в сетях передачи данных. Поскольку техника коммутации каналов является простой и распространенной, возникает вопрос, почему она не подходит для сетей передачи данных. Пусть λ обозначает скорость поступления сообщений в заданном сеансе s . Точнее, $1/\lambda$ есть среднее время между моментами поступления сообщений сеанса s . Пусть \bar{X} обозначает среднее время передачи сообщения по какой-либо заданной линии пути; т. е. если \bar{L} — средняя длина сообщений сеанса s , выраженная в битах, и r_s — скорость передачи для сеанса s , то $\bar{X} = \bar{L}/r_s$. На рис. 1.4 показаны эти моменты поступления сообщений и времена передачи.

Судя по рисунку, отведенная сеансу s часть линии связи действительно передает сообщения в течение довольно малой доли времени; остальное время эта часть линии связи простаивает. Поскольку $1/\lambda$ — среднее время между моментами поступления сообщений, а \bar{X} — среднее время занятости между моментами поступления, то интуитивно кажется правдоподобным, что отношение \bar{X} к $1/\lambda$, т. е. $\lambda\bar{X}$, есть доля времени, в течение которой отведенная сеансу s часть линии связи занята передачей. Это утверждение уточняется в гл. 3. Оно состоит в том, что если $\lambda\bar{X} \ll 1$, то отведенная сеансу s часть линии связи остается незагруженной большую часть времени (т. е. используется неэффективно).

Чтобы завершить нашу аргументацию неэффективности коммутации каналов для сетей передачи данных, нужно установить соотношение между \bar{X} и допустимой средней задержкой T при передаче сообщения от источника до пункта назначения. Поскольку \bar{X} — это средняя задержка передачи по одной линии,

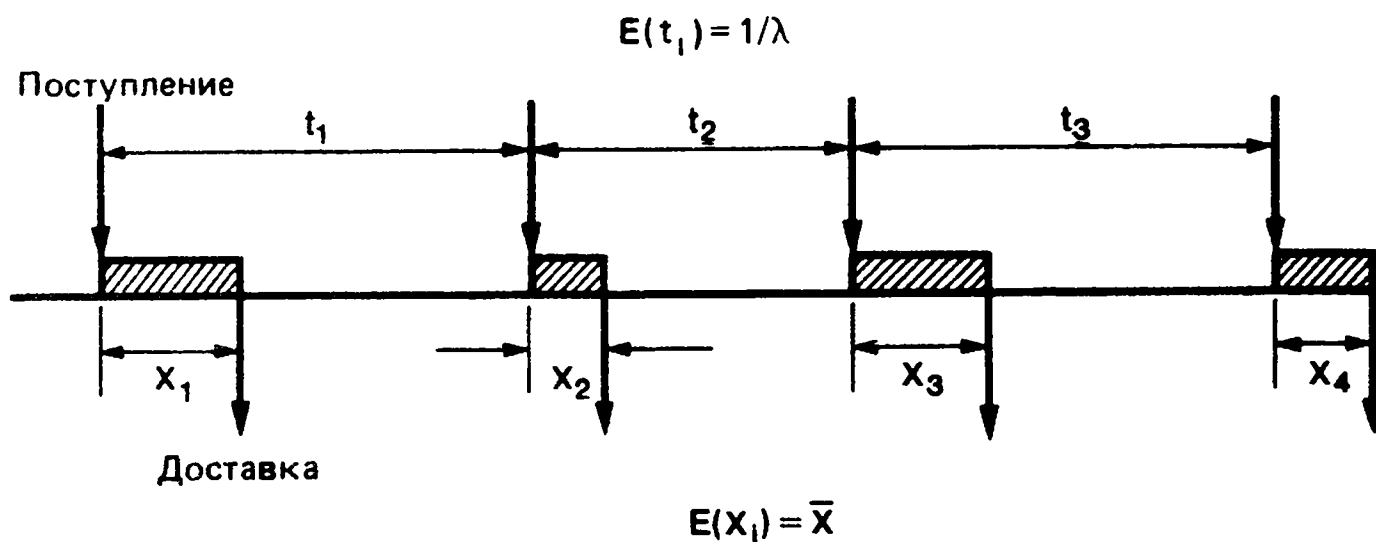


Рис. 1.4. Использование линии связи. \bar{X} — среднее время передачи сообщения. $1/\lambda$ — средний период между моментами поступления. Таким образом, доля времени, когда линия используется, не более $\lambda\bar{X}$.

ясно, что необходимо иметь $\bar{X} \ll T$, чтобы удовлетворить ограничению на полную задержку; поскольку $\bar{X} = \bar{L}/r_s$, это означает, что r_s должна быть достаточно большой, чтобы $\bar{X} \ll T$. Таким образом, если $\lambda T \ll 1$ (т. е. сообщения поступают редко и требуют малой задержки), $\lambda\bar{X} \ll 1$ и в сеансе линии связи используются неэффективно.

Для многих сеансов в сетях передачи данных λT имеет порядок 0,01 или меньше. Это означает, что доля времени, в течение которого с пользой используются отведенные этим сеансам части линий, не превышает 1 %. Суммируя вышеизложенное, отметим, что скорость передачи битов r_s , отводимая сеансу, должна быть достаточно большой, чтобы передавать сообщения в пределах требуемой задержки; когда λT мало, это приводит к неэффективному использованию линий. Поскольку затраты на передачу составляют главную часть стоимости сети, коммутация каналов не представляется привлекательной для сети передачи данных, когда λT мало для большинства сеансов. Эта аргументация не учитывает задержки распространения, коммутации в узлах и ожидания в очередях. (Задержка из-за очереди возникает в случае, когда в сеансе s появляется сообщение в то время, когда другое сообщение этого сеанса находится в стадии передачи.) Поскольку эти задержки должны добавляться к времени передачи по линии \bar{X} при проверке выполнения ограничения T на задержку, то \bar{X} часто должно быть существенно меньше, чем T . Это делает коммутацию каналов еще менее эффективной. Если задержки распространения и коммутации часто пренебрежимо малы, то задержкой ожидания в очередях, как показано в гл. 3, пренебрегать нельзя, особенно если λT превышает единицу.

Если в подсети связи используется передача с промежуточным накоплением, то каждый сеанс инициируется без предвари-

тельного отведения для него скорости передачи. Не проводится также обычное уплотнение линий связи. Вместо этого каждый раз один пакет или сообщение передается по линии связи, используя всю ее пропускную способность. Линия совместно используется в различных сеансах, но на основе требований (запросов), а не фиксированного отведения. Пакет или сообщение, которое поступает в узел коммутации, лежащий на пути к пункту назначения, ожидает в очереди, чтобы быть переданным по следующей линии их пути.

Передача с промежуточным накоплением имеет преимущество перед коммутацией каналов, состоящее в том, что каждая линия связи используется полностью всегда, когда имеется трафик, который нужно передать. В гл. 3, где исследуются очереди, показывается, что использование линий связи на основе запросов заметно уменьшает задержку передачи по сравнению с использованием на основе коммутации каналов. Передача с промежуточным накоплением имеет, однако, недостаток, заключающийся в том, что задержками, обусловленными очередями в узлах, трудно управлять. Пакеты, стоящие в очереди в узле, приходят в сеть с входов, расположенных во многих различных пунктах, и, таким образом, необходимы механизмы управления, которые уменьшали бы потоки с этих входов, когда время пребывания в очереди слишком велико или, что еще хуже, емкость буфера в узле близка к переполнению. Каждый такой механизм управления имеет задержку обратной связи. Во-первых, перегруженный узел должен как-то послать (по линиям сети) очень активным входным узлам некоторую управляющую информацию, требующую уменьшения их потоков. Во-вторых, значительное число пакетов может уже находиться в подсети, направляясь к заданному узлу. В преодолении этих трудностей состоит основная задача управления потоками, которая рассматривается в гл. 6. Читателю следует, однако, помнить, что эта задача возникает при использовании передачи с промежуточным накоплением; она, как правило, не существует при использовании коммутации каналов.

Передача с промежуточным накоплением используется в значительном числе разных вариантов. *Коммутация сообщений* — это один из вариантов передачи с промежуточным накоплением, при котором сообщения передаются как единое целое, а не разбиваются на пакеты. Если коммутация сообщений используется, то должна существовать максимальная длина сообщений, что, по существу, означает, что пользователь должен сам пакетировать сообщения, а не передавать выполнение этой операции сети. *Коммутация пакетов* — это вариант передачи с промежуточным накоплением, когда сообщения разбиваются на пакеты; из предыдущего обсуждения мы видим, что передача с промежуточным

накоплением и коммутация пакетов являются, по существу, синонимами. *Маршрутизацией с виртуальными каналами* называется вариант передачи с промежуточным накоплением, при котором во время инициирования сеанса выбирается некоторый конкретный путь, который сохраняется в течение сеанса. Это аналогично коммутации каналов в том смысле, что используется фиксированный путь, но он является виртуальным в том смысле, что пропускная способность используется только тогда, когда это необходимо. *Динамической маршрутизацией* (или *дейтаграммной маршрутизацией*) называется вариант передачи с промежуточным накоплением, при котором для каждого пакета выбирается определенный путь по сети на основе текущей информации, поступающей из посещаемых ими узлов. На практике в основном используется маршрутизация с виртуальными каналами, хотя существует много подходов, занимающих промежуточное положение между маршрутизацией с виртуальными каналами и динамической маршрутизацией. Основные вопросы маршрутизации рассматриваются в гл. 5.

1.3. Разбиение на уровни

Разбиение на уровни, или уровневая архитектура, является формой функциональной модульности, которая является центральной при проектировании сетей передачи данных. Понятие функциональной модульности (но, может быть, не сам термин) столь же старо, как и техника. В дальнейшем слово *модуль* используется для обозначения как устройства, так и процесса в некоторой вычислительной системе. Важно, что модуль выполняет некоторую выделенную функцию. Разработчики модуля должны глубоко понимать внутренние детали и работу этого модуля. Однако тот, кто использует этот модуль как компонент при построении более сложной системы, будет считать его «черным ящиком», т. е. пользователя интересуют входы, выходы и особенно функциональная связь выходов с входами, а не внутренняя работа модуля. Таким образом, черный ящик — это модуль, который описывается характеристикой вход-выход. Он может использоваться вместе с другими черными ящиками для построения более сложного модуля, который будет опять рассматриваться на более высоких уровнях как большой черный ящик.

Этот подход к проектированию, естественно, приводит к иерархии вложенных модулей. Сложная система должна быть построена как взаимосвязанное множество модулей высокого уровня и, возможно, некоторых дополнительных простых модулей, необходимых для реализации взаимосвязей и выполнения дополнительных простых функций. С точки зрения самого высокого уровня — уровня всей системы — каждый из этих модулей считается чер-

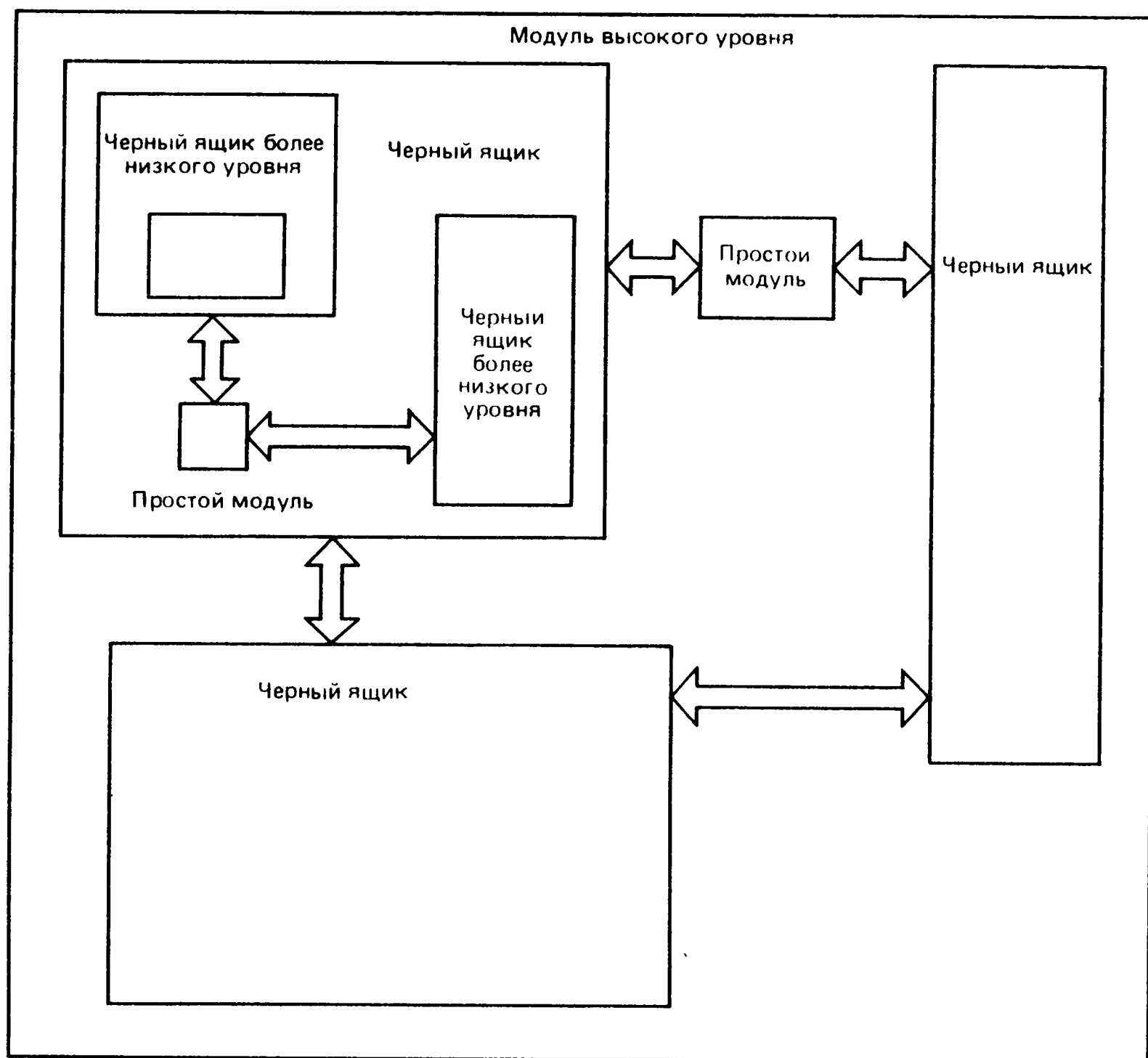


Рис. 1.5. Иерархия вложенных черных ящиков. Каждый черный ящик (кроме находящегося на самом нижнем уровне) содержит черные ящики более низкого уровня и, возможно, другие модули.

ным ящиком, но на следующем более низком уровне каждый модуль высокого уровня рассматривается как взаимосвязанное множество модулей следующего более низкого уровня, опять, возможно, дополненное простыми модулями. (Простым модулем называется такой модуль, который не разбивается на модули более низкого уровня.) Каждый модуль следующего более низкого уровня снова разбивается на модули еще более низкого уровня и так далее до самого низкого уровня иерархической цепи (рис. 1.5).

В качестве примера иерархического подхода можно представить вычислительную систему как множество процессорных модулей, множество модулей памяти и шинный модуль. Процессорный модуль можно в свою очередь представить как совокупность устройства управления, арифметического устройства, устройства выборки команд и устройства ввода-вывода. Аналогично арифметическое устройство может быть разбито на сумматоры, накапливающие регистры и т. д.

В большинстве случаев пользователю черного ящика нет необходимости знать детальный отклик выхода на входное воздействие. Например, неважно, когда точно изменится выходной сигнал в ответ на изменение входного сигнала, до того как он будет использован. Таким образом, модули (т. е. черные ящики) могут быть описаны с помощью допустимых отклонений, а не точных значений. Это приводит к стандартизированным модулям, и далее в свою очередь к возможности использования многих идентичных, ранее созданных (т. е. готовых) модулей в той же самой системе. К тому же такие стандартизированные модули можно легко заменить на новые функционально эквивалентные модули, которые дешевле и более надежны.

Все эти преимущества функциональной модульности (т. е. простота проектирования, легкость понимания и стандартные, взаимозаменяемые, широко распространенные модули) дали основания для введения уровневой архитектуры сетей передачи данных. Уровневую архитектуру можно рассматривать как иерархию вложенных модулей или черных ящиков, как описано выше. На каждом заданном уровне иерархии следующий более низкий уровень рассматривается как один или несколько черных ящиков с некоторым определенным функциональным описанием, которое используется на этом заданном более высоком уровне.

Необычным в уровневой архитектуре сетей передачи данных является то, что линии связи представляются черными ящиками на самом низком уровне иерархии. Вследствие этого черные ящики на каждом более высоком уровне являются на самом деле распределенными черными ящиками. Таким образом, черный ящик каждого более высокого уровня состоит из множества простых модулей (обычно по одному на каждый коммутационный узел или внешний пункт, входящий в систему) плюс один или несколько черных ящиков более низкого уровня. Простые модули из черного ящика на заданном уровне называются *паритетными процессами* или *паритетными модулями*.

В простейшем случае черный ящик состоит из двух паритетных процессов, по одному на каждый из двух узлов, и черного ящика, который находится на более низком уровне и представляет систему связи, соединяющую эти два паритетных процесса. Каждый процесс передает сообщение паритетному процессу в другом узле по нижнему уровню, т. е. через черный ящик, представляющий систему связи. Черный ящик этого нижнего уровня, как показано на рис. 1.6, может состоять из двух паритетных процессов более низкого уровня, принадлежащих разным узлам и соединенных системой связи — черным ящиком еще более низкого уровня. В качестве примера можно указать ситуацию, когда два руководителя государств не владеют общим языком. Каждый руководитель может передавать сообщение паритетному руко-

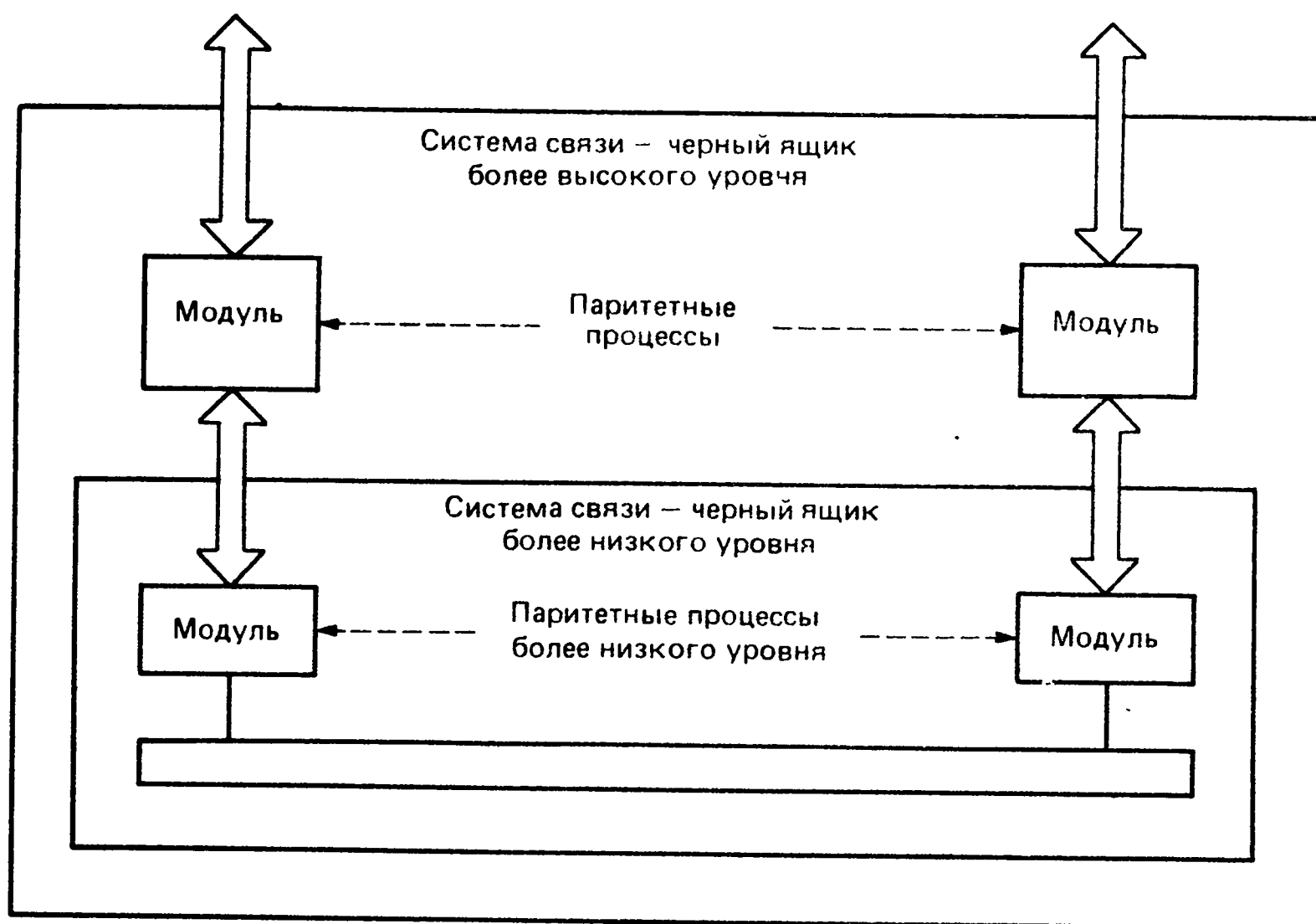


Рис. 1.6. Паритетные процессы в черном ящике, представляющем систему связи. Паритетные процессы ведут передачу через более низкий уровень, т. е. через черный ящик — систему связи, которая сама содержит паритетные процессы более низкого уровня.

водителю через свой транслятор (переводчик), который передает на языке, известном паритетному транслятору, а тот уже доставляет сообщение на языке паритетного руководителя государства.

Заметим, что у процесса передачи информации между двумя паритетными модулями уровня n , принадлежащими разным узлам, имеются два совершенно различных аспекта. Первый из них — это протокол (или распределенный алгоритм), с помощью которого паритетные модули обмениваются сообщениями или последовательностями битов, с тем чтобы обеспечить требуемое обслуживание для следующего более верхнего уровня. Второй — это описание точного интерфейса между модулем уровня n некоторого узла и модулем уровня $n - 1$ того же узла; через этот интерфейс происходит фактический обмен указанными сообщениями между уровнем n и черным ящиком — системой связи более низкого уровня. Первый из отмеченных аспектов является более важным (и более интересным) для концептуального понимания работы уровневой архитектуры, а второй имеет существенное значение при проектировании и стандартизации системы. В предыдущем примере общения руководителей государств первый аспект связан с переговорами между руководителями государств, тогда как второй связан с тем, что каждый руководитель госу-

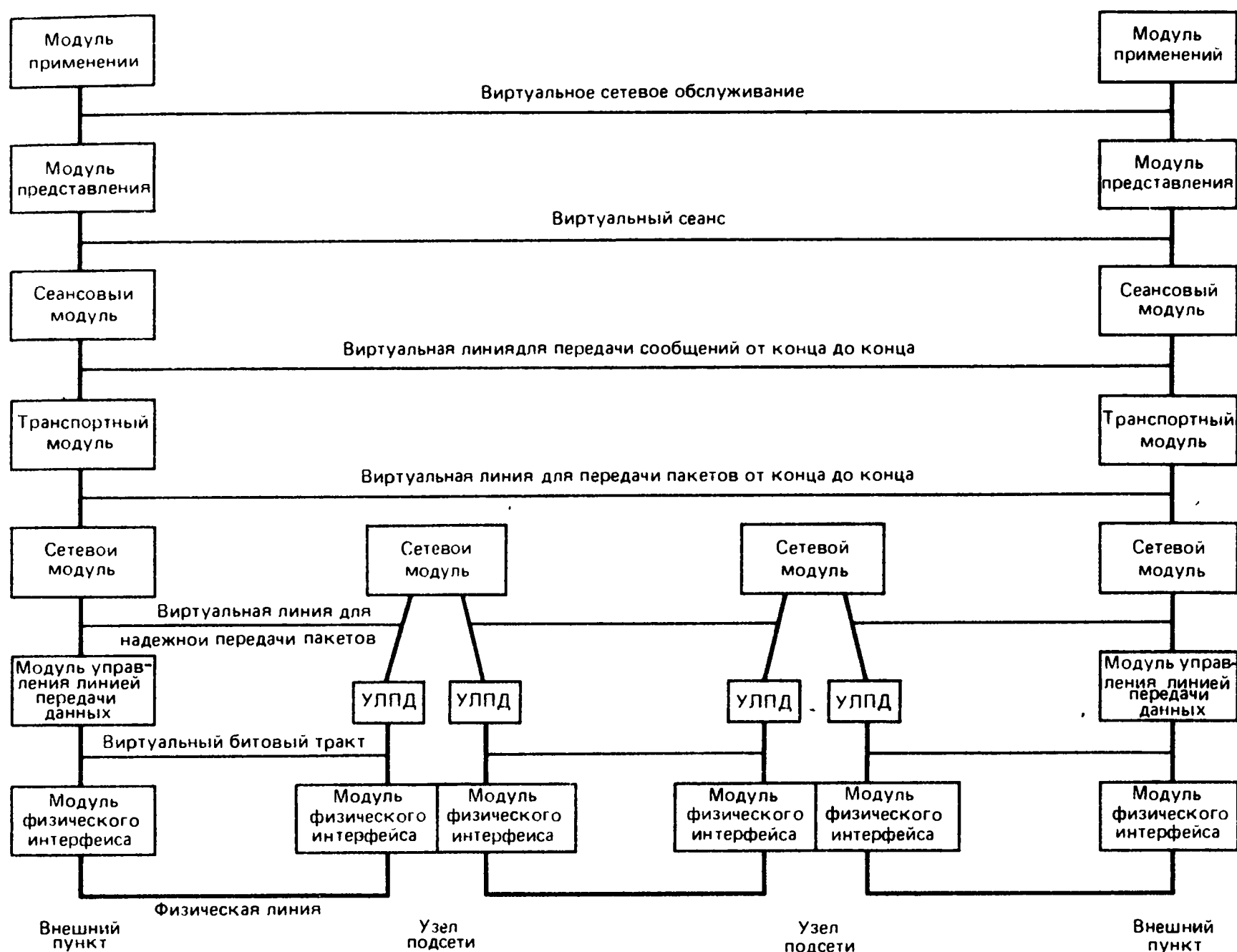


Рис. 1.7. Семиуровневая сетевая архитектура ВОС. Каждый уровень представляет виртуальную линию связи с заданными свойствами для следующего более высокого уровня.

дарства должен быть уверен в том, что транслятор действительно может переводить сообщения верно.

На рис. 1.7 представлена такая уровневая архитектура. Показанные уровни являются уровнями эталонной модели взаимодействия открытых систем (ВОС), предложенной Международной организацией стандартов (МОС) в качестве международного стандарта для сетей передачи данных. Многие существующие сети, в том числе SNA, DECNET, ARPANET и TYMNET, имеют несколько отличные уровни по сравнению с предлагаемыми этим стандартом. Однако уровни ВОС имеют особо ясную структуру, которая помогает понять концепцию разбиения на уровни. Некоторые отличия других сетей будут рассмотрены в дальнейшем.

1.3.1. Физический уровень

Функция *физического уровня* заключается в том, чтобы обеспечить виртуальную линию для передачи последовательности битов между любой парой узлов (или любым узлом и внешним пунк-

том), соединенных физическим каналом связи. Такая виртуальная линия называется *виртуальным битовым трактом*. Для выполнения этой функции на каждой стороне канала связи находится модуль физического интерфейса, функция которого состоит в преобразовании входящих битов, поступающих от следующего более высокого уровня (т. е. уровня управления линией передачи данных (УЛПД)), в сигналы, предназначенные для передачи по каналу, и на приемном конце в обратном преобразовании сигналов в биты. Модуль физического интерфейса, который выполняет эти преобразования, часто называют *модемом* (цифровой модулятор и демодулятор данных). Термин «модем» широко используется здесь для обозначения любого модуля, который выполняет указанную функцию, независимо от того, производится модуляция или нет.

Модемы и каналы связи рассматриваются в разд. 2 гл. 2. Разработчик модема должен знать детальные характеристики канала связи (ибо для различных типов каналов должны разрабатываться различные модемы). Однако по отношению к более высоким уровням черный ящик, образованный комплексом модем—канал—модем, представляет собой битовый тракт, причем сложности физического канала скрыты внутри этого тракта. Даже при представлении в виде тракта есть вопросы, которые следует обсудить.

Первый вопрос связан с синхронизацией битовой последовательности, поступающей в битовый тракт. Существуют три общие ситуации. К первой относится *синхронный* битовый тракт, когда биты передаются и принимаются через регулярные интервалы времени, т. е. 1 бит в течение t секунд. Модуль более высокого уровня УЛПД должен передавать биты с этой равномерной скоростью независимо от того, имеет ли он реальные данные для передачи. Ко второй ситуации относится *прерывисто-синхронный* битовый тракт, когда модуль УЛПД передает биты с равномерной скоростью в случае, когда он имеет биты для передачи, и прекращает поставлять биты, когда данных для передачи нет. В третьей ситуации имеются *асинхронные символы*, которыми пользуются обычно персональные компьютеры и низкоскоростные терминалы. В этой ситуации символы клавиатуры или различные управляющие символы преобразуются в битовые последовательности фиксированной длины (обычно используются 8-битовые последовательности, соответствующие коду ASCII) и отдельные символы в виде битовых последовательностей передаются асинхронно по мере того, как они генерируются.

Следующий вопрос связан с интерфейсом между модулем УЛПД и модемом. Можно подумать, что не должно возникать много трудностей при передаче последовательности битов от одного модуля к другому, особенно если они физически близки.

К сожалению, существует ряд досадных обстоятельств, связанных с таким интерфейсом. Например, модуль одной из сторон интерфейса может быть временно неработоспособным и, когда оба модуля становятся готовыми к работе, требуется некоторое инициирование, запускающее передачу битов. Кроме того, при синхронной работе одна из сторон должна обеспечивать синхронизацию. Ситуация усложняется из-за того, что многие производители поставляют модули одной из двух сторон интерфейса, вследствие чего необходима стандартизация интерфейса. Фактически существует большое число таких стандартов. Наиболее известными являются RS-232-C и физический уровень протокола X21.

Интерфейс RS-232-C позволяет решить проблему посредством выделения отдельного соединяющего модули провода для каждого типа управляющего сигнала, который может потребоваться. Эти идущие от модуля провода соединяются через стандартный 21-контактный разъем (хотя обычно требуется значительно меньше проводов). На языке связистов такой интерфейс соединяет DCE (оборудование передачи данных), которым в данном случае является модем, и DTE (оконечное оборудование обработки данных), которым в данном случае является уровень УЛПД и более высокие уровни.

В качестве примера использования интерфейса предположим, что DTE хочет начать передачу данных (или при инициировании, или из-за новой последовательности данных при прерывистой синхронной передаче). В этом случае DTE посылает сигнал DCE по проводу «запрос на передачу». DCE отвечает сигналом по проводу «свободен для передачи». DCE посылает также сигнал по проводу «DCE готово» в случае готовности к работе и сигнал по проводу «обнаружение несущей», когда оно убеждается, что противоположный модем и канал готовы к работе. Если DTE получает все эти сигналы (которые передаются просто уровнями напряжения), то оно затем начинает передавать данные через интерфейс по проводу данных от DTE к DCE.

Это взаимодействие является очень простым примером *протокола* или *распределенного алгоритма*. Каждый модуль выполняет операции, зависящие как от его собственного состояния, так и от информации, полученной от другого модуля. Много менее тривиальных протоколов описывается в последующих главах. Протокол RS-232-C содержит много других деталей, но в нем нет новых идей.

При исследовании интерфейса между модулем УЛПД и модемом удобно рассматривать провода, соединяющие модули, как физический канал, а УЛПД и модем — как паритетные процессы, выполняющие протокол интерфейса. Чтобы не путать основную функцию модуля УЛПД как паритетного процесса по отношению к противоположному модулю УЛПД с его функцией

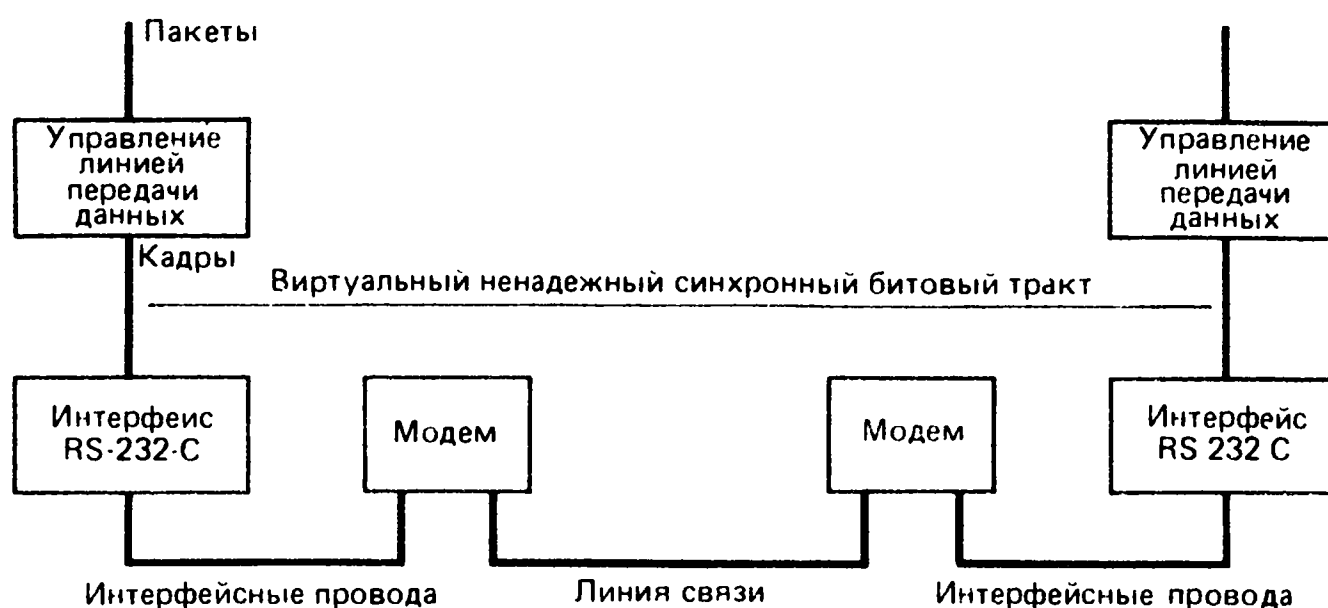


Рис. 1.8. Разбиение на уровни в случае, когда интерфейс между УЛПД и модемом рассматривается как интерфейс в физической среде, образованной набором проводов.

более низкого уровня по взаимодействию с модемом, вводится дополнительный модуль (рис. 1.8), который выполняет протокол взаимодействия с модемом.

Протокол физического уровня X21 функционально аналогичен RS-232-C, но в нем используется меньшее количество проводов (берется восемь проводов, хотя разъем имеет 15 контактов). Идея состоит в том, что устраняется выделение отдельного провода каждому возможному сигналу и удваивается использование проводов цифровой логикой в модулях. Физический уровень X21 используется как физический уровень в протоколе X25, который рассматривается в гл. 2.

Из вышеизложенного должно быть ясно, что отделение вопросов, связанных с модемом и его интерфейсом, от более высоких уровней сети, имеет большое концептуальное преимущество. Отметим, что это уже было сделано, по существу, в предыдущих разделах, когда говорилось о скорости, с которой биты могут передаваться по линиям связи. Однако следует также отметить, что модемы невозможно полностью отделить от сетевых вопросов. Например, что лучше: иметь модем, который передает R бит в секунду с частотой ошибок 10^{-6} , или модем, который передает $2R$ бит в секунду с частотой ошибок 10^{-4} ? На этот вопрос нельзя ответить, не зная, как исправляются ошибки на верхних уровнях архитектуры.

1.3.2. Уровень управления линией передачи данных (УЛПД)

Второй уровень на рис. 1.7 является *уровнем управления линией передачи данных (УЛПД)*. Каждая линия связи (т. е. двунаправленный виртуальный битовый тракт, который предоставляется уровнем 1) имеет на каждом своем конце модули управ-

ления линией передачи данных (паритетные процессы). Цель управления линией передачи данных — превратить ненадежный битовый тракт уровня 1 в виртуальную линию связи более высокого уровня, способную в обоих направлениях передавать пакеты асинхронно, но безошибочно. С точки зрения уровня УЛПД пакет представляет собой просто последовательность битов, которая поступает от следующего более высокого уровня.

Связь на этом уровне является асинхронной в двух отношениях. Во-первых, время прохождения пакета от входа в модуль УЛПД на одном конце линии до выхода на другом конце не остается постоянным. Эта изменчивость обусловлена как необходимостью исправлять ошибки, которые возникают на физическом уровне, так и непостоянством длины пакетов. Во-вторых, временной интервал между последовательными входами пакетов в модуль УЛПД на одном конце линии также меняется. Это вызывается как тем, что более высокие уровни могут не иметь пакетов для передачи в заданный момент, так и тем, что УЛПД не может принять новые пакеты, когда слишком много старых пакетов передается повторно из-за ошибок передачи.

Управление линией передачи данных будет рассматриваться более подробно в гл. 2. По существу, передающий модуль УЛПД помещает некоторое число управляющих битов в начале и конце каждого пакета, в результате чего получается более длинная последовательность битов, называемая *кадром*. Некоторые из этих управляющих битов обнаруживают возникновение ошибок в переданных кадрах, некоторые запрашивают повторные передачи, если возникли ошибки, а некоторые обозначают начало и конец кадров. Алгоритмы (или протоколы), выполняющие эти задачи, распределены между паритетными модулями УЛПД, находящимися на обоих концах линии связи, и довольно сложны, так как сами управляющие биты подвержены ошибкам передачи. Обычно уровень УЛПД гарантирует, что пакеты покидают принимающий модуль УЛПД в том же порядке, в каком они входят в передающий модуль УЛПД, однако не все методы управления линией передачи данных обеспечивают это; относительные преимущества упорядочения обсуждаются в гл. 2.

Наше предыдущее описание физического уровня и уровня УЛПД было основано на двухточечных линиях связи, в которых сигнал, принятый на одном конце линии, является зашумленной версией сигнала, переданного на другом конце. В некоторых сетях все передачи или некоторые из них совершаются по линиям с множественным доступом. Для таких линий сигнал, принятый в одном узле, является суммой сигналов, передаваемых от целого ряда передающих узлов, а сигнал, переданный из одного узла, может быть услышан в целом ряде других узлов. Такая ситуация возникает в спутниковых каналах, радиоканалах, а также при

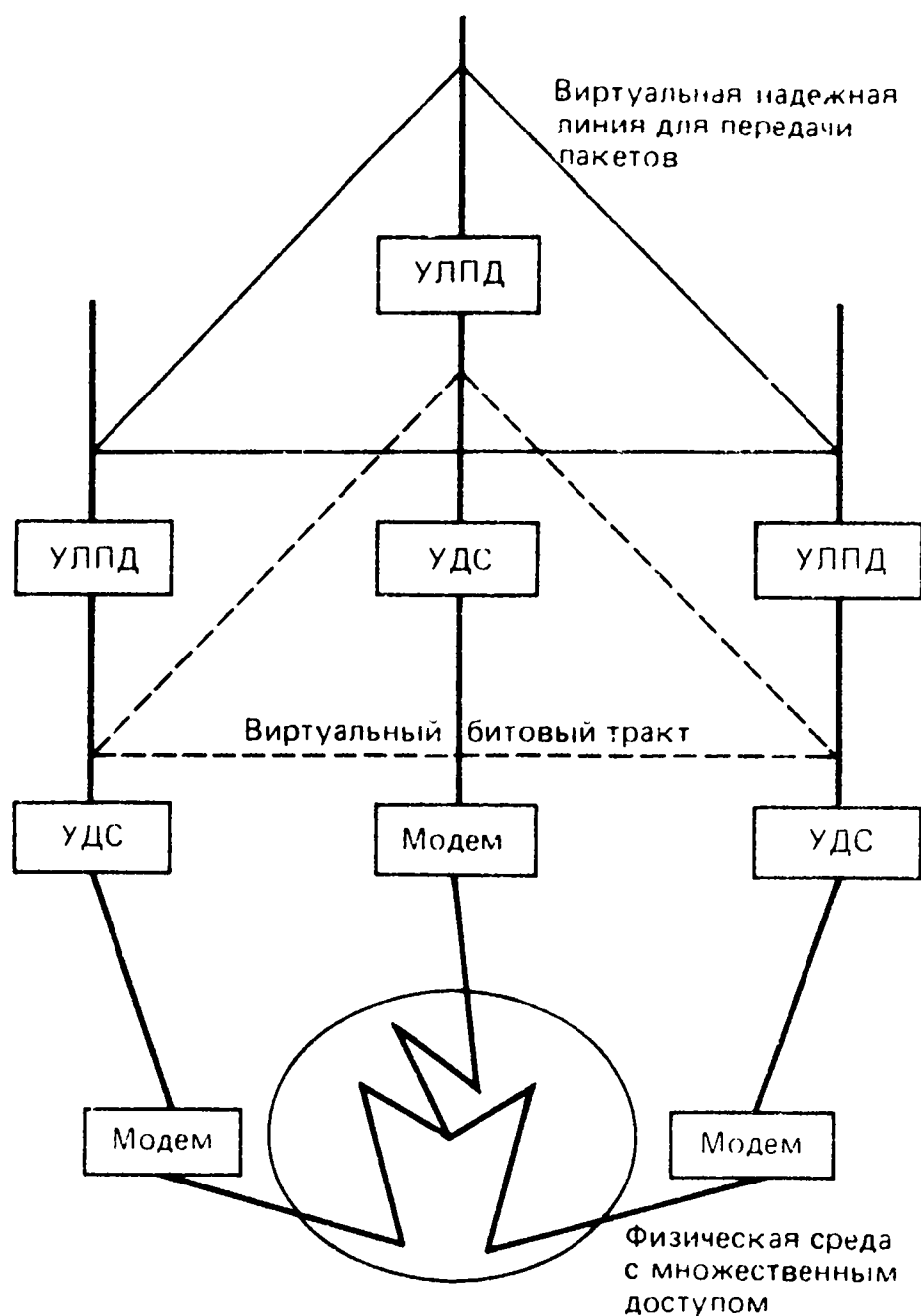


Рис. 1.9. Разбиение на уровни для канала с множественным доступом. Физическая среда доступна всем трем пользователям, каждый из которых слышит сигналы, переданные другими пользователями. На подуровне УЛПД используются виртуальные двухточечные битовые тракты, находящиеся под ним. На подуровне УДС используется битовый тракт с множественным доступом, а модемы имеют доступ к реальному каналу.

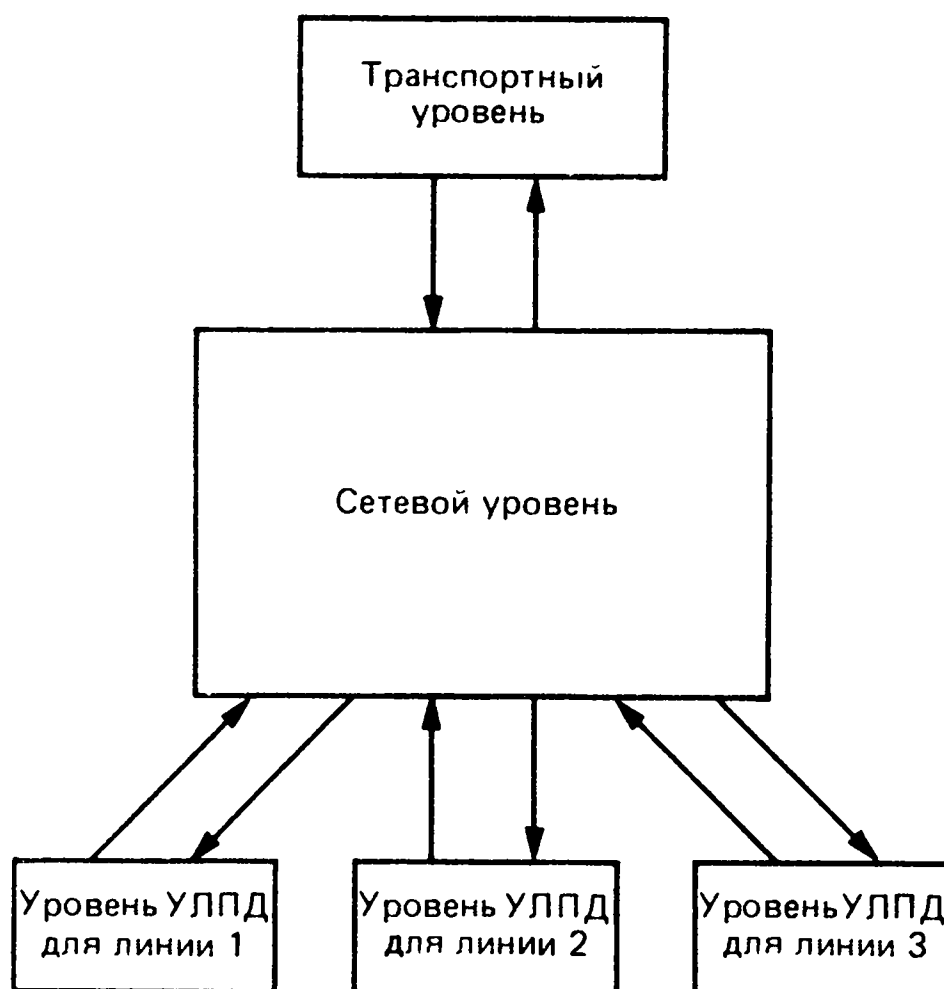
передаче по кабелям, оптическим линиям и телефонным линиям с несколькими отводами. Связь в режиме множественного доступа будет рассматриваться в гл. 4.

Соответствующие уровни при передаче в режиме множественного доступа несколько отличны от тех, что используются в сетях с двухточечными линиями. Как и ранее, уровень УЛПД должен обеспечивать верхние уровни виртуальной линией для безошибочной передачи пакетов, а физический уровень должен предоставлять битовый тракт. Однако появляется необходимость в промежуточном уровне для управления линией с множественным доступом таким образом, чтобы из каждого узла можно было передавать кадры без постоянной интерференции с остальными узлами. Этот уровень называется *управлением доступом к среде* (УДС). Обычно его считают подуровнем уровня 2, а традиционное УЛПД — вторым подуровнем этого уровня. На рис. 1.9 показана взаимосвязь этих уровней.

1.3.3. Сетевой уровень

Третий уровень на рис. 1.7 является сетевым. С каждым узлом и с каждым внешним пунктом сети связан свой процесс сетевого уровня. Все эти процессы паритетные, все они взаимодействуют

Рис. 1.10. На сетевой уровень узла или пункта могут поступать пакеты от уровня УЛПД каждой входящей линии и (в случае пункта) от транспортного уровня. С сетевого уровня эти пакеты могут направляться к тому же множеству модулей.



друг с другом при выполнении маршрутизации и управления потоком в сети. Когда пакет входит в узел или пункт по одной из входных линий связи, он проходит через физический уровень на уровень УЛПД и, если последний его принял, переправляется выше, на сетевой уровень этого узла (рис. 1.10). Новые пакеты, генерируемые на внешних пунктах, приходят на сетевой уровень с транспортного уровня. Аналогично управляющие пакеты более высокого уровня входят в сетевой уровень во внешнем пункте из транспортного уровня. Наконец, управляющие пакеты для маршрутизации и функций управления потоком, о которых говорится ниже, могут генерироваться на сетевом уровне в узле.

Процесс сетевого уровня в узле выполняет функцию маршрутизации или коммутации, когда решает, куда посылать все эти пакеты. Пакеты, предназначенные для некоторого другого узла или пункта, посылаются по соответствующей линии через уровень УЛПД этой линии, тогда как управляющие пакеты маршрутизации или управления потоком обрабатываются прямо в этом модуле. Пакеты, поступающие в данный пункт, переправляются на транспортный уровень.

Процесс сетевого уровня определяет также, когда принимать пакеты от более высокого уровня и когда передавать пакеты другим узлам или пунктам. Последние решения осуществляются управлением потоком, которое следит за перегрузкой в сети. Из-за связанных с этим задержек, а также из-за задержек при приеме пакетов на уровень УЛПД на сетевом уровне должна осуществляться буферизация пакетов.

Паритетные процессы сетевого уровня должны обмениваться информацией, чтобы производить маршрутизацию и управление потоком. Часть этого обмена происходит через управляющие биты, которые добавляются к пакетам данных, перемещающимся от узла к узлу. Эти управляющие биты указывают источник пакета и пункт назначения, а также помогают управлять потоком пакетов определенного сеанса. Другая часть этого обмена происходит через пакеты, которые содержат только управляющие биты. Эти пакеты устанавливают новые сеансы и передают информацию, необходимую для образования маршрутных таблиц.

Сетевой уровень концептуально наиболее сложный в уровне-вой иерархии, поскольку все паритетные процессы этого уровня должны действовать совместно. На более низких уровнях (кроме подуровня УДС при множественном доступе) паритетные процессы составляют пары, по одному процессу на каждой стороне линии связи. На более высоких уровнях паритетные процессы также составляют пары, по одному на каждой стороне сеанса. Таким образом, сетевой уровень и подуровень УДС являются единственными уровнями, на которых общие алгоритмы распределены между многими географически разделенными процессами.

Развитие способности разрабатывать и понимать такие распределенные алгоритмы является одной из основных целей этой книги. В гл. 2 описываются более простые формы распределенных алгоритмов, включающих только два паритетных процесса, находящихся на противоположных концах линии. В гл. 4 рассматривается подуровень УДС, в который входит множество паритетных процессов, а в гл. 5 и 6 при описании маршрутизации и управления потоком используются распределенные алгоритмы сетевого уровня, в которые входит много паритетных процессов.

Когда сетевой уровень и все более низкие уровни во всех узлах и пунктах рассматриваются как один черный ящик, пакет, поступающий на сетевой уровень со следующего более высокого уровня в некотором пункте, через некоторое время появляется в интерфейсе сетевого уровня со следующим более высоким уровнем в пункте назначения. Таким образом, сетевой уровень подобен виртуальной сквозной линии, которая переносит пакеты от пункта генерации до пункта назначения. В зависимости от конструкции сетевого уровня эта виртуальная линия может быть надежной, доставляющей каждый пакет безошибочно один и только один раз, или ненадежной, не доставляющей некоторые пакеты или доставляющей их с ошибками. На более высоких уровнях в последнем случае эти ошибки должны исправляться. Сетевой уровень также может обеспечивать доставку всех пакетов каждого сеанса в порядке их следования или беспорядочно. Относительные преимущества этих вариантов обсуждаются далее в разд. 2.8.

1.3.4. Транспортный уровень

Четвертым на рис. 1.7 является *транспортный уровень*. Здесь для каждой виртуальной сквозной линии, предоставляемой сетевым уровнем, имеется пара паритетных процессов, по одному на каждом конце виртуальной линии сетевого уровня. На транспортном уровне выполняется ряд функций, однако в некоторых конкретных сетях не все из них обязательно требуются.

Во-первых, на транспортном уровне сообщение разбивается на пакеты на передающем конце, а на приемном конце пакеты собираются в сообщения. Эта функция сборки относительно проста, если процесс транспортного уровня имеет в распоряжении достаточное буферное пространство, но может быть довольно сложной, если буферное пространство ограничено и должно совместно использоваться многими виртуальными сквозными линиями. Если сетевой уровень нарушает порядок доставляемых пакетов, то проблема сборки становится еще более трудной.

Во-вторых, на транспортном уровне могут мультиплексироваться несколько низкоскоростных сеансов, имеющих один и тот же источник и пункт назначения, в один сеанс сетевого уровня. Поскольку подсеть связи в этом случае обслуживает только один сеанс, количество сеансов в подсети и соответствующая управляющая информация уменьшаются. До некоторой степени аналогичным образом на транспортном уровне один высокоскоростной канал может расщепляться на несколько сеансов сетевого уровня. Это может потребоваться в том случае, когда управление потоком на сетевом уровне не способно предоставлять одним сеансам более высокоскоростное обслуживание по сравнению с другими сеансами, однако ясно, что лучшее решение этой проблемы состоит в том, чтобы на сетевом уровне скорость передачи устанавливалась в соответствии с требованием сеанса.

В-третьих, если на сетевом уровне возникают ошибки, на транспортный уровень может возлагаться функция обеспечения надежной сквозной связи для тех сеансов, которым это необходимо. Даже когда сетевой уровень проектируется так, чтобы обеспечивать надежную связь, транспортный уровень должен принимать в этом участие в том случае, когда тот или иной конечный пункт выходит из строя или когда сеть становится разъединенной из-за повреждений линий связи. Эти проблемы выхода из строя обсуждаются далее в разд. 2.8 и в гл. 5 и 6.

Наконец, сети передачи данных часто соединяются; при этом коммуникационные пути проходят по нескольким подсетям (это известно как *межсетевое соединение*). Обычно сети соединяются посредством специальных узлов, называемых *шлюзами*. Соединяемые сети часто имеют несовместимые сетевые уровни и поэтому требуется, чтобы на транспортный уровень в шлюзе поступали

пакеты от сетевого уровня одной сети и им придавалась такая форма, которая принята в другой сети. Если сети имеют различные максимальные размеры пакетов, то на транспортном уровне должны иногда разбиваться длинные пакеты на два или большее число пакетов меньшего размера.

Особенно важным примером межсетевого соединения является случай, когда ряд локальных сетей (рассматриваемых в гл. 4) соединяется посредством глобальной сети. Поскольку в локальных сетях связь относительно дешевая и можно легко достичь очень малых задержек, конструкция трех нижних уровней локальной сети обычно сильно отличается от того, что свойственно глобальной сети. Поэтому неудивительно, что сетевые уровни являются несовместимыми.

1.3.5. Сеансовый уровень

Сеансовый уровень является следующим уровнем, лежащим над транспортным уровнем, согласно иерархии ВОС, показанной на рис. 1.7. Одна из функций сеансового уровня подобна справочной службе телефонных сетей, т. е. если пользователь хочет воспользоваться предоставляемым сетью видом обслуживания, но не знает, куда надо обратиться, чтобы получить к нему доступ, то с этого уровня на транспортный уровень передается информация, необходимая для установления сеанса. Например, было бы целесообразно возложить на этот уровень функцию распределения нагрузки между множеством процессоров, совместно выполняющих вычислительные задачи в сети.

Сеансовый уровень также связан с *правами доступа* при инициировании сеансов. Например, если корпорация использует сеть общего пользования для передачи документов между своими филиалами, то эти документы не должны быть доступны несанкционированным пользователям. Аналогично, когда пользователь получает доступ к обслуживанию, сеансовый уровень помогает решить вопрос о том, кто платит за обслуживание.

По существу, на сеансовом уровне осуществляется управление взаимодействием между двумя конечными точками при установлении сеанса, тогда как на сетевом уровне производится управление теми аспектами установления сеанса, которые связаны с подсетью связи. Способ распределения функций по инициированию сеанса между сеансовым, транспортным и сетевым уровнями меняется от сети к сети и во многих сетях эти три уровня не являются разделенными.

1.3.6. Уровень представления

Главными функциями *уровня представления* являются шифрование данных, их сжатие и кодовое преобразование. Необходимость шифрования в военных организациях очевидна, но в до-

полнение к этому корпорации и индивидуальные пользователи часто должны посылать сообщения, которые должны читать только санкционированные получатели. Сети передачи данных следует проектировать так, чтобы предотвратить попадание сообщений к ошибочным получателям, однако возможны случайные сбои в работе как внешних пунктов, так и подсети связи; это приводит к необходимости шифрования важных сообщений.

О желательности сжатия данных для уменьшения числа передаваемых битов уже упоминалось. Эту функцию можно было бы выполнять на любом уровне, но отдельное сжатие данных каждого сеанса имеет преимущество, поскольку различные сеансы обладают различными типами избыточности своих сообщений. В частности, если сжатие данных необходимо, оно должно выполняться перед шифрованием, поскольку зашифрованные данные не будут иметь какую-либо легко расшифровываемую избыточность.

Наконец, кодовое преобразование иногда необходимо из-за несовместимости терминалов, печатающих устройств, графических терминалов, файловых систем и т. д. Например, некоторые терминалы используют код ASCII для представления символов в виде 8-битовых байтов, тогда как другие терминалы применяют код EBCDIC. Сообщения, записанные в одном коде, должны быть преобразованы в другое кодовое представление, чтобы их мог читать терминал, использующий этот другой код.

1.3.7. Уровень применений

Уровень применений — это просто то, что остается не выполненным после того, как будут выполнены функции всех других уровней. Для каждого приложения требуется свое собственное программное обеспечение (т. е. свои паритетные процессы). На более нижних уровнях выполняются те части общей задачи, которые требуются для многих различных приложений, а на уровне применений выполняется та часть работы, которая специфична для конкретного приложения.

Теперь, когда преимущества уровневого подхода должны быть ясны, можно поинтересоваться, должны ли уровни иметь точно такое разделение, как рассмотренное здесь (особенно верхние уровни), и не слишком ли много семи уровней. С одной стороны, имеется большая потребность в стандартизации интерфейсов и функций каждого уровня, даже если стандарт является в чем-то не подходящим. Эта потребность особенно велика для международных сетей и для производителей оборудования, которое должно правильно работать в сочетании с оборудованием других производителей. С другой стороны, стандартизация препятствует нововведениям и это особенно чувствительно в такой новой области, как сети передачи данных, которые еще не до конца поняты.

Один конкретный недостаток, связанный с семью уровнями, состоит в том, что каждое сообщение должно пройти через семь процессов только для того, чтобы войти в подсеть связи, и все это может привести к значительной задержке. Автор данной книги не ратует за использование этого конкретного набора уровней, но и не предлагает иного набора. Скорее он преследует цель, состоящую в том, чтобы помочь лучше понять функции, которые должны выполняться; есть надежда, что это будет способствовать стандартизации. Уровни существующих сетей, рассматриваемых в последующих главах, на самом деле не соответствуют полностью уровням ВОС.

1.4. Пример задачи, связанной с использованием распределенного алгоритма

Все уровни, рассмотренные в предыдущем разделе, имеют много общего. Они содержат паритетные процессы, по одному на каждой из двух или более графически разделенных точек, и паритетные процессы связываются с помощью средств связи, которые обеспечиваются следующим более низким уровнем. Паритетные процессы данного уровня имеют общую цель (т. е. задачу или функцию), которую они пытаются достичь совместно с помощью обработки и обмена информацией. Алгоритм достижения цели является *распределенным алгоритмом*, или *протоколом*. Распределенный алгоритм состоит из множества *локальных алгоритмов*, каждый из которых выполняется некоторым паритетным процессом. Локальный алгоритм, выполняемый одним из паритетных процессов, включает выполнение различных операций над имеющимися данными; кроме того, в различных точках алгоритма производится либо посылка данных одному или большему числу других паритетных процессов, либо чтение (или ожидание) данных, посланных другим паритетным процессом.

В простейшем распределенном алгоритме порядок выполнения операций различными локальными алгоритмами полностью детерминирован. Например, один локальный алгоритм может выполнить несколько операций и затем надежно передать некоторые данные другому локальному алгоритму, который затем выполнит некоторые операции и передаст обратно некоторые данные. Только один локальный алгоритм действует в каждый момент времени. Распределенный алгоритм аналогичен централизованному алгоритму, который выполняет все операции последовательно в одном географическом пункте.

В более сложных случаях несколько локальных алгоритмов могут действовать одновременно, но каждый все же ждет в заданных точках определенных сообщений от других локальных алгоритмов. В этом случае общий распределенный алгоритм действует

детерминированным образом (при заданных входных данных для паритетных процессов), но устраняется строгий порядок выполнения операций различными локальными алгоритмами.

В наиболее сложном случае (который наиболее интересен) порядок выполнения локальным алгоритмом операций зависит от порядка поступления данных (как от следующего более высокого уровня, так и от какого-либо паритетного процесса). Кроме того, если более низкие по уровню средства связи ненадежны, то данные, посланные паритетным процессом, могут никогда не прийти или поступить с ошибками.

Большинству людей хорошо знакома эта ситуация, поскольку часто приходится выполнять задачи, требующие взаимодействия с другими людьми, причем часто используя при этом ненадежную связь. В пробных ситуациях, однако, люди приступают к решению проблемы после ее возникновения, а не продумывают все возможные случаи предварительно, как это должно быть в случае распределенного алгоритма.

Чтобы дать некоторое представление о распределенных алгоритмах, вниманию читателя предлагается очень простая задача, которая содержит ненадежную связь; эта задача не имеет решения. Аналогичные ситуации часто возникают при изучении сетей передачи данных, поэтому полезно понять эту задачу в наиболее упрощенном варианте.

Имеются три армии, две из которых окрашены в красный цвет, одна — в лиловый. Лиловая армия располагается между двумя красными армиями, разделяя их; если красные армии смогут атаковать одновременно, то они победят, но, если они будут атаковать порознь, то одержит верх лиловая армия. Красные армии могут обмениваться сообщениями только с помощью связного, проникающего через расположение лиловой армии, поэтому с некоторой вероятностью каждый такой связной может быть взят в плен и сообщение не будет доставлено (рис. 1.11). Красные армии хотят атаковать одновременно в некоторое заданное время, но каждая из них не хочет начинать атаку, если не уверена полностью, что другая армия тоже будет атаковать. В этом случае первая армия может послать сообщение: «Давайте атаковать в субботу, в полдень; пожалуйста, пришлите подтверждение, если вы согласны».

Вторая армия, получив такое сообщение, может послать сообщение: «Мы согласны; пришлите уведомление, если вы получите наше сообщение». Нетрудно понять, что эта стратегия ведет к бесконечной последовательности сообщений, причем последняя посылающая сообщение армия не желает атаковать, пока не получит подтверждение от другой стороны.

Более удивительно то, что не существует стратегии, которая позволила бы двум армиям синхронизировать свои действия.

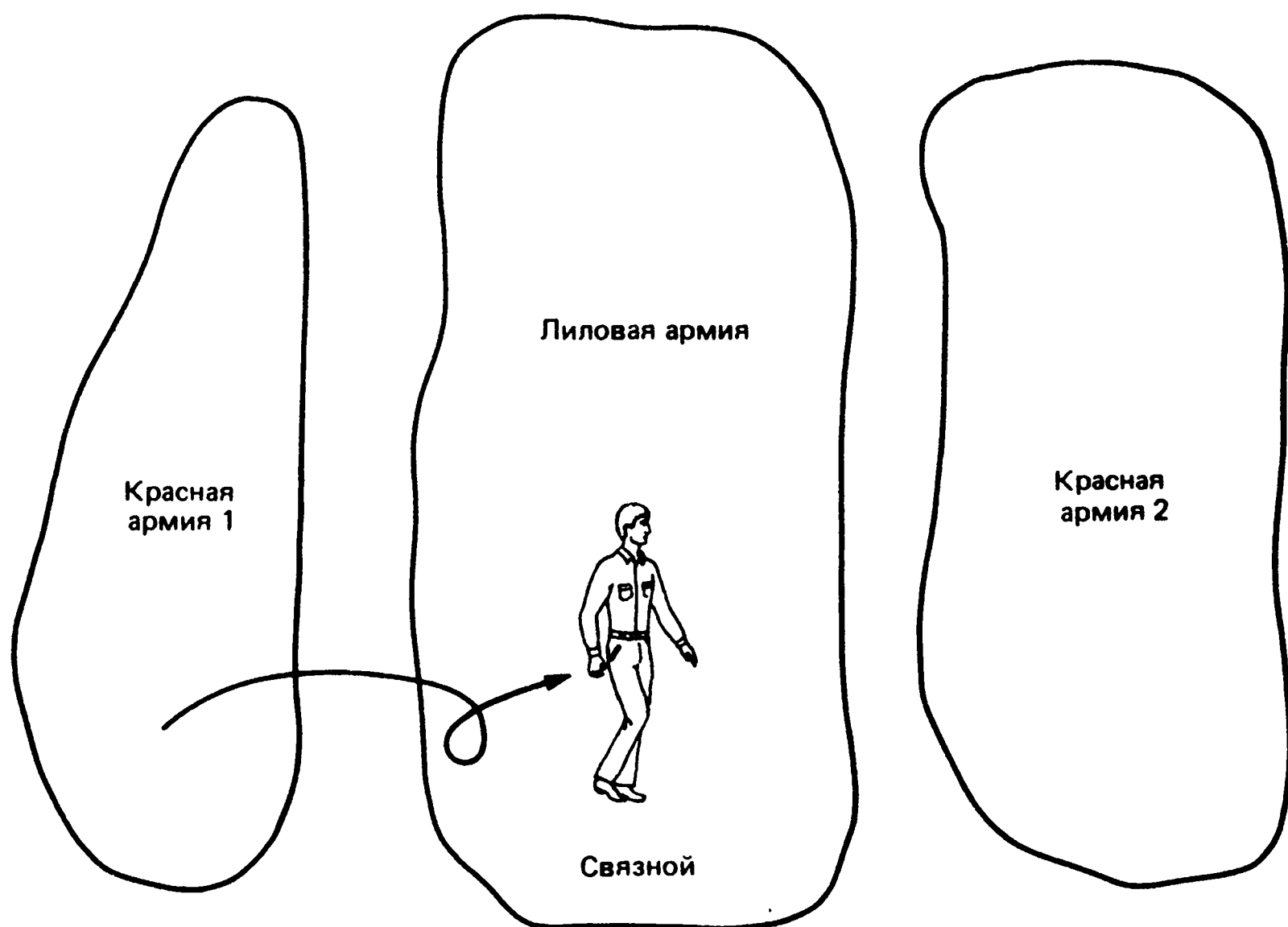


Рис. 1.11. Связной проносит сообщение от красной армии 1 для красной армии 2 через расположение противника. Если связной захватывается в плен, то сообщение не доставляется. Красная армия 1 не знает о захвате, а красная армия 2 не знает о существовании сообщения.

Чтобы увидеть это, предположим, что каждая армия первоначально находится в состоянии 0 и остается в этом состоянии, если она не получает никаких сообщений. Если армия идет в атаку, то она переходит в состояние 1, но она не будет переходить в состояние 1, если нет уверенности, что другая армия перейдет в состояние 1. Мы также предполагаем, что армия может перейти в другое состояние только в момент, когда она получает сообщение (в задаче синхронизации армий это предположение, по существу, устраняет всякую информацию, не относящуюся к сообщениям). Теперь рассмотрим любой порядок получения сообщений двумя армиями. Первая армия, получив сообщение, не может перейти в состояние 1, поскольку у нее нет уверенности, что какое-либо сообщение будет получено другой армией. Вторая армия, получив сообщение, также не может перейти в состояние 1, поскольку она не уверена, что другая сторона получит последующие сообщения, и даже если она знает, что другая сторона получила первое сообщение, она знает, что другая сторона не находится в данный момент в состоянии 1. Действуя таким образом (или более формально с помощью индукции), получаем, что ни одна армия не сможет никогда перейти в состояние 1.

Что удивительно, убедиться в справедливости этой аргументации довольно трудно. Трудность связана не с индукцией в аргументации, а с вопросом, точно ли модель представляет описанную ситуацию. По-видимому, проблема состоит в том, что мы не имеем дело с распределенными вопросами на строгом уровне; в области классических технических задач не встречаются ситуации, в которых распределенные решения должны приниматься на основе распределенной информации.

Если указанные выше условия несколько ослабить, потребовав лишь большой вероятности одновременной атаки, то задачу можно будет решить. Первая армия просто решает начать атаку в некоторый момент времени и посылает другой стороне одновременно большое число связных. Первая армия в этом случае уверена, что с большой вероятностью вторая армия получит сообщение, а вторая армия уверена, что первая армия начнет атаку.

К счастью, большинство задач связи между паритетными процессами, с которыми сталкиваются в сетях передачи данных, не требуют такого одновременного согласия. Обычно нужно, чтобы один процесс входил в определенное состояние, будучи уверенным, что паритетный процесс в конце концов войдет в соответствующее состояние. Может потребоваться, чтобы первый процесс ожидал подтверждения этого события, но тупиковой ситуации задачи трех армий, в которой ни один процесс не может действовать до тех пор, пока другие не начали действовать, не возникает.

1.5. Замечания и дополнительная литература

В учебниках Таненбаума [227] и Сталлингса [220] предлагается альтернативное изложение материала этой главы. Книга Таненбаума, хотя и немного устарела, широко распространена и содержит несколько глав о верхних уровнях архитектуры ВОС. В книге Сталлингса приведено множество практических деталей современной сетевой практики. Некоторые перспективы исторической эволюции сетей передачи данных указаны в [102].

Новые технические разработки критически важны как для проектирования сетей, так и для их использования. В журналах IEEE Spectrum, IEEE Communications Magazine и IEEE Computer часто публикуются статьи по этим новым разработкам.

Хорошим справочником по уровневой архитектуре является [101], а некоторые интересные комментарии относительно будущей стандартизации уровней приводятся в [103].

2. Управление линией передачи данных и каналы связи

2.1. Обзор

В этой главе исследуются два нижних уровня архитектуры, представленной на рис. 1.7, т. е. физический уровень и уровень управления линией передачи данных. К физическому уровню относятся сами каналы связи, используемые сетью, а также все, что требуется интерфейсным модулям, находящимся на концах каналов для передачи и приема цифровых данных (рис. 2.1). Мы называем эти модули модемами (модуляторами и демодуляторами цифровых данных), хотя в большинстве случаев никакой модуляции или демодуляции не требуется. При передаче данных модем преобразует входящие двоичные данные в сигнал, соответствующий каналу. При приеме модем преобразует принятый сигнал в двоичные данные.

В разд. 2.2 дается краткое описание каналов связи и способа выполнения модемами своих функций при заданном канале. Причина краткости состоит не в том, что эта тема недостаточно важна или интересна, а в том, что для полного понимания требуется математическая подготовка в области теории линейных систем, случайных процессов и современной теории связи. В этом разделе даются как обзор, предназначенный для тех, кто не имеет такой подготовки, так и более детальные сведения и перспектива для более подготовленных читателей. Пока можно не рассматривать физический уровень, представляя просто, что он обеспечивает виртуальный битовый тракт, используемый на уровне управления линией передачи данных (УЛПД).

Существует несколько различных типов виртуальных битовых трактов, которые могут обеспечиваться физическим уровнем. Одним из них является *синхронный* битовый тракт, когда передающая сторона модуля УЛПД поставляет биты передающей стороне модема с синхронной скоростью, т. е. один бит каждые T секунд при некотором фиксированном T . Если модуль УЛПД временно не имеет данных для передачи, то он должен продолжать передачу фиктивных битов, называемых *холостым заполнением*, до тех пор пока не получит данные. Принимающий модем восстанавливает эти биты синхронно (с задержкой и случайными ошибками) и передает биты, в том числе и холостое заполнение, соответствующему модулю УЛПД.

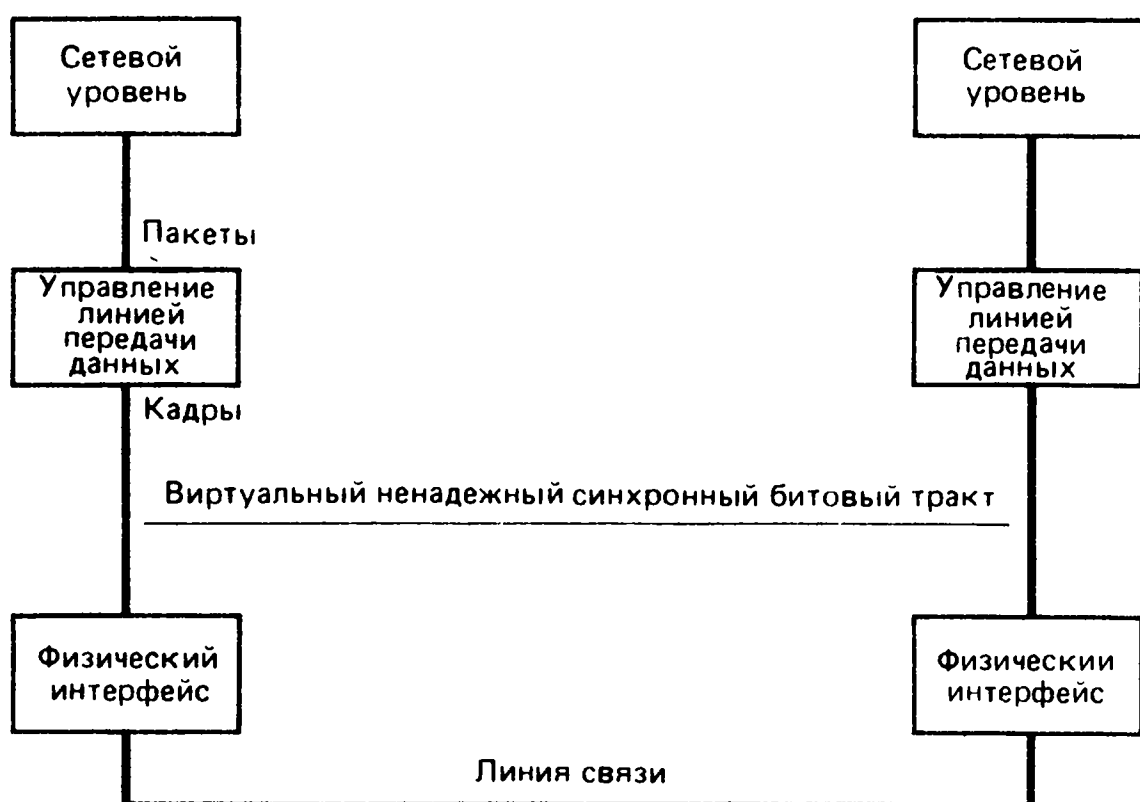


Рис. 2.1. Уровень управления линией передачи данных и интерфейсы со смежными уровнями.

Прерывисто-синхронный битовый тракт является другим типом битового тракта, по которому передающий модуль УЛПД поставляет синхронно биты модему, когда он (модуль) имеет данные для передачи, но ничего не поставляет, когда не имеет данных. Передающий модем не передает никаких сигналов во время этих холостых периодов, а принимающий модем обнаруживает холостые периоды и ничего не передает принимающему модулю УЛПД. Это несколько усложняет принимающий модем, поскольку он должен делать различие между 0 и 1 и холостым состоянием на каждом временном интервале и восстанавливать синхронизацию в конце холостого периода. В гл. 4 будет показано, что способность ничего не передавать очень важна для каналов с множественным доступом.

Последним типом виртуального битового тракта является *асинхронный символьный* тракт, когда биты одного и того же символа передаются с фиксированной скоростью, но последовательные символы поступают через интервалы, не равные друг другу, но не меньшие определенного минимума. Как будет видно в следующем разделе, этот тип тракта применяется только тогда, когда высокая скорость передачи не требуется.

В разд. 2.3—2.6 рассматривается уровень УЛПД, которому уделяется основное внимание в этой главе. На каждую линию, идущую от одной точки к другой, приходится два паритетных модуля УЛПД, по одному модулю на каждом конце. По отношению к трафику с заданным направлением передающий модуль УЛПД получает пакеты от модуля сетевого уровня этого же узла. Паритетные модули УЛПД используют распределенный алгоритм, или протокол, чтобы передать эти пакеты принимающему

модулю УЛПД, а следовательно, и модулю сетевого уровня этого же узла. В большинстве случаев цель состоит в том, чтобы доставлять пакеты в порядке их прихода, причем без повторений и ошибок. Выполняя эту задачу, модули УЛПД используют виртуальный битовый тракт (с ошибками), предоставляемый физическим уровнем.

Основной проблемой при достижении этой цели является исправление битовых ошибок, которые возникают в виртуальном битовом тракте. Как правило, эта цель достигается посредством метода, называемого ARQ (автоматический запрос на повторение). Согласно этому методу, ошибки сначала обнаруживаются принимающим модулем УЛПД, а затем на передающий модуль УЛПД посылается запрос на повторение. Как для обнаружения ошибок, так и для повторной передачи по запросам требуется передача по битовому тракту определенного количества дополнительной управляющей информации. Это обеспечивается добавлением определенного числа битов (называемых *заголовком*) к началу каждого пакета и дополнительного числа битов (называемых *трейлером*) к концу пакета (рис. 2.2). Пакет, к которому добавлен его заголовок и трейлер, называется *кадром*. С точки зрения уровня УЛПД пакет — это просто последовательность битов, переданная как единое целое сетевым уровнем; позднее будут встречаться примеры, в которых такие пакеты, поступающие от сетевого уровня, содержат информацию от множества различных сеансов, но это не имеет никакого значения для уровня УЛПД.

В разд. 2.3 исследуется проблема обнаружения ошибок и показывается, что помещенное в трейлере некоторое множество избыточных битов может использоваться для обнаружения ошибок. Затем в разд. 2.4 рассматриваются запросы на повторную передачу. Этот вопрос не такой простой, как кажется, во-первых, потому что запросы должны вставляться в поток данных, текущий в противоположном направлении, и, во-вторых, потому что противоположное направление также не свободно от ошибок. Здесь будет использоваться распределенный алгоритм, который представляет особый концептуальный интерес, поскольку он должен работать при наличии ошибок.

Затем в разд. 2.5 обсуждается формирование кадров. Рассматриваемый там вопрос состоит в том, что принимающий модуль УЛПД должен обнаруживать начало и конец каждого следующего кадра. В случае синхронного битового тракта биты кадра должны содержать информацию, позволяющую выделить конец кадра; кроме того, холостое заполнение между кадрами должно распознаваться однозначным способом. Эта проблема становится еще более интересной при наличии ошибок.

Широко признанным стандартом для управления линией передачи данных является протокол HDLC. Он обсуждается

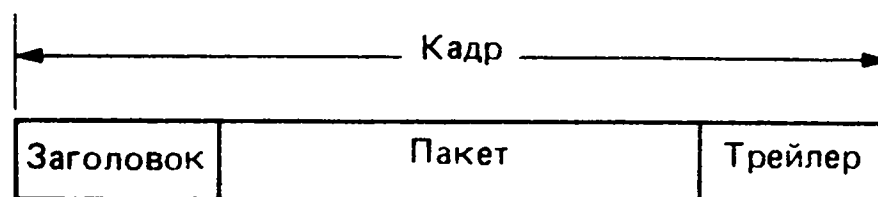


Рис. 2.2. Структура кадра. Пакет, идущий от сетевого уровня, дополняется на уровне УЛПД управляющими битами, которые обрамляют пакет с двух сторон.

в разд. 2.6. В этом разделе рассматриваются также некоторые дополнительные функции, которые должны выполняться на уровне УЛПД; среди них — инициирование и завершение работы каналов.

Темой обсуждения разд. 2.7 является адресация. Каждый пакет, движущийся по сети, должен содержать информацию, определяющую его пункт назначения; существует множество способов представления этой информации. Фактически этот вопрос относится к сетевому уровню, но способ адресации тесно связан с длиной кадров.

Наконец, в разд. 2.8 проводится краткое обсуждение того, как исправляются ошибки на сетевом и транспортном уровнях. В некоторых ситуациях (особенно там, где требуется очень малая задержка) нежелательно использовать ARQ на уровне УЛПД. Кроме того, с некоторой вероятностью (хотя и довольно малой) ошибки возникают в узлах подсети; использование ARQ при передаче по линиям не будет обеспечивать исправление этих ошибок. Наконец, существует вероятность того, что линии или узлы выйдут из строя полностью; при этом пакеты могут теряться. Во всех этих случаях за исправление ошибок могут нести ответственность сетевой или транспортный уровень; в разделе обсуждаются преимущества и недостатки каждого из таких вариантов.

2.2. Физический уровень. Каналы и модемы

Как отмечалось в последнем разделе, виртуальный канал, который используется на уровне УЛПД, обеспечивает передачу битов или символов от модуля УЛПД, находящегося на одном конце линии, к модулю на другом конце (см. рис. 2.1). В этом разделе дается обзор того, как каналы связи могут использоваться для выполнения такой передачи. Мы сосредоточим внимание на двухточечных каналах (т. е. каналах, которые соединяют только два узла) и отложим до гл. 4 рассмотрение каналов с множественным доступом (т. е. каналов, соединяющих более двух узлов). Мы также рассмотрим только передачу в одном направлении, не учитывая, таким образом, любое возможное взаимодействие между одновременными передачами в обоих направлениях.

Существуют два больших класса двухточечных каналов: цифровые и аналоговые. При модульном подходе цифровой канал

является просто битовым трактом с битовыми потоками на входе и выходе. Что касается аналогового канала, то на его вход поступает непрерывный сигнал (т. е. произвольная функция времени), и с его выхода также уходит непрерывный сигнал. Основное внимание здесь уделяется аналоговым каналам; цифровые каналы в равной степени важны для практики, но их лучше всего можно понять на основе изучения аналоговых каналов.

На входе аналогового канала должен находиться модуль, который преобразовывал бы цифровые данные, приходящие от модуля УЛПД, в аналоговые сигналы, посылаемые в канал. Аналогично «приемник» должен включать модуль, который преобразовывал бы обратно принятые сигналы в цифровые данные. Эти модули называются модемами (модулятор и демодулятор цифровых данных). Термин «модем» широко используется здесь; при этом необязательно подразумевается какая-либо модуляция, а просто указывается на требуемые операции преобразования.

Пусть функция времени $s(t)$ обозначает сигнал на входе аналогового канала; $s(t)$ может быть зависимостью напряжения или тока от времени. Аналогично пусть $r(t)$ обозначает зависимость напряжения или тока на выходе аналогового канала. Выход $r(t)$ является искаженной, задержанной во времени и претерпевшей затухание копией $s(t)$. Наша цель состоит в том, чтобы получить некоторое интуитивное представление о том, как установить соответствие между цифровыми данными и $s(t)$ так, чтобы минимизировать вредное воздействие этих искажений. Заметим, что здесь используется модульный подход к анализу физического канала, поскольку рассматривается связь входа с выходом, а не внутренние детали аналогового канала. Например, если обычный речевой телефонный канал используется как аналоговый канал, то физически он, как правило, включает многочисленные коммутаторы, устройства уплотнения, устройства разделения, модуляторы и демодуляторы. Для коммутируемых линий этот канал будет меняться при каждом последующем вызове, хотя требования на допустимые отклонения характеристик, описывающих связь входа с выходом, остаются неизменными.

2.2.1. Фильтрация

Одним из наиболее важных искажающих воздействий, присущих аналоговым каналам, является линейная фильтрация с не зависящими от времени параметрами. Фильтрация возникает не только из-за фильтров, вводимых проектировщиком канала; она также обуславливается внутренними свойствами среды распространения. Одним из следствий фильтрации является сглаживание переданного сигнала $s(t)$. На рис. 2.3 показаны два примера, в одном из которых $s(t)$ является единичным прямоугольным

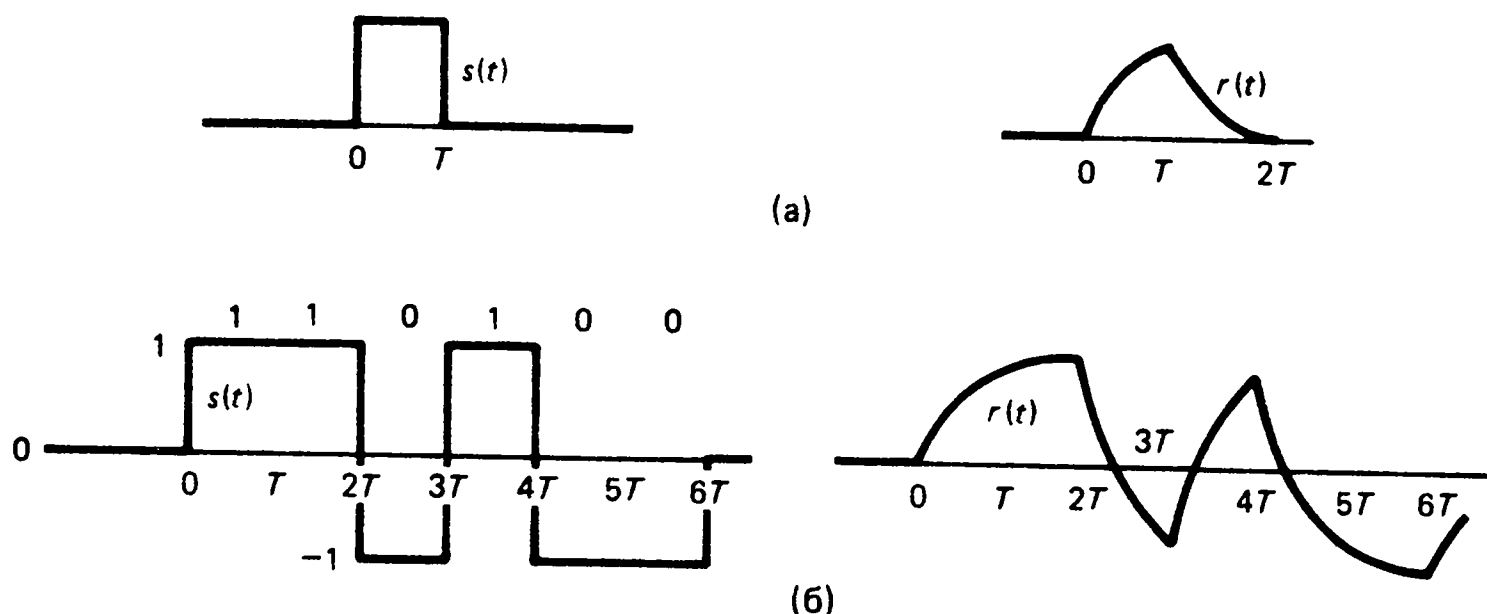


Рис. 2.3. Соотношение между входным и выходным сигналами канала связи с фильтрацией.

a — отклик $r(t)$ на вход $s(t)$, являющийся прямоугольным импульсом; b — отклик на последовательность импульсов. Здесь также иллюстрируется NRZ-код, при котором последовательности двоичных входных сигналов (110100) ставится в соответствие последовательность прямоугольных импульсов. Длительность каждого импульса равна времени между поступлениями двоичных входных сигналов.

импульсом, а в другом — последовательностью прямоугольных импульсов. Линейные инвариантные во времени фильтры определяются следующими свойствами: 1) если входной сигнал $s(t)$ порождает выходной сигнал $r(t)$, то для любого τ входной сигнал $s(t - \tau)$ порождает выходной сигнал $r(t - \tau)$; 2) если $s(t)$ порождает $r(t)$, то для любого действительного числа α входной сигнал $\alpha s(t)$ порождает выходной сигнал $\alpha r(t)$; 3) если $s_1(t)$ порождает $r_1(t)$ и $s_2(t)$ порождает $r_2(t)$, то $s_1(t) + s_2(t)$ порождает $r_1(t) + r_2(t)$.

Первое из этих свойств выражает временную инвариантность, а именно если входной сигнал задержан на время τ , то соответствующий выходной сигнал остается таким же, за исключением того, что он задерживается на время τ . Второе и третье свойства выражают линейность, т. е. если вход изменяется в α раз, то соответствующий выход остается таким же, за исключением того, что он изменяется в α раз. Кроме того, выходной сигнал, порожденный суммой двух входных сигналов, равен сумме соответствующих выходных сигналов. Эти свойства позволяют, например, рассчитать выходной сигнал на рис. 2.3, б, если известен выходной сигнал, приведенный на рис. 2.3, а. В частности, отклик на прямоугольный импульс с отрицательной амплитудой отличается только знаком от отклика на импульс с положительной амплитудой (свойство 2), откликом задержанного импульса является задержанный исходный выход (свойство 1) и выходом, порожденным суммой импульсов, является сумма выходов, порожденных исходными импульсами (свойство 3).

На рис. 2.3 также показан простой способ установления соответствия между входящими битами и формой аналогового сигнала.

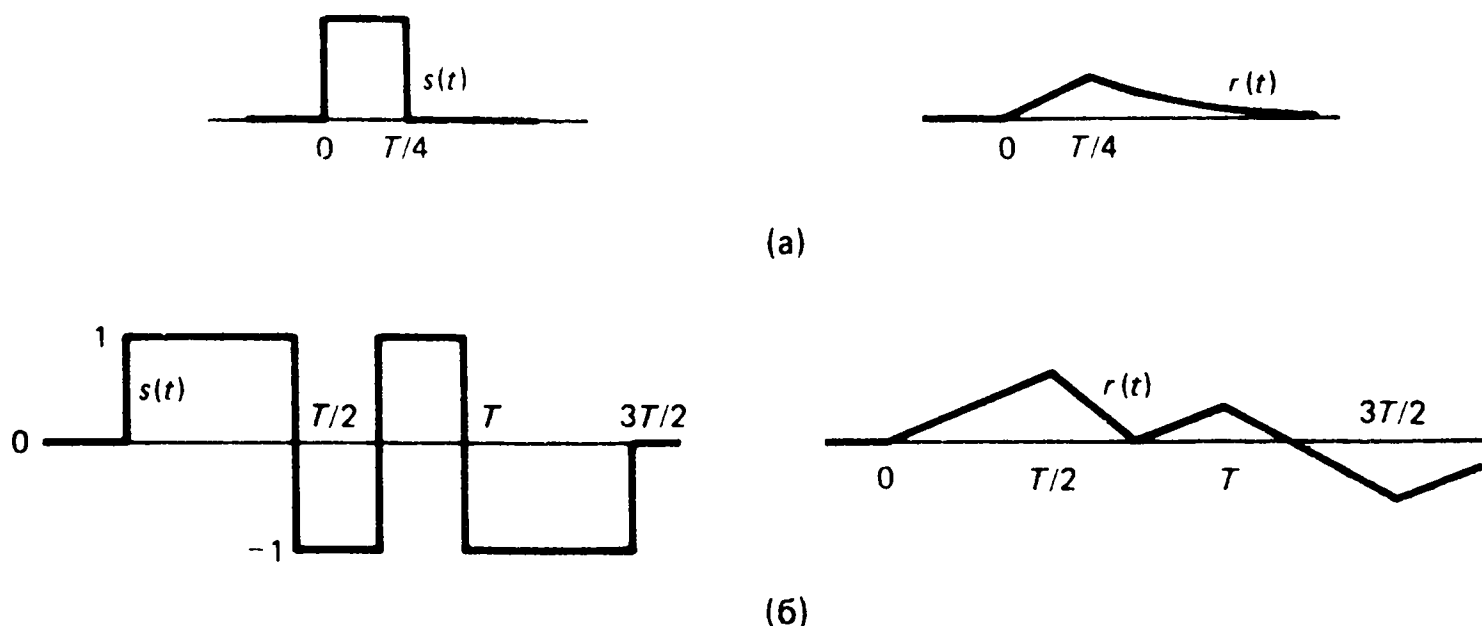


Рис. 2.4. Соотношение между входным и выходным сигналами канала, показанного на рис. 2.3. Двоичные цифры здесь поступают в четыре раза быстрее, чем на рис. 2.3, и прямоугольные импульсы в четыре раза короче. Заметим, что выходной сигнал $r(t)$ в большей степени искажен и ослаблен, чем на рис. 2.3.

Виртуальный канал получает новый бит от модуля УЛПД каждые T секунд; биту со значением 1 ставится в соответствие прямоугольный импульс с амплитудой $+1$, а биту со значением 0 — импульс с амплитудой -1 . Таким образом, на рис. 2.3, б представлен сигнал $s(t)$ для последовательности данных 110100. Этот метод соответствия часто называется кодом без возврата к нулю (кодом NRZ). Обозначение NRZ возникло из-за наличия альтернативного метода, при котором импульсы сигнала $s(t)$ по длительности меньше времени между поступлениями битов T , в результате чего между импульсами сигнал $s(t)$ возвращается к нулю на некоторое время. Относительные преимущества этих методов станут ясными в дальнейшем.

Теперь рассмотрим изменение скорости поступления битов на вход виртуального канала. На рис. 2.4 показан результат увеличения скорости в четыре раза (т. е. период сигнала уменьшается от T до $T/4$). Можно заметить, что выходной сигнал $r(t)$ искажается в большей степени, чем раньше. Проблема состоит в том, что отклик импульса длится много дольше, чем сам импульс, так что выходной сигнал в заданный момент времени t зависит в значительной степени от полярности нескольких входных импульсов; это явление называется *межсимвольной интерференцией*.

Приняв более общую точку зрения, предположим, что $h(t)$ является выходным сигналом, соответствующим бесконечно узкому импульсу единичной площади в момент времени 0. Функция $h(t)$ называется импульсной характеристикой канала. Представим произвольный входной сигнал $s(t)$ в виде суперпозиции очень узких импульсов, как показано на рис. 2.5. Часть сигнала от $s(\tau)$ до $s(\tau + \delta)$ можно рассматривать как поступивший в момент τ узкий импульс площадью $\delta s(\tau)$; это приводит к появлению на выходе $\delta s(\tau) h(t - \tau)$ в момент t . Сложение откликов в момент t

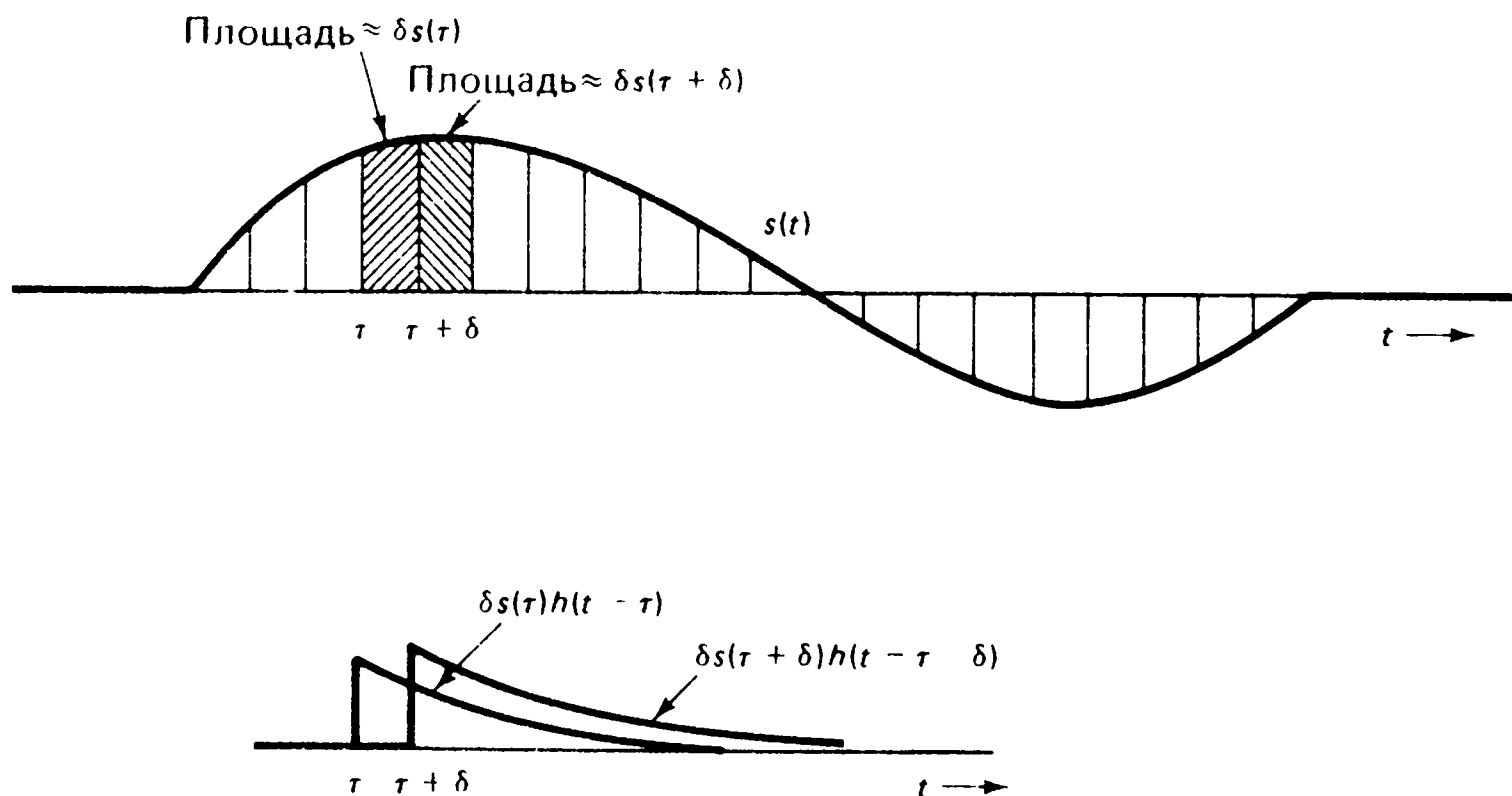


Рис. 2.5. Графическая интерпретация уравнения свертки. Входной сигнал $s(t)$ рассматривается как суперпозиция узких импульсов шириной δ . Каждый такой импульс порождает на выходе $\delta s(\tau) h(t - \tau)$. Полный выходной сигнал является суммой этих откликов на импульсы.

на каждый из этих входных импульсов и переход к пределу $\delta \rightarrow 0$ показывают, что

$$r(t) = \int_{-\infty}^{+\infty} s(\tau) h(t - \tau) d\tau. \quad (2.1)$$

Эта формула называется *интегралом свертки*, а $r(t)$ является сверткой $s(t)$ и $h(t)$. Заметим, что из этой формулы следует, что фильтрующие свойства канала полностью определяются импульсной характеристикой $h(t)$; если $h(t)$ известна, то $r(t)$ для любого $s(t)$ может быть вычислена. В случае примера на рис. 2.3 предполагалось, что $h(t)$ равна 0 при $t < 0$ и $\alpha e^{-\alpha t}$ при $t \geq 0$, где $\alpha = 2/T$. Зная это, легко вычислить отклики, показанные на рисунке.

Заметим, что, согласно (2.1), выходной сигнал в заданный момент времени t существенно зависит от входного сигнала $s(\tau)$ в интервале, где $h(t - \tau)$ заметно отличается от нуля. Это означает, что если этот интервал много больше периода сигнала T , т. е. времени между последовательными входными импульсами, то будет возникать значительная межсимвольная интерференция.

На выходе канала не может появляться отклик на входной сигнал до тех пор, пока входной сигнал не будет подан, поэтому $h(t)$ должна быть равна 0 при $t < 0$; таким образом, верхний предел интегрирования в (2.1) можно положить равным t . Однако часто полезно использовать различные отсчеты времени в приемнике и в передатчике, что скрывает эффект задержки распространения. В этом случае $h(t)$ может быть ненулевой при $t < 0$, что является основанием бесконечного верхнего предела интегрирования.

2.2.2. Передаточная характеристика

Чтобы получить более полное представление о фильтрации, необходимо наряду с временной областью рассматривать частотную область; другими словами, желательно определить действие фильтрации на синусоиды с различными частотами. Аналитически удобно рассмотреть здесь более общую ситуацию и считать, что входной сигнал канала является комплексной функцией t ; т. е. $s(t) = \operatorname{Re} [s(t)] + j \operatorname{Im} [s(t)]$, где $j = \sqrt{-1}$. Фактический входной сигнал канала, конечно, является всегда действительной функцией. Однако, если сигнал $s(t)$ может быть комплексным в (2.1), $r(t)$ тоже будет комплексным, но выходной сигнал, соответствующий $\operatorname{Re} [s(t)]$, равен просто $\operatorname{Re} [r(t)]$ (в предположении, что $h(t)$ — действительная), а выходной сигнал, соответствующий $\operatorname{Im} [s(t)]$, равен $\operatorname{Im} [r(t)]$. При заданной частоте f положим, что $s(\tau)$ в (2.1) является комплексной синусоидой $e^{j2\pi f\tau} = \cos(2\pi f\tau) + j \sin(2\pi f\tau)$. Интегрируя (2.1) (см. задачу 2.3), получаем

$$r(t) = H(f) e^{j2\pi ft}, \quad (2.2)$$

где

$$H(f) = \int_{-\infty}^{\infty} h(\tau) e^{-j2\pi f\tau} d\tau. \quad (2.3)$$

Таким образом, откликом на комплексную синусоиду с частотой f является комплексная синусоида той же частоты, умноженная на коэффициент $H(f)$. $H(f)$ есть комплексная функция частоты f , называемая *передаточной характеристикой* канала. Она определена как для положительных, так и для отрицательных значений f . Пусть $|H(f)|$ обозначает модуль, а $\angle H(f)$ — фазу $H(f)$ (т. е. $H(f) = |H(f)| e^{j\angle H(f)}$). Отклик $r_1(t)$ на действительную синусоиду $\cos(2\pi ft)$ дается действительной частью (2.2), или

$$r_1(t) = |H(f)| \cos[2\pi ft + \angle H(f)]. \quad (2.4)$$

Таким образом, действительная синусоида с частотой f на входе приводит к появлению на выходе действительной синусоиды с той же частотой; $|H(f)|$ обозначает амплитуду, а $\angle H(f)$ — фазу этого выходного сигнала по отношению к фазе входного сигнала.

Рассматривая пример передаточной характеристики, положим $h(t) = \alpha e^{-\alpha t}$ при $t \geq 0$; тогда интегрирование в (2.3) дает

$$H(f) = \alpha / (\alpha + j2\pi f). \quad (2.5)$$

Соотношение (2.3) отображает произвольную функцию времени $h(t)$ в частотную функцию $H(f)$; математически $H(f)$ является *преобразованием Фурье* функции $h(t)$. Можно показать, что

функцию времени $h(t)$ можно восстановить, зная $H(f)$, с помощью обратного преобразования Фурье

$$h(t) = \int_{-\infty}^{\infty} H(f) e^{j2\pi ft} df. \quad (2.6)$$

Равенство (2.6) имеет интересную интерпретацию; оно показывает, что, по существу, произвольная функция времени $h(t)$ может быть представлена в виде суперпозиции множества комплексных синусоид, при этом вес каждой синусоиды равен $H(f)$, определяемой равенством (2.3). Таким образом, входной сигнал $s(t)$ можно представить (по крайней мере на любом конечном интервале) с помощью частотной функции

$$S(f) = \int_{-\infty}^{\infty} s(t) e^{-j2\pi ft} dt, \quad (2.7)$$

$$s(t) = \int_{-\infty}^{\infty} S(f) e^{j2\pi ft} df. \quad (2.8)$$

Выходной сигнал канала $r(t)$ и его частотная функция $R(f)$ связаны аналогичным образом. Наконец, поскольку соотношение (2.2) определяет выходной сигнал для единичной комплексной синусоиды с частотой f и $s(t)$ является суперпозицией комплексных синусоид, как показано в (2.8), то из линейности канала следует, что отклик на $s(t)$ равен

$$r(t) = \int_{-\infty}^{\infty} H(f) S(f) e^{j2\pi ft} df. \quad (2.9)$$

Так как $r(t)$ является обратным преобразованием Фурье $R(f)$, то частотная связь входа и выхода выражается простым равенством

$$R(f) = H(f) S(f). \quad (2.10)$$

Таким образом, свертке $h(t)$ и $s(t)$ во временной области соответствует умножение $H(f)$ и $S(f)$ в частотной области. Из равенства (2.10) видно, почему спектральный подход способствует пониманию фильтрации.

Если $H(f) = 1$ в некотором диапазоне частот и $S(f)$ отлична от нуля только в этом диапазоне, то $R(f) = S(f)$, так что $r(t) = s(t)$. Обычно не удастся достичь выполнения $H(f) = 1$ в требуемом диапазоне частот, однако можно пропустить принятый сигнал через дополнительный фильтр с передаточной характеристикой $H^{-1}(f)$ в требуемом диапазоне; эта дополнительная фильтрация, естественно, приводит к тому, что окончательный выходной сигнал удовлетворяет равенству $r(t) = s(t)$. На практике можно получить хорошее приближение к характеристикам таких фильтров при условии, что допускается дополнительная задержка,

т. е. при некоторой задержке τ выполняется равенство $r(t) = s(t - \tau)$. Такие фильтры могут быть построены так, что они автоматически адаптируются к фактической характеристике канала $H(f)$. Подобные фильтры обычно создаются на цифровой основе, на их вход подаются отсчеты выходного сигнала канала. Они называются *адаптивными компенсаторами*. Эти фильтры могут применяться для подавления межсимвольной интерференции.

Теперь возникает вопрос о том, какой диапазон частот должен использоваться сигналом $s(t)$; на первый взгляд кажется, что может быть взят любой диапазон частот, где $H(f)$ отлична от нуля. Недостаток этого предложения состоит в том, что не учитывается всегда присутствующий в канале аддитивный шум. Шум добавляется к сигналу в различных местах на пути передачи, в том числе и в приемнике. Таким образом, шум не подвергается той же фильтрации в канале, что и сигнал. Если $|H(f)|$ очень мал на некотором интервале частот, то сигнал сильно затухает на этих частотах, но шум, как правило, не затухает. Таким образом, если принятый сигнал подается в приемнике на фильтр с характеристикой $H^{-1}(f)$, сигнал восстанавливается до своего первоначального уровня, но шум значительно усиливается.

Из этого следует (в предположении, что шум равномерно распределен в полосе частот), что отображение цифровых данных в $s(t)$ должно быть таким, чтобы $|S(f)|$ имел большое значение на тех частотах, где $|H(f)|$ велико, и $|S(f)|$ имел малое значение (в идеале равное нулю) на всех других частотах. Граница, разделяющая понятия «большое и малое значения», зависит от уровня шума и уровня сигнала и не очень важна здесь, в частности, потому, что приведенное рассуждение является качественным и сильно упрощенным. Возвращаясь к примеру, в котором $h(t) = \alpha e^{-\alpha t}$ при $t \geq 0$, напомним, что соответствующая передаточная характеристика получена в (2.5), $H(f) = \alpha / (\alpha + j2\pi f)$. Видно, что $|H(f)|$ равен приближенно 1 при малых f и уменьшается как $1/f$ при больших f . Мы не будем пытаться вычислить $S(f)$ для NRZ-кода, показанного на рис. 2.3, отчасти потому, что $s(t)$ и $S(f)$ зависят от конкретной кодируемой последовательности, но легко найти, как влияет на $s(t)$ и $S(f)$ период сигнала T . Если T уменьшается, то $s(t)$ сжимается во времени, и это приводит соответственно к расширению $S(f)$ по частоте (см. задачу 2.5). В этом случае требуется компенсация, чтобы либо выравнять $H(f)$ на более широкой полосе частот, увеличивая таким образом шум, либо допустить увеличение межсимвольной интерференции.

2.2.3. Теорема отсчетов

Более глубоко рассмотреть вопрос о скоростях передачи сигналов позволяет теорема отсчетов. Эта теорема утверждает, что если сигнал $s(t)$ имеет спектр, ограниченный частотой W (т. е.

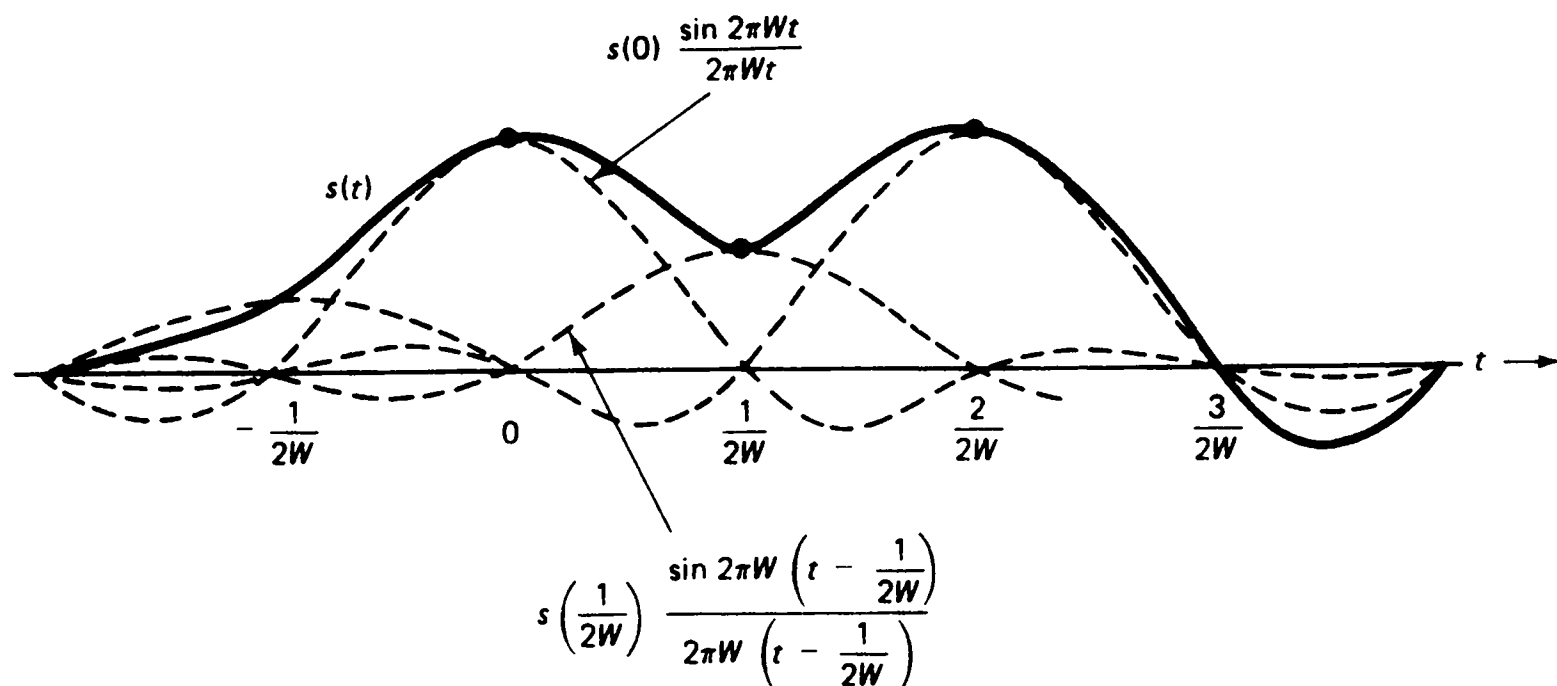


Рис. 2.6. Теорема отсчетов, примененная к функции $s(t)$, спектр которой ограничен частотой W . Функция представлена в виде суперпозиции функций $(\sin x)/x$. Для каждого отсчета имеется одна такая функция с центром в точке отсчета и множителем, равным значению отсчета.

$S(f) = 0$ при $|f| > W$, то (в предположении, что $S(f)$ не имеет скачка в точке $f = W$) $s(t)$ полностью определяется своими значениями в моменты, отстоящие друг от друга на $1/(2W)$ секунд; в частности,

$$s(t) = \sum_{i=-\infty}^{\infty} s[i/(2W)] \frac{\sin [2\pi W (t - i/(2W))]}{2\pi W [t - i/(2W)]}. \quad (2.11)$$

Кроме того, для любого набора значений отсчетов, взятых в точках, следующих через интервалы длиной $1/(2W)$, существует сигнал с ограниченным спектром, определяемый равенством (2.11), который совпадает с этими отсчетами в указанных точках. На рис. 2.6 иллюстрируется этот результат.

Следствие этого результата, связанное с нашими задачами, состоит в том, что входящим цифровым данным можно сопоставить отсчеты через $1/(2W)$ секунд и использовать их для получения соответствующего сигнала, который ограничен по частоте $|f| \leq W$. Если этот сигнал пропускается затем через идеальный фильтр нижних частот, у которого $H(f) = 1$ при $|f| \leq W$ и $H(f) = 0$ для всех остальных f , то принятый сигнал будет идентичен переданному (при отсутствии шумов); таким образом, его отсчеты могут использоваться для восстановления исходных цифровых данных.

Можно считать, что NRZ-код сопоставляет входящим битам отсчеты сигнала $s(t)$, но отсчеты передаются в виде прямоугольных импульсов, а не идеальных импульсов с формой $(\sin x)/x$, как это определяется в (2.11). В известном смысле форма им-

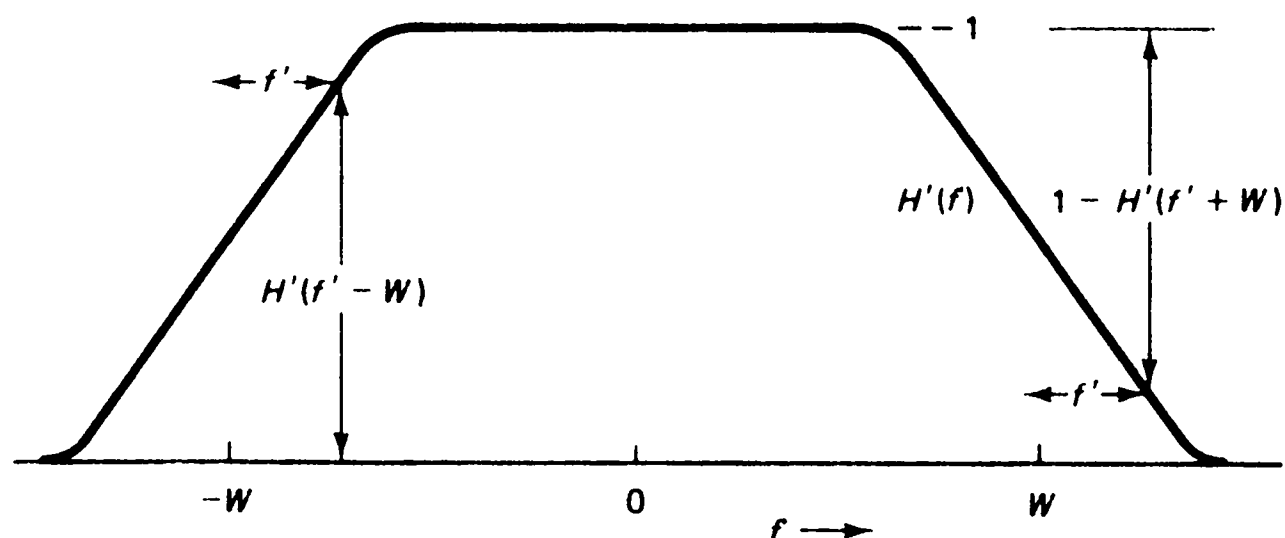


Рис. 2.7. Передаточная характеристика $H'(f)$, которая удовлетворяет критерию отсутствия межсимвольной интерференции Найквиста. Заметим, что характеристика имеет нечетную симметрию в окрестности точки $f = W$.

пульса, используемая в передатчике, не имеет решающего значения, так как можно считать, что форма импульса создается посредством фильтрации импульсов отсчетов. Если последовательность, состоящая из фильтра, формирующего импульсы на входе канала, канального фильтра и выравнивающего фильтра, дает на выходе сигнал типа $(\sin x)/x$, как в (2.11) (или, что эквивалентно, полная передаточная характеристика совпадает с характеристикой идеального фильтра нижних частот), то отсчеты на выходе будут воспроизводить отсчеты на входе.

Идеальный фильтр нижних частот и форма импульса $(\sin x)/x$, соответствующая теореме отсчетов, удобны в теоретическом отношении. На практике более реалистичен фильтр, у которого отсчеты на выходе копируют значения отсчетов на входе (т. е. чтобы отсутствовала межсимвольная интерференция), а высокочастотный шум не пропускается.

Изящное решение этой проблемы было найдено в классической работе Найквиста [185]. Он показал, что удастся избежать межсимвольной интерференции, используя фильтр $H'(f)$ с нечетной симметрией на границе полосы частот; т. е. $H'(f+W) = 1 - H'(f-W)$ при $|f| \leq W$ и $H'(f) = 0$ при $|f| > 2W$ (рис. 2.7). Фильтр $H'(f)$ является составным; он состоит из фильтра, формирующего импульсы в передатчике, канального фильтра и выравнивающего фильтра. На практике такие фильтры имеют довольно круто убывающую характеристику в окрестности $f = W$ для подавления высокочастотного шума, а не идеальную ступеньку, и, таким образом, допускается значительная гибкость при проектировании.

Важно понимать, что теорема отсчетов определяет число отсчетов в секунду, которое может быть передано по низкочастотному каналу, но она не определяет число битов, содержащихся в одном отсчете. Например, двухбитовые отсчеты можно полу-

чить, сопоставляя $11 \rightarrow 3$, $10 \rightarrow 1$, $00 \rightarrow -1$ и $01 \rightarrow -3$. Как покажет дальнейшее рассмотрение, число бит в отсчете ограничивает шум.

2.2.4. Полосовые каналы

До сих пор рассматривались низкочастотные каналы, у которых значение $|H(f)|$ велико при малых f . Большинство физических каналов не попадает в эту категорию, а обладает тем свойством, что $|H(f)|$ значительно отличен от нуля только в некоторой частотной полосе $f_1 \leq |f| \leq f_2$, где $f_1 > 0$. Такие каналы называются полосовыми, причем многие из них имеют $H(0) = 0$. Говорят, что канал или сигнал не имеют постоянной составляющей, если $H(0) = 0$. Используя (2.3) нетрудно понять, что из

этого следует $\int_{-\infty}^{\infty} h(t) dt = 0$. Импульсная характеристика таких каналов колеблется около 0, как показано на рис. 2.8; это явление часто называют «звоном». Такое поведение импульсной характеристики наводит на мысль, что код NRZ не очень многообещающий для передачи по полосовому каналу.

Чтобы преодолеть эти трудности, большинство модемов для полосовых каналов или непосредственно кодирует цифровые данные в сигналы без постоянной составляющей или использует методы модуляции. Наиболее известным непосредственным кодированием такого типа является кодирование, использующее манчестерский код (рис. 2.9). Как показано на рисунке, у каждого из сигналов, используемых для представления 0 или 1, нет постоянной составляющей, а в центре каждого периода символа имеется перепад (или от 1 до -1 , или от -1 до 1). Эти перепады упрощают восстановление синхронизации в приемнике (заметим, что при использовании NRZ-кода восстановление синхронизации может затрудняться на длинных участках из одних 0 или 1 даже при отсутствии фильтрации и шума). Расплатой за такое удаление постоянной составляющей является использование намного более широкой, чем требуется, полосы частот. Манчестерский код широко используется на практике, особенно в системе Ethernet и соответствующем стандарте IEEE 802.4, рассматриваемым

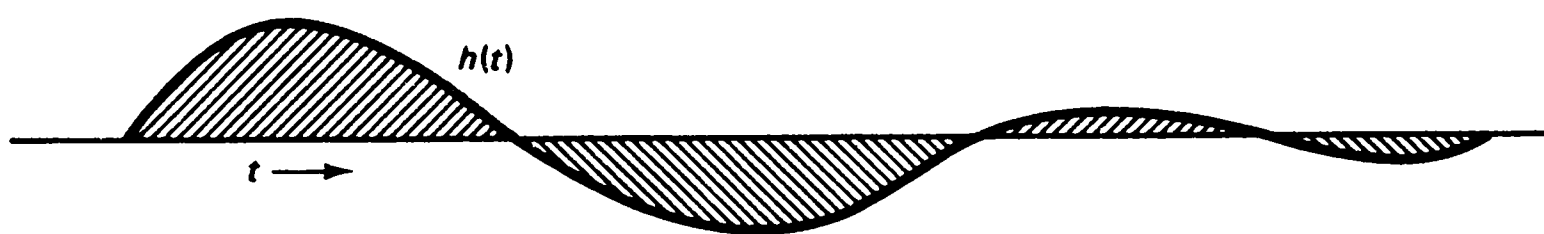


Рис. 2.8. Импульсная характеристика $h(t)$, для которой $H(f) = 0$ при $f = 0$. Заметим, что площадь под функцией, где $h(t)$ положительна, равна площади, где она отрицательна.

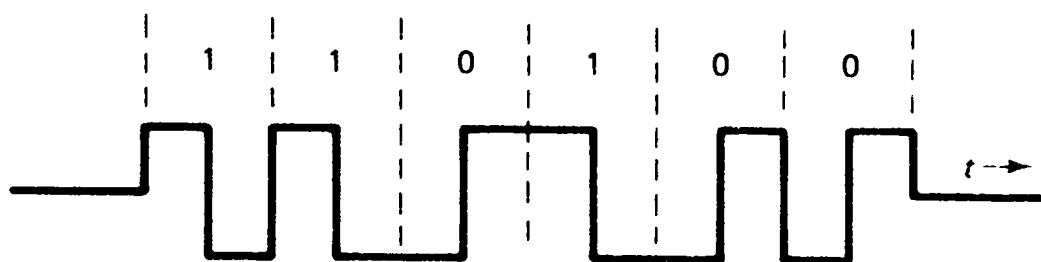


Рис. 2.9. Манчестерский код. Двоичному числу 1 соответствует положительный импульс, за которым следует отрицательный импульс, а двоичному числу 0 соответствует отрицательный импульс, за которым следует положительный импульс. Отметим наличие перепада в центре каждого периода сигнала.

в разд. 4.5 гл. 4. Существует много других непосредственных методов кодирования; все они в определенной степени предназначены для конкретных применений, имеют различные частотные характеристики и не используют постоянную составляющую.

2.2.5. Модуляция

Одним из простейших методов модуляции является амплитудная модуляция (АМ). При ней сигнал $s(t)$ (называемый модулирующим) получается, как и ранее, из цифровых данных, например с помощью NRZ-кода. Затем он умножается на синусоидальную несущую, например $\cos(2\pi f_0 t)$, для получения модулированного сигнала $s(t) \cos(2\pi f_0 t)$. В задаче 2.6 показывается, что спектральное представление этого модулированного сигнала дается выражением $[S(f - f_0) + S(f + f_0)]/2$ (рис. 2.10). В приемнике модулированный сигнал снова умножается на $\cos(2\pi f_0 t)$; в результате получается сигнал

$$r(t) = s(t) \cos^2(2\pi f_0 t) = s(t)/2 + s(t) \cos(4\pi f_0 t)/2. \quad (2.12)$$

Высокочастотная составляющая с частотой $2f_0$ затем отфильтровывается в приемнике, и остается демодулированный сигнал $s(t)/2$, который затем преобразуется в цифровые данные. На практике поступающие биты преобразуются в более короткие

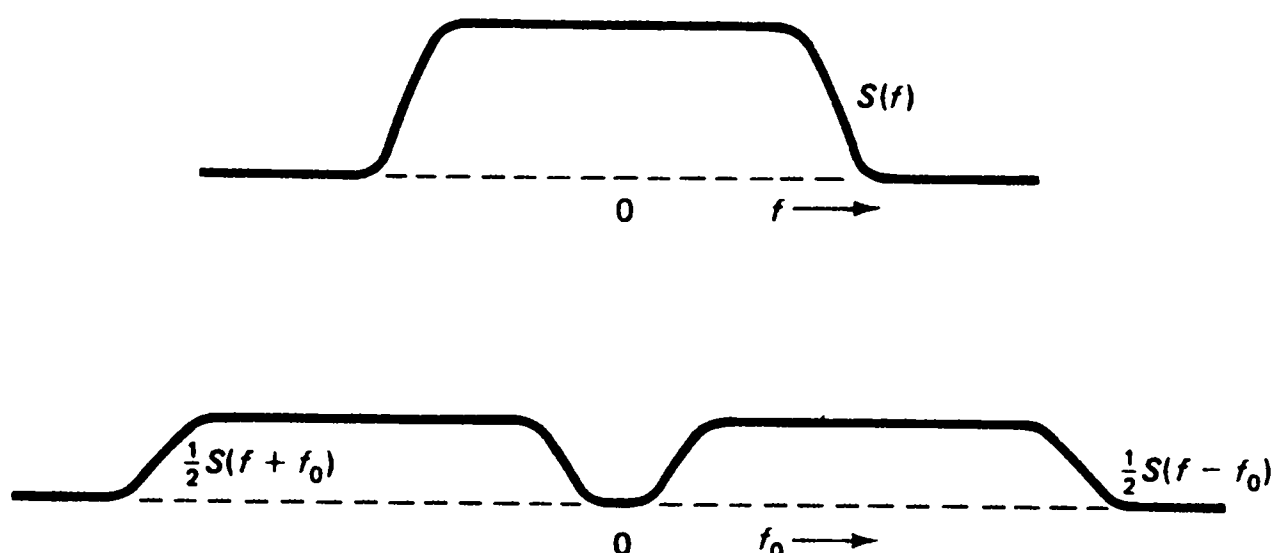


Рис. 2.10. Амплитудная модуляция. Частотная характеристика сигнала $s(t)$ сдвигается вверх и вниз по частоте на f_0 .

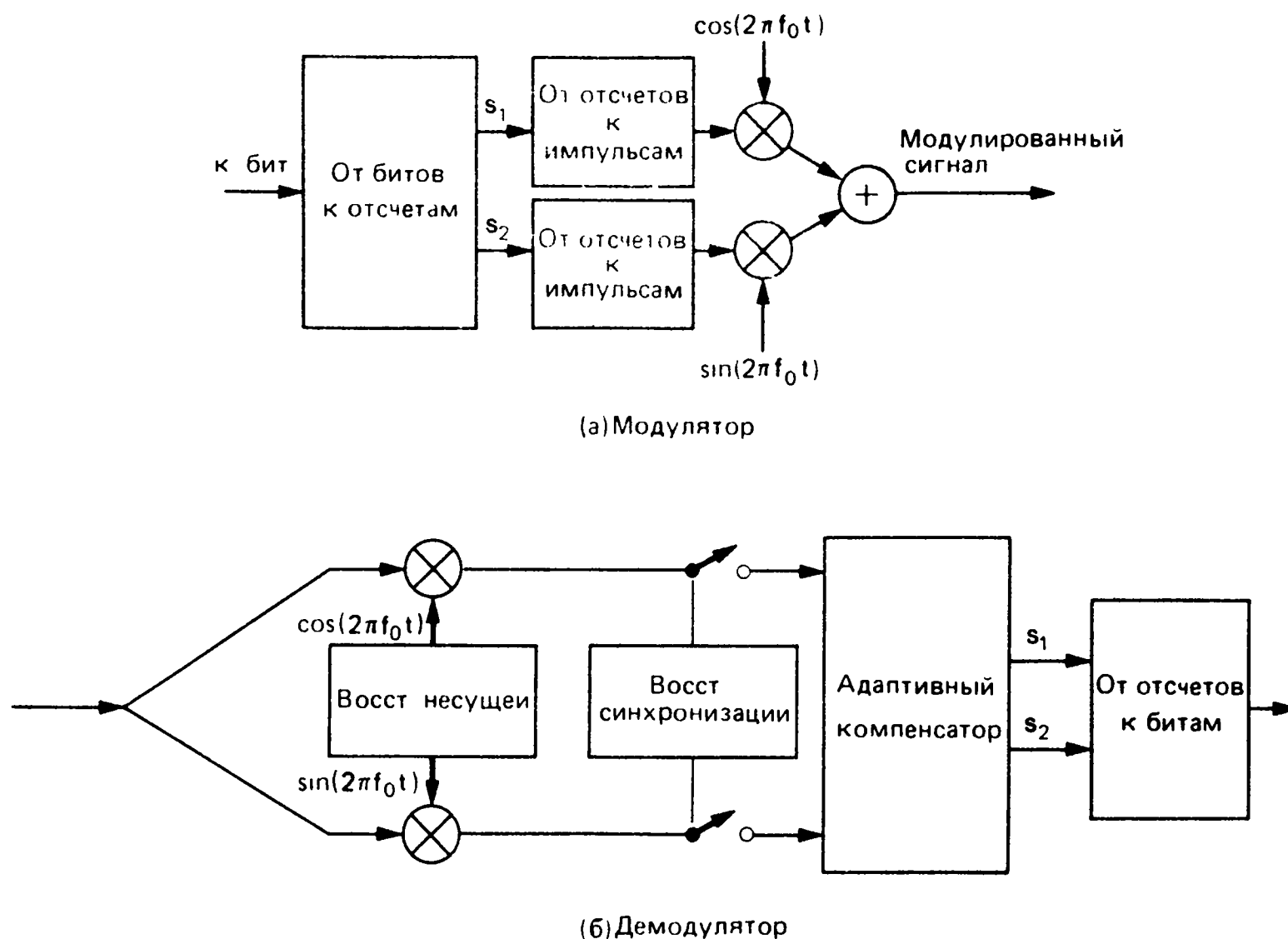


Рис. 2.11. Квадратурная амплитудная модуляция.

a — в течение каждого периода отсчета в модулятор поступают k бит, которые преобразуются в квадратурные амплитуды, модулируют функции синус и косинус соответственно, а затем суммируются; b — обратные операции совершаются в демодуляторе.

импульсы, чем те, что используются при NRZ-коде, и затем фильтруются, чтобы удалить до некоторой степени высокочастотные компоненты. Интересно отметить, что кодирование манчестерским кодом можно рассматривать как амплитудную модуляцию (АМ), при которой NRZ-код умножается на меандр. Хотя меандр не является синусоидой, он может быть представлен в виде комбинации синусоиды и нечетных гармоник; гармоники не выполняют никакой полезной функции и обычно частично отфильтровываются каналом.

АМ довольно чувствительна к тому, насколько точно в приемнике известна фаза несущей. Например, если модулированный сигнал $s(t) \cos(2\pi f_0 t)$ умножался бы на $\sin(2\pi f_0 t)$, то низкочастотный демодулированный сигнал полностью исчезал. Вместе с тем это подсказывает возможность передачи данных с удвоенной скоростью посредством метода, известного как квадратурная амплитудная модуляция (КАМ). В этом случае поступающие биты преобразуются в два модулирующих сигнала $s_1(t)$ и $s_2(t)$. Затем $s_1(t)$ умножается на $\cos(2\pi f_0 t)$, а $s_2(t)$ — на $\sin(2\pi f_0 t)$; сумма этих произведений образует передаваемый КАМ сигнал (рис. 2.11). Принятый сигнал умножается отдельно на $\cos(2\pi f_0 t)$

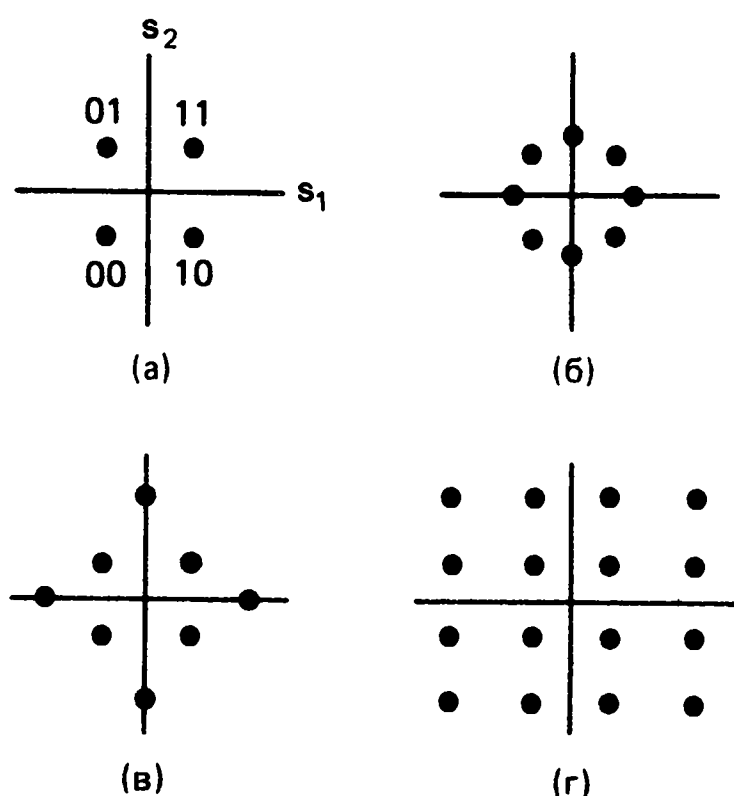


Рис. 2.12. Сигнальные диаграммы для КАМ.

a — отображение двух двоичных цифр в отсчет квадратурных амплитуд; *б* и *в* — отображения трех двоичных цифр; *г* — отображение четырех двоичных цифр. Диаграммы *a* и *б* можно также рассматривать как фазовую манипуляцию.

и $\sin(2\pi f_0 t)$. Первое произведение (после отфильтровывания высокочастотной составляющей) дает $s_1(t)/2$, а второе — $s_2(t)/2$.

КАМ широко используется в высокочастотных модемах для речевых телефонных каналов. Эти каналы имеют полезную полосу частот примерно от 500 до 2900 Гц; несущая частота находится обычно в окрестности 1700 Гц. Сигналы $s_1(t)$ и $s_2(t)$ обычно имеют низкочастотный спектр, ограниченный частотой 1200 Гц, что позволяет иметь 2400 отсчетов в секунду для каждого модулирующего сигнала. В приемнике обычно применяется адаптивное выравнивание (до или после демодуляции) для компенсации изменений передаточной характеристики канала.

Следующий вопрос состоит в том, как преобразовать поступающие биты в низкочастотные колебания $s_1(t)$ и $s_2(t)$. Форма импульса, используемая при преобразовании, не имеет первостепенного значения; в любом случае на нее будет воздействовать фильтрация и при подавлении межсимвольной интерференции выравнивающая цепь автоматически восстановит форму импульса. Таким образом, важным является то, что поступающие биты отображаются в амплитуды отсчетов $s_1(t)$ и $s_2(t)$. Можно отобразить один бит в отсчет $s_1(t)$, преобразуя 1 в $+1$ и 0 в -1 , и аналогично второй бит — в отсчет $s_2(t)$. Это показано на рис. 2.12, *a*, представляющем отображение двух битов в две амплитуды. Аналогично при любом заданном целом числе k можно отобразить k бит в две амплитуды. Каждая из 2^k комбинаций из k бит отображается в свою пару амплитуд. Это множество из 2^k пар амплитуд, которое возникает при отображении, называется сигнальной диаграммой (см. рис. 2.12, где показан ряд примеров). Первые две диаграммы на рис. 2.12 называются также фазовой манипуляцией (PSK), поскольку можно считать, что в этих случаях просто меняется фаза несущей, а не ее амплитуда. У модемов, предназначенных для речевых каналов, $W = 2400$ Гц, и $k = 2$ (как на рис. 2.12, *a*) дает 4800 бит/с, тогда как $k = 4$ (как на рис. 2.12, *г*) дает 9600 бит/с.

Чтобы уменьшить вероятность того, что из-за шума какая-либо точка диаграммы ошибочно принимается за другую, жела-

тельно размещать сигнальные точки по возможности дальше друг от друга, соблюдая при этом ограничение на мощность сигнала (т. е. на среднеквадратичное расстояние от центра до точки диаграммы, вычисленное по всем точкам). Можно показать, что при $k = 3$ диаграмма на рис. 2.12, в имеет преимущество в этом смысле перед показанной на рис. 2.12, б.

Существуют также модемы для речевых каналов, имеющие скорости передачи 14 400 и 19 200 бит/с. В этих модемах используются сигнальные диаграммы с соответственно бóльшим числом точек и коды, исправляющие ошибки. Уместно спросить в данный момент, какой предел накладывается на скорость передачи данных из-за ограничения на полосу пропускания канала и из-за его шума. Шеннон [215] показал, что пропускная способность (т. е. максимальная безошибочная скорость передачи данных в бит/с) такого канала определяется выражением

$$C = W \log_2 [1 + S/(N_0 W)], \quad (2.13)$$

где W — ширина доступной полосы пропускания канала (2400 Гц для речевого телефона), S — допустимая мощность сигнала на входе приемника и $N_0 W$ — мощность шума в полосе частот с шириной W , которая поступает в приемник (т. е. N_0 — это мощность шума в единичной полосе частот при условии равномерного распределения на W).

Это глубокий и тонкий результат; здесь не был развит аппарат, необходимый для его точной формулировки, не говоря уже о доказательстве его справедливости. Однако описание КАМ дает по крайней мере некоторое интуитивное обоснование формы результата. Скорость следования отсчетов $1/T$ равна W по теореме отсчетов (вспомним, что W равна удвоенной ширине полосы низкочастотных сигналов $s_1(t)$ и $s_2(t)$). Поскольку с каждым отсчетом передается k бит, можно передавать со скоростью Wk бит/с. Теперь заметим, что, для того чтобы учесть шум, каждую точку сигнальной диаграммы должна окружать некоторая минимальная площадь, которая пропорциональна $N_0 W$. Общая площадь, занимаемая всей диаграммой, пропорциональна мощности сигнала S . Таким образом, число точек в диаграмме 2^k должно быть пропорционально $S/(N_0 W)$, что делает k пропорциональным $\log_2 [S/(N_0 W)]$.

Инженеры-связисты обычно выражают отношение сигнал/шум (т. е. $S/(N_0 W)$) в децибелах (дБ); количество децибел определяется выражением $10 \lg [S/(N_0 W)]$. Таким образом, k — число битов в квадратурном отсчете пропорционально отношению сигнал/шум в децибелах. Добавление 1 в (2.13) объяснить труднее; заметим, однако, что 1 требуется, для того чтобы пропускная способность оставалась положительной, когда отношение сигнал/шум очень мало. Приведенное интуитивное рассмотрение должно показать,

что вероятность ошибки зависит от расположения точек в диаграмме и, следовательно, от k . Вместе с тем изящество теоремы Шеннона состоит в том, что она утверждает, что, используя коды, исправляющие ошибки, можно достичь любой скорости передачи, меньшей C , при произвольно малой вероятности ошибки.

Пропускная способность речевых телефонных каналов обычно составляет примерно 25 000 бит/с; это показывает, что скорости передачи указанных речевых модемов заметно близки к теоретическому пределу.

Синхронизация сигнала, определение фазы несущей и адаптивная компенсация в высокоскоростных модемах обычно обеспечиваются посредством обновления соответствующих величин на основе информации, поступающей от принятых сигналов. Следовательно, для этих модемов важна синхронность работы; каждые T секунд k бит должны поступать в модем, и устройство УЛПД должно производить холостое заполнение, когда отсутствуют данные для передачи. Вместе с тем для низкоскоростных модемов допустимо, чтобы физический канал становился пустым в промежутках между кадрами или даже между отдельными символами. Например, у речевых модемов на скоростях порядка 2400 бит/с или более передача данных является синхронной, тогда как на более низких скоростях передача может быть прерывисто-синхронной или символьно-асинхронной. У коаксиального же кабеля полоса пропускания настолько велика, что прерывисто-синхронная передача может использоваться на скорости 10 Мбит/с или более. Как упоминалось ранее, это существенно для систем с множественным доступом, подобным Ethernet.

2.2.6. Частотное и временное уплотнение

В предыдущих подразделах были рассмотрены ограничения на ширину полосы пропускания, обусловленные фильтрацией физического канала и шумом. Однако часто физический канал используется совместно многими сигналами, каждому из которых выделена отдельная часть доступной полосы пропускания; это называется частотным уплотнением (ЧУ). Наиболее распространенные примеры такого уплотнения дают радиовещание с применением амплитудной или частотной модуляции и телевизионное вещание, когда каждая станция использует отдельную полосу частот. Другим распространенным примером является часто применяемое частотное уплотнение речевых каналов в телефонных сетях. В этом случае каждому уплотняемому сигналу приписывается его собственная выделенная полоса частот, чтобы не возникла интерференция (иногда называемая перекрестной) с другими сигналами. Методы модуляции, представленные в предыдущих разделах, одинаково применимы как в случае, когда ограничение

на ширину полосы пропускания обусловлено ЧУ, так и в случае, когда оно обусловлено фильтрацией канала.

Можно считать, что ЧУ — это метод разбиения большого канала на множество малых. Предположим, что физический канал имеет полосу пропускания шириной W Гц и требуется разбить его на m равных подканалов. Тогда на каждый подканал приходится полоса в W/m Гц и, таким образом, W/m квадратурных отсчетов в секунду для передачи данных. (На практике полоса частот, приходящаяся на один подканал, должна быть несколько меньше, чтобы между подканалами оставались защитные полосы, но здесь это не учитывается ради простоты.) На каждый подканал приходится $1/m$ от всей мощности сигнала и $1/m$ от всего шума, так что, согласно равенству (2.13), каждый подканал имеет $1/m$ от всей пропускной способности.

Временное уплотнение (ВУ) — это другой метод разбиения большого канала на множество малых. В этом случае один модем обычно использует всю полосу пропускания W . Если задано m потоков двоичных данных с одинаковой скоростью, которые необходимо передать, то эти m битовых потоков можно объединить в один битовый поток. Обычно при этом передача данных проходит в следующих друг за другом кадрах. Каждый кадр содержит m окон, по одному на каждый битовый поток, который надо уплотнить; окно содержит фиксированное число битов, иногда 1, иногда 8, а иногда больше. Обычно каждый кадр также имеет дополнительные биты, чтобы помогать приемнику поддерживать кадровую синхронизацию. Например, система T1, которая широко используется в телефонных сетях, уплотняет 24 потока данных посредством восьмибитовых окон и кадров, содержащих один дополнительный бит для синхронизации. Полная скорость передачи данных равна 1,544 Мбит/с, а на каждый из уплотняемых потоков приходится 64 000 бит/с.

Одна из точек зрения на ЧУ и ВУ состоит в том, что в каждом случае для передачи выбирается W квадратурных отсчетов в секунду (или в более общем виде $2W$ отдельных отсчетов); в одном случае отсчеты распределяются по частоте, а в другом — во времени.

2.2.7. Другие недостатки каналов

Результаты воздействия фильтрации и шума были обсуждены в предыдущих подразделах. Существует ряд других возможных недостатков физических каналов. Иногда внутри физического канала производятся многоэтапная модуляция и демодуляция. Это может вызвать небольшие отклонения фазы и смещение частоты несущей в принимаемом сигнале. Усиление сигналов в канале и модемах может вызвать заметное нелинейное искажение.

Импульсы шума могут возникать из-за грозových разрядов или коммутации. Обслуживающий персонал может производить короткое замыкание или разъединение канала на некоторое время. Наконец, могут возникать перекрестные воздействия со стороны других частотных диапазонов или близлежащих проводов.

Все воздействия до некоторой степени можно рассматривать как дополнительные источники шума. Однако эти источники шума имеют другие статистические свойства по сравнению с шумом, учитываемым в теореме Шеннона (среди специалистов этот шум называется аддитивным белым гауссовым шумом). Принципиальное различие состоит в том, что ошибки, вызываемые дополнительными источниками шума, имеют тенденцию возникать пачками произвольной длины. В результате на практике невозможно добиться произвольно малых вероятностей ошибок, о которых говорится в теореме Шеннона. Необходимо использовать также обнаружение ошибок и повторные передачи на уровне УЛПД; эти вопросы рассматриваются в разд. 2.3 и 2.4.

2.2.8. Цифровые каналы

В большинстве случаев физические каналы проектируются так, чтобы передавать непосредственно цифровые данные, и блок УЛПД может взаимодействовать с цифровым каналом, а не с модемом, преобразующим цифровые данные в аналоговые сигналы. До некоторой степени это просто вопрос о том, кто поставляет модем: поставщик каналов или пользователь каналов? Однако это слишком упрощенный подход. Канал, спроектированный для непосредственной передачи цифровых данных (такой как система T1, упомянутая ранее), часто способен обеспечивать более высокие скорости передачи данных с меньшими вероятностями ошибок, чем тот, который передает аналоговые сигналы.

Основной причиной этого преимущества в характеристиках является тип регенераторов, используемых в цифровых каналах. Регенераторы — это, по существу, усилители в различных точках пути следования сигнала, предназначенные для устранения его затухания. В канале, спроектированном для аналоговых сигналов, как сигнал, так и шум должны усиливаться на каждом регенераторе. Следовательно, шум в последнем приемнике является суммой шумов каждого участка пути. Вместе с тем в цифровом канале цифровой сигнал можно восстановить на каждом регенераторе. Это означает, что шумы участков не накапливаются, и ошибка возникает только в том случае, когда шум участка достаточно силен, чтобы вызвать ошибку. В сущности шум подавляется в значительной степени на каждом этапе. Из-за подавления шума, а также из-за низкой стоимости цифровой обработки все шире используются цифровые каналы (такие как система T1) для пере-

дачи аналоговых сигналов, например речи. Аналоговые сигналы дискретизируются по времени (обычно 8000 отсчетов в секунду для речи) и затем квантуются по уровню (обычно 8 бит на отсчет) при получении цифрового входного сигнала для цифрового канала. Увеличение потребности в объединении передачи аналоговых и цифровых данных в цифровых каналах является одним из доводов в пользу цифровых сетей интегрального обслуживания (ЦСИО).

2.2.9. Передающая среда в физических каналах

Наиболее распространенными передающими средами в физических каналах являются скрученная пара (т. е. пара проводов, скрученных между собой, что частично подавляет воздействие электромагнитного излучения других источников), коаксиальный кабель, оптическое волокно, радио, микроволновый и спутниковый каналы. В первых трех средах мощность распространяемого сигнала падает экспоненциально с расстоянием (т. е. затухание в децибелах зависит линейно от расстояния). Из-за затухания регенераторы устанавливаются через каждые несколько километров. Скорость затухания меняется с частотой, поэтому, если регенераторы расставлены более часто, полезная полоса частот увеличивается, что приводит к необходимости выбора компромиссного решения относительно скорости передачи данных и стоимости регенераторов. Несмотря на это, полезно иметь приближенную оценку скоростей передачи данных в каналах, использующих эти среды.

Скрученная пара проводов широко используется в телефонной сети для соединения абонентов с местными станциями и для передачи данных. Один Мбит/с является типичной скоростью передачи данных для расстояний порядка километра или меньше. Коаксиальный кабель широко используется в локальных сетях, кабельном телевидении и высокоскоростных двухточечных линиях. Типичные скорости передачи данных лежат в интервале от десяти до нескольких сотен Мбит/с. Скорости передачи данных по оптическому волокну могут достигать 1000 Мбит/с и более. Значимость оптического волокна быстро увеличивается, главные проблемы связаны с генерированием, приемом, усилением и коммутацией такого огромного количества данных.

В радиоканале, микроволновом и спутниковом каналах используется распространение электромагнитных волн в открытом пространстве. Затухание падает с расстоянием обычно много медленнее, чем в проводных каналах, так что можно или вообще обойтись без регенераторов или размещать их гораздо реже, чем в случае проводных линий. Частоты ниже 1000 МГц обычно называются радиочастотами, а более высокие — микроволновыми.

Радиочастоты также разделяются частотой 30 МГц. Выше 30 МГц ионосфера прозрачна для электромагнитных волн, тогда как ниже 30 МГц волны могут отражаться от ионосферы. Таким образом, выше 30 МГц распространение происходит по траекториям прямой видимости. При этом антенны часто устанавливаются на башнях или возвышенностях для увеличения длины траекторий, но длина траектории все же несколько ограничена и регенераторы часто бывают необходимы. Частотный диапазон от 30 до 1000 МГц используется телевидением на УВЧ и ОВЧ, радиовещанием с ЧМ и в множестве различных специализированных приложений. Этот диапазон используется в пакетных радиосетях, которые будут рассматриваться в разд. 4.6 гл. 4. Типичные скорости передачи данных в этом диапазоне имеют большой разброс, например, в пакетной радиосети DARPA (Управление перспективных исследований министерства обороны США) используются 100 000 и 400 000 бит/с.

Ниже 30 МГц возможно распространение на далекие расстояния, превышающие пределы прямой видимости, за счет отражения от ионосферы. По иронии судьбы полоса частот от 3 до 30 МГц называется высокочастотным (ВЧ) диапазоном. Эта терминология восходит к начальным этапам развития радио. Высокочастотный диапазон сильно зашумлен, но интенсивно используется (например, любительской радиосвязью) и в нем происходят замирания. Замирание можно рассматривать как фильтрацию в канале с переходной характеристикой, которая относительно быстро изменяется со временем; оно вызывается изменяющимся со временем многолучевым характером распространения сигнала от источника до пункта назначения. Типичные скорости передачи в этом диапазоне не превышают 2400 бит/с.

В микроволновых линиях (выше 1000 МГц) сигналы распространяются по траекториям прямой видимости. Антенны (как правило, остронаправленные параболические) обычно устанавливаются на башнях или вершинах возвышенностей, типичные длины участков при этом находятся в пределах от 10 до 200 км. Более протяженные траектории можно получить, используя регенераторы. Эти линии могут пропускать около 1000 Мбит/с. Обычно они совместно используются для дальней телефонной связи, телевизионного вещания и передачи данных.

В спутниковых линиях используются микроволновые частоты, при этом спутник является регенератором. По скорости передачи и применению они аналогичны микроволновым линиям. Один спутник-регенератор может принимать сигналы от множества наземных станций и ширококвещательным образом передавать обратно в другой частотной полосе всем наземным станциям. Спутник может иметь антенну с многолепестковой диаграммой направленности, что позволяет ему действовать как коммутатору

для множества микроволновых линий. В гл. 4 описываются методы множественного доступа, позволяющие разным наземным станциям совместно использовать общую полосу частот.

В этом разделе представлено краткое введение в область физических каналов и их применения для передачи данных. В качестве линии подсети связи может использоваться любой из этих физических каналов или часть подобного канала на основе ВУ или ЧУ. Несмотря на сложность области (которую мы рассмотрели лишь в общих чертах), эти линии могут рассматриваться верхними уровнями просто как ненадежные битовые тракты.

2.3. Обнаружение ошибок

В следующих четырех разделах рассматривается управление линией передачи данных. Этот раздел посвящен обнаружению ошибок в передаче, а следующий раздел — запросам на повторную передачу. Предположим сначала, что в принимающем модуле управления линией передачи данных (УЛПД) известно, где начинаются кадры и где кончаются. В этом случае следующая проблема состоит в определении, в каких кадрах содержатся ошибки. Из рассмотрения уровневой архитектуры следует, что пакеты, поступающие на уровень УЛПД, являются произвольными последовательностями битов (т. е. задача уровня УЛПД заключается в том, чтобы обеспечить безошибочную передачу пакетов следующего более высокого уровня независимо от того, какие последовательности битов содержат пакеты). Таким образом, любая последовательность битов, появляющаяся на принимающем модуле УЛПД, может считаться пакетом, и ошибки невозможно обнаружить, анализируя пакет. Заметим, что преобразование пакета из k битов в некоторое другое представление длины k не может помочь решению проблемы; существует 2^k возможных пакетов и все возможные последовательности битов длины k должны использоваться для представления всех возможных пакетов. Из этого следует, что к пакету нужно добавлять дополнительные биты, чтобы обнаружить ошибки.

2.3.1. Однобитовые проверки на четность

Простейший способ обнаружения ошибки — это добавление одного бита, который называется *битом проверки на четность*, к последовательности битов данных. Этот бит проверки на четность имеет значение 1, если число единиц в последовательности битов нечетное, и значение 0 — в противном случае (рис. 2.13). Другими словами, бит проверки на четность равен сумме по модулю 2 значений битов в исходной последовательности битов

s_1	s_2	s_3	s_4	s_5	s_6	s_7	c
1	0	1	1	0	0	0	1

Рис. 2.13. Однобитовая проверка на четность. Последний бит c равен сумме по модулю 2 s_1, s_2, \dots, s_k , где $k = 7$.

(k по модулю j , где k — целое и j — положительное целое, равное целому числу m , $0 \leq m < j$, такому, что $k - m$ делится на j).

В символьном коде ASCII символам ставятся в соответствие последовательности из семи битов, к которым добавляется восьмой бит проверки на четность. Можно подумать, что бит проверки на четность также можно добавлять в конце пакета, но, как станет ясно, это недостаточно надежное средство обнаружения ошибок.

Заметим, что общее число единиц в закодированной последовательности (т. е. в исходной последовательности плюс добавочный бит проверки на четность) всегда четное. Если закодированная последовательность передается и при передаче возникает единственная ошибка из-за перехода 0 в 1 или 1 в 0, то число единиц в последовательности становится нечетным и ошибка может быть обнаружена приемником. Заметим, что приемнику неизвестно, какой бит ошибочный и сколько ошибок возникло; имеется только информация, что ошибки возникли, так как число единиц нечетно.

Замечательно то, что при любой длине последовательности битов однобитовая проверка на четность позволяет обнаружить любую единичную ошибку в закодированной последовательности. К сожалению, две ошибки в закодированной последовательности всегда восстанавливают четность числа единиц, так что две ошибки обнаружить невозможно. В общем случае нечетное число ошибок обнаруживается, а любое четное число не обнаруживается.

Несмотря на привлекательную простоту однобитовой проверки на четность, она непригодна для надежного обнаружения ошибок; во многих ситуациях она обнаруживает ошибки только примерно у каждой второй закодированной последовательности, содержащей ошибки. Существуют две причины такого плохого показателя работы. Первая состоит в том, что многие модемы преобразуют несколько битов в один отсчет входного сигнала физического канала (см. подразд. 2.2.5), и ошибка при приеме такого отсчета обычно порождает ошибки в нескольких битах. Второй причиной является то, что многие виды шума, такие как молния и временные разрывы соединений, порождают длинные пакеты ошибок. По двум этим причинам при возникновении одной или более ошибок в закодированной последовательности четное число ошибок почти также вероятно, как и нечетное число, и однобитовая проверка на четность является неэффективной.

Рис. 2.14. Проверки на четность по горизонтали и вертикали. Каждый бит проверки на четность по горизонтали проверяет свою строку, а каждый бит проверки на четность по вертикали проверяет свой столбец. Заметим, что в случае (б), если каждый из заключенных в кружок битов изменяет значение, то четность в каждом столбце и каждой строке тем не менее сохраняется.

1	0	0	1	0	1	0	1
0	1	1	1	0	1	0	0
1	1	1	0	0	0	1	0
1	0	0	0	1	1	1	0
0	0	1	1	0	0	1	1
1	0	1	1	1	1	1	0

Проверки по вертикалям

(а)

1	0	0	1	0	1	0	1
0	1	1	1	0	1	0	0
1	1	①	0	0	①	1	0
1	0	0	0	1	1	1	0
0	0	①	1	0	①	1	1
1	0	1	1	1	1	1	0

(б)

2.3.2. Проверки на четность по вертикали и горизонтали

Другой простой интуитивный подход к обнаружению ошибок состоит в том, что последовательность битов данных перестраивают в двумерный массив (рис. 2.14) и вычисляют биты проверки на четность для каждой строки и каждого столбца. Бит проверки на четность в правом нижнем углу может относиться к строке или столбцу битов проверки на четность или же к массиву данных. Если четное число ошибок находится в пределах одной строки, то каждую из них можно обнаружить с помощью соответствующего бита проверки на четность по столбцу; аналогично ошибки только в одном столбце можно обнаружить с помощью битов проверки на четность по строкам. К сожалению, любой набор из четырех ошибок, принадлежащих двум строкам и двум столбцам (т. е. образующих прямоугольник, как показано на рис. 2.14, б), обнаружить невозможно.

Эта схема чаще всего используется в случае, когда данные состоят из последовательности символов, представленных в коде ASCII. Каждый закодированный символ можно считать строкой массива на рис. 2.14; бит проверки на четность по строке в этом случае является просто последним битом закодированного символа. Биты проверки по столбцам тривиально вычисляются программным или аппаратным способом. Основной недостаток этой схемы заключается в том, что она допускает сбои при обнаружении довольно коротких пакетов ошибок (например, ошибок

в двух соседних битах в каждой из двух соседних строк). Поскольку идущие друг за другом ошибки довольно часто встречаются на практике, вероятность подобных сбоев недопустимо высока.

2.3.3. Коды с проверкой на четность

Привлекательной особенностью проверок на четность по горизонтали и вертикали является еще и то, что лежащая в их основе идея легко распространяется на случай произвольных кодов с проверкой на четность. Эта идея сводится к тому, что в исходной последовательности битов (массив битов данных в случае, показанном на рис. 2.14) производятся проверки на четность различных подмножеств битов (строк и столбцов в случае на рис. 2.14). Преобразование последовательности битов данных в последовательность битов данных и битов проверки на четность называется *кодом с проверкой на четность* или *линейным кодом*. Пример кода с проверкой на четность (отличного от проверок по горизонтали и вертикали) показан на рис. 2.15. Код с проверкой на четность определяется конкретным набором подмножеств, используемых для выполнения проверок на четность. Заметим, что слово *код* относится к самому преобразованию; закодированную последовательность битов (данные плюс проверки на четность) называют *словом*.

Пусть K обозначает длину последовательности данных заданного кода с проверкой на четность и пусть L — число битов проверки на четность. При структуре кадра, показанной на рис. 2.2, можно считать, что последовательностью данных является заголовок и пакет и что множество битов проверки на четность является трейлером. Заметим, что наряду с обнаружением ошибок в самих пакетах важно обнаруживать ошибки в управляющих битах заголовка. Итак, $K + L$ есть длина кадра, которая считается постоянной. При заданном коде каждая из 2^K возможных последовательностей данных длины K отображается на кадр (т. е. кодовое слово) длины $K + L$. В системе с обнаружением ошибок при передаче кадра принимающий модуль УЛПД определяет, каждый ли бит проверки на четность равен по-прежнему сумме по модулю 2 соответствующего подмножества битов данных. Если это так, то кадр считается свободным от ошибок, в противном случае обнаруживается наличие ошибок. Если ошибки при передаче по линии превратили одно кодовое слово в другое, то кадр считается свободным от ошибок, в этом случае говорят, что кадр содержит необнаруживаемые ошибки.

При некотором заданном коде желательно уметь находить вероятность необнаруживаемых ошибок в кадре при передаче по конкретной линии. К сожалению, это очень трудно. Во-первых,

s_1	s_2	s_3	c_1	c_2	c_3	c_4	
1	0	0	1	1	1	0	
0	1	0	0	1	1	1	$c_1 = s_1 + s_3$
0	0	1	1	1	0	1	$c_2 = s_1 + s_2 + s_3$
1	1	0	1	0	0	1	$c_3 = s_1 + s_2$
1	0	1	0	0	1	1	$c_4 = s_2 + s_3$
1	1	1	0	1	0	0	
0	0	0	0	0	0	0	

Рис. 2.15. Пример кода с проверкой на четность. Кодовые слова приведены слева, а правила вычисления битов проверки на четность — справа.

ошибки в линиях появляются не независимо и в виде пакетов; не существует хороших моделей для длины или интенсивности этих пакетов, которые меняются в широком диапазоне для линий одного и того же типа. Во-вторых, при любом разумном коде частота необнаруживаемых ошибок очень мала и, следовательно, ее трудно измерить экспериментально и к тому же она зависит от редких, трудно моделируемых событий. В литературе имеется много расчетов вероятности необнаруживаемых ошибок, но эти расчеты обычно основываются на предположении о независимости ошибок; результаты обычно отличаются на порядки величин от действительных значений.

Вследствие этих трудностей эффективность кода по обнаружению ошибок обычно измеряется тремя параметрами: 1) минимальное расстояние кода, 2) способность обнаружения пакетов ошибок и 3) вероятность того, что совершенно случайная последовательность будет воспринята как свободная от ошибок. Минимальное расстояние кода определяется как наименьшее число ошибок, которое может превратить одно кодовое слово в другое. Как мы установили, минимальное расстояние кода с однокбитовой проверкой на четность равно 2, а минимальное расстояние кода с проверками на четность по горизонтали и вертикали равно 4.

Длиной пакета ошибок называется число ошибочных бит, считая от первой ошибки до последней включительно. Способность кода обнаруживать пакеты ошибок определяется как наибольшее целое B , такое, что код может обнаруживать все пакеты длины B или меньше. У кода с однокбитовой проверкой на четность способность обнаруживать пакеты равна 1, тогда как у кода с проверками на четность по горизонтали и вертикали способность обнаруживать пакеты равна 1 плюс длина строки (предполагается, что передача данных производится по строкам).

Под совершенно случайной последовательностью длины $K + L$ подразумевается такая последовательность длины $K + L$, которая

принимается с вероятностью 2^{-K-L} . Так как число кодовых слов равно 2^K , то вероятность необнаруженной ошибки представляет собой вероятность того, что случайная последовательность совпадает с одним из кодовых слов; это имеет место с вероятностью 2^{-L} (вероятность того, что принятая случайная последовательность окажется такой же, как и переданный кадр, не учитывается). Обычно это дает хорошую оценку вероятности необнаруживаемых ошибок для кадра, в котором ошибки значительно превышают как минимальное расстояние, так и способность кода обнаруживать пакеты ошибок.

Коды с проверкой на четность могут использоваться не только для обнаружения ошибок, но и для исправления ошибок. Например, в случае проверок на четность по горизонтали и вертикали любая одиночная ошибка может быть исправлена посредством простого поиска той строки и того столбца, у которых проверка дает нечетный результат. В задаче 2.10 показывается, что код с минимальным расстоянием d может использоваться для исправления любой комбинации из менее чем $d/2$ ошибок. Коды с проверкой на четность (и сверточные коды, которые тесно связаны с ними, но не имеют кадровой структуры) широко используются для исправления ошибок на физическом уровне. Современный взгляд на исправление ошибок состоит в том, что использование этого средства наряду с разработкой методов передачи сигналов считается направлением, ведущим к созданию высокоскоростных виртуальных битовых трактов с относительно низкой частотой появления ошибок.

Исправление ошибок, как правило, не используется на уровне УЛПД, поскольку характеристики исправляющего ошибки кода сильно зависят от физических характеристик канала. Однако обнаружение ошибок необходимо уровню УЛПД для обнаружения редких остаточных ошибок, обусловленных длительными периодами сильного шума.

2.3.4. Циклические избыточные проверки

В настоящее время кодами с проверкой на четность, используемыми в большинстве случаев на уровне УЛПД, являются коды с циклическими избыточными проверками (ЦИП). Биты проверки на четность называются ЦИП. Как и ранее, пусть L — длина ЦИП (т. е. число проверочных битов), а K — длина последовательности битов данных (т. е. заголовка и пакета в кадре). Удобно обозначать биты данных через $s_{K-1}, s_{K-2}, \dots, s_1, s_0$ и представлять последовательность в виде многочлена $s(D)$ с коэффициентами s_{K-1}, \dots, s_0 ,

$$s(D) = s_{K-1}D^{K-1} + s_{K-2}D^{K-2} + \dots + s_0. \quad (2.14)$$

Можно считать, что степени переменной D сохраняют порядок битов; предполагается, что передача начинается с членов старшего порядка. ЦИП представляется другим многочленом

$$c(D) = c_{L-1}D^{L-1} + \dots + c_1D + c_0. \quad (2.15)$$

Весь кадр из передаваемой информации и ЦИПа можно в этом случае представить как $x(D) = s(D)D^L + c(D)$, т. е. как

$$x(D) = s_{K-1}D^{L+K-1} + \dots + s_0D^L + c_{L-1}D^{L-1} + \dots + c_0. \quad (2.16)$$

Многочлен $c(D)$, представляющий ЦИП, является функцией информационного многочлена $s(D)$; эта функциональная связь определяется *порождающим многочленом* $g(D)$; это многочлен степени L с двоичными коэффициентами, который задает конкретный код с ЦИП,

$$g(D) = D^L + g_{L-1}D^{L-1} + \dots + g_1D + 1. \quad (2.17)$$

При заданном $g(D)$ отображение информационного многочлена в многочлен $c(D)$, представляющий ЦИП, задается равенством

$$c(D) = \text{остаток } [s(D)D^L/g(D)]. \quad (2.18)$$

Операция деления многочленов, использованная выше, является обычным делением одного многочлена на другой за исключением того, что коэффициенты принимают только двоичные значения, а арифметические операции над коэффициентами выполняются по модулю 2. Таким образом, например, $(1 + 1) \bmod 2 = 0$, а $(0 - 1) \bmod 2 = 1$. Заметим, что вычитание в арифметике по модулю 2 эквивалентно сложению. Приведем пример выполнения операций в (2.18)

$$\begin{array}{r} D^2 + D \\ D^3 + D^2 + 1 \overline{) D^5 + D^3} \\ \underline{D^6 + D^4 + D^2} \\ D^4 + D^3 + D^2 \\ \underline{D^4 + D^3 + D} \\ D^2 + D = \text{остаток.} \end{array}$$

Поскольку степень многочлена $g(D)$ не превышает L , то степень остатка не превышает $L - 1$. Если степень $c(D)$ меньше, чем $L - 1$, то соответствующие старшие коэффициенты c_{L-1}, \dots в (2.18) полагаются равными нулю.

Это деление легко реализуется аппаратно при помощи цепи с регистром сдвига и обратными связями, показанной на рис. 2.16. Сравнивая цепь с приведенным выше примером деления, можно убедиться, что последовательные значения разрядов регистра

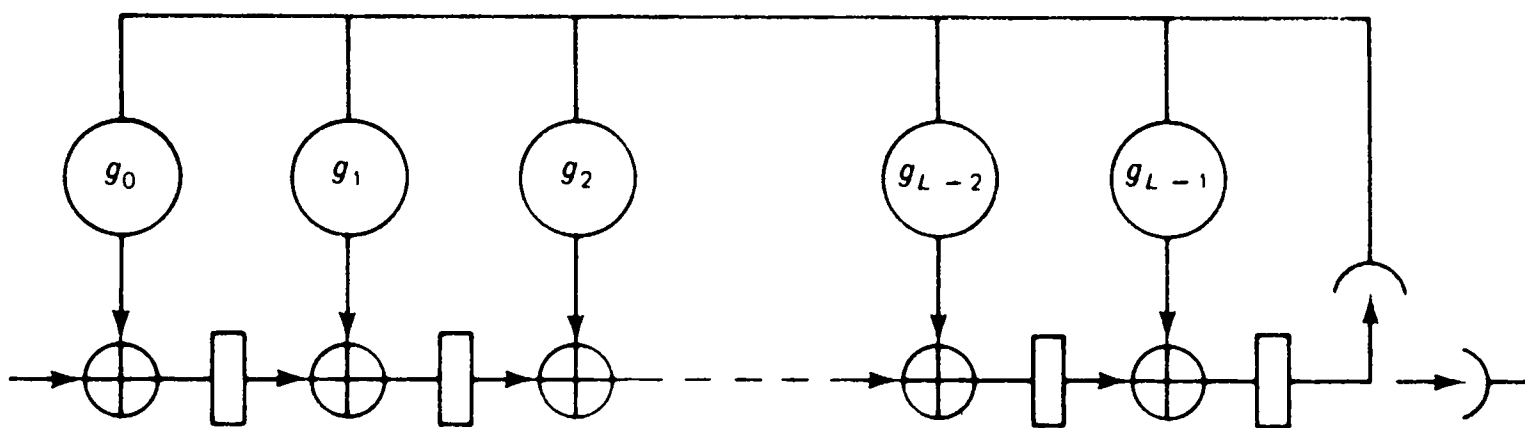


Рис. 2.16. Цепь с регистром сдвига для деления многочленов и нахождения остатка. Каждый прямоугольник обозначает однобитовый запоминающий элемент, а находящаяся перед ним окружность обозначает сумматор по модулю 2. Большие окружности, расположенные выше, обозначают устройства умножения на величины g_i . Первоначально в регистр записываются первые L битов $s(D)$, при этом s_{K-1} находится справа. В каждом такте новый бит многочлена $s(D)$ поступает слева и в регистр записываются соответствующие суммы по модулю 2 значений обратной связи и предыдущих разрядов. После K сдвигов переключатель, находящийся справа, переходит в горизонтальное положение и результат ЦИП считывается.

сдвига совпадают с коэффициентами промежуточных остатков, появляющихся в процессе деления. На практике ЦИП обычно вычисляются в СБИС, которые часто выполняют также другие функции УЛПД.

Пусть $z(D)$ обозначает частное от деления $s(D) D^L$ на $g(D)$. Тогда $s(D)$ можно представить следующим образом:

$$s(D) D^L = g(D) z(D) + c(D). \quad (2.19)$$

Вычитая $c(D)$ (по модулю 2) из обеих частей этого равенства и вспоминая, что вычитание по модулю 2 эквивалентно сложению, получаем

$$x(D) = s(D) D^L + c(D) = g(D) z(D). \quad (2.20)$$

Таким образом, все кодовые слова делятся на $g(D)$ и все многочлены, делящиеся на $g(D)$, являются кодовыми словами. Пока еще не показано, что отображение $s(D)$ в $x(D)$ соответствует некоторому коду с проверкой на четность. Это демонстрируется в задаче 2.15, однако в дальнейшем не требуется.

Теперь предположим, что $x(D)$ передается, а принятая последовательность представляет собой многочлен $y(D)$, где $y(D)$ отличается от $x(D)$ из-за ошибок в канале связи. Если последовательность ошибок представляется многочленом $e(D)$, тогда $y(D) = x(D) + e(D)$, где знак $+$ здесь, как и везде в этом разделе, обозначает сложение по модулю 2; каждой ошибке в кадре соответствует ненулевой коэффициент $e(D)$, т. е. несовпадение коэффициентов $y(D)$ и $x(D)$. Можно вычислить $\text{REM}[y(D)/g(D)]$ (т. е. остаток от деления) при помощи цепи, которая, по существу,

аналогична приведенной выше. Так как было показано, что $x(D)$ делится на $g(D)$, то

$$\text{REM}[y(D)/g(D)] = \text{REM}[e(D)/g(D)]. \quad (2.21)$$

Если ошибки не возникли, то $e(D) = 0$ и остаток будет равен нулю. Правило, которое используется в приемнике, состоит в том, что кадр не содержит ошибок, если этот остаток равен нулю, и содержит ошибки в противном случае. Может случиться, что ошибки действительно возникают (т. е. $e(D) \neq 0$), а в приемнике не удастся их обнаружить из-за того, что этот остаток равен нулю; это имеет место только тогда, когда $e(D)$ является некоторым кодовым словом. Другими словами, $e(D) \neq 0$ не обнаруживается тогда и только тогда, когда

$$e(D) = g(D) z(D) \quad (2.22)$$

при некотором ненулевом многочлене $z(D)$. Теперь исследуем условия, при которых могут возникать необнаруживаемые ошибки.

Сначала предположим, что возникла одиночная ошибка, например, $e_i = 1$, так что $e(D) = D^i$. Так как $g(D)$ имеет по крайней мере два ненулевых члена (т. е. D^L и 1), то произведение $g(D) z(D)$ должно также иметь по крайней мере два ненулевых члена при любом ненулевом $z(D)$ (см. задачу 2.13). Итак, $g(D) z(D)$ не может быть равно D^i ; поскольку это справедливо при любом i , то все одиночные ошибки обнаруживаются. Поскольку разница между степенями самого старшего и самого младшего членов в $g(D)$ (т. е. D^L и 1 соответственно) равна L , то степени самого старшего и самого младшего членов $g(D) z(D)$ отличаются по крайней мере на L при любом ненулевом $z(D)$. Следовательно, если $e(D)$ является кодовым словом, то длина пакета ошибок не меньше $L + 1$ ($+1$ возникает из-за определения длины пакета ошибок, как числа позиций начиная с первой ошибки и кончая последней ошибкой *включительно*).

Далее предположим, что возникла двойная ошибка, например, в позициях i и j , так что

$$e(D) = D^i + D^j = D^j (D^{i-j} + 1), \quad i > j. \quad (2.23)$$

Выше было установлено, что D^i не делится на $g(D)$ или на любой множитель $g(D)$; поэтому $e(D)$ не удастся обнаружить, если только $D^{i-j} + 1$ делится на $g(D)$. Для произвольного двоичного многочлена $g(D)$ степени L существует наименьшее n , для которого $D^n + 1$ делится на $g(D)$. Из теории конечных полей известно, что это наименьшее n может быть не больше чем $2^L - 1$; более того, для любого $L > 0$ существуют специальные многочлены степени L , называемые *примитивными многочленами*, для которых это наименьшее n равно $2^L - 1$. Итак, если в качестве

$g(D)$ использовать такой примитивный многочлен степени L и если длина кадра ограничена сверху величиной $2^L - 1$, то $D^{i-j} + 1$ не может делиться на $g(D)$; следовательно, все двойные ошибки обнаруживаются.

Фактически на практике для вычисления ЦИП обычно используется порождающий полином $g(D)$, который является произведением примитивного многочлена степени $L - 1$ и многочлена $D + 1$. В задаче 2.14 показывается, что многочлен $e(D)$ делится на $D + 1$, если только $e(D)$ содержит четное число ненулевых коэффициентов. Это обеспечивает обнаружение любого нечетного числа ошибок, а примитивный многочлен позволяет обнаруживать любую двойную ошибку (при условии, что длина блока меньше 2^{L-1}). Следовательно, у любого кода такого типа минимальное расстояние не меньше 4, способность обнаруживать пакеты ошибок не меньше L , а вероятность необнаружения ошибок в совершенно случайной последовательности равна 2^{-L} . Существуют два стандартных ЦИП-кода с $L = 16$, один называется ЦИП-16, а другой — ЦИП-МККТТ. Оба имеют приведенный выше вид (и, следовательно, указанные свойства). Их порождающие многочлены имеют вид

$$g(D) = D^{16} + D^{15} + D^2 + 1 \text{ у ЦИП-16,}$$

$$g(D) = D^{16} + D^{12} + D^5 + 1 \text{ у ЦИП-МККТТ.}$$

2.4. ARQ—методы повторной передачи

Основной смысл автоматического запроса повторной передачи ARQ состоит в обнаружении в приемном модуле УЛПД кадров, которые содержат ошибки, и последующем запросе передающего модуля УЛПД для повторной передачи информации, которая передавалась в этих ошибочных кадрах. Обнаружение ошибок обсуждалось в предыдущем разделе, а в этом разделе рассматривается задача запроса повторной передачи. Существуют два различных аспекта алгоритмов или протоколов повторной передачи. Первый аспект — это безошибочность, т. е. обеспечивает ли протокол то, что каждый пакет не имеет ошибок и ровно один раз выходит из приемного модуля УЛПД? Второй аспект — это эффективность, т. е. как много теряется пропускной способности битового тракта из-за ненужных ожиданий и повторных передач? Сначала строятся некоторые классы протоколов и показывается, что они безошибочны (в смысле, который будет позже определен более точно). Затем рассматривается влияние различных параметров в протоколах этих классов на эффективность.

Из рис. 2.2 видно, что пакеты поступают на уровень УЛПД из сетевого уровня; заголовок и трейлер приставляются к каждому пакету в модуле УЛПД для формирования кадров, а кадры

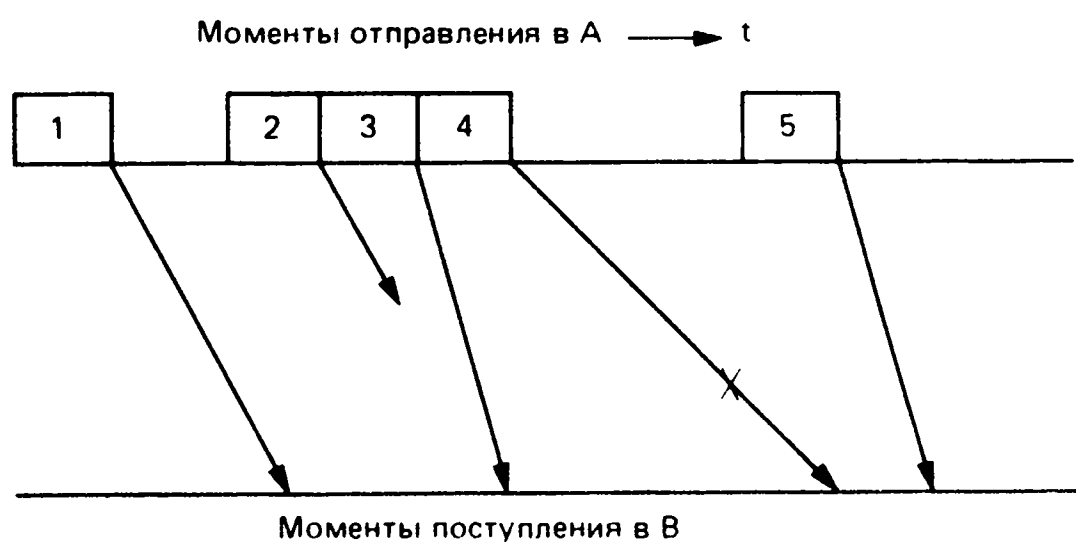


Рис. 2.17. Модель передачи кадров: кадр 2 теряется и не поступает в приемник; кадр 4 содержит ошибки; кадры имеют различную задержку передачи, но поступают в приемник в том порядке, в каком они передавались.

передаются по виртуальному битовому тракту (т. е. посылаются на физический уровень для передачи). Когда обнаруживается ошибка в кадре, передается новый кадр, содержащий старый пакет. Таким образом, первый передающийся кадр может содержать первый пакет, следующий кадр — второй пакет, третий кадр — повторение первого пакета и т. д. Когда пакет передается повторно, заголовок и трейлер могут быть, а могут и не быть такими же, как в более ранней попытке передачи.

Так как формирование кадра не будет рассматриваться вплоть до следующего раздела, мы по-прежнему предположим, что в приемном модуле УЛПД известно, когда начинаются и кончаются кадры; таким образом, ЦИП (или другой способ) может быть использован для обнаружения ошибок. Предположим также, отчасти нереалистично, что обнаруживаются *все* кадры, содержащие ошибки. Причина такого предположения заключается в том, что мы хотим показать, что ARQ работает безошибочно за исключением того случая, когда ошибки не обнаруживаются. Это лучшее, на что можно надеяться, так как обнаружение ошибки не может происходить совершенно надежно и с ограниченной задержкой; в частности, любое кодовое слово может стать другим кодовым словом под воздействием некоторого набора ошибок передачи. Это может вызвать выход ошибочных данных из модуля УЛПД или, что еще хуже, — привести к изменению управляющих битов.

В дальнейшем ряд предположений делается относительно битовых трактов, по которым передаются кадры. Эти предположения в некоторой степени являются более общими, чем предыдущие. Главная причина такого обобщения будет ясна, когда будет рассматриваться вопрос формирования кадров; в действительности эти общие условия позволяют ослабить предположение, что приемный УЛПД имеет кадррованную информацию.

Сначала допустим, что каждый передаваемый кадр задерживается на произвольное переменное время перед тем, как поступит в приемник, и что некоторые кадры могут быть потеряны и вообще не поступить в приемник. Однако считается, что кадры с ошибками или без них поступают в приемник в том же порядке, в каком они передавались. Это иллюстрируется на рис. 2.17.

2.4.1. ARQ с остановкой и ожиданием

Самый простой тип протокола повторной передачи называется протоколом с *остановкой и ожиданием*. Основная идея этого протокола состоит в том, что каждый пакет должен быть безошибочно принят до начала передачи следующего пакета. Таким образом, при передаче пакетов из точки *A* в *B* первый пакет передается в первом кадре, а затем передающий модуль УЛПД ждет. Если кадр правильно принят в точке *B*, из этой точки посылается обратно в точку *A* подтверждение (обозначаемое как *ack*); если кадр принят с ошибками, из точки *B* посылается обратно в точку *A* отрицательное подтверждение (обозначаемое как *nak*). Поскольку ошибки могут возникать как при передаче из точки *B* в точку *A*, так и при передаче из точки *A* в точку *B*, *ack* или *nak* защищаются с помощью ЦИП.

Если кадр принят без ошибок в точке *B* и соответствующее подтверждение принято без ошибок в точке *A*, то из *A* в новом кадре может начинаться передача следующего пакета. В противном случае ошибки могут быть обнаружены либо при передаче кадра, либо при обратной передаче *ack* или *nak*; при этом в новом кадре из *A* повторно передается старый пакет. Наконец, если либо кадр, либо *ack* или *nak* потеряны, *A* должен сделать тайм-аут, а затем повторить старый пакет. Так как кадры могут иметь любую задержку, для *A* возможна следующая последовательность

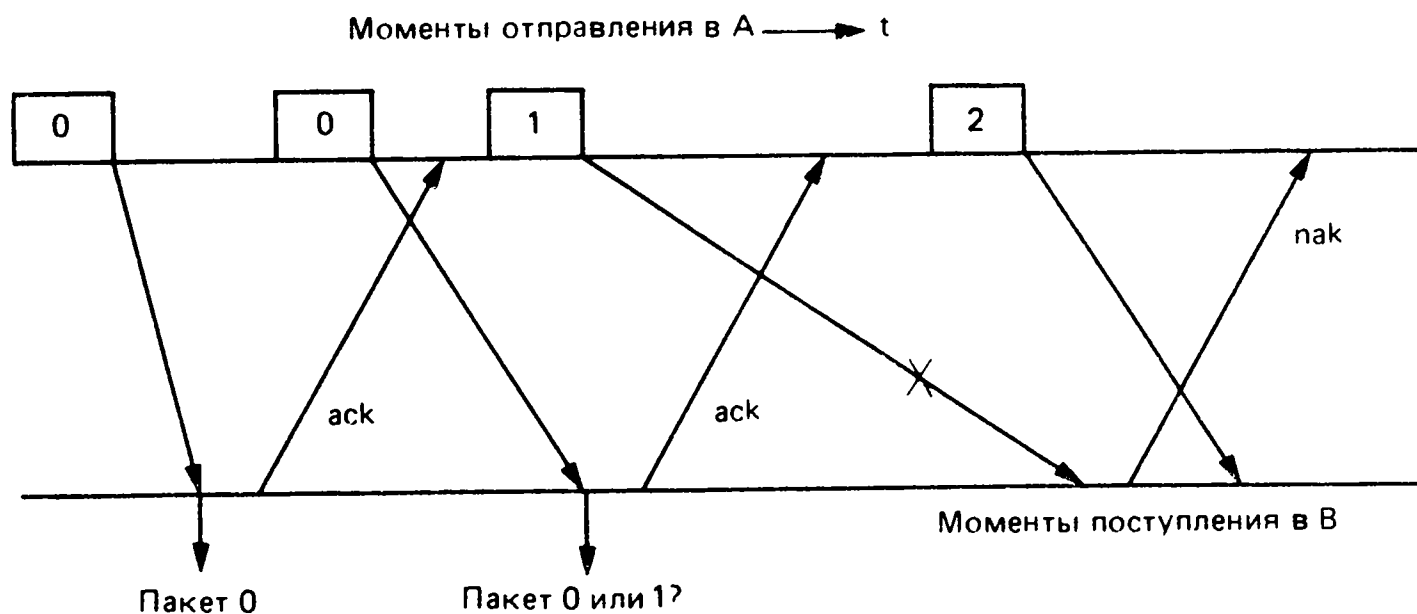


Рис. 2.18. Ошибки из-за отсутствия нумерации пакетов. Если передатчик в *A* ждет тайм-аут, а затем посылает пакет 0 второй раз, то на приемной стороне *B* неизвестно, была ли во втором кадре повторная передача пакета 0 или первая передача пакета 1.

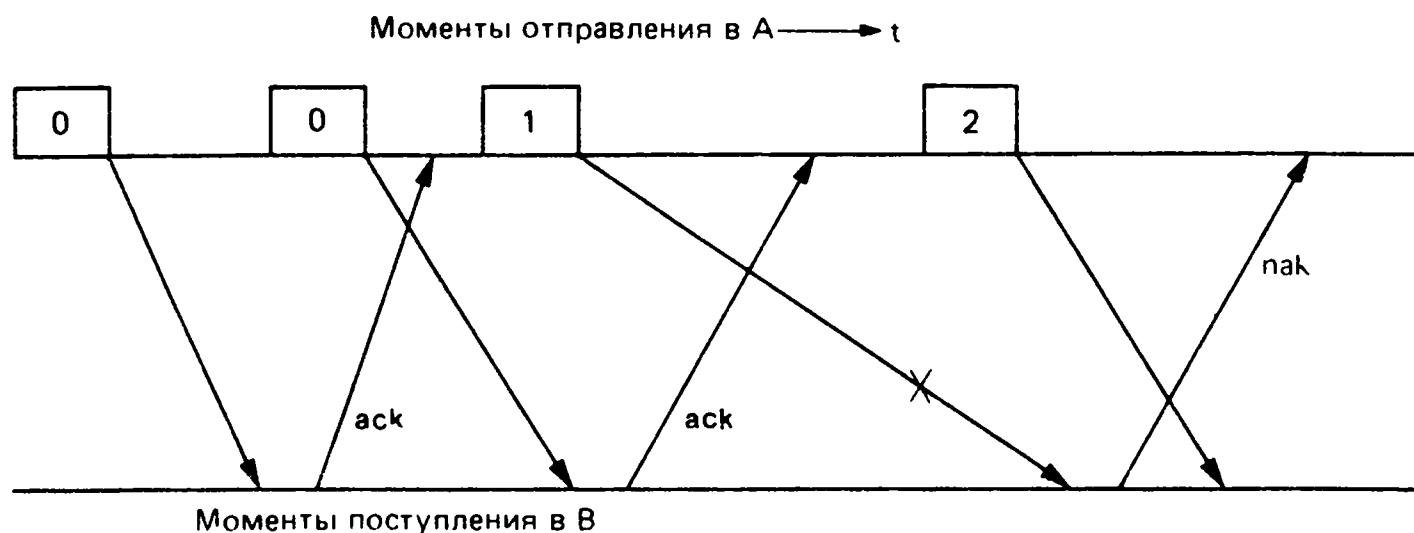


Рис. 2.19. Ошибки, вызванные отсутствием нумерации подтверждений ask. Если передатчик в А имеет тайм-аут и посылает пакет 0 второй раз, а затем передает пакет 1 после приема первого подтверждения ask, в узле А нельзя установить, соответствует ли второе подтверждение ask пакету 0 или 1.

событий — тайм-аут, повторная передача старого пакета, получение ask или пак от предыдущей передачи (рис. 2.18).

Заметим, что из-за тайм-аутов, а также из-за потенциальных ошибок в канале от В к А возможно, что в В будет правильно принят первый пакет в первом кадре, а затем он снова будет принят во втором кадре (рис. 2.18). По предположению пакеты состоят из произвольной последовательности битов, поэтому первый и второй пакеты сами по себе могут быть идентичными. Таким образом, если в узле В будет принят один и тот же пакет дважды, то в этом узле нельзя будет определить, что было во втором кадре — повторная передача первого пакета или посылка второго пакета (который случайно имеет ту же последовательность битов, что и первый пакет).

Самым простым решением этой проблемы является добавление в передающем модуле УЛПД (в узле) порядкового номера для распознавания успешно переданных пакетов. Хотя этот способ кажется очень простым, в таком виде он не будет работать. Проблема состоит в том, что подтверждения могут потеряться в канале обратной связи и, таким образом, узел В получит один и тот же пакет два раза подряд правильно, из этого узла новое подтверждение будет передано после второго приема (рис. 2.19). Из узла А после передачи пакета два раза и приема только одного подтверждения может передаваться следующий пакет; полученное второе подтверждение можно интерпретировать как подтверждение нового пакета, что свидетельствует о потенциальных недостатках системы.

Для того чтобы избежать недостатков этого типа, приемный модуль УЛПД (в В) вместо того, чтобы посылать ask или пак по линии обратной связи, возвращает по этой линии номер следующего ожидаемого пакета. Это включает всю информацию об ask и пак и устраняет двусмысленность по поводу того, какой

кадр подтверждается. Другое решение состоит в том, чтобы возвращать номер только что принятого пакета, однако так обычно не делают. Для получения пакета узел B может запрашивать этот следующий ожидаемый пакет через периодические интервалы времени или в произвольно выбранные моменты времени. Во многих приложениях существует поток данных от B к A ; в этом случае запросы на следующий пакет для одного направления обычно добавляются в заголовок кадров противоположного направления.

Должно быть ясно, что этот алгоритм передачи с остановкой и ожиданием, включая посылку порядкового номера (ПН) для каждого передаваемого пакета и номера запроса (НЗ), посылаемого приемником по цепи обратной связи, работает безошибочно (в предположении, что ЦИП обнаруживает все ошибки). Из узла A никогда не посылается новый пакет, пока он не будет запрошен узлом B , что подтверждает, что предыдущий пакет принят; в узле B всегда известно, какой пакет принимается, если выполняется ЦИП. Единственный недостаток состоит в том, что числа ПН и НЗ могут становиться неограниченно большими по мере продолжения передачи. Последняя модификация стратегии остановки и ожидания состоит в том, чтобы передавать ПН и НЗ по некоторому модулю; причем оказывается, что значение модуля, равное двум, является достаточным.

Покажем теперь, почему передача ПН и НЗ по модулю 2 при данных предположениях осуществляется безошибочно. Предположим, что система начинает работать в момент, когда нет кадра, передающегося по битовому тракту, и из узла A начинает посылаться пакет 0; а B ожидает пакет 0. Пусть A имеет два возможных состояния 0 и 1, соответствующие номеру (по модулю 2) пакета, посылаемого в настоящее время (или ожидаемого из более высокого уровня, если в настоящее время пакета нет). Таким образом, A стартует из состояния 0; переход в состояние 1 возникает при приеме безошибочного запроса на пакет с номером 1 по модулю 2. Заметим, что в узле A должно храниться больше информации, чем информации о данном состоянии; так должны храниться содержание пакета и время до конца тайм-аута, но нас здесь интересует только указанное двоичное состояние.

Аналогично считается, что B имеет два возможных состояния 0 и 1, соответствующие взятому по модулю 2 номеру ожидаемого в настоящее время пакета. Когда принимается пакет ожидаемого номера по модулю 2, модуль УЛПД в узле B выдает этот пакет на более высокий уровень и меняет свое состояние, ожидая следующий пакет. Начальное совместное состояние A и B есть (0, 0); когда первый пакет безошибочно принимается, состояние B переходит в 1, давая совместное состояние (0, 1). Когда A примет новое значение НЗ, т. е. 1, A перейдет в состояние 1 и совместным

состоянием будет (1, 1). Заметим, что существует фиксированная последовательность совместных состояний (0, 0), (0, 1), (1, 1), (1, 0), (0, 0) и т. д.; таким образом, A и B чередуются в изменении состояния. Интересно, что в момент переходов от (0, 0) к (0, 1) в узле B известно совместное состояние, но потом вплоть до следующего перехода от (1, 1) к (1, 0) совместное состояние неизвестно, т. е. в B неизвестно, что в A принята информация о подтверждении до тех пор, пока не будет принят следующий пакет. Аналогично в узле A известно совместное состояние в моменты перехода от (0, 1) к (1, 1) и от (1, 0) к (0, 0). Совместное состояние всегда неизвестно либо в A , либо в B , и часто неизвестно ни в A , ни в B . Ситуация, которая здесь возникает, очень похожа на задачу трех армий, которая обсуждалась в разд. 1.4. Однако здесь информация передается, даже если совместное состояние никогда не известно обеим станциям, тогда как в задаче трех армий нельзя скоординировать атаку из-за невозможности получения этой совместной информации.

Способ передачи с остановкой и ожиданием не является особо полезным для современных сетей передачи данных из-за очень низкой эффективности использования линий связи. Проблема состоит в том, что должна быть возможность что-либо делать во время ожидания подтверждения. Существует три известных способа обобщения основной идеи способа передачи с остановкой и ожиданием для достижения более высокой эффективности: ARQ, ARQ на n шагов назад и, наконец, ARQ выборочного повтора.

2.4.2. ARPANET ARQ

В ARPANET высокая эффективность достигается путем параллельного использования восьми стратегий остановки и ожидания и мультиплексирования (уплотнения) битового тракта для них, т. е. каждый пришедший пакет назначается на один из восьми виртуальных каналов в предположении, что один из восьми свободен. Если все виртуальные каналы заняты, пришедший пакет ждет за пределами модуля УЛПД (рис. 2.20). Занятые виртуальные каналы уплотняются в битовом тракте в том смысле, что кадры для различных виртуальных каналов посылаются по линии один за другим. Конкретный порядок, в котором посылаются кадры, не очень важен, но простой подход состоит в их передаче в порядке кругового опроса. Если очередь некоторого виртуального канала для передачи наступает раньше, чем для этого канала будет принято подтверждение, то пакет посылается снова. Таким образом, уплотнение устраняет необходимость тайм-аутов. (В реальном протоколе ARPANET, однако, тайм-ауты используются.) Когда принимается подтверждение для

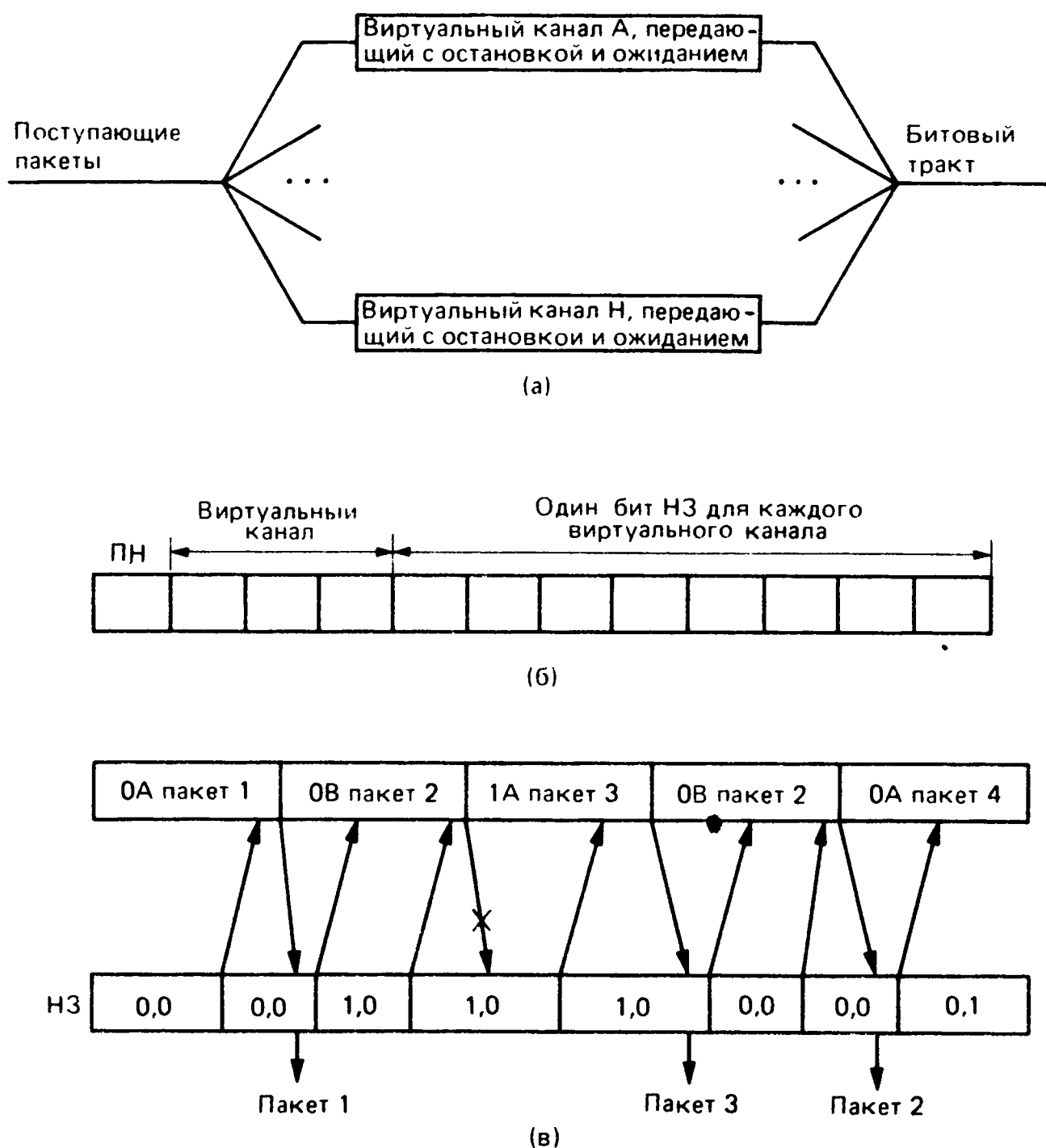


Рис. 2.20. ARPANET ARQ.

а — восемь уплотненных виртуальных каналов, передающих с остановкой и ожиданием; *б* — биты в заголовке для управления ARQ; *в* — функционирование уплотненного канала при уплотнении двух виртуальных каналов, передающих с остановкой и ожиданием. Кадры, идущие сверху вниз, показывают ПН и номер канала, а кадры, идущие снизу вверх, — НЗ для обоих каналов. Третий кадр, идущий снизу вверх, положительно подтверждает пакет 1 по каналу А.

кадра данного виртуального канала, этот канал становится пустым и может принять новый пакет от более высокого уровня.

В некоторой степени здесь требуется большее количество служебной информации, чем в основном протоколе остановки и ожидания. В частности, каждый кадр содержит как номер виртуального канала, требующий три бита, так и порядковый номер пакета по данному виртуальному каналу, взятый по модулю 2 (т. е. один бит). Информация о подтверждении добавляется в кадры, идущие в обратном направлении. Каждый такой кадр фактически содержит информацию для всех восьми виртуальных каналов. В частности, поле из восьми битов в заголовке каждого обратного кадра содержит номер по модулю 2 ожидаемого пакета для каждого виртуального канала.

Одной из привлекательных особенностей этого способа передачи является то, что информация подтверждения повторяется

так часто (т. е. для всех виртуальных каналов в каждом обратном кадре), что требуется относительно немного повторных передач из-за ошибок передачи в обратном направлении. Обычно для каждого кадра с ошибкой в прямом направлении нужна только одна повторная передача. Мы увидим, что протокол ARQ на n шагов назад, который будет обсуждаться ниже, требует больше повторных передач и, таким образом, имеет в некоторой степени более низкую эффективность использования линии. Нежелательная черта протокола ARPANET состоит в том, что пакеты поступают из приемного модуля УЛПД на более высокий уровень не обязательно в том порядке, в котором они поступают на передающий модуль УЛПД. Уровень УЛПД в принципе может восстановить порядок следования пакетов, но так как пакет в одном виртуальном канале может иметь произвольные задержки, потребуется хранить в памяти произвольно большое число более поздних пакетов. В ARPANET не предпринимаются никакие усилия для упорядочивания пакетов на отдельных линиях, так что нет проблем при использовании этого протокола в сети ARPANET. Ниже мы увидим, что отсутствие упорядоченности на линиях порождает некоторое число проблем на более высоких уровнях. По этой причине в более современных сетях поддерживается упорядоченность пакетов и, следовательно, не используется этот протокол несмотря на его эффективность и небольшой объем служебной информации. Для очень плохих линий связи, где эффективность и объем служебной информации весьма важны, выбор этого протокола является разумным.

2.4.3. ARQ на n шагов назад

ARQ на n шагов назад является наиболее широко используемым типом протокола ARQ; он используется в различных стандартных протоколах УЛПД, таких, как HDLC, SDLC, ADCCP и LAPB. Знание значения этих аббревиатур не прибавляет ясности, но фактически эти стандарты почти одинаковы. Они рассматриваются в разд. 2.6 и некоторые их отличия там упоминаются.

Основная идея алгоритма ARQ на n шагов назад очень проста. Пакеты, приходящие на передающий модуль УЛПД для передачи от A к B , последовательно нумеруются и этот порядковый номер (ПН) посылается в заголовке кадра, содержащего пакет. В отличие от алгоритма с остановкой и ожиданием следующие пакеты посылаются без ожидания запроса на очередной пакет. Приемный модуль УЛПД в точке B посылает номер запроса (НЗ) обратно в узел A , запрашивая пакет НЗ и подтверждая все пакеты до НЗ.

Число шагов назад n в протоколе ARQ на n шагов назад является параметром, который определяет, сколько идущих

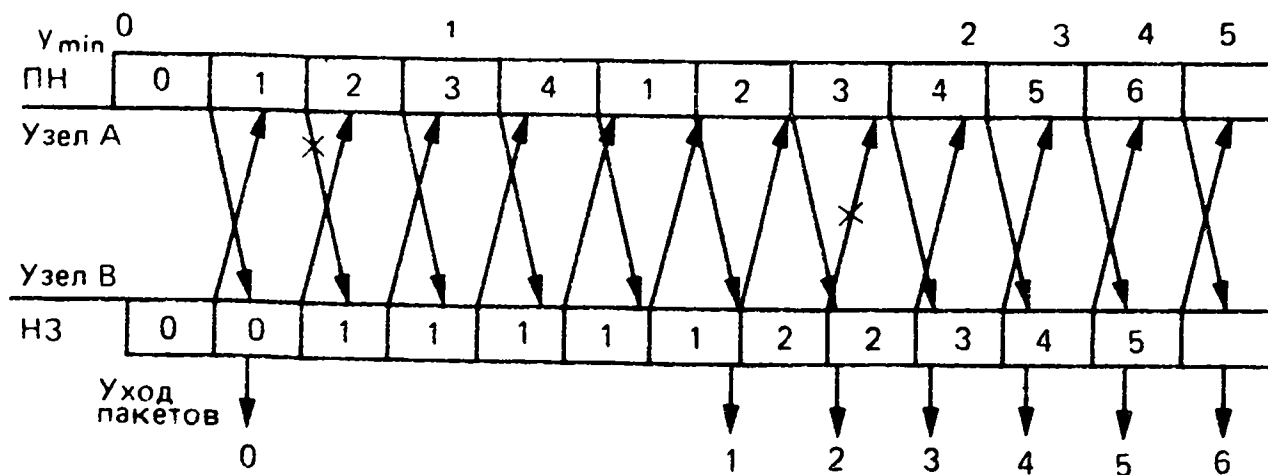


Рис. 2.21. Протокол на четыре шага назад. Номера ПН показаны для кадров, идущих из A в B , а номера запросов НЗ — в кадрах из B в A . Заметим, что ошибка во втором кадре вынуждает узел A повторно передавать пакет 1 после пакета 4. Ошибка в обратном кадре, который передает $НЗ = 2$, не вызывает осложнений.

друг за другом пакетов могут быть посланы при отсутствии запроса на новый пакет. Более точно узлу A не разрешается посылать пакет $i + n$, пока пакет i не получит подтверждение (т. е. пока не будет запрошен пакет $i + 1$). Для любого данного момента времени пусть y_{min} обозначает номер последнего запроса НЗ, который в A принят от B . Тогда в системе с ARQ на n шагов назад из узла A могут посылаться пакеты с номерами в «окне» от y_{min} до $y_{min} + n - 1$ и не могут посылаться пакеты с большими номерами. По мере того как запрашиваются пакеты со следующими, более высокими номерами, y_{min} возрастает и это окно скользит вверх; таким образом, протоколы ARQ на n шагов назад часто называют протоколами ARQ со *скользящим окном*.

Работа приемного модуля УЛПД (узел B) в протоколе ARQ на n шагов назад, по существу, такая же, как и в протоколе с остановкой и ожиданием. Пакеты принимаются по порядку и номер следующего запрашиваемого пакета НЗ посылается в заголовке каждого пакета обратного направления. Пока предположим, что НЗ и ПН могут быть произвольно большими целыми числами; позже будет показано, что они могут быть переданы по модулю m , если $m > n$. Таким образом, когда B принимает пакет, содержащий ошибки, он не может принять никакие пакеты с большими номерами до тех пор, пока A не вернется назад и не повторит передачу пакета, который был принят с ошибкой.

На рис. 2.21 показана работа алгоритма для случая возвращения назад на 4 шага. Рассмотрим поток пакетов из A в B . Порядковые номера ПН показаны для кадров, идущих из A в B , а номера принятых пакетов НЗ¹⁾, которые обеспечивают подтверждение для потока из A в B , показаны в кадрах обратной

¹⁾ Здесь и иногда далее не делается различия между тем, что НЗ — это номер запрашиваемого пакета, и тем, что НЗ показывает номер последнего принятого пакета, для которого ЦИП удовлетворяется. — Прим. ред.

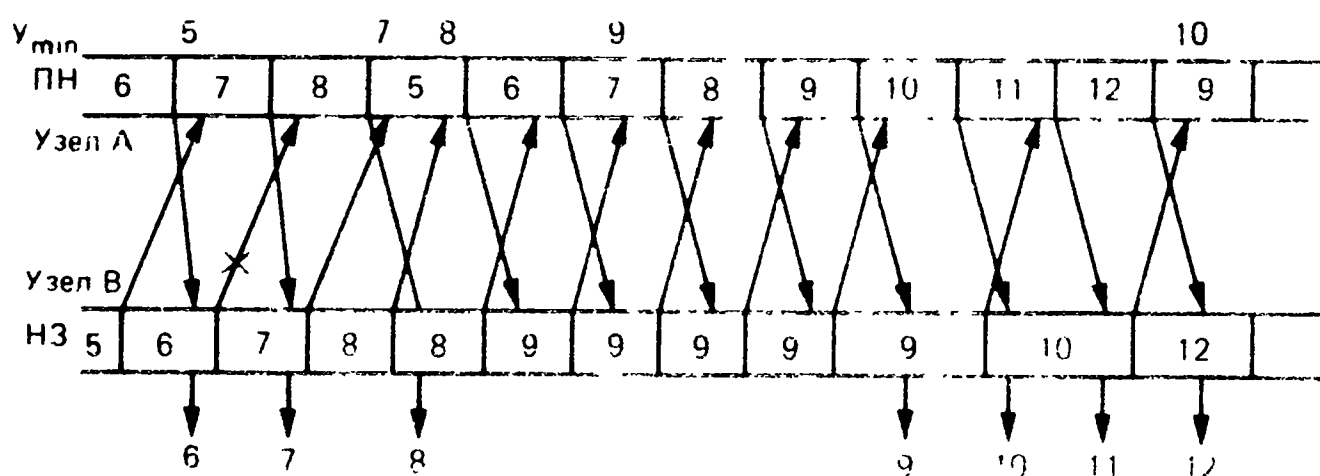


Рис. 2.22. Продолжение рис. 2.21 для ARQ на четыре шага назад. Заметим, что ошибка в обратном кадре, передающем НЗ = 6, заставляет узел А перейти обратно к передаче пакета 5. Заметим также, что длинные обратные кадры заставляют узел А вернуться назад и повторно передавать пакет 9.

связи из B в A . Предположим, что НЗ помещается в заголовке кадра, так что, когда пакет безошибочно принимается в B , новый НЗ отправляется обратно в следующем кадре обратного потока. Этот НЗ не может быть использован в A до тех пор, пока ЦИП, которая находится в конце данного обратного кадра, не будет выполнена. Рассмотрим, в частности, подтверждение для пакета под номером 0; пакет принимается в B в то время, когда на B посылается собственный второй кадр, и, таким образом, пакет подтверждается (с помощью НЗ = 1) в третьем кадре узла B . В узле A нельзя «верить» информации до тех пор, пока в конце кадра не будет проведена ЦИП; в это время y_{\min} изменяется, как показано на рисунке.

Заметим, что, когда при передаче из A ошибка возникает во втором окне, из узла A продолжают передаваться новые пакеты до тех пор, пока это возможно, а потом повторяется пакет с номером 1 после пакета с номером 4 (как ни странно, но это называется движением назад на четыре шага от пятого пакета, который обычно следует за четвертым пакетом, к первому). Таким образом, узел A повторяет передачу четырех пакетов из-за одной ошибки.

Далее, рассмотрим ошибку в первом обратном кадре, который передает НЗ = 2; это не вызывает сложностей для прямого трафика, так как следующий обратный кадр поступает перед тем, как из узла A потребуются идти назад. На рис. 2.22, который является продолжением рис. 2.21, показан пример другой ошибки, ошибки в первом кадре, идущем из B и передающем НЗ = 6; в этом кадре узлу A необходимо возвращаться назад.

В этом примере, в то время когда из узла A повторно передавался пятый пакет, из узла B поступил НЗ = 7. В некоторых вариантах протокола ARQ на n шагов назад (один из которых показан на рисунке) будет в таких ситуациях неразумно передаваться за пятым пакетом шестой, а в других вариантах про-

токала будет осуществляться прыжок вперед и посылаться следующим седьмой пакет, как было запрошено. Наконец, в конце диаграммы видно, что узел A возвращается назад из-за некоторых длинных кадров из B , которые задерживают информацию о подтверждении. Все повторные передачи на рис. 2.22 могли бы быть устранены выбором большего значения n . Эти аспекты эффективности протокола ARQ на n шагов назад исследуются позже, а сначала опишем протокол подробнее и покажем, что он работает безошибочно (корректно).

Существует много вариантов протокола ARQ на n шагов назад. Один из них показан на рис. 2.22. В другом более важном варианте используются тайм-ауты или дополнительная информация, идущая по обратной связи, для определения, когда возвращаться назад и передавать повторно. Мы хотим показать корректность работы целого класса протоколов. Предположим, что кадры могут быть потеряны или произвольно задержаны и что все кадры, принятые как безошибочные, являются действительно безошибочными и поступают в порядке их передачи. Будем считать также, что ПН и НЗ могут быть произвольными целыми числами; интересный для практики случай, когда ПН и НЗ посылаются по модулю некоторого числа, рассматривается ниже. Сейчас приведем инструкции, зависящие от n , которыми руководствуется передающий модуль УЛПД, а затем дадим пояснения.

Инструкции для передающего модуля УЛПД

В начале переменная y_{\min} принимает значение, равное нулю. Всякий раз, когда безошибочный кадр принимается по обратной линии передающим модулем УЛПД, значение y_{\min} обновляется и становится равным значению НЗ в этом кадре. Существует также переменная y_{\max} с начальным значением 0; после начала передач пакетов y_{\max} поддерживается на единицу больше, чем номер наибольшего уже посланного нумерованного пакета. Когда новый кадр готовится для передачи, пакет, помещаемый в кадр, должен иметь номер $z \geq 0$ из интервала (рис. 2.23)

$$y_{\max} - n \leq z \leq y_{\min} + n - 1. \quad (2.24)$$

Порядковый номер ПН z посылается в заголовке кадра. Новые пакеты поступают по запросу или по прибытии из следующего более высокого уровня и нумеруются последовательными номерами, начиная с нуля. Пакеты, имеющие порядковые номера из интервала (2.24), помещаются в буфер внутри модуля УЛПД. Наконец, существует некоторое $T > 0$, такое, что если имеется пакет с номером y_{\min} , он будет передаваться по меньшей мере каждые T секунд до тех пор, пока не изменится y_{\min} (т. е. пока не подтвердится пакет y_{\min}).

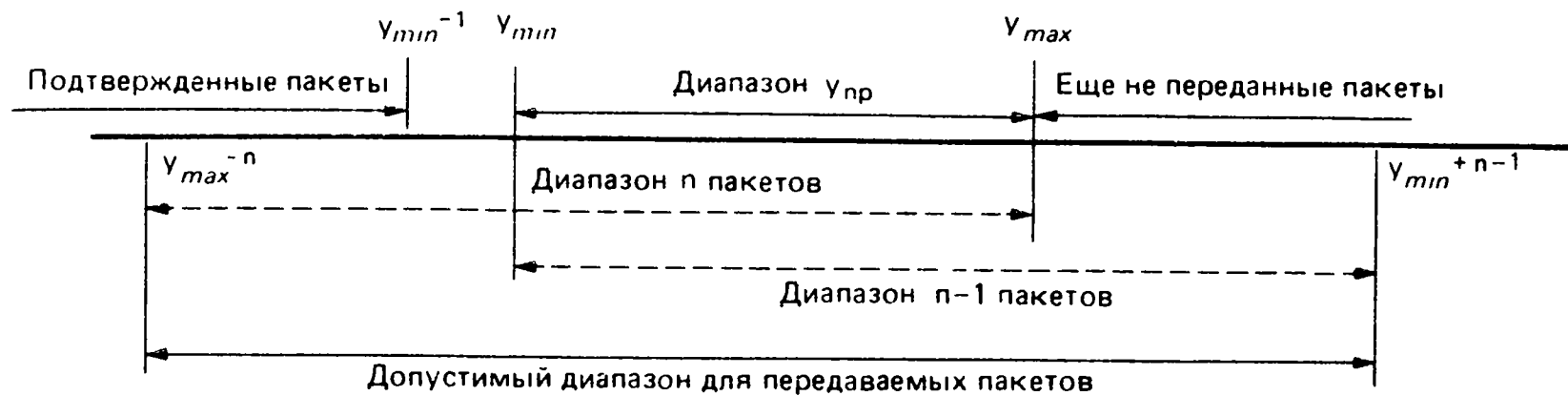


Рис. 2.23. Допустимый диапазон номеров пакетов в системе ARQ на n шагов назад.

y_{\min} — наименьший номер еще не получившего подтверждения пакета, $y_{\max} - 1$ — наибольший номер уже посланного пакета. Ожидаемый пакет в приемном УЛПД лежит между y_{\min} и y_{\max} . Верхней штриховой линией указан диапазон от наименьшего допустимого возможного номера передаваемого пакета до наибольшего возможного ожидаемого номера. Нижняя штриховая линия является диапазоном от наименьшего ожидаемого номера до наибольшего передаваемого номера.

Смысл принятого выше обозначения y_{\max} следующий. Предположим, что передающий модуль УЛПД посылает из A в B ; так как пакет с номером y_{\max} никогда еще не посылался, то y_{\max} является верхней границей для номеров пакетов, ожидаемых в узле B . Аналогично, как отмечалось выше, y_{\min} является нижней границей для номеров пакетов, ожидаемых в узле B . Ниже будет показано, что y_{\min} никогда не может убывать и поэтому формула (2.24) утверждает, что самый большой номер уже переданного пакета никогда не может превышать $y_{\min} + n - 1$. Таким образом, y_{\max} должен удовлетворять

$$y_{\max} \leq y_{\min} + n. \quad (2.25)$$

Это означает, что нижний предел для z в (2.24) всегда меньше или равен y_{\min} ; таким образом, он всегда позволяет передачу пакета y_{\min} . Заметим, что в примере на рис. 2.22 A возвращается назад к $y_{\max} - n$; таким образом, нижний предел в (2.24) позволяет в узле A после движения назад сохранить посылаемые пакеты в упорядоченном виде, даже когда y_{\min} прыгает к более высокой величине.

Подобные правила позволяют A первично и повторно посылать пакеты в любом порядке, если они лежат в пределах интервала (2.24) (рис. 2.23). Это обеспечивает гибкость использования информации, касающейся следования по времени, или вспомогательной информации, идущей по обратной связи, для увеличения эффективности алгоритма, не вызывая беспокойства о потере его корректности (если предварительно установлено, что протоколы из рассматриваемого класса являются корректными).

Ограничение временного характера, заключающееся в том, что пакет с номером y_{\min} посылается по меньшей мере раз в каждые T секунд, является в некотором смысле искусственным; его цель

в исключении протоколов, которые *никогда* не возвращаются назад и не передают требуемый пакет. Заметим, однако, что более высокий уровень в течение произвольного долгого времени может не иметь новых пакетов для передачи. Таким образом, если последний пакет из более высокого уровня имеет номер y_{\min} , то указание ограничения (так же как и здравый смысл) запрещает ожидание до того момента, когда из более высокого уровня поступит пакет с номером $y_{\min} + n - 1$. Другими словами, чистая стратегия ARQ на n шагов назад, которая возвращается назад к y_{\min} только после передачи $y_{\min} + n - 1$, не будет работать так, как нужно, если более высокий уровень может исчерпать свой запас пакетов; такая стратегия исключается вышеприведенными предположениями, касающимися следования по времени. В протоколах ARQ на n шагов назад требуются все-таки какие-то тайм-ауты.

Наконец, заметим, что НЗ сравнивается с y_{\max} для *каждого* безошибочного обратного кадра. Из задачи 2.19 видно, что протокол может иметь тупик, т. е. никакие дальнейшие пакеты уже не смогут приниматься, если НЗ сравнивается только для кадров, содержащих ожидаемый пакет в обратном направлении.

Инструкции для приемного модуля УЛПД

Приемный модуль имеет переменную $y_{\text{пр}}$, начально установленную в 0. Когда появляется безошибочный кадр, порядковый номер ПН в кадре сравнивается с $y_{\text{пр}}$. Если $\text{ПН} = y_{\text{пр}}$, то пакет, который передавался в кадре, считается принятым; он выдается на более высокий уровень, а переменная $y_{\text{пр}}$ получает приращение. Самое последнее значение $y_{\text{пр}}$ используется как принятый номер НЗ в каждом кадре на обратной линии. Наконец, существует некоторое $T > 0$, такое, что по крайней мере один обратный кадр посылается каждые T секунд. Это означает, что если модуль УЛПД не имеет пакетов для передачи, он тем не менее должен посылать номера принятых пакетов по крайней мере каждые T секунд для подтверждения обратного потока, для этого требуется некоторый тип управляющего кадра, не содержащего пакета с данными. То, что $y_{\text{пр}}$ не убывает с течением времени, а обратные кадры, несущие НЗ, поступают в правильном порядке, обеспечивают то, что y_{\min} также не убывает с течением времени.

Теперь покажем, что этот класс протоколов является корректным в том смысле, что если приемный модуль УЛПД ожидает пакет с номером $y_{\text{пр}}$ и если этот пакет имеется в наличии на передающей стороне, то он будет в конечном счете верно доставлен на более высокий уровень на принимающей стороне. Предположим, что вероятность ошибки в любом кадре, передаваемом в одном из двух направлений, меньше некоторой константы p , которая строго меньше, чем 1.

В то время когда принимающая сторона ожидает кадр с ПН $= y_{\text{пр}}$, кадры на обратном направлении содержат НЗ $= y_{\text{пр}}$ и передаются по крайней мере каждые T секунд. С вероятностью 1 один из этих кадров в конце концов принимается безошибочно (при условии, что $y_{\text{пр}}$ не меняется первым). В этот момент y_{min} на передающей стороне становится равным $y_{\text{пр}}$ и это равенство сохраняется до тех пор, пока $y_{\text{пр}}$ не изменится (т. е. до тех пор, пока верно не будет принят пакет). Передающая сторона должна затем передавать пакет $y_{\text{пр}}$ по крайней мере раз в каждые T секунд; в конце концов он будет верно принят с вероятностью 1. Таким образом, с учетом данных предположений этот класс алгоритмов является корректным.

Протокол ARQ на n шагов назад по модулю $m > n$

Теперь покажем, что алгоритм остается корректным, если порядковый номер ПН и номер принятого пакета НЗ передаются по модулю m при некотором $m > n$. Интуитивное подтверждение этого утверждения иллюстрируется на рис. 2.23. Одна штриховая линия на рисунке идет от наименьшего возможного номера пакета z к наибольшему возможному $y_{\text{пр}}$, а другая штриховая линия — от наименьшего $y_{\text{пр}}$ к наибольшему z . Величина $|z - y_{\text{пр}}|$ самое большее равна n , поэтому она меньше, чем m . Таким образом, если ПН (здесь это число z по модулю m) равен $y_{\text{пр}}$, он должен быть таким, чтобы $z = y_{\text{пр}}$. Для того чтобы сделать это рассуждение точным, нужно учесть временные положения, а именно что пакет z может быть произвольно задержан в линии.

Сначала предположим, что только ПН передается по модулю m , тогда как НЗ, как и раньше, является неограниченным целым числом. Рассмотрим линию из узла A в B и пусть y_{min} , $y_{\text{пр}}$, y_{max} и номер пакета z , как и раньше, могут быть любыми целыми числами. Инструкции на передающем модуле УЛПД (в узле A) такие же, как и раньше, за исключением того, что ПН $= z$ по модулю m . Инструкции для приемного модуля УЛПД (в узле B) такие же, как и раньше, за исключением того, что ПН $= y_{\text{пр}}$ по модулю m является новым критерием для принятия пакета, выдачи его на следующий более высокий уровень и увеличения $y_{\text{пр}}$. Покажем, что для системы, в которой ПН $= z$ по модулю m , пакеты считаются в узле B принятыми тогда и только тогда, когда они были бы приняты в системе без передачи ПН по модулю (т. е. тогда и только тогда, когда $z = y_{\text{пр}}$).

Пусть t является моментом времени, когда безошибочный кадр поступает в узел B , и пусть $t' < t$ есть момент времени, когда этот кадр был подготовлен для передачи в узле A . Пусть y'_{min} и y'_{max} являются значениями величин y_{min} и y_{max} в момент t' .

Важное соотношение между этими величинами для системы без модуля m имеет вид

$$y'_{\min} \leq y_{\text{пр}} \leq y'_{\max}. \quad (2.26)$$

Первое из этих неравенств справедливо в силу того, что y'_{\min} является последним значением НЗ, принятым в узле A перед моментом t' ; это значение НЗ является предыдущим по сравнению со значением $y_{\text{пр}}$ в узле B , которое меньше или равно значению $y_{\text{пр}}$ в момент t . Второе неравенство справедливо в силу того, что пакет с номером y'_{\max} не может быть послан из узла A до момента t' ; так как имеется упорядоченность кадров на линии от A к B , пакет с номером y'_{\max} не может достичь узла B перед пакетом с номером z (т. е. до момента времени t).

Для доказательства корректности алгоритма в случае, когда $\text{ПН} = z$ по модулю m , будет использоваться метод индукции по последовательности моментов, когда безошибочные кадры поступают в узел B . Используем неравенства (2.26) как предположение индукции и для того, чтобы показать, что значение $y_{\text{пр}}$ по модулю m равно ПН тогда и только тогда, когда $y_{\text{пр}} = z$. Будет сделан вывод, что система с модулем выполняет те же обновления в момент t , что и система без модуля; это докажет справедливость (2.26) до момента поступления следующего безошибочного кадра в узел B . Укажем, что неравенства (2.26) справедливы до момента поступления первого безошибочного кадра в узел B , что обеспечивает основу для проведения индукции.

Если пакет z подготовлен к передаче в момент t' , из формулы (2.24) следует, что

$$y'_{\max} - n \leq z \leq y'_{\min} + n - 1. \quad (2.27)$$

Используя первое из неравенств в (2.27) с последним неравенством из (2.26), получаем

$$z \geq y'_{\max} - n \geq y_{\text{пр}} - n. \quad (2.28)$$

Использование неравенств (2.26) и (2.27) также дает

$$z \leq y'_{\min} + n - 1 \leq y_{\text{пр}} + n - 1. \quad (2.29)$$

Две штриховые линии на рис. 2.23 иллюстрируют эти соотношения; к первоначальному интуитивному обсуждению добавлена интерпретация соответствующих моментов появления событий на передающем и приемном модулях УЛПД. Объединяя эти два соотношения, получаем

$$|z - y_{\text{пр}}| \leq n < m. \quad (2.30)$$

Это означает, что $z - y_{\text{пр}}$ по модулю m равно 0 тогда и только тогда, когда $z = y_{\text{пр}}$. Таким образом, класс алгоритмов с $\text{ПН} = z$ по модулю m при $n < m$ является корректным.

Наконец, предположим, что номера принятых пакетов НЗ также передаются по модулю m . Таким образом, принимающий модуль УЛПД устанавливает в каждом обратном кадре НЗ равным $y_{\text{пр}}$ по модулю m , используя текущее значение $y_{\text{пр}}$. Правило, по которому обновляется y_{min} на передающем модуле УЛПД, изменяется следующим образом: всякий раз, когда в обратном направлении поступает безошибочный кадр, значение НЗ в этом кадре обновляет y_{min} по правилу:

$$\begin{aligned} &\text{Пока } y_{\text{min}} \text{ по модулю } m \text{ не будет равно НЗ,} \\ &\text{полагать } y_{\text{min}} := y_{\text{min}} + 1. \end{aligned} \quad (2.31)$$

Для того чтобы показать, что корректность сохраняется, используем метод индукции по моментам времени, когда на передающий модуль УЛПД (узел A) поступают безошибочные кадры обратного направления. Пусть t является моментом времени, когда кадр с данным НЗ поступил в узел A , и пусть t' есть момент времени, когда этот кадр был подготовлен в узле B . Пусть $y'_{\text{пр}}$ обозначает $y_{\text{пр}}$ в узле B в этот более ранний момент времени. Соответствующие неравенства, которые справедливы для системы без какого-либо модуля (и, таким образом, также справедливы для системы, где только ПН посылается по модулю), будут иметь вид

$$y_{\text{min}} \leq y'_{\text{пр}} \leq y_{\text{max}} \leq y_{\text{min}} + n. \quad (2.32)$$

Первое неравенство в (2.32) следует из того, что на линии от B к A не нарушается порядок следования кадров; кадр, посланный из B после t' , не может поступить перед t . Второе неравенство в (2.32) следует из того, что пакет с номером y_{max} не может быть послан из A до момента t и определенно не может поступить в узел B до t' . Третье неравенство следует из неравенства (2.25).

Так же как и раньше, используем (2.32) как предположение индукции. Правило (2.31) обновляет значение y_{min} до значения y , удовлетворяющего равенству: y по модулю m равно НЗ $= y'_{\text{пр}}$ по модулю m , а также неравенствам $y_{\text{min}} \leq y \leq y_{\text{min}} + m - 1$. Так как $n < m$ из (2.32) видно, что $y = y'_{\text{пр}}$, это завершает доказательство корректности алгоритма.

Здесь мы убеждаемся, что нет необходимости разрешать числам y_{min} , $y_{\text{пр}}$ и y_{max} принимать любые целые значения; все величины могут храниться в памяти по модулю m .

Эффективность работы алгоритма ARQ на n шагов назад

Повторные передачи, или задержки, возникающие в результате тайм-аутов, появляются в ARQ на n шагов назад по следующим трем причинам: первая — ошибки при передаче в прямом направлении, вторая — ошибки при передаче в направлении

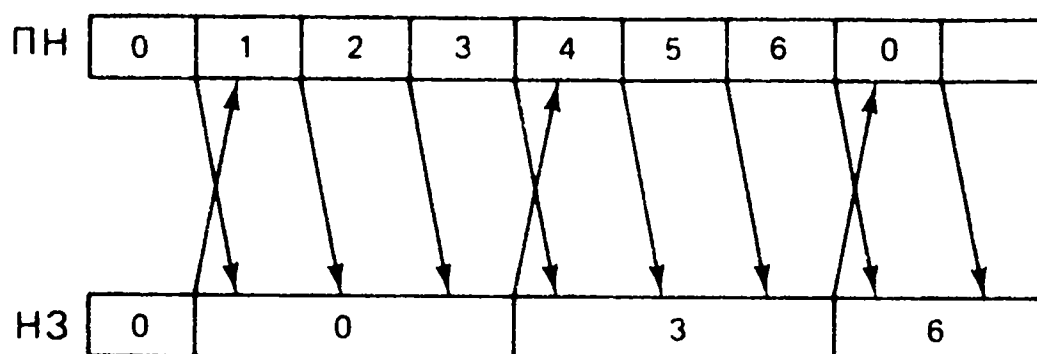


Рис. 2.24. ARQ на семь шагов назад с длинными кадрами в обратном направлении. Заметим, что подтверждение для пакета 1 поступает на передающую сторону после момента, когда пакет 6 прекращает передачу, что заставляет повторно передавать пакет 0.

обратной связи и третья — длина кадров в обратной связи больше, чем в прямом направлении. Эти причины будут обсуждаться в обратном порядке.

Вероятность повторных передач, вызванная длинными кадрами обратного направления, может быть уменьшена путем увеличения n — числа шагов назад. К сожалению, нормальная величина модуля в стандартных протоколах равна $m = 8$, в результате чего n не может быть больше 7. На рис. 2.24 показано, что даже при отсутствии задержки распространения обратные кадры, длиной более чем в три раза большие, чем кадры прямого направления, могут при $n = 7$ вызвать повторные передачи. Задача 2.23 также дает возможность понять, что если кадры имеют длины, которые распределены экспоненциально, с одинаковым распределением в каждом направлении, то вероятность q того, что кадр не получит подтверждения к моменту окончания окна, дается формулой

$$q = (1 + n) 2^{-n}. \quad (2.33)$$

Для $n = 7$ имеем $q = 1/16$. Обычно кадры имеют максимально допустимую длину (и минимальную длину из-за управляющей информации), поэтому в действительности q немного меньше. Однако линии иногда доставляют в одном направлении более длинные кадры, чем в другом, и в этом случае при $n = 7$ могут быть значительные потери эффективности использования линии. Когда время распространения велико по отношению к длине кадра (как, например, для спутниковых линий), то потери в эффективности могут быть вполне серьезными. К счастью, стандартные протоколы имеют возможность выбора другого модуля, например $m = 128$.

Когда ошибки возникают в обратном кадре, информация подтверждения откладывается до передачи добавочного обратного кадра. Эта потеря эффективности линии в одном направлении из-за ошибок в другом направлении может также быть устранена путем выбора достаточно большого n .

Наконец, рассмотрим влияние ошибок в прямом направлении. Если n достаточно большое для того, чтобы избежать повторных передач или задержек, вызванных длинными кадрами или ошибками в обратном направлении, и если передающий модуль УЛПД, прежде чем начать повторную передачу, ждет, когда кончится окно из n пакетов, то большое число пакетов передаются повторно из-за каждой ошибки в прямом направлении. Обычное решение этой проблемы — использовать тайм-ауты. Простейший такой вариант состоит в том, что если пакет не подтверждается в течение фиксированного тайм-аута после передачи, то он передается повторно. Этот тайм-аут должен быть выбран достаточно большим, чтобы включить задержку обработки и время распространения туда и обратно плюс время передачи двух пакетов в обратном направлении, имеющих максимальную длину (один пакет, который содержится в кадре, находящемся в процессе передачи, когда принят пакет, и один, с которым передается новое значение НЗ).

В более сложном варианте в передающем модуле УЛПД известны задержки обработки и время распространения и можно определить, какой обратный кадр должен доставить подтверждение для данного пакета; можно возвращаться назад, если этот кадр не содержит ошибок, а подтверждение не доставил.

В более общем смысле увеличение эффективности работы линии и уменьшение задержки достигаются путем быстрого возвращения назад, когда возникает ошибка в прямом направлении, и предотвращения повторных передач, вызванных длинными кадрами или ошибками в обратном направлении. Одна из возможностей здесь состоит в том, чтобы приемный модуль УЛПД посылал короткие управляющие кадры в ответ на прием кадра с ошибкой. Это позволяет передающей стороне возвращаться назад намного раньше, чем если бы НЗ был просто приставлен к длинному обратному кадру с данными. Другая возможность — поместить НЗ в трейлер обратного кадра перед ЦИП в последний момент после того, как будет передана пакетная часть кадра. Это не может быть сделано в стандартных протоколах УЛПД, хотя это и уменьшило бы задержку обратной связи почти до длины одного кадра.

Практически нетрудно изобрести новые способы уменьшения в протоколах ARQ как задержки обратной связи, так и объема управляющей информации. Однако надо знать, что непросто установить корректность таких протоколов. За исключением специальных применений улучшения должны быть существенными с тем, чтобы перевесить преимущества стандартизации.

2.4.4 ARQ с выборочным повтором

Даже когда избегаются ненужные повторные передачи, протокол ARQ на n шагов назад должен повторно передать по крайней мере одну порцию кадров с задержкой, равной времени рас-

пространения туда и обратно, если единственная ошибка возникает в ожидаемом пакете. Во многих случаях вероятность одной или более ошибок в кадре равна 10^{-4} или меньше; при этом повторная передача многих пакетов из-за одного ошибочного оказывает слабое влияние на эффективность. Однако существуют линии связи, для которых малые вероятности ошибок на кадр очень трудно достижимы даже с исправлением ошибок в модемах. Для других линий (например, спутниковых) число кадров, переданных за время распространения туда и обратно, является очень большим. В этих случаях ARQ с выборочным повтором может использоваться для повышения эффективности.

Основная идея ARQ с выборочным повтором на линии (A, B) состоит в том, чтобы для B (т. е. для приемного модуля УЛПД) считать принятыми пакеты, идущие в произвольном порядке, и запрашивать повторную передачу из A только для тех пакетов, которые принимаются неправильно. Величина n — число шагов назад, или размер окна, указывает, как далеко A может уйти вперед относительно $y_{\text{пр}}$ наименьшего номера пакета, который еще не принят правильно в узле B . Эта стратегия в некотором смысле аналогична ARQ ARPANET, в том смысле, что в них обычно требуется только одна повторная передача при возникновении ошибочного кадра в прямом направлении. Главное отличие состоит в том, что, если в узле B должно производиться упорядочение пакетов, необходимо хранить самое большее n пакетов, а в ARPANET не существует такой границы.

Заметим, что при использовании любого протокола ARQ только безошибочные кадры могут доставить пакет в узел B , и таким образом, если p — вероятность ошибочного кадра, для максимального среднего числа пакетов, доставленных в узел B за один кадр, идущий от A к B , имеем

$$\eta \leq 1 - p. \quad (2.34)$$

В принципе этот предел $(1 - p)$ может быть достигнут протоколом ARQ с выборочным повтором; он иногда называется пропускной способностью идеального выборочного повтора. Аналогично идеальный протокол ARQ на n шагов назад может быть определен как протокол, который повторно передает пакеты с задержкой, равной времени передачи туда и обратно, каждый раз, когда имеются ошибки в кадре, несущем пакет, ожидаемый приемным модулем УЛПД. Пропускная способность этого идеального алгоритма, как сказано в задаче 2.26, равна правой части неравенства

$$\eta \leq (1 - p)/(1 + p\beta), \quad (2.35)$$

где β — среднее число кадров в интервале, равном времени передачи туда и обратно. Это показывает, что увеличение пропускной способности, которое возможно благодаря выборочным повторам,

является значительным, только когда $p\beta$ не мало по сравнению с единицей.

Ограничения для узла A в протоколе ARQ с выборочным повтором такие же, как и в протоколе ARQ на n шагов назад. В узле B устанавливается величина $y_{пр}$, так же как и раньше, и посылается в каждом кадре НЗ $= y_{пр}$ по модулю m обратно в узел A . В узле A считаются принятыми пакеты, отличные от $y_{пр}$, на основе правила, что если $ПН = (y_{пр} + i)$ по модулю m при $0 \leq i < n$, то считается принятым пакет с номером $y_{пр} + 1$. Неравенства (2.28) и (2.29) получаются, как и раньше; они ограничивают номер принятого пакета z интервалом около числа $y_{пр}$. Так как в узле A теперь считаются принятыми пакеты, даже идущие не по порядку, в узле B необходимо уметь определять z по $ПН = z$ по модулю m для всех принятых кадров, а не только для кадров, несущих пакет $y_{пр}$. Так как интервал z в неравенствах (2.28) и (2.29) имеет длину $2n$, потребуем, чтобы m удовлетворяло условию:

$$m \geq 2n \text{ для протокола выборочного повтора.} \quad (2.36)$$

С таким изменением корректность этого класса протоколов доказывается, как и раньше. Однако важным вопросом, относящимся к протоколу с выборочными повторами, является вопрос об его эффективном использовании для достижения пропускных способностей, относительно близких к идеальной $1 - p$. Заметим сначала, что использование только НЗ для доставки информации подтверждения не очень эффективно, так как, если несколько ошибочных кадров появляется в течение одного периода передачи туда и обратно, в узле A не будет известно о втором ошибочном кадре до тех пор, пока не пройдет время, равное времени передачи туда и обратно после повторной передачи первой ошибки. Существует несколько способов доставки в узел A требуемой дополнительной информации подтверждения. Один из них состоит в том, что из узла B передается обратно наименьший номер пакета j , который еще не принят; j должно быть больше, чем $p\beta$ (среднего числа ошибочных кадров в интервале времени, равном времени передачи туда и обратно), но ограничивается сверху объемом служебной информации, которая передается по обратной связи. Другая возможность состоит в том, чтобы посылать НЗ плюс k дополнительных битов (k — заданная величина); каждый бит дает информацию о том, имеет ли место аск или пак для каждого из k пакетов после $y_{пр}$.

Предположим теперь, что обратные кадры несут достаточно информации для A , чтобы определить был или не был пакет успешно принят после средней задержки в B кадров. Типичный алгоритм для узла A при этом состоит в повторении пакетов, как только становится ясно, что предыдущая передача содержала ошибки; если в узле A обнаруживаются одновременно многократ-

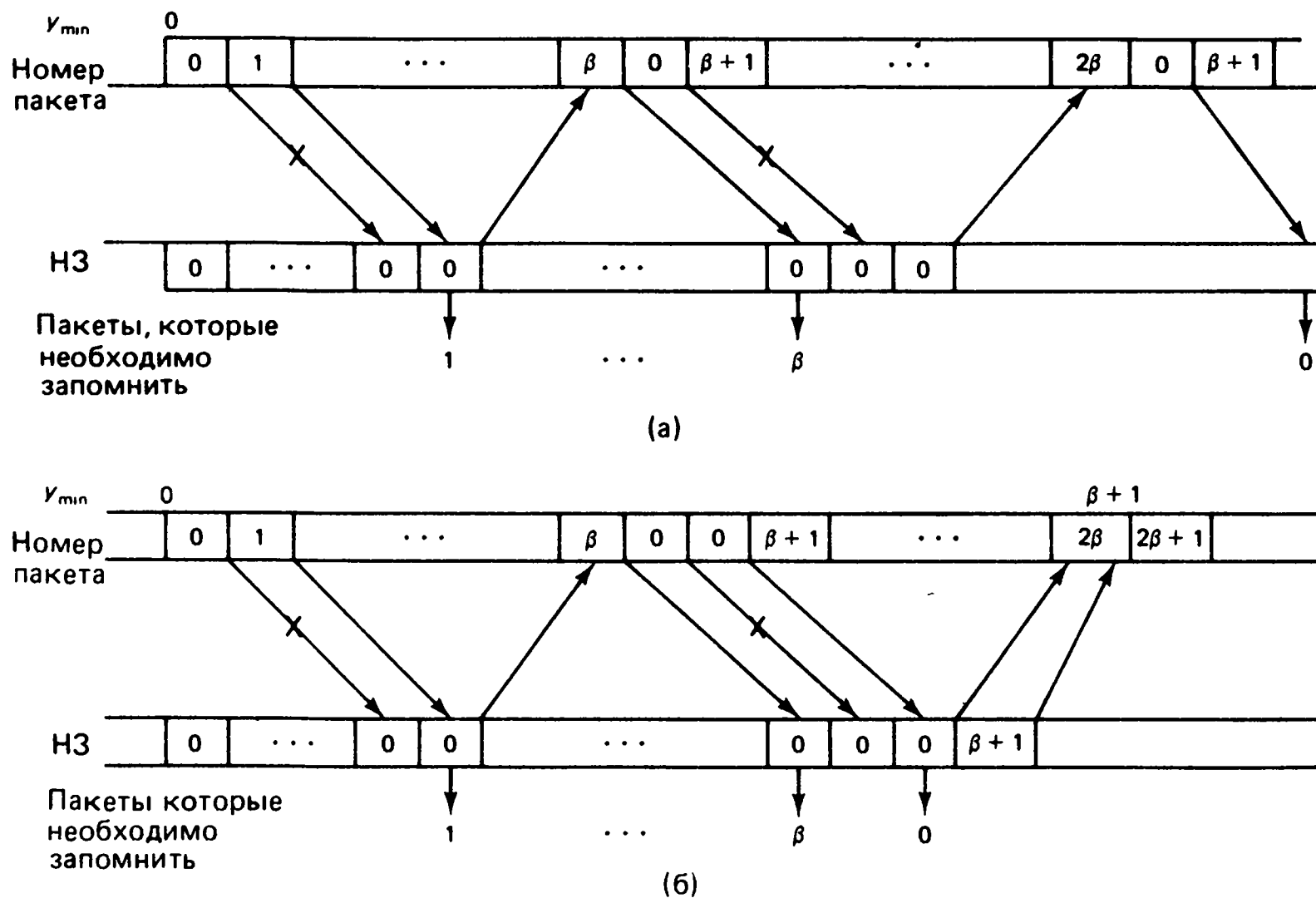


Рис. 2.25. ARQ с выборочным повтором при $n = 2\beta + 2$ и объемом памяти приемника, равным $\beta + 1$ пакету.

a — показаны бесполезные передачи, если данный пакет 0 передавался дважды с ошибками; *b* — показано, что эта трудность устраняется ценой одного дополнительного кадра, если вторая передача пакета 0 дублируется. Обратная связь содержит не только НЗ, но и дополнительную информацию о принятых пакетах.

ные ошибки, они передаются в порядке номеров их пакетов. Если нет запроса на повторную передачу из узла *A*, продолжают передаваться новые пакеты до $y_{\min} + n - 1$. При достижении этого предела узел *A* может сделать тайм-аут или сразу вернуться назад к y_{\min} для того, чтобы последовательно передавать неподтвержденные пакеты.

Узел *A* действует так же, как и идеальная система выборочного повтора до того момента, когда он должен ждать или вернуться назад от $y_{\min} + n - 1$. Однако, когда это случается, пакет с номером y_{\min} уже безуспешно передавался около n/β раз. Поэтому, делая n достаточно большим, вероятность такого возвращения назад можно уменьшить до незначительной величины. Существуют две трудности при очень больших значениях n (в предположении, что пакеты должны быть упорядочены в приемном модуле УЛПД). Первая трудность состоит в том, что в узле *B* должно быть обеспечено хранение всех пакетов, которые считаются принятыми и имеют номера выше, чем $y_{\text{пр}}$. Вторая трудность состоит в том, что большое число хранящихся пакетов задерживается, ожидая пакет $y_{\text{пр}}$.

Объем памяти, необходимый для хранения данных, без большого вреда может быть уменьшен до $n - \beta$, так как всякий раз,

когда имеет место возвращение назад от $y_{\min} + n - 1$, из узла А в течение времени передачи туда и обратно могут не посылать новые пакеты с номерами, большими чем $y_{\min} + n - 1$; в это время из этого узла могли бы повторно передаваться еще неподтвержденные пакеты с номерами от $y_{\min} + n - \beta$ до $y_{\min} + n - 1$. Это означает, что в узле В нет необходимости хранить эти пакеты. Значение n может также быть уменьшено без увеличения вероятности возвращения назад; это делается путем повторной передачи пакета несколько раз подряд, если предыдущая передача содержала ошибки. Например, на рис. 2.25 при $n = 2\beta + 2$ сравниваются двойные повторные передачи с одиночными повторными передачами. Одиночная повторная передача оказывается неудачной с вероятностью p и вызывает β дополнительных повторных передач; двойная повторная передача редко бывает неудачной, но всегда требует одну дополнительную повторную передачу. Таким образом, двойные повторные передачи могут увеличить пропускную способность, если $pn > 1$, а также могут быть полезны в других случаях для уменьшения дисперсии задержки (см. [253] для дальнейшего анализа).

2.5. Кадрирование

Задача кадрирования состоит в том, чтобы решить в приемном модуле УЛПД, где кончаются и начинаются следующие один за другим кадры. В случае синхронного битового тракта иногда существует период холостого заполнения между следующими друг за другом кадрами, так что также необходимо разделять холостое заполнение от кадров. Для прерывающегося синхронного битового тракта холостое заполнение заменяется на периоды молчания, когда никакие биты вообще не прибывают. Это не упрощает проблему, так как, во-первых, следующие друг за другом кадры часто передаются без периода молчания между ними, а, во-вторых, после периода молчания модемы на физическом уровне обычно требуют некоторого холостого заполнения для восстановления синхронизации.

На практике используются три способа кадрирования. Первый — *знаковое кадрирование*, при котором используются специальные управляющие связью знаки для холостого заполнения и для указания начала и конца кадров. Второй — *битовое кадрирование* с флагами, при котором используется специальная последовательность битов, называемая флагом, как для холостого заполнения, так и для указания начала и конца кадров. Третий — кадрирование с измерением длины, при котором указывается длина кадра в поле заголовка. В следующих трех подразделах описываются эти три способа, а в третьем подразделе также дается более общее представление о проблеме. В этих подразделах,

за исключением немногочисленных комментариев, не учитывается возможность ошибок в битовом тракте. В подразд. 2.5.4 совместно рассматриваются задача ARQ и задача кадрирования при наличии ошибок. Наконец, в подразд. 2.5.5 рассматриваются компромиссы, на которые приходится идти при выборе длины кадра.

2.5.1. Знаковое кадрирование

Знаковые коды, такие как ASCII, обычно содержат двоичные кодовые слова не только для знаков клавиатуры и управляющих знаков терминала, но и для различных знаков, управляющих связью. В ASCII все эти кодовые слова имеют длину семь бит обычно с дополнительным битом для проверки, который при осуществлении связи может передаваться, а может и не передаваться (так как циклические избыточные проверки могут быть использованы для обнаружения ошибок в кадрах).

Знак SYN (синхронное заполнение) — один из знаков управления связью; последовательность знаков SYN может быть использована для промежутка холостого заполнения между кадрами, если передающий УЛПД не имеет данных, а синхронный модем требует биты. Знак SYN может быть также использован внутри кадров иногда для синхронизации более старых модемов, а иногда для компенсации задержек, возникающих при поступлении символов данных. Знаки STX (начало текста) и ETX (конец текста) — два других знака управления связью; они используются для указания начала и конца кадра, как показано на рис. 2.26.

В ориентированных на знаки протоколах связи, которые использовались на практике, например в двоичной синхронной системе связи IBM (известной как Bisynch или BSC), все значительно сложнее, но наша цель здесь просто показать, что кадрирование не представляет собой непреодолимую проблему. В приведенном примере существует одна небольшая трудность, которая состоит в том, что заголовок или ЦИП могут случайно содержать знак управления связью. Так как этот знак появляется в известных позициях после STX или ETX, это не вызывает трудностей для приемника. Однако, если пакет может быть произвольной битовой последовательностью, а не последовательностью знаков клавиатуры ASCII, возникает ряд проблем; например, пакет может содержать знак ETX, который может быть интерпретирован как конец кадра. Поэтому для передачи таких данных в ориентированных на знаки протоколах используется специальный режим передачи, который называется *режимом прозрачности*.

Режим прозрачности использует специальный управляющий знак DLE (data link escape — линия передачи данных исчезла). В режиме прозрачности знак DLE ставится перед знаком STX

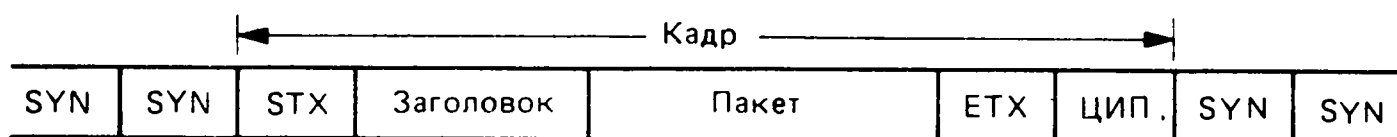


Рис. 2.26. Упрощенная структура кадра при знаковом кадрировании. SYN — синхронное заполнение, STX — начало текста, ETX — конец текста.

для того, чтобы указать начало кадра, идущего в режиме прозрачности. Он также вставляется внутри такого кадра перед использованием знаков управления связью. Знак DLE не вставляется перед случайными появлениями этих знаков как части двоичных данных. Еще одна трудность возникает, если знак DLE появляется в потоке данных, но это решается введением дополнительного DLE перед каждым случайным появлением DLE в потоке данных. Приемный модуль УЛПД потом отбрасывает один DLE из каждой поступившей пары знаков DLE и воспринимает каждый STX или ETX, которому предшествовал неспаренный DLE, как действительное начало или конец кадра. Таким образом, например, DLE ETX (которому предшествует что-то другое, чем DLE) будет воспринят как конец кадра, в отличие от DLE DLE ETX (которому предшествует что-то другое, чем DLE), который будет воспринят как случайное появление DLE ETX в двоичных данных.

При таком типе протокола структура кадра будет иметь вид, показанный на рис. 2.27. Эта структура кадра используется в ARPANET. Она имеет два недостатка, один из которых состоит в некотором чрезмерном использовании служебной информации при кадрировании (т. е. шести знаков, включая два знака SYN, которые необходимо вставлять между кадрами, знаков DLE STX, которые требуются для начала кадра, и знаков DLE ETX, необходимых для его окончания). Другой недостаток состоит в том, что каждый кадр должен состоять из целого числа знаков.

Рассмотрим коротко, что происходит в этом алгоритме при наличии ошибок. ЦИП выполняется для заголовка и пакета кадра, и, таким образом, ошибки в них будут нормально обнаружены. Однако, если ошибка возникает в конце кадра в DLE ETX, приемник не обнаружит конец кадра и, таким образом, не выполнит ЦИП; поэтому в предыдущем разделе мы предположили, что пакеты могут быть потеряны. Аналогичная трудность

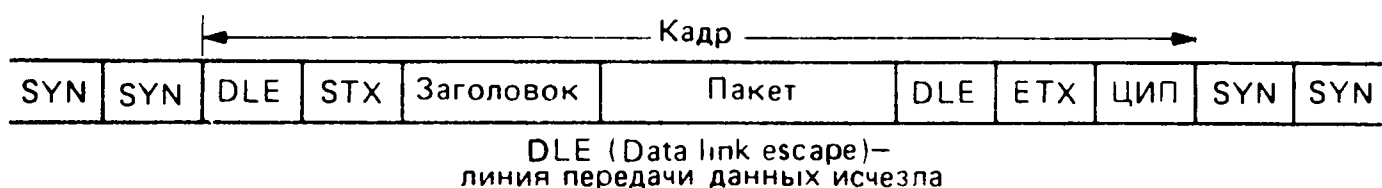


Рис. 2.27. Знаковое кадрирование в режиме прозрачности, которое используется в сети ARPANET.

состоит в том, что ошибки могут вызвать ложное появление DLE ETX в потоке данных; приемник воспримет это как конец кадра и воспримет следующие биты как ЦИП. Таким образом, с вероятностью 2^{-L} , где L — длина ЦИП, случайный набор битов воспринимается как ЦИП и предшествующие данные считаются принятыми как пакет. Некоторые из этих трудностей встретятся в протоколах битового кадрирования (см. также подразд. 2.5.4).

2.5.2. Битовое кадрирование и флаги

При знаковом кадрировании в режиме прозрачности специальные знаковые пары DLE ETX указывают на конец кадра и не встречаются внутри кадра благодаря дублированию каждого знака DLE. Здесь мы обратим внимание на другой подход — использование *флага* на конце кадра. Флагом называется известная последовательность битов, подобная DLE ETX, которая указывает на конец кадра. Для избежания путаницы между ложным флагом, появляющимся внутри кадра и настоящим флагом, указывающим на конец кадра, используется метод *добавления битов*, аналогичный дублированию знаков DLE. Одно важное отличие состоит в том, что при битовом кадрировании кадры могут иметь любую длину (между заданной минимальной и максимальной), а не только равную целому числу знаков. Поэтому мы должны предохранять флаг от ложных появлений, начиная с любой битовой позиции, а не только начиная с границ знаков.

На самом деле флаг — это последовательность битов 01^j0 , где обозначение 1^j означает последовательность из j единиц. Правило добавления битов состоит в том, что добавляется (вставляется) 0 в последовательность данных кадра после каждой серии идущих друг за другом пяти единиц (рис. 2.28). Таким образом, кадр после добавления битов никогда не содержит более пяти последовательных единиц, и флаг в конце кадра в силу своей уникальности оказывается распознаваемым. В приемном модуле УЛПД отбрасывается первый 0 после каждой серии из пяти единиц; если вместо нуля серия из пяти единиц продолжается единицей, это означает, что кадр кончился.

Добавление битов имеет несколько других целей помимо устранения флагов внутри кадра. Стандартные модули УЛПД имеют возможность оборвать кадр, т. е. возможность, при которой кадр может быть оборван путем передачи серии семи или более единиц; кроме того, линия считается пустой, если принимаются подряд 15 или более единиц. Это значит, что 01^6 есть на самом деле последовательность, обозначающая конец кадра. Если 01^6 продолжается нулем, то это флаг, указывающий нормальное окончание кадра; если продолжается единицей, то это указы-

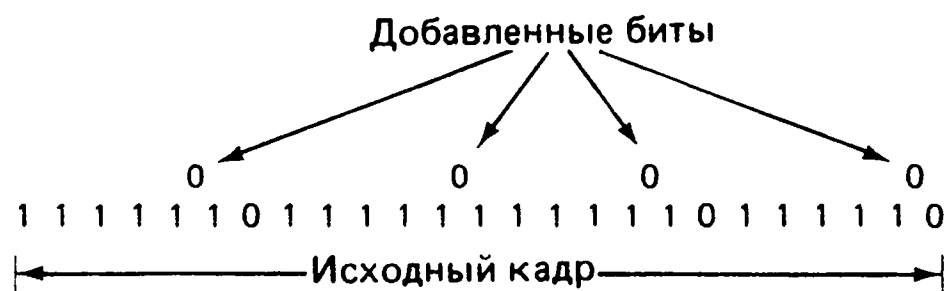


Рис. 2.28. Добавление битов. Нуль вставляется после каждой последовательности из пяти единиц исходного кадра. Флаг 01111110 без вставки передается в конце кадра.

вает на ненормальное окончание. Добавление битов лучше рассматривать как предотвращение появления последовательности 01^6 внутри кадра. Другая незначительная цель вставки битов состоит в том, чтобы оборвать длинные серии единиц, которые вызывают в некоторых более старых модемах потерю синхронизации.

Ясно, что приведенное правило вставки битов устраняет возможность появления последовательности 01^6 внутри кадра, но менее ясно, почему так много битов необходимо вставлять. Например, рассмотрим первый вставленный бит на рис. 2.28. Так как кадр начинается с шести единиц, следующих за опознанным флагом, они не могут быть логически приняты за последовательность 01^6 . Таким образом, помещение бита после пяти единиц в начале кадра не является необходимым (при условии, что правила приемника для отбрасывания вставленных бит соответственно изменены).

Второй вставленный на рисунке бит, очевидно, необходим для того, чтобы избежать появления последовательности 01^6 . Со строго логической точки зрения третий вставленный бит мог бы быть устранен (за исключением случая, когда существует трудность синхронизации в более старых модемах). Задача 2.31 дает возможность понять, как могли бы быть соответственно обновлены правила приемника. Заметим, что уменьшение объема служебной информации путем изменения приведенных правил добавления битов является почти незаметным; цель, преследуемая здесь, — познакомить с правилами добавления битов, а не предложить их изменить.

Четвертый вставленный на рисунке бит определенно требуется, хотя причина этого не совсем ясна. Исходная последовательность $01^5 0$, в которую вставлен этот бит, не может быть ошибочно воспринята как последовательность 01^6 , но приемный модуль УЛПД имеет правило для отбрасывания вставленных бит; он не может различить вставленный 0, следующий за последовательностью 01^5 , от нуля данных, который следует за последовательностью 01^5 . В задаче 2.32 подробно развиваются эти доводы.

Нет ничего особенно магического в последовательности 01^6 как битовой последовательности, используемой для передачи сигнала об окончании кадра (за исключением того, что ее использование предотвращает передачу по линии длинных последовательностей единиц), и фактически любая битовая последовательность с добавленными битами после предпоследнего бита могла бы быть использована (см. задачу 2.33). Такие последовательности с добавленными битами часто оказываются полезными в линиях передачи сигналов.

Рассмотрим избыточность, возникающую из-за использования флага, указывающего на конец кадра. Предположим, что кадр (перед добавлением битов и флага) состоит из независимых, одинаково распределенных двоичных случайных величин с равными вероятностями нуля и единицы. Допустим для увеличения общности, что сигналом окончания кадра является последовательность 01^j , где j — некоторое число ($01^j 0$ — флаг, а 01^{j+1} — индикатор ненормального окончания); $j = 6$ — для стандартного флага. Бит будет добавляться после i -го бита исходного кадра (при $i \geq j$), если последовательность от $i - j + 1$ до i равна 01^{j-1} ; вероятность этого 2^{-j} . Бит будет также добавляться (при $i \geq 2j - 1$), если последовательность от $i - 2j + 2$ до i равна 01^{2j-2} ; вероятность этого 2^{-2j+1} . Мы пренебрегаем последней вероятностью и вероятностью добавления после еще более длинных последовательностей единиц, во-первых, из-за того, что эти вероятности практически очень малы, а во-вторых, из-за того, что эти вставки используются скорее для предотвращения длинных последовательностей единиц, а не для обеспечения кадрирования. Бит в позиции кадра с номером $j - 1$ немного отличается от других битов, так как добавление бита здесь имеет вероятность 2^{-j+1} (т.е. если все первые $j - 1$ бит кадра единицы).

Вспомним, что математическое ожидание суммы случайных величин равно сумме математических ожиданий (независимо от того, зависимы величины или нет). Таким образом, математическое ожидание числа вставок в кадр исходной длины K равно сумме по i математических ожиданий числа вставок после каждого бита i кадра. Однако математическое ожидание числа вставок после данного бита равно вероятности добавления здесь бита. Поэтому математическое ожидание числа вставок в последовательность длины $K \geq j - 1$ равно

$$(K - j + 3) 2^{-j}.$$

Беря математическое ожидание этого выражения по длинам кадров K (в предположении, что все кадры длиннее, чем $j - 1$) и добавляя $j + 1$ бит в конце, получаем математическое ожидание избыточности при кадрировании

$$E \{СЛ\} = (E \{K\} - j + 3) 2^{-j} + j + 1. \quad (2.37)$$

Так как $E\{K\}$ обычно намного больше j , имеем следующее приближение и верхнюю границу (при $j \geq 3$):

$$E\{СЛ\} \leq E\{K\} 2^{-j} + j + 1. \quad (2.38)$$

Один дополнительный бит необходим для различения нормального и ненормального окончания кадра.

Интересно найти целое значение j , которое минимизирует это выражение при заданном значении математического ожидания длины кадра. По мере того как j возрастает от 1 и далее, значение правой части неравенства (2.38) сначала уменьшается, а потом возрастает. Таким образом, оптимальное j равно наименьшему целому j , для которого правая часть неравенства меньше, чем значение этой части при j , увеличенной на единицу, т. е.

$$E\{K\} 2^{-j} + j + 1 < E\{K\} 2^{-j-1} + j + 2. \quad (2.39)$$

Это неравенство упрощается до $E\{K\} 2^{-j-1} < 1$ и наименьшее j — это то, которое удовлетворяет равенству

$$j = \lfloor \log_2 E\{K\} \rfloor, \quad (2.40)$$

где $\lfloor x \rfloor$ — целая часть x . В задаче 2.34 надо понять, что для этого оптимального значения j

$$E\{СЛ\} \leq \log_2 E\{K\} + 2. \quad (2.41)$$

Например, если математическое ожидание длины кадра равно 1000 бит, то оптимальное j равно 9, а математическое ожидание избыточности при кадрировании меньше 12 бит. Для стандартного флага с $j = 6$ математическое ожидание избыточности равно приблизительно 23 бит (едва ли это вызовет желание изменить стандарт).

2.5.3. Поля для указания длин

Основная задача кадрирования состоит в необходимости информирования приемного модуля УЛПД о том, где расположен конец каждого холостого заполнения и где конец каждого кадра. В принципе задача определения конца холостого заполнения тривиальна; холостое заполнение представляется в виде некоторой фиксированной последовательности (например, повторения знаков SYN или повторения флагов) и кончается, когда эта последовательность обрывается. В принципе инвертирование одного бита в этой последовательности является достаточным, хотя на практике холостое заполнение обычно кончается на границе между флагами или знаками SYN.

Так как кадр состоит из произвольной и неизвестной последовательностей битов, то иногда заголовок указывает, где он кончается. Простая альтернатива флагам или специальным знакам

(например, используемая в DECNET) состоит во включении в заголовок кадра поля длины. Предположим, что нет ошибок передачи, тогда приемный модуль УЛПД просто читает эту длину и узнает, где кончается кадр. Если длина представляется обычными двоичными числами, число бит в поле длины должно быть равно по крайней мере $\lfloor \log_2 K_{\max} \rfloor + 1$, где K_{\max} — максимальный размер кадра. Это избыточность при кадрировании таким способом; сравнивая ее с неравенством (2.41) для флагов, можно увидеть, что два способа требуют примерно равной избыточности.

Существуют ли другие методы кодирования длин кадров, которые требуют меньшего среднего числа битов? Ответ на этот вопрос дает теория информации. При любом заданном распределении вероятностей $P(K)$ длины кадра теорема кодирования для источника из теории информации утверждает, что минимальное математическое ожидание числа битов, которые могут закодировать такую длину, равно по крайней мере энтропии этого распределения, которая дается формулой

$$H = \sum_K P(K) \log_2 [1/P(K)]. \quad (2.42)$$

Согласно теореме, в среднем по меньшей мере столько избыточных битов для кадрирования должно быть передано по линии в каждом кадре, чтобы приемник узнал, где кончается каждый кадр. Если $P(K) = 1/K_{\max}$ при $1 \leq K \leq K_{\max}$, то H легко вычисляется и равно $\log_2 K_{\max}$. Аналогично для геометрического распределения длин с заданным $E\{K\}$ энтропия распределения длины при больших $E\{K\}$ приближенно равна $\log_2 E\{K\} + \log_2 e$. Это примерно на $1/2$ бит меньше выражения, стоящего в неравенстве (2.41). Таким образом, для геометрического распределения длин избыточность при использовании флагов при кадрировании, по существу, является минимальной. Геометрическое распределение имеет интересное экстремальное свойство; можно показать, что оно имеет наибольшую энтропию среди распределений вероятностей положительных целых чисел с данным $E\{K\}$ (т. е. оно требует больше бит, чем любое другое распределение).

Общая идея кодирования источника состоит в кодировании более вероятных значений K в короткие битовые последовательности, а менее вероятных в длинные битовые последовательности; точнее, желательно кодировать данное K примерно $\log_2 [1/P(K)]$ битами. Для геометрического распределения используется кодирование, известное как унарно-двоичное кодирование. В частности, для данного j длина кадра K представляется в виде

$$K = i2^j + r; \quad 0 \leq r < 2^j. \quad (2.43)$$

Такой код для K состоит из i нулей, затем следует 1 (это унарный код числа i) и обычный двоичный код числа r (использующий

j битов). Например, если $j = 2$, $K = 6$, то K представляется с помощью $i = 1$, $r = 3$, которые кодируются в 0111 (где 01 — унарный код числа $i = 1$, а 11 — двоичный код числа $r = 3$). Заметим, что различные значения K кодируются в различное число битов, но конец кода может быть всегда опознан как j -й по счету битов после первой единицы.

В общем случае при использовании этого кода заданное число K кодируется в последовательность битов длины $\lfloor K/2^j \rfloor + 1 + j$. Если символом целой части пренебречь и взять математическое ожидание величины K , то

$$E \{СЛ\} = E \{K\} 2^{-j} + 1 + j. \quad (2.44)$$

Заметим, что правая часть этого равенства совпадает с правой частью неравенства (2.38) для флаговой избыточности. Это выражение снова минимизируется путем выбора $j = \lfloor \log_2 E \{K\} \rfloor$. Таким образом, унарно-двоичное кодирование длины и кадрирование с помощью флага требуют, по существу, минимально возможной избыточности при кадрировании в случае геометрического распределения и небольшой избыточности в случае любых других распределений с заданным $E \{K\}$.

2.5.4. Кадрирование при наличии ошибок

Несколько специфических задач возникает, когда ошибки в линии связи искажают информацию, необходимую для кадрирования. Сначала рассмотрим технику с флагом. Если ошибка возникает в флаге на конце кадра, то приемник не обнаружит конец кадра и не будет выполнять ЦИП. В этом случае, когда обнаруживается следующий флаг, на приемной стороне предполагается, что ЦИП находится в позиции перед флагом. Эта последовательность, воспринятая как ЦИП, будет на самом деле ЦИПом следующего кадра, но приемник интерпретирует ее как ЦИП для того, что было передано в течение двух кадров. В другом случае, если некоторое холостое заполнение продолжает кадр, в котором флаг был потерян, последовательность, воспринимаемая как ЦИП, может включать искаженный ошибкой флаг. В любом случае то, что было понято как ЦИП, по существу, является случайной последовательностью битов по отношению к тому, что воспринимается как предшествующий кадр, и приемник не может обнаружить ошибки с вероятностью, по существу, равной 2^{-L} , где L — длина ЦИП.

Другая возможность состоит в том, что ошибка внутри кадра превращает последовательность битов во флаг, как показано для флага 01⁶0:

0 1 0 0 1 1 0 1 1 1 0 0 1 ...	(посылаемая)
0 1 0 0 1 1 1 1 1 1 0 0 1 ...	(принимаемая)

В задаче 2.35 нужно показать, что вероятность того, что это произойдет где-либо в кадре длины K , приближенно равна $(1/32) Kp$, где p — вероятность ошибки в одном бите. В этой схеме, как и раньше, биты перед последовательностью, которая воспринимается как флаг, интерпретируются приемником как ЦИП, и вероятность считать принятым ложный кадр в результате возникновения такой ошибки равна 2^{-L} . Эта проблема часто называется проблемой достоверности данных для модуля УЛПД, так как, даже если ЦИП в состоянии обнаружить любую комбинацию из трех или менее ошибок, единственная ошибка, которая создает или уничтожает флаг, плюс специальная комбинация битов данных для обеспечения выполнения предыдущей ЦИП вызовут необнаруженную ошибку.

Если кадрирование производится с помощью поля длины в заголовке кадра, то ошибка в этом поле длины снова приводит к тому, что приемник воспринимает ЦИП в неправильном месте, и опять ошибочный кадр считается принятым с вероятностью около 2^{-L} . Вероятность такой ошибки меньше при использовании поля длины, чем при использовании флага (так как ошибки могут создать ложный флаг внутри кадра). Однако, после того как ошибка возникнет в поле длины, приемник не знает, где искать следующие кадры. Таким образом, если поле длины используется для кадрирования, некоторые синхронизирующие последовательности должны быть использованы перед началом кадра, когда передающий модуль УЛПД возвращается назад к повторной передаче. (Можно пометить начало каждого окна, но тогда поле длины стало бы почти ненужным.)

Существуют различные частичные решения этих задач, но все они не без недостатков. В сети DECNET используется заголовок фиксированной длины для каждого кадра и длина кадра помещается в этот заголовок; кроме того, заголовок имеет собственную ЦИП. Таким образом, если ошибка возникает в поле длины в заголовке, приемник может обнаружить ее с помощью ЦИП заголовка, место расположения которой известно. Одна трудность этого способа состоит в том, что передатчик и приемник после такой ошибки не будут синхронизованы, так как, даже если ошибка обнаруживается, на приемной стороне не известно, когда начинается следующий кадр. Другая трудность состоит в том, что две ЦИП должны использоваться вместо одной, что отчасти неэффективно.

Существует также способ, при котором поле длины кадра вставляется в трейлер предыдущего кадра. Это устраняет неэффективность подхода в DECNET, но дополнительно требует специальной синхронизирующей последовательности после каждой обнаруженной ошибки. При этом подходе нужен также специальный заголовок кадра, который передается всякий раз, когда неиз-

вестна длина следующего кадра в момент передачи данного кадра.

Другой подход (при любом методе кадрирования) состоит в использовании длинной ЦИП. Это по крайней мере уменьшает вероятность считать принятым ложный кадр, если при кадрировании возникают ошибки. По-видимому, это наиболее подходящий вариант, используемый на практике; существует стандарт ЦИП, равный 32 бит, как один из возможных вариантов в стандартных модулях УЛПД.

Наконец, существует способ, когда кадрирование считается процедурой более высокого уровня, чем ARQ. В такой системе пакеты разделены флагами, а суммарная последовательность пакетов и флагов разбита на сегменты фиксированной длины. Таким образом, разграничение сегментов и разграничение пакетов никак не связаны. Если пакет кончается в середине сегмента и нет в наличии следующих пакетов, то сегмент продолжается холостым заполнением. Эти сегменты затем вводятся в систему ARQ и из-за фиксированной длины сегментов ЦИП будет всегда в известном месте. Недостатком этого способа является большая задержка; пакет не может считаться принятым, пока не будет полностью принят сегмент, содержащий конец пакета. Эта дополнительная задержка возникает на каждой линии по пути следования пакета.

2.5.5. Максимальный размер кадра

Выбор максимальной длины кадра или максимальной длины пакета в сетях передачи данных зависит от многих факторов. Один из наиболее важных факторов состоит в том, что служебная информация включается в каждый кадр. Пусть V будет число битов служебной информации в кадре, включая заголовок кадра, трейлер, флаг и заголовок пакета, но без фиксированной служебной информации, связанной с сообщением. Пусть K_{\max} — максимальная длина пакета, не включая заголовок пакета, а M — длина данного сообщения, включая фиксированную служебную информацию, связанную с сообщением. Тогда общее число битов, которые должны быть переданы для того, чтобы передать одно сообщение, равно

$$\text{Общее число битов} = M + \lceil M/K_{\max} \rceil V, \quad (2.45)$$

где $\lceil x \rceil$ обозначает наименьшее целое число, большее или равное x . Другими словами, по мере того как K_{\max} уменьшается, требуемое число кадров возрастает и служебная информация V должна повторяться для каждого кадра. Этот фактор затем используется как аргумент для увеличения максимальной длины пакета. Другой момент, тесно связанный с этим фактором, состоит в том, что станция (узел или внешний пункт) должна выполнять определенный объем обработки каждого кадра. Будет показано, что все

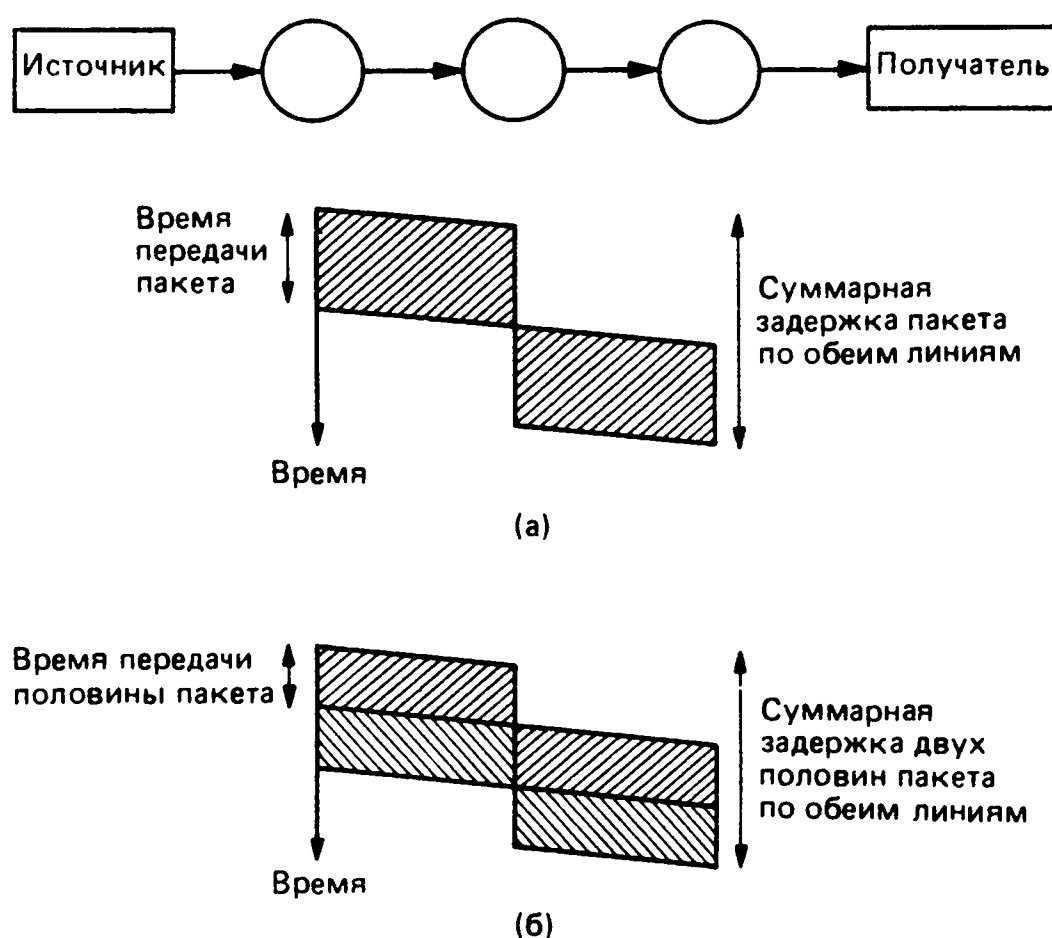


Рис. 2.29. Уменьшение задержки в результате укорочения пакетов показывает преимущество непрерывной передачи пакетов по тракту.

а — общая задержка пакета при передаче по двум пустым линиям равна удвоенному времени передачи пакета по линии плюс суммарная задержка распространения; *б* — когда каждый пакет разделен на два, можно увидеть эффект непрерывной передачи по тракту. Суммарная задержка для двух половин пакета в 1,5 раза больше времени передачи исходного пакета по линии плюс суммарная задержка распространения.

другие факторы, влияющие на выбор максимального размера пакета, используются как аргументы для его уменьшения.

Наиболее важный из этих факторов связан с эффектом непрерывной передачи по тракту (рис. 2.29). Идея состоит в том, что, если сообщение передается как один пакет, полное сообщение должно быть принято на линии (с учетом использования ARQ) перед началом передачи по следующей линии. Однако, если сообщение разбивается на несколько пакетов, более ранние пакеты могут продолжать продвижение по своему пути, когда более поздние пакеты еще передаются по первой линии, что уменьшает общую задержку пакета.

Теперь исследуем совместное влияние объема служебной информации и эффекта непрерывной передачи по тракту. Предположим, что сообщение длины M разбивается в пакеты максимальной длины с последним пакетом обычно меньшей длины. Допустим, что пакеты должны передаваться по j линиям одинаковой пропускной способности и что сеть слабо загружена, так что можно пренебречь ожиданиями в узлах, которые вызваны другими потоками. Мы пренебрегаем ошибками на линиях (которые будут обсуждены позже) и не учитываем время распространения (которое не зависит от максимальной длины пакета). Тогда общее время T , требуемое для передачи сообщения получателю, равно

времени доставки первого пакета по первым $j - 1$ линиям плюс время передачи полного сообщения по последней линии (т. е. если не последний кадр заканчивает передачу по линии, следующий кадр всегда готов начать передачу по этой линии). Пусть C — пропускная способность каждой линии в битах за секунду, так что TC — число передаваемых битов, требуемых для доставки сообщения. Тогда

$$TC = (K_{\max} + V)(j - 1) + M + \lceil M/K_{\max} \rceil V. \quad (2.46)$$

Если взять математическое ожидание этого выражения по длинам сообщений M , получим приближенно, что

$$E \{ \lceil M/K_{\max} \rceil \} = E \{ M/K_{\max} \} + \frac{1}{2}$$

(это приемлемо, если распределение M однородно на промежутках из K_{\max} бит). Тогда

$$E(TC) \approx (K_{\max} + V)(j - 1) + E \{ M \} + E \{ M \} V/K_{\max} + V/2. \quad (2.47)$$

Если пренебречь тем, что K_{\max} должно быть целым, можно продифференцировать это выражение по K_{\max} для того, чтобы найти величину K_{\max} , которая минимизирует $E \{ TC \}$. В результате получаем

$$K_{\max} \approx \sqrt{\frac{E \{ M \} V}{j - 1}}. \quad (2.48)$$

Это значение K_{\max} является компромиссным между объемом служебной информации и эффектом непрерывной передачи по тракту. По мере того как увеличивается объем служебной информации V , величина K_{\max} также увеличивается, а с увеличением длины пути j величина K_{\max} должна уменьшаться. Здесь уместно напомнить одну практическую деталь: задержка при передаче файлов часто менее важна, чем при передаче других сообщений, так что передача файлов должна, вероятно, учитываться с меньшим весом, чем другие сообщения, при оценке $E \{ M \}$; таким образом, это дает несколько меньшее значение K_{\max} .

С увеличением нагрузки в сети эффект непрерывной передачи по тракту сохраняется, несмотря на то что пакеты обычно должны стоять в очереди в узле перед началом передачи. Однако при больших нагрузках влияние объема служебной информации становится более значительным из-за роста числа битов, которые должны передаваться при малом размере пакета. Кроме того, существуют несколько других причин при больших нагрузках, которые используются как аргументы для уменьшения размера пакета. Одна из этих причин — эффект «медленного грузовика». Если много пакетов с совершенно различными длинами передаются

по одному и тому же пути, то короткие пакеты скапливаются позади длинных пакетов из-за большой задержки передачи длинных пакетов на каждой линии; это аналогично скоплению автомобилей за медленно идущим грузовиком на однополосной дороге. Аналогичное влияние на задержку ожидания в очереди для пакетов из разных путей следования будет рассмотрено в гл. 3; большая дисперсия длины пакетов (которая возникает при большом максимальном размере пакета) увеличивает задержку.

В подразд. 2.4.3 было показано влияние большой дисперсии длины кадров на системы ARQ на n шагов назад. Большая дисперсия либо увеличивает число пакетов, которые должны быть повторно переданы, либо увеличивает незанятое время. Это опять является причиной выбора малого максимального размера пакета. И наконец, существует эффект ошибок передачи. Длинные кадры имеют более высокую вероятность ошибки, чем короткие кадры. Для большинства используемых линий связи (за исключением радиолиний) вероятность ошибки при разумных размерах кадров имеет порядок 10^{-4} , или меньше, так что этот эффект обычно менее важен, чем другие рассмотренные эффекты. К сожалению, в литературе описано много методов выбора максимальной длины кадра с учетом только этого эффекта. Эти методы можно использовать лишь в тех специальных случаях, когда вероятность ошибки велика.

На практике максимальная длина пакета обычно равна нескольким тысячам битов. Если объем служебной информации можно уменьшить, то разумно перейти к более коротким максимальным длинам пакетов, ослабляя, таким образом, влияние дисперсии пакета, на которое указывалось выше.

2.6. Стандартные модули УЛПД

Существует ряд аналогичных стандартов для уровня управления линией передачи данных, а именно HDLC, ADCCP, LAPB и SDLC. Стандарт HDLC разработан Международной организацией стандартов (МОС), ADCCP — Американским национальным институтом стандартов (АНИС), LAPB — Международным консультативным комитетом по телеграфии и телефонии (МККТТ), а SDLC — фирмой IBM. Стандарты HDLC и ADCCP, которые фактически совпадают, рассматриваются здесь. Они имеют много различных вариантов и режимов работы. LAPB является уровнем УЛПД для X.25, который является первоначальным стандартом для подключения внешнего пункта в подсеть. Как будет описано позже, LAPB представляет собой, по существу, ограниченное подмножество возможностей и режимов стандарта HDLC. Аналогично SDLC, который был предвестником стандартов HDLC и ADCCP, является, по существу, другим подмножеством возможностей и режимов того же стандарта HDLC.

Стандарты HDLC и ADCCP разработаны для различных типов линий связи, включая линии с множественным доступом, двухточечные линии, а также дуплексные (в которых обе станции могут передавать одновременно) и полудуплексные линии (в которых только одна станция может передавать в любой момент). Существуют три возможных *режима* работы, из которых выбирается один, когда линия иницируется для использования.

Первый — *режим нормального ответа* (РНО) предназначен для использования линии в условиях, когда имеется главная и подчиненная ей станция; это типично при связи ЭВМ с одним или несколькими периферийными устройствами (слово «нормальный» имеет только историческое значение). Итак, имеются первичная, или главная, станция (ЭВМ) и одна или несколько вторичных, или подчиненных, станций. Вторичные станции могут передавать кадры только в ответ на команды опроса от первичной станции. Этот режим является разумным в ситуациях множественного доступа, когда команды опроса гарантируют, что только одна станция передает пакет в данный момент (см. гл. 4). РНО также разумно использовать на полудуплексных линиях или для связи с периферией, которая не способна действовать в других режимах.

Второй режим — это *режим асинхронного ответа* (РАО), режим типа «главная станция — подчиненная станция», в котором вторичные узлы не так сильно ограничены в своих действиях. Он широко не используется и не будет обсуждаться в дальнейшем.

Третий режим — *асинхронный сбалансированный режим* (АСР), предназначен для дуплексных двухточечных линий между станциями, которые несут равную ответственность за управление линией. Этот режим представляет огромный интерес применительно к двухточечным линиям в сетях передачи данных. Он применяется в несколько модифицированном виде для связи на основе множественного доступа в локальных сетях. В стандарте LARV используется только этот режим в отличие от SDLC, в котором используются только первые два режима.

Сначала опишем структуру кадра, общую для всех трех режимов и четырех стандартов. Потом более подробно рассмотрим работу в режиме нормального ответа (РНО) и в асинхронном сбалансированном режиме (АСР). Хотя режим АСР представляет больший интерес для нас, некоторые особенности его работы станут более понятны после рассмотрения РНО.

На рис. 2.30 показана структура кадра. Флаг, как и в подразд. 2.5.2, — это последовательность 01^60 , которая используется со вставкой битов внутри кадра (флаг не рассматривается как часть кадра). Каждую пару кадров должны разделять один или несколько флагов. Для ЦИП используется порождающий многочлен ЦИП-МККТТ $g(D) = D^{16} + D^{12} + D^5 + 1$, как было объ-

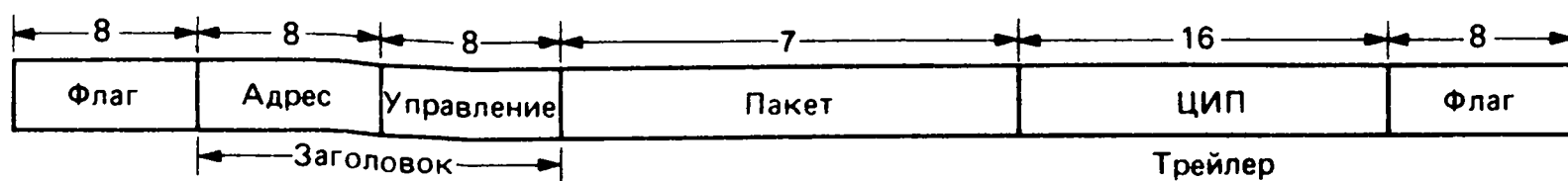


Рис. 2.30. Структура кадра стандартного УЛПД. Адресное поле, поле управления и ЦИП могут быть вместе или по отдельности выбраны увеличенной длины.

яснено в подразд. 2.3.4; ЦИП позволяет проверить весь кадр (не включая флага). Одна модификация состоит в том, что первые 16 бит кадра инвертируются при вычислении ЦИП, хотя при передаче они не инвертируются. Также инвертируется остаток для ЦИП. В приемном модуле УЛПД первые 16 бит кадра аналогично инвертируются для повторного вычисления остатка, который потом инвертируется для сравнения с принятой ЦИП. Одна причина этой модификации состоит в том, чтобы избежать полностью нулевой последовательности (появляющейся обычно, если линия или модем неисправны), которая удовлетворяет ЦИП.

Другая причина состоит в том, чтобы избежать удовлетворения ЦИП, если несколько нулей добавлены в начало или конец кадра или удалены оттуда. Как один из вариантов может использоваться 32-битовая ЦИП; коэффициенты порождающего многочлена в этом случае имеют значения, равные 1 на позициях 32, 26, 23, 22, 16, 12, 11, 10, 8, 7, 5, 4, 2, 1 и 0. Этот 32-битовый ЦИП используется в стандарте IEEE 802 для локальных сетей связи.

Адресное поле обычно состоит из одного байта (восьми битов). Для РНО адрес всегда соответствует адресу вторичной станции, т. е. когда первичная станция посылает кадр на вторичную станцию, она использует адрес вторичной станции, а когда вторичная станция отвечает, она использует свой собственный адрес. Заметим, что этот адрес в сеансе не играет роль адреса получателя, а просто используется для идентификации станций в системе множественного доступа. Для двухточечных линий адресное поле не выполняет свою естественную функцию, хотя позже мы увидим, что в АСР адресное поле выполняет некоторую довольно своеобразную роль. Существует вариант, в котором адресное поле может быть равно произвольному числу байтов (1 и более). Первый бит в байте равен 1, если этот байт является последним адресным байтом, в противном случае первый бит равен 0.

Обычный формат поля управления показан на рис. 2.31. Существуют три различных формата кадра: *информационный*, *супервизорный*, *ненумерованный*. Информационные кадры — это кадры, которые передают пакеты; они обычно используют ARQ на n шагов назад по модулю 8. Как показано на рисунке, номера передачи и приема ПН и НЗ являются частью поля управления.

1	2	3	4	5	6	7	8	
0	ПН			P/F	НЗ			Информационный
1	0	Тип		P/F	НЗ			Супервизорный
1	1	Тип		P/F	Тип			Ненумерованный

Рис. 2.31. Структура поля управления для информационных, супервизорных и ненумерованных форматов кадра.

Супервизорные кадры используются для обратной информации ARQ (т. е. для НЗ), либо когда нет пакетов с данными для передачи, либо когда необходимо быстро получить положительное или отрицательное подтверждение. Наконец, ненумерованные кадры используются для инициирования или прекращения работы линии и для посылки различного типа дополнительной управляющей информации.

Существует вариант, при котором поле управления имеет длину два байта, а ПН и НЗ каждый имеют длину семь битов и передаются по модулю 128. За исключением удлинения ПН и НЗ, этот вариант, по существу, такой же, как и обычный с однобайтовым полем управления, который сейчас будет рассмотрен.

Первый бит поля управления отличает информационный формат от супервизорного и ненумерованного форматов. Аналогично для неинформационных пакетов второй бит отличает супервизорный формат от ненумерованного формата. Пятый бит поля управления — бит *запроса или конца* (P/F). В режиме нормального ответа первичная станция посылает 1 в позиции этого бита для запроса вторичной станции. Вторичная станция может потом ответить одним или большим числом кадров и послать 1 в позиции этого бита для обозначения последнего кадра своего ответа. Эта функция запроса и отклика бита P/F в РНО используется в супервизорных и ненумерованных кадрах, а также в информационных кадрах. Как будет описано позже, в асинхронном сбалансированном режиме бит P/F используется более странным способом.

Существуют четыре типа супервизорных кадров: *готов к приему* (RR), *не готов к приему* (RNR), *отказ* (REJ) и *селективный отказ* (SREJ). Тип кадра кодируется третьим и четвертым битами поля управления. Основная функция всех этих супервизорных кадров состоит в передаче обратной информации ARQ (т. е. они подтверждают все пакеты с номерами, меньшими чем НЗ); все они не содержат информационного поля. RR является обычным ответом, если нет пакета с данными, с которым можно передать НЗ. RNR означает, кроме того, что станция временно не способна принимать дальнейшие пакеты; это позволяет осуществить элементарную

форму управления потоком на линии, а именно если буфер на одном конце линии полностью заполнен принятыми пакетами, на другом конце линии можно прерывать дальнейшие передачи.

Супервизорный кадр REJ посылается для индикации того, что либо кадр принят с ошибками, либо принят кадр с другим порядковым номером, чем ожидалось. Его цель улучшить эффективность протокола ARQ на n шагов назад, как было объяснено в подразд. 2.4.3. Аналогично кадр SREJ обеспечивает примитивную форму выборочного повтора. Он положительно подтверждает все пакеты до НЗ, запрашивает повторную передачу только пакета с номером НЗ, за которым следуют новые пакеты; он не может передать какую-либо дополнительную информацию о требовании многократных повторных передач. Тем не менее эта возможность может быть полезна на спутниковых линиях связи, где имеется очень малая вероятность более чем одного ошибочного кадра за промежуток времени, равный времени передачи туда и обратно. Использование управляющих кадров REJ и SREJ не обязательно, а использование кадров RR и RNR необходимо.

Ненумерованные кадры не переносят порядковых номеров и используются для установления линии связи, разъединения линии и для специальных целей управления. В поле управления имеется пять битов, которые пригодны для распознавания различных типов ненумерованных кадров, но не все они используются. Шесть типов ненумерованных кадров называются обобщенно командами *установления режима*. Существует одна такая команда для каждого режима (РНО, РАО, АСР) и одна для каждого режима, в котором используется расширенное поле управления. Эта команда посылается первичной станцией (в РНО или в РАО) либо любым узлом (в АСР) для инициирования связи по линии. После передачи или приема этой команды станция устанавливает свои номера передачи и приема в нуль. Получатель этой команды должен подтвердить ее с помощью ненумерованного кадра либо с помощью ненумерованного положительного подтверждения, которое означает согласие начать использовать линию, либо с помощью *режима разъединения*, который не разрешает начать использовать линию. Аналогично существует команда разъединения, которая используется для прекращения использования линии; в ответ она тоже требует ненумерованное положительное подтверждение.

На рис. 2.32 показана работа в РНО первичной и двух вторичных станций. Заметим, что первичная станция иницирует связь с каждой станцией отдельно и разъединяется с каждой станцией также отдельно. Заметим, кроме того, что P/F бит устанавливается в 1 в каждом ненумерованном кадре, который требует ответа. Кадры с $P = 1$, идущие к данной вторичной станции, строго

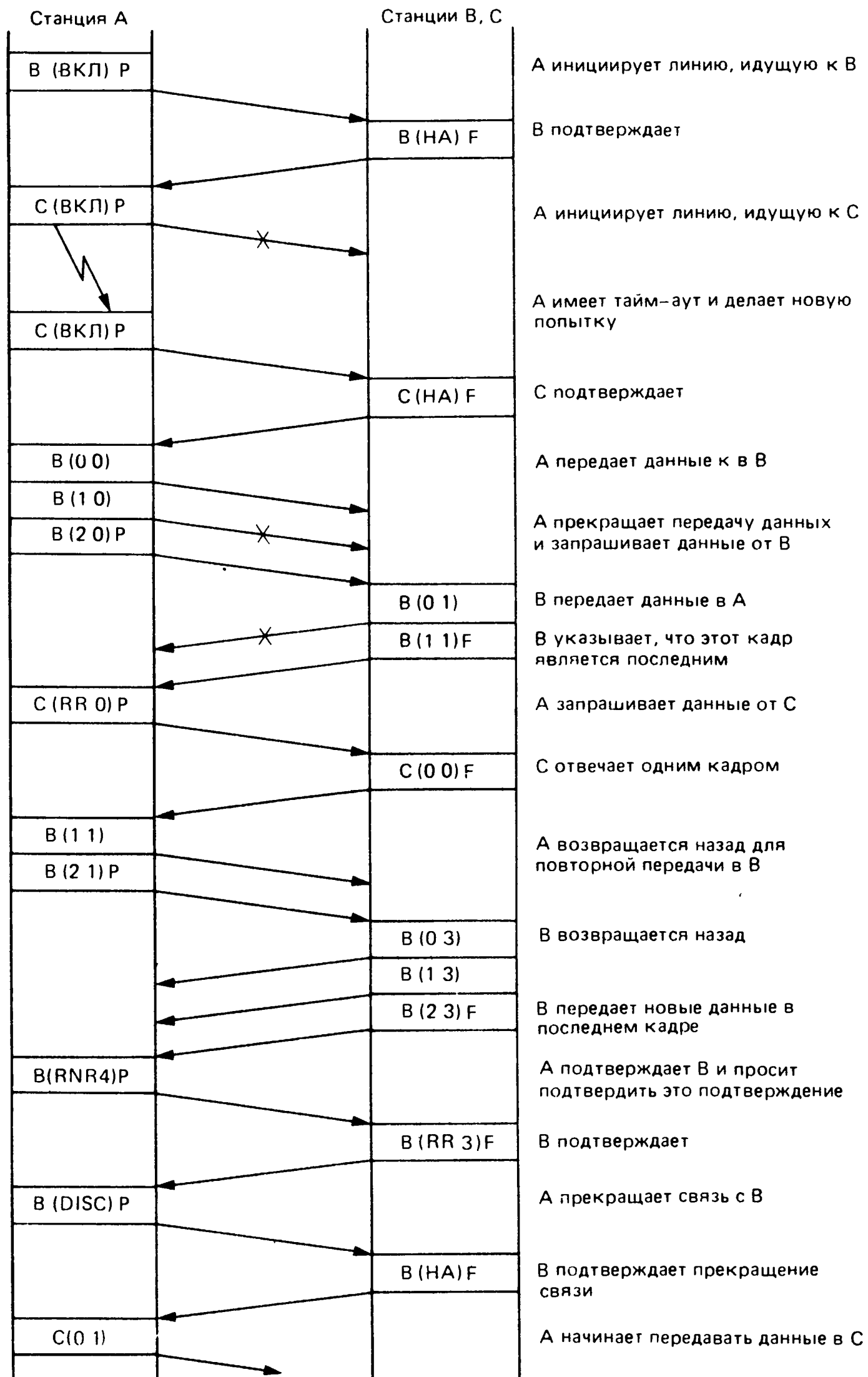


Рис. 2.32. Работа в режиме нормального ответа первичной станции А с двумя вторичными В и С. Команда ВКЛ (SETM) устанавливает режим РНО с ПН = 0 и НЗ = 0 на обеих станциях. Адрес каждого кадра приводится первым, затем (ПН, НЗ) для информационных кадров, (тип, НЗ) для супервизорных кадров и (тип) для нумерованных кадров, после этого следует бит Р/Ф, если он установлен в 1. Заметим, что в узле А используются разные значения ПН и НЗ для станций В и С.

чередуются с кадрами из этой вторичной станции с $F = 1$. Кадры с $P = 1$ могут рассматриваться как посланные в соответствии с протоколом с остановкой и ожиданием, $F = 1$ служит тогда как положительное подтверждение. Это чередование имеет тот же недостаток, что и протокол с ожиданием и остановкой без порядковых номеров, т. е. при наличии ошибок или произвольных задержек подтверждения не могут однозначно соответствовать запросам на передачу.

Трудно определить, сколько вреда причинит этот несовершенный протокол с остановкой и ожиданием. Прежде всего задержки в линии связи не произвольные, а довольно легко определяемые; таким образом, на практике нетрудно выбрать таймауты достаточно длинными для того, чтобы ответы не поступали после них. Далее, указанный протокол ARQ на n шагов назад является корректным при любых предположениях, касающихся моментов событий, и таким образом, даже если запросы и положительные подтверждения данных запутаются, единственным результатом будет потеря эффективности и возникновение иногда передач нескольких вторичных станций одновременно. Наконец, эта путаница может возникнуть при инициировании линии лишь вследствие ошибок в предыдущем разъединении линии, которые могут возникнуть из-за выхода из строя линии или станции, но мы уже знаем, что определенные поломки (например, если ЦИП не обнаруживает ошибки или станции теряют счет порядковых номеров пакетов) приводят к потере пакета в любом случае. Однако, несмотря на все эти разумные объяснения, хорошо, если бы биты запроса подтверждались надлежащим образом и надежно.

В асинхронном сбалансированном режиме (как было принято при разработке протоколов) каждая станция должна иметь возможность работать как первичная и как вторичная (таким образом, простая задача заменяется на более известную, но более сложную). Когда станция работает как первичная, она использует в адресном поле адрес другой станции и бит P/F интерпретируется как бит запроса. Когда станция работает как вторичная, она использует свой собственный адрес, и бит P/F интерпретируется как бит конца передачи. При нормальной работе все информационные кадры и все нумерованные кадры, требующие положительного подтверждения, посылаются, как с первичных станций; все нумерованные кадры, выполняющие роль положительных подтверждений или ответов, и большинство супервизорных кадров (которые положительно подтверждают кадры с данными, переданные с первичной станции) посылаются, как со вторичных станций.

В действительности это означает, что бит P/F есть либо P , либо F в зависимости от адресного поля. Этот бит в сочетании с адресным полем имеет три возможных значения: $P = 1$, требую-



Рис. 2.33. Асинхронный сбалансированный режим (АСР) работы. Команда ВКЛ (SETM) называется командой установки АСР, которая инициирует линию связи с ПН и НЗ, равными 0 на обоих концах линии. Обратите внимание на использование адреса в кадрах различного типа.

щее ответа от другой станции, которая работает как вторичная; $F = 1$, обеспечивающее ответ вторичной станции; и ни то ни другое. Это модифицированное использование бита P/F является неудачным, так как отчасти не соответствует выполнению чередования запрос—конец.

На рис. 2.33 показаны некоторые типичные операции в АСР. Наиболее интересная часть этих операций состоит в передаче предпоследнего кадра станцией А; супервизорный кадр посылается с $P = 1$, требуя, таким образом, подтверждения. Так как станция А выразила нежелание принимать далее пакеты, ответ от станции В означает, что В знает, что пакеты вплоть до пятого включительно считаются принятыми и что никакие дальнейшие пакеты приниматься не будут. Таким образом, в ответ на прием подтверждения об этом А может разъединиться, зная, что станции В точно известно, какие пакеты от В считаются принятыми; А также знает, какие из ее пакетов были приняты, поэтому при разъединении не возникает неопределенности о конечном состоянии любых пакетов, которые находятся либо на станции А, либо на станции В. Заметим, что это «идеальное» разъединение требует некоторых предварительных соглашений (не определяемых стандартом) между А и В. В частности, А должна послать RNR с $P = 1$ и получить положительный ответ перед разъединением. Это позволяет В быть уверенной в том, что А временно не вернется назад к принятым пакетам перед разъединением. «Идеальное» разъединение, конечно, не получается, если линия разъединя-

ется из-за неполадок; в этом случае идея строгого выполнения последовательности RNR, состоящей из положительного подтверждения, разъединения и положительного подтверждения, является нерелевантной.

Другой тип нумерованных кадров — это «отказ от кадра». Он посылается в ответ на принятый кадр с правильной ЦИП, но имеющий недопустимый параметр (например, превышена максимальная длина кадра; кадр короче, чем минимальная длина кадра; НЗ имеет значение не между y_{\min} и y_{\max} или поле управления является недопустимым).

Это означает, что либо возникла необнаруженная ошибка, либо одна из станций неисправна. Такая ошибка находится за пределами области того, для чего разработан ARQ, и, таким образом, она должна быть сообщена всем станциям. Обычный ответ на «отказ от кадра» для противоположной станции состоит либо в повторном иницировании линии с помощью команды запуска, либо путем передачи нумерованной команды повторного запуска. Первая устанавливает НЗ и ПН на каждой станции в 0, а вторая устанавливает ПН в 0 на станции, которая передала дефектный кадр, и НЗ в 0 на приемной станции. Обе станции будут снова начинать передавать пакеты (начиная с первого положительно не подтвержденного), но из-за повторной установки некоторые пакеты могут поступить более чем один раз и с нарушением порядка следования. Это не является недостатком протокола, а только говорит о том, что протокол не может иметь дело с неисправным узлом и необнаруживаемыми ошибками.

2.7. Идентификация сеансов и адресация

Вопрос идентификации сеансов уместно решать на сетевом уровне. Пакет, принятый узлом, должен содержать достаточно информации для того, чтобы в узле можно было определить, как передать его по направлению к получателю. Этот вопрос обсуждается здесь, так как кодирование такой информации очень похоже на кодирование информации кадрирования. Грубый подход к решению этой задачи состоит в том, чтобы заголовок каждого пакета содержал идентификационные номера как пункта отправления, так и пункта назначения, а также дополнительные идентификационные номера, определяющие сеанс на каждом пункте. Этот подход позволяет пакетам одного и того же сеанса проходить в подсети различными путями; он также требует передачи значительного объема служебной информации.

Если в сети используются виртуальные цепи, то оказывается, что в заголовках пакетов требуется передавать значительно меньше служебной информации. При использовании виртуальных цепей каждый (однонаправленный) сеанс, использующий сеть в данное

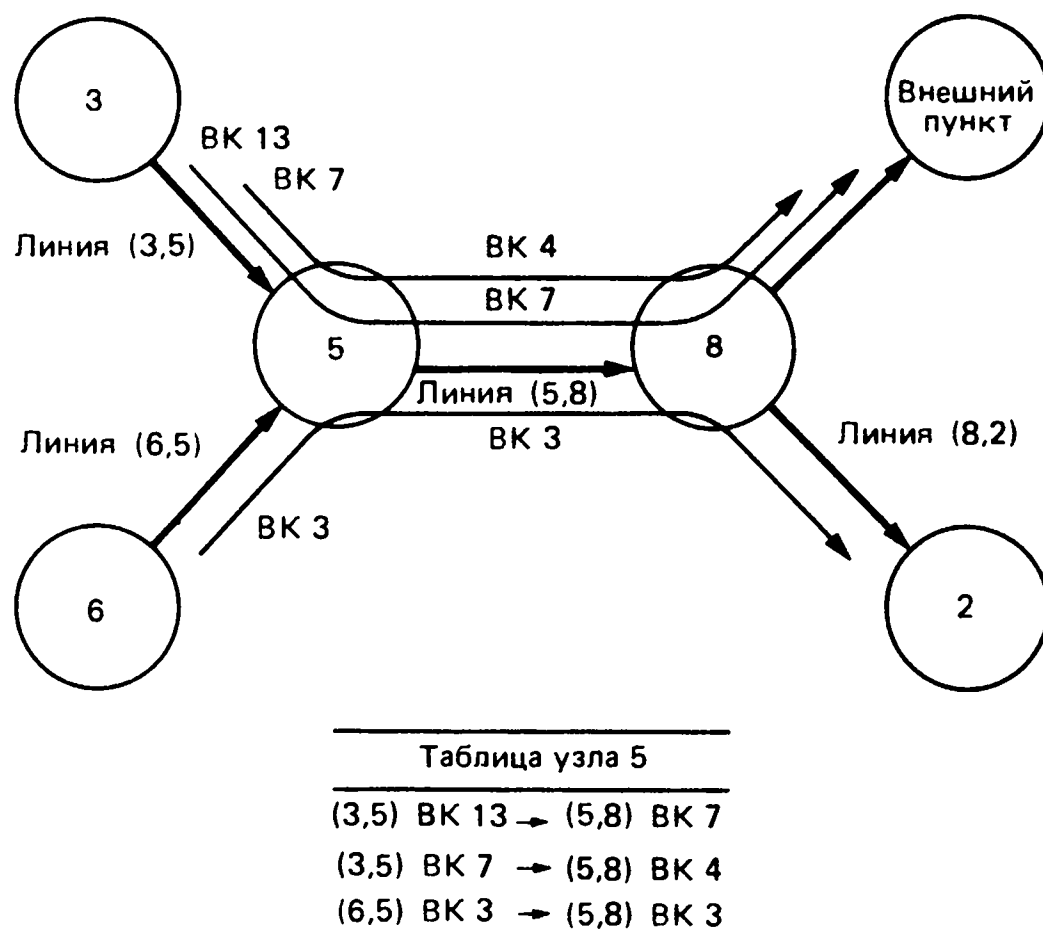


Рис. 2.34. Использование номеров виртуальных каналов (VK) для установления виртуальных цепей в сети. Каждый пакет передает номер своего виртуального канала по линии, а в узлах эти номера (в соответствии с таблицей виртуальных цепей) изменяются для передачи по следующим линиям на пути пакетов.

время, имеет заданный путь через сеть и, таким образом, существует определенный набор сеансов, которые используют каждую линию. Полезно представлять себе каждую линию как набор виртуальных каналов, которые различаются по номерам. Когда устанавливается новый сеанс, путь в сети образуется путем назначения на каждой линии пути одного неиспользуемого виртуального канала для этого сеанса. На каждой станции затем составляется таблица, которая отображает каждый занятый входящий виртуальный канал на каждой линии на соответствующий выходящий виртуальный канал и линию для каждого данного сеанса (рис. 2.34). Например, если данный сеанс имеет путь от узла 3 к 5 и далее к 8, использующий виртуальный канал 13 на линии (3,5) и виртуальный канал 7 на линии (5,8), то узел 5 будет отображать [линию (3,5), виртуальный канал 13] на [линию (5,8), виртуальный канал 7], а узел 8 будет отображать [линию (5,8), виртуальный канал 7] на линию, которая ведет к соответствующему пункту назначения с номером виртуального канала, соответствующего этой линии. Одна тонкая деталь при использовании этих виртуальных каналов состоит в том, что каждая линия имеет свой собственный набор виртуальных каналов, так что не требуется никакой координации между линиями относительно нумерации каналов.

При таком подходе с использованием виртуального канала в заголовке пакета должен быть передан только код номера вир-

туального канала. Самый простой способ сделать это, конечно, состоит в том, чтобы представить номер виртуального канала в виде двоичного числа в фиксированном поле заголовка. Из-за ограниченной пропускной способности линии существует некоторая естественная верхняя граница на число сеансов, которые линия может поддерживать, так что не возникает трудности при представлении номера виртуального канала в виде фиксированного числа двоичных знаков.

Ниже будут рассмотрены другие способы представления номеров виртуальных каналов, а сейчас заметим, что любой метод различения сеансов должен использовать нечто эквивалентное рассмотренному представлению номеров виртуальных каналов. Другими словами, линия передает данные для нескольких сеансов и по передаваемым битам нужно различать, какие данные относятся к тому или иному сеансу. Заголовок пакета может передавать большое количество дополнительной информации, например, такой как информация об источнике и получателе, но нужно свести к минимуму эту дополнительную информацию, необходимую для распознавания сеансов на данной линии. Рассмотрим, как это осуществляется в сети TUMNET.

2.7.1. Идентификация сеансов в сети TUMNET

Сеть TUMNET является сетью, разработанной в 1970 г. первоначально для обеспечения доступа в системе ЭВМ с разделением времени, но вскоре TUMNET была расширена для того, чтобы позволить пользователям связываться с их собственными ЭВМ. Так как терминалы того периода времени обычно за один раз передавали только один знак и перед передачей следующего знака ждали, пока эхо (отклик) на этот знак придет обратно, то разработаны были протоколы для работы с очень короткими сообщениями, состоящими из одного или нескольких знаков. Хотя сеть никогда широко не освещалась, много идей, впервые осуществленных в ней, вошли теперь в обычную сетевую практику. Кадры в сети TUMNET очень короткие, максимальной длины 66 байт; кадры имеют формат, показанный на рис. 2.35.

В заголовке кадра, состоящем из двух байт, пять бит используются как синхронизирующая последовательность, пять бит — как счетчик числа слов, три бита для ПН и три бита для НЗ. Счетчик слов указывает число 16-битовых слов в кадре, не считая заголовка и трейлера. Так как узлы были выполнены на основе 16-битовых миниЭВМ, все поля кадра организовывались путем приращений по 16 бит (два байта). Заметим, что пятибитовый счетчик слов может насчитать до 31 слова, по два байта каждое, или 62 байт. Таким образом, ограничение на то, что размер кадра может быть только кратным 16 бит, позволяет уменьшить (немного) объем служебной информации, необходимой при кадриро-

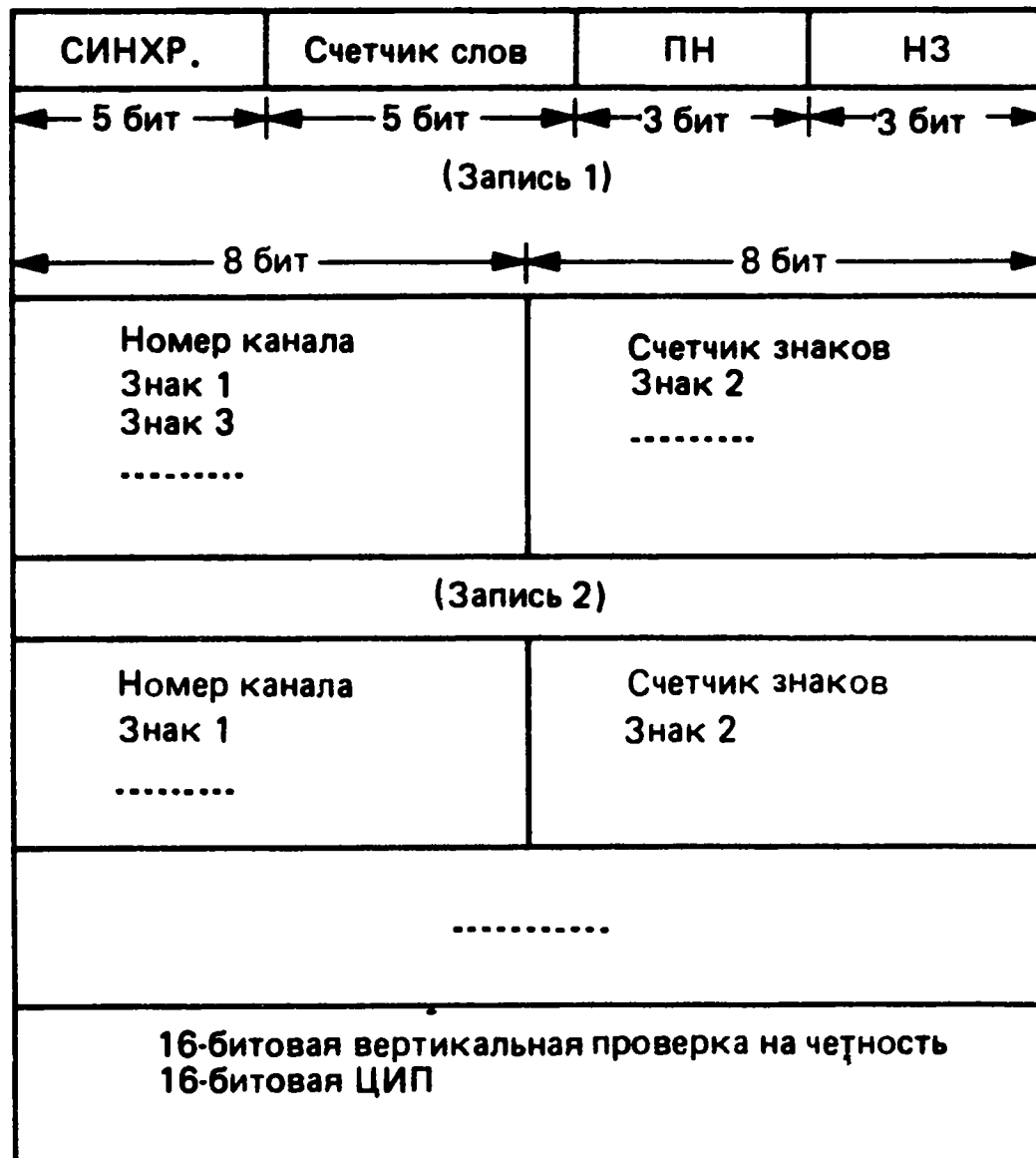


Рис. 2.35. Структура кадра в сети TUMNET. Первые 16 бит являются заголовком кадра, а последние 32 бит — трейлером кадра. Заметим, что данные из нескольких сеансов включаются в один кадр.

вании. Поля ПН и НЗ дают возможность реализовать обычный протокол ARQ на n шагов назад с модулем $m = 8$. Последние четыре байта (трейлер кадра) обеспечивают обнаружение ошибки, с использованием как 16-битовой ЦИП, так и 16 вертикальных проверок на четность. Таким образом обеспечивается несколько лучшее обнаружение ошибки, чем при стандартной 16-битовой ЦИП, но немного худшее, чем при 32-битовой ЦИП.

Каждая запись внутри кадра соответствует данным из разных сеансов. Запись начинается с байта, указывающего номер виртуального канала, который идентифицирует сеанс. Следующий байт указывает, сколько символов данных содержится в записи. Если это число нечетное, то последний байт записи является просто знаком заполнения, поставленным для того, чтобы следующая запись могла начинаться только на границе слова. Запись здесь может трактоваться как очень короткий пакет, так что номер канала и счетчик знаков соответствуют заголовку пакета. Таким образом, эти пакеты являются настолько короткими, что несколько пакетов обычно группируются в один кадр, с тем чтобы избежать чрезмерного объема служебной информации в кадре. Эти пакеты немного отличаются от обычного понятия пакета в том смысле,

что они не обязательно содержат одинаковое число знаков при передаче по разным линиям на пути своего следования.

Такой способ передачи обычно называется *статистическим уплотнением*. Это означает, что виртуальные каналы уплотняются на линии, но они уплотняются с учетом потребности. Обычная коммутация пакетов, при которой существует только один пакет на кадр, также является формой статистического уплотнения, но обычно она называется коммутацией пакетов, а способ передачи, который используется при этом, называется статистическим уплотнением. В узле, где проводится сборка кадра для передачи, сеансы обслуживаются в порядке кругового опроса и берется по несколько знаков из каждого сеанса, имеющего знаки, готовые для передачи.

2.7.2. Идентификация сеансов в сетях Codex

В сетях Codex используется подобный способ статистического уплотнения и передаются знаки многих сеансов внутри одного кадра. Однако в них применяется более эффективное кодирование для представления номера виртуального канала и количества данных в сеансе. При сборке кадра сеансы также обслуживаются в последовательном порядке, начиная с виртуального канала 1. Однако вместо передачи номера виртуального канала посылается код пробела (т. е. число пустых виртуальных каналов) между номером обслуживаемого сеанса и номером предыдущего обслуженного сеанса (рис. 2.36). Таким образом, когда все сеансы имеют знаки для передачи, все эти пробелы равны нулю, а если многие, но не все сеансы имеют знаки, большинство пробелов мало. Таким образом, путем кодирования небольших пробелов малым числом битов, а больших пробелов — большим числом битов достигается очень высокая эффективность, когда активно большое число сеансов. Вместе с тем, если активно малое число сеансов, но каждый из них имеет большой объем данных, число записей в кадре будет мало и опять эффективность будет высокой.

Для удобства реализации все поля в кадрах Codex кратны четырем битам (полубайту). Таким образом, наименьшее число битов, которое может быть использовано для представления пробела, с помощью которого определяется номер сеанса, равно одному полубайту. Способ, с помощью которого определяется число знаков в записи, использует специальный символ начала записи 0000 для начала каждой новой записи. Знак каждого сеанса сначала проходит блок сжатия данных, который отображает высоковероятные знаки в один полубайт, менее вероятные знаки — в два полубайта, а самые маловероятные знаки — в три полубайта. Это делается адаптивно с автоматическим использованием текущей статистики знаков в сеансе. Полубайт 0000 резервируется

Унарный двоичный	0 → 10
код пробела	1 → 11
	2 → 010
	3 → 011
	4 → 0110
<hr/>	
Знак начала записи	0000
<hr/>	
Образец кодирования данных	
(0000)(10)(зн.1)(зн.2)(зн.3)(зн.4)(0000)(10)(зн.1)(зн.2)	
(0000)(11)(зн.1)(0000)(0010)(зн.1)(зн.2)(зн.3)...	

Рис. 2.36. Эффективное кодирование для номеров виртуальных каналов и длин записей. В примере показано, что выделено четыре знака для виртуального канала ВК1, два для ВК2, один для ВК4 и три для ВК9. Каждая запись начинается с 0000, что указывает на начало новой записи (0000 не может появиться среди знаков данных); затем следует код пробела между виртуальными каналами с данными. Протоколы Codex аналогичны, но пробелы кодируются в блоки, которые имеют длину, кратную четырем битам. Круглые скобки используются для наглядности.

для обозначения начала новой записи. Таким образом, «заголовок пакета» состоит из первого полубайта для индикации нового пакета (записи) и следующих одного или более полубайтов, которые указывают новый номер сеанса. При большой нагрузке заголовок пакета обычно равен длине одного байта.

Этот способ кодирования разницы между следующими один за другим номерами виртуальных каналов также может быть использован в обычных сетях коммутации пакетов. Использование такого способа имеет два преимущества: во-первых, объем служебной информации пакета уменьшается при больших нагрузках, а, во-вторых, обслуживание сеансов на линии в порядке кругового опроса приводит к повышению справедливости при обслуживании сеансов.

2.8. Восстановление после ошибок на сетевом и транспортном уровнях

Доказательство корректности протоколов ARQ в подразд. 2.4.3 основывалось на ряде предположений, а именно на том, что сохраняется очередность кадров при передаче по линии, ЦИП всегда обнаруживает ошибки, линия остается работоспособной (напомним, что кадры правильно принимаются с некоторой ненулевой вероятностью), и модуль УЛПД всегда работает исправно и никогда не выходит из строя. На практике необнаруженные ошибки

возникают, линии и узлы выходят из строя (хотя эти случаи редки). Таким образом, возникает вопрос, что делать при возникновении этих неполадок на сетевом или транспортном уровне?

Одна точка зрения состоит в том, что функцией сетевого уровня является обеспечение безошибочного тракта для пакетов от источника к пункту назначения и что, следовательно, восстановление после ошибок должно происходить на сетевом уровне. Эта точка зрения популярна среди разработчиков подсетей, которые хотят быть уверенными в том, что их подсеть действительно работает правильно. Одно из преимуществ этого подхода состоит в том, что механизм восстановления после ошибок может иметь преимущества механизмов, используемых на сетевом уровне для управления маршрутизацией и потоком. Восстановление после ошибок также будет происходить внутри подсети, где его выполнение проще из-за сходной природы узлов.

Другая точка зрения состоит в том, что пакеты часто должны пройти через несколько подсетей для достижения пункта назначения (это обычно бывает, когда локальные сети используются для доступа к глобальным сетям). Отдельные подсети тогда не способны обеспечить восстановление после ошибок на отрезке от конца до конца, что приводит к необходимости решить эту проблему на транспортном уровне во внешних пунктах. Если размеры пакета меняются при передаче от одной сети к другой, естественно для подтверждения от конца в конец использовать сообщение, а не пакет, что также делает более удобным восстановление после ошибок на транспортном уровне.

Некоторые сеансы требуют более надежной степени защиты от ошибок, чем другие. Например, для финансовых операций или передачи некоторых файлов нужна более надежная передача по сравнению с цифровой речью, для которой требуется очень низкая степень защиты (и вероятно, не нужен ARQ на уровне УЛПД). Восстановление после ошибок на сетевом, транспортном или более высоких уровнях может обеспечить различные степени дополнительных защит для различных сеансов. Это не исключает необходимости ARQ на уровне УЛПД, потому что, как будет показано, восстановление после ошибок на более высоких уровнях затруднительнее и имеет большую задержку, чем на уровне УЛПД.

2.8.1. Подтверждения от конца до конца, управление потоком и разрешения на передачу

С теоретической точки зрения восстановление после ошибок как на сетевом, так и на транспортном уровне соответствует протоколу ARQ на уровне УЛПД. Один общий подход состоит в том, чтобы на передающей стороне для данного сеанса последовательно нумеровать следующие один за другим пакеты или сообщения для

обеспечения потом подтверждения от получателя, передаваемого из конца в конец и содержащего порядковый номер, а также для обеспечения повторной передачи тех пакетов или сообщений, для которых подтверждение не поступило обратно к источнику. Эти повторные передачи могут осуществляться на основе либо протокола на n шагов назад, либо протокола с выборочным повтором.

На сетевом уровне этот подход часто находится во взаимосвязи с управлением потоком. Источник посылает порядковый номер ПН в каждом пакете, а получатель посылает обратно номер принятого пакета НЗ, обозначающий номер следующего ожидаемого получателем пакета. НЗ добавляется в заголовок обратных пакетов или передается в специальном управляющем пакете, если отсутствует обратный график. При заданном размере окна n источнику разрешается передавать пакеты с номерами от НЗ до $\text{НЗ} + n - 1$ включительно.

Это означает, что для данного сеанса самое большее n пакетов может находиться в сети одновременно; таким образом обеспечивается некоторая защита от перегрузки в сети. По мере того как создается перегрузка и возрастает задержка, подтверждения задерживаются и источник начинает реже посылать пакеты. Этот эффект будет более глубоко рассмотрен в гл. 6. Если получатель желает принимать пакеты не так быстро, он может просто задерживать передачу НЗ; таким образом, этот способ управления потоком может быть полезным для уменьшения перегрузки как в подсети, так и в пункте назначения. Однако главный момент здесь состоит в том, что этот способ управления потоком служит также для доставки подтверждений от конца до конца, которые могут быть использованы при восстановлении после ошибок.

Заметим, что, хотя этот алгоритм очень похож на алгоритмы ARQ, которые используются при управлении линией передачи данных, они не являются одинаковыми. Номера ПН и НЗ в ARQ относятся к следующим друг за другом пакетам в линии и не имеют никакой связи с сеансами; они находятся в заголовках кадров и не видны с сетевого или более высоких уровней. Здесь же ПН и НЗ находятся в заголовках пакетов (или в заголовках транспортного уровня) и являются разными для разных сеансов; они сохраняют свое значение при передаче по следующим друг за другом линиям связи и просто считаются частью пакета (т. е. частью произвольного потока битов) на уровне УЛПД.

Другая разница между ARQ и алгоритмом окна состоит в том, что ARQ обычно не выполняет функцию управления потоком на линии. Линия просто передает фиксированное число битов в секунду и нет причины для ограничения этого потока (ограничена только способность приемного узла обрабатывать поступающие данные). Вспомним, что стандартные модули УЛПД используют

супервизорный пакет «не готов к приему» как способ защиты приемного узла от слишком большого числа поступающих пакетов. Этот механизм дает возможность быстро получать подтверждения, предотвращая таким образом ненужные повторные передачи, обусловленные тайм-аутами, а также позволяет приемному узлу осуществлять некоторое управление потоком в передающем узле.

Объединение подтверждений от конца до конца и управления потоком имеет некоторые ограничения. Если подтверждения не поступают из-за того, что имеют место ошибки, то пакет должен быть повторно передан, в случае же когда положительное подтверждение задерживается из-за большой нагрузки в сети, пакет не нужно повторно передавать, так как это только повысит нагрузку на сеть. Поскольку задержки в подсети весьма различны, трудно различить эти случаи при выборе длительности тайм-аута. Дело осложняется еще тем, что получатель может быть перегружен и отсрочит передачу подтверждений, чтобы предотвратить дальнейшие передачи от источника; это также может вызвать ненужные повторные передачи, обусловленные тайм-аутами.

Одно частичное решение этих проблем состоит в том, чтобы получатель имел возможность снизить активность источника без задержки подтверждений. Один из способов сделать это похож на использование супервизорных кадров «не готов к приему» в стандартных протоколах управления линией передачи данных. Такой кадр обеспечивает как быструю доставку положительного подтверждения, так и прекращение дальнейшей передачи. Аналогичный способ может быть использован на сетевом уровне (он используется в стандарте сетевого уровня X.25 и будет кратко рассмотрен). Схема *разрешения на передачу* является более усовершенствованным способом для этой цели. Получатель передает обратно к источнику не один, а два номера. Первый, обычный НЗ выполняет функцию положительного подтверждения, а второй — разрешение на передачу — указывает источнику, сколько еще пакетов получатель способен принять. Таким образом, если номер разрешения на передачу равен j , источнику разрешается передавать пакеты с номерами от НЗ до $\text{НЗ} + j - 1$ включительно. В результате это позволяет получателю изменять по желанию размер окна с каждым подтверждением, передаваемым источнику. Это также позволяет получателю посылать разрешения на передачу только для такого числа пакетов, которое соответствует имеющемуся объему буфера. Такая схема с разрешениями на передачу могла бы также использоваться в модулях УЛПД, но она менее полезна в них из-за того, что ограничения на буферизацию в узле относительно скромные, так как к узлу идет фиксированное число линий.

2.8.2. Использование подтверждений от конца до конца для восстановления после ошибок

Сейчас рассмотрим, как положительные подтверждения из конца в конец могут быть использованы для восстановления после ошибок? Одна распространенная причина, по которой пакеты или сообщения могут быть не приняты, состоит в выходе из строя линии или узла. В этом случае, если применяются виртуальные цепи, все сеансы, использующие неисправную линию или узлы, требуют новых виртуальных цепей. Как узлы и внешние пункты узнают об этих неисправностях, является интересным вопросом, который исследуется в гл. 5; здесь мы предположим, что такое оповещение имеет место. Когда старые виртуальные цепи устраняются и создаются новые виртуальные цепи, то источник в сеансе может начать передачу пакетов от последнего принятого значения НЗ; альтернативный вариант при создании новой виртуальной цепи состоит в том, что получатель может вернуть к источнику текущее значение НЗ.

Если восстановление после ошибок выполняется на сетевом уровне, то обычно устанавливает новую виртуальную цепь узел источника или получателя подсети; это несколько легче, чем установление виртуальной цепи на транспортном уровне, так как вовлекается меньшее число уровней.

Другая обычная причина того, что пакеты остаются непринятыми, состоит в том, что многие подсети и внешние пункты тайно выбрасывают пакеты, в случае если нет места в буфере для их хранения. При таком выбрасывании считается, что если осуществляется восстановление после ошибок от конца до конца либо на сетевом, либо на транспортном уровне, то выброшенные пакеты будут в конечном счете повторно переданы после некоторого тайм-аута источником и, таким образом, их выбрасывание не причинит никакого вреда.

Это неправильная точка зрения, так как пакеты начинают выбрасываться, когда сеть перегружена, и как раз в это время повторные передачи нежелательны. Дело осложняется еще тем, что если неприятие пакета становится относительно обычным, то тайм-ауты перед повторной передачей будут относительно малыми, тогда как, если неприятые пакеты редки, тайм-ауты перед повторной передачей могут быть очень большими. Таким образом, когда пакеты часто выбрасываются, можно ожидать много ненужных повторных передач.

Должны ли сети разрабатываться так, чтобы не было никаких выбрасываний пакетов, — это спорный вопрос. Однако при современном уровне техники цена буферной памяти мала по сравнению со стоимостью передачи, поэтому можно снабдить оборудование буфером достаточного объема для того, чтобы пакеты очень редко

выбрасывались из-за нехватки места для их хранения. Конечно, также требуется эффективное управление потоком, которое рассматривается в гл. 6.

Если ошибки возникают из-за того, что ЦИП не обнаруживает последовательность ошибок, или из-за временной неисправности узла, ситуация намного сложнее. Ошибки могут остаться необнаруженными и попасть к пользователю. Ошибки могут появиться в адресном поле пакета, что приведет к доставке пакета к ложному получателю. Недопустимый заголовок кадра может поступить на приемный модуль УЛПД и заставить УЛПД осуществить повторное инициирование, при котором могут потеряться или повторно передаться несколько пакетов. И наконец, ошибки в заголовках на более высоких уровнях могут быть обнаружены путем проверок на согласование в пункте назначения, что может привести к разъединению сеанса.

Некоторые из этих трудностей могут быть разрешены путем использования дополнительной ЦИП внутри пакета (или сообщения) при передаче от конца до конца. Эта ЦИП будет игнорироваться подсетью и проверяться только в пункте назначения, что полезно при обнаружении ошибок, возникших внутри узлов подсети, а также при улавливании ошибок, не обнаруженных обычной кадровой ЦИП. Однако вопрос состоит в том, являются ли эти ошибки достаточно частыми для оправдания введения ЦИП от конца до конца. Конечно, можно использовать ЦИП от конца до конца только для тех сеансов, которые требуют очень высокой надежности.

Имеется другой существенный недостаток использования ЦИП от конца до конца. Нельзя проверить те области заголовка пакета, которые меняются от линии к линии. В частности, если адресная информация передается в виде номера виртуального канала (это рассматривалось в предыдущем разделе), ошибка в этом номере, сделанная в узле подсети, будет не обнаружена и пакет поступит к ложному получателю (это один из самых плохих типов ошибок). Эта трудность может быть устранена путем помещения однозначного идентификационного номера сеанса в заголовке пакета наряду с номером виртуального канала, при этом ЦИП от конца до конца будет охватывать этот идентификационный номер. В сети ARPANET используется некоторое множество битов для проверки от конца до конца. В этой сети адрес не меняется от линии к линии и, таким образом, указанная трудность не возникает.

Для дейтаграммных сетей, в которых каждый пакет может иметь различный путь к получателю, восстановление после ошибок более сложно, так как пакеты могут поступать к получателю с нарушением порядка. Например, данный пакет может иметь очень плохой маршрут, и источник может сделать тайм-аут и

повторно передать этот пакет и многие другие пакеты, которые все могли бы быть приняты перед поступлением к получателю данного пакета. Если порядковые номера посылаются по некоторому модулю, то они могут заиклиться и данный пакет будет принят вместо некоторого более позднего пакета. Заметим, что при доказательстве корректности ARQ в разд. 2.4 предполагается, что на линии кадры передаются по порядку; это предположение остается в силе для виртуальных цепей, но нарушается для дейтаграммных сетей. Одно решение этой проблемы состоит в том, чтобы использовать большое значение модуля для порядковых номеров, а также временной «марки» в каждом пакете; когда пакет в подсети становится слишком старым, он выбрасывается. Модуль выбирается достаточно большим с тем, чтобы заикливание не могло возникнуть, пока данный пакет еще существует. Несовершенство этого решения является одной из причин, из-за которых большинство сетей использует виртуальные цепи.

2.8.3. Стандарт сетевого уровня X.25

Стандарт X.25 был разработан МККТТ для обеспечения стандартного интерфейса между внешними пунктами и узлами подсети. Напомним, что физический уровень (X.21) этого стандарта был кратко рассмотрен в разд. 2.2, а уровень УЛПД (LAPB) — в разд. 2.6. По терминологии стандарта X.25 внешний пункт называется DTE (оконечное оборудование обработки данных), а узел подсети — DCE (оборудование передачи данных), но мы будем продолжать называть их как пункты и узлы.

Структура пакета данных стандарта X.25 показана на рис. 2.37. Вспомним, что на уровне УЛПД добавляются заголовок и трейлер кадра к этой структуре пакета перед передачей по линии. Третий байт заголовка пакета очень похож на управляющий байт стандартных модулей УЛПД (за исключением того, что, как указывалось ранее, ПН и НЗ здесь относятся к порядковым номерам внутри сеанса). Управляющий бит *C* равен 0 для пакетов данных и 1 для управляющих пакетов (они обсуждаются ниже). Этот управляющий бит отличается от управляющего бита в заголовке кадра; на уровне УЛПД все пакеты с данными и управляющие считаются неразличимыми и используют 0 как управляющий бит уровня УЛПД. Наконец, бит *M* в заголовке пакета равен 1 для непоследних пакетов в сообщении и 0 для последнего пакета.

Вторая половина первого байта и второй байт заголовка пакета — это 12-битовый номер виртуального канала, который рассматривался в разд. 2.7. Этот номер для данного сеанса может быть разным на разных концах виртуальной цепи. Во многих подсетях используется такая же структура пакета внутри подсети

1	2	3	4	5	6	7	8	
Q	D	Модуль		Виртуальный канал				Байт1
Виртуальный канал								Байт 2
НЗ			М	ПН			С=0	Байт 3

Рис. 2.37. Заголовок пакета данных в стандарте X.25. Для управляющего пакета бит С равен 1.

и номер виртуального канала может меняться от линии к линии, как упоминалось в разд. 2.7.

Первый бит *Q* в заголовке пакета (если он используется) имеет значение 1 для управляющих пакетов на транспортном и более высоких уровнях и 0 для управляющих пакетов на сетевом уровне и для пакетов с данными. Следующий бит *D* указывает, имеет ли подтверждение (НЗ) смысл подтверждения из конца в конец или только смысл подтверждения для линии. Если *D* = 1, то НЗ означает, что пункт назначения принял все пакеты сеанса до номера НЗ, что ожидается пакет с номером НЗ и что источник может передавать пакеты с номерами, лежащими в окне, которое начинается от НЗ. Если *D* = 0, НЗ означает, что узел или пункт локальной подсети принял пакеты с номерами до НЗ, что ожидается пакет с номером НЗ и что другой пункт или узел может посылать пакеты с номерами, лежащими в окне, которое начинается от НЗ. Для некоторых сеансов возможно использование подтверждения как подтверждения из конца в конец, а для других как подтверждения только одной линии. Как отмечалось ранее, подтверждения из конца в конец очень полезны для восстановления после ошибок. В некоторых сеансах, однако, не требуется восстановления после ошибок, а в некоторых это осуществляется на более высоком уровне, например используя взаимодействие вопрос — ответ между пунктами. При подтверждении только одной линии функция подтверждения НЗ является излишней (уровень УЛПД гарантирует, что пакет принят). Таким образом, НЗ используется только для управления потоком, и приемный узел задерживает передачу НЗ до тех пор, пока не будет готов принять окно пакетов с номерами, большими чем НЗ.

Третий и четвертый биты заголовка указывают, равен ли модуль 8 или 128. Если модуль равен 128, то к третьему байту добавляются еще два байта для обеспечения семи битов для ПН и семи для НЗ.

Управляющие пакеты в X.25 совершенно аналогичны супервизорным кадрам и нумерованным кадрам в стандартных модулях УЛПД. Третий байт заголовка пакета указывает тип управляющего пакета; его последний бит (управляющий бит) всегда

равен 1 для различения управляющего пакета и пакета данных. Номер виртуального канала такой же, как в пакете данных, так что управляющие пакеты всегда относятся к определенному сеансу. Существуют управляющие пакеты «готов к приему», «не готов к приему» и «отклоняю», каждый из которых передает НЗ, они используются для подтверждений таким же образом, как и в модулях УЛПД. Пакет «не готов к приему» обеспечивает некоторую возможность подтверждения без запроса новых пакетов, но возможности для разрешений передач не существует.

К интересному типу управляющих пакетов относится пакет «вызов соединения». После трехбайтового заголовка пакет содержит адресное поле, имеющее в первом байте длину вызываемого адреса, и длину вызываемого адреса, а затем сами эти адреса. Этот пакет используется для установления виртуальной цепи для нового сеанса в сети. Заметим, что номер виртуального канала оказывается бесполезным при нахождении пункта назначения; он просто указывает, после того как установлена виртуальная цепь, номер используемого виртуального канала. Узел подсети при приеме пакета «вызов соединения» определяет путь к пункту назначения и передает этот пакет по направлению к пункту назначения. Если получатель в пункте назначения желает и способен принять участие в сеансе, он посылает обратно пакет «вызов принят»; сеанс установится, когда этот пакет достигнет исходного пункта.

Как пакет «вызов соединения», так и пакет «вызов принят» содержат поле для определения параметров сеанса, которое включает такие данные, как максимальная длина пакета для сеанса, размер окна, кто платит за сеанс и т. д. Максимальная длина пакета, по умолчанию, равна 128 байт, а размер окна, по умолчанию, равен 2. Если пакет «вызов соединения» указывает другие значения для максимальной длины пакета или размера окна, получатель может указать значения, более близкие к значениям по умолчанию, которые потом должны будут использоваться. Это разумная операция для внешних пунктов, но она отчасти неудобна для подсети. Подсеть может указать максимальную длину пакета и максимальный размер окна, которые могут быть использованы сеансами в подсети, но желательно также иметь возможность уменьшения размеров окон при больших нагрузках, а это не может быть сделано в стандарте X.25.

Если подсеть не способна установить виртуальную цепь (например, из-за большой нагрузки) или если получатель не желает принять вызов, то управляющий пакет «отбой запроса» отправляется обратно на инициирующий пункт, объясняя, почему сеанс не может быть установлен.

2.9. Заключение

В этой главе представлены вводный обзор физического уровня, довольно полное рассмотрение уровня управления линией передачи данных и несколько близко связанных с ними тем на сетевом уровне. С теоретической точки зрения важным является рассмотрение распределенных алгоритмов между двумя станциями при наличии ошибок и произвольных задержек. Это рассмотрение проводилось при описании ARQ, кадрировании и исправлении ошибок на высоких уровнях. Доказательство корректности класса протоколов на n шагов назад является важным потому, что оно дает пример аккуратно проведенного рассуждения о таких алгоритмах. В сетях передачи данных используется много специальных алгоритмов, которые имеют трудно обнаруживаемые недостатки, важно, чтобы разработчики понимали эти алгоритмы в более общем виде.

Другим теоретически важным вопросом является кодирование управляющей информации. Рассмотрение этого вопроса проводилось при описании кадрирования и идентификации сеансов. Хотя в главе был сделан акцент на эффективность представления, наиболее важным является рассмотрение информации, которая на самом деле передается; например, важно знать, что флаги со вставкой бита, поля счетчика длины и особые знаки управления связью передают одну и ту же информацию, но совершенно разными способами.

С более практической точки зрения рассматривались вопросы о том, как работают модемы, как ЦИП обнаруживает (и не обнаруживает) ошибки, как работает протокол ARQ и как его работа может быть сделана более эффективной при тщательном выборе тайм-аутов и управляющей обратной связи, насколько эффективны (и не эффективны) различные методы кадрирования, как может быть выполнена идентификация сеансов и как можно реализовать исправление ошибок на более высоком уровне. Многие существующие сети были использованы для иллюстрации этих методов.

Рассмотрены различные стандарты, но не в достаточных деталях для того, чтобы можно было их реализовать. К сожалению, если кто-либо желает реализовать протокол в соответствии с некоторым стандартом, он должен читать длинную и скучную документацию по этому стандарту. Наша цель состояла в том, чтобы дать некоторое теоретическое описание стандарта. Существует много сетей специального назначения, к которым не подходят стандартные протоколы и полезно знать, почему это так.

2.10. Замечания, источники и дополнительная литература

Раздел 2.2. Для получения дополнительной информации по линейным фильтрам и их представлению во временной и частотной областях можно использовать любой общий курс (например, [216]) по линейным системам. Вопросы модуляции и демодуляции цифровых сигналов глубоко и на современном уровне изложены в работе [194]. В книге [195] дано полное изложение адаптивного выравнивания. По теоретико-информационным вопросам этой главы (теорема Шеннона, энтропия, кодирование источника) см. [86]. Наконец, для простого изучения тем этого раздела, связанных с современными практическими вопросами, см. [220].

Раздел 2.3. Коды с проверкой на четность и циклические коды (ЦИП) описаны в работах [34], [47] и [86].

Разделы 2.4 и 2.5. Одной из первых статей по ARQ и кадрированию является классическая статья [99]. Целое научное направление было развито для доказательства корректности двухточечных протоколов. Обычно (см., например, [36]) используют сочетание перебора на ЭВМ и аналитических методов для уменьшения объема перебора. Дополнительные сведения по высокоэффективным протоколам ARQ и методам кадрирования даны в [89]. Выборочный повтор и преимущество передачи пакетов несколько раз после возникновения ошибок обсуждаются в работе [253]. Более полный анализ выбора длин кадров дается в [7].

Раздел 2.6. Превосходное описание стандартных модулей УЛПД дается в [43]. Задачи, связанные с чередованием битов опрос — конец в нумерованных кадрах, рассмотрены в [13].

Раздел 2.7. Сеть TUMNET описывается в [198] и [246]. Сети Codex рассматриваются в [124].

Раздел 2.8. См. [207] для дальнейшего обсуждения стандарта X.25.

Задачи

- 2.1. Предположим, что сигнал на выходе рис. 2.3, а равен $1 - e^{-2t/T}$ при $0 \leq t \leq T$; $(e^2 - 1) e^{-2t/T}$ при $t > T$ и нулю при $t < 0$. Определите сигнал на выходе на рис. 2.3, б, используя линейность фильтра и неизменность его характеристики во времени.
- 2.2. Пусть $s(t) = 1$ при $0 \leq t \leq T$ и $s(t) = 0$ в остальных случаях. Пусть $h(t) = \alpha e^{-\alpha t}$ при $t \geq 0$ и $h(t) = 0$ при $t < 0$. Используйте выражение свертки для нахождения сигнала на выходе, если входной сигнал $s(t)$ проходит через фильтр с импульсной характеристикой $h(t)$.

- 2.3. Проинтегрируйте выражение (2.1) при $s(\tau) = e^{j2\pi f\tau}$ для получения выражений (2.2) и (2.3). *Указание:* замените переменную интегрирования на $\tau' = t - \tau$.
- 2.4. Предположим, что канал имеет идеальную частотную характеристику $H(f) = 1$ при $-f_0 \leq f \leq f_0$ и $H(f) = 0$ при других f . Найдите импульсную характеристику канала (используйте обратное преобразование Фурье, формулу (2.6)).
- 2.5. Пусть $s(t)$ — заданный сигнал и пусть $s_1(t) = s(\beta t)$ для некоторого заданного положительного β . Покажите на рисунке связь между $s(t)$ и $s_1(t)$ при $\beta = 2$. Пусть $S(f)$ — преобразование Фурье $s(t)$. Выразите $S_1(f)$ — преобразование Фурье $s_1(t)$ через $S(f)$. Покажите на рисунке связь $S_1(f)$ и $S(f)$ при $\beta = 2$ (предполагая, что $S(f)$ действительная).
- 2.6. Пусть $S(f)$ — преобразование Фурье функции $s(t)$.
- Покажите, что преобразование Фурье функции $s(t) \cos(2\pi f_0 t)$ есть $[S(f - f_0) + S(f + f_0)]/2$.
 - Найдите преобразование Фурье функции $s(t) \cos^2(2\pi f_0 t)$, выразив его через функцию $S(f)$.
- 2.7. Предположим, что математическое ожидание длины кадра на линии равно 1000 бит и что стандартное отклонение составляет 500 бит.
- Найдите математическое ожидание времени передачи кадра по линии со скоростью 9600 бит/с и по линии со скоростью 50 000 бит/с.
 - Найдите математическое ожидание и стандартное отклонение времени, которое необходимо для передачи 10^6 кадров по каждой из указанных линий (в предположении, что все длины кадров являются статистически независимыми).
 - Скорость*, с которой кадры могут быть переданы, обычно определяется как величина, обратная математическому ожиданию времени передачи. Используя результат п. (б), обсудите, разумно ли это определение.
- 2.8. Покажите, что окончательная проверка на четность в коде с горизонтальной и вертикальной проверками на четность (когда берется по модулю 2 сумма всех битов данных) равна по модулю 2 сумме горизонтальных проверок на четность, а также равна по модулю 2 сумме вертикальных проверок на четность.
- 2.9. (а) Найдите пример комбинации из шести ошибок, которая не может быть обнаружена при использовании горизонтальных и вертикальных проверок на четность. *Указание.* Каждая строка с ошибками и каждый столбец с ошибками будут содержать ровно две ошибки.
- (б) Найдите число различных последовательностей ошибок, которые не будут обнаружены кодом с вертикальными и горизонтальными проверками; предположите, что решетка имеет K бит в строке и J бит в столбце.
- 2.10. Предположим, что код с проверкой на четность имеет минимальное расстояние d . Определим расстояние между кодовым словом и принятой последовательностью (той же длины) как число битовых позиций, в которых они различаются. Покажите, что если расстояние между одним кодовым словом и данной последовательностью меньше, чем $d/2$, то расстояние между любым другим кодовым словом и данной последовательностью должно превышать $d/2$. Покажите, что если декодер декодирует данную принятую последовательность в кодовое слово с наименьшим расстоянием от последовательности, то все комбинации из менее чем $d/2$ ошибок будут исправлены.
- 2.11. Рассмотрим код с проверкой на четность с тремя битами данных и четырьмя проверками на четность. Предположим, что есть три кодовых слова 1001011, 0101101 и 0011110. Найдите правило, по которому выполняется проверка на четность, и все восемь кодовых слов. Какое минимальное расстояние этого кода?
- 2.12. Пусть $g(D) = D^4 + D^2 + D + 1$ и $s(D) = D^3 + D + 1$. Найдите остаток, если $D^4 s(D)$ делится на $g(D)$, используйте арифметику по модулю 2.

- 2.13. Пусть $g(D) = D^L + g_{L-1}D^{L-1} + \dots + g_1D + 1$ (предполагается, что $L > 0$). Пусть $z(D)$ — ненулевой многочлен, члены наибольшей и наименьшей степени которого имеют степень j и i соответственно, т. е. $z(D) = D^j + z_{j-1}D^{j-1} + \dots + D^i$ (или $z(D) = D^j$, если $i = j$). Покажите, что $g(D)z(D)$ имеет по крайней мере два ненулевых члена. *Указание.* Посмотрите на коэффициенты при D^{L+j} и D^i .
- 2.14. Покажите, что если $g(D)$ содержит множитель $1 + D$, то обнаруживаются все последовательности ошибок с нечетным числом ошибок. *Указание.* Вспомните, что ненулевой многочлен ошибки $e(D)$ обнаруживается, если не выполняется равенство $e(D) = g(D)z(D)$ для некоторого многочлена $z(D)$. Посмотрите, что произойдет, если подставить D , равное 1, в это уравнение.
- 2.15. Пусть для данного порождающего многочлена $g(D)$ степени L и данной длины данных K результатом ЦИП последовательности данных, которая имеет единственную единицу в позиции i (т. е. $s(D) = D^i$ при $0 \leq i \leq K-1$), будет многочлен $c^{(i)}(D) = c_{L-1}^{(i)}D^{L-1} + \dots + c_1^{(i)}D + c_0^{(i)}$.
- (а) Для произвольного многочлена данных $s(D)$ покажите, что многочлен ЦИП равен $c(D) = \sum_{i=0}^{K-1} s_i c^{(i)}(D)$ (используйте арифметику по модулю 2).
- (б) Предполагая, что $c(D) = c_{L-1}D^{L-1} + \dots + c_1D + c_0$, покажите, что
- $$c_j = \sum_{i=0}^{K-1} s_i c_j^{(i)}; \quad 0 \leq j < L.$$

Это показывает, что каждый коэффициент c_j является проверкой на четность и код с циклически избыточной проверкой является кодом с проверкой на четность.

- 2.16. (а) Рассмотрите алгоритм ARQ с остановкой и ожиданием, в котором передающий модуль УЛПД вместо того, чтобы использовать порядковый номер для идущих друг за другом пакетов, посылает столько раз, сколько данный пакет был повторно передан. Таким образом, формат передаваемых кадров следующий: $[j \mid \text{пакет} \mid \text{ЦИП}]$, где j равно 0 при первой передаче пакета, 1 при первой повторной передаче и т. д. Приемный модуль УЛПД возвращает положительное подтверждение или отрицательное подтверждение (без всякого номера принятого пакета) для каждого принятого им кадра. Покажите, например, что этот алгоритм работает некорректно независимо от того, какое правило используется в приемном модуле УЛПД для того, чтобы считать пакеты принятыми (используйте предположения, которые предшествуют подразд. 2.4.1).
- (б) Выполните повторно п. (а) со следующими двумя изменениями: (1) указанное число j равно 0 при первой передаче пакета и 1 при *всех* последующих повторных передачах этого пакета, (2) кадры имеют фиксированную задержку в каждом направлении, всегда распознаются как кадры и передающий модуль УЛПД никогда не делает тайм-ауты, а просто ожидает положительное или отрицательное подтверждение, либо ошибку.
- 2.17. (а) Пусть T_t — математическое ожидание времени передачи кадра с данными, а T_f — время передачи по линии обратной связи кадра с положительным подтверждением или кадра с отрицательным подтверждением. Пусть T_d — математическое ожидание задержки распространения и обработки в одном направлении (считается одинаковым в обоих направлениях). Найдите T — математическое ожидание промежутка времени между последовательными передачами кадров в системе с оста-

новкой и ожиданием (в предположении, что не существуют задержки по другим причинам и кадры не теряются).

- (б) Пусть p_t — вероятность ошибки в кадре с данными (независимая от длины кадра), и p_f — вероятность ошибки при обратной связи в кадре с положительным или отрицательным подтверждением. Найдите q — вероятность того, что пакет с данными безошибочно принимается и данная передача положительно подтверждается. Покажите, что $1/q$ — математическое ожидание числа передач пакета в системе с остановкой и ожиданием (предполагается независимость ошибок во всех кадрах).
- (в) Сопоставляя пункты (а) и (б), найдите математическое ожидание времени, которое требуется для передачи пакета. Оцените это время при $T_t = 1$, $T_f = T_d = 0,1$ и $p_t = p_f = 10^{-3}$.

- 2.18. Перечертите рис. 2.21 и 2.22 с теми же длинами кадров и с тем же набором ошибочных кадров, но сосредоточив внимание на пакетах от B к A , т. е. покажите ПН и y_{\min} для узла B и НЗ и выход пакетов для узла A (предполагая, что все кадры передают пакеты).
- 2.19. Приведите пример, в котором алгоритм на n шагов назад будет иметь тупик, если в приемных модулях УЛПД не учитывается номер принятого пакета НЗ в каждом кадре, который не передает ожидаемый пакет. *Указание.* Постройте последовательность ошибочных кадров, которая приводит к ситуации, когда $y_{\text{пр}}$ в узле B равно n плюс значение y_{\min} в узле A , и наоборот.
- 2.20. Приведите пример, в котором протокол ARQ на n шагов назад приводит к ошибкам, если модуль m равен n . Для $m > n$ приведите пример, когда (2.28) выполняются со знаком равенства, и пример, когда (2.29) выполняются со знаком равенства.
- 2.21. Предположим, что в момент t узел A имеет данное значение y_{\min} как наименьший номер еще не подтвержденного пакета и $y_{\max} - 1$ как наибольший номер уже переданного пакета. Пусть T_m — минимальное время передачи пакета по линии, а T_d — задержка распространения. Покажите, что значение $y_{\text{пр}}$ в узле B должно лежать в интервале от y_{\min} до y_{\max} для моментов времени в интервале от $t - T_m - T_d$ до $t + T_m + T_d$.
- 2.22. (а) Пусть T_{\min} — минимальное время передачи кадров с данными, а T_d — задержка распространения и обработки в каждом направлении. Найдите максимально возможное значение T_{\max} для времени передачи, такое, что система ARQ на n шагов назад (с данным n) не будет никогда возвращаться назад или ожидать при отсутствии ошибок передачи и потере кадров. *Указание:* посмотрите на рис. 2.24.
- (б) Выполните повторно п. (а) с учетом возможности того, что отдельные ошибки могут возникать в направлении обратной связи.
- 2.23. Предположим, что длительности передачи кадров τ распределены экспоненциально с плотностью вероятности $p(\tau) = e^{-\tau}$ при $\tau \geq 0$. Предположим, что новый кадр в каждом направлении всегда начинается, как только кончается предыдущий кадр этого направления. Пусть все длительности передач независимы и задержки обработки и распространения незначительны. Покажите, что вероятность q того, что данный кадр положительно не подтвердится до конца передачи $n - 1$ дополнительных кадров (т. е. до окончания окна) равна $q = (1 + n) 2^{-n}$. *Указание.* Моменты времени, в которых кончаются передачи кадров в любом узле, образуют пуассоновский точечный процесс. Любая точка этого процесса может равновероятно быть как моментом окончания пакета в A , так и моментом окончания пакета в B , при этом имеет место независимость интервалов между последовательными окончаниями.
- 2.24. Покажите, что если одиночная ошибка возникает при передаче по линии обратной связи положительного подтверждения данного пакета, то q в задаче 2.23 становится равным $q = [2 + n + (n + 1) n/2] 2^{-n-1}$.

- 2.25. Пусть длительности передачи кадров τ имеют распределение Эрланга с плотностью вероятности $p(\tau) = \tau e^{-\tau}$. Найдите соответствующее значение q в задаче 2.23. *Указание.* Моменты окончания передачи пакетов в A могут рассматриваться как точки пуассоновского процесса, то же самое справедливо для B .
- 2.26. Пусть γ — математическое ожидание числа переданных кадров от A к B , необходимых для успешного принятия пакета в B . Пусть β — математическое ожидание числа переданных кадров от A к B за время между передачей данного кадра и получением подтверждения по линии обратной связи об этом кадре (включая кадр, передаваемый в тот момент, когда поступает подтверждение по линии обратной связи). Пусть p — вероятность того, что кадр, поступающий в B , содержит ошибки (предполагается, что они независимы в последовательных кадрах). Предположим, что из A всегда передаются кадры и что n достаточно велико, так что A никогда не возвращается назад при отсутствии обратной связи и всегда возвращается назад на следующем кадре после момента, когда он узнает, что ожидаемый кадр содержит ошибки. Покажите, что γ удовлетворяет уравнению $\gamma = 1 + p(\beta + \gamma)$. Определите эффективность η как $1/\gamma$ и найдите η как функцию β и p .
- 2.27. Предположим, что система с выборочным повтором имеет ту особенность, что передающий модуль УЛПД всегда узнает, был ли данный переданный пакет успешно принят в течение или до начала передачи β -го по счету кадра после кадра, в котором передавался данный пакет. Приведите точное доказательство того, что передающему модулю УЛПД никогда не надо помнить более чем $\beta + 1$ неподтвержденных пакетов.
- 2.28. Найдите эффективность системы ARPANET ARQ при условии, что все восемь виртуальных каналов всегда передают пакеты, и информация, идущая по линии обратной связи, всегда поступает в течение или до начала передачи седьмого по счету кадра, следующего после кадра, передающего данный пакет.
- 2.29. Рассмотрите обобщенную систему ARQ с выборочным повтором, которая работает следующим образом. Как и в обычных протоколах на n шагов назад и с выборочным повтором, пусть $y_{\text{пр}}$ — наименьший номер пакета, который еще не принят безошибочно, y_{min} — оценка передатчиком величины $y_{\text{пр}}$ и y_{max} — число, на единицу большее, чем наибольший номер уже переданного пакета. Пусть $y_{\text{наиб}}$ — наибольший номер безошибочно принятого пакета, который считается принятым (таким образом, для протокола на n шагов назад $y_{\text{наиб}}$ было бы равно $y_{\text{пр}} - 1$, а для обычного протокола с выборочным повтором $y_{\text{наиб}}$ могло бы достигать величины $y_{\text{пр}} + n - 1$). Инструкция на передатчике такая же, как и для обычных протоколов на n шагов назад или с выборочным повтором: номер передаваемого пакета z должен удовлетворять неравенствам

$$y_{\text{max}} - n \leq z \leq y_{\text{min}} + n - 1.$$

Как следствие всегда выполняется неравенство $y_{\text{max}} \leq y_{\text{min}} + n$. Инструкция для приемника для некоторого заданного положительного числа k состоит в том, что безошибочно принятый пакет с номером z считается принятым, если

$$y_{\text{пр}} \leq z \leq y_{\text{наиб}} + k. \quad (2.49)$$

- (а) Сначала предположим, что z (а не $\text{ПН} = z$ по модулю m) передается с каждым пакетом и что $y_{\text{пр}}$ (а не $\text{НЗ} = y_{\text{пр}}$ по модулю m) передается с каждым пакетом в обратном направлении. Покажите, что на приемной стороне для каждого принятого пакета выполняются неравенства

$$\begin{aligned} z &\leq y_{\text{пр}} + n - 1, \\ z &\geq y_{\text{наиб}} - n + 1. \end{aligned} \quad (2.50)$$

- (б) Теперь предположим, что $ПН = z$ по модулю m и $НЗ = y_{пр}$ по модулю m . Когда принимается пакет с номером z , неравенства (2.50) выполняются, но работа алгоритма будет ошибочной, если $z - m$ или $z + m$ лежат в диапазоне неравенств (2.49). Какое нужно выбрать m , чтобы $z + m > y_{наиб} + k$ (т. е. чтобы $z + m$ было за диапазоном неравенств (2.49))?
- (в) Является ли величина m , определенная в п. (б), достаточно большой для гарантии того, что $z - m < y_{пр}$?
- (г) Сколь большим должно быть m (как функция n и k) для гарантии корректной работы?
- (д) Дайте интерпретацию частных случаев $k = 1$ и $k = n$.
- 2.30. (а) Примените правила добавления бита из подразд. 2.5.2 для следующего кадра:

0110111110011111101011111111101111010

- (б) Пусть принимается следующая последовательность битов:

011111101111101100111110011111011111011000111111010111110

Отбросьте вставленные биты и покажите, где находятся флаги.

- 2.31. Предположим, что правило добавления бита из подразд. 2.5.2 изменяется так, что 0 вставляется только после появления последовательности 01^5 в исходных данных. Опишите, как правило выбрасывания на приемной стороне должно быть модифицировано при таком изменении. Покажите, как бы ваше правило позволило выбросить биты из следующей последовательности:

01101111101111110111110101111110

Если ваше правило выбрасывания является верным, вы должны отбросить только два нуля и обнаружить только один настоящий флаг.

- 2.32. Добавление бита должно устранять появление последовательностей 01^6 внутри передаваемого кадра, поэтому считается, что исходная последовательность 01^6 будет всегда превращена в последовательность 01^501 . Используйте это и в предположении безошибочного выбрасывания в приемнике, покажите, что 0 должен всегда быть вставлен после 01^5 . *Указание.* Рассмотрите последовательность данных $01^501x_1x_2\dots$. Если 0 не вставляется после 01^5 , приемник не может отличить эту последовательность от последовательности $01^6x_1x_2\dots$ со вставкой бита, так что в этом случае добавление бита необходимо. Распространите этот пример на последовательность $01^50^k1x_1x_2\dots$ при любом $k > 1$.

- 2.33. Предположим, что последовательность 0101 используется как последовательность битов для указания конца кадра, а правило добавления бита состоит в том, чтобы вставлять 0 после каждого появления 010 в исходных данных; таким образом, последовательность 010101 преобразуется добавлением бита в 01001001. Кроме того, если сам кадр кончается последовательностью 01, то 0 вставляется (после первого 0) в последовательность 0101, которая теперь будет концом кадра. Покажите, как последовательность 11011010010101011101 видоизменяется этим правилом. Опишите правило выбрасывания в приемнике. Как надо выбросить нули из последовательности 11010001001001100101?

- 2.34. Пусть $A = E\{K\} 2^{-j} + j + 1$ есть верхняя граница избыточности в (2.38) и $j = \lfloor \log_2 E\{K\} \rfloor$. Покажите, что A удовлетворяет соотношению

$$1,914 \dots + \log_2 E\{K\} \leq A \leq 2 + \log_2 E\{K\}.$$

Найдите аналитическое выражение для 1,914... *Указание.* Определите γ как $\log_2 E\{K\} - j$ и найдите минимум и максимум выражения $A - \log_2 E\{K\}$ при изменении γ от 0 до 1.

- 2.35.** Покажите, что вероятность того, что ошибки будут причиной появления флага 01^k0 внутри передаваемого кадра, приближенно равна $(1/32) Kp$, где K — число битов в кадре перед добавлением битов, а p — вероятность ошибки в одном бите. Предположим, что p очень мало и вероятность нескольких ошибок в кадре незначительна. Предположим, что биты исходного кадра являются н. о. р. (независимыми и одинаково распределенными) случайными величинами, принимающими с равной вероятностью значения 0 и 1. *Указание.* Не так просто найти вероятность появления флага, так как биты после вставки не являются более н. о. р. Сначала найдите вероятность того, что ошибка во вставленном бите вызывает появление флага (а не обрыва кадра) и используйте рассуждения из подразд. 2.5.2 для приближенного вычисления математического ожидания числа вставленных битов, которое равно $K2^{-6}$. Затем найдите вероятность появления флага из-за ошибок в исходных битах кадра.
- 2.36.** Сеть с виртуальными цепями используется в случае, когда все сеансы почти всегда имеют большой трафик. Пусть в этом случае обслуживание различных сеансов на каждой линии происходит в круговом порядке, сначала передается один пакет из первого сеанса, затем один из второго и т. д. до последнего, а потом передается опять пакет из первого сеанса и т. д., таким образом, нет необходимости передавать в каждом пакете номер сеанса. Для того чтобы сохранить маловероятную возможность того, что сеанс не имеет пакета для передачи по линии, когда подходит его очередь, перед пакетом с информацией из следующего сеанса, который имеет пакет, поместим флаг 01^{k-1} из k бит. За этим флагом поместим двоичный код числа сеансов, идущих подряд и не имеющих пакетов для передачи, 1 для одного сеанса, 01 для двух, 001 для трех и т. д. Таким образом, например, пусть пакет второго сеанса был только что передан, третий и четвертый сеансы не имеют пакетов, а пятый сеанс имеет пакет \bar{x}_5 , тогда передаем $01^{k-1}01\bar{x}_5$ в виде следующего пакета. Кроме того, используется вставка, если информационный пакет начинается с 01^{k-2} . В противном случае не требуются добавления битов в информационные пакеты, и информационные пакеты сохраняют собственную длину.
- (а) Предположим, что p — вероятность того, что сеанс не имеет пакета, когда подошла его очередь; это событие предполагается независимым от других сеансов и от прошлой истории. Найдите математическое ожидание числа избыточных битов (в расчете на один переданный пакет), затрачиваемых для флагов, кода числа идущих подряд пустых сеансов и для вставок. Пренебрегите как незначительными любыми трудностями типа что делать, если нет вообще сеанса, который имеет пакет для передачи.
- (б) Предположим, что такие модифицированные пакеты с флагами, кодом длины пробела и вставками поступают на вход системы управления линией передачи данных, в которой используются ARQ для исправления ошибок и флаги для указания конца пакетов. Объясните, возникают ли какие-либо трудности из-за использования флагов как системой УЛПД, так и системой более высокого уровня, которые служат в последней для привязки пакетов к сеансам.
- 2.37.** Рассмотрим протокол управления линией передачи данных с остановкой и ожиданием. Каждый переданный кадр содержит порядковый номер по модулю 2 содержащегося в нем пакета. Когда кадр поступает в приемник, порядковый номер пакета сравнивается с номером пакета, ожидаемого приемником; если эти номера равны по модулю 2 и выполняется ЦИП, пакет считается принятым и номер ожидаемого пакета получает приращение. Подтверждение, содержащее новый номер (по модулю 2) ожидаемого пакета, посылается обратно на передатчик. Если для принятого кадра выполняется ЦИП, но порядковый номер пакета не соответствует ожидаемому,

подтверждение, содержащее номер (по модулю 2) ожидаемого пакета, также посылается обратно на передатчик. Новый момент здесь состоит в том, что кадры могут передаваться по каналу не по порядку. Более точно существует известная максимальная задержка T в канале. Если пакет передается в некоторый момент времени t , то он либо не принимается вообще, либо принимается в некоторый момент времени из интервала от t до $t + T$ независимо от других передач. Обратный канал для подтверждений ведет себя таким же образом с максимальной задержкой T . Предположим, что приемник посылает подтверждения мгновенно по принятии пакетов с выполненными ЦИП.

- (а) Опишите правила, которым следует передатчик для обеспечения того, что каждый пакет в конце концов считается принятым один раз и пакеты принимаются по порядку. Не разрешается помещать в пакеты дополнительную протокольную информацию или менять операции приемника. Нужно предоставить передатчику по возможности максимум свободы, лишь бы сохранялась корректная работа системы.
- (б) Объясните, почему невозможно построить корректный протокол, если нет границы для задержки передачи пакета.

3. Задержки в сетях передачи данных и математические модели

3.1. Введение

Одной из наиболее важных характеристик сети передачи данных является средняя задержка, необходимая для доставки пакета от источника к месту назначения. Более того, результаты исследования задержки оказывают сильное влияние на выбор и работу сетевых алгоритмов, таких как алгоритмы маршрутизации и алгоритм управления потоком. По этой причине важно понять природу и механизм задержки, а также каким образом задержка зависит от характеристик сети.

Главной методологической основой для анализа задержки сети является теория массового обслуживания. Однако ее использование часто требует упрощающих предположений, так как, к сожалению, более реалистичные предположения делают содержательный анализ чрезвычайно сложным. По этой причине в некоторых случаях невозможно провести точные количественные расчеты задержки на основе моделей теории массового обслуживания. Тем не менее эти модели часто являются основой для разумных аппроксимаций задержки, а также позволяют получить полезные качественные результаты.

В дальнейшем мы сосредоточим внимание на задержке пакета внутри подсети связи (т. е. на сетевом уровне). Эта задержка является суммой задержек на каждой линии подсети, которую проходит пакет. Каждая задержка на линии в свою очередь состоит из четырех компонент.

1. *Задержка на обработку* — задержка между моментом, когда пакет был правильно принят на начальном узле линии, и моментом, когда пакет был поставлен в очередь на передачу по линии. (В некоторых системах следует прибавить к этой задержке некоторое дополнительное время, затрачиваемое на обработку на уровнях УЛПД и физическом.)
2. *Задержка в очереди* — задержка между моментом, когда пакет был поставлен в очередь на передачу, и моментом, когда он начинает передаваться. В течение этого времени пакет ждет, пока будут переданы другие пакеты из очереди.
3. *Задержка передачи* — задержка между моментами, когда передадутся первый и последний биты пакета.

4. *Время распространения* — промежуток времени от момента, когда последний бит был передан на начальном узле линии, до момента, когда он будет принят в конечном узле этой линии. Эта задержка пропорциональна физическому расстоянию между передатчиком и приемником и обычно мала, за исключением случая спутниковой линии связи.

При таком рассмотрении не учитывается возможность того, что может потребоваться повторная передача по линии из-за ошибок передачи или каких-либо других причин. Для большинства реальных линий связи, за исключением линий множественного доступа, которые будут описаны в гл. 4, повторные передачи бывают редко и они не будут рассматриваться. Время распространения зависит от физических характеристик линии связи и не зависит от потока, передаваемого по линии. Задержка обработки также не зависит от величины потока, обрабатываемого в соответствующем узле, если вычислительная мощность узла не является ограниченной. Мы примем это предположение в наших рассмотрениях. В противном случае отдельная очередь на обработку должна быть введена перед очередями на передачу. Большая часть нашего последующего анализа сосредоточена на задержках в очереди и задержках передачи. Сначала мы рассмотрим единственную линию связи и проанализируем некоторые классические модели теории массового обслуживания. Затем рассмотрим сеть и обсудим те приближения, которые допускаются, когда выбираются аналитические модели для расчета задержки.

Хотя главный акцент делается на модели сети с коммутацией пакетов, некоторые разработанные модели пригодны также для сети с коммутацией каналов. Теория массового обслуживания широко развивалась исходя из потребностей создания математических моделей для телефонии.

3.1.1. Уплотнение потока в линии связи

Рассматриваемая линия связи наглядно может быть представлена как труба для передачи битов; по ней может быть передано заданное число битов в секунду. Это число называется *пропускной способностью* линии. Пропускная способность зависит как от физического канала, так и от интерфейса (например, модемов) и является скоростью, с которой биты проходят через интерфейс. Пропускная способность линии может распределяться между несколькими потоками (например, виртуальными цепями или группами виртуальных цепей), уплотненными в ней. Способ распределения пропускной способности среди этих потоков оказывает сильное влияние на задержку пакета.

В более общей схеме *статистического уплотнения* пакеты всех потоков ставятся в одну очередь и передаются в порядке

«первый пришел—первый обслуживается». Разновидность этой схемы, для которой характерна примерно та же средняя задержка пакета, имеет отдельную очередь для каждого потока и обслуживает очереди последовательно, передавая по одному пакету из очереди. Однако, если очередь некоторого потока пуста, обслуживается следующий поток и, таким образом, напрасно ресурс передачи не тратится. Так как вся пропускная способность C бит/с предоставляется каждый раз одному пакету, для передачи пакета длины L бит требуется L/C с.

При *временном* (ВУ) и *частотном* (ЧУ) уплотнении m потоков пропускная способность линии, по существу, разбивается на m частей, по одной на поток. При ЧУ канал с полосой частот W разбивается на m каналов, каждый шириной полосы W/m (на самом деле немного меньшей из-за того, что необходимы защитные полосы между каналами). Пропускная способность каждого канала приближенно равна C/m , где C — пропускная способность, которая получается, когда вся полоса частот отдается одному каналу. Время передачи пакета длины L бит равно Lm/C , или в m раз больше, чем в соответствующей схеме статистического уплотнения. При ВУ распределение канала производится путем деления временной оси на окна (отрезки) фиксированной длины (например, равной длине одного бита или байта или длине одного пакета, если пакеты имеют фиксированную длину). Снова в принципе можно рассматривать линию связи как состоящую из m отдельных линий с пропускными способностями C/m . В случае когда окна малы по сравнению с длиной пакета, можно опять считать, что время передачи пакета длины L бит равно Lm/C . В случае когда окна равны длине пакета, время передачи пакета из L бит равно L/C , но между передачами пакетов из одного потока имеется задержка, равная времени передачи $m - 1$ пакета.

Один из результатов, который будет получен при нашем анализе систем массового обслуживания, состоит в том, что в случае статистического уплотнения средняя задержка пакета меньше, чем при ВУ или ЧУ. Это особенно заметно, когда уплотняемые потоки имеют относительно короткий период занятости по сравнению с периодом молчания. Основная причина большей задержки при ВУ и ЧУ состоит в том, что ресурсы системы связи напрасно тратятся, когда они отдаются потоку, в котором нет очереди пакетов, тогда как другие потоки имеют пакеты, стоящие в очереди. По аналогии с потоком рассмотрим дорогу с m полосами для двух случаев. В одном случае автомобилям не разрешается менять полосу движения (это соответствует ВУ или ЧУ), а в другом случае автомобили могут менять полосу движения (это приближенно соответствует статистическому уплотнению). Запрещение перестроения увеличивает время поездки; причина здесь та же, по которой задержки при ВУ или ЧУ увеличиваются, а именно она

состоит в том, что некоторые ресурсы системы (полосы для движения или каналы связи) в данный момент не могут быть использованы, тогда как другие ее ресурсы перегружены.

При определенных условиях ВУ и ЧУ могут иметь преимущество. Предположим, что каждый поток имеет регулярный характер, т. е. все моменты возникновения пакетов достаточно разделены, так что никакой пакет не ожидает в то время, когда передается предыдущий пакет. Если эти потоки объединить в одну очередь, то можно показать, что средняя задержка пакета при этом уменьшится, но дисперсия времени ожидания в очереди, как правило, станет положительной (для иллюстрации см. задачу 3.7). Поэтому если малое колебание задержки важнее, чем ее уменьшение, то более предпочтительным является использование ВУ или ЧУ. Другое преимущество ВУ и ЧУ состоит в том, что нет необходимости во включении в каждый пакет идентификатора потока, что приводит к сокращению некоторой избыточности. Это существенно, так как если избыточность пренебрежимо мала, то можно попытаться сделать пакеты очень малыми и снизить задержку, используя эффект непрерывной передачи пакетов по тракту (см. рис. 2.29).

3.2. Модели теории массового обслуживания, теорема Литтла

Рассмотрим системы массового обслуживания, в которых требования на обслуживание появляются в случайные моменты времени. Распределение вероятностей времени между двумя следующими один за другим моментами появления требований и распределение вероятностей времени обслуживания требования заданы.

Применительно к сетям передачи данных требования представляют собой пакеты, предназначенные для передачи по линии связи. Время обслуживания соответствует времени передачи пакета и равно L/C , где L — длина пакета в битах, а C — пропускная способность линии в бит/с. В этой главе удобно не учитывать различие между пакетами и кадрами, которое имеется на уровне 2; поэтому включим в длины пакетов заголовки и трейлеры кадров. В некоторых других применениях (на которых мы не будем подробно останавливаться) требования представляют собой активные обмены информацией (или виртуальные цепи) между точками в сети, а время обслуживания соответствует продолжительности обмена.

Обычно мы будем интересоваться оценкой следующих величин:

1. Среднее число требований в системе (т. е. типичное число требований, которые либо ждут в очереди, либо обслуживаются).

2. Средняя задержка требования (т. е. «типичное» время, которое требование проводит, ожидая в очереди, плюс время обслуживания).

Сначала необходимо пояснить приведенные термины. Введем обозначение

$p_n(t)$ — вероятность того, что n требований находится в очереди или на обслуживании в момент t .

Типичной является ситуация, когда даны начальные вероятности $p_n(0)$ в момент 0 и имеется достаточно статистической информации для определения, по крайней мере в принципе, вероятностей $p_n(t)$ для всех моментов t . Введя обозначение $\bar{N}(t)$ — среднее число требований в системе в момент t , имеем

$$\bar{N}(t) = \sum_{n=0}^{\infty} n p_n(t).$$

Заметим, что как $\bar{N}(t)$, так и $p_n(t)$ зависят от t и начального распределения вероятностей $\{p_0(0), p_1(0), \dots\}$. Однако системы массового обслуживания, которые будут рассмотрены, обычно *достигают равновесия* в том смысле, что для некоторых p_n и N (независимо от начального распределения)

$$\lim_{t \rightarrow \infty} p_n(t) = p_n, \quad n = 0, 1, \dots \quad (3.1)$$

и

$$N = \sum_{n=0}^{\infty} n p_n = \lim_{t \rightarrow \infty} \bar{N}(t).$$

Прежде всего нас будут интересовать равновесные вероятности и среднее число требований в системе. Заметим, что, возможно, $N = \infty$; это справедливо, когда скорость поступления требований превышает возможности обслуживания в системе. Функции реализации числа требований в системе будут обозначаться через $N(t)$. Функция реализации, усредненная по времени на отрезке $[0, t]$, обозначается как

$$N_t = \frac{1}{t} \int_0^t N(\tau) d\tau.$$

Почти любая система, интересующая нас, является эргодической в том смысле, что

$$\lim_{t \rightarrow \infty} N_t = \lim_{t \rightarrow \infty} \bar{N}(t) = N$$

с вероятностью единица. Равенство среднего по большому интервалу времени и среднего по реализациям различных случайных процессов часто будет приниматься в этой главе на интуи-

тивном уровне, так как при строгом математическом доказательстве используются выкладки, которые лежат за пределами этого текста.

Что касается средней задержки требования, то имеется достаточно статистической информации, чтобы определить в принципе распределение вероятностей задержки каждого конкретного требования (т. е. первого, второго и т. д.). Таким образом, можно определить среднюю задержку каждого требования. Средняя задержка k -го требования, обозначаемая через \bar{T}_k , обычно сходится при $k \rightarrow \infty$ к стационарному значению

$$T = \lim_{k \rightarrow \infty} \bar{T}_k.$$

Указанный предел будем называть средней задержкой требования. (Опять возможно, что $T = \infty$.) Для интересующих нас систем стационарная средняя задержка T также равна (с вероятностью единица) задержке требования, усредненной по большому интервалу времени, т. е.

$$T = \lim_{k \rightarrow \infty} \bar{T}_k = \lim_{k \rightarrow \infty} \frac{1}{k} \sum_{i=1}^k T_i,$$

где T_i — задержка i -го требования.

Среднее число требований в системе N и средняя задержка T связаны простой формулой, которая дает возможность выражать одно среднее через другое. Этот результат, известный как *теорема Литтла*, имеет вид

$$N = \lambda T,$$

где λ — средняя скорость поступления требований определяется как

$$\lambda = \lim_{t \rightarrow \infty} \frac{\text{математическое ожидание числа требований, поступивших в интервале } [0, t]}{t}.$$

(Будем предполагать, что указанный предел существует.) Явления, отражающие теорему Литтла, встречаются в повседневной жизни. Например, в дождливый день поток машин в час пик движется медленнее, чем в среднем (большое T), поэтому на улицах больше машин (большое N). Аналогично ресторан быстрого обслуживания (небольшое T) нуждается в меньшем помещении (малое N), чем обычный ресторан, при одной и той же скорости поступления требований.

Справедливость теоремы Литтла может быть доказана, и ее вывод очень прост. Дадим графическое доказательство, которое предполагает, что требования обслуживаются в порядке их поступления. Аналогичное доказательство возможно для случая

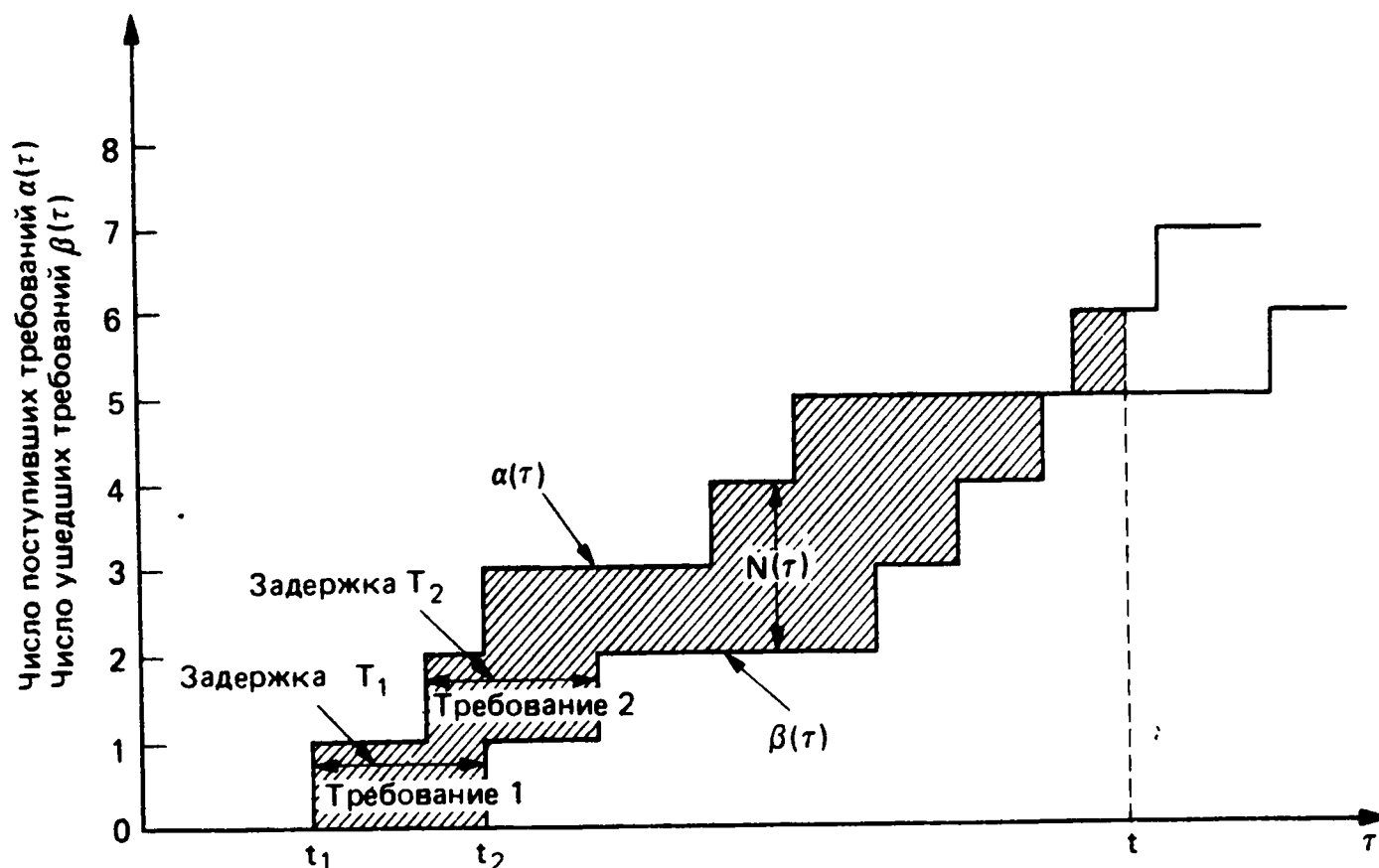


Рис. 3.1. Доказательство теоремы Литтла. Площадь заштрихованной области может быть выражена как $\int_0^t N(\tau) d\tau$ и как $\sum_{i=1}^{\beta(t)} T_i + \sum_{i=\beta(t)+1}^{\alpha(t)} (t - t_i)$. Разделив оба выражения на t , приравняв их и взяв предел при $t \rightarrow \infty$, получим теорему Литтла.

с произвольным порядком обслуживания (см. задачи 3.31 и 3.32). Введем обозначения

$\alpha(t)$ — число требований, поступивших в систему за время $[0, t]$,

$\beta(t)$ — число требований, покинувших систему за время $[0, t]$.

Предполагая, что система пуста в момент времени 0, получим, что число требований в системе в момент t равно

$$N(t) = \alpha(t) - \beta(t).$$

Пусть t_i и T_i — момент поступления и время пребывания в системе соответственно i -го требования. Рассмотрим в любой момент времени t заштрихованную область на рис. 3.1, которая лежит между графиками $\alpha(\tau)$ и $\beta(\tau)$ до момента t . Площадь этой области может быть выражена как

$$\int_0^t N(\tau) d\tau,$$

а также как

$$\sum_{i=1}^{\beta(t)} T_i + \sum_{i=\beta(t)+1}^{\alpha(t)} (t - t_i).$$

Разделив оба выражения на t и приравняв их, получим

$$N_t = \lambda_t T_t, \quad (3.2)$$

где

$$N_t = \frac{\int_0^t N(\tau) d\tau}{t} \text{ — среднее на отрезке } [0, t] \text{ число требований в системе,}$$

$$\lambda_t = \frac{\alpha(t)}{t} \text{ — средняя на отрезке } [0, t] \text{ скорость поступления требований,}$$

$$T_t = \frac{\sum_{i=1}^{\beta(t)} T_i + \sum_{i=\beta(t)+1}^{\alpha(t)} (t - t_i)}{\alpha(t)} \text{ — среднее на отрезке } [0, t] \text{ время пребывания требований в системе.}$$

Предполагая, что

$$N_t \rightarrow N, \quad \lambda_t \rightarrow \lambda, \quad T_t \rightarrow T,$$

получаем из соотношения (3.2) искомую формулу.

Заметим, что выражение для T_t включает общее время пребывания в системе всех требований с номерами от 1 до $\beta(t)$, но не учитывает время пребывания в системе после момента t требований, которые были в системе в момент t . В предположении, что $N_t \rightarrow N \ll \infty$, этот граничный эффект из-за требований в системе, которые остались в момент t , будет малым по сравнению с суммарным временем пребывания в системе требований с номерами от 1 до $\beta(t)$ и T_t при больших t может рассматриваться как среднее время пребывания в системе.

Строго говоря, для того чтобы приведенное рассуждение было корректным, необходимо установить, что средние по времени N_t , λ_t , T_t сходятся с вероятностью единица к соответствующим средним по вероятностям N , λ и T . Это справедливо в каждом случае, интересующем нас, и в последующих рассуждениях будем использовать теорему Литтла без дополнительного подробного анализа.

Значимость теоремы Литтла заключается в ее большой общности. Она справедлива почти для всех систем массового обслуживания, которые в пределе достигают статистического равновесия. Система необязательно должна иметь только одну очередь. Теорема справедлива для многих сложных систем, в которые что-то поступает и что-то уходит, при соответствующей интерпретации величин N , λ и T . Следующие примеры иллюстрируют широкий диапазон ее применений.

Пример 1

Если λ — скорость поступления требований в линию передачи, N_Q — среднее число пакетов, ожидающих в очереди (но не передающихся), W — среднее время, которое пакет ожидает в очереди (не включая время передачи), то, согласно теореме Литтла, имеем

$$N_Q = \lambda W.$$

Более того, если \bar{X} — среднее время передачи, то, согласно теореме Литтла, получаем, что среднее число пакетов, находящихся в процессе передачи, равно

$$\rho = \lambda \bar{X}.$$

Так как может передаваться только один пакет, получаем, что ρ также равно коэффициенту использования линии, т. е. доле времени, в течение которого линия передает пакеты.

Пример 2

Рассмотрим сеть линий связи, в которой пакеты поступают в n различных узлов с соответствующими скоростями $\lambda_1, \dots, \lambda_n$. Если N — среднее общее число пакетов внутри сети, то (независимо от распределения длины пакетов и способа маршрутизации пакетов) средняя задержка пакета равна

$$T = \frac{N}{\sum_{i=1}^n \lambda_i}.$$

Более того, из теоремы Литтла получаем также, что $N_i = \lambda_i T_i$, где N_i и T_i соответственно среднее число пакетов в системе и средняя задержка для пакетов, поступающих в узел i .

Пример 3

Пакет поступает в линию передачи через каждые K секунд, причем первый пакет поступает в момент 0. Все пакеты имеют одинаковую длину и требуют αK секунд для передачи, где $\alpha < 1$. Задержка на обработку и распространение одного пакета равна P секундам. Скорость поступления $\lambda = 1/K$. Так как пакеты поступают с постоянной скоростью (промежутки между поступлениями равны), то нет задержки из-за очереди; таким образом, время пребывания пакета в системе (с учетом времени распространения) равно

$$T = \alpha K + P.$$

Согласно теореме Литтла, имеем

$$N = \lambda T = \alpha + \frac{P}{K}.$$

Одно замечание должно быть сделано для правильного понимания формулы в этом примере. Здесь число требований в системе $N(t)$ является детерминированной функцией времени. Ее вид показан на рис. 3.2 для случая, когда $K < \alpha K + P < 2K$; можно увидеть, что $N(t)$ не сходится ни к какому пределу (система не достигает статистического равновесия). Поэтому теорема Литтла будет справедлива, если N рассматривать как среднее значение $N(t)$ по большому интервалу времени, т. е.

$$N = \lim_{t \rightarrow \infty} \frac{\int_0^t N(\tau) d\tau}{t}.$$

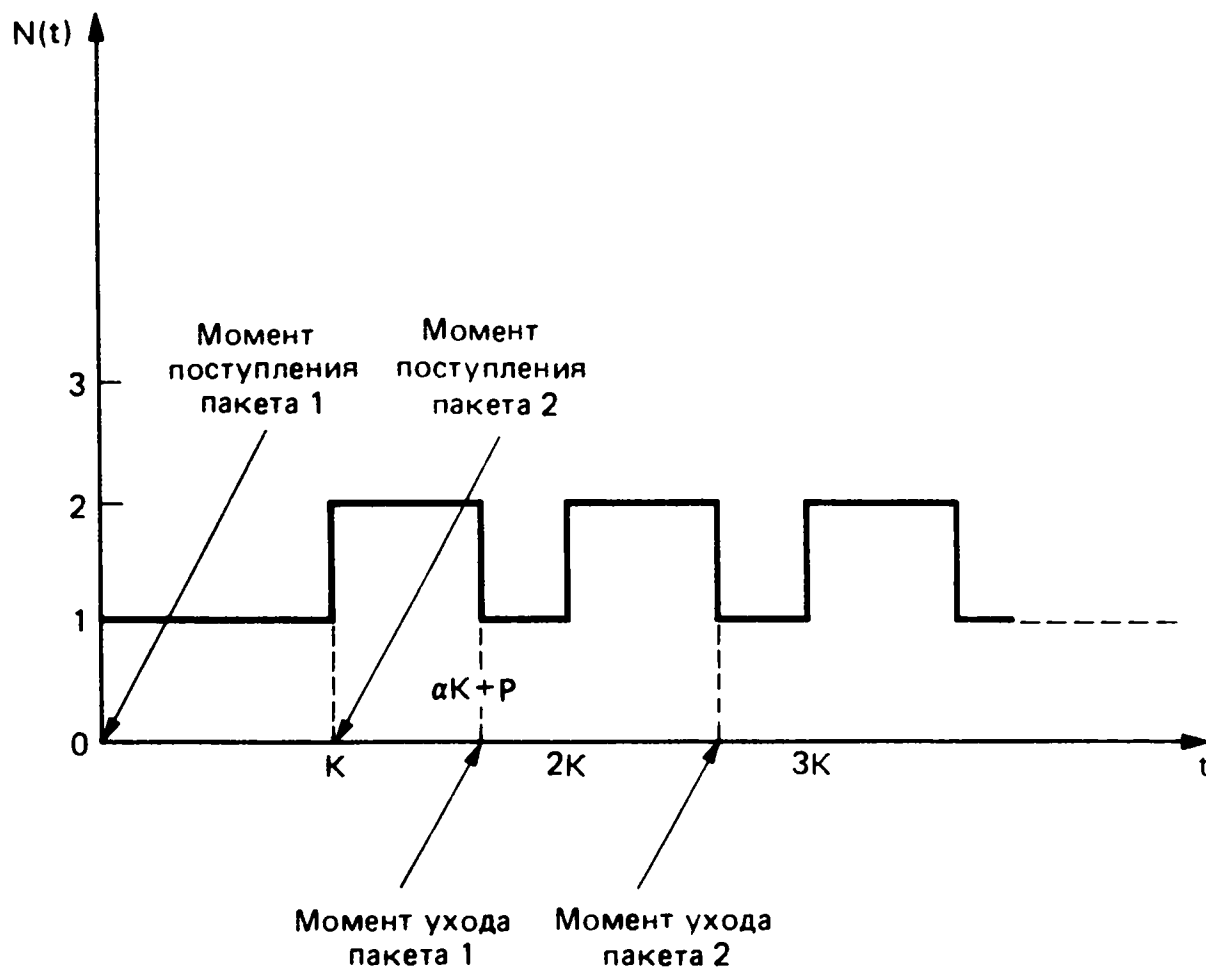


Рис. 3.2. Число требований в системе в примере 3. $N(t)$ является детерминированной функцией и не сходится при $t \rightarrow \infty$. Однако теорема Литтла справедлива, если N , λ и T интерпретируются как средние по времени.

Пример 4

Рассмотрим систему управления потоком, такую как в подразд. 2.8.1, с размером окна для каждого сеанса равным N . Предположим, что сеанс всегда имеет пакеты для передачи и что для подтверждений требуется пренебрежимо малое время; тогда, если пакет i поступает к месту назначения, пакет $i + N$ сразу же входит в сеть. Так как число пакетов в системе в расчете на один сеанс всегда равно N , согласно теореме Литтла, скорость поступления пакетов в систему для каждого сеанса λ и средняя задержка пакета связаны равенством $N = \lambda T$. Таким образом, если возрастает нагрузка в сети и T увеличивается, λ должно уменьшаться. Заметим также, что если сеть перегружена и способна передавать в единицу времени только λ пакетов для каждого сеанса, то увеличение для всех сеансов размера окна N приведет лишь к росту задержки T .

Пример 5

Рассмотрим систему массового обслуживания с K обслуживающими приборами и местом для ожидания самое большее для $N \geq K$ требований (либо стоящих в очереди, либо обслуживаемых). Система всегда заполнена; предположим, что она начинает работать с N требованиями и что если требование покидает систему, то оно немедленно заменяется на новое. (Системы массового обслуживания такого типа называются *замкнутыми*.) Предположим, что среднее время обслуживания требования равно \bar{X} . Нужно найти среднее время пребывания требования в системе T . Применяя теорему Литтла дважды, первый раз для всей системы, получаем $N = \lambda T$, а затем для части системы, в которой обслуживаются требования, получаем $K = \lambda \bar{X}$ (так как все обслуживающие приборы постоянно заняты). Исключая λ в этих соотношениях, находим

$$T = N\bar{X}/K.$$

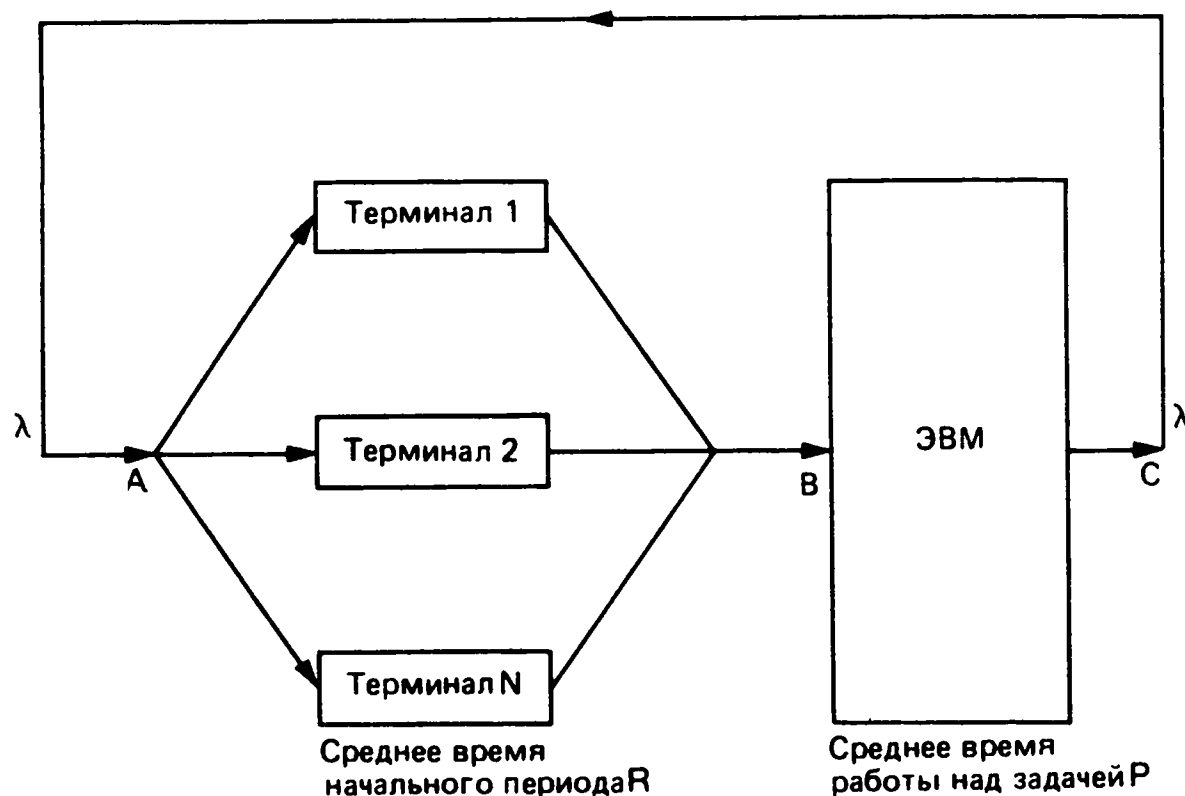


Рис. 3.3. N терминалов подсоединены к ЭВМ с разделением времени. Для того чтобы оценить максимально достижимую пропускную способность, предположим, что уходящий пользователь сразу же повторно входит в систему или, что эквивалентно, сразу же заменяется новым пользователем.

Пример 6. Оценка пропускной способности для системы с разделением времени

Теорема Литтла иногда может использоваться для получения границ для достижимой пропускной способности системы λ . В частности, известные границы для N и T с помощью соотношения $\lambda = N/T$ могут быть преобразованы в границы для пропускной способности. Например, рассмотрим систему с разделением времени для ЭВМ с N терминалами. Пользователь входит в систему через терминал и после начального периода средней длительности R его задача, которая требует среднего времени работы P , поступает в ЭВМ. Очередь задач находится внутри ЭВМ и обслуживается одним центральным процессором в соответствии с некоторыми (здесь не указываемыми) приоритетами или с разделением времени.

Оценим пропускную способность системы (в числе выполняемых задач за единицу времени), а также среднюю задержку пользователя. Так как нас интересует максимально достижимая пропускная способность, предположим, что всегда найдется пользователь, готовый занять место пользователя, который покинул систему, поэтому число пользователей в системе всегда равно N . По этой причине можно принять модель, в которой пользователь, который покинул систему, сразу же повторно входит в нее, как показано на рис. 3.3.

Применяя теорему Литтла к части системы между входом терминалов и выходом системы (точки A и C на рис. 3.3), получаем

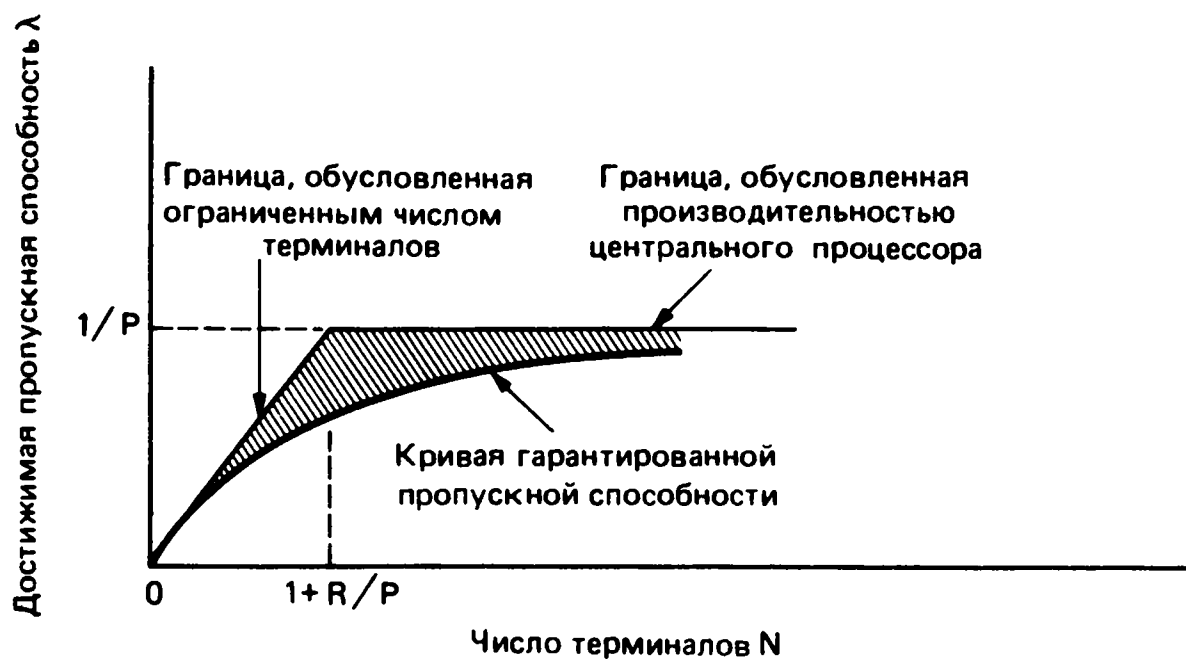
$$\lambda = N/T, \quad (3.3)$$

где T — среднее время пребывания пользователя в системе. Имеем

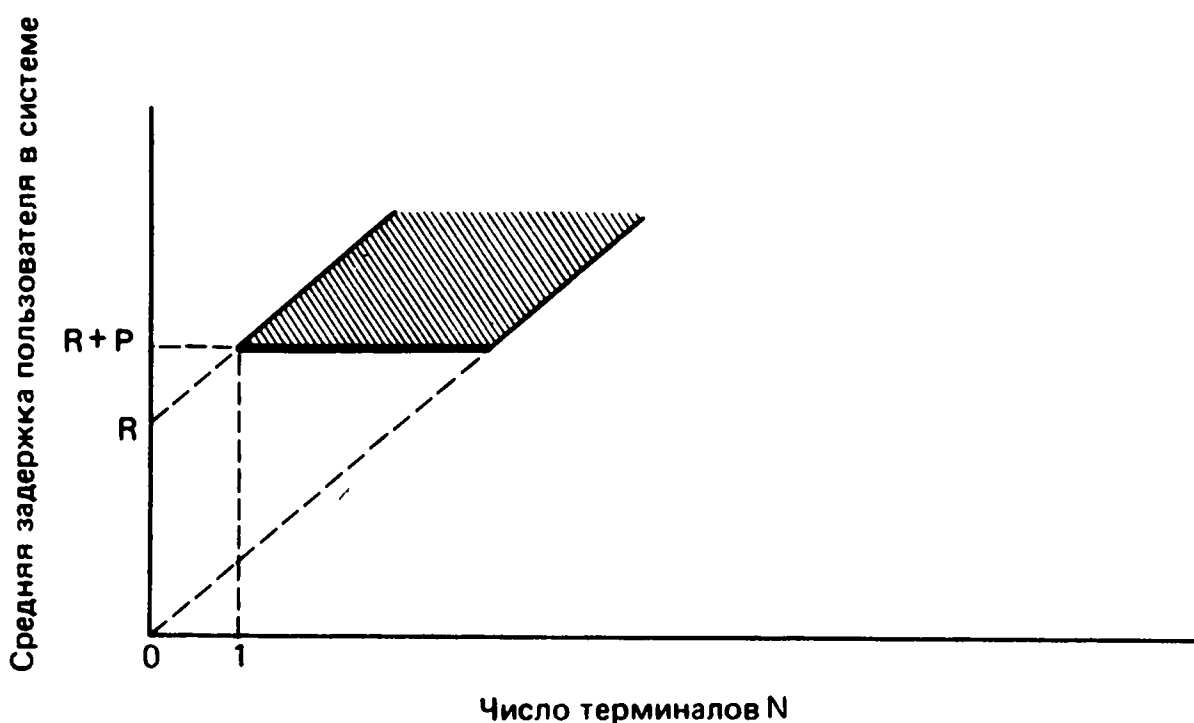
$$T = R + D, \quad (3.4)$$

где D — средняя задержка между моментом, когда задача поступила в ЭВМ, и моментом, когда ее выполнение было завершено. Так как величина D может изменяться от P (случай, когда задача пользователя не должна ждать, пока другие программы будут выполнены) до NP (случай, когда задача пользователя должна ждать выполнения задач всех других пользователей; ср. с примером 5), имеем

$$R + P \leq T \leq R + NP. \quad (3.5)$$



(а)



(б)

Рис. 3.4. Границы для пропускной способности и средней задержки пользователя в системе с разделением времени.

а — границы достижимой пропускной способности [неравенства (3.8)]; *б* — границы средней задержки пользователя при полностью загруженной системе [неравенства (3.9)]. Задержка возрастает, по существу, пропорционально числу терминалов N .

Учитывая эти неравенства и равенство (3.3), получаем

$$\frac{N}{R + NP} \leq \lambda \leq \frac{N}{R + P}. \quad (3.6)$$

Пропускная способность λ также ограничена сверху производительностью ЭВМ. В частности, так как время решения задачи в среднем равно P единицам, то ЭВМ не может при длительной работе выполнять более чем $1/P$ задач в единицу времени, т. е.

$$\lambda \leq 1/P. \quad (3.7)$$

(Этот результат можно также получить, применяя теорему Литтла между точками входа и выхода центрального процессора ЭВМ.)

Из соотношений (3.6) и (3.7) получаем границы

$$\frac{N}{R + NP} \leq \lambda \leq \min \left\{ \frac{1}{P}, \frac{N}{R + P} \right\} \quad (3.8)$$

для максимально достижимой пропускной способности. Используя $T = N/\lambda$, получим также границы для средней задержки пользователя в случае, когда система полностью загружена

$$\max \{NP, R + P\} \leq T \leq R + NP. \quad (3.9)$$

Эти соотношения показаны на рис. 3.4.

Можно увидеть, что по мере увеличения числа терминалов N пропускная способность приближается к максимальной $1/P$, тогда как средняя задержка пользователя возрастает, по существу, пропорционально N . При $N < 1 + R/P$ пропускная способность критически зависит от числа терминалов; в этом случае ресурс ЭВМ остается неиспользуемым в течение значительной доли времени, так как пользователи большую часть времени находятся в процессе начального периода входа в ЭВМ. В противоположность этому при $N > 1 + R/P$ пропускная способность критически зависит от вычислительной мощности ЭВМ. Интересно заметить, что, в то время как точное значение максимально достижимой пропускной способности зависит от параметров системы, таких как вероятностные характеристики времени начального периода и времени обработки, а также от способа, по которому задачи обслуживаются центральным процессором, полученные границы не зависят от этих параметров. Это удобство является следствием общности теоремы Литтла.

3.3. Система массового обслуживания $M/M/1$

Система массового обслуживания $M/M/1$ — это система с одной очередью и одним обслуживающим прибором (в контексте систем связи — с одной линией связи). Моменты поступления требований образуют пуассоновский поток с интенсивностью λ , а распределение вероятностей времени обслуживания является экспоненциальным со средним значением, равным $1/\mu$ секунд. Объясним коротко смысл принятых обозначений. Обозначение $M/M/1$ является стандартным для теории массового обслуживания.

1. Первый символ указывает на природу процесса поступления требований (например, M обозначает процесс без памяти, который здесь называется пуассоновским процессом (в нем интервалы между поступлениями требований распределены экспоненциально), G обозначает произвольное распределение интервалов между требованиями, D — детерминированные интервалы).

2. Второй символ указывает на природу распределения вероятностей длительностей обслуживания (например, M , G , D — обозначения для экспоненциального, произвольного и детерминированного распределений соответственно). Во всех случаях последовательные интервалы между поступлениями требований и длительности обслуживаний считаются статистически независимыми.

3. Последнее число указывает на количество обслуживающих приборов.

Мы уже установили из теоремы Литтла соотношения

$$N = \lambda T, N_Q = \lambda W$$

между основными величинами:

- N — среднее число требований в системе,
- T — среднее время пребывания требования в системе,
- N_Q — среднее число требований, ожидающих в очереди,
- W — среднее время, в течение которого требование ожидает в очереди.

Однако величины N , T , N_Q и W не могут быть найдены, пока мы не узнаем нечто большее о статистических характеристиках системы. Используя эти характеристики, можно определить стационарные вероятности

- p_n — вероятность пребывания n требований в системе, $n = 0, 1, \dots$

С помощью этих вероятностей можно получить

$$N = \sum_{n=0}^{\infty} n p_n,$$

а использование теоремы Литтла дает

$$T = N/\lambda.$$

Имеются аналогичные формулы для N_Q и W . В приложении Б приводится сводка результатов по системе $M/M/1$ и другим системам, которые рассматриваются позже.

Анализ системы $M/M/1$, так же как и анализ других родственных систем, таких как $M/M/m$ или $M/M/\infty$, основывается на теории цепей Маркова, краткое изложение результатов которой дается в приложении А. Другой подход состоит в том, чтобы использовать простые графические рассуждения, основанные на понятии среднего остаточного времени, которое вводится в разд. 3.5. При использовании этого подхода не требуется предположение об экспоненциальном распределении времени обслуживания, т. е. он применим к системе $M/G/1$. Однако определение выражений для стационарных вероятностей оказывается менее простым и удобным, чем в системе $M/M/1$. Читатель, желающий пропустить анализ, основанный на теории цепей Маркова, после ознакомления с предварительными сведениями о пуассоновском процессе, которые даются в подразд. 3.3.1 и 3.3.2, может начать сразу с изучения системы $M/G/1$ в разд. 3.5.

3.3.1. Основные результаты

Случайный процесс $\{A(t) \mid t \geq 0\}$, принимающий неотрицательные целые значения, называется *пуассоновским процессом* с интенсивностью λ , если

1) $A(t)$ — считающий процесс, который дает суммарное число моментов поступлений за время от 0 до t , т. е. $A(0) = 0$ и при $s \leq t$ величина $A(t) - A(s)$ равна числу моментов поступлений на интервале $(s, t]$;

2) числа моментов поступлений на непересекающихся временных интервалах являются независимыми;

3) число моментов поступлений в любом интервале длины τ имеет пуассоновское распределение с параметром $\lambda\tau$, т. е. для любых $t, \tau > 0$

$$P\{A(t + \tau) - A(t) = n\} = e^{-\lambda\tau} \frac{(\lambda\tau)^n}{n!}, \quad n = 0, 1, \dots \quad (3.10)$$

Перечислим некоторые интересующие нас свойства пуассоновского процесса.

(а) Интервалы между моментами поступлений независимы и распределены экспоненциально с параметром λ , т. е. если обозначить через t_n момент поступления i -го требования, то интервалы времени $\tau_n = t_{n+1} - t_n$ имеют функцию распределения вероятностей

$$P\{\tau_n \leq s\} = 1 - e^{-\lambda s}, \quad s \geq 0 \quad (3.11)$$

и взаимно независимы. (Соответствующая плотность вероятности $p(\tau_n) = \lambda e^{-\lambda\tau_n}$. Среднее значение и дисперсия величины τ_n равны $1/\lambda$ и $1/\lambda^2$ соответственно.)

(б) Для любых $t \geq 0$ и $\delta \geq 0$

$$P\{A(t + \delta) - A(t) = 0\} = 1 - \lambda\delta + o(\delta), \quad (3.12)$$

$$P\{A(t + \delta) - A(t) = 1\} = \lambda\delta + o(\delta), \quad (3.13)$$

$$P\{A(t + \delta) - A(t) \geq 2\} = o(\delta), \quad (3.14)$$

где используется общее обозначение $o(\delta)$ для любой функции от δ , такой, что

$$\lim_{\delta \rightarrow 0} \frac{o(\delta)}{\delta} = 0.$$

Эти равенства могут быть получены из (3.10) (см. задачу 3.10).

Заметим, что, если числа моментов поступлений в n непересекающихся интервалах независимы и имеют пуассоновские распределения с параметрами $\lambda\tau_1, \dots, \lambda\tau_n$, число моментов поступлений в объединении этих интервалов имеет пуассоновское распределение с параметром $\lambda(\tau_1 + \dots + \tau_n)$. Это следует из свойств пуассоновского распределения и обеспечивает согласованность (3.10) с требованием независимости в определении пуассоновского процесса (см. задачу 3.10). Другой факт, который мы будем часто использовать, состоит в том, что если два или более независимых пуассоновских процессов A_1, \dots, A_k объединяются в один

процесс $A = A_1 + A_2 + \dots + A_k$, то последний процесс является пуассоновским с интенсивностью, равной сумме интенсивностей его компонент (см. задачу 3.10).

Наше предположение относительно процесса обслуживания состоит в том, что *длительности обслуживания требований имеют экспоненциальное распределение с параметром μ , т. е. если s_n — время обслуживания n -го требования, то*

$$P\{s_n \leq s\} = 1 - e^{-\mu s}, \quad s \geq 0.$$

(Плотность распределения s_n равна $p(s_n) = \mu e^{-\mu s_n}$, а среднее значение и дисперсия равны $1/\mu$ и $1/\mu^2$ соответственно.) Кроме того, *длительности обслуживания s_n взаимно независимы, а также не зависят от всех интервалов между моментами поступления требований*. Параметр μ называется *скоростью обслуживания* и показывает, с какой скоростью работает обслуживающий прибор (т. е. число требований, обслуживаемых в единицу времени), если он занят.

Важный факт, относящийся к экспоненциальному распределению, заключается в том, что это распределение *не имеет памяти*, т. е. для интервалов между поступлениями и длительностей обслуживания τ_n и s_n соответственно

$$P\{\tau_n > r + t | \tau_n > t\} = P\{\tau_n > r\} \quad \text{при } r, t \geq 0,$$

$$P\{s_n > r + t | s_n > t\} = P\{s_n > r\} \quad \text{при } r, t \geq 0.$$

Это значит, что дополнительное время, необходимое для завершения обслуживания требования, не зависит от того, когда началось обслуживание. Аналогично время до момента поступления следующего требования не зависит от того, когда поступило предыдущее требование. Справедливость свойства, что система не имеет памяти, следует из следующих преобразований:

$$\begin{aligned} P\{\tau_n > r + t | \tau_n > t\} &= \frac{P\{\tau_n > r + t\}}{P\{\tau_n > t\}} = \\ &= \frac{e^{-\lambda(r+t)}}{e^{-\lambda t}} = e^{-\lambda r} = P\{\tau_n > r\}. \end{aligned}$$

Свойство отсутствия памяти и сделанные ранее предположения о независимости интервалов между поступлениями и длительностью обслуживания означают, что если известно $N(t)$, т. е. число требований в системе в момент t , то будущие моменты поступления или завершения обслуживания требований не зависят от моментов поступления требований, которые находятся в данный момент в системе, и от того, сколько времени уже обслуживалось требование, которое в данный момент находится в обслуживающем приборе (если такое имеется). Это означает, что $\{N(t) | t \geq 0\}$

является цепью Маркова с непрерывным временем (марковским процессом).

Мы могли бы рассматривать процесс $N(t)$ в терминах теории цепей Маркова с непрерывным временем как в большей части литературы по системам массового обслуживания. Однако для наших целей в этом разделе достаточно использовать более простую теорию цепей Маркова с дискретным временем (краткая сводка результатов которой приводится в приложении А).

Обратим внимание на моменты

$$0, \delta, 2\delta, \dots, k\delta, \dots,$$

где δ — малое положительное число. Введем обозначение

N_k — число требований в системе в момент $k\delta$.

Поскольку $N_k = N(k\delta)$, а как уже было сказано, $N(t)$ — цепь Маркова с непрерывным временем, получаем, что $\{N_k \mid k = 0, 1, \dots\}$ — цепь Маркова с дискретным временем. Обозначим через P_{ij} соответствующие переходные вероятности

$$P_{ij} = P\{N_{k+1} = j \mid N_k = i\}.$$

Заметим, что P_{ij} зависит от δ , но, для того чтобы сохранить простые обозначения, мы не будем учитывать эту зависимость. Используя формулы (3.12)–(3.14), получаем

$$P_{00} = 1 - \lambda\delta + o(\delta), \quad (3.15)$$

$$P_{ii} = 1 - \lambda\delta - \mu\delta + o(\delta), \quad i \geq 1, \quad (3.16)$$

$$P_{i, i+1} = \lambda\delta + o(\delta), \quad i \geq 0, \quad (3.17)$$

$$P_{i, i-1} = \mu\delta + o(\delta), \quad i \geq 1, \quad (3.18)$$

$$P_{ij} = o(\delta), \quad i \text{ и } j \neq i, i+1, i-1.$$

Для того чтобы понять, как получаются эти уравнения, заметим, что в состоянии $i \geq 1$ вероятность того, что в интервале $I_k = (k\delta, (k+1)\delta)$ не поступит ни одно требование и ни одно требование не уйдет из системы, равна $(e^{-\lambda\delta})(e^{-\mu\delta})$; это следует из того, что число поступлений и число уходов имеют пуассоновское распределение и не зависят друг от друга. Разлагая эту вероятность в ряд по δ , получаем

$$\begin{aligned} P\{\text{в интервале } I_k \text{ 0 требований поступило и 0 ушло}\} &= \\ &= 1 - \lambda\delta - \mu\delta + o(\delta). \end{aligned} \quad (3.19)$$

Аналогично имеем, что

$$\begin{aligned} P\{\text{в интервале } I_k \text{ 0 требований поступило и 1 ушло}\} &= \\ &= \mu\delta + o(\delta), \end{aligned}$$

$$\begin{aligned} P\{\text{в интервале } I_k \text{ 1 требование поступило и 0 ушло}\} &= \\ &= \lambda\delta + o(\delta). \end{aligned}$$

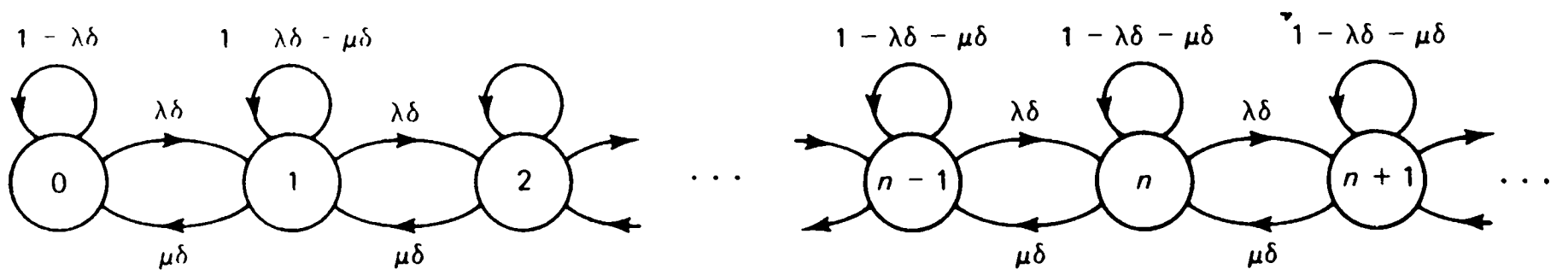


Рис. 3.5. Цепь Маркова с дискретным временем для системы $M/M/1$. Состояние n соответствует наличию n требований в системе. Переходные вероятности показаны с точностью до члена $o(\delta)$.

Сумма этих вероятностей равна единице плюс $o(\delta)$. Таким образом, вероятность более чем одного поступления или ухода пренебрежимо мала при малых δ . Это означает, что p_{ii} — вероятность одинакового числа $i \geq 1$ поступлений и уходов в интервале I_k с точностью до $o(\delta)$ совпадает с вероятностью (3.19); это показывает справедливость равенства (3.16). Соотношения (3.15), (3.17) и (3.18) устанавливаются аналогичным образом.

Диаграмма переходов из состояния в состояние цепи Маркова $\{N_k\}$ показана на рис. 3.5, на нем члены $o(\delta)$ опущены.

Рассмотрим теперь стационарные вероятности

$$p_n = \lim_{k \rightarrow \infty} P \{N_k = n\} = \lim_{t \rightarrow \infty} P \{N(t) = n\}.$$

Заметим, что при любых $k \geq 1$, $n \geq 0$ в течение времени от δ до $k\delta$ общее число переходов из состояния n в $n+1$ должно отличаться от общего числа переходов из $n+1$ в n не более чем на 1. Таким образом, в стационарном режиме вероятность того, что система находится в состоянии n и при следующем переходе попадает в состояние $n+1$, равна вероятности того, что система находится в состоянии $n+1$ и делает переход в состояние n , т. е.

$$p_n \lambda \delta + o(\delta) = p_{n+1} \mu \delta + o(\delta). \quad (3.20)$$

(Эти уравнения называются уравнениями глобального баланса, соответствующими множеству состояний $\{0, 1, \dots, n\}$ и $\{n+1, n+2, \dots\}$. См. приложение А, где приведены эти уравнения в более общей форме и дана их интерпретация аналогично приведенной при получении (3.20).) Так как p_n не зависит от δ , устремляя $\delta \rightarrow 0$ в (3.20), получаем

$$p_{n+1} = \rho p_n, \quad n = 0, 1, \dots, \text{ где } \rho = \lambda/\mu.$$

Отсюда следует, что

$$p_{n+1} = \rho^{n+1} p_0, \quad n = 0, 1, \dots \quad (3.21)$$

Если $\rho < 1$ (скорость обслуживания превышает скорость поступлений), все вероятности p_n являются положительными, а сумма их равна единице; таким образом,

$$1 = \sum_{n=0}^{\infty} p_n = \sum_{n=0}^{\infty} \rho^n p_0 = \frac{p_0}{1-\rho}. \quad (3.22)$$

Это соотношение вместе с формулой (3.21) окончательно дает

$$p_n = \rho^n (1 - \rho), \quad n = 0, 1, \dots \quad (3.23)$$

Теперь можно вычислить среднее число требований в системе в стационарном режиме

$$\begin{aligned} N &= \lim_{t \rightarrow \infty} E \{N(t)\} = \sum_{n=0}^{\infty} n p_n = \sum_{n=0}^{\infty} n \rho^n (1 - \rho) = \\ &= \rho (1 - \rho) \sum_{n=0}^{\infty} n \rho^{n-1} = \rho (1 - \rho) \frac{\partial}{\partial \rho} \left(\sum_{n=0}^{\infty} \rho^n \right) = \\ &= \rho (1 - \rho) \frac{\partial}{\partial \rho} \left(\frac{1}{1-\rho} \right) = \rho (1 - \rho) \frac{1}{(1-\rho)^2} \end{aligned}$$

и, наконец, используя равенство $\rho = \lambda/\mu$, получаем

$$N = \frac{\rho}{1-\rho} = \frac{\lambda}{\mu - \lambda}. \quad (3.24)$$

Это равенство показано графически на рис. 3.6. При увеличении ρ значение N также увеличивается и при $\rho \rightarrow 1$ имеем $N \rightarrow \infty$. График справедлив при $\rho < 1$. Если $\rho > 1$, обслуживающий прибор не может справиться с потоком поступлений требований и очередь неограниченно возрастает. В терминах системы пакетной передачи $\rho > 1$ означает, что $\lambda L > C$, где λ — скорость поступления (в числе пакетов в секунду), L — средняя длина пакета в битах, C — пропускная способность канала в битах в секунду.

Средняя задержка требования (время ожидания в очереди плюс время обслуживания), согласно теореме Литтла, равна

$$T = \frac{N}{\lambda} = \frac{\rho}{\lambda(1-\rho)}. \quad (3.25)$$

Подставляя $\rho = \lambda/\mu$, получаем

$$T = 1/(\mu - \lambda). \quad (3.26)$$

Среднее время ожидания в очереди W равно средней задержке пакета T минус среднее время обслуживания $1/\mu$, поэтому

$$W = \frac{1}{\mu - \lambda} - \frac{1}{\mu} = \frac{\rho}{\mu - \lambda}.$$

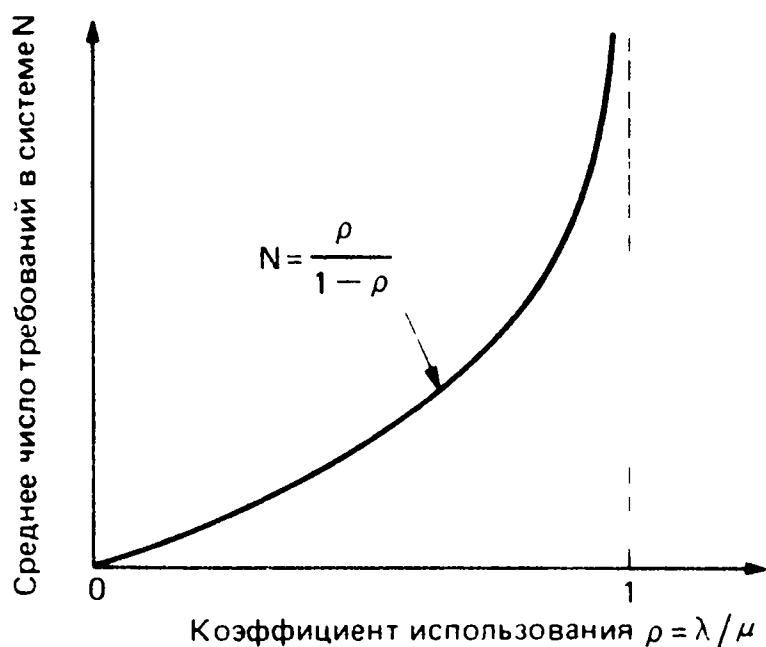


Рис. 3.6. Среднее число требований в системе $M/M/1$ в зависимости от коэффициента использования. При $\rho \rightarrow 1$ имеем $N \rightarrow \infty$.

Имеем $\rho = 1 - p_0$, где p_0 — вероятность того, что в системе нет требований; получается другое доказательство формулы для p_0 (см. (3.22)).

Проиллюстрируем эти результаты на некоторых примерах, относящихся к сетям передачи данных.

Пример 7. Увеличение скорости поступления и скорости передачи в одинаковое число раз

Рассмотрим систему пакетной передачи, в которой скорость поступления (в пакетах в секунду) увеличивается от λ до $K\lambda$, где K — некоторый числовой коэффициент. Распределение длины пакетов остается неизменным, а пропускная способность возрастает в K раз, так что среднее время передачи пакета становится равным $1/(K\mu)$ вместо $1/\mu$. Отсюда следует, что коэффициент использования линии ρ и, следовательно, среднее число пакетов в системе остаются неизменными

$$N = \frac{\rho}{1 - \rho} = \frac{\lambda}{\mu - \lambda}.$$

Однако средняя задержка пакета теперь будет равна $T = N/(K\lambda)$ и, следовательно, уменьшается в K раз. Иначе говоря, *линия передачи, которая передает в K раз быстрее, будет передавать в K раз больше пакетов в секунду со средней задержкой пакета в K раз меньшей*. Этот результат общий и даже применим к сетям связи. Как показано на рис. 3.7, при увеличении скорости поступления и скорости обслуживания в K раз вероятностные характеристики процесса массового обслуживания не изменяются, за исключением временного масштаба — процесс ускоряется в K раз. Таким образом, когда пакет поступает в систему, он обнаруживает перед собой в вероятностном смысле такое же число пакетов, как и в случае низкоскоростной линии передачи. Однако пакеты, стоящие впереди него, будут продвигаться в K раз быстрее.

Пример 8. Статистическое уплотнение по сравнению с временным и частотным уплотнением

Предположим, что m статистически одинаковых и независимых пуассоновских потоков пакетов передаются по линии связи со скоростью λ/m пакетов в секунду каждый. Длины пакетов во всех потоках независимы и имеют экспо-

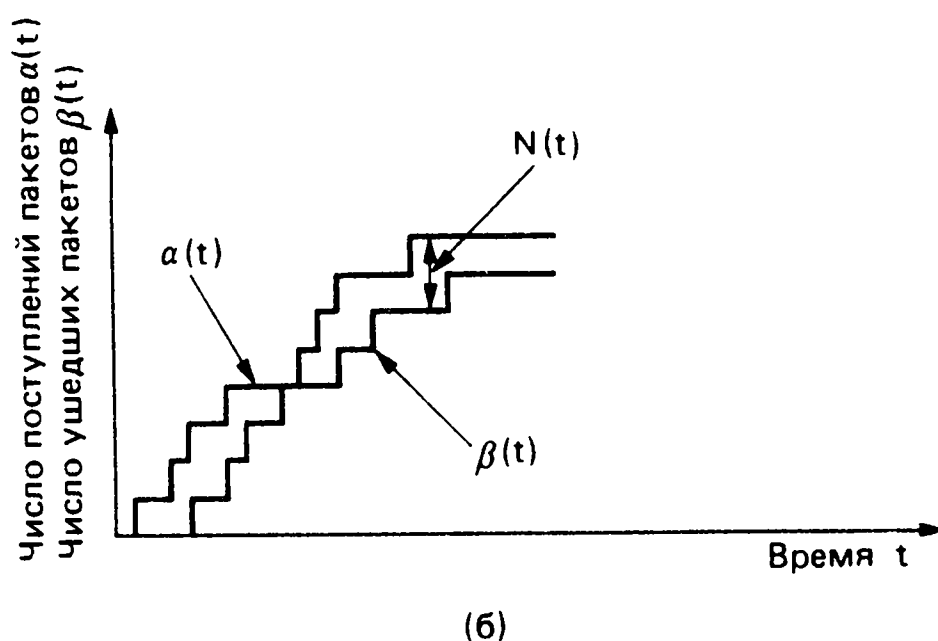
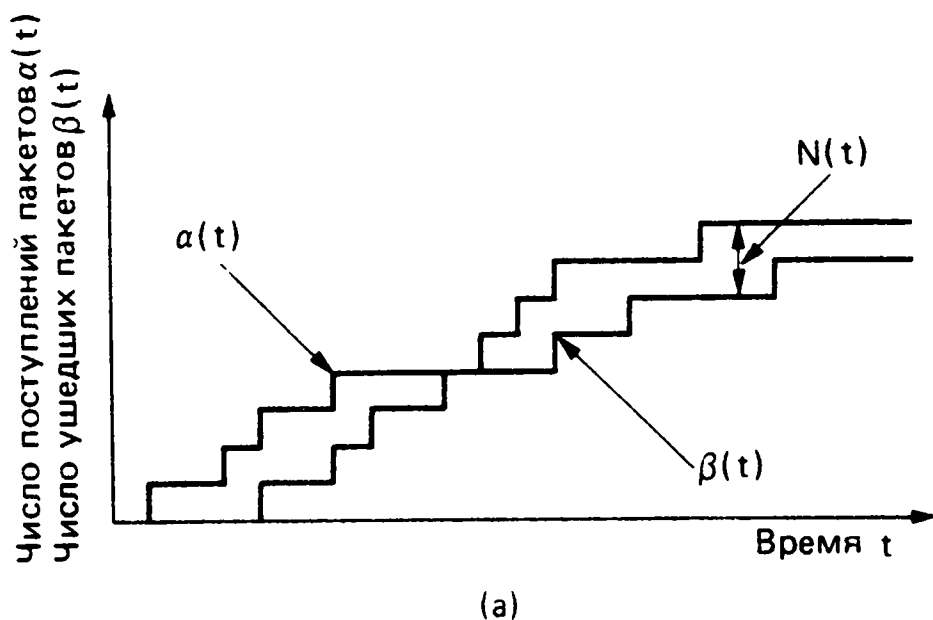
Из теоремы Литтла получаем, что среднее число требований в очереди равно

$$N_Q = \lambda W = \frac{\rho^2}{1 - \rho}.$$

Очень полезно рассматривать величину ρ как *коэффициент использования* системы массового обслуживания, т. е. как стационарную долю времени, в течение которой занят обслуживающий прибор. Мы этот коэффициент использовали ранее в более общих условиях при применении теоремы Литтла (пример 1 из разд. 3.2).

Рис. 3.7. Увеличение интенсивности поступления пакетов и скорости обслуживания в одинаковое число раз (см. пример 7).

a — траектории реализации числа поступивших пакетов $\alpha(t)$ и числа ушедших пакетов $\beta(t)$ в исходной системе; b — соответствующие траектории реализации числа поступивших пакетов $\alpha(t)$ и числа ушедших пакетов $\beta(t)$ в «ускоренной» системе, в которой интенсивность поступления и скорость обслуживания увеличены в два раза. Среднее число пакетов в системе остается прежним, а средняя задержка уменьшается в два раза, так как требования движутся в два раза быстрее.



ненциальное распределение. Среднее время передачи равно $1/\mu$. Если потоки сливаются в один пуассоновский поток с интенсивностью λ , как это происходит при статистическом уплотнении, средняя задержка пакета становится равной

$$T = 1/(\mu - \lambda).$$

Если вместо этого пропускная способность канала делится на m равных частей, по одной для каждого потока пакетов, как при временном или частотном уплотнении, каждая часть ведет себя как система массового обслуживания $M/M/1$ со скоростью поступления λ/m и средней скоростью обслуживания μ/m . Следовательно, средняя задержка пакета равна

$$T = \frac{m}{\mu - \lambda},$$

т. е. в m раз больше, чем при статистическом уплотнении.

Приведенные соображения показывают, что уплотнение большого числа потоков путем распределения их по отдельным каналам линии передачи является очень неэффективным в смысле задержки. Эффективность системы становится еще хуже, если пропускная способность каналов распределяется не прямо пропорционально интенсивностям соответствующих потоков; пропорциональное распределение невозможно провести (по крайней мере в модели, которая рассматривается здесь), если интенсивности потоков меняются с течением времени, поэтому сети передачи данных со многими конкурирующими потоками, имеющими малые периоды занятости, сгоятся на основе некоторой формы статистического уплотнения. Соображения в пользу временного или частотного уплотнения появляются, если все потоки являются регулярными (в противоположность

пуассоновскому) в том смысле, что пакет не поступает, если передается другой, и поэтому, если такой поток передается по предназначенной для него линии, пакетов, ожидающих в очереди, нет. Если несколько потоков такого типа статистически уплотняются в одной линии передачи, средняя задержка пакета уменьшается, но среднее время ожидания в очереди становится положительным. Например, в телефонии каждый поток представляет собой разговор, который является регулярным в указанном смысле, и временное и частотное уплотнение здесь еще широко используется.

3.3.2. Распределение числа требований в системе в момент поступления нового требования

В последующем анализе в нескольких случаях понадобятся вероятностные характеристики систем массового обслуживания в момент поступления нового требования. В некоторых системах моменты поступления требований являются в некотором смысле нетипичными, так что стационарные вероятности числа требований в момент поступления

$$a_n = \lim_{t \rightarrow \infty} P \{N(t) = n \mid \text{требование поступило сразу же после момента } t\} \quad (3.27)$$

необязательно равны соответствующим безусловным стационарным вероятностям

$$p_n = \lim_{t \rightarrow \infty} P \{N(t) = n\}. \quad (3.28)$$

Однако оказывается, что для системы $M/M/1$ имеем

$$p_n = a_n, \quad n = 0, 1, \dots \quad (3.29)$$

Это равенство для систем массового обслуживания с пуассоновским потоком поступления требований справедливо при очень общих условиях независимо от вида распределения длительностей обслуживания. Одно дополнительное требование, которое необходимо, состоит в том, чтобы будущие моменты поступления не зависели от числа требований, находящихся в данный момент в системе. Говоря более точно, предположим, что для любого момента t и длины интервала $\delta > 0$ число требований, поступивших в интервале $(t, t + \delta)$, не зависит от числа требований, находящихся в системе в момент t . По существу, при использовании свойств пуассоновского процесса это эквивалентно предположению, что в любой момент времени длительность обслуживания поступивших ранее требований и интервалы между будущими моментами поступлений являются независимыми; до некоторой степени это выполняется в системах пакетной передачи. В частности, это предположение справедливо, если процесс поступления требований является пуассоновским, а интервалы между моментами поступления и длительности обслуживания независимы.

Равенство $a_n = p_n$ выполняется, так как при нашем предположении события $\{N(t) = n\}$ и $\{\text{требование поступило сразу же}$

после момента t независимы. В результате условная вероятность (3.27) равна безусловной (3.28). Для более формального доказательства допустим, что $A(t, t + \delta)$ соответствует событию, что требование поступает в интервале $(t, t + \delta)$. Пусть

$$p_n(t) = P\{N(t) = n\}, \quad (3.30)$$

$$a_n(t) = P\{N(t) = n \mid \text{требование поступает сразу же после момента } t\}. \quad (3.31)$$

Используя формулы Байеса, имеем

$$\begin{aligned} a_n(t) &= \lim_{\delta \rightarrow 0} P\{N(t) = n \mid A(t, t + \delta)\} = \\ &= \lim_{\delta \rightarrow 0} \frac{P\{N(t) = n, A(t, t + \delta)\}}{P\{A(t, t + \delta)\}} = \\ &= \lim_{\delta \rightarrow 0} \frac{P\{A(t, t + \delta) \mid N(t) = n\} P\{N(t) = n\}}{P\{A(t, t + \delta)\}}. \end{aligned} \quad (3.32)$$

По предположению событие $A(t, t + \delta)$ не зависит от числа требований, находящихся в системе в момент t . Следовательно,

$$P\{A(t, t + \delta) \mid N(t) = n\} = P\{A(t, t + \delta)\}$$

и из (3.32) находим

$$a_n(t) = P\{N(t) = n\} = p_n(t).$$

Устремляя $t \rightarrow \infty$, получаем (3.29).

Таким образом, мы показали, что в момент поступления требования вероятность обнаружить в системе n требований равна вероятности (безусловной) того, что n требований находятся в системе. Это справедливо в любой момент времени, а также в стационарном состоянии независимо от вида распределения вероятностей длительности обслуживания. Кратко это можно сказать так: *если процесс поступления требований пуассоновский, то поступившее в систему требование застаёт ее в обычном состоянии.*

Рассмотрим, что может произойти, если процесс поступления не является пуассоновским. Предположим, что интервалы между моментами поступления независимы и равномерно распределены между двумя и четырьмя секундами, в то время как длительности обслуживания требований равны одной секунде. Тогда требование, поступившее в систему, всегда обнаружит ее пустой. Но с другой стороны, среднее число требований, находящихся в системе, которое обнаруживает в случайный момент времени внешний наблюдатель, равно $1/3$.

Для того чтобы привести пример, когда моменты поступления образуют пуассоновский процесс, а длительности обслуживания требований, находящихся в системе, и будущие моменты поступления требований коррелированы, рассмотрим систему пакетной

передачи, в которой пакеты поступают в соответствии с пуассоновским процессом. Время передачи n -го пакета равно половине интервала между моментами поступления n -го и $(n + 1)$ -го пакета. Пакет, поступивший в систему, обнаружит ее пустой. Однако среднее число пакетов в системе, которое обнаружит внешний наблюдатель, как легко увидеть, равно $1/2$.

3.3.3. Распределение числа требований в системе в момент, когда требование ее покидает

Рассмотрим распределение числа требований в системе сразу же после того, как требование покидает систему, т. е. рассмотрим вероятность

$$d_n(t) = P \{ N(t) = n \mid \text{требование покидает систему непосредственно перед моментом } t \}.$$

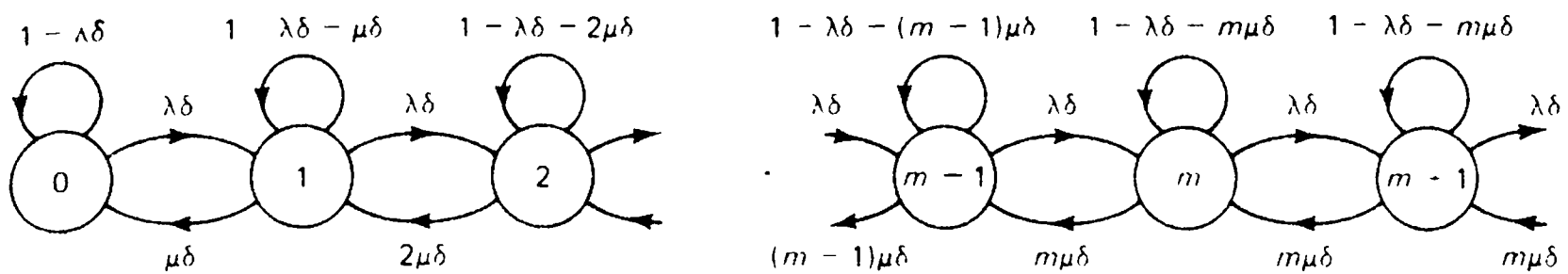
Соответствующие стационарные вероятности обозначаются как

$$d_n = \lim_{t \rightarrow \infty} d_n(t), \quad n = 0, 1, \dots$$

Оказывается, что при очень общих предположениях

$$d_n = a_n, \quad n = 0, 1, \dots$$

единственное требование, по существу, состоит в том, чтобы система достигала стационарного состояния, в котором стационарные вероятности положительны при всех n , и чтобы число требований $N(t)$ имело приращения, равные единице. (Эти предположения выполняются для стационарной системы $M/M/1$ ($\rho < 1$) и для большинства интересующих нас стационарных систем с одной очередью.) Для любой реализации поведения системы и любого n число заходов в состояние n равно бесконечности (с вероятностью единица). Это означает, что для любого увеличения состояния системы от n до $n + 1$ из-за поступления нового требования в дальнейшем будет соответствующее уменьшение от $n + 1$ до n из-за ухода требования из системы. Следовательно, при длительной работе доля переходов из n в $n + 1$ среди всех переходов из любого k в $k + 1$ равна доле переходов из $n + 1$ в n среди всех переходов из любого $k + 1$ в k ; это означает, что $d_n = a_n$. Следовательно, в стационарном состоянии для поступающего и уходящего требования система выглядит как статистически одинаковая. Если моменты поступления образуют пуассоновский процесс, то, как было установлено ранее, $a_n = p_n$, поэтому в этом случае в стационарном состоянии для поступающего и уходящего требования система будет выглядеть статистически идентичной той, которую обнаруживает в случайный момент времени внешний наблюдатель.


 Рис. 3.8. Цепь Маркова с дискретным временем для системы $M/M/t$.

3.4. Системы $M/M/t$, $M/M/\infty$ и $M/M/t/t$

Рассмотрим теперь ряд систем массового обслуживания, которые аналогичны системе $M/M/1$ в том, что процесс поступления требований является пуассоновским, длительности обслуживания независимы, экспоненциально распределены и независимы от интервалов между моментами поступления. Благодаря этим предположениям такие системы могут исследоваться с помощью цепей Маркова с непрерывным или дискретным временем. На основе соответствующих диаграмм переходов можно получить ряд уравнений, из которых могут быть найдены стационарные вероятности числа требований, находящихся в системе. Из теоремы Литтла затем можно получить среднюю задержку требования.

3.4.1. Система $M/M/t$ с t обслуживающими приборами

Система $M/M/t$ подобна системе $M/M/1$ за исключением того, что в ней имеется t обслуживающих приборов (или каналов в линии передачи в случае передачи данных). Требования, стоящие в начале очереди, направляются к любому имеющемуся в наличии обслуживающему прибору. Соответствующая диаграмма переходов показана на рис. 3.8.

Записывая уравнения равновесия для стационарных вероятностей p_n и устремляя $\delta \rightarrow 0$, получаем

$$\begin{aligned} \lambda p_{n-1} &= n\mu p_n, & n \leq t, \\ \lambda p_{n-1} &= t\mu p_n, & n > t. \end{aligned}$$

Из этих уравнений следует

$$p_n = \begin{cases} p_0 \frac{(t\rho)^n}{n!}, & n \leq t, \\ p_0 \frac{t^m \rho^n}{m!}, & n > t, \end{cases} \quad (3.33)$$

$$\text{где } \rho = \frac{\lambda}{t\mu} < 1. \quad (3.34)$$

Можно вычислить p_0 , используя формулу (3.33) и условие $\sum_{n=0}^{\infty} p_n = 1$. Получаем

$$p_0 = \left[1 + \sum_{n=1}^{m-1} \frac{(m\rho)^n}{n!} + \sum_{n=m}^{\infty} \frac{(m\rho)^n}{m!} \frac{1}{m^{n-m}} \right]^{-1}$$

и окончательно

$$p_0 = \left[\sum_{n=0}^{m-1} \frac{(m\rho)^n}{n!} + \frac{(m\rho)^m}{m! (1-\rho)} \right]^{-1}. \quad (3.35)$$

Вероятность того, что поступившее требование обнаружит, что в системе все обслуживающие приборы заняты, и будет поставлено в очередь для ожидания, равна

$$\begin{aligned} P \{ \text{встать в очередь} \} &= \sum_{n=m}^{\infty} p_n = \\ &= \sum_{n=m}^{\infty} \frac{p_0 m^m \rho^n}{m!} = \frac{p_0 (m\rho)^m}{m!} \sum_{n=m}^{\infty} \rho^{n-m} \end{aligned}$$

и окончательно

$$P_Q \triangleq P \{ \text{встать в очередь} \} = \frac{p_0 (m\rho)^m}{m! (1-\rho)}, \quad (3.36)$$

где p_0 можно найти из (3.35). Это соотношение известно как *С-формула Эрланга*; она широко используется в телефонии. (Датский ученый А. К. Эрланг считается одним из основателей теории массового обслуживания.)

Математическое ожидание числа требований, ожидающих в очереди (необслуживаемых), равно

$$N_Q = \sum_{n=0}^{\infty} n p_{m+n}.$$

Используя (3.33), получаем

$$N_Q = \sum_{n=0}^{\infty} n p_0 \frac{m^m \rho^{m+n}}{m!} = \frac{p_0 (m\rho)^m}{m!} \sum_{n=0}^{\infty} n \rho^n.$$

Используя (3.36) и равенство $(1-\rho) \sum_{n=0}^{\infty} n \rho^n = \rho/(1-\rho)$, кото-

рое встречалось при анализе системы $M/M/1$, окончательно будем иметь

$$N_Q = P_Q \frac{\rho}{1-\rho}. \quad (3.37)$$

Заметим, что

$$\frac{N_Q}{P_Q} = \frac{\rho}{1-\rho}$$

дает условное математическое ожидание числа требований, ожидающих в очереди при поступлении требования, при условии, что это требование направляется в очередь для ожидания; это математическое ожидание при заданном $\rho = \lambda/t\mu$ не зависит от числа обслуживающих приборов. Оно указывает, в частности, на то, что, если имеются требования, ожидающие в очереди, длина очереди в системе $M/M/t$ ведет себя так же, как в системе $M/M/1$ со скоростью обслуживания $t\mu$, равной суммарной скорости t обслуживающих приборов. Некоторое соображение, показывающее, что это действительно верно, следует из свойства отсутствия памяти у экспоненциального распределения.

Используя теорему Литтла и равенство (3.37), получаем среднее время W , которое требование ожидает в очереди,

$$W = \frac{N_Q}{\lambda} = \frac{\rho P_Q}{\lambda(1-\rho)}. \quad (3.38)$$

Следовательно, средняя задержка требования равна

$$T = \frac{1}{\mu} + W = \frac{1}{\mu} + \frac{\rho P_Q}{\lambda(1-\rho)},$$

и учитывая, что $\rho = \lambda/t\mu$, окончательно получаем

$$T = \frac{1}{\mu} + W = \frac{1}{\mu} + \frac{P_Q}{t\mu - \lambda}. \quad (3.39)$$

Снова, используя теорему Литтла, находим, что среднее число требований в системе равно

$$N = \lambda T = \frac{\lambda}{\mu} + \frac{\lambda P_Q}{t\mu - \lambda}$$

и с учетом $\rho = \lambda/t\mu$ получаем

$$N = t\rho + \frac{\rho P_Q}{1-\rho}.$$

Пример 9. Сравнение случаев использования одного и нескольких каналов при статистическом уплотнении

Рассмотрим линию связи, которая обслуживает t независимых пуассоновских потоков с интенсивностью λ/t каждый. Предположим, что линия разделена на t каналов, причем каждый канал предназначается для одного потока. Однако,

если поток не имеет пакета, ожидающего передачи, канал, соответствующий этому потоку, используется для передачи пакета из другого потока. Длительности передач пакетов по любому каналу распределены экспоненциально со средним значением $1/\mu$. Эта система подобно системе $M/M/1$ может быть описана цепью Маркова. Сравним среднюю задержку пакета в этой системе со средней задержкой в системе $M/M/1$ с той же интенсивностью поступления λ и скоростью обслуживания $m\mu$ (статистическое уплотнение одного канала, который имеет в m раз большую пропускную способность). В первом случае, согласно формуле (3.39), средняя задержка пакета

$$T = \frac{1}{\mu} + \frac{P_Q}{m\mu - \lambda},$$

а во втором случае средняя задержка пакета

$$\hat{T} = \frac{1}{m\mu} + \frac{\hat{P}_Q}{m\mu - \lambda},$$

где P_Q и \hat{P}_Q обозначают вероятности постановки в очередь для каждого случая. При $\rho \ll 1$ (слабо загруженная система) имеем $P_Q \cong 0$, $\hat{P}_Q \cong 0$ и

$$T/\hat{T} \cong m.$$

При ρ , близком к единице, имеем $P_Q \cong 1$, $\hat{P}_Q \cong 1$, $1/\mu \ll 1/(m\mu - \lambda)$ и

$$T/\hat{T} = 1.$$

Следовательно, при слабой нагрузке статистическое уплотнение с m каналами дает в m раз большую задержку, чем статистическое уплотнение с m каналами, объединенными в один (примерно так же, как временное и частотное уплотнение). При больших нагрузках отношение задержек близко к единице.

3.4.2. $M/M/\infty$ — система с бесконечным числом обслуживающих приборов

В предельном случае, когда в системе $M/M/m$ имеем $m = \infty$, получаем, согласно рис. 3.8,

$$\lambda p_{n-1} = n\mu p_n, \quad n = 1, 2, \dots$$

и, таким образом,

$$p_n = p_0 \left(\frac{\lambda}{\mu} \right)^n \frac{1}{n!}, \quad n = 1, 2, \dots$$

Из условия $\sum_{n=0}^{\infty} p_n = 1$ находим

$$p_0 = \left[1 + \sum_{n=1}^{\infty} \left(\frac{\lambda}{\mu} \right)^n \frac{1}{n!} \right]^{-1} = e^{-\lambda/\mu},$$

поэтому окончательно

$$p_n = \left(\frac{\lambda}{\mu} \right)^n \frac{e^{-\lambda/\mu}}{n!}, \quad n = 0, 1, \dots$$

Следовательно, в стационарном состоянии распределение вероятностей числа требований в системе — пуассоновское распределение с параметром λ/μ . Среднее число требований в системе равно

$$N = \lambda/\mu.$$

Согласно теореме Литтла, средняя задержка равна N/λ и

$$T = 1/\mu.$$

Последнее равенство можно получить иначе, используя тот факт, что в системе $M/M/\infty$ нет ожидания в очереди. Можно показать, что распределение вероятностей числа требований в системе пуассоновское, даже если распределение времени обслуживания не является экспоненциальным, т. е. это справедливо в системе $M/G/\infty$ (см. задачу 3.37).

Пример 10. Предположение о квазистационарности

Часто удобно предположить, что внешний поток пакетов, поступающий в узел подсети и предназначенный для некоторого другого узла подсети, может быть описан случайным стационарным процессом, который имеет постоянную скорость поступления битов (среднее число битов в секунду). Это является приближением ситуации, в которой скорость поступления медленно меняется со временем, т. е. имеет место квазистационарность.

Когда существует только несколько активных сеансов (т. е. пар пользователей) для данной пары источник — узел назначения, предположение о квазистационарности серьезно нарушается, так как добавление или окончание единственного сеанса могут изменить суммарную скорость поступления битов в значительное число раз. Однако если активно большое число сеансов, причем скорость поступления битов каждого сеанса мала по сравнению с общей скоростью, то правдоподобно, что предположение о квазистационарности приближенно выполняется. Дело в том, что добавление сеансов статистически сбалансировано с их прекращением; при этом изменение суммарной скорости относительно мало. В виде аналитического подтверждения предположим, что сеансы возникают в соответствии с пуассоновским процессом с интенсивностью λ и прекращаются через время, которое распределено экспоненциально со средним значением $1/\mu$. Тогда число активных сеансов n вычисляется так же, как и число требований, находящихся в системе $M/M/\infty$; в стационарном состоянии оно имеет пуассоновское распределение с параметром λ/μ . В частности, среднее значение и стандартное отклонение величины n равны

$$N = E \{n\} = \lambda/\mu,$$

$$\sigma_n = [E \{(n - N)^2\}]^{1/2} = (\lambda/\mu)^{1/2}.$$

Предположим, что i -й активный сеанс порождает поток в соответствии со случайным стационарным процессом, который имеет скорость поступления битов, равную γ_i бит/с. Допустим, что скорости γ_i — независимые случайные величины с одинаковым средним значением $E \{\gamma_i\} = \Gamma$ и вторым моментом $s_\gamma^2 = E \{\gamma_i^2\}$. Тогда суммарная скорость поступления битов для n активных сеансов будет слу-

чайной величиной $f = \sum_{i=1}^n \gamma_i$ со средним значением

$$F = E \{f\} = (\lambda/\mu) \Gamma.$$

Стандартное отклонение величины \bar{f} обозначается через σ_f и может быть записано в виде

$$\sigma_f^2 = E \left\{ \left(\sum_{i=1}^n \gamma_i \right)^2 \right\} - F^2.$$

Производя соответствующие преобразования (задача 3.21), получаем

$$\sigma_f = (\lambda/\mu)^{1/2} s_\gamma.$$

Следовательно, имеем

$$\sigma_f/F = (s_\gamma/\Gamma) (\mu/\lambda)^{1/2}. \quad (3.40)$$

Предположим теперь, что средняя скорость поступления битов в сеансе Γ мала по сравнению с общей F , т. е. справедливо предположение о том, что сеансов много и скорость поступления битов в каждом из них мала. Тогда, так как $\Gamma/F = \mu/\lambda$, имеем, что μ/λ мало. Если можно предположить, что s_γ/Γ небольшое, то из (3.40) следует, что σ_f/F мало. Следовательно, общая скорость поступления \bar{f} приблизительно равна константе, что оправдывает предположение о квазистационарности.

3.4.3. Система $M/M/t/t$ с потерями и с t обслуживающими приборами

Эта система подобна системе $M/M/t$ за исключением того, что, если требование при поступлении в систему обнаружит, что все t обслуживающих приборов заняты, оно не поступит в систему, а потеряется; эта модель широко используется в телефонии. (Последнее t в обозначении $M/M/t/t$ указывает на максимальное число требований, которые могут находиться в системе.) В сетях передачи данных такая модель может использоваться для исследования системы, в которой моменты поступления соответствуют запросам на установление виртуальных цепей между двумя узлами, а максимально возможное число виртуальных цепей равно t . Средняя длительность обслуживания $1/\mu$ в этом случае равна среднему времени использования виртуальной цепи.

Соответствующая диаграмма переходов состояний показана на рис. 3.9. Имеем

$$\lambda p_{n-1} = n\mu p_n, \quad n = 1, 2, \dots, t,$$

так что

$$p_n = p_0 \left(\frac{\lambda}{\mu} \right)^n \frac{1}{n!}, \quad n = 1, 2, \dots, t.$$

С учетом равенства $\sum_{n=0}^t p_n = 1$ получаем

$$p_0 = \left[\sum_{n=0}^t \left(\frac{\lambda}{\mu} \right)^n \frac{1}{n!} \right]^{-1}.$$

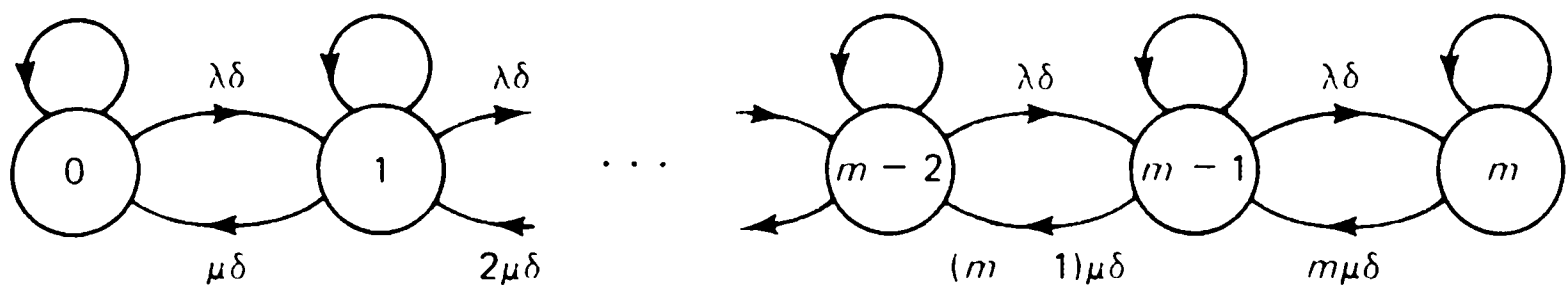


Рис. 3.9. Цепь Маркова с дискретным временем для системы $M/M/m/m$.

Вероятность того, что поступившее требование обнаружит, что все m обслуживающих приборов заняты, и, следовательно, будет потеряно, равна

$$p_m = \frac{(\lambda/\mu)^m / m!}{\sum_{n=0}^m (\lambda/\mu)^n / n!}.$$

Это равенство известно как *B-формула Эрланга*. Можно показать, что она остается в силе, даже если распределение вероятностей длительности обслуживания произвольное, т. е. справедлива для системы $M/G/m/m$ [204].

3.5. Система $M/G/1$

Рассмотрим систему массового обслуживания с одной очередью, в которой требования поступают в соответствии с пуассоновским процессом с интенсивностью λ , но длительности обслуживания требований имеют произвольное распределение, не обязательно экспоненциальное, как в системе $M/M/1$. Предположим, что требования обслуживаются в порядке поступления, X_i — длительность обслуживания i -го требования, случайные величины (X_1, X_2, \dots) одинаково распределены, взаимно независимы и не зависят от интервалов между моментами поступления.

Пусть

$\bar{X} = E \{X\} = \frac{1}{\mu}$ — средняя длительность обслуживания,

$\bar{X}^2 = E \{X^2\}$ — второй момент длительности обслуживания.

Наша цель состоит в том, чтобы получить и понять формулу Поллачека — Хинчина

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)}, \quad (3.41)$$

где W — математическое ожидание времени пребывания требования в очереди, а $\rho = \lambda \bar{X} = \lambda/\mu$. Согласно (3.41), общее время пребывания в очереди и в обслуживающем приборе равно

$$T = \bar{X} + \frac{\lambda \bar{X}^2}{2(1-\rho)}. \quad (3.42)$$

Применяя формулу Литтла для W и T , получим математическое ожидание числа требований в очереди N_Q и математическое ожидание числа требований в системе

$$N_Q = \frac{\lambda^2 \overline{X^2}}{2(1-\rho)}, \quad (3.43)$$

$$N = \rho + \frac{\lambda^2 \overline{X^2}}{2(1-\rho)}. \quad (3.44)$$

Например, если длительности обслуживания распределены экспоненциально, как в системе $M/M/1$, то имеем $\overline{X^2} = 2/\mu^2$ и равенство (3.41) сводится к формуле (см. подразд. (3.3.2))

$$W = \frac{\rho}{\mu(1-\rho)} \quad (M/M/1).$$

Если длительности обслуживания одинаковы для всех требований (система $M/D/1$, где D означает детерминированность), имеем $\overline{X^2} = 1/\mu^2$ и

$$W = \frac{\rho}{2\mu(1-\rho)} \quad (M/D/1). \quad (3.45)$$

Так как в случае $M/D/1$ при данном μ получается минимально возможное значение $\overline{X^2}$, из этого следует, что при одинаковых значениях λ и μ величины W , T , N_Q и N для системы массового обслуживания $M/D/1$ являются нижними границами для соответствующих величин в системе $M/G/1$. Интересно заметить, что W и N_Q для системы $M/D/1$ равны половине их значений в системе $M/M/1$. Вместе с тем значения T и N при малых ρ для $M/D/1$ такие же, как в системе $M/M/1$, и приближаются к половине их значений в системе $M/M/1$ по мере того, как ρ приближается к 1. Дело в том, что математическое ожидание длительности обслуживания одно и то же в обоих случаях и при малых ρ большую часть времени пребывания в системе требования находятся в обслуживающем приборе, а при больших ρ большую часть времени требования стоят в очереди.

Доказательство формулы Поллачека — Хинчина проведем, используя понятие *среднего остаточного времени обслуживания*. Понятие, аналогичное этому, будет полезно в некоторых последующих рассуждениях, например при рассмотрении системы $M/G/1$ с приоритетами и системы с резервированием, в которых часть времени обслуживания отводится для передачи пакетов (т. е. для обслуживания требований), а часть — для передачи управляющей информации или для резервирования передач пакетов.

Введем обозначения

W_i — время, которое i -е требование ожидает в очереди;

R_i — остаточное время обслуживания в момент поступления i -го требования. Это означает, что если требование j находится на обслуживании в момент появления требования i , то R_i равно промежутку времени от этого момента до завершения обслуживания требования j . Если нет требования, которое обслуживается (т. е. система пуста в момент поступления требования i), R_i равно нулю;

X_i — длительность обслуживания i -го требования;

N_i — число требований, находящихся в очереди в момент поступления i -го требования.

Имеем

$$W_i = R_i + \sum_{j=i-N_i}^{i-1} X_j.$$

Усредняя и используя то, что случайные величины N_i и $X_{i-1}, \dots, X_{i-N_i}$ независимы, получаем

$$E\{W_i\} = E\{R_i\} + E\left\{\sum_{j=i-N_i}^{i-1} E\{X_j | N_i\}\right\} = E\{R_i\} + \bar{X}E\{N_i\}.$$

Переходя к пределу при $i \rightarrow \infty$, будем иметь

$$W = R + \frac{1}{\mu} N_Q, \quad (3.46)$$

где R — среднее остаточное время, определяемое как

$$R = \lim_{i \rightarrow \infty} E\{R_i\}.$$

В равенстве (3.46) (и везде в этом разделе) все средние значения по большому интервалу времени рассматриваются как пределы, когда время или номера требований стремятся к бесконечности. Таким образом, W , R и N_Q — это пределы (при $i \rightarrow \infty$) соответственно средних времени ожидания, остаточного времени и числа требования в очереди в момент поступления i -го требования. Предположим, что такие пределы существуют и это справедливо для почти всех интересующих нас систем при условии, что $\lambda < \mu$. Заметим, что в (3.46) среднее число требований в очереди N_Q и среднее остаточное время R в момент поступления требования также равны среднему числу требований в очереди и среднему остаточному времени, которые внешний наблюдатель обнаруживает в случайный момент времени. Это справедливо вследствие пуассоновского характера процесса поступления требований, последнее означает, что распределение числа требований в системе в момент поступления нового требования является типичным (см. подразд. 3.3.2).

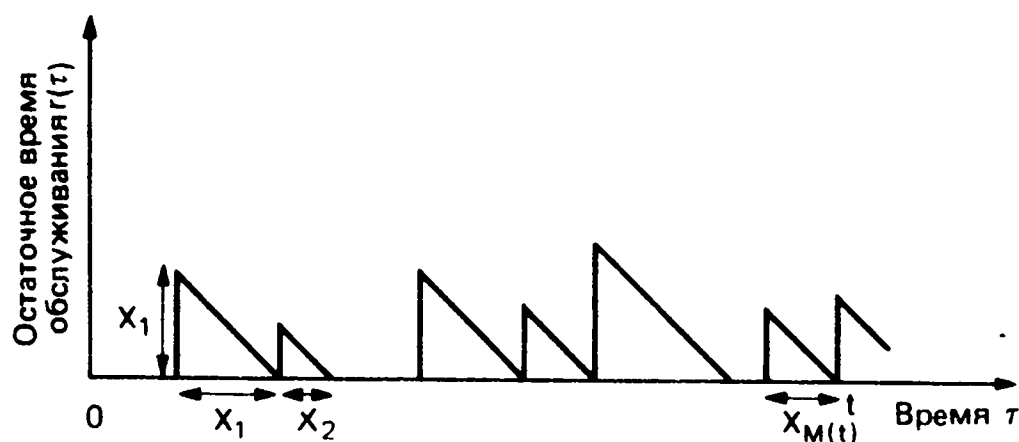


Рис. 3.10. Вычисление среднего остаточного времени обслуживания. Усредненное по интервалу времени $[0, t]$ среднее остаточное время обслуживания $r(\tau)$ равно

$$\frac{1}{t} \int_0^t r(\tau) d\tau = \frac{1}{t} \sum_{i=1}^{M(t)} \frac{1}{2} X_i^2 = \frac{1}{2} \frac{M(t)}{t} \frac{\sum_{i=1}^{M(t)} X_i^2}{M(t)},$$

где X_i — время обслуживания i -го требования, а $M(t)$ — число обслуживаний, которые завершились на отрезке $[0, t]$. Устремляя $t \rightarrow \infty$ и приравнявая средние по времени и по вероятности, получаем, что среднее остаточное время $R = (1/2) \lambda \bar{X}^2$.

Согласно теореме Литтла, получаем

$$N_Q = \lambda W$$

и подстановка в (3.46) дает

$$W = R + \rho W, \quad (3.47)$$

где $\rho = \lambda/\mu$ — коэффициент использования системы; таким образом, окончательно имеем

$$W = R/(1 - \rho). \quad (3.48)$$

Величину R можно вычислить из графических соображений. На рис. 3.10 остаточное время обслуживания $r(\tau)$ (т. е. время, необходимое для завершения обслуживания требования в момент τ) изображено как функция τ . Заметим, что если начинается новое обслуживание длительностью X , то $r(\tau)$ принимает значение X и убывает линейно в течение X единиц времени. Рассмотрим момент t , для которого $r(t) = 0$. Среднее по времени значение $r(\tau)$ на отрезке $[0, t]$ равно

$$\frac{1}{t} \int_0^t r(\tau) d\tau = \frac{1}{t} \sum_{i=1}^{M(t)} \frac{1}{2} X_i^2, \quad (3.49)$$

где $M(t)$ — число обслуживаний, которые завершились на отрезке $[0, t]$, а X_i — время обслуживания i -го требования. Можно записать это равенство в виде

$$\frac{1}{t} \int_0^t r(\tau) d\tau = \frac{1}{2} \frac{M(t)}{t} \frac{\sum_{i=1}^{M(t)} X_i^2}{M(t)}. \quad (3.50)$$

Предполагая, что следующие пределы существуют, получаем

$$\lim_{t \rightarrow \infty} \frac{1}{t} \int_0^t r(\tau) d\tau = \frac{1}{2} \lim_{t \rightarrow \infty} \frac{M(t)}{t} \lim_{t \rightarrow \infty} \frac{\sum_{i=1}^{M(t)} X_i^2}{M(t)}. \quad (3.51)$$

Оба предела в правой части являются соответственно средними по времени скоростью, с которой требования покидают систему (она равна скорости поступления требований), и вторым моментом времени обслуживания, а предел в левой части — среднее по времени остаточное время. Предполагая, что среднее по времени может быть заменено на среднее по вероятности, получаем

$$R = \frac{1}{2} \lambda \overline{X^2}. \quad (3.52)$$

Формула Поллачека — Хинчина

$$W = \frac{\lambda \overline{X^2}}{2(1 - \rho)} \quad (3.53)$$

теперь следует из (3.48) и (3.52). При выводе использовались два предположения: а) существуют стационарные средние значения W , R и N_0 и б) среднее по большому интервалу времени (равенство (3.51)) равно (с вероятностью единица) соответствующему среднему по вероятности. Эти предположения могут быть подтверждены путем применения закона больших чисел, но связанные с этим подробности находятся за пределами области нашего текста. Однако эти предположения на самом деле справедливы для интересующих нас систем, и мы будем без дальнейших обсуждений проводить аналогичные доказательства на основе графических построений и заменять средние по времени на средние по вероятности.

Одно любопытное свойство равенства (3.53) состоит в том, что очередь $M/G/1$ может иметь $\rho < 1$, но бесконечное W , если второй момент распределения времени обслуживания равен ∞ . В этом случае малая доля требований имеет очень большие длительности обслуживания. Если обнаруживается одно из этих требований, очень большое число вновь поступивших требований становится

в очередь и задерживается на значительную часть времени обслуживания. Таким образом, это приводит к увеличению W пропорционально квадрату времени обслуживания, что дает бесконечное значение W при бесконечном $\overline{X^2}$.

При выводе формулы Поллачека — Хинчина предполагалось, что требования обслуживаются в порядке их поступления, т. е. что число требований, полностью обслуженных между моментом поступления i -го требования и моментом начала его обслуживания, точно равно числу требований, которые находились в очереди в момент поступления i -го требования. Однако оказывается, что эта формула справедлива при любом порядке обслуживания требований, если порядок обслуживания определяется независимо от времени, требуемого для обслуживания. Для того чтобы понять это, предположим, что i -е и j -е требования находятся в очереди и меняются местами. Математическое ожидание (по длительностям обслуживаний требований, стоящих в очереди) времени ожидания в очереди для i -го требования поменяется тогда с аналогичным временем для требования j , но среднее по всем требованиям останется неизменным. Так как любой порядок обслуживания может рассматриваться как последовательность перестановок в очереди, формула Поллачека — Хинчина остается справедливой (см. также задачу 3.16).

Для того чтобы понять, почему формула Поллачека — Хинчина неверна в случае, когда порядок обслуживания зависит от времени обслуживания, рассмотрим очередь из двух требований, для которых нужно соответственно 10 и 2 единицы времени обслуживания. Предположим, что обслуживающий прибор начинает функционировать в момент 0; если первое требование обслуживается первым, то одно требование начинает обслуживаться в момент 0, а другое в момент 10. Если второе требование обслуживается первым, то одно требование начинает обслуживаться в момент 0, а другое в момент 2. Таким образом, средняя задержка ожидания в очереди для двух требований равна 5 в первом случае и 1 во втором случае. Очевидно, что время ожидания в очереди уменьшится, если обслуживать в первую очередь требования с меньшим временем обслуживания. В этом случае вывод формулы Поллачека — Хинчина невозможен из-за невыполнения равенства (3.46); дело в том, что требования, которые будут обслуживаться перед вновь поступившим, теперь не имеют среднее время обслуживания, равное $1/\mu$.

Пример 11. Задержка в системе ARQ

Рассмотрим одну из описанных в разд. 2.4 систем ARQ на n шагов назад. Предположим, что пакеты передаются в кадрах, которые имеют длину, равную единице времени, и максимальное время ожидания подтверждения перед началом повторной передачи пакета равно $n - 1$ кадру (рис. 3.11). В этой системе пакеты повторно передаются по двум причинам.

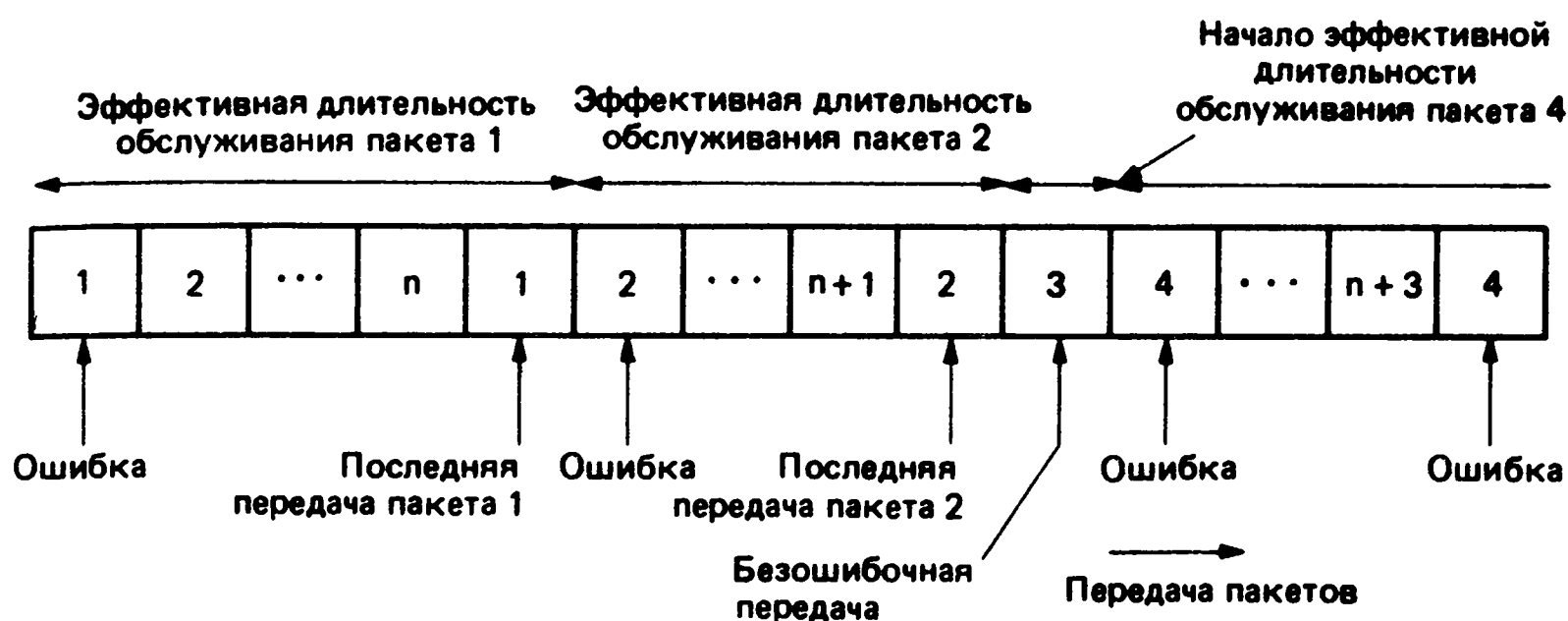


Рис. 3.11. Иллюстрация эффективных длительностей обслуживания пакетов в системе ARQ из примера 11. Например, пакет 2 имеет эффективную длительность обслуживания, равную $n + 1$, так как была ошибка при первой попытке передачи, идущей после последней передачи пакета 1, и не было ошибки при второй попытке.

1. Пакет, переданный в кадре i , может быть отброшен на приемной стороне из-за наличия в нем ошибок; в этом случае передатчик будет передавать пакеты (если они имеются) в кадрах $i + 1, i + 2, \dots, i + n - 1$, а потом вернется назад для повторной передачи пакета в кадре $i + n$.

2. Пакет, переданный в кадре i , мог бы считаться принятым приемником, но соответствующее подтверждение (в виде номера принятого пакета) может не поступить на передатчик к моменту, когда будет завершена передача пакета $i + n - 1$. Это может произойти из-за ошибок в обратном канале, больших задержек распространения, длинных (относительно числа шагов назад n) обратных кадров или из-за нескольких причин одновременно.

Будем предполагать (отчасти нереалистично), что повторные передачи возникают только по первой причине и что пакет отвергается на приемнике с вероятностью p независимо от других пакетов.

Рассмотрим случай, когда пакеты поступают в передатчик в соответствии с пуассоновским процессом с интенсивностью λ . Это означает, что интервал времени между началом первой передачи данного пакета (после последней передачи предыдущего пакета) и окончанием последней передачи данного пакета равен $1 + kn$ единицам времени с вероятностью $(1 - p) p^k$ (это соответствует k повторным передачам, которые следуют за последней передачей предыдущего пакета — см. рис. 3.11). Таким образом, очередь на передатчике ведет себя подобно очереди в системе $M/G/1$ и имеет распределение времени обслуживания

$$P \{X = 1 + kn\} = (1 - p) p^k, \quad k = 0, 1, \dots$$

Первые два момента времени обслуживания имеют вид

$$\begin{aligned} \bar{X} &= \sum_{k=0}^{\infty} (1 + kn) (1 - p) p^k = (1 - p) \left(\sum_{k=0}^{\infty} p^k + n \sum_{k=0}^{\infty} k p^k \right), \\ \bar{X}^2 &= \sum_{k=0}^{\infty} (1 + kn)^2 (1 - p) p^k = \\ &= (1 - p) \left(\sum_{k=0}^{\infty} p^k + 2n \sum_{k=0}^{\infty} k p^k + n^2 \sum_{k=0}^{\infty} k^2 p^k \right). \end{aligned}$$

Теперь заметим, что $\sum_{k=0}^{\infty} p^k = 1/(1-p)$, $\sum_{k=0}^{\infty} kp^k = p/(1-p)^2$ и $\sum_{k=0}^{\infty} k^2 p^k = (p + p^2)/(1-p)^3$. (Первая сумма является обычной суммой геометрической прогрессии, а две другие суммы получаются путем двойного дифференцирования первой суммы.) Подставляя эти формулы в приведенные уравнения для \bar{X} и \bar{X}^2 , получаем

$$\bar{X} = 1 + \frac{np}{1-p},$$

$$\bar{X}^2 = 1 + \frac{2np}{1-p} + \frac{n^2(p+p^2)}{(1-p)^2}.$$

По формуле Поллачека—Хинчина получаем выражения для среднего времени пребывания пакета в очереди и в системе (до окончания последней передачи)

$$W = \frac{\lambda \bar{X}^2}{2(1 - \lambda \bar{X})},$$

$$T = \bar{X} + W.$$

3.5.1. Системы M/G/1 с перерывами

Предположим, что в конце каждого периода занятости обслуживающий прибор делает перерыв на некоторый случайный период времени. Таким образом, новые поступления в пустой системе не направляются сразу в обслуживающий прибор, а ожидают окончания перерыва (рис. 3.12). Если система остается пустой к моменту завершения перерыва, сразу же начинается новый перерыв. В сетях передачи данных перерывы соответствуют передаче различного рода управляющих и измерительных пакетов, которые используются, если отсутствует поток данных; другие применения перерывов будут описаны позже.

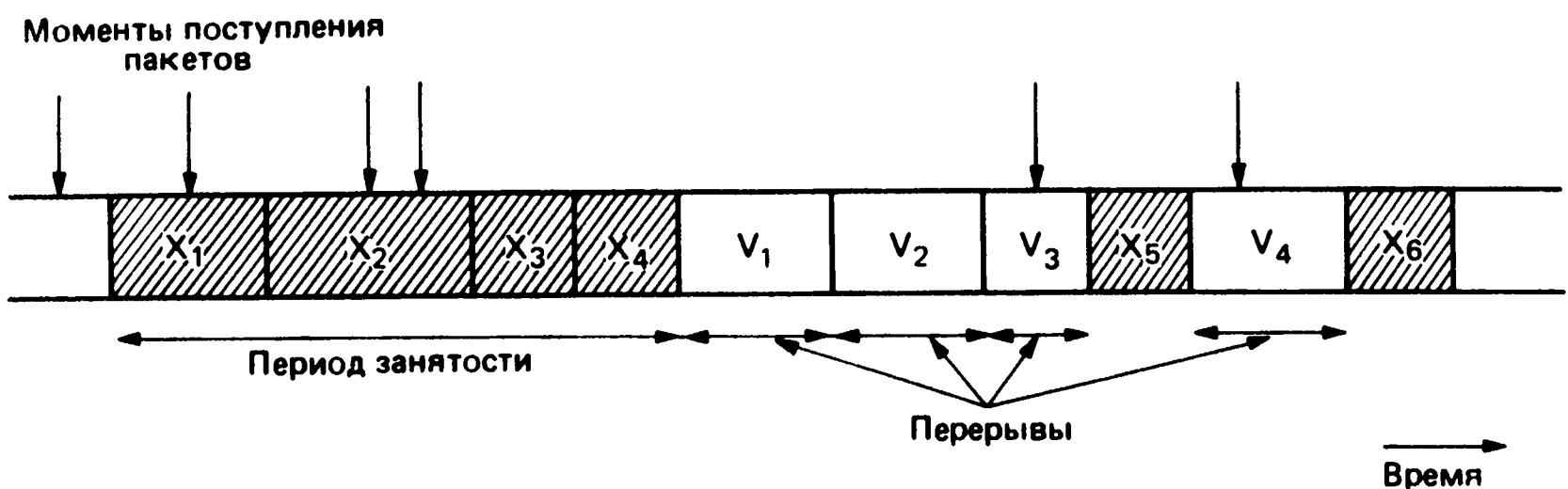


Рис. 3.12. Система M/G/1 с перерывами. В конце периода занятости обслуживающий прибор делает перерыв на время V , которое имеет первый и второй моменты, равные соответственно \bar{V} и \bar{V}^2 . Если система пуста в момент окончания перерыва, обслуживающий прибор делает новый перерыв. Требование, поступившее в пустую систему для того, чтобы попасть в обслуживающий прибор, должно ждать до тех пор, пока не кончится перерыв.

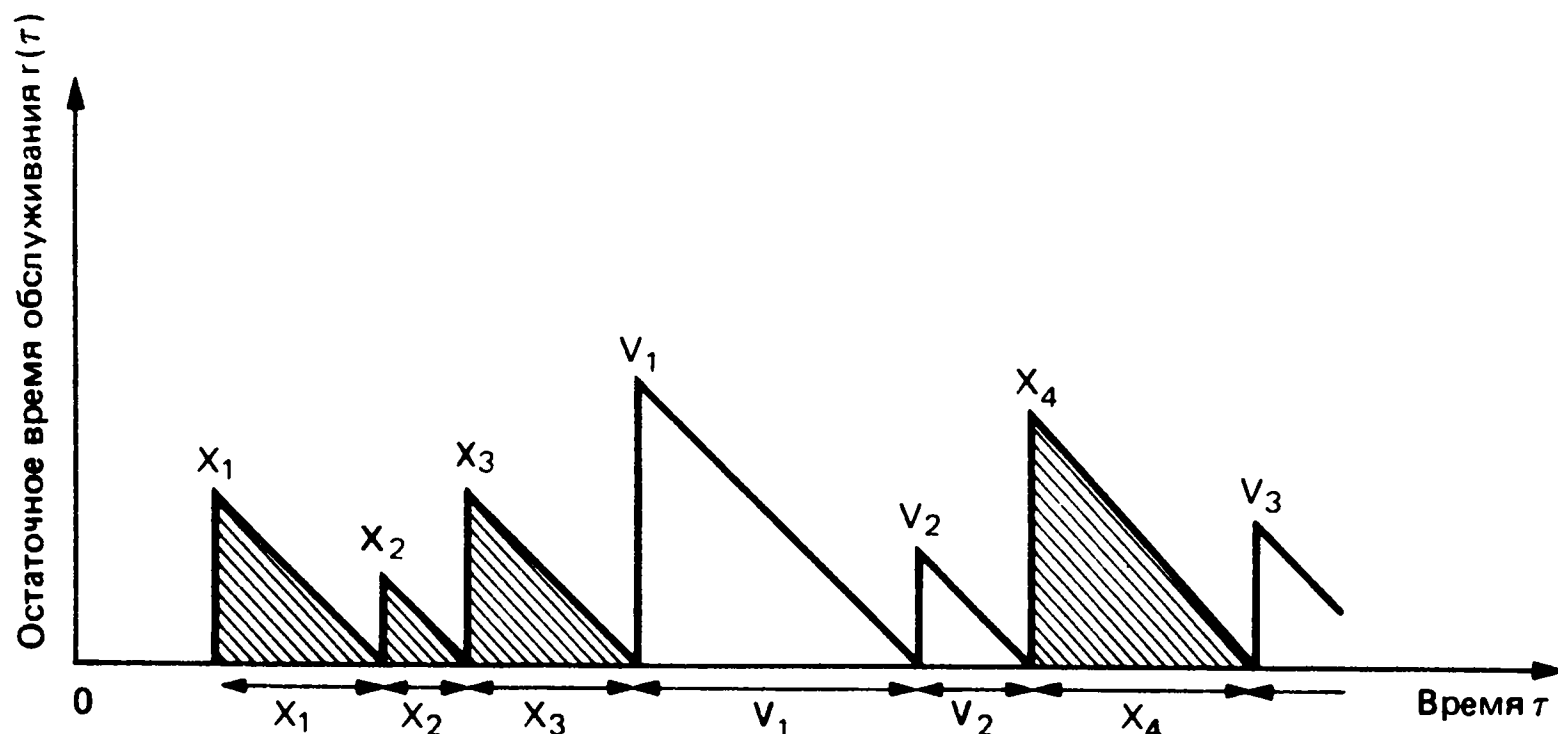


Рис. 3.13. Остаточные длительности обслуживания для системы $M/G/1$ с перерывами. Периоды занятости чередуются с перерывами.

Пусть V_1, V_2, \dots — продолжительности последовательных перерывов, которые делает обслуживающий прибор. Предположим, что V_1, V_2, \dots — независимые одинаково распределенные случайные величины, которые также не зависят от интервалов между моментами поступления требований и длительностей обслуживания требований. Как и ранее, моменты поступления образуют пуассоновский поток, а длительности обслуживания являются независимыми и одинаково распределенными и имеют произвольное распределение. Вновь поступившее в систему требование должно ждать в очереди завершения обслуживания текущего требования или завершения перерыва, а затем должно ждать, пока будут обслужены все требования, стоящие впереди него. Таким образом, равенство (3.48) выполняется (т. е. $W = R/(1 - \rho)$), где R теперь — среднее остаточное время для завершения обслуживания или перерыва в момент, когда поступает i -е требование.

Анализ этой новой системы аналогичен выводу формулы Поллачека — Хинчина за исключением того, что перерывы должны быть включены в график остаточного времени обслуживания $r(\tau)$ (рис. 3.13). Пусть $M(t)$ — число обслуживаний, которые завершились к моменту t , а $L(t)$ — число перерывов, которые закончились к моменту t . Тогда (как и в (3.49)) для любого момента t , когда точно завершается обслуживание или перерыв, имеем

$$\frac{1}{t} \int_0^t r(\tau) d\tau = \frac{1}{t} \sum_{i=1}^{M(t)} \frac{1}{2} X_i^2 + \frac{1}{t} \sum_{i=1}^{L(t)} \frac{1}{2} V_i^2. \quad (3.54)$$

Как и ранее, предположим, что существует стационарное состояние, $M(t)/t$ стремится к λ с ростом t , а первый член правой части

равенства (3.54) стремится к $\lambda \bar{X}^2/2$, как и при выводе формулы Поллачека — Хинчина (ср. с (3.52)). Относительно второго члена заметим, что при $t \rightarrow \infty$ доля времени, которая тратится на обслуживание требований, стремится к ρ , и, таким образом, доля времени на перерывы стремится к $1 - \rho$. Предполагая, что среднее по времени можно заменить на среднее по вероятности, получаем $t(1 - \rho)/L(t) \rightarrow \bar{V}$ при возрастании t , и, таким образом, второй член в равенстве (3.54) стремится к $(1 - \rho) \bar{V}^2/(2\bar{V})$, где \bar{V} и \bar{V}^2 — первый и второй моменты длительности перерыва соответственно. Учитывая это и то, что $W = R/(1 - \rho)$, а также предполагая равенство средних по времени и средних по вероятности для величины R , получаем выражение

$$W = \frac{\lambda \bar{X}^2}{2(1 - \rho)} + \frac{\bar{V}^2}{2\bar{V}} \quad (3.55)$$

для математического ожидания времени, проведенного в очереди в системе $M/G/1$ с перерывами.

Если аккуратно проследить вывод формулы (3.55), то станет понятно, что взаимная независимость длительностей перерывов не является необходимой (хотя средние по времени и средние по вероятности длительностей перерывов должны оставаться равными), и длина перерыва, который начинается в момент t , не обязательно должна быть независимой от уже завершенных интервалов времени обслуживания или моментов поступления. Однако при такой зависимости становится труднее вычислить \bar{V} и \bar{V}^2 , так как эти величины могут теперь зависеть от основополагающих процессов, происходящих в системе $M/G/1$.

Пример 12. Частотное и временное уплотнение с использованием окон

Имеется m потоков пакетов одинаковой длины, моменты поступления которых образуют пуассоновские процессы с интенсивностью λ/m . Если потоки частотно уплотняются по m подканалам имеющегося канала, то время передачи каждого пакета равно m единицам времени. В этом случае каждый подканал может рассматриваться как система массового обслуживания $M/D/1$; учитывая равенство (3.45) при $\rho = \lambda$, $\mu = 1/m$, получаем, что средняя задержка из-за пребывания пакета в очереди равна

$$W_{\text{ЧУ}} = \frac{\lambda m}{2(1 - \lambda)}. \quad (3.56)$$

Рассмотрим систему, аналогичную ЧУ, но с тем отличием, что передачи пакетов начинаются только в моменты $m, 2m, 3m, \dots$, т. е. в начале окна из m единиц времени. Такая система называется системой *синхронного частотного уплотнения* (СЧУ); она может рассматриваться как система $M/D/1$ с перерывами. Если нет пакетов в очереди данного потока в момент начала окна, обслуживающий прибор делает перерыв длиной в одно окно, или m единиц времени. Таким образом, $\bar{V} = m$, $\bar{V}^2 = m^2$ и равенство (3.55) принимает вид

$$W_{\text{СЧУ}} = W_{\text{ЧУ}} + \frac{m}{2}. \quad (3.57)$$

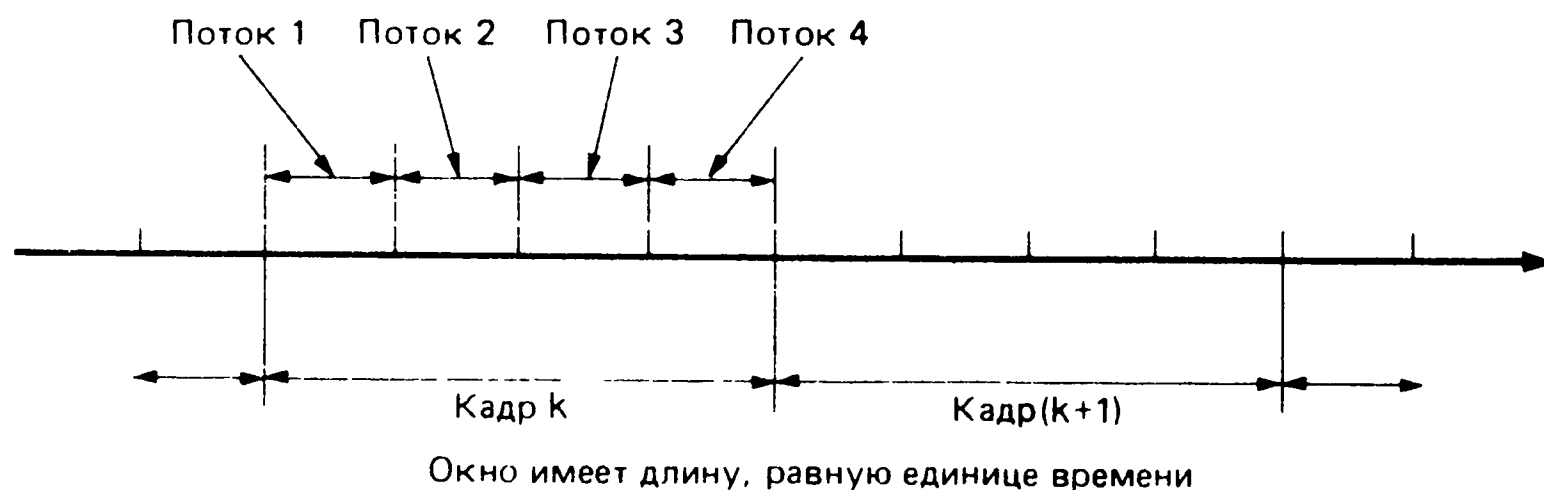


Рис. 3.14. Временное уплотнение (ВУ) $m = 4$ потоков.

Наконец, рассмотрим случай, когда m потоков уплотняются во времени следующим образом: ось времени делится на кадры, которые содержат по m окон, т. е. по одному окну на каждый поток (рис. 3.14). Длина каждого окна равна единице времени и в окне можно передать один пакет. Если сравнить эту систему ВУ с СЧУ, то можно увидеть, что очередь при данном потоке в системе ВУ в точности такая же, как и в системе СЧУ, и

$$W_{ВУ} = W_{СЧУ} = W_{ЧУ} + \frac{m}{2} = \frac{m}{2(1-\lambda)}. \quad (3.58)$$

Если посмотреть теперь на общую задержку системы ВУ, то можно увидеть другую картину, так как время обслуживания равно одной единице времени, а не m единицам, как в системе СЧУ. Суммируя длительности обслуживания и задержки из-за очереди, получаем

$$T_{ЧУ} = m + \frac{\lambda m}{2(1-\lambda)},$$

$$T_{СЧУ} = T_{ЧУ} + \frac{m}{2},$$

$$T_{ВУ} = 1 + \frac{m}{2(1-\lambda)} = T_{ЧУ} - \left(\frac{m}{2} - 1\right). \quad (3.59)$$

Таким образом, с точки зрения средней суммарной задержки требования система ВУ предпочтительнее, чем система ЧУ (в предположении, что $m > 2$). Большее среднее время пребывания требования в очереди в системе ВУ компенсируется более быстрым обслуживанием. Укажем, что в задаче 3.22 предлагается выполнить приведенный анализ в более общем виде.

3.5.2. Резервирование и опрос

Для организации передач нескольких потоков пакетов в системе со статистическим уплотнением требуется введение некоторого расписания для передач. В некоторых случаях такое расписание, естественно, реализуется просто; в других случаях необходимо ввести некоторую систему резервирования или опроса.

Типичным случаем является ситуация, когда имеется канал связи, к которому имеют доступ несколько разнесенных в пространстве пользователей; пусть в данный момент времени только

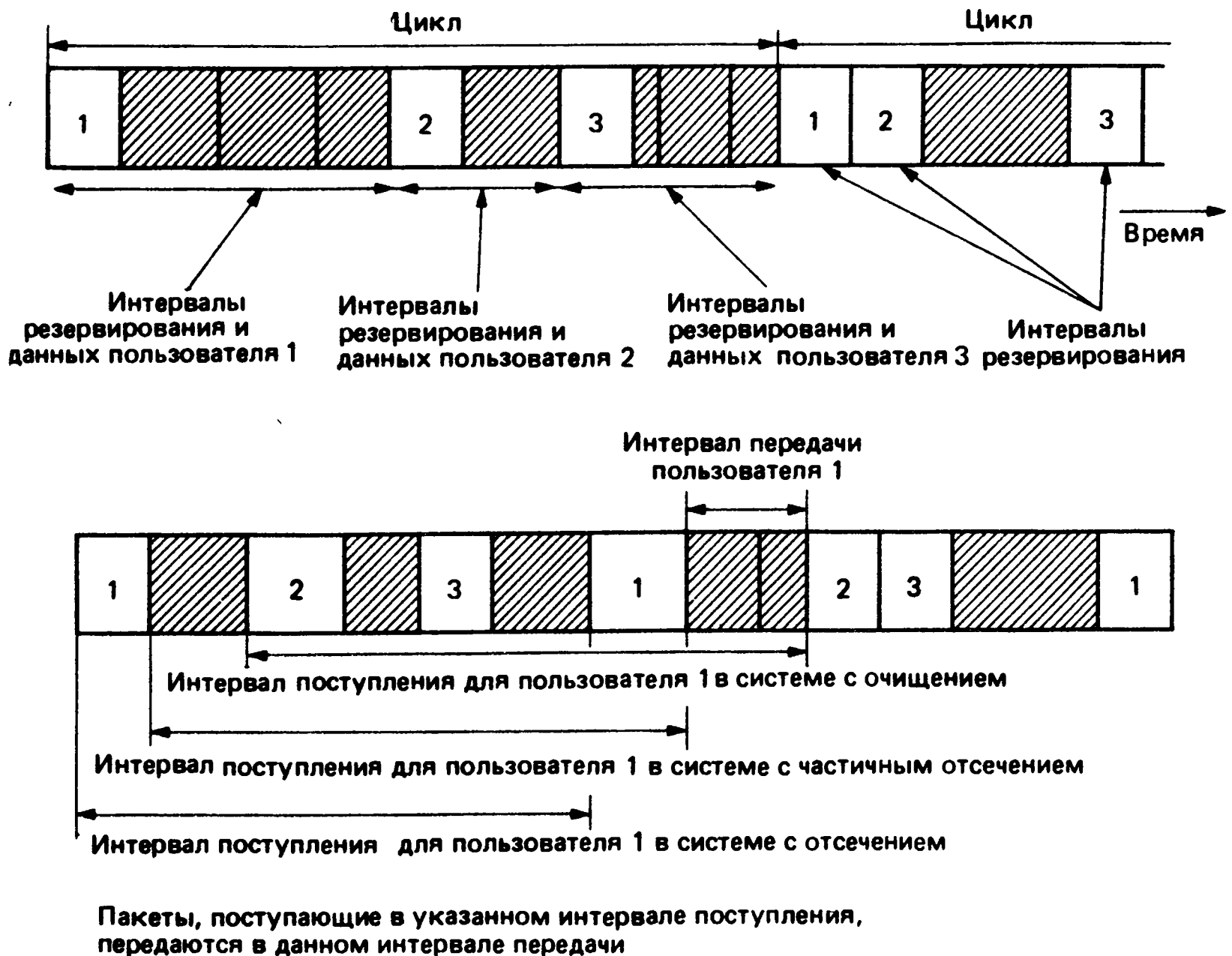


Рис. 3.15. Система с резервированием или опросом для трех пользователей. В системе с очищением пакет пользователя, который поступает в течение интервала резервирования или данных этого пользователя, передается в этом же интервале данных. В системе с частичным отсечением пакет пользователя, поступивший в течение интервала данных этого пользователя, должен ждать полный цикл и будет передан в течение следующего интервала данных этого пользователя. В системе с полным отсечением пакеты, поступившие в течение интервала резервирования данного пользователя, должны также ждать полный цикл. На рисунке для этих трех систем показана связь между интервалом, в котором пакет поступает, и интервалом, в котором пакет передается.

один пользователь может успешно передавать по каналу. (Такой канал называется *каналом множественного доступа*, и он подробно рассматривается в гл. 4.) Время передачи по каналу может быть разделено на часть, в течение которой осуществляется передача пакетов, и часть, которая используется для резервирования или опроса сообщений с целью осуществления координации передач пакетов. Другими словами, ось времени делится на *интервалы для передачи данных* и *интервалы резервирования*, которые используются для составления расписания передач данных. Для однородности описания мы используем термин «резервирование» даже тогда, когда термин «опрос» больше соответствует исследуемой ситуации.

Рассмотрим m потоков (которые также называются пользователями) и предположим, что каждый интервал для передачи данных содержит пакеты одного пользователя. Резервирование этих пакетов происходит непосредственно в предыдущих интервалах. Все пользователи получают доступ в циклическом порядке (рис. 3.15). Имеется несколько вариантов этой системы, отличающихся правилом, по которому решается, какие пакеты каждого пользователя должны передаваться в течение интервала для данных. В системе с *отсечением* это правило состоит в том, что в интервале данных должны передаваться только те пакеты, которые поступили до интервала резервирования, предшествующего этому интервалу данных. В противоположность этому в системе с *очищением* правило состоит в том, что все имеющиеся у пользователя пакеты должны передаваться в течение соответствующего интервала для данных, включая те, которые поступили во время этого интервала или предшествующего ему интервала резервирования. В промежуточном варианте, который называется системой с *частичным отсечением*, в интервале для данных пользователя передаются только те его пакеты, которые поступили до момента начала этого интервала (или до конца соответствующего интервала резервирования). Типичными примерами таких систем резервирования являются наиболее распространенные локальные кольцевые сети связи с передачей маркера. Пользователи соединены кабелем, образующим однонаправленное кольцо. Каждый пользователь передает пакеты, накопленные к текущему моменту, а затем дает возможность передавать соседней станции, и процесс повторяется. (Более подробное описание кольцевой системы с передачей маркера приводится в гл. 4.)

Предположим, что процессы поступления требований всех пользователей являются независимыми пуассоновскими с интенсивностью λ/m и что первый и второй моменты длительностей передач пакетов равны $\bar{X} = 1/\mu$ и \bar{X}^2 соответственно. Коэффициент использования $\rho = \lambda/\mu$. Интервалы между поступлениями и длительности передач, как обычно, считаются независимыми. Хотя мы предполагаем, что все пользователи имеют одинаковые статистики для моментов поступления и длительностей обслуживания, будем считать, что интервалы резервирования различных пользователей могут иметь различные статистики.

Система с одним пользователем

Общая идея исследования систем с резервированием может быть лучше понята на частном случае, когда $m = 1$, поэтому такую систему можно рассматривать как систему, в которой все пользователи участвуют в интервалах резервирования и передачи данных. Пусть V_l — длительность l -го интервала резервиро-

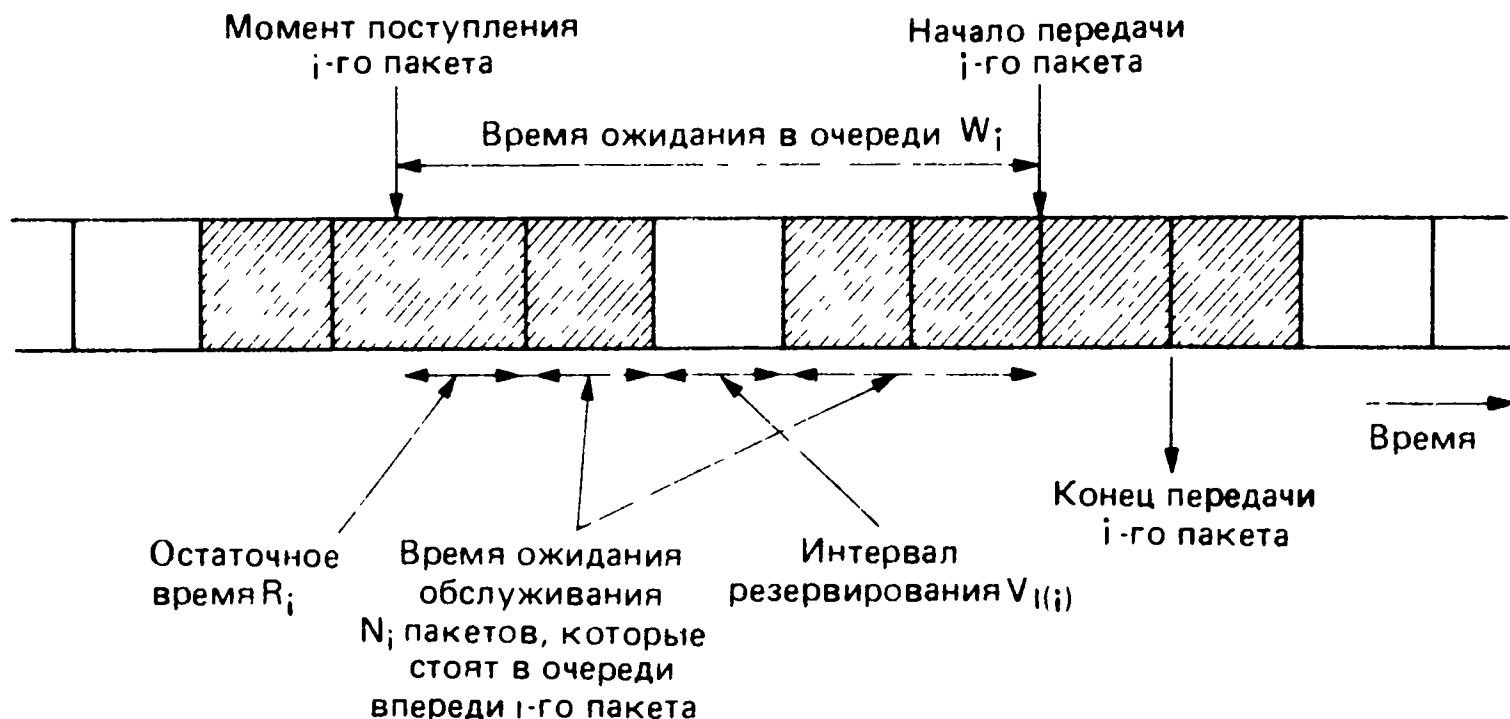


Рис. 3.16. Вычисление среднего времени ожидания в очереди в системе с отсечением и одним пользователем. Математическое ожидание времени ожидания $E\{W_i\}$ для i -го пакета равно

$$E\{W_i\} = E\{R_i\} + E\{N_i\}/\mu + E\{V_{l(i)}\}.$$

ния; предположим, что последовательные интервалы резервирования являются независимыми и одинаково распределенными случайными величинами с первым и вторым моментами, равными соответственно \bar{V} и \bar{V}^2 . Рассмотрим систему с отсечением и предположим, что интервалы резервирования статистически независимы от моментов поступления и длительностей обслуживания. Наконец, для удобства изложения предположим, что пакеты передаются в порядке их поступления. Как и при выводе формулы Поллачека — Хинчина, математические ожидания задержек и длин очередей не зависят от порядка обслуживания, если порядок обслуживания не зависит от требуемого времени обслуживания (т. е. длины пакета).

Рассмотрим i -й поступивший в систему пакет с данными. Этот пакет должен ждать в очереди в течение остаточного времени R_i до окончания текущей передачи пакета или до окончания интервала резервирования. Он должен также ждать, пока будут переданы N_i пакетов, стоящих в данный момент в очереди (пакеты, для которых передача была зарезервирована в последнем интервале резервирования, и более поздние пакеты, которые ожидают резервирования). Наконец, пакет должен находиться в очереди в течение следующего интервала резервирования $V_{l(i)}$, в котором будет сделано резервирование для него (рис. 3.16). Таким образом, математическое ожидание времени, проведенного в очереди, для i -го пакета равно

$$E\{W_i\} = E\{R_i\} + E\{N_i\}/\mu + E\{V_{l(i)}\}. \quad (3.60)$$

Необходимо отметить, что эта система с резервированием аналогична системе $M/G/1$ с перерывами. Единственное отличие состоит в том, что в отсекающей системе с резервированием интервал резервирования начинается, когда закончено обслуживание всех пакетов, заявленных в предыдущем интервале резервирования, а в системе с перерывами перерыв начинается, когда будут обслужены все предыдущие пакеты. (Фактически вариант с очищением этой системы с резервированием эквивалентен системе с перерывами.) Усредненное по времени среднее по пакетам остаточное время для обеих систем (рис. 3.13) равно $\lambda \bar{X}^2/2 + (1 - \rho) \bar{V}^2/2\bar{V}$. Значение предела $\lim_{i \rightarrow \infty} E \{N_i\}/\mu$ равно ρW для обеих систем и, наконец, значение предела $\lim_{i \rightarrow \infty} E \{V_{i(t)}\}$ равно \bar{V} . Таким образом, из формулы (3.60) получаем, что математическое ожидание времени, проведенного в очереди, для системы с резервированием и с одним пользователем равно

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{\bar{V}^2}{2\bar{V}} + \frac{\bar{V}}{1-\rho} \quad \text{(один пользователь, система с отсечением).} \quad (3.61)$$

В типичной ситуации, когда интервал резервирования равен константе A , это означает, что

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{A}{2} \left(\frac{3-\rho}{1-\rho} \right). \quad (3.62)$$

Имеется интересный парадокс, связанный с равенством (3.61). Мы видели, что часть времени $1 - \rho$ используется для резервирования. Так как имеется один интервал резервирования со средней длиной на цикл \bar{V} , можно заключить, что математическое ожидание длины цикла должно быть равно $\bar{V}/(1 - \rho)$ (в задаче 3.23 этот результат показан более аккуратно). Среднее время, проведенное в очереди, в равенстве (3.61) может в произвольно большее число раз превышать среднюю длину цикла, что кажется парадоксальным, так как каждый пакет передается в цикле, который следует после его поступления. Объяснение этого противоречия состоит в том, что большее число пакетов поступает в длинных циклах, чем в коротких, и, таким образом, указанная средняя длина цикла не является средней длиной циклов в момент поступления пакетов; это аналогично ситуации, из-за которой среднее остаточное время обслуживания, используемое при выводе формулы Поллачека — Хинчина, получалось большим, чем можно было бы предположить (см. также задачу 3.15).

Система с многими пользователями

Предположим, что в системе имеется m пользователей, каждый из которых имеет независимый пуассоновский процесс поступления требований с интенсивностью λ/m . Как и ранее, \bar{X} и \bar{X}^2 — первые два момента времени обслуживания пакетов каждого пользователя. Обозначим соответственно через \bar{V}_i и \bar{V}_i^2 первые два момента длительности интервалов резервирования пользователя i . Интервалы обслуживания и интервалы резервирования считаются независимыми в совокупности. Пронумеруем пользователей числами $0, 1, \dots, m-1$ и предположим, что l -й интервал резервирования используется пользователем с номером $l \bmod (m)$, а следующий за ним l -й интервал для передачи данных используется для передачи пакетов, соответствующих этому резервированию.

Рассмотрим i -й поступивший в систему пакет (пакеты нумеруются в порядке их поступления и независимо от номера пользователя). Как и ранее, математическое ожидание задержки этого пакета складывается из трех частей: 1) среднего остаточного времени до завершения передачи пакета или до окончания резервирования, 2) математического ожидания передачи N_i пакетов, которые должны быть переданы перед пакетом i , и 3) математического ожидания длительности интервалов резервирования (рис. 3.17). Таким образом,

$$E\{W_i\} = E\{R_i\} + E\{N_i\}/\mu + E\{Y_i\}, \quad (3.63)$$

где Y_i — длительность всех полных интервалов резервирования, во время которых пакет i ожидает начала своей передачи. Усред-

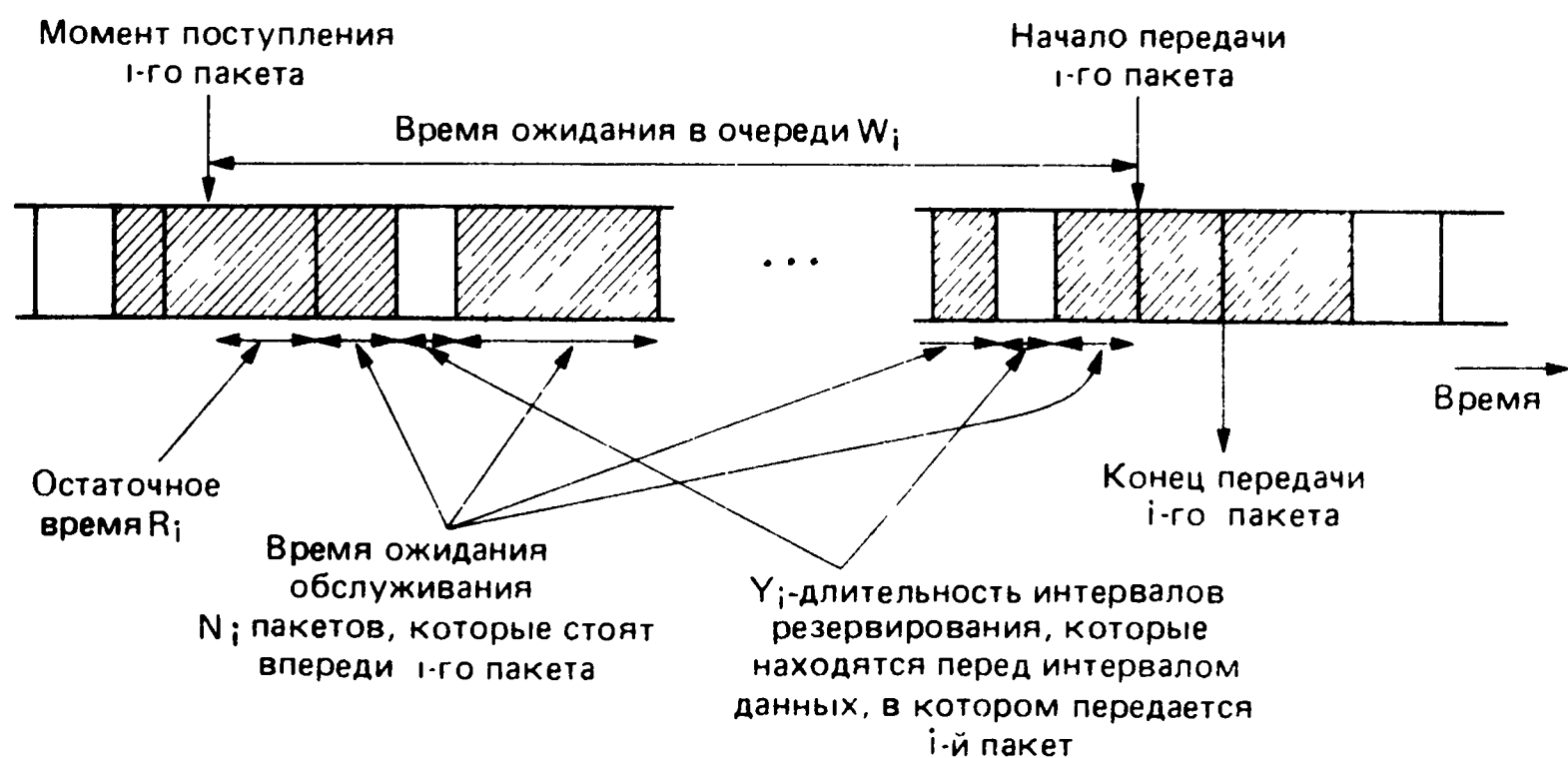


Рис. 3.17. Вычисление среднего времени ожидания в системе с многими пользователями. Математическое ожидание времени ожидания в очереди $E\{W_i\}$ для i -го пакета равно

$$E\{W_i\} = E\{R_i\} + E\{N_i\}/\mu + E\{Y_i\}.$$

ненное по времени среднее по пакетам остаточное время вычисляется таким же образом, как и ранее, оно равно

$$R = \frac{\lambda \bar{X}^2}{2} + \frac{(1 - \rho) \sum_{l=0}^{m-1} \bar{V}_l^2}{2 \sum_{l=0}^{m-1} \bar{V}_l}. \quad (3.64)$$

Число пакетов N_i , в течение передачи которых пакет i должен ждать, не равно числу пакетов, уже находящихся в очереди, но порядок обслуживания пакетов не зависит от времени обслуживания пакета; таким образом, каждый пакет, обслуживаемый перед пакетом i , имеет среднее время передачи $1/\mu$ (это показано в (3.63)); используя теперь формулу Литтла, получаем, что предел $\lim_{i \rightarrow \infty} E \{N_i\}/\mu$ равен, как и ранее, ρW . Полагая $Y = \lim_{i \rightarrow \infty} E \{Y_i\}$, можно записать выражение (3.63) для стационарного случая в следующем виде:

$$W = R + \rho W + Y$$

или, что эквивалентно,

$$W = \frac{R + Y}{1 - \rho}. \quad (3.65)$$

Вычислим сначала Y для системы с очищением. Обозначим $\alpha_{lj} = E \{Y_i | \text{пакет } i \text{ поступает в интервале резервирования или интервале данных } l\text{-го пользователя и принадлежит пользователю с номером } (l + j) \bmod (m)\}$.

Имеем

$$\alpha_{lj} = \begin{cases} 0 & \text{при } j = 0, \\ \bar{V}_{(l+1) \bmod (m)} + \dots + \bar{V}_{(l+j) \bmod (m)} & \text{при } j > 0. \end{cases}$$

Так как пакет i принадлежит любому пользователю с вероятностью $1/m$, имеем

$$E \{Y_i | \text{пакет } i \text{ поступает в интервале резервирования или интервале данных } l\text{-го пользователя}\} =$$

$$= \frac{1}{m} \sum_{j=1}^{m-1} \alpha_{lj} = \sum_{j=1}^{m-1} \frac{m-j}{m} \bar{V}_{(l+j) \bmod (m)}. \quad (3.66)$$

Так как для всех пользователей скорость поступления данных одинакова, интервалы данных всех пользователей в стационарном состоянии имеют равную среднюю длину. Следовательно, в стационарном состоянии пакет поступит в течение интервала данных l -го пользователя с вероятностью ρ/m и поступит в течение интер-

вала резервирования l -го пользователя с вероятностью $(1 - \rho) \bar{V}_l / \left(\sum_{k=0}^{m-1} \bar{V}_k \right)$. Подставляя это в равенство (3.66), получаем следующее выражение для $Y = \lim_{l \rightarrow \infty} E \{Y_l\}$:

$$\begin{aligned}
 Y &= \sum_{l=0}^{m-1} \left(\frac{\rho}{m} + \frac{(1-\rho) \bar{V}_l}{\sum_{k=0}^{m-1} \bar{V}_k} \right) \sum_{j=1}^{m-1} \frac{m-j}{m} \bar{V}_{(l+j) \bmod m} = \\
 &= \frac{\rho}{m} \sum_{j=1}^{m-1} \frac{m-j}{m} \left(\sum_{l=0}^{m-1} \bar{V}_l \right) + \\
 &+ \frac{(1-\rho)}{\sum_{k=0}^{m-1} \bar{V}_k} \sum_{l=0}^{m-1} \sum_{j=1}^{m-1} \frac{m-j}{m} \bar{V}_l \bar{V}_{(l+j) \bmod m}. \quad (3.67)
 \end{aligned}$$

Последняя сумма может быть записана следующим образом:

$$\sum_{l=0}^{m-1} \sum_{j=1}^{m-1} \frac{m-j}{m} \bar{V}_l \bar{V}_{(l+j) \bmod m} = \frac{1}{2} \left[\left(\sum_{l=0}^{m-1} \bar{V}_l \right)^2 - \sum_{l=0}^{m-1} \bar{V}_l^2 \right].$$

(Чтобы понять это, заметим, что правая часть этого равенства равна сумме всех возможных произведений $\bar{V}_l \bar{V}_{l'}$ при $l \neq l'$. Левая часть равна сумме всех возможных членов $(j/m) \bar{V}_l \bar{V}_{l'}$ и $[(m-j)/m] \bar{V}_l \bar{V}_{l'}$, где $j = |l - l'|$, а $l \neq l'$.) Используя эти выкладки и обозначая через

$$\bar{V} = \frac{1}{m} \sum_{l=0}^{m-1} \bar{V}_l$$

интервал резервирования, усредненный по всем пользователям, из (3.67) получаем

$$\begin{aligned}
 Y &= \frac{\rho \bar{V} (m-1)}{2} + \frac{(1-\rho) m \bar{V}}{2} - \frac{(1-\rho) \sum_{l=0}^{m-1} \bar{V}_l^2}{2m \bar{V}} = \\
 &= \frac{(m-\rho) \bar{V}}{2} - \frac{(1-\rho) \sum_{l=0}^{m-1} \bar{V}_l^2}{2m \bar{V}}. \quad (3.68)
 \end{aligned}$$

Из равенств (3.64), (3.65) и (3.68) находим

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{(m-\rho)\bar{V}}{2(1-\rho)} + \frac{\sum_{l=0}^{m-1} (\bar{V}_l^2 - \bar{V}_l^2)}{2m\bar{V}}.$$

Обозначая через

$$\sigma_V^2 = \frac{\sum_{l=0}^{m-1} (\bar{V}_l^2 - \bar{V}_l^2)}{m}$$

дисперсию интервалов резервирования, усредненную по всем пользователям, окончательно получаем

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{(m-\rho)\bar{V}}{2(1-\rho)} + \frac{\sigma_V^2}{2\bar{V}} \text{ (система с очищением)}. \quad (3.69)$$

Система с частичным отсечением совпадает с системой с очищением за исключением того, что, если пакет пользователя поступает в течение интервала данных этого пользователя (вероятность такого события в стационарном состоянии равна ρ/m), он дополнительно задерживается на время $m\bar{V}$, равное средней сумме интервалов резервирования в цикле. Таким образом, Y в предыдущих вычислениях увеличивается на $\rho\bar{V}$, поэтому получаем

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{(m+\rho)\bar{V}}{2(1-\rho)} + \frac{\sigma_V^2}{2\bar{V}} \text{ (система с частичным отсечением)}. \quad (3.70)$$

Рассмотрим, наконец, систему с полным отсечением. Она подобна системе с частичным отсечением за исключением того, что, если пакет пользователя поступает в течение интервала резервирования этого пользователя (вероятность такого события в стационарном состоянии равна $(1-\rho)/m$), он дополнительно задерживается на время $m\bar{V}$. Это увеличивает Y дополнительно на $(1-\rho)\bar{V}$ и приводит к соотношению

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{(m+2-\rho)\bar{V}}{2(1-\rho)} + \frac{\sigma_V^2}{2\bar{V}} \text{ (система с отсечением)}. \quad (3.71)$$

Для сравнения этих результатов с результатами для системы с одним пользователем рассмотрим случай, когда интервал резервирования имеет постоянную длительность, равную A/m . Таким образом, A равно избыточности или времени резервирования для полного цикла резервирования всех пользователей и обычно

является параметром, используемым для сравнения с параметром A в (3.62). Тогда имеем

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{A}{2} \left(\frac{1-\rho/m}{1-\rho} \right) \text{ (система с очищением), (3.72)}$$

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{A}{2} \left(\frac{1+\rho/m}{1-\rho} \right) \text{ (система с частичным отсечением), (3.73)}$$

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{A}{2} \left(\frac{1+(2-\rho)/m}{1-\rho} \right) \text{ (система с отсечением). (3.74)}$$

Можно увидеть, что в случае многих пользователей задержка немного уменьшается; существенно, что во всех случаях пакеты задерживаются приблизительно на одно и то же время до момента резервирования, а задержка после резервирования в случае многих пользователей очень мала.

Система с ограниченным обслуживанием

Рассмотрим теперь другой вариант системы с многими пользователями, в котором в каждом интервале данных пользователя передается *только первый* из очереди пакет пользователя (а не все его ждущие пакеты), конечно, если очередь не пустая. Рассмотрим системы с отсечением и частичным отсечением, так как система с очищением сейчас не имеет смысла. Как и ранее, имеем

$$E\{W_i\} = E\{R_i\} + E\{N_i\}/\mu + E\{Y_i\}$$

и устремляя $i \Rightarrow \infty$, получаем

$$W = R + \rho W + Y. \quad (3.75)$$

Здесь, как и ранее, R дает равенство (3.64). Для получения новой формулы для Y для системы с частичным отсечением будем рассуждать следующим образом. Пакет, поступивший в течение интервала резервирования или интервала данных l -го пользователя, будет принадлежать любому пользователю с равной вероятностью $1/m$. Следовательно, в стационарном состоянии математическое ожидание числа пакетов, ожидающих в индивидуальной очереди пользователя, которому принадлежит поступивший пакет, усредненное по всем пользователям, равно $\lim_{i \rightarrow \infty} E\{N_i\}/m = \lambda W/m$. Каждый из этих пакетов требует дополнительный цикл резервирования длительностью $m\bar{V}$, поэтому Y увеличивается на величину $\lambda W\bar{V}$. Используя это в равенстве (3.75), получаем

$$W = \frac{R + \tilde{Y}}{1 - \rho - \lambda \bar{V}},$$

где \tilde{Y} — значение Y , полученное ранее для системы с частичным отсечением без ограничения, что за интервал данных должен передаваться только один пакет. Это эквивалентно тому, что, как видно из (3.65), ограничение, которое состоит в том, что за интервал данных должен передаваться только один пакет, приводит к увеличению среднего времени ожидания для системы с частичным отсечением в

$$\frac{1 - \rho}{1 - \rho - \lambda \bar{V}} \text{ раз.}$$

Поэтому из равенства (3.70) получаем

$$W = \frac{\lambda \bar{X}^2}{2(1 - \rho - \lambda \bar{V})} + \frac{(m + \rho) \bar{V}}{2(1 - \rho - \lambda \bar{V})} + \frac{\sigma_V^2 (1 - \rho)}{2\bar{V}(1 - \rho - \lambda \bar{V})} \text{ (система с ограниченным обслуживанием и частичным отсечением).} \quad (3.76)$$

Рассмотрим теперь систему с отсечением. Величина Y_i совпадает с соответствующей величиной в системе с частичным отсечением за исключением дополнительного цикла интервалов резервирования средней длины $m\bar{V}$, необходимого в том случае, когда пакет i поступает в течение своего собственного интервала резервирования и последующий интервал данных оказывается пустым. Легко установить (задача 3.24), что последнее событие происходит со стационарной вероятностью $(1 - \rho - \lambda \bar{V})/m$. Следовательно, для системы с отсечением величина Y равна соответствующей величине для системы с частичным отсечением плюс $(1 - \rho - \lambda \bar{V}) \bar{V}$. В результате этого к величине W для системы с частичным отсечением добавляется величина \bar{V} и среднее время ожидания теперь становится равным

$$W = \frac{\lambda \bar{X}^2}{2(1 - \rho - \lambda \bar{V})} + \frac{(m + 2 - \rho - 2\lambda \bar{V}) \bar{V}}{2(1 - \rho - \lambda \bar{V})} + \frac{\sigma_V^2 (1 - \rho)}{2\bar{V}(1 - \rho - \lambda \bar{V})}$$

(система с ограниченным обслуживанием и отсечением). (3.77)

Заметим, что, для того чтобы W было ограничено, недостаточно, чтобы $\rho = \lambda/\mu < 1$; требуется, чтобы $\rho + \lambda \bar{V} < 1$ или, что эквивалентно,

$$\lambda \left(\frac{1}{\mu} + \bar{V} \right) < 1.$$

Это обусловлено тем, что каждый пакет требует отдельного интервала резервирования средней длины \bar{V} , в результате чего среднее время передачи возрастает от $1/\mu$ до $1/\mu + \bar{V}$.

Наконец, рассмотрим случай очень большого числа пользова-

телей m и очень малой средней длительности интервала резервирования \bar{V} . Исследование формулы для среднего времени ожидания W для каждой системы, рассмотренной до сих пор, показывает, что при $m \rightarrow \infty$, $\bar{V} \rightarrow 0$, $\sigma_V^2/\bar{V} \rightarrow 0$ и $m\bar{V} \rightarrow A$, где A — константа, имеем

$$W \rightarrow \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{A}{2(1-\rho)}.$$

Можно показать (задача 3.23), что средняя длина цикла (m последовательных интервалов резервирования и данных) равна $A/(1-\rho)$. Таким образом, W стремится к сумме среднего времени ожидания в системе $M/G/1$ и половины средней длины цикла.

3.5.3. Системы массового обслуживания с приоритетами

Рассмотрим систему $M/G/1$ с тем отличием, что поступающие требования делятся на n классов различного приоритета, причем 1-й класс имеет наивысший приоритет, 2-й класс — второй по величине приоритет и т. д. Скорость поступления и первые два момента времени обслуживания для класса k обозначаются соответственно через λ_k , $\bar{X}_k = 1/\mu_k$ и \bar{X}_k^2 . Предполагается, что процессы поступления для всех классов независимые, пуассоновские и не зависят от длительностей обслуживания.

Приоритет без прерывания обслуживания

Сначала рассмотрим правило приоритета без прерывания обслуживания, согласно которому требованию, находящемуся на обслуживании, позволяется завершить обслуживание без прерывания, даже если во время его обслуживания поступает требование более высокого приоритета. Отдельные очереди формируются для каждого приоритета. В момент, когда освобождается обслуживающий прибор, первое требование из непустой очереди наивысшего приоритета поступает на обслуживание. Это правило приоритета является одним из самых подходящих для описания систем пакетной передачи.

Мы получим формулы для средней задержки требований каждого приоритета, которые аналогичны формуле Поллачека — Хинчина и допускают аналогичное доказательство. Введем обозначения

- N_Q^k — среднее число требований в очереди k -го приоритета;
- W_k — среднее время ожидания в очереди для требования k -го приоритета;
- $\rho_k = \lambda_k/\mu_k$ — коэффициент использования системы для k -го приоритета;
- R — среднее остаточное время обслуживания.

Предположим, что общий коэффициент использования системы меньше единицы, т. е.

$$\rho_1 + \rho_2 + \dots + \rho_n < 1.$$

Если это предположение не выполняется, будет некоторый класс приоритета k , такой, что средняя задержка требований приоритета k и более низких будет бесконечной, тогда как средняя задержка требований с более высокими приоритетами будет конечной. В задаче 3.16 эту ситуацию требуется рассмотреть более подробно.

Как и в приведенном ранее выводе формулы Поллачека — Хинчина для наивысшего приоритета, имеем

$$W_1 = R + \frac{1}{\mu_1} N_Q^1.$$

Исключая N_Q^1 отсюда с помощью теоремы Литтла, получаем

$$N_Q^1 = \lambda_1 W_1,$$

а также

$$W_1 = R + \rho_1 W_1$$

и, наконец,

$$W_1 = \frac{R}{1 - \rho_1}. \quad (3.78)$$

Для второго приоритета имеем аналогичное выражение для времени ожидания в очереди W_2 за исключением того, что необходимо учесть дополнительную задержку, возникающую из-за требований более высокого приоритета, которые поступают в то время, когда требование ожидает в очереди. Это учитывает последний член в формуле

$$W_2 = R + \frac{1}{\mu_1} N_Q^1 + \frac{1}{\mu_2} N_Q^2 + \frac{1}{\mu_1} \lambda_1 W_2.$$

Согласно теореме Литтла ($N_Q^k = \lambda_k W_k$), получаем

$$W_2 = R + \rho_1 W_1 + \rho_2 W_2 + \rho_1 W_2,$$

откуда следует, что

$$W_2 = \frac{R + \rho_1 W_1}{1 - \rho_1 - \rho_2}.$$

Используя ранее полученное выражение $W_1 = R/(1 - \rho_1)$, окончательно имеем

$$W_2 = \frac{R}{(1 - \rho_1)(1 - \rho_1 - \rho_2)}.$$

Для всех приоритетов $k > 1$ эта формула получается аналогично и среднее время ожидания в очереди

$$W_k = \frac{R}{(1 - \rho_1 - \dots - \rho_{k-1})(1 - \rho_1 - \dots - \rho_k)}. \quad (3.79)$$

Выражение для средней задержки требования k -го приоритета имеет вид

$$T_k = \frac{1}{\mu_k} + W_k. \quad (3.80)$$

Теперь нужно найти среднее остаточное время обслуживания R . Как и ранее, при выводе формулы Поллачека — Хинчина (ср. с рис. 3.10) имеем

$$R = \frac{1}{2} \left(\sum_{i=1}^n \lambda_i \right) \overline{X^2}, \quad (3.81)$$

где $\overline{X^2}$ обозначает второй момент времени обслуживания, усредненный по всем классам приоритета:

$$\overline{X^2} = \frac{\lambda_1}{\sum_{i=1}^n \lambda_i} \overline{X_1^2} + \dots + \frac{\lambda_n}{\sum_{i=1}^n \lambda_i} \overline{X_n^2}.$$

Подставляя это в равенство (3.81), находим

$$R = \frac{1}{2} \sum_{i=1}^n \lambda_i \overline{X_i^2}. \quad (3.82)$$

Среднее время ожидания в очереди и средняя задержка требования для каждого приоритета получаются из равенств (3.79), (3.80) и (3.82)

$$W_k = \frac{\sum_{i=1}^n \lambda_i \overline{X_i^2}}{2(1 - \rho_1 - \dots - \rho_{k-1})(1 - \rho_1 - \dots - \rho_k)}, \quad (3.83)$$

$$T_k = \frac{1}{\mu_k} + W_k.$$

Заметим, что имеется возможность оказывать влияние на среднюю задержку требования путем соответствующего выбора приоритета. Как правило, средняя задержка уменьшается, если требования с малым временем обслуживания получают более высокий приоритет. (Примером из повседневной жизни является работа магазина самообслуживания, в котором имеется специальная касса для покупателей с небольшим числом покупок. Аналогичную ситуацию можно наблюдать в очередях на копировальную машину, где часто получают приоритет по отношению к дру-

им те люди, которым необходимо сделать только небольшое число копий.) Аналитическое обоснование этого можно получить путем рассмотрения системы без прерывания с двумя классами требований A и B и соответствующими скоростями поступления и скоростями обслуживания λ_A, μ_A и λ_B, μ_B . Непосредственное вычисление по приведенным выше формулам показывает, что если $\mu_A > \mu_B$, то средняя задержка требования (усредненная по обоим классам) равна

$$T = \frac{\lambda_A T_A + \lambda_B T_B}{\lambda_A + \lambda_B},$$

в случае, когда A имеет приоритет над B меньше, чем в случае, когда B имеет приоритет над A . Аналогичные результаты требуется получить в задаче 3.19.

Приведенный анализ нельзя легко распространить на случай системы с несколькими обслуживающими приборами в первую очередь потому, что нет простой формулы для среднего остаточного времени. Однако если длительности обслуживания для всех приоритетов имеют одинаковое экспоненциальное распределение, то имеется удобное выражение для R . В этом случае формула (3.79) дает замкнутое выражение для средних длительностей ожидания W_k (см. задачу 3.17).

Приоритеты с прерыванием и дообслуживанием

Одной из характерных особенностей приоритета без прерывания является то, что средняя задержка данного класса приоритета зависит от скорости поступления в классах более низкого приоритета. Это ясно из равенства (3.83) и следует из того, что требования более высоких приоритетов должны ждать завершения обслуживания требований более низких приоритетов. Эта зависимость отсутствует в дисциплине *приоритета с прерыванием и дообслуживанием*, в которой обслуживание требования прерывается, когда поступает требование более высокого приоритета, и возобновляется от момента прерывания сразу же, после того как обслужены все требования более высокого приоритета.

При вычислении T_k — среднего времени пребывания в системе требований k -го приоритета, будем иметь в виду, что присутствие требований с приоритетами от $k+1$ до n не влияет на это вычисление. Следовательно, можно рассматривать каждый класс приоритета в системе как самый низкий.

Среднее время пребывания в системе T_k состоит из трех частей. Первая часть — среднее время обслуживания требования, равное $1/\mu_k$. Вторая часть — среднее время, необходимое для обслуживания требований с приоритетом от 1 до k , которые уже находятся в системе в момент поступления данного требования

с приоритетом k , т. е. средняя продолжительность неоконченной работы, соответствующая приоритетам от 1 до k . Этот член равен среднему времени ожидания в соответствующей обычной системе $M/G/1$ (без приоритетов), в которой требования с приоритетами от $k + 1$ до n не учитываются, т. е. (ср. с равенством (3.48))

$$\frac{R_k}{1 - \rho_1 - \dots - \rho_k},$$

где R_k — среднее остаточное время

$$R_k = \frac{\sum_{i=1}^k \lambda_i X_i^2}{2}. \quad (3.84)$$

Это справедливо, так как в любой момент времени неоконченная работа (сумма оставшихся длительностей обслуживания всех требований в системе) в системе типа $M/G/1$ не зависит от дисциплины приоритета. Это верно для всех систем массового обслуживания, которые являются консервативными в том смысле, что обслуживающий прибор всегда занят, когда система не пуста, и требования покидают систему только после требуемого обслуживания. Третья часть в выражении для T_k равна среднему времени ожидания для требований с приоритетами от 1 до $k - 1$, которые поступили во время пребывания в системе данного требования с приоритетом k . Этот член равен

$$\sum_{i=1}^{k-1} \frac{1}{\mu_i} \lambda_i T_k = \sum_{i=1}^{k-1} \rho_i T_k$$

при $k > 1$ и 0 при $k = 1$. Используя эти части, получим равенство

$$T_k = \frac{1}{\mu_k} + \frac{R_k}{1 - \rho_1 - \dots - \rho_k} + \left(\sum_{i=1}^{k-1} \rho_i \right) T_k. \quad (3.85)$$

Окончательный результат

$$T_1 = \frac{(1/\mu_1)(1 - \rho_1) + R_1}{1 - \rho_1} \quad (3.86)$$

при $k = 1$ и

$$T_k = \frac{(1/\mu_k)(1 - \rho_1 - \dots - \rho_k) + R_k}{(1 - \rho_1 - \dots - \rho_{k-1})(1 - \rho_1 - \dots - \rho_k)} \quad (3.87)$$

при $k > 1$, где R_k дается формулой (3.84). Так же как и для системы без прерываний обслуживания, эту формулу не просто обобщить на случай нескольких обслуживающих приборов за исключением случая, когда длительности обслуживания для всех приоритетов имеют одинаковое экспоненциальное распределение (см. задачу 3.17).

3.6. Сети линий связи

В сетях передачи данных имеется много очередей на передачу, которые взаимодействуют друг с другом в том смысле, что поток, уходящий из одной очереди, поступает в одну или несколько других очередей, возможно после слияния с частями других потоков из каких-либо других очередей. С аналитической точки зрения это неблагоприятно влияет и усложняет характер процессов поступления на очереди, расположенные по течению потока. Трудность состоит в том, что когда пакеты передаются за пределами первой по отношению к точке их входа в сеть очереди, то интервалы между моментами поступления пакетов становятся сильно коррелированными с длинами пакетов. В результате невозможно выполнить точный и эффективный анализ, сравнимый с анализом, проведенным для таких систем массового обслуживания, как $M/M/1$, $M/G/1$ и др.

Для иллюстрации явлений, которые усложняют анализ, рассмотрим две тандемные линии передачи равной пропускной способности, показанные на рис. 3.18. Предположим, что в первую очередь поступает пуассоновский поток с интенсивностью λ пакетов в секунду и что все пакеты имеют *равную* длину. Следовательно, первая очередь соответствует системе $M/D/1$ и средняя задержка пакета находится по формуле Поллачека — Хинчина. Однако для второй очереди интервалы между моментами поступления должны быть больше или равны $1/\mu$ (времени передачи пакета). Более того, так как длительности передачи пакетов равны для обеих очередей, передача любого пакета, поступившего во вторую очередь, завершится раньше или в момент поступления следующего пакета. Таким образом, пакеты не стоят *во второй очереди*. Следовательно, математическая модель, основанная на предположении пуассоновского характера поступления, совершенно не подходит для вычисления задержки во второй очереди.

Рассмотрим далее случай двух тандемных линий, когда длины пакетов распределены экспоненциально и не зависят друг от друга и от интервалов между моментами поступления в первую очередь. Тогда первая очередь соответствует системе $M/M/1$. Однако вторая очередь *не может* быть рассмотрена как система $M/M/1$. Дело опять в том, что *интервалы между моментами поступления во вторую очередь сильно коррелируют с длинами пакетов*. Для того чтобы понять это, рассмотрим период заня-

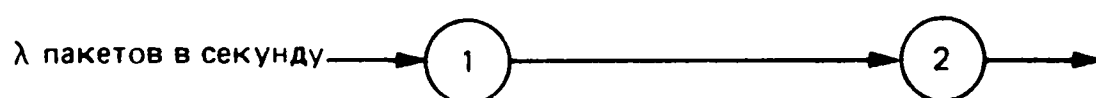


Рис. 3.18. Две тандемные линии передачи с равными пропускными способностями. Если все пакеты имеют одинаковую длину, то пакеты не стоят во второй очереди.

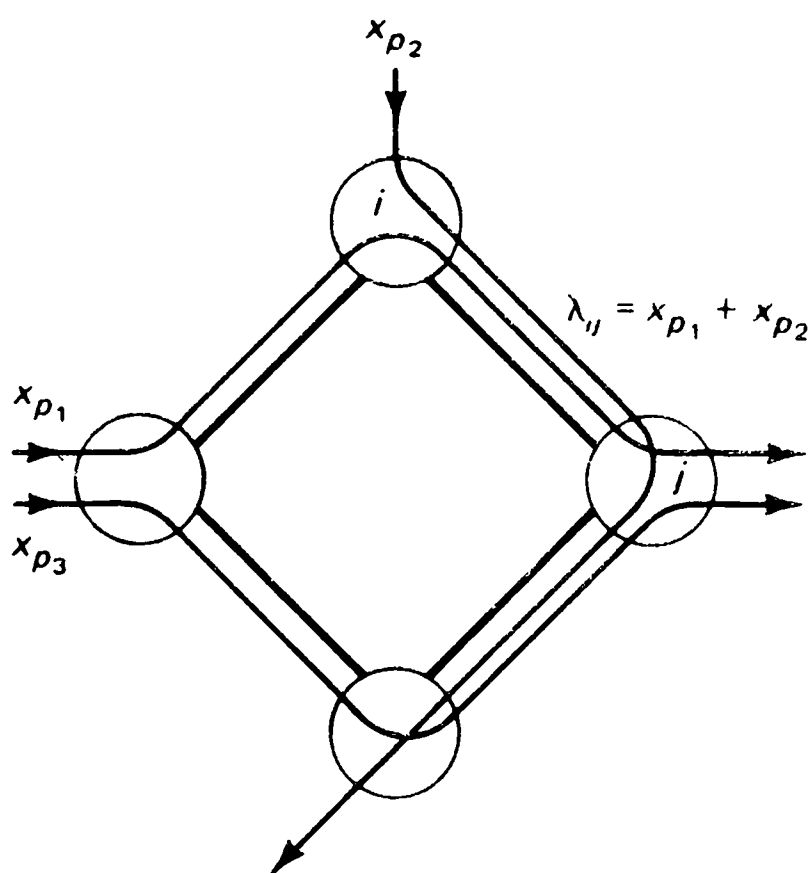


Рис. 3.19. Сеть линий связи. Общая интенсивность поступления $\lambda_{i,j}$ в линию (i, j) равна сумме интенсивностей x_p всех пакетных потоков p , проходящих через эту линию.

тости в первой очереди, когда несколько пакетов передаются один за другим. Интервал между моментами поступления во вторую очередь двух таких пакетов равен времени передачи второго пакета. В результате длинные пакеты будут обычно ждать меньше времени во второй очереди, чем короткие пакеты, так как их передача по первой линии длится дольше, вследствие чего у второй очереди будет больше времени для того, чтобы стать пустой. Как аналогию потока пакетов можно рассмотреть медленно движущийся грузовик и несколько более быстрых машин, следующих по узкой улице. Грузовик будет обычно иметь перед собой пустое пространство, а за ним

вплотную будут следовать более быстро движущиеся машины. Исследование сетей с очередями, в которых учитывается зависимость длительностей интервалов между моментами поступления и длительностей обслуживания, довольно сложное; даже простая задача для тандемных очередей (рис. 3.18) при пуассоновских моментах поступления и экспоненциально распределенных длительностях обслуживания аналитически не решена. В реальных ситуациях, когда длины пакетов и интервалы между моментами поступления коррелированы, численное моделирование показывает, что при больших нагрузках средняя задержка пакета меньше, чем в идеализированной ситуации, когда нет такой корреляции. При малых нагрузках справедливо обратное. Неизвестно, в какой форме этот результат может быть распространен на более общие сети и возможно ли это. Рассмотрим теперь сеть линий связи, показанную на рис. 3.19. Предположим, что имеется несколько потоков пакетов, каждый из которых следует по пути p , который состоит из последовательности линий сети. Пусть x_p (в пакетах в секунду) — скорость поступления потока пакетов, относящегося к пути p . Тогда общая скорость поступления в линию (i, j) равна

$$\lambda_{ij} = \sum_{\substack{\text{все пути } p, \\ \text{проходящие} \\ \text{через ли-} \\ \text{нию } (i, j)}} x_p. \quad (3.88)$$

Из частного случая двух тандемных очередей видно, что если даже потоки пакетов в точках входа в сеть являются пуассоновскими с независимыми длинами пакетов, то это свойство потоков теряется после передачи по первой линии. Для того чтобы разрешить эту трудность, Клейнрок [141] предположил, что при объединении нескольких потоков пакетов в линии передачи сохраняется независимость между интервалами поступления и длинами пакетов. Было заключено, что при этом для каждой линии связи часто можно приближенно принять модель системы $M/M/1$ независимо от взаимодействия потока на этой линии с потоками на других линиях (см. также обсуждение, предшествующее теореме Джексона в разд. 3.8). Это известно как *аппроксимация Клейнрока* (или гипотеза о независимости); она дает довольно хорошее приближение при умеренных и больших нагрузках для сильно связанных сетей с пуассоновскими моментами поступления потоков во входные точки сети, с длинами пакетов, которые распределены почти экспоненциально. На основе модели $M/M/1$ получаем, что среднее число пакетов в очереди или на обслуживании в линии (i, j) равно

$$N_{ij} = \frac{\lambda_{ij}}{\mu_{ij} - \lambda_{ij}}, \quad (3.89)$$

где $1/\mu_{ij}$ — среднее время передачи пакета по линии. Среднее число пакетов, просуммированное по всем очередям, равно

$$N = \sum_{(i,j)} \frac{\lambda_{ij}}{\mu_{ij} - \lambda_{ij}}, \quad (3.90)$$

таким образом, согласно теореме Литтла, средняя задержка пакета (без учета задержек обработки и распространения) равна

$$T = \frac{1}{\gamma} \sum_{(i,j)} \frac{\lambda_{ij}}{\mu_{ij} - \lambda_{ij}}, \quad (3.91)$$

где $\gamma = \sum_p x_p$ — общая скорость поступления в систему. Если средней задержкой распространения и обработки в линии (i, j) , обозначаемой через α_{ij} , нельзя пренебречь, эта формула принимает вид

$$T = \frac{1}{\gamma} \sum_{(i,j)} \left(\frac{\lambda_{ij}}{\mu_{ij} - \lambda_{ij}} + \lambda_{ij} d_{ij} \right). \quad (3.92)$$

Окончательно средняя задержка пакета в потоке, идущем по пути p , равна

$$T_p = \sum_{\substack{\text{по всем } (i,j) \\ \text{на пути } p}} \left(\frac{\lambda_{ij}}{\mu_{ij}(\mu_{ij} - \lambda_{ij})} + \frac{1}{\mu_{ij}} + d_{ij} \right), \quad (3.93)$$

где три члена под знаком суммы относятся соответственно к среднему времени ожидания в очереди, среднему времени передачи и к задержке распространения и обработки.

Для многих сетей предположение об экспоненциальном распределении длины пакетов неприменимо. При заданной функции распределения длины пакетов можно сохранить приближение о независимости очередей, а формулу (3.89) заменить на формулу Поллачека — Хинчина для среднего числа пакетов в системе. Равенства (3.90)—(3.93) при этом изменяются очевидным образом.

Следует отметить, что главное приближение, которое делается при получении выражения (3.90), состоит в том, что не учитывается корреляция длин пакетов и интервалов между моментами поступления пакетов в очередях сети. Если по каким-либо причинам эта корреляция отсутствует, например если пакету в момент, когда он покидает линию передачи, присваивается новая длина, которая является случайной величиной с экспоненциальным распределением, то среднее число пакетов в системе на самом деле определяется по формуле

$$N = \sum_{(i,j)} \frac{\lambda_{ij}}{\mu_{ij} - \lambda_{ij}}.$$

Этот факт (который не является очевидным) есть следствие теоремы Джексона, которая рассматривается в разд. 3.8.

В дейтаграммных сетях, в которых используется множество маршрутов передачи пакетов для пары источник — получатель, точность аппроксимации системой $M/M/1$ ухудшается по другой причине, которую лучше проиллюстрировать на следующем примере.

Пример 13

Предположим, что узел A передает поток в узел B по двум линиям со скоростью обслуживания μ в простой сети, показанной на рис. 3.20. Пакеты поступают в A в соответствии с пуассоновским процессом с интенсивностью λ пакетов в секунду. Длительности передач пакетов распределены экспоненциально и не зависят от интервалов между моментами поступления, как и в системе $M/M/1$. Предположим, что поступающий поток делится поровну между двумя линиями. Однако как выполнить это разделение? Рассмотрим следующие возможности.

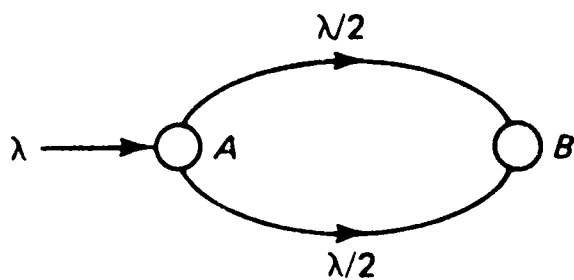


Рис. 3.20. Пуассоновский процесс с интенсивностью λ разделяется по двум линиям. Если деление имеет место случайно, каждая линия ведет себя аналогично системе $M/M/1$. Если деление происходит с помощью измерения длины очереди, система в целом ведет себя подобно системе массового обслуживания $M/M/2$.

1. **Рандомизация.** В этом случае каждый пакет в момент поступления в узел A случайно направляется в одну из двух линий равновероятно и независимо. Можно показать, что в этом случае процесс поступления пакетов в каждую из двух очередей является пуассоновским и независимым от длин пакетов (см. задачу 3.11). Следовательно, каждая из двух очередей ведет себя аналогично очереди в системе $M/M/1$ с интенсивностью поступлений пакетов $\lambda/2$ и средней задержкой пакета

$$T_R = \frac{1}{\mu - \lambda/2} = \frac{2}{2\mu - \lambda}, \quad (3.94)$$

что совпадает с аппроксимацией Клейнрока.

2. **Измерение.** В этом случае поступающие пакеты направляются в очередь, которая в настоящее время имеет наименьшую общую «задолженность» в битах (это эквивалентно тому, что пакеты направляются в очередь, которая первая ликвидирует эту задолженность, которая имеется в текущий момент). Эта схема работает аналогично системе $M/M/2$ с интенсивностью поступления λ , в которой каждая линия выполняет роль обслуживающего прибора. Используя результат из подразд. 3.4.1, получаем, что средняя задержка пакета вычисляется по формуле

$$T_M = \frac{2}{(2\mu - \lambda)(1 + \rho)}, \quad (3.95)$$

где $\rho = \lambda/2\mu$.

Из сравнения равенств (3.94) и (3.95) видно, что в смысле задержки система с измерением работает в $1/(1 + \rho)$ раз лучше, чем система с рандомизацией. Система с измерением имеет такое же преимущество перед системой с рандомизацией, как статистическое уплотнение с многими каналами перед временным уплотнением; последнее обсуждалось в примере 9 из подразд. 3.4.1. В общем случае, когда поток разделяется по разным маршрутам, предпочтительно использовать некоторую форму измерения, а не рандомизацию. Однако в противоположность случаю с рандомизацией измерение нарушает пуассоновский характер процесса поступления в точке разделения потоков. В нашем примере при использовании измерения интервалы между моментами поступления в каждую линию не являются экспоненциально распределенными и независимыми от длин предшествующих пакетов. Следовательно, использование измерения, которое рекомендуется для повышения эффективности работы, приводит к потере точности в аппроксимации системой $M/M/1$.

3.7. Обращение времени; теорема Берка

Исследование систем $M/M/1$, $M/M/t$, $M/M/\infty$ и $M/M/t/t$ основывалось на том, что для любого состояния j стационарная вероятность состояния j , умноженная на вероятность перехода из j в $j + 1$, равна стационарной вероятности состояния $j + 1$, умноженной на вероятность перехода из $j + 1$ в j . Такие соотношения называются *уравнениями детального баланса*; они справедливы для любой марковской цепи с целочисленными состояниями, в которой переходы происходят только между соседними состояниями, т. е. из j в $j - 1$, j или $j + 1$; такие цепи Маркова называются процессами *рождения и гибели*. Уравнения детального баланса приводят к важному свойству, которое, как мы сейчас объясним, называется свойством обратимости времени.

Рассмотрим неприводимую апериодическую цепь Маркова с дискретным временем X_n, X_{n+1}, \dots , которая имеет переходные

вероятности P_{ij} и стационарное распределение $\{p_j \mid j \geq 0\}$. Предположим, что цепь находится в стационарном состоянии, т. е.

$$P \{X_n = j\} = p_j \text{ при всех } n$$

(это происходит, если начальное состояние имеет стационарное распределение и эквивалентно предположению, что процесс начинается при $t = -\infty$).

Рассмотрим последовательность состояний в обратном по отношению ко времени порядке, т. е. последовательность X_n, X_{n-1}, \dots , начиная с некоторого n . Эта последовательность является цепью Маркова, что видно из следующих соотношений:

$$\begin{aligned} P \{X_m = j \mid X_{m+1} = i, X_{m+2} = i_2, \dots, X_{m+k} = i_k\} &= \\ = \frac{P \{X_m = j, X_{m+1} = i, X_{m+2} = i_2, \dots, X_{m+k} = i_k\}}{P \{X_{m+1} = i, X_{m+2} = i_2, \dots, X_{m+k} = i_k\}} &= \\ = \frac{P \{X_m = j\} P \{X_{m+1} = i \mid X_m = j\} P \{X_{m+2} = i_2, \dots, X_{m+k} = i_k \mid X_m = j, X_{m+1} = i\}}{P \{X_{m+1} = i\} P \{X_{m+2} = i_2, \dots, X_{m+k} = i_k \mid X_{m+1} = i\}} &= \\ = \frac{p_j P_{ji} P \{X_{m+2} = i_2, \dots, X_{m+k} = i_k \mid X_{m+1} = i\}}{p_i P \{X_{m+2} = i_2, \dots, X_{m+k} = i_k \mid X_{m+1} = i\}} = \frac{p_j P_{ji}}{p_i}. \end{aligned}$$

Таким образом, при условии, что задано состояние в момент $m+1$, состояние в момент m не зависит от состояний в моменты $m+2, m+3, \dots$. Вероятности обратных переходов равны

$$P_{ij}^* = P \{X_m = j \mid X_{m+1} = i\} = \frac{p_j P_{ji}}{p_i}, \quad i, j > 0. \quad (3.96)$$

Если $P_{ij}^* = P_{ij}$ при любых i, j , т. е. переходные вероятности прямой и обратной цепей равны, то цепь называется *обратимой по времени*.

Перечислим некоторые свойства обратной цепи.

1. Обратная цепь является неприводимой апериодической и имеет то же стационарное распределение, что и прямая цепь. (Это свойство может быть получено либо простыми рассуждениями, использующими определение обратной цепи, либо путем доказательства с помощью (3.96) справедливости соотношения $p_j = \sum_{i=0}^{\infty} p_i P_{ij}^*$. Интуитивная идея здесь состоит в том, что обратная цепь соответствует тому же процессу, только в обратном направлении времени. Таким образом, если стационарные вероятности соответствуют средним по времени (это должно выполняться для того, чтобы имело смысл понятие стационарного состояния), то стационарные вероятности одинаковы в обоих направлениях. Заметим, что ввиду равенства стационарных распределений для прямой и обратной цепей равенство (3.96) имеет интуитивное объяснение. Оно выражает то, что (с вероятностью единица) доля переходов из j в i среди всех переходов прямой цепи (эта доля равна $p_i P_{ij}$) равна доле переходов из i в j среди всех переходов обратной цепи (эта доля равна $p_i P_{ij}^*$).

2. Если можно найти неотрицательные числа $p_i, i \geq 0$, в сумме равные единице, и матрицу переходных вероятностей $P^* = [P_{ij}^*]$, такую, что

$$p_i P_{ij}^* = p_j P_{ji}, \quad i, j \geq 0, \quad (3.97)$$

то $\{p_i \mid i \geq 0\}$ является стационарным распределением, а P_{ij}^* — переходные вероятности обратной цепи. (Это можно понять, просуммировав (3.97) по всем j , тогда получается

$$\sum_{j=0}^{\infty} p_j P_{ji} = p_i \sum_{j=0}^{\infty} P_{ij}^* = p_i. \quad (3.98)$$

Заметьте, что $\{p_i \mid i \geq 0\}$ является стационарным распределением.) Это свойство, которое справедливо независимо от того, является ли цепь обратимой по времени или нет, полезно, если можно угадать природу обратной цепи и установить справедливость уравнения (3.97), из которого можно получить как p_j , так и P_{ij}^* ; см. разд. 3.8.

3. Цепь является обратимой по времени тогда и только тогда, когда справедливы уравнения детального баланса

$$p_i P_{ij} = p_j P_{ji}, \quad i, j \geq 0.$$

Это следует из равенства (3.96) и определения обратимости по времени. Другими словами, цепь является обратимой по времени, если для любых i и j доля переходов из i в j среди всех переходов равна доле переходов из j в i . В частности, цепи, соответствующие системам массового обслуживания $M/M/1$, $M/M/t$, $M/M/\infty$ и $M/M/t/t$, которые рассматривались в разд. 3.3 и 3.4, являются обратимыми по времени (в пределе при $\delta \rightarrow 0$). Кроме того, более общие цепи, соответствующие процессам рождения и гибели ($P_{ij} = 0$, если $|i - j| > 1$), являются обратимыми по времени.

Идея обратимости по времени просто обобщается на цепи Маркова с непрерывным временем (марковские процессы). Соответствующий анализ может быть выполнен либо непосредственно, либо путем дискретизации времени интервалами длины δ , рассмотрения соответствующей цепи с дискретным временем и перехода обратно к непрерывной цепи путем устремления $\delta \rightarrow 0$. Все результаты, касающиеся обратной цепи, получаются из аналогичных результатов для цепи с дискретным временем путем замены переходных вероятностей на интенсивности переходов. В частности, если цепь с непрерывным временем имеет интенсивности переходов q_{ij} , стационарное распределение $\{p_j \mid j \geq 0\}$ и является неприводимой, то

1) обратная цепь является цепью Маркова с непрерывным временем, имеющей то же стационарное распределение, что и пря-

мая цепь, и интенсивности переходов

$$q_{ij}^* = \frac{p_j q_{ji}}{p_i}, \quad i, j \geq 0; \quad (3.99)$$

- 2) если существуют распределения вероятностей $\{p_j \mid j \geq 0\}$ и неотрицательные числа q_{ij}^* ($i, j = 0, 1, \dots$), такие, что

$$p_i q_{ij}^* = p_j q_{ji}, \quad i, j \geq 0, \quad (3.100)$$

и для любого $i \geq 0$

$$\sum_{j=0}^{\infty} q_{ij} = \sum_{j=0}^{\infty} q_{ij}^*, \quad (3.101)$$

то $\{p_j \mid j \geq 0\}$ является стационарным распределением как прямой, так и обратной цепи, а q_{ij}^* представляют собой интенсивности переходов обратной цепи;

- 3) прямая цепь является обратимой по времени тогда и только тогда, когда ее стационарное распределение и интенсивности переходов удовлетворяют уравнениям детального баланса

$$p_i q_{ij} = p_j q_{ji}, \quad i, j \geq 0.$$

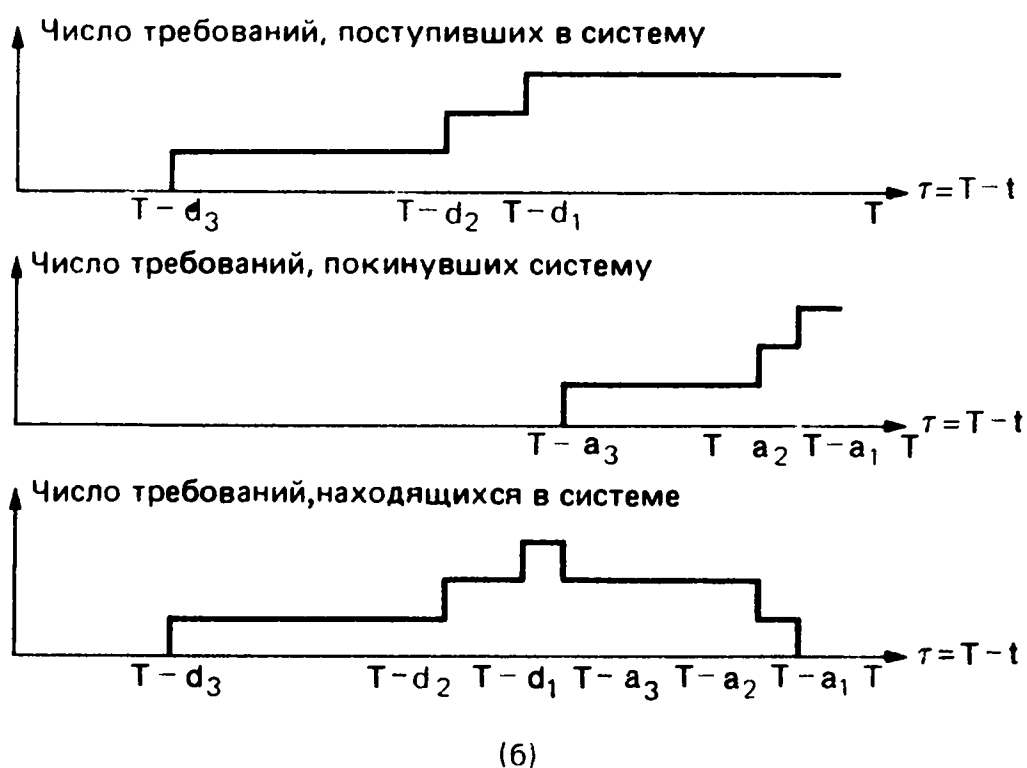
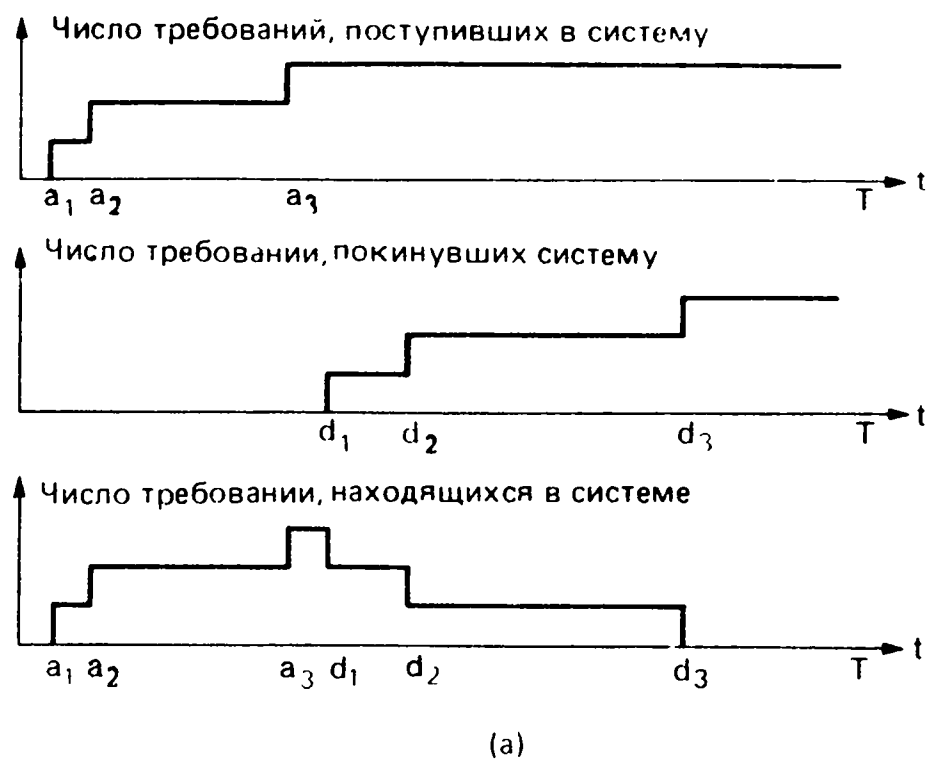
Рассмотрим теперь системы массового обслуживания $M/M/1$, $M/M/m$ и $M/M/\infty$. Предположим, что начальное состояние имеет стационарное распределение, так что каждая система массового обслуживания находится в стационарном состоянии в любой момент времени. Обратный процесс можно представить другой системой массового обслуживания, в которой моменты ухода требований из системы соответствуют моментам поступления в исходной системе, а моменты поступления — моментам ухода в исходной системе (рис. 3.21). Так как обратимость по времени справедлива для всех рассматриваемых систем, то в стационарном состоянии прямые и обратные системы являются статистически неразличимыми. В частности, используя то, что процесс ухода требований в прямой системе соответствует процессу поступления требований в обратной системе, получаем следующий результат.

Теорема Берка. Рассмотрим систему $M/M/1$, $M/M/m$ или $M/M/\infty$ с интенсивностью поступления требований λ . Предположим, что система начинает действовать, находясь в стационарном состоянии. Тогда справедливы следующие утверждения.

- (а) Процесс ухода требований из системы является пуассоновским с интенсивностью λ .
- (б) В любой момент времени t число требований в системе не зависит от последовательности моментов ухода до момента t .
- (в) Если требования обслуживаются в порядке их поступления, то при условии, что требование уходит из системы в момент t , момент поступления этого требования не зависит от процесса ухода до момента t .

Рис. 3.21.

a — число требований, поступивших, ушедших и находящихся в прямой системе в течение отрезка времени $[0, T]$; b — число требований, поступивших, ушедших и находящихся в обратной системе в течение отрезка времени $[0, T]$.



Доказательство. (а) Это утверждение следует из того, что в стационарном состоянии прямая и обратная системы статистически неразличимы, а процесс ухода в прямой системе является процессом поступления требований в обратной системе.

(б) Как показано на рис. 3.22, при фиксированном моменте времени t моменты ухода требований в прямом процессе являются также моментами поступления после t в обратном процессе. Процесс поступления в обратной системе является независимым пуассоновским процессом; таким образом, будущее процесса поступления не зависит от числа требований в системе в настоящий момент, что в терминах прямой системы означает, что прошлое процесса ухода не зависит от числа требований в системе в настоящий момент.

(в) Рассмотрим требование, поступившее в момент t_1 и ушедшее в момент t_2 (рис. 3.23). В терминах обратной системы процесс поступления является независимым пуассоновским процес-

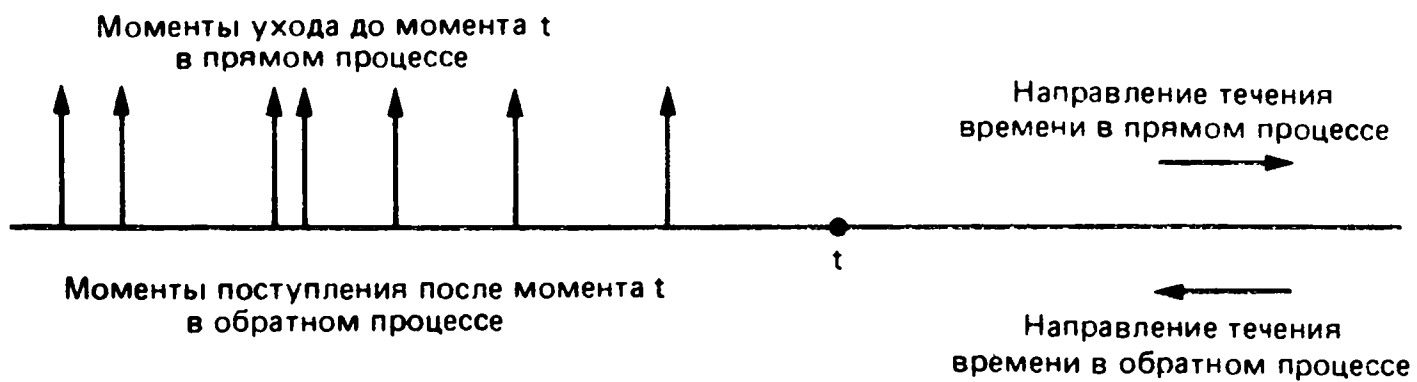


Рис. 3.22. Моменты ухода требований из прямой системы до момента t становятся моментами поступления требований в обратной системе после момента t .

сом, поэтому процесс поступления слева от t_2 не зависит от времени пребывания в системе требований, поступивших в момент t_2 или справа от него. В частности, величина $t_2 - t_1$ не зависит от процесса поступления (в обратной системе) слева от t_2 . В терминах прямой системы это значит, что величина $t_2 - t_1$ не зависит от процесса ухода слева от t_2 .

Следует здесь заметить, что утверждения (б) и (в) теоремы Берка заметно противоречат интуиции. Казалось бы, возникновение близко расположенных моментов ухода предполагает, что система загружена нетипично большим числом требований в очереди. Однако теорема Берка показывает, что это не так. При более аккуратном рассмотрении заметим, что теорема Берка ничего не говорит о состоянии системы *перед* возникновением близко расположенных моментов ухода. Такое состояние в соответствии с интуицией стремилось бы иметь ненормально большое число требований в очереди.

Как применение теоремы рассмотрим простую сеть массового обслуживания с пуассоновскими потоками поступления требований и двумя тандемными очередями, имеющими экспоненциальные длительности обслуживания (рис. 3.24). Главное различие между

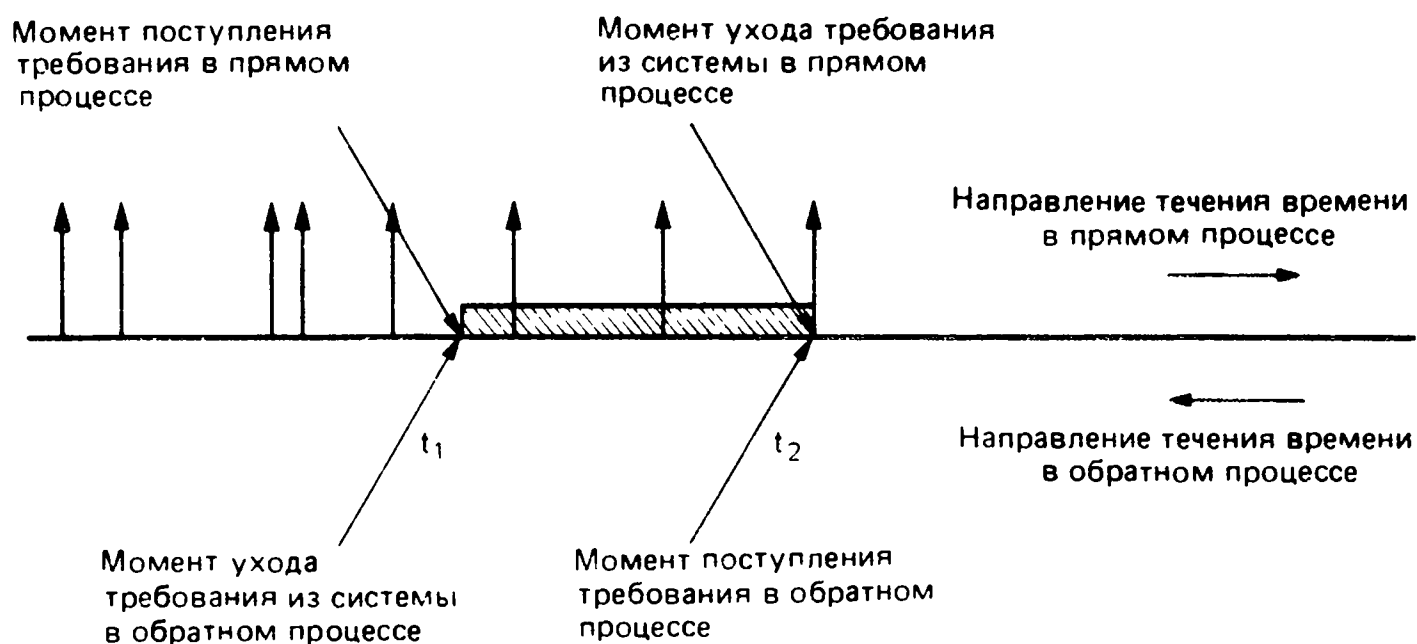


Рис. 3.23. Доказательство утверждения (в) теоремы Берка. Для обратной системы требования, поступающие после момента t_2 , не влияют на время пребывания в системе требования, которое поступило в момент t_2 . Для прямой системы время пребывания требования в системе ($t_2 - t_1$) не зависит от процесса уходов до момента ухода этого требования из системы.

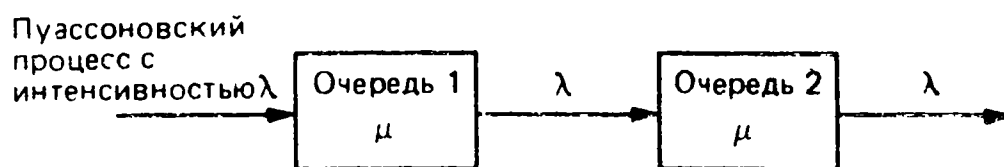


Рис. 3.24. Система из двух тандемных очередей. Длительности обслуживания в обеих очередях распределены экспоненциально и взаимно независимы. Используя теорему Берка, можно показать, что числа требований в очередях 1 и 2 в данный момент времени взаимно независимы и

$$P \{n \text{ в очереди } 1, m \text{ в очереди } 2\} = \rho_1^n (1 - \rho_1) \rho_2^m (1 - \rho_2),$$

т. е. две очереди ведут себя так, как если бы они были независимыми отдельными системами $M/M/1$.

этой системой и системой, рассмотренной в предыдущем разделе, состоит в том, что здесь мы предполагаем, что длительности обслуживания требований в первой и второй очередях взаимно независимы, а также не зависят от процесса поступления требований. В предыдущем разделе это называлось аппроксимацией Клейнрока. Из этого предположения будет следовать, что распределение числа требований в обеих очередях такое же, как если они были бы изолированными, независимыми системами $M/M/1$. Это будет также показано при более общем рассмотрении в следующем разделе.

Пусть интенсивность пуассоновского процесса поступления равна λ и средние длительности обслуживания в очередях 1 и 2 равны соответственно $1/\mu_1$ и $1/\mu_2$. Пусть $\rho_1 = \lambda/\mu_1$ и $\rho_2 = \lambda/\mu_2$ — соответствующие коэффициенты использования, и предположим, что $\rho_1 < 1$ и $\rho_2 < 1$. Покажем, что в стационарном режиме справедливы следующие утверждения.

(а) Числа требований, ожидающих в данный момент в очередях 1 и 2, независимы и не зависят от последовательности предыдущих моментов ухода требований из очереди 2¹⁾. Кроме того,

$$P \{n \text{ в очереди } 1, m \text{ в очереди } 2\} = \rho_1^n (1 - \rho_1) \rho_2^m (1 - \rho_2). \quad (3.102)$$

(б) Предполагая, что требования обслуживаются в каждой очереди в порядке поступления, времена пребывания требования (включая обслуживание) в очереди 1 и очереди 2 независимы и не зависят от процесса ухода из очереди 2, рассматриваемого до момента, когда это требование покидает систему.

Для доказательства утверждения (а) сначала заметим, что очередь 1 является очередью в системе $M/M/1$, поэтому в силу

¹⁾ Здесь и далее, а также на рис. 3.24, если это специально не оговаривается, очередью называется система массового обслуживания, т. е. очередь и следующий за ней обслуживающий прибор. — Прим. ред.

утверждения (а) теоремы Берка процесс ухода из очереди 1 является пуассоновским и независимым от длительностей обслуживания в очереди 2. Следовательно, очередь 2, рассматриваемая отдельно, является системой $M/M/1$. Таким образом, согласно результатам, полученным в разд. 3.1, имеем

$$\begin{aligned} P \{n \text{ в очереди } 1\} &= \rho_1^n (1 - \rho_1), \\ P \{m \text{ в очереди } 2\} &= \rho_2^m (1 - \rho_2). \end{aligned} \quad (3.103)$$

Из утверждения (б) теоремы Берка следует, что число требований, находящихся в настоящий момент в очереди 1, не зависит от последовательности более ранних поступлений в очередь 2 и, следовательно, от числа требований, находящихся в настоящий момент в очереди 2. Это означает, что

$$\begin{aligned} P \{n \text{ в очереди } 1, m \text{ в очереди } 2\} &= \\ &= P \{n \text{ в очереди } 1\} \cdot P \{m \text{ в очереди } 2\}, \end{aligned}$$

используя формулы (3.103), получаем требуемое произведение в виде (3.102).

Для доказательства утверждения (б) заметим, что, согласно утверждению (в) теоремы Берка, время пребывания требования в очереди 1 не зависит от последовательности моментов поступления в очередь 2, рассматриваемую до момента поступления данного требования в очередь 2. Однако эти моменты поступления (вместе с соответствующими независимыми длительностями обслуживания) определяют время пребывания требования в очереди 2, а также процесс ухода из очереди 2 до момента ухода этого требования из системы. Это доказывает утверждение (б).

Утверждение (б) о независимости времен пребывания одного и того же требования в очередях 1 и 2 противоречит интуиции, так как, казалось бы, что большое число требований, пребывающих в очереди 1, вероятно, повторно встретится в очереди 2. Для иллюстрации того, каким тонким является этот результат о независимости и как опасно использовать его без достаточного продумывания, заметим, что времена ожидания требований *до поступления на обслуживание* в двух очередях *не* являются независимыми величинами. Для того чтобы понять это, предположим, что $\mu_1 = \mu_2 = \mu$ и что λ очень мало по сравнению с μ . Тогда почти все требования имеют нулевое время ожидания в обеих очередях. Однако при условии, что время ожидания требований в очереди 1 положительное, время ожидания этого же требования в очереди 2 будет положительным по крайней мере с вероятностью $1/2$ (вероятностью того, что рассматриваемое требование будет иметь время обслуживания в очереди 1 меньшее, чем время обслуживания требования, стоящего непосредственно перед ним в очереди 2). Следовательно, для одного и того же требования длительности

ожидания в очередях 1 и 2 не являются независимыми — они становятся независимыми, если добавляются соответствующие длительности обслуживания.

Заметим, что, согласно утверждению (а) теоремы Берка, процессы поступления и ухода в обеих очередях пуассоновские. Этот факт, вместе с утверждениями (а) и (б), может быть установлен для более широкого класса сетей массового обслуживания с пуассоновскими процессами поступлений и независимыми экспоненциально распределенными длительностями обслуживания. Такие сети назовем *ациклическими* и определим их следующим образом. Назовем очередь j *соседом очереди i по направлению движения потока*, если существует положительная вероятность того, что требование, покинувшее очередь i , сразу поступит в очередь j . Мы говорим, что очередь j находится за очередью i по направлению потока, если существует последовательность очередей, начиная от i и кончая j , такая, что каждая после i очередь в этой последовательности является соседом предшествующей очереди по направлению движения потока. Сеть с очередями называется *ациклической*, если невозможно найти две очереди i и j , такие, что j находится за i , а i — за j по направлению потока. Свойство ацикличности сети является существенным для того, чтобы каждая очередь имела пуассоновский характер процессов поступления и ухода (см. следующий раздел). Однако, как будет показано в следующем разделе, тот факт, что распределение числа находящихся в очереди требований имеет форму произведения (3.102), естественно, обобщается на сети, которые не являются ациклическими.

3.8. Сети очередей¹⁾; теорема Джексона

Как указывалось в разд. 3.6, основная трудность анализа сетей из линий связи заключается в том, что интервалы между моментами поступления пакетов после прохождения пакетов через первую очередь начинают коррелировать с их длинами. Оказывается, что если каким-либо образом устранить эту корреляцию (как раз на это направлена аппроксимация Клейнрока) и использовать рандомизацию для разделения потока по различным маршрутам, то можно найти среднее число пакетов в системе, рассматривая каждую очередь в сети как систему $M/M/1$. Этот важный результат известен как теорема Джексона. В этом разделе доказывается простой вариант этой теоремы.

Рассмотрим сеть из K систем массового обслуживания, каждая с одним обслуживающим прибором; требования поступают в сеть извне в каждую очередь i ; для различных i процессы постуже-

¹⁾ См. примечание на с. 209. — Прим. ред.

ний являются независимыми пуассоновскими процессами с интенсивностями r_i . После того как требование будет обслужено в очереди i , оно поступает в очередь j с вероятностью P_{ij} или выходит из сети с вероятностью $1 - \sum_{j=1}^K P_{ij}$. Суммарная интенсивность поступления требований в очередь j , обозначаемая через λ_j , удовлетворяет условию

$$\lambda_j = r_j + \sum_{i=1}^K \lambda_i P_{ij}, \quad j = 1, \dots, K. \quad (3.104)$$

Эти уравнения представляют собой линейную систему, в которой суммарные интенсивности λ_j , $j = 1, \dots, K$ образуют набор из K неизвестных. Предположим, что эти уравнения решаются однозначно и дают выражение λ_j , $j = 1, \dots, K$, через r_j и P_{ij} , $i, j = 1, \dots, K$. Можно показать, что это предположение выполняется при очень общих условиях, например, если все вероятности выхода из системы $(1 - \sum_{j=1}^K P_{ij})$, $i = 1, \dots, K$, положительные или в более общем виде, если для каждой очереди i_1 существует очередь i с $(1 - \sum_{j=1}^K P_{ij}) > 0$ и последовательность i_1, i_2, \dots, i_k, i , такая, что $P_{i_1 i_2} > 0, \dots, P_{i_k i} > 0$.

Длительности обслуживания требований в i -й очереди предполагаются экспоненциально распределенными со средним значением $1/\mu_i$ и взаимно независимыми и независимыми от процесса поступления в очередь. Коэффициент использования для каждой очереди обозначается через

$$\rho_i = \frac{\lambda_i}{\mu_i}, \quad i = 1, \dots, K; \quad (3.105)$$

предполагается, что $\rho_i < 1$ при всех i .

Для того чтобы для сети пакетной передачи, рассмотренной в разд. 3.6, построить модель описанного выше типа, необходимо принять несколько упрощающих предположений в дополнение к предположению о пуассоновском распределении поступления пакетов и экспоненциальном распределении длин пакетов. Первое из них — это предположение о независимости длин пакетов и интервалов между их поступлениями; оно обсуждалось ранее. Второе предположение, существенное для дейтаграммных сетей, состоит в том, что разветвление потока в узле сети достаточно хорошо моделируется процессом рандомизации, в соответствии с которым каждый уходящий из очереди i пакет поступает в очередь j с вероятностью P_{ij} ; как отмечалось в разд. 3.6, такое разветвление не всегда имеет место. Сеть пакетной передачи отличается от модели из этого раздела еще и тем, что она имеет несколько потоков, которые могут иметь различные вероятности маршрутов в каждом узле и сохраняют свои свойства по мере того, как передаются по различным маршрутам. Эта трудность может

быть частично решена путем использования обобщения теоремы Джексона, которое относится к сети с многими классами требований. В рамках этой более общей постановки можно рассматривать потоки, относящиеся к различным парам источник—получатель, как соответствующие различным классам требований. Если все потоки имеют одинаковую среднюю длину пакета, то оказывается, что теорема Джексона в том виде, как она формулируется ниже, будет верна, когда справедливы указанные выше упрощающие предположения.

Переходя к анализу системы, рассмотрим ее как цепь Маркова с непрерывным временем (марковский процесс) с состояниями n_1, n_2, \dots, n_K , где n_i обозначает число требований в очереди i . Пусть $P(n_1, \dots, n_K)$ — стационарное распределение цепи.

Теорема Джексона. Предположим, что $\rho_i < 1$, $i = 1, \dots, K$, тогда для всех $n_1, \dots, n_K \geq 0$

$$P(n_1, \dots, n_K) = P_1(n_1) P_2(n_2) \dots P_K(n_K), \quad (3.106)$$

где

$$P_i(n) = \rho_i^n (1 - \rho_i), \quad n \geq 0. \quad (3.107)$$

Доказательство. Используем метод, описанный в предыдущем разделе, для определения интенсивностей переходов обратного процесса и покажем, что вместе со стационарными вероятностями (3.106) и (3.107) они удовлетворяют условиям (3.100) и (3.101) из предыдущего раздела. (Цепь Маркова не является здесь обратимой по времени. Тем не менее использование обратного процесса является как аналитически удобным, так и концептуально полезным.)

Переходя к деталям, обозначим вектор состояния через

$$n = (n_1, n_2, \dots, n_K)$$

и введем обозначение (ср. с равенством (3.106))

$$P(n) = P_1(n_1) P_2(n_2) \dots P_K(n_K). \quad (3.108)$$

Для любых двух векторов состояния n и n' пусть $q_{nn'}$ будет соответствующей интенсивностью перехода. Теорема Джексона будет доказана, если мы сможем найти интенсивности $q_{nn'}^*$, такие, что для любых n, n'

$$P(n) q_{nn'} = P(n') q_{n'n}, \quad (3.109)$$

$$\sum_n q_{nm} = \sum_m q_{nm}^* \quad (3.110)$$

Для векторов состояний n и n' вида

$$n = (n_1, \dots, n_i, \dots, n_K)$$

$$n' = (n_1, \dots, n_i - 1, \dots, n_K)$$

имеем

$$q_{nn'} = r_i \quad (3.111)$$

$$q_{n'n} = \mu_i (1 - \sum_j P_{ij}). \quad (3.112)$$

Если положить

$$q_{nn'}^* = \lambda_i (1 - \sum_j P_{ij}), \quad (3.113)$$

$$q_{n'n}^* = \frac{\mu_i r_i}{\lambda_i}, \quad (3.114)$$

то видно, что равенство (3.109) выполняется.

Далее рассмотрим векторы состояний n и n' вида

$$n = (n_1, \dots, n_i, \dots, n_j, \dots, n_K),$$

$$n' = (n_1, \dots, n_i + 1, \dots, n_j - 1, \dots, n_K).$$

Имеем

$$q_{nn'} = \mu_j P_{ji}. \quad (3.115)$$

Если положить

$$q_{n'n}^* = \frac{\mu_i \lambda_j P_{ji}}{\lambda_i}, \quad (3.116)$$

то снова видно, что равенство (3.109) выполняется.

Так как для всех других типов пар векторов состояний n, n' имеем

$$q_{nn'} = 0, \quad (3.117)$$

то можно определить

$$q_{n'n}^* = 0 \quad (3.118)$$

и быть уверенными, что равенство (3.109) выполняется при всех n и n' . Непосредственные вычисления с использованием формул (3.111)—(3.118) и (3.104) позволяют установить справедливость равенства (3.110).

Заметим, что интенсивности переходов $q_{nn'}^*$, определяемые равенствами (3.113), (3.114), (3.116) и (3.118), являются интенсивностями обратного процесса. Можно показать, что обратный процесс соответствует сети очередей, в которой поток, поступающий в очередь i извне, является пуассоновским процессом с интенсивностью $\lambda_i (1 - \sum_j P_{ij})$ (ср. с равенством (3.113)). В обратном процессе вероятность перехода из очереди i в очередь j равна $\lambda_j P_{ji} / (r_i + \sum_k \lambda_k P_{ki})$ (ср. с равенствами (3.114) и (3.116)).

Существует несколько обобщений теоремы Джексона. Например, форма произведения (3.106) остается в силе, если каждая очередь i имеет много обслуживающих приборов, например m_i , а не один. В этом случае формула, соответствующая (3.107), совпадает с формулой для системы $M/M/m_i$. Имеются обобщения теоремы на случай замкнутых сетей, внутри которых циркулирует фиксированное число требований; поступление извне или уход из сети не разрешается (см. задачу 3.28).

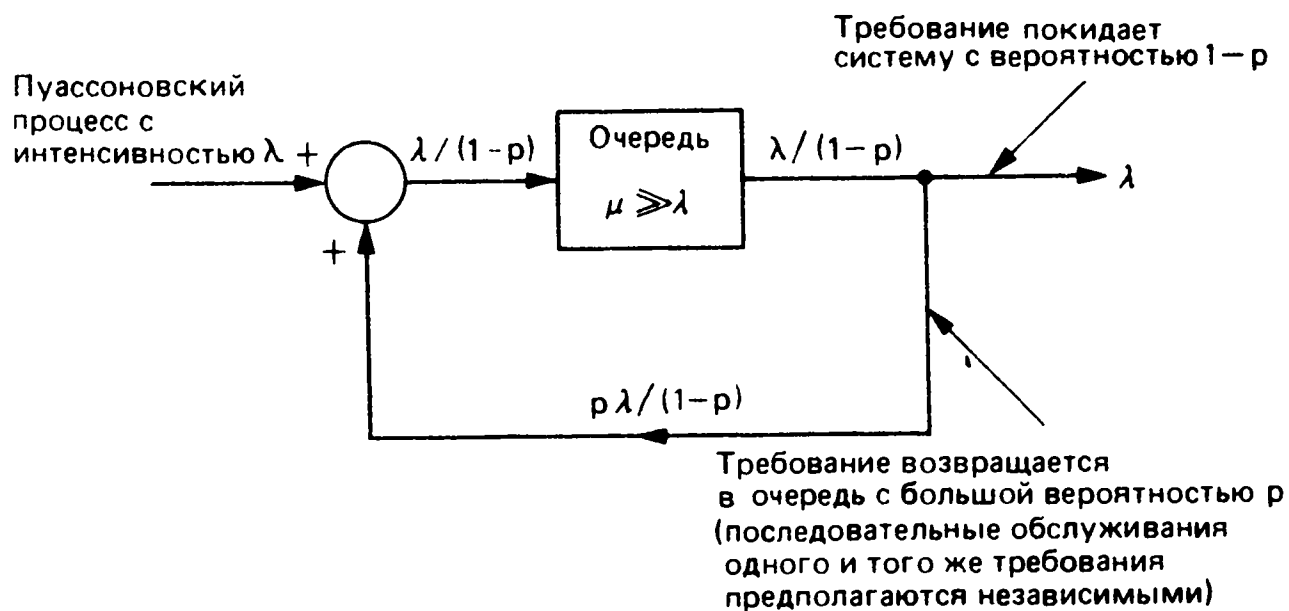


Рис. 3.25. Пример очереди внутри сети, в которой процесс внешних поступлений пуассоновский, а суммарный процесс поступлений в очередь не является пуассоновским. Требование, поступившее извне, обычно быстро обслуживается (так как μ много больше λ) и с большой вероятностью возвращается в очередь по петле обратной связи. В результате суммарный процесс поступления в очередь обычно состоит из пачек поступлений, причем каждая пачка появляется при поступлении извне одного требования.

Перейдем теперь к интерпретации теоремы Джексона. Во-первых, из формулы (3.106) видно, что числа требований в различных очередях в данный момент времени являются *независимыми*. Выражение для распределения числа требований в каждой очереди i совпадает с соответствующим выражением для системы $M/M/1$ (ср. равенство (3.107) и соответствующие равенства из разд. 3.3). Этот результат является весьма примечательным, так как можно показать на примере, что процесс поступления в каждую очередь не обязательно должен быть пуассоновским. В самом деле, если существует вероятность того, что требование может побывать в некоторой очереди более чем один раз (ситуация с обратной связью), процесс поступления не будет пуассоновским. Например (рис. 3.25), предположим, что имеется единственная очередь со скоростью обслуживания, которая очень велика по сравнению со скоростью поступления извне. Предположим также, что с вероятностью p , близкой к единице, требование сразу же после окончания обслуживания возвращается обратно в очередь. Следовательно, в момент поступления требования в очередь имеется большая вероятность другого поступления в очередь через короткий промежуток времени (а именно поступления по обратной связи), а в произвольный момент имеется очень малый шанс скорого поступления, так как λ мало. Другими словами, имеется тенденция поступления в очередь пачками из-за поступления одного требования извне. Следовательно, процесс поступления в очередь не имеет независимых интервалов между моментами поступления и не может быть пуассоновским.

К сожалению, наше доказательство теоремы Джексона основывается на алгебраических преобразованиях и не дает достаточ-

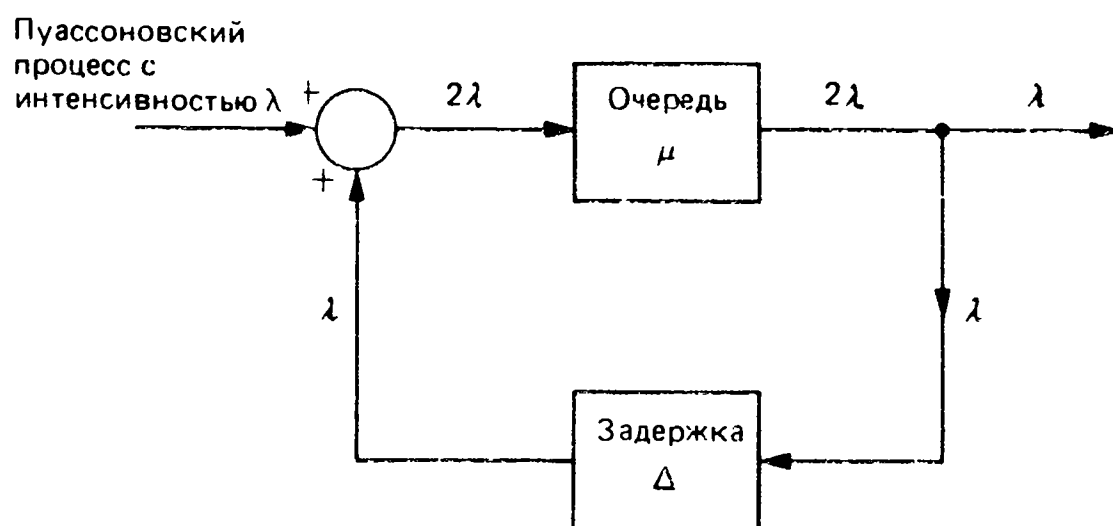


Рис. 3.26. Эвристическое объяснение теоремы Джексона. Рассмотрим сеть, показанную на рис. 3.25, в которую добавлена произвольно малая положительная задержка Δ в петлю обратной связи. Распределение числа требований в очереди равно распределению числа требований в стационарной системе $M/M/1$, а содержимое линии задержки, которое является независимым отрезком длины Δ пуассоновского процесса, формирует стационарное распределение всей системы. Следовательно, в соответствии с теоремой Джексона распределение в стационарной системе $M/M/1$ является стационарным распределением для рассматриваемой очереди, даже если суммарный процесс поступления в очередь не пуассоновский.

ного понимания того, почему на самом деле справедлив этот замечательный результат. Поэтому дадим эвристическое объяснение теоремы для случая сети с обратной связью, показанной на рис. 3.25. Это объяснение может быть обобщено и сделано строгим, хотя ценой значительного технического усложнения [250].

Предположим, что в сети с одной очередью, которая рассматривалась ранее, введена задержка Δ в цепь обратной связи (см. рис. 3.26, на котором для удобства выбрано $\rho = 1/2$). Обозначим через $n(t)$ число требований в очереди в момент t , а через $f_\Delta(t)$ — содержимое линии задержки в момент t . Функция $f_\Delta(t)$ здесь интерпретируется как функция времени, которая определяет выход требований из линии задержки в следующем за t интервале длины Δ , т. е. в интервале $(t, t + \Delta]$. Предположим, что начальное распределение состояния очереди в момент времени 0 (это состояние есть $n(0)$) является стационарным распределением для системы $M/M/1$, т. е.

$$P\{n(0) = n\} = \rho^n (1 - \rho), \quad (3.119)$$

где $\rho = 2\lambda/\mu$ — коэффициент использования. Предположим также, что $f_\Delta(0)$ является отрезком пуассоновского процесса поступления с интенсивностью λ . Требования в $f_\Delta(0)$ имеют длительности обслуживания, которые являются независимыми, экспоненциально распределенными с параметром μ . Предположим, что $n(0)$ и $f_\Delta(0)$ независимы. Тогда поступление в очередь за интервал $[0, \Delta)$ будет равно сумме двух независимых пуассоновских потоков, которые не зависят от числа требований в очереди в момент 0. Из этого следует, что в интервале $[0, \Delta)$ очередь будет вести себя

как система $M/M/1$ в стационарном состоянии. Поэтому $n(\Delta)$ будет иметь распределение, совпадающее со стационарным распределением (3.119) для системы $M/M/1$, и, согласно утверждению (б) теоремы Берка, $n(\Delta)$ не будет зависеть от процесса ухода из очереди в интервале $[0, \Delta)$ или, что эквивалентно, от $f_\Delta(\Delta)$ — содержимого линии задержки в момент Δ . Кроме того, согласно утверждению (а) теоремы Берка, $f_\Delta(\Delta)$ будет отрезком пуассоновского процесса. Таким образом, подводя итог, видим, что мы запустили систему при независимых начальных условиях $n(0)$ и $f_\Delta(0)$, которые имели соответственно стационарное распределение для системы $M/M/1$ и статистики пуассоновского процесса, а через Δ секунд получили $n(\Delta)$ и $f_\Delta(\Delta)$ с теми же свойствами. Для любого t аналогично можно показать, что $n(t)$ и $f_\Delta(t)$ имеют те же самые свойства. Из этого следует, что стационарное распределение (3.119) для системы $M/M/1$ является стационарным распределением для рассматриваемой системы массового обслуживания с произвольной положительной величиной задержки обратной связи Δ ; это наводит на мысль о справедливости теоремы Джексона. Заметим, что для этого доказательства не требуется, чтобы процесс поступления по обратной связи и, следовательно, также общий процесс поступления в очередь были пуассоновскими. Действительно, можно показать, что последовательные части Δ потока, поступающего по обратной связи, коррелированы, так как с вероятностью 0,5 требование, покинувшее очередь, поступает через Δ секунд обратно в очередь. Следовательно, в интервале $[0, \infty)$ процесс поступления по обратной связи не является пуассоновским. Это согласуется с нашими более ранними замечаниями относительно примера, приведенного на рис. 3.25.

3.9. Заключение

Математические модели теории массового обслуживания дают возможность качественного понимания работы сетей передачи данных и дают количественные оценки для средней задержки пакета. Примером первого является сравнение временного и статистического уплотнения, а примером последнего — анализ задержки для систем с резервированием.

Для получения поддающихся исследованию моделей систем массового обслуживания для сетей передачи данных часто необходимо сделать упрощающие предположения. Простой пример их — это аппроксимация Клейнрока, которая рассматривалась в разд. 3.6. Оценки задержки, основанные на этой аппроксимации, пригодны для многих приложений. Более точный способ вычисления задержек заключается в моделировании, однако оно может потребовать много времени, оказаться дорогим и не привести к пониманию существа дела.

Теорема Литтла является простым, но чрезвычайно полезным результатом, так как она справедлива при очень общих предположениях. Более сильные результаты, чем эта теорема, были получены в предположении, что процесс поступления требований пуассоновский, а интервалы между поступлениями требований и длительности обслуживания являются независимыми. Это привело к системе $M/G/1$ и ее обобщениям на системы с резервированием и системы массового обслуживания с приоритетом. Простым графическим методом проанализировано большое число важных математических моделей систем массового обслуживания. Другой способ исследования основывался на теории цепей Маркова; он привел к выводу распределения вероятностей для числа требований в системах $M/M/1$, $M/M/m$ и аналогичных им.

Обратимость по времени является важным понятием, которое помогло доказать и понять смысл теоремы Джексона, а также получить представление о современных вопросах теории массового обслуживания.

3.10. Замечания, источники и дополнительная литература

Раздел 3.2. Теорема Литтла была сформулирована в [164]. Строгие доказательства при различных предположениях приведены в [222, 223]. Некоторые применения этой теоремы для отыскания границ для характеристик компьютерных систем даны в [220].

Раздел 3.3. Для общей подготовки по пуассоновским процессам, цепям Маркова и родственным вопросам см. [203, 204 и 132]. Стандартное изложение теории массового обслуживания содержится в [49, 104, 116 и 143]. Упоминание того, что при пуассоновском процессе в момент поступления нового требования распределение числа требований в системе является таким же, как обычно (подразд. 3.3.2), имеется в [255].

Раздел 3.4. Системы массового обслуживания, которые допускают анализ с помощью теории цепей Маркова, включают также системы, в которых длительности обслуживания имеют распределение Эрланга; см. [144, гл. 4]. Для изучения более общих моделей, а также численных методов см. [108, 135, 182 и 245].

Раздел 3.5. Формула Поллачека—Хинчина выводится с помощью z -преобразований; см. [143]. Этот вывод не очень полезен для понимания существа вопроса, но он дает распределение вероятностей числа требований в системе (а не только среднее число, которое было получено с помощью нашего намного более простого доказательства). Для дальнейшего изучения задержек в системах ARQ см. [8 и 232].

Результаты по системам с опросом и резервированием получены недавно, см. в [48, 59, 68, 77, 126 и 152]. Источниками, на которых основывалось наше рассмотрение, являются работа [112] по системам массового обслуживания с неограниченным обслуживанием, работа [184] по системам с ограниченным обслуживанием и [121] по несимметричным системам с опросом. Работа [226] является монографией, посвященной системам с опросом. В литературе рассматриваются в основном две системы с резервированием и опросом: симметричная система, в которой все пользователи имеют одинаковые статистики для интервалов между моментами поступления и интервалов резервирования, и несимметричная система, в которой эти статистики зависят от пользователя. Для первой системы допускаются простые выражения для средних длительностей ожидания, а для второй не допускаются. Мы рассмотрели частично симметричную систему, в которой все пользователи имеют одинаковые статистики интервалов резервирования. То, что для этой системы получаются простые выражения, не было известно ранее, и в этом смысле наши формулы являются новыми. Наше изложение с использованием простых графических доказательств также является оригинальным. Результат, сформулированный в задаче 3.25 для систем с ограниченным обслуживанием и уплотнением интервалов резервирования и данных, является новым.

Обширное рассмотрение систем массового обслуживания с приоритетами имеется в книге [129]. Более простое, но менее исчерпывающее изложение дается в [143].

Раздел 3.6. Исследование задержки в сетях передачи данных с использованием аппроксимации системой $M/M/1$ было впервые проведено в книге [141]. В [191] и [256] с помощью аналитического исследования и численного моделирования изучается поведение системы из двух тандемных станций в условиях, когда длительности обслуживания требований на станциях являются зависимыми. Специальный выпуск журнала [126] содержит обзор недавних работ по этой теме.

Раздел 3.7. Понятие обратимости по времени в сетях массового обслуживания глубоко исследовалось в книге [136].

Раздел 3.8. Имеется обширная литература по получению для сетей массового обслуживания решений в форме произведений; она следует за статьей Джексона [129]. Обзор [57] содержит 314 ссылок. Имеются две книги по этому вопросу [37, 137]. Эвристическое объяснение теоремы Джексона содержится в [250].

Задачи

- 3.1.** Посетители приходят в ресторан быстрого обслуживания со скоростью 5 человек в минуту и ожидают в среднем выполнения заказа 5 минут. Посетители едят в ресторане с вероятностью 0,5 и проводят время без еды

- с вероятностью 0,5. Прием пищи требует в среднем 20 минут. Какое среднее число посетителей в ресторане?
- 3.2. Рассеянный профессор назначает двум студентам встречи в одно и то же время. Длительности встреч независимы и экспоненциально распределены со средним тридцать минут. Первый студент приходит вовремя, а второй на пять минут позже. Чему равно математическое ожидание между приходом первого студента и уходом второго студента?
- 3.3. Посетитель входит в банк и обнаруживает, что все четыре клерка заняты обслуживанием клиентов. В банке нет других клиентов, и, таким образом, посетителя начнут обслуживать, как только один из обслуживаемых клиентов уйдет. Длительности обслуживания клиентов независимы, имеют одинаковые экспоненциальные распределения.
- Какая вероятность, что посетитель последним покинет банк, предполагая, что больше никто из посетителей не придет?
 - Если среднее время обслуживания равно одной минуте, какое среднее время посетитель проведет в банке?
 - Изменится ли ответ в случае (а), если есть несколько дополнительных посетителей, ожидающих в общей очереди, и посетители обслуживаются в порядке их прихода?
- 3.4. Рассмотрите поток, пакеты которого поступают в соответствии с пуассоновским процессом с интенсивностью 10 пакетов в секунду. Если время между поступлениями любых двух следующих один за другим пакетов меньше, чем время передачи первого пакета, то говорят, что два пакета попали в конфликтную ситуацию. (Это понятие станет более ясным в гл. 4, в которой будут рассматриваться системы множественного доступа) Найдите вероятность того, что пакет конфликтует либо с его предшественником или со следующим за ним пакетом предполагая, что
- все пакеты имеют время передачи 20 мсек;
 - пакеты имеют независимые экспоненциальные распределения длительности передач со средним 20 мсек.
- 3.5. Линия связи с пропускной способностью 50 кбит/с используется для передачи 10 сеансов, каждый из которых представляет собой пуассоновский поток с интенсивностью 150 пакетов в минуту. Длины пакетов имеют экспоненциальное распределение со средним 1000 бит.
- Для каждого сеанса найдите среднее число пакетов в очереди, среднее число пакетов в системе и среднюю задержку пакета, в случаях когда линия используется сеансами в режиме
 - 10 каналов равной пропускной способности, уплотненных по времени,
 - статистического уплотнения.
 - Выполните (а) для случая, когда в 5 сеансах передаются 250 пакетов в минуту, а в других 5 сеансах - 50 пакетов в минуту.
- 3.6. Повторно выполните п. (а) в задаче 3.5 для случая, когда длины пакетов распределены не экспоненциально, а так, что 10 % пакетов имеют длину 100 бит, а остальные 1500 бит. Решите задачу для случая, когда короткие пакеты имеют приоритет без прерывания обслуживания перед длинными пакетами.
- 3.7. Линия связи разделена на два одинаковых канала, каждый из которых обслуживает поток пакетов, в котором все пакеты имеют равное время передачи T и равные интервалы между поступлениями $R > T$. Наряду с этой линией связи рассмотрите статистическое уплотнение двух потоков путем объединения двух каналов в один канал с временем передачи $T/2$ для каждого пакета. Покажите, что среднее время пребывания пакетов в системе уменьшится от T к некоторому значению между $T/2$ и $3T/4$, тогда как дисперсия времени ожидания в очереди увеличится от 0 до $T^2/16$.
- 3.8. Пассажиры подходят к стоянке такси с местом для пяти машин в соответствии с пуассоновским процессом с интенсивностью один человек в минуту.

Пассажир садится в такси, если оно имеется, в противном случае он становится в очередь. Такси подъезжают на стоянку в соответствии с пуассоновским процессом с интенсивностью две машины в минуту. Если такси подъезжает в момент, когда стоянка заполнена, то оно отъезжает сразу, в противном случае забирает клиента (если хотя бы один ожидает) или присоединяется к очереди ожидающих такси. Найдите стационарное распределение вероятности длины очереди из такси.

- 3.9. Узел связи A принимает пуассоновский поток пакетов из двух других узлов 1 и 2 с интенсивностями λ_1 и λ_2 соответственно и передает их по линии с пропускной способностью C бит в секунду. Предполагается, что два входных потока независимы и длины их пакетов имеют одинаковые экспоненциальные распределения со средним значением L бит. Пакет из узла 1 всегда принимается узлом A . Пакет из узла 2 принимается только в том случае, если число пакетов в A (в очереди или в процессе передачи) меньше, чем заданное число $K > 0$; в противном случае пакет считается потерянным.
- (а) Найдите диапазон значений λ_1 и λ_2 , для которого математическое ожидание числа пакетов в A с течением времени будет оставаться ограниченным.
- (б) Для λ_1 и λ_2 в диапазоне из п. (а) найдите стационарную вероятность пребывания n пакетов в узле A ($0 \leq n < \infty$). Найдите среднее время, которое необходимо для того, чтобы пакет из источника 1 покинул узел A при условии, что он попал в A , и среднее число пакетов в A из источника 1. Повторите рассмотрение для пакетов из источника 2.
- 3.10. (а) Выведите уравнения (3.11)–(3.14).
- (б) Покажите, что если моменты поступления в двух непересекающихся временных интервалах являются независимыми и имеют пуассоновские распределения с параметрами $\lambda\tau_1$, $\lambda\tau_2$, то число поступлений на объединенный интервал имеет пуассоновское распределение с параметром $\lambda(\tau_1 + \tau_2)$.
- (в) Покажите, что если k независимых пуассоновских процессов A_1, \dots, A_k сливаются в один процесс $A = A_1 + A_2 + \dots + A_k$, то A — пуассоновский процесс с интенсивностью λ , равной сумме интенсивностей $\lambda_1, \dots, \lambda_k$ процессов A_1, \dots, A_k . Также покажите, что вероятность того, что первое поступление общего процесса принадлежит процессу A_1 , равна λ_1/λ независимо от момента, когда произошло это поступление.
- (г) Предположим, что на отрезке $[t_1, t_2]$ происходит только одно поступление пуассоновского процесса. Покажите, что при этом условии момент поступления равномерно распределен на отрезке $[t_1, t_2]$.
- 3.11. Пакеты поступают в передающее устройство в соответствии с пуассоновским процессом с интенсивностью λ . Каждый пакет независимо и с вероятностью p поступает в одну из передающих линий, а с вероятностью $(1 - p)$ в другую. Покажите, что процессы поступления в эти две передающие линии пуассоновские с интенсивностями λp и $\lambda(1 - p)$ соответственно и что эти два процесса независимы.
- 3.12. Рассмотрите систему, аналогичную системе $M/M/1$, за исключением того, что, когда система становится пустой, обслуживание не начинается снова, пока в систему не поступят k требований (k задано). Как только обслуживание возобновляется, оно продолжается нормально, пока система опять не станет пустой. Найдите стационарную вероятность числа требований в системе, среднее число требований в системе и среднюю задержку требования.
- 3.13. Фирма, устанавливающая прямую телефонную связь между двумя городами, предполагает пуассоновский поток с интенсивностью 30 звонков в минуту. Продолжительности разговоров независимы и экспоненциально распределены со средним 3 минуты. Интервалы между поступлениями не зависят от длительностей разговоров. Сколько каналов должна обеспечить фирма, для того чтобы гарантировать, что попытка звонка будет заблокирована (из-за того, что все каналы заняты) с вероятностью меньше

чем 0,01? Предполагается, что блокировка звонка ведет к его потере, т. е. заблокированный звонок не возобновляется.

- 3.14. Рассмотрите систему $M/M/\infty$ с обслуживающими приборами 1, 2, Дополнительное ограничение состоит в том, что поступившее требование будет направляться в свободный обслуживающий прибор с наименьшим номером. Найдите долю времени, когда каждый обслуживающий прибор занят. Изменится ли ответ, если число обслуживающих приборов конечно? *Указание.* Докажите, что в стационарном состоянии вероятность того, что все первые m обслуживающих приборов заняты, определяется согласно В-формуле Эрланга для системы $M/M/m/m$. Найдите суммарную интенсивность поступления в обслуживающие приборы с номерами $(m+1)$ и выше, а затем интенсивность поступления требований в каждый прибор.

- 3.15. Для системы $M/G/1$ покажите, что

$$P \{\text{система пуста}\} = 1 - \lambda \bar{X},$$

$$\text{средняя длина между периодами занятости} = 1/\lambda,$$

$$\text{средняя длина периода занятости} = \frac{\bar{X}}{1 - \lambda \bar{X}},$$

$$\text{среднее число требований, обслуженных за период занятости} = \frac{1}{1 - \lambda \bar{X}}.$$

Рассмотрите следующее доказательство. В момент поступления требования вероятность того, что другое требование обслуживается, равна $\lambda \bar{X}$. Так как обслуживаемое требование имеет среднее время обслуживания \bar{X} , то среднее время, необходимое для того, чтобы завершить обслуживание, равно $\bar{X}/2$. Поэтому среднее оставшееся время обслуживания равно $\lambda \bar{X}^2/2$. Что неправильно в этом доказательстве?

- 3.16. Система $M/G/1$ с произвольным порядком обслуживания. Рассмотрите систему $M/G/1$ с той разницей, что требования обслуживаются не в порядке их поступления. В момент завершения обслуживания требования одно из требований, ожидающих в очереди, выбирается в соответствии с некоторым правилом и обслуживается следующим. Покажите, что формула Поллачека—Хинчина для среднего времени ожидания в очереди W остается верной при условии, что очередность поступления требований не зависит от длительностей обслуживания требований, ожидающих в очереди. *Указание.* Докажите, что это предположение о независимости означает, что в любой момент t число требований, ожидающих в очереди, $N_Q(t)$ не зависит от длительностей обслуживания этих требований. Покажите, что это в свою очередь означает, что $U = R + \rho W$, где R — среднее оставшееся время, а U — среднее стационарное время до окончания работы в системе (общее оставшееся время обслуживания требований, находящихся в системе). Докажите, что U и R не зависят от порядка обслуживания требований.

- 3.17. Системы с приоритетами и несколькими обслуживающими приборами. Рассмотрите системы из подразд. 3.5.3. Все классы приоритетов имеют экспоненциально распределенные длительности обслуживания со средним значением $1/\mu$. Предполагается, что используется m обслуживающих приборов.

- (а) Рассмотрите систему без прерывания обслуживания. Покажите, что равенство (3.79) дает средние времена ожидания в очереди со средним остаточным временем R , определяемым формулой

$$R = \frac{P_Q}{m\mu},$$

где P_Q — стационарная вероятность того, что требование направляется в очередь. Ее можно найти по формуле Эрланга (равенство (3.36)).

(Здесь $\rho_i = \lambda_i/(m\mu)$, а $\rho = \sum_{i=1}^n \rho_i$.)

- (б) Рассмотрите систему с прерыванием и дообслуживанием. Напишите формулу для $W_{(k)}$ — среднего времени ожидания в очереди, усредненного по первым k приоритетам. Используя теорему Литтла, покажите, что среднее время ожидания в очереди требования k -го приоритета может быть получено рекуррентно из формул

$$W_1 = W_{(1)},$$

$$W_k = \frac{1}{\lambda_k} \left[W_k \sum_{i=1}^k \lambda_i - W_{(k-1)} \sum_{i=1}^{k-1} \lambda_i \right], \quad k = 2, 3, \dots, n.$$

- 3.18. Рассмотрите систему массового обслуживания с приоритетами без прерывания обслуживания из подразд. 3.5.3 для случая, когда имеющаяся пропускная способность достаточна для того, чтобы передавать поток с наибольшим приоритетом, но недостаточна для обслуживания потока всех приоритетов, т. е.

$$\rho_1 < 1 < \rho_1 + \rho_2 + \dots + \rho_n.$$

Найдите среднее время задержки требования каждого приоритета. *Указание.* Определите интенсивность ухода из системы требований наибольшего приоритета, которые будут иметь бесконечную среднюю задержку и бесконечное среднее оставшееся время обслуживания.

- 3.19. Рассмотрите систему без прерывания обслуживания с n классами приоритетов.

- (а) Покажите, что сумма $\sum_{k=1}^n \rho_k W_k$ не зависит от упорядочения классов приоритетов и на самом деле

$$\sum_{k=1}^n \rho_k W_k = \frac{R\rho}{1-\rho},$$

где $\rho = \rho_1 + \rho_2 + \dots + \rho_n$ (этот результат известен как закон сохранения в системе $M/G/1$ [141]). *Указание.* Используйте равенство (3.79).

Кроме того, докажите, что $U = R + \sum_{k=1}^n \rho_k W_k$, где U — среднее стационарное время для завершения работы в системе (общее время обслуживания требований, оставшихся в системе), и что U и R не зависят от упорядочения классов приоритетов.

- (б) Пусть c_k — стоимость единицы времени для каждого класса требований k , ожидающих в очереди. Покажите, что

$$\frac{\bar{X}_1}{c_1} \leq \frac{\bar{X}_2}{c_2} \leq \dots \leq \frac{\bar{X}_n}{c_n}.$$

Указание. Выразите стоимость в виде $\sum_{k=1}^n (c_k/\bar{X}_k) (\rho_k W_k)$ и используйте п. (а). Используйте также то, что перестановка любых двух соседних классов сохраняет время ожидания всех остальных классов неизменным.

- 3.20. Система $M/M/1$ с разделением обслуживания. Рассмотрим систему, аналогичную системе $M/M/1$, за исключением того, что всякий раз, когда в системе имеется n требований, они все обслуживаются одновременно с одинаковой скоростью, равной $1/n$ требований в единицу времени. Покажите, что стационарное распределение числа требований в системе такое же, как в системе $M/M/1$. *Замечание.* Можно показать, что стационарное распределение числа требований в системе такое же, как в системе $M/M/1$, даже если распределение времени обслуживания не является экспоненциальным, т. е. в системе типа $M/G/1$ ([204], с. 171).

3.21. В примере 10 подразд. 3.4.2 докажите справедливость формулы $\sigma_f^2 = (\lambda/\mu)^{1/2} s_r$. *Указание.* Учтите, что

$$E \{f^2\} = E \left\{ \left(\sum_{i=1}^n \gamma_i \right)^2 \right\} = E \left\{ E \left\{ \left(\sum_{i=1}^n \gamma_i \right)^2 \middle| n \right\} \right\},$$

и используйте то, что величина n имеет пуассоновское распределение.

3.22. Покажите, что равенство (3.59) для средней задержки системы с временным уплотнением может быть получено как частный случай результатов для системы с ограниченным обслуживанием и резервированием. Обобщите выражение (3.59) на случай, когда длины окон являются случайными независимыми величинами, а потоки имеют неравные интенсивности, соответствующие распределениям длин окон. *Указание.* Рассмотрите систему с отсечением при длине пакета, равной нулю.

3.23. Покажите, что математические ожидания длины циклов для системы с резервированием в случае одного пользователя и в случае многих пользователей соответственно равны $\bar{V}/(1 - \rho)$ и $m\bar{V}/(1 - \rho)$. *Указание.* Покажите, что если L_k — длина k -го цикла, то $E \{L_{k+1} | L_k\} = \bar{V} + \rho L_k$ в случае одного пользователя.

3.24. Рассмотрите системы с ограниченным обслуживанием и резервированием. Покажите, что как для системы с отсечением, так и для системы с частичным отсечением

(а) стационарная вероятность поступления пакета в течение интервала резервирования равна $1 - \rho$;

(б) стационарная вероятность того, что за интервалом резервирования будет следовать пустой интервал данных, равна $(1 - \rho - \lambda\bar{V})/(1 - \rho)$. *Указание.* Если p — требуемая вероятность, докажите, что отношение продолжительностей интервалов передачи данных и интервалов резервирования равно $(1 - p) \bar{X}/\bar{V}$.

3.25. Система с ограниченным обслуживанием, резервированием и уплотнением интервалов резервирования и данных. Рассмотрите систему с отсечением, резервированием и ограниченным обслуживанием с тем отличием, что m пользователей уплотняют интервалы резервирования и данных, т. е. все пользователи резервируют свою передачу в одном и том же интервале и передают не более одного пакета каждый в последующем интервале данных. Покажите, что

$$W = \frac{\lambda \bar{X}^2}{2(1 - \rho - \lambda\bar{V}/m)} + \frac{(1 - \rho) \bar{V}^2}{2(1 - \rho - \lambda\bar{V}/m) \bar{V}} + \frac{(1 - \rho\alpha - \lambda\bar{V}/m) \bar{V}}{1 - \rho - \lambda\bar{V}/m},$$

где \bar{V} и \bar{V}^2 — первые два момента интервала резервирования, а α удовлетворяет неравенствам

$$\frac{\bar{K} + (\hat{K} - 1)(2\bar{K} - \hat{K})}{2m\bar{K}} - \frac{1}{2m} \leq \alpha \leq \frac{1}{2} - \frac{1}{2m},$$

где $\bar{K} = \frac{\lambda\bar{V}}{1 - \rho}$ есть среднее число пакетов, переданных за интервал данных,

а \hat{K} — наименьшее целое число, большее \bar{K} . Установите, что формула для W становится точной при $\rho \rightarrow 0$ (слабая нагрузка) и $\rho \rightarrow 1 - \lambda\bar{V}/m$ (большая нагрузка). *Указание.* Покажите, что

$$W = R + \lambda W + \left(1 + \frac{\lambda W}{m} - s\right) \bar{V},$$

где $S = \lim_{i \rightarrow \infty} E \{S_i\}$, а S_i — число (равное 0 или 1) пакетов пользователя, которому принадлежит пакет i ; эти пакеты будут передаваться между моментом поступления пакета i и моментами окончания цикла, в котором поступает пакет i . Попробуйте получить границы для S путем отдельного рассмотрения случаев, когда пакет i поступает в интервале резервирования и когда он поступает в интервале данных.

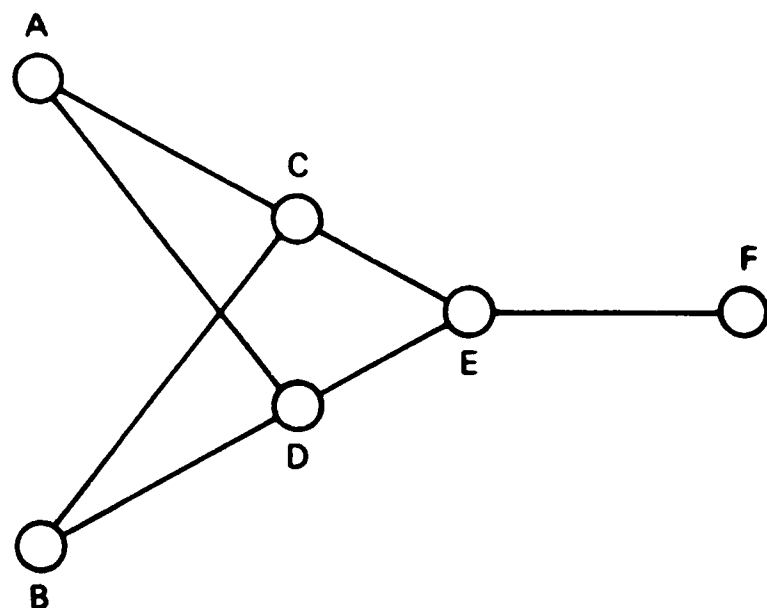


Рис. 3.27.

- 3.26. Рассмотрите сеть, представленную на рис. 3.27. Имеются четыре сеанса: ACE, ADE, BCEF и BDEF, по которым передается пуассоновский поток с интенсивностями соответственно 100, 200, 500 и 600 пакетов в минуту. Длины пакетов распределены экспоненциально со средним 1000 бит. Все линии передачи имеют пропускную способность, равную 50 кбит/с; задержка распространения в каждой линии равна 2 мс. Используя аппроксимацию Клейнрока, найдите среднее число пакетов в системе, среднюю задержку пакета (независимую от сеанса) и среднюю задержку пакета в каждом сеансе.
- 3.27. Границы для пропускной способности замкнутой сети очередей. Пакеты входят в точке A в сеть линий связи, показанную на рис. 3.28, а выходят в точке B . Пакет сначала передается по одной из линий L_1, \dots, L_K (для этой передачи требуется среднее время \bar{X}), а потом передается по линии L_{K+1} (для этого требуется среднее время \bar{Y}). В результате управления потоком максимум $N \geq K$ пакетов допускаются в систему. Каждый раз, когда пакет выходит из системы в точке B , подтверждение передается обратно и достигает точки A после фиксированного интервала времени \bar{Z} . В этот момент новому пакету разрешается войти в систему. Используя

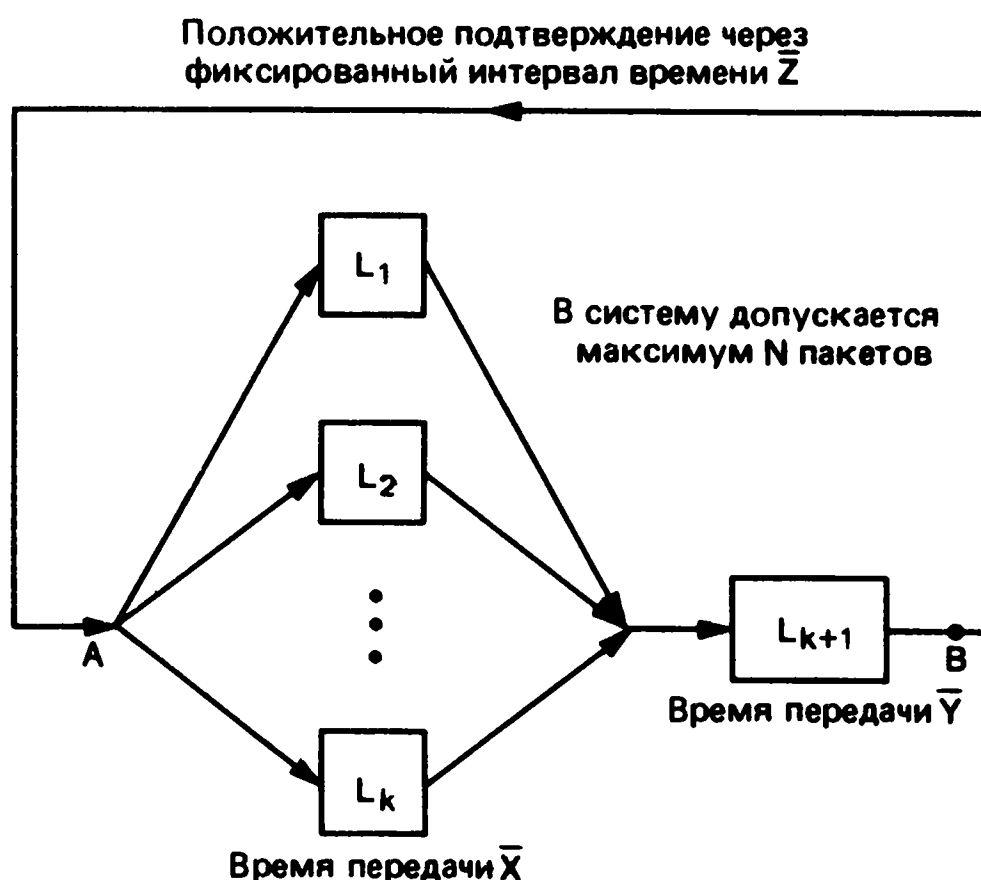


Рис. 3.28.

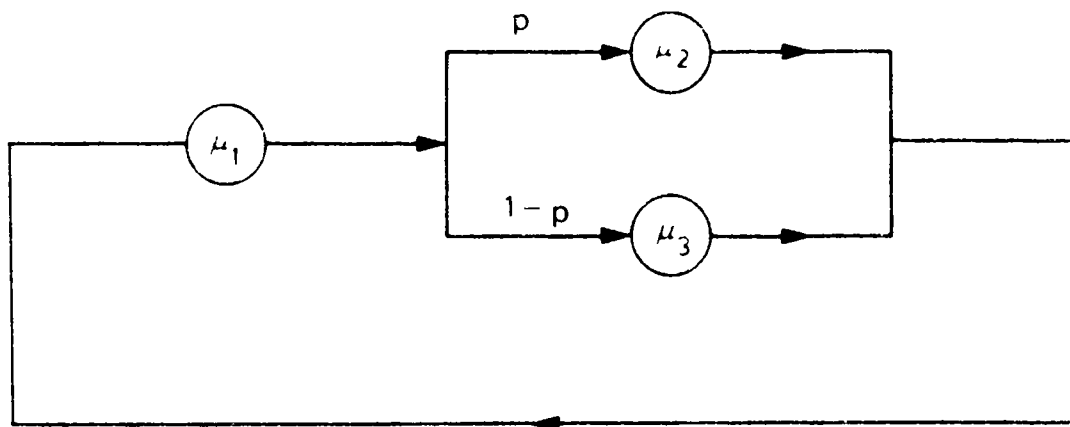


Рис. 3.29.

теорему Литтла, найдите нижнюю и верхнюю границы для пропускной способности системы при следующих двух условиях.

(а) Способ, по которому пакет направляется в одну из линий L_1, \dots, L_K , не определен.

(б) Способ, по которому пакет направляется в линию, такой, что если одна из линий L_1, \dots, L_K свободна, то нет пакета, ожидающего передачи по какой-либо другой линии.

3.28. Анализ замкнутой сети очередей. Определенные системы лучше описываются замкнутыми сетями очередей, в которых число требований в системе фиксировано. Рассмотрим замкнутую сеть очередей с тремя требованиями, которая показана на рис. 3.29, вместе с вероятностями того, что требование, покидающее очередь, поступает в другую очередь ¹⁾.

(а) Как бы вы определили состояние этой системы? Нарисуйте диаграмму переходов и нанесите на нее состояния и интенсивности переходов.

(б) Покажите, что система является обратимой по времени и что вероятности пребывания в системе определенного числа требований имеют форму произведения

$$P(n_1, n_2, n_3) = \alpha \left(\frac{1}{\mu_1} \right)^{n_1} \left(\frac{p}{\mu_2} \right)^{n_2} \left(\frac{1-p}{\mu_3} \right)^{n_3},$$

где $n_1 + n_2 + n_3 = 3$, а α — нормирующий множитель.

3.29. Рассмотрите две очереди с независимыми пуассоновскими потоками моментов поступлений и независимыми экспоненциально распределенными длительностями обслуживания. Интенсивности поступления и скорости обслуживания обозначаются соответственно через λ_i, μ_i при $i = 1, 2$. Две очереди совместно используют места для ожидания с общим их числом B (включая требования, находящиеся на обслуживании). Требование, поступившее в систему и обнаружившее, что все места для ожидания заняты, теряется. Покажите, что при $m + n \leq B$ стационарные вероятности равны

$$P\{m \text{ в очереди } 1, n \text{ в очереди } 2\} = c \rho_1^m \rho_2^n,$$

где $\rho_i = \lambda_i / \mu_i$, а c — нормирующий множитель.

3.30. Рассмотрите систему $M/M/1/m$, которая подобна системе $M/M/1$ за исключением того, что в ней не может быть более m требований, а требование, поступающее в момент, когда система полностью заполнена, теряется.

(а) Получите стационарные вероятности числа требований в системе.

(б) Покажите, что система является обратимой по времени, и охарактеризуйте процесс уходов.

3.31. Теорема Литтла при произвольном порядке обслуживания. Используя рис. 3.30, докажите теорему Литтла для систем, в которых порядок об-

¹⁾ См. примечание на с. 209. — Прим. ред.

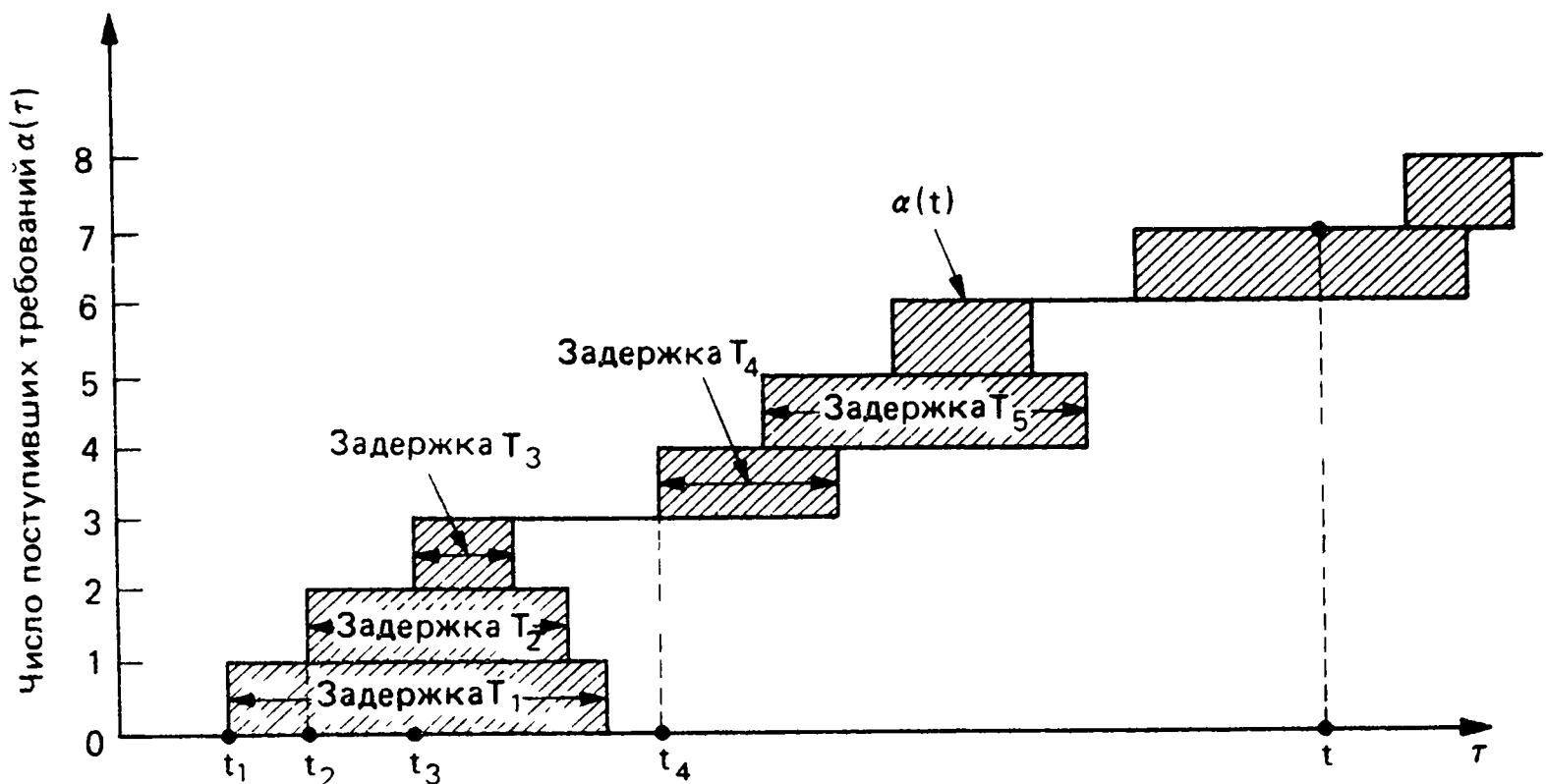


Рис. 3.30.

обслуживания требований является произвольным, включая случаи, когда обслуживающие приборы могут коллективно использоваться несколькими требованиями и когда обслуживание требования может быть прервано для обслуживания требования более высокого приоритета. На этом рисунке t_i — момент поступления i -го требования, а T_i — время пребывания этого требования в системе, т. е. время между моментом поступления требования и моментом его ухода из системы. *Указание.* Вычислите площадь заштрихованной области до момента t двумя различными способами.

3.32. Теорема Литтла при произвольном порядке обслуживания; аналитическое доказательство. Рассмотрите анализ теоремы Литтла, проведенный в разд. 3.2, и обозначения, введенные там. Предположим, что средние по времени интенсивности поступления и ухода требований существуют и равны

$$\lambda = \lim_{k \rightarrow \infty} \alpha(t)/t = \lim_{t \rightarrow \infty} \beta(t)/t$$

и что существует предел, с помощью которого определяется среднее по времени время пребывания требования в системе

$$T = \lim_{k \rightarrow \infty} \frac{1}{k} \sum_{i=1}^k T_i.$$

Покажите, что независимо от порядка обслуживания требований теорема Литтла ($N = \lambda T$) справедлива с

$$N = \lim_{t \rightarrow \infty} \frac{1}{t} \int_0^t N(\tau) d\tau.$$

Указание. Переходя ниже к пределу при $t \rightarrow \infty$ и обозначая через $D(t)$ множество требований, ушедших к моменту t , имеем (см. [223])

$$\frac{1}{t} \sum_{i \in D(t)} T_i \leq \frac{1}{t} \int_0^t N(\tau) d\tau \leq \frac{1}{t} \sum_{i=1}^{\alpha(t)} T_i.$$

3.33. Обобщение теоремы Литтла. Рассмотрите систему с поступающим и уходящим потоками. Пусть интенсивность поступающего потока равна λ и

поступающее требование должно платить системе деньги по некоторому правилу.

(а) Докажите, что справедливо следующее равенство:

Средняя скорость, с которой система получает деньги, равна $\lambda \cdot$ (средняя денежная сумма, которую платит требование).

(б) Покажите, что теорема Литтла является частным случаем этой ситуации.

(в) Рассмотрите систему $M/G/1$ со следующим правилом уплаты денег. Каждое требование платит со скоростью y за единицу времени, если время, оставшееся до конца его обслуживания, равно y , независимо от того, находится ли оно в очереди или на обслуживании. Покажите, что формула из п. (а) может быть переписана в виде

$$W = \lambda \left(\bar{X}W + \frac{\bar{X}^2}{2} \right);$$

это формула Поллачека—Хинчина.

3.34. *Очередь $M/G/1$ со случайным размером поступающей группы требований.* Рассмотрите систему $M/G/1$ с той разницей, что требования поступают в систему группами в соответствии с пуассоновским процессом интенсивности λ . Каждая группа состоит из n требований, где n имеет заданное распределение и не зависит от длительностей обслуживания требований. Используйте доказательство из разд. 3.5, для того чтобы показать, что время ожидания в очереди равно

$$W = \frac{\lambda n \bar{X}^2}{2(1-\rho)} + \frac{\bar{X}(\bar{n}^2 - \bar{n})}{2\bar{n}(1-\rho)}.$$

Указание. Используйте равенство $W = R + \rho W + W_B$, где W_B — среднее время ожидания требования из-за требований, которые поступили с ним в одной и той же группе.

3.35. *Очередь $M/G/1$ с избыточностью в каждом периоде занятости.* Рассмотрите очередь $M/G/1$ с тем отличием, что обслуживание первого требования в каждом периоде занятости требует дополнительное время Δ по сравнению с обычным временем обслуживания требования. Предположим, что Δ имеет заданное распределение и не зависит от всех других случайных величин в системе. Пусть $\rho = \lambda \bar{X}$ — коэффициент использования. Покажите, что

(а) $P_0 = P \{\text{система пуста}\} = (1 - \rho)/(1 + \lambda \bar{\Delta})$;

(б) средняя длина периода занятости $= (\bar{X} + \bar{\Delta})/(1 - \rho)$;

(в) среднее время ожидания в очереди равно

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{\lambda [(\bar{X} + \bar{\Delta})^2 - \bar{X}^2]}{2(1 + \lambda \bar{\Delta})}.$$

3.36. *Система $M/G/1$ с одним перерывом.* Рассмотрите систему $M/G/1$ с отличием, что каждый период занятости продолжается одним перерывом. Если этот перерыв уже завершился, требование, поступившее в пустую систему, немедленно начинает обслуживаться. Предположим, что длительности перерывов независимы, одинаково распределены и не зависят от интервалов между моментами поступления требований и длительностей обслуживания. Найдите среднее время ожидания в очереди.

3.37. *Система $M/G/\infty$.* Рассмотрите систему массового обслуживания с пуассоновским процессом поступления требований с интенсивностью λ . Имеется бесконечное число обслуживающих приборов, так что каждое поступившее требование начинает обслуживаться на пустом обслуживающем приборе сразу же после поступления. Каждый обслуживающий прибор имеет одинаковое для всех требований распределение длительности обслужива-

ния и $F_X(x) = P\{X \leq x\}$ обозначает вероятность того, что обслуживание, которое началось в момент τ , завершится к моменту $\tau + x$ ($F_X(x) = 0$ при $x \leq 0$). Обслуживающие приборы имеют независимые и одинаково распределенные длительности обслуживания.

- (а) При сколь угодно малых x и δ ($0 < \delta < x$) найдите вероятность того, что на отрезке $[\tau - x, \tau - x + \delta]$ поступило требование и что это требование продолжает обслуживаться в момент τ .
- (б) Покажите, что среднее время обслуживания требования, поступившего в произвольный момент времени, равно

$$\bar{X} = \int_0^{\infty} [1 - F_X(x)] dx.$$

Указание. Используйте графическое доказательство.

- (в) С помощью (а) и (б) покажите, что число требований в системе имеет пуассоновское распределение со средним $\lambda \bar{X}$.

Приложение А

Краткий обзор теории цепей Маркова

Цель этого приложения состоит в том, чтобы дать краткую сводку необходимых результатов по теории цепей Маркова с дискретным и непрерывным временем. Для более подробного изучения мы отсылаем читателя к книгам по случайным процессам.

3А.1. Цепи Маркова с дискретным временем

Рассмотрим дискретный по времени случайный процесс $\{X_n | n = 0, 1, 2, \dots\}$, который принимает значения из множества неотрицательных целых чисел; таким образом, $i = 0, 1, \dots$ — состояния, в которых может находиться процесс. Процесс называется *цепью Маркова*, если всякий раз, когда он находится в состоянии i , имеется фиксированная вероятность P_{ij} того, что в следующий момент он будет в состоянии j независимо от истории процесса до момента поступления в состояние i , т. е. для любых $n \geq 0$, $i_{n-1}, \dots, i_0, i, j$

$$\begin{aligned} P_{ij} &= P\{X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, \dots, X_0 = i_0\} = \\ &= P\{X_{n+1} = j | X_n = i\}. \end{aligned}$$

Вероятности P_{ij} называются *переходными вероятностями*. Они должны удовлетворять условиям

$$P_{ij} \geq 0, \quad \sum_{j=0}^{\infty} P_{ij} = 1, \quad i = 0, 1, \dots$$

Соответствующая матрица переходных вероятностей обозначается как

$$P = \begin{bmatrix} P_{00} & P_{01} & P_{02} & \dots \\ P_{10} & P_{11} & P_{12} & \dots \\ \dots & \dots & \dots & \dots \\ P_{i0} & P_{i1} & P_{i2} & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix}.$$

Рассмотрим переходные вероятности за n шагов

$$P_{ij}^n = P \{X_{n+m} = j \mid X_m = i\}, \quad n \geq 0, \quad i, j \geq 0.$$

Уравнения Чепмена—Колмогорова, которые позволяют вычислить P_{ij}^n , имеют вид

$$P_{ij}^{n+m} = \sum_{k=0}^{\infty} P_{ik}^n P_{kj}^m, \quad n, m \geq 0, \quad i, j \geq 0.$$

Из этих уравнений видно, что P_{ij}^n является элементом матрицы P^n (матрицы переходных вероятностей P , возведенной в степень n).

Говорят, что два состояния i и j сообщающиеся, если $P_{ij}^n > 0$, $P_{ji}^{n'} > 0$ для некоторых n и n' . Если все состояния сообщающиеся, то говорят, что цепь Маркова является *неприводимой*. Цепь Маркова является *апериодической*, если для каждого состояния i не существует целого $d \geq 2$, такого, что $P_{ii}^n = 0$, за исключением случаев, когда n кратно d . Распределение вероятностей $\{p_j \mid j \geq 0\}$ называется *стационарным распределением* цепи Маркова, если

$$p_j = \sum_{i=0}^{\infty} p_i P_{ij}, \quad j \geq 0. \quad (3A.1)$$

Ограничимся рассмотрением неприводимых и апериодических цепей Маркова, так как нам будут встречаться только такие цепи. Для такой цепи введем обозначение

$$p_j = \lim_{n \rightarrow \infty} P_{jj}^n, \quad j \geq 0.$$

Можно показать, что указанный предел существует и если $p_j > 0$, то $1/p_j$ равно *среднему времени возвращения в состояние j* , т. е. математическому ожиданию числа переходов между двумя последовательными посещениями состояния j . Если $p_j = 0$, среднее время возвращения равно бесконечности. Другая интерпретация состоит в том, что p_j равно средней доле времени, в течение которого процесс пребывает в состоянии j . Следующий результат имеет первостепенное значение.

Теорема. Для неприводимой апериодической цепи Маркова имеются две возможности:

1) $p_j = 0$ при всех $j \geq 0$, в этом случае цепь не имеет стационарного распределения;

2) $p_j > 0$ при всех $j \geq 0$; в этом случае $\{p_j \mid j \geq 0\}$ — единственное стационарное распределение цепи.

Типичный пример приведенного первого случая — система массового обслуживания $M/M/1$, в которой интенсивность поступления требований λ превышает скорость обслуживания μ .

Во втором случае возникает вопрос о нахождении стационарного распределения $\{p_j \mid j \geq 0\}$. Для систем массового обслуживания часто оказывается полезным следующий метод. Умножая

уравнение $P_{jj} + \sum_{\substack{i=0 \\ i \neq j}}^{\infty} P_{ji} = 1$ на p_j и используя формулу (3А.1), получаем уравнения

$$p_j \sum_{\substack{i=0 \\ i \neq j}}^{\infty} P_{ji} = \sum_{\substack{i=0 \\ i \neq j}}^{\infty} p_i P_{ij}, \quad (3A.2)$$

которые известны как *уравнения глобального баланса*. Согласно этим уравнениям, в равновесии вероятность перехода из j (левая часть уравнения (3А.2)) равна вероятности перехода в j (правая часть уравнения (3А.2)).

Уравнения глобального баланса могут быть обобщены на целое множество состояний. Рассмотрим подмножество состояний S . Суммируя (3А.2) по всем $j \in S$, получаем

$$\sum_{j \in S} p_j \sum_{i \notin S} P_{ji} = \sum_{i \notin S} p_i \sum_{j \in S} P_{ij}; \quad (3A.3)$$

это означает, что *вероятность перехода из множества состояний S равна вероятности перехода в множество S* .

Интуитивное объяснение этих уравнений основывается на том, что, если цепь Маркова неприводима, состояние (с вероятностью единица) будет возвращаться в множество S бесконечное число раз. Следовательно, для каждого перехода из S должен быть (с вероятностью единица) обратный переход в S в некоторый последующий момент. В результате доля переходов из S (среди всех переходов) равна доле переходов в S . Это и есть точный смысл уравнений глобального баланса (3А.3).

3А.2. Уравнения детального баланса

В качестве применения уравнений глобального баланса рассмотрим цепь Маркова, типичную для систем массового обслуживания и более общих систем, таких как системы рождения и ги-

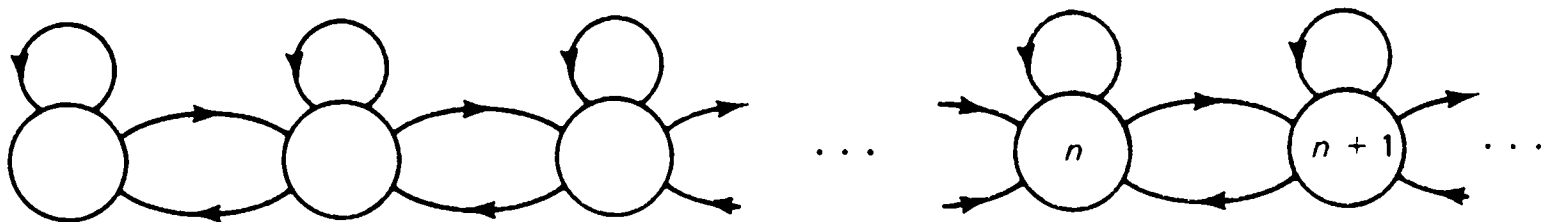


Рис. 3А.1. Диаграмма переходных вероятностей процесса рождения и гибели.

бели, в которых два последовательных состояния могут отличаться только на единицу, как показано на рис. 3А.1. Предположим, что $P_{i, i+1} > 0$ и $P_{i+1, i} > 0$ для любого i . Это необходимое и достаточное условие неприводимости цепи. Рассмотрим множество состояний

$$S = \{0, 1, \dots, n\}.$$

Применяя уравнение (3А.3), получаем

$$p_n P_{n, n+1} = p_{n+1} P_{n+1, n}, \quad n = 0, 1, \dots, \quad (3А.4)$$

т. е. в стационарном состоянии вероятность перехода из n в $n + 1$ равна вероятности перехода из $n + 1$ в n . Эти уравнения могут быть очень полезны при вычислении стационарного распределения $\{p_j \mid j \geq 0\}$ (см. разделы 3.3 и 3.4).

Уравнение (3А.4) является частным случаем уравнений

$$p_j P_{ji} = p_i P_{ij}, \quad i, j \geq 0, \quad (3А.5)$$

которые известны как *уравнения детального баланса*. Эти уравнения не обязательно справедливы для любой цепи Маркова. Однако во многих частных случаях они справедливы и сильно упрощают вычисление стационарного распределения. При доказательстве справедливости уравнений детального баланса для данной неприводимой апериодической цепи Маркова предполагается, что они справедливы, затем делается попытка решить их и найти стационарные вероятности $p_j, j \geq 0$. Имеются две возможности: либо система (3А.5) вместе с условием $\sum_j p_j = 1$ несовместна, либо распределение $\{p_j \mid j \geq 0\}$, удовлетворяющее уравнению (3А.5), будет найдено. В последнем случае ясно, что это распределение также будет удовлетворять уравнениям глобального баланса (3А.2). Эти уравнения эквивалентны условию

$$p_j = \sum_{i=0}^{\infty} p_i P_{ij}, \quad j = 0, 1, \dots,$$

поэтому, согласно приведенной ранее теореме, распределение $\{p_j \mid j \geq 0\}$ является единственным стационарным распределением.

3А.3. Уравнения частичного баланса

Некоторые цепи Маркова обладают тем свойством, что их стационарное распределение $\{p_j \mid j \geq 0\}$ удовлетворяет системе уравнений, которые занимают положение, промежуточное между уравнениями глобального и детального баланса. Для каждого узла j рассмотрим разбиение S_j^1, \dots, S_j^k дополнительного к узлу j множества $\{i \mid i \geq 0, i \neq j\}$ и уравнения

$$p_j \sum_{i \in S_j^m} P_{ji} = \sum_{i \in S_j^m} p_i P_{ij}, \quad m = 1, 2, \dots, k. \quad (3A.6)$$

Уравнения указанного вида известны как система *уравнений частичного баланса*. Если распределение $\{p_j \mid j \geq 0\}$ является решением уравнений частичного баланса, то оно также является решением уравнений глобального баланса; таким образом, оно будет единственным стационарным распределением цепи. Способ, который часто оказывается полезным при решении этих уравнений, состоит в том, чтобы определить систему уравнений частичного баланса, которой удовлетворяет стационарное распределение, а затем приступить к их решению.

3А.4. Цепи Маркова с непрерывным временем (марковские процессы)

Цепь Маркова с непрерывным временем представляет собой случайный процесс $\{X(t) \mid t \geq 0\}$, который принимает значения из множества состояний $i = 0, 1, \dots$ и имеет то свойство, что в любой момент времени, когда он переходит в состояние i , имеет место следующее.

1. Время пребывания в состоянии i распределено экспоненциально с параметром ν_i . Можно рассматривать ν_i как среднюю интенсивность (в переходах за секунду) перехода из состояния i .
2. Если процесс уходит из состояния i , он переходит в состояние j с вероятностью P_{ij} , где $\sum_j P_{ij} = 1$.

Будем рассматривать цепи, для которых:

- 1) число переходов в течение любого конечного интервала времени конечно с вероятностью единица (такие цепи называются *регулярными*);
- 2) цепь Маркова с дискретным временем и переходными вероятностями P_{ij} (называемая *вложенной цепью*) неприводима.

Можно показать, что при выполнении перечисленных условий предел

$$p_j = \lim_{t \rightarrow \infty} P \{X(t) = j \mid X(0) = i\} \quad (3A.7)$$

существует и не зависит от начального состояния i . Кроме того, если вложенная цепь имеет стационарное распределение $\{\pi_j \mid j \geq 0\}$, то все стационарные вероятности p_j непрерывной цепи являются положительными и выражаются в виде

$$p_j = \frac{\pi_j/v_j}{\sum_{i=0}^{\infty} \pi_i/v_i}, \quad j = 0, 1, \dots \quad (3A.8)$$

Вероятность π_j интерпретируется здесь как доля заходов в состояние j , а p_j интерпретируется как доля времени пребывания в состоянии j при типичном режиме работы системы.

Введем обозначение для любых i и j

$$q_{ij} = v_i P_{ij}. \quad (3A.9)$$

Так как v_i — интенсивность перехода процесса из состояния i , а P_{ij} — вероятность того, что при этом он перейдет в состояние j , то q_{ij} — это интенсивность перехода процесса в состояние j из состояния i . Поэтому q_{ij} называется *интенсивностью перехода из i в j* .

Так как мы часто будем анализировать цепи Маркова с непрерывным временем, используя их варианты с дискретным временем, опишем общий способ проведения такого анализа.

При любом $\delta > 0$ рассмотрим цепь Маркова с дискретным временем $\{X_n \mid n \geq 0\}$, где

$$X_n = X(n\delta), \quad n = 0, 1, \dots$$

Ясно, что стационарное распределение цепи $\{X_n\}$ совпадает со стационарным распределением непрерывной цепи $\{p_j \mid j \geq 0\}$ (ср. с равенством (3A.7)). Переходные вероятности цепи $\{X_n \mid n \geq 0\}$ равны

$$\begin{aligned} \bar{P}_{ij} &= \delta q_{ij} + o(\delta), \quad i \neq j, \\ \bar{P}_{ii} &= 1 - \delta \sum_{j \neq i} q_{ij} + o(\delta). \end{aligned}$$

Подставляя эти выражения в уравнения глобального баланса для дискретной цепи (ср. с формулой (3A.2)) и полагая $\delta \rightarrow 0$, получаем

$$p_j \sum_{\substack{i=0 \\ i \neq j}}^{\infty} q_{ji} = \sum_{\substack{i=0 \\ i \neq j}}^{\infty} p_i q_{ij}, \quad j = 0, 1, \dots \quad (3A.10)$$

Эти уравнения являются уравнениями глобального баланса для непрерывной цепи. Аналогично уравнения детального баланса имеют вид

$$p_j q_{ji} = p_i q_{ij}, \quad i, j = 0, 1, \dots \quad (3A.11)$$

Можно также записать систему уравнений частичного баланса и попытаться решить их для распределения $\{p_j \mid j \geq 0\}$. Полученное решение будет стационарным распределением цепи с непрерывным временем.

Приложение Б

Сводка результатов

Обозначения

- p_n — стационарная вероятность пребывания n требований в системе
- λ — интенсивность поступления требований (величина, обратная среднему интервалу времени между моментами поступления)
- μ — скорость обслуживания (величина, обратная среднему времени обслуживания)
- N — среднее число требований в системе
- N_Q — среднее число требований, ожидающих в очереди
- T — среднее время пребывания требования в системе
- W — среднее время, которое требование ожидает в очереди (не включая время обслуживания)
- \bar{X} — среднее время обслуживания
- \bar{X}^2 — второй момент времени обслуживания

Теорема Литтла

$$N = \lambda T,$$

$$N_Q = \lambda W.$$

Пуассоновское распределение с параметром m

$$p_n = \frac{e^{-m} m^n}{n!}, \quad n = 0, 1, \dots$$

Среднее = дисперсии = m .

Экспоненциальное распределение с параметром λ

$$P\{\tau \leq s\} = 1 - e^{-\lambda s}, \quad s \geq 0,$$

плотность: $\rho(\tau) = \lambda e^{-\lambda \tau}$,
 среднее = $1/\lambda$,
 дисперсия = $1/\lambda^2$.

Сводка результатов по системе $M/M/1$

(а) Коэффициент использования (доля времени, когда обслу-

живающий прибор занят)

$$\rho = \lambda/\mu.$$

(б) Вероятность пребывания n требований в системе

$$p_n = \rho^n (1 - \rho), \quad n = 0, 1, \dots$$

(в) Среднее число требований в системе

$$N = \frac{\rho}{1 - \rho}.$$

(г) Среднее время пребывания требования в системе

$$T = \frac{\rho}{\lambda(1 - \rho)} = \frac{1}{\mu - \lambda}.$$

(д) Среднее число требований, ожидающих в очереди,

$$N_Q = \frac{\rho^2}{1 - \rho}.$$

(е) Среднее время, в течение которого требование ожидает в очереди,

$$W = \frac{\rho}{\mu - \lambda}.$$

Сводка результатов по системе $M/M/m$

(а) Отношение интенсивности поступления к максимальной скорости обслуживания системы

$$\rho = \frac{\lambda}{m\mu}.$$

(б) Вероятность пребывания n требований в системе

$$p_0 = \left[\sum_{k=0}^{m-1} \frac{(m\rho)^k}{k!} + \frac{(m\rho)^m}{m!(1 - \rho)} \right]^{-1}, \quad n = 0,$$

$$p_n = \begin{cases} p_0 \frac{(m\rho)^n}{n!}, & n \leq m, \\ p_0 \frac{m^m \rho^n}{m!}, & n > m. \end{cases}$$

(в) Вероятность того, что вновь поступившее требование должно ждать в очереди (в системе m или более требований)

$$P_Q = \frac{p_0 (m\rho)^m}{m!(1 - \rho)} \quad (\text{С-формула Эрланга}).$$

(г) Среднее время, в течение которого требование ожидает в очереди,

$$W = \frac{\rho P_Q}{\lambda (1 - \rho)}.$$

(д) Среднее число требований, ожидающих в очереди,

$$N_Q = \frac{\rho P_Q}{1 - \rho}.$$

(е) Среднее время пребывания требования в системе

$$T = \frac{1}{\mu} + W.$$

(ж) Среднее число требований в системе

$$N = m\rho + \frac{\rho P_Q}{1 - \rho}.$$

Сводка результатов по системе $M/M/m/m$

(а) Вероятность пребывания n требований в системе

$$p_0 = \left[\sum_{n=0}^m \left(\frac{\lambda}{\mu} \right)^n \frac{1}{n!} \right]^{-1},$$

$$p_n = p_0 \left(\frac{\lambda}{\mu} \right)^n \frac{1}{n!}, \quad n = 1, 2, \dots, m.$$

(б) Вероятность того, что вновь поступившее требование теряется,

$$p_m = \frac{(\lambda/\mu)^m / m!}{\sum_{n=0}^m (\lambda/\mu)^n / n!} \quad (\text{В-формула Эрланга}).$$

Сводка результатов по системе $M/G/1$

(а) Коэффициент использования

$$\rho = \lambda/\mu.$$

(б) Среднее остаточное время обслуживания

$$R = \frac{\lambda \overline{X^2}}{2}.$$

(в) Формула Поллачека—Хинчина

$$W = \frac{R}{1 - \rho} = \frac{\lambda \overline{X^2}}{2(1 - \rho)},$$

$$T = \frac{1}{\mu} + W,$$

$$N_Q = \frac{\lambda^2 \bar{X}^2}{2(1-\rho)},$$

$$N = \rho + \frac{\lambda^2 \bar{X}^2}{2(1-\rho)}.$$

(г) Формула Поллачека—Хинчина для системы $M/G/1$ с перерывами

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{\bar{V}^2}{2\bar{V}},$$

$$T = \frac{1}{\mu} + W,$$

где \bar{V} и \bar{V}^2 — первые два момента длительности перерыва.

Сводка результатов по системам с резервированием (опросом)

(а) Среднее время ожидания (система с m пользователями и неограниченным обслуживанием)

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{(m-\rho)\bar{V}}{2(1-\rho)} + \frac{\sigma_V^2}{2\bar{V}} \quad (\text{система с очищением}),$$

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{(m+\rho)\bar{V}}{2(1-\rho)} + \frac{\sigma_V^2}{2\bar{V}} \quad (\text{система с частичным отсечением}),$$

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho)} + \frac{(m+2-\rho)\bar{V}}{2(1-\rho)} + \frac{\sigma_V^2}{2\bar{V}} \quad (\text{система с отсечением}),$$

где $\rho = \lambda/\mu$, а \bar{V} и σ_V^2 — соответственно среднее и дисперсия длительностей интервалов резервирования, усредненные по всем пользователям,

$$\bar{V} = \frac{1}{m} \sum_{l=0}^{m-1} \bar{V}_l,$$

$$\sigma_V^2 = \frac{1}{m} \sum_{l=0}^{m-1} (\bar{V}_l^2 - \bar{V}_l^2).$$

(б) Среднее время ожидания (система с m пользователями и ограниченным обслуживанием)

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho-\lambda\bar{V})} + \frac{(m+\rho)\bar{V}}{2(1-\rho-\lambda\bar{V})} + \frac{\sigma_V^2(1-\rho)}{2\bar{V}(1-\rho-\lambda\bar{V})} \quad (\text{система с частичным отсечением}),$$

$$W = \frac{\lambda \bar{X}^2}{2(1-\rho-\lambda\bar{V})} + \frac{(m+2-\rho-2\lambda\bar{V})\bar{V}}{2(1-\rho-\lambda\bar{V})} + \frac{\sigma_V^2(1-\rho)}{2\bar{V}(1-\rho-\lambda\bar{V})} \quad (\text{система с отсечением}).$$

(в) Среднее время пребывания в системе

$$T = \frac{1}{\mu} + W.$$

Сводка результатов по системам с приоритетами

(а) *Приоритеты без прерывания обслуживания.* Среднее время ожидания в очереди для требований k -го приоритета

$$W_k = \frac{\sum_{i=1}^n \lambda_i \overline{X_i^2}}{2(1 - \rho_1 - \dots - \rho_{k-1})(1 - \rho_1 - \dots - \rho_k)}.$$

(б) *Приоритеты без прерывания обслуживания.* Среднее время пребывания в системе требований k -го приоритета

$$T_k = \frac{1}{\mu_k} + W_k.$$

(в) *Приоритеты с прерыванием и дообслуживанием.* Среднее время пребывания в системе требований k -го приоритета

$$T_k = \frac{(1/\mu_k)(1 - \rho_1 - \dots - \rho_k) + R_k}{(1 - \rho_1 - \dots - \rho_{k-1})(1 - \rho_1 - \dots - \rho_k)},$$

где

$$R_k = \frac{\sum_{i=1}^k \lambda_i \overline{X_i^2}}{2}.$$

4. Связь в системах с множественным доступом

4.1. Введение

Рассмотренные до сих пор подсети связи состояли из узлов, соединенных двухточечными (от точки к точке) линиями связи. Каждая такая линия может состоять из витой пары проводов, коаксиального кабеля, оптического волокна, микроволновой радиолinii и т. д. Относительно двухточечных линий всегда неявно предполагается, что принятый из линии сигнал зависит только от переданного сигнала и шума этой линии.

Существует много широко используемых передающих сред, таких как в спутниковых системах, радиовещании, многоточечных телефонных линиях, системах с шиной, имеющей много отводов, у которых принятый на одном узле сигнал зависит от сигналов, переданных двумя другими или более узлами. Обычно такой принятый сигнал является суммой претерпевших затухание сигналов, переданных множеством других узлов; эти сигналы приходят с задержками, испытывают воздействие искажений и шума. Такие среды, называемые *средами с множественным доступом*, являются основой локальных сетей (ЛС), сетей городского масштаба (ГС), спутниковых сетей и радиосетей.

Разбиение на уровни, рассмотренное в гл. 1 и 2, не совсем подходит для среды с множественным доступом. Между уровнем управления линией передачи данных (УЛПД) и модемом или физическим уровнем должен находиться дополнительный уровень, который часто называют уровнем *управления доступом к среде* (УДС). Задача этого дополнительного уровня заключается в распределении среды с множественным доступом между различными узлами. Изучая эту задачу распределения, мы увидим, что распределение функций между уровнями не так ясно, как в случае двухточечных линий. Например, обратная связь, оповещающая об ошибках передачи, относится к функции ARQ уровня УЛПД, но часто она также занимает центральное место в проблеме распределения. Аналогично многие функции маршрутизации выполняются автоматически благодаря ширококвещательной природе каналов с множественным доступом.

Вообще можно рассматривать связь в системах с множественным доступом в терминах теории массового обслуживания. Каждый узел имеет очередь пакетов, которые должны быть переданы, а канал с множественным доступом является обслуживающим

прибором. В идеальном случае обслуживающий прибор должен рассматривать все ожидающие пакеты как стоящие в одной очереди, которую нужно обслужить, используя соответствующую дисциплину обслуживания очереди. К сожалению, обслуживающий прибор не знает, какие узлы имеют пакеты; аналогично в узлах не известно о наличии пакетов в других узлах. Таким образом, интересная особенность проблемы состоит в том, что информация о состоянии очереди является распределенной.

Среди многих методов, которые были разработаны для решения этой проблемы, имеются два полярных. Одним из них является метод свободного доступа для всех, при котором из узлов обычно посылаются новые пакеты немедленно в надежде на то, что другие узлы передавать не будут. В этом случае возникает интересный вопрос о том, когда и как пакеты передаются повторно при возникновении конфликтов (интерференции). Другой полярный метод — это полное упорядочение, при котором существует некоторый порядок (например, круговой), в соответствии с которым узлы получают зарезервированные интервалы времени для использования канала. Интересными вопросами здесь являются следующие: 1) что определяет порядок распределения времени (может он быть динамическим)? 2) как долго может длиться зарезервированный интервал? 3) как узлы узнают о наступлении своей очереди?

В разд. 4.2 и 4.3 исследуется метод свободного доступа для всех в простых идеализированных условиях, что позволяет сосредоточить внимание на методах повторения передачи вступивших в конфликт пакетов. Последовательно рассматриваются все более сложные алгоритмы, которые уменьшают задержку, увеличивают предельно допустимую скорость передачи и обеспечивают устойчивую работу. В последующих разделах эти алгоритмы адаптируются для лучшего использования свойств некоторых конкретных каналов и, таким образом, еще большего уменьшения задержки и увеличения скорости передачи. Менее заинтересованный читатель может опустить подразд. 4.2.3 и разд. 4.3.

В разд. 4.4 исследуется множественный доступ с прослушиванием несущей (МДПН). В этом случае метод свободного доступа для всех изменяется; не разрешается начинать передачу пакета, если прослушивание несущей показало, что он занят. Мы убедимся, что этот класс методов является относительно простым обобщением идей, представленных в разд. 4.2 и 4.3. Ценность этих методов критически зависит от отношения времени распространения к времени передачи пакета, которое обозначается через β . Если $\beta \ll 1$, то МДПН может значительно уменьшить задержку и увеличить скорость передачи по сравнению с методами, описанными в разд. 4.2 и 4.3. При выборочном чтении можно опустить подразд. 4.4.2 и 4.4.4.

В разд. 4.5 рассматривается распределение или резервирование канала по динамическим запросам отдельных узлов. Мы начинаем с рассмотрения спутниковых каналов в подразд. 4.5.1; интересной особенностью здесь является то, что $\beta \gg 1$. Далее в подразд. 4.5.2—4.5.4 обсуждаются основные подходы, используемые в локальных сетях (ЛС). Эти подходы можно рассматривать как системы резервирования; они отличаются тем, как производится резервирование: на основе свободного доступа для всех или на основе кругового упорядочения. ЛС обычно проектируются исходя из предположения о том, что β мало, но в подразд. 4.5.5 исследуются системы с более высокой скоростью или большим географическим охватом, у которых β велико.

Наконец, в разд. 4.6 исследуется пакетная радиосвязь. В этом случае интересно, что каждый узел взаимодействует только с ограниченным подмножеством других узлов; следовательно, много узлов могут передавать одновременно без взаимных помех.

Перед детальным рассмотрением указанных направлений мы кратко обсудим некоторые наиболее широко используемые среды с множественным доступом.

4.1.1. Спутниковые каналы

В типичной системе связи с геостационарным спутником множество наземных станций может передавать сообщения общему приемнику спутника и принятые им сообщения ретранслируются наземным станциям (рис. 4.1). Такие спутники часто имеют антенны с различными направлениями излучения для разных географических зон, что позволяет производить наземный прием и ретрансляцию между зонами. Кроме того, может использоваться ЧУ (или ВУ), что дает возможность производить независимый прием от различных наземных станций, находящихся в зоне действия одного и того же направления излучения антенны.

Таким образом, спутниковый канал можно использовать как совокупность виртуальных двухточечных линий; при этом виртуальные линии создаются или с помощью диаграммы направленности антенны, или путем уплотнения. Потенциальный недостаток этого подхода аналогичен недостатку, присущему использованию одной двухточечной линии многими сеансами при помощи ЧУ или ВУ, а именно повышенная задержка и неэффективное использование среды.

В подразд. 4.5.1 мы установим, что задержку можно сократить, а коэффициент использования повысить, если делить среду с учетом поступающих требований. Это труднее, чем с учетом требований делить (т. е. статистически уплотнять) двухточечную линию, потому что наземные станции не знают текущие потребности в пере-

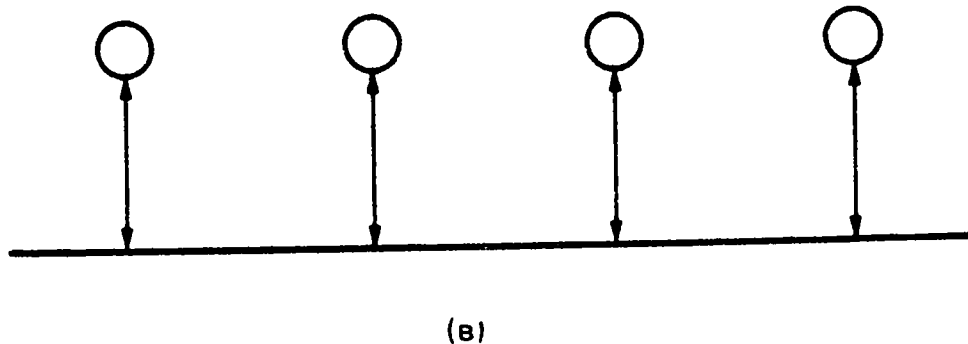
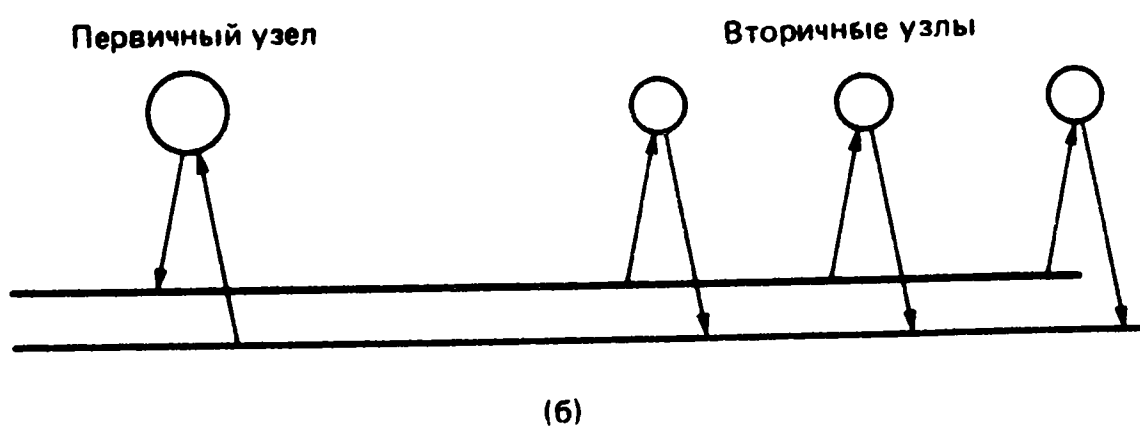
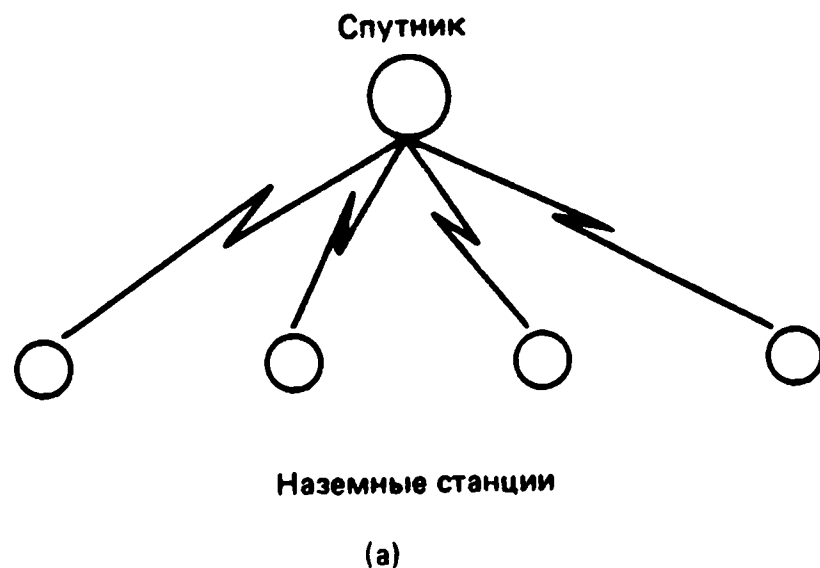


Рис. 4.1. Типичные каналы с множественным доступом.

a — спутниковый канал с множественным доступом; *б* — многоточечная телефонная линия; *в* — многоотводная шина.

даче данных на других наземных станциях. Если несколько станций передают в одно и то же время (и в одной и той же полосе частот), то их сигналы искажаются и принимаются неправильно.

4.1.2. Многоточечные телефонные линии

Другим примером канала с множественным доступом является многоточечная телефонная линия. Такие линии соединяют один первичный узел с рядом вторичных узлов; сигнал, переданный первичным узлом, распространяется по одной паре проводов и принимается всеми вторичными узлами. Аналогично существует обратная пара проводов, по которой на первичный узел поступает сумма сигналов, переданных всеми вторичными узлами. Концептуально это аналогично спутниковому каналу. Вторичные узлы

или наземные станции совместно используют линию связи с первичным узлом или спутником, тогда как первичный узел или спутник передает в широкополосном режиме всем вторичным узлам или наземным станциям.

При традиционном способе работы многоточечных телефонных линий первичный узел делает опрос, т. е. запрашивает информацию у каждого вторичного узла в некотором порядке. Каждый вторичный узел отвечает на его запрос или посылает данные первичной станции, или сообщая об отсутствии данных для передачи. Этот метод не допускает интерференции вторичных узлов, поскольку узлы опрашиваются поочередно, но он в некоторой степени недостаточно эффективен в том смысле, что тратится время на передачу запроса, а также на ожидание ответа от узла, не имеющего данных.

4.1.3. Многоотводная шина

Третьим примером канала с множественным доступом является шина с множеством отводов. В этом случае каждый узел может получить сигнал, переданный любым другим узлом, но опять-таки, если несколько узлов передают в одно и то же время, то принимается искаженный сигнал. Мы обсудим этот пример позднее при рассмотрении ETHERNET, а сейчас заметим, что концептуально этот канал в значительной мере подобен спутниковому каналу. Каждый узел может устанавливать связь с любым другим узлом, но если узлы передают одновременно, то поступающий сигнал не удастся принять безошибочно. Тот факт, что узлы в этом случае могут слышать друг друга непосредственно в отличие от прослушивания через ретрансляционный спутник, имеет некоторые важные практические следствия, но мы пока не будем их учитывать.

4.1.4. Пакетные радиосети

Четвертым примером связи в условиях множественного доступа является пакетная радиосеть. В этом случае каждый узел находится в зоне приема некоторого подмножества других узлов. Следовательно, если передает только один узел подмножества, данный узел может принять передаваемый пакет, однако если передают несколько узлов одновременно и в одной полосе частот, то прием будет искажен. Аналогично передача одного узла будет прослушиваться подмножеством других узлов. В общем случае из-за того, что шумовые условия в узлах неодинаковы, подмножество узлов, из которых данный узел может принимать, отличается от подмножества, которому данный узел может передавать. Тот факт, что приемник слышит сигналы не всех передатчиков,

а их подмножества, делает пакетную радиосвязь значительно более сложной, чем другие рассмотренные примеры. В следующих четырех разделах мы будем рассматривать такие каналы с множественным доступом, в которых приемник может слышать сигналы всех передатчиков, а затем в разд. 4.6 обсудим более детально пакетные радиосети.

4.2. Синхронный множественный доступ и система ALOHA

Спутниковые каналы, многоточечные телефонные линии и многоотводные шинные системы имеют подобные свойства, так как во всех случаях множество узлов совместно использует канал связи. Если два или более узла передают одновременно, то прием искажается, если ни один узел не передает, то канал не используется. Проблема состоит в том, чтобы упорядочить каким-либо образом использование канала с тем, чтобы точно один узел передавал в течение большей части времени. Мы начнем с рассмотрения сильно идеализированной модели. В дальнейшем мы увидим, что каналы с множественным доступом часто могут использоваться на практике гораздо более эффективно, чем это возможно в идеализированной модели, однако в этом легче разобраться с помощью нашей идеализированной модели.

4.2.1. Идеализированная модель синхронного множественного доступа

Идеализированная модель, которую предстоит построить, позволит нам сосредоточить внимание на проблеме конкуренции, которая возникает всякий раз, когда много узлов пытаются использовать канал одновременно. Концептуально мы считаем, что система, подобная показанной на рис. 4.1, *а*, состоит из m передающих узлов и одного приемника.

Если не учитывать некоторые вопросы, связанные с временем распространения, то эта модель может применяться в случае, когда m узлов могут слышать друг друга (т. е. в случае многоотводной шины). Сначала перечислим предположения модели, а затем обсудим их значение.

1. *Синхронная система.* Будем считать, что все передаваемые пакеты имеют одинаковую длину и каждый пакет требует для передачи единичный интервал времени (называемый окном). Все передатчики синхронизированы, так что прием любого пакета начинается в целочисленный момент времени и заканчивается перед следующим целочисленным моментом.

2. *Пуассоновские входные потоки.* Будем считать, что пакеты поступают на передачу в каждый из m передающих узлов в соответствии с независимыми пуассоновскими процессами. Пусть λ обозначает суммарную скорость поступления в систему, а λ/m — скорость поступления на каждый из передающих узлов.

3. *Конфликт или успешный прием.* Будем считать, что если два или более узла передают пакеты в данном временном окне, то возникает *конфликт* и приемник не получает никакой информации о содержании или отправителе переданных пакетов. Если только один узел передает пакет в данном окне, тогда пакет принимается безошибочно.

4. *Быстрая обратная связь с сигналами 0, 1, e.* Будем считать, что в конце каждого окна каждый узел получает сигнал обратной связи от приемника, который сообщает о передаче 0 пакетов, 1 пакета или более чем одного пакета (e — начальная буква слова *eggo*) в этом окне.

5. *Разрешение конфликтов.* Будем считать, что каждый пакет, попавший в конфликт, должен передаваться повторно в одном из последующих окон, причем повторные передачи продолжаются до тех пор, пока пакет не будет успешно принят. Об узле с пакетом, который должен передаваться повторно, говорят, что он имеет *задолженность*.

6а. *Отсутствие буферизации.* Если в узле уже имеется один пакет, ожидающий передачи или вступивший в конфликт с другим пакетом во время своей передачи, то новые пакеты, поступающие в этот узел, сбрасываются и никогда не передаются. Это предположение дополняется следующим.

6б. *Бесконечное число узлов ($m = \infty$).* Система имеет бесконечное множество узлов и каждый новый пакет поступает в новый узел.

Обсуждение предположений

Предположение 1 о синхронности системы имеет два следствия. Первое — это дискретность системы во времени, что упрощает анализ. Второе — это устранение (пока) возможности прослушивания несущей или раннего обнаружения конфликта. Прослушивание несущей рассматривается в разд. 4.4, а раннее обнаружение конфликта — в разд. 4.5. Оба механизма позволяют использовать канал с множественным доступом намного более эффективно, но их проще понять, рассматривая как расширение представленной модели. Синхронизация передатчиков, обеспечивающая синхронный прием, не является полностью тривиальной задачей, но ее можно установить с помощью относительно стабильных часов, небольшой по объему передаваемой информации обратной связи со стороны приемника и некоторого защитного

промежутка времени между концом передачи пакета и началом следующего окна.

Предположение 2 о пуассоновских входных потоках нереально в случае многопакетных сообщений. Мы обсудим этот случай в разд. 4.5, рассматривая его как систему, в которой узлы производят резервирование канала перед его использованием.

Предположение 3 о конфликте или успешном приеме устраняет возможность появления ошибок, обусловленных шумом, а также возможность «захватов», при которых приемник может иногда выделить одну передачу из множества других передач.

Предположение 4 о быстрой обратной связи совершенно нереально, особенно в случае спутниковых каналов. Оно принимается для упрощения анализа, и мы увидим в дальнейшем, что обратная связь с задержкой усложняет алгоритмы множественного доступа, однако не порождает никаких фундаментальных проблем. Ниже будут также рассмотрены более ограниченные типы обратной связи.

Предположение 5 о том, что вступившие в конфликт пакеты должны передаваться повторно, несомненно, необходимо для обеспечения надежной связи. В свете этого предположения неиспользование буферизации (предположение 6а) кажется довольно странным, поскольку новые пакеты, поступающие в узлы с задолженностью, сбрасываются безнаказанно, а однажды переданные пакеты должны передаваться повторно, пока не будут успешно приняты. На практике обычно осуществляется буферизация наряду с некоторой формой управления потоком для того, чтобы не очень много пакетов отвергалось узлом. В этом разделе, однако, нас интересуют каналы с множественным доступом, с большим числом узлов, относительно малой скоростью поступления λ и малой требуемой задержкой (т. е. условия, при которых ВУ действует неудовлетворительно). В этих условиях доля узлов с задолженностью обычно мала и поступлением новых пакетов в такие узлы можно пренебречь. Таким образом, задержка системы без буферизации должна быть относительно близка к задержке с буферизацией. Кроме того, задержка в системе без буферов дает нижнюю границу для задержки большого разнообразия систем с буферизацией и управляемым потоком.

Аналогично предположение 6б о бесконечном числе узлов позволяет получить верхнюю границу для задержки, которую можно достичь при конечном числе узлов. В частности, при любом заданном алгоритме множественного доступа (т. е. любом наборе правил, которым пользуется каждый узел для определения моментов передачи пакета) каждый узел из конечного множества узлов может действовать как множество виртуальных узлов, по одному на каждый возникший пакет. Если заданный алгоритм применяется независимо к каждому такому пакету, то ситуация

эквивалентна той, которая имеет место при бесконечном числе узлов, т. е. при выполнении предположения бб. При таком подходе узел с несколькими задолженными пакетами будет иногда передавать несколько пакетов в одном окне, вызывая неизбежный конфликт. Таким образом, избавляясь от таких неизбежных конфликтов и зная число пакетов в буфере узла, в системе с конечным числом узлов и буферизацией можно бы достичь меньших задержек, чем при $m = \infty$; однако в любом случае задержка системы с $m = \infty$ всегда может быть достигнута.

Если характеристики работы при выполнении предположения ба аналогичны тем, которые имеют место при выполнении предположения бб, то мы уверены в том, что имеем хорошее приближение к характеристикам системы при произвольных предположениях о буферизации. Что касается теоретического анализа, то предположение бб выделяет сущность связи в условиях множественного доступа значительно сильнее, чем ба. Мы используем ба, во-первых, потому что оно менее абстрактно, и, во-вторых, потому что оно приводит к некоторым важным заключениям относительно соотношения между ВУ и другими алгоритмами.

4.2.2. Синхронная АЛОНА

Сеть АЛОНА была разработана примерно в 1970 г. для обеспечения радиосвязи между центральной ЭВМ и различными терминалами данных, установленными в подразделениях Гавайского университета. Этот подход с использованием множественного доступа будет описан в разд. 4.2.4, а сначала мы обсудим улучшенный вариант, называемый синхронная АЛОНА [200]. Основная идея этого алгоритма состоит в том, что каждый узел, не имеющий задолженности, просто передает новый поступивший пакет в первом окне после момента прихода, рискуя, таким образом, попасть в случайные конфликты, но это приводит к очень маленькой задержке, если конфликты возникают редко. Этот подход следует сравнивать с ВУ, при котором поступивший на один из m узлов пакет должен ожидать своей очереди на передачу в среднем $m/2$ окон. Таким образом, при синхронной системе АЛОНА пакеты передаются почти немедленно, при этом иногда попадая в конфликты, тогда как при ВУ конфликты не возникают за счет больших задержек.

Когда в синхронной системе АЛОНА возникает конфликт, в каждом узле, который передавал один из вступивших в конфликт пакетов, обнаруживается конфликт в конце окна и он становится задолжником. Если бы каждый узел с задолженностью просто повторял передачу в следующем окне, после того как оказался вовлеченным в конфликт, неизбежно возникал бы следующий конфликт. Вместо этого такие узлы пропускают некоторое случайное число окон перед повторной передачей.

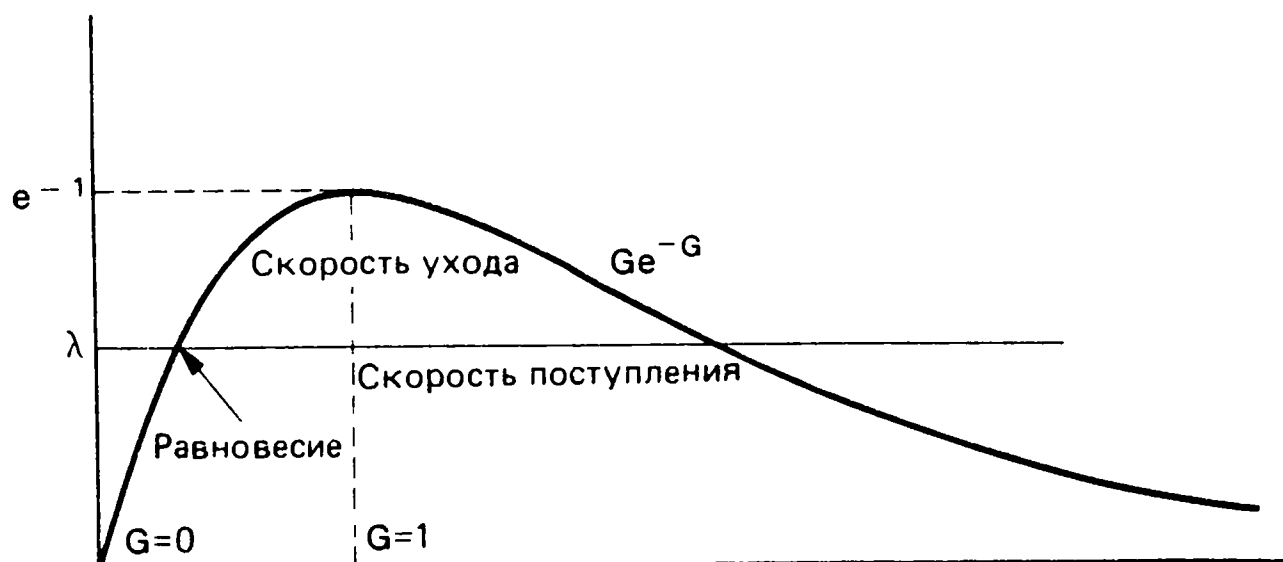


Рис. 4.2. Скорость ухода пакетов как функция интенсивности попыток передач для системы синхронная ALOHA. Если не учитывать динамическое поведение G , то уходы (т. е. успешные передачи) происходят с интенсивностью Ge^{-G} , а поступают пакеты со скоростью λ ; это позволяет предположить существование точки равновесия, показанной на рисунке.

Чтобы получить некоторое начальное интуитивное представление о синхронной системе ALOHA, мы начнем с поучительного, но не безупречного анализа. В силу предположения о бесконечном числе узлов (6б) количество новых пакетов, переданных в окне, является пуассоновской случайной величиной с параметром λ . Если повторные передачи узлов с задолженностью достаточно рандомизированы, то будет правдоподобной аппроксимация общего числа повторных передач и новых передач в заданном окне пуассоновской случайной величиной с некоторым параметром $G > \lambda$. При таком приближении вероятность успешной передачи в окне равна Ge^{-G} . Наконец, в состоянии равновесия скорость поступления λ в систему должна быть такой же, как скорость ухода Ge^{-G} . Это соотношение иллюстрируется на рис. 4.2.

Мы видим, что максимально возможная скорость ухода (согласно приведенному рассуждению) достигается при $G = 1$ и равна $1/e \approx 0,368$. Мы также замечаем, правда с некоторой неуверенностью в правоте, что для любой скорости поступления, меньшей чем $1/e$, существуют два значения G , при которых скорость поступления равна скорости ухода. Несовершенство этого элементарного подхода в том, что он не дает никакого представления о динамике системы. При изменении числа задолженных пакетов меняется параметр G ; это ведет к обратному эффекту, приводящему к дальнейшему изменению числа задолженных пакетов. Чтобы понять эту динамику, следует проанализировать систему более тщательно. Однако приведенная простая модель правильно определяет максимальную пропускную способность синхронной ALOHA как равную $1/e$ и также показывает, что G , среднее число попыток передачи в окне, должно быть порядка 1, чтобы скорость передачи была близка к $1/e$. При $G < 1$ появляется слишком

много пустых окон, а при $G > 1$ возникает слишком много конфликтов.

Для построения более точной модели предположим, что каждый узел с задолженностью передает повторно с некоторой фиксированной вероятностью q_r в каждом последующем окне до тех пор, пока не добьется успеха в передаче. Другими словами, число окон, которое проходит от конфликта до того момента, когда данный вовлеченный в конфликт узел произведет повторную передачу, является случайной величиной с геометрическим распределением, принимающей значение $i \geq 1$ с вероятностью $q_r (1 - q_r)^{i-1}$. В первоначальной версии синхронной АЛОНА применялось равномерное распределение момента повторной передачи, но это затрудняет анализ и не имеет ощутимых преимуществ по сравнению с геометрическим распределением. Мы будем использовать предположение ба об отсутствии буферов и примем несколько позже предположение бб о бесконечном числе узлов.

Поведение синхронной АЛОНА можно теперь описать дискретной цепью Маркова. Пусть n обозначает число узлов с задолженностью в начале заданного окна. Каждый из этих узлов будет передавать пакет в данном окне независимо от других узлов с вероятностью q_r . Каждый из $m - n$ остальных узлов будет передавать пакет в данном окне, если один (или более) таких пакетов поступили во время предыдущего окна. Поскольку моменты поступления распределены по пуассоновскому закону со средним λ/m , то вероятность непоступления новых пакетов равна $e^{-\lambda/m}$; следовательно, вероятность того, что узел, не имеющий задолженности, передаст пакет в данном окне, равна $q_a = 1 - e^{-\lambda/m}$. Пусть $Q_a(i, n)$ обозначает вероятность того, что i узлов, не имеющих задолженности, передают пакеты в данном окне, и пусть $Q_r(i, n)$ обозначает вероятность того, что i узлов, имеющих задолженность, передают:

$$Q_a(i, n) = \binom{m-n}{i} (1 - q_a)^{m-n-i} q_a^i, \quad (4.1)$$

$$Q_r(i, n) = \binom{n}{i} (1 - q_r)^{n-i} q_r^i. \quad (4.2)$$

Заметим, что при переходе от одного окна к другому состояние (т. е. число задолженных пакетов) увеличивается на число новых пакетов, переданных узлами, не имеющими задолженности, минус один, если пакет передается успешно. Однако пакет передается успешно, если передается только один новый и ни один задолженный пакет или ни один новый и один задолженный пакет. Таким образом, вероятность перехода из состояния n в состояние $n + i$ равна

$$P_{n, n+i} = \begin{cases} Q_a(i, n), & 2 \leq i \leq (m - n), \\ Q_a(1, n) [1 - Q_r(0, n)], & i = 1, \\ Q_a(1, n) Q_r(0, n) + Q_a(0, n) [1 - Q_r(1, n)], & i = 0, \\ Q_a(0, n) Q_r(1, n), & i = -1. \end{cases} \quad (4.3)$$

На рис. 4.3 показана эта цепь Маркова. Заметим, что состояние может уменьшиться за один переход не более чем на 1; это позволяет легко вычислить стационарные вероятности с помощью итерации, выразив p_n при каждом последующем n через p_0 и в заключение определив p_0 как нормирующий множитель (см. задачу 4.1). После этого можно найти среднее число узлов с задолженностью и при помощи теоремы Литтла вычислить среднюю задержку.

К сожалению, эта система имеет некоторые очень странные свойства при большом числе узлов и полученные таким образом стационарные распределения и результаты в значительной мере ошибочны. Чтобы это интуитивно понять, заметим, что нам желательно выбрать вероятность повторной передачи q_r достаточно большой, чтобы не допускать больших задержек после конфликтов. Если скорость поступления мала и не очень много пакетов вовлекается в конфликты, то система работает хорошо и повторные передачи, как правило, успешны. Вместе с тем если система попадает в полосу невезения и количество задолженных пакетов n становится настолько большим, что удовлетворяется неравенство $q_r n \gg 1$, то конфликты возникают почти во всех последующих окнах и система остается в состоянии большой задолженности долгое время.

Чтобы описать это явление количественно, определим $snoo$ в состоянии n (D_n) как среднее изменение задолженности за одно временное окно при старте из состояния n . Таким образом, D_n

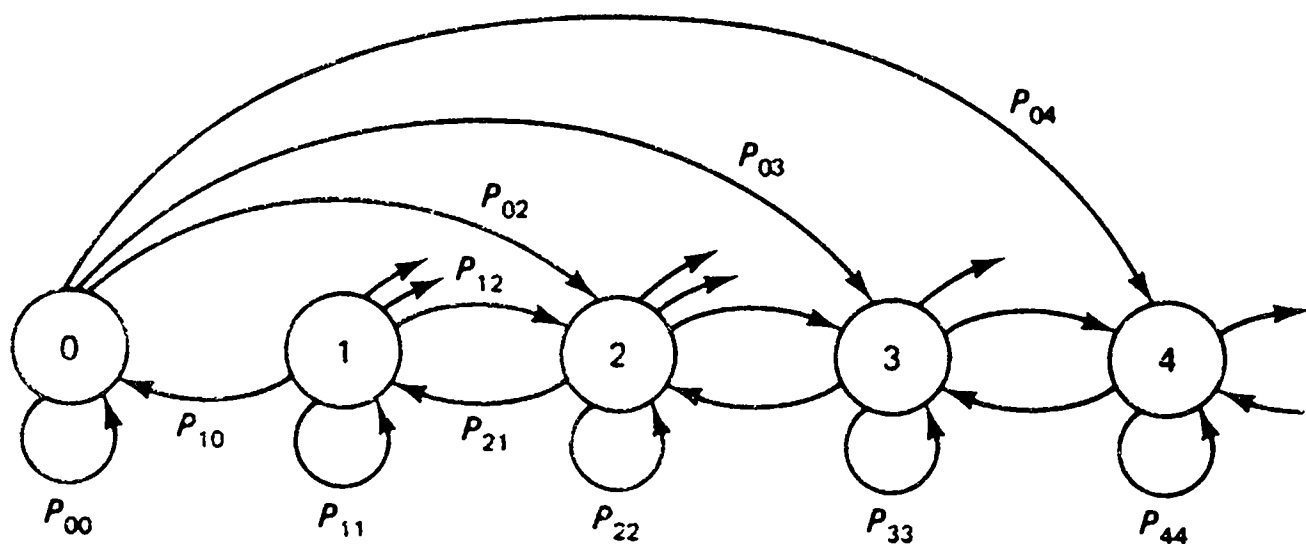


Рис. 4.3. Цепь Маркова для синхронной АЛОНА. Состояние (т. е. задолженность) может уменьшиться за переход в лучшем случае на единицу, а увеличиться — на произвольную величину.

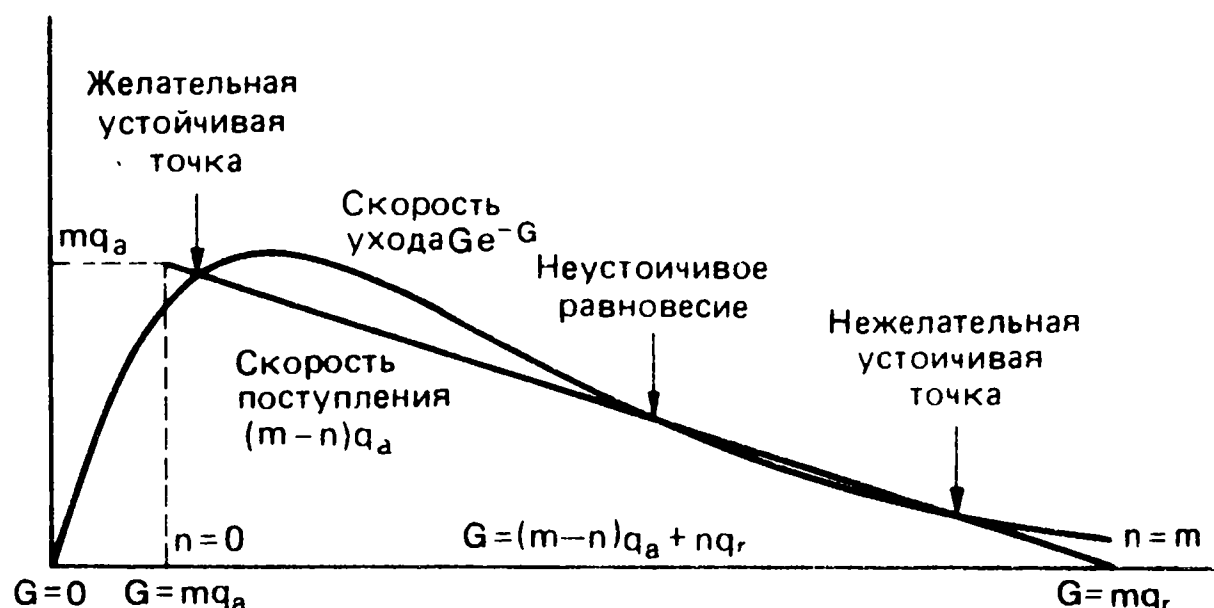


Рис. 4.4. Неустойчивость синхронной АЛОНА. На горизонтальной оси откладываются состояние n и интенсивность попыток G , которые связаны линейным соотношением $G = (m - n) q_a + n q_r$, где $q_r > q_a$. Для значений n , находящихся слева от неустойчивой точки, величина D отрицательна и n сносится в направлении желательной устойчивой точки. Для значений n , находящихся справа от неустойчивой точки, D положительна и n сносится в направлении нежелательной устойчивой точки.

равно среднему числу новых пакетов, поступивших в систему, минус среднее число успешных передач в окне; среднее число успешных передач равно вероятности успешной передачи, обозначаемой через $P_{\text{усп}}$. Таким образом,

$$D_n = (m - n) q_a - P_{\text{усп}}, \quad \text{где} \quad (4.4)$$

$$P_{\text{усп}} = Q_a(1, n) Q_r(0, n) + Q_a(0, n) Q_r(1, n). \quad (4.5)$$

Определим интенсивность попыток $G(n)$ как среднее число попыток передач в окне, когда система находится в состоянии n ; имеем

$$G(n) = (m - n) q_a + n q_r.$$

Если q_a и q_r малы, $P_{\text{усп}}$ хорошо аппроксимируется следующей функцией интенсивности попыток:

$$P_{\text{усп}} \approx G(n) e^{-G(n)}. \quad (4.6)$$

Это приближение можно непосредственно получить из (4.5), используя приближение $(1 - x)^y \approx e^{-xy}$ для малых x в выражениях для Q_a и Q_r . Аналогично вероятность пустого окна равна приближенно $e^{-G(n)}$. Таким образом, число пакетов, передаваемых в окне, хорошо аппроксимируется пуассоновской случайной величиной (как и в первоначальном интуитивном анализе), но параметр $G(n)$ зависит от состояния. Рис. 4.4 иллюстрирует равенства (4.4) и (4.6) в случае $q_r > q_a$ (как выяснится в дальнейшем, только этот случай интересен). Снос равен разности между кривой и прямой линиями. Поскольку снос определяется как среднее изменение состояния при переходе от одного окна к дру-

тому, система, хотя и, быть может, флуктуирует, но в среднем движется в направлении сноса и, следовательно, имеет тенденцию к блужданию вокруг двух устойчивых точек с редкими переходами от одной устойчивой точки к другой.

Из этого рисунка следуют два важных заключения. Во-первых, скорость ухода, т. е. $P_{\text{усп}}$, при больших m не превышает $1/e$. Во-вторых, скорость ухода почти равна нулю в течение длинных периодов, когда система переходит в нежелательную устойчивую точку. Рассмотрим последствия изменения q_r . При увеличении q_r задержка повторной передачи попавших в конфликт пакетов уменьшается, но линейное соотношение между n и интенсивностью попыток $G(n) = (m - n)q_a + nq_r$ также меняется (т. е. $G(n)$ увеличивается с ростом n быстрее, если q_r больше). Если горизонтальный масштаб для n остается неизменным на рис. 4.4, то это изменение интенсивности попыток соответствует сжатию горизонтального масштаба для $G(n)$ и, следовательно, горизонтальному сжатию кривой Ge^{-G} . Это означает, что число задолженных пакетов, необходимых для перехода за неустойчивую точку равновесия, уменьшается. Наоборот, если q_r уменьшается, то задержка повторной передачи увеличивается, но становится труднее перейти за неустойчивую точку равновесия. Если q_r уменьшится в достаточной степени (тем не менее оставаясь больше q_a), то кривая Ge^{-G} на рис. 4.4 растянется так, что останется только одно устойчивое состояние. В этой устойчивой точке и аналогично при $q_r = q_a$ задолженность составляет значительную часть m и это означает, что значительная часть поступающих пакетов сбрасывается и задержка чрезмерно велика.

Вопросу о том, при каких значениях q_r и скорости поступления поведение системы устойчиво, особенно если q_r и скорость поступления меняются от узла к узлу и в каждом узле имеется бесконечный буфер, посвящено немало теоретических исследований (например, [244]). В этих работах обычно рассматривается случай, когда узлы становятся задолжниками сразу в момент поступления пакетов. Из задачи 4.8 видно, что если вероятность q_r достаточно мала для того, чтобы система работала устойчиво, то задержка значительно больше, чем в случае ВУ; следовательно, системы АЛОНА не имеют большого практического значения.

Если предположение об отсутствии буферов заменить на предположение о бесконечном числе станций, то интенсивность попыток $G(n)$ станет равной $\lambda + nq_r$, и прямая линия, представляющая на рис. 4.4 поступление новых пакетов, станет горизонтальной. В этом случае нежелательная устойчивая точка исчезает и, если система хотя бы однажды перейдет через неустойчивое равновесие, ее состояние приобретает тенденцию к неограниченному росту. В этом случае соответствующая бесконечная цепь Маркова не имеет стационарного распределения и средняя

задолженность увеличивается неограниченно по мере того, как система продолжает работать. Математические исследования подобных ситуаций даны в работах [131, 186].

На практике можно рассчитывать на то, что если скорость поступления λ намного меньше, чем $1/e$, и если q_r имеет умеренную величину, то система будет оставаться в желательном устойчивом состоянии в течение очень длинных периодов. В случае маловероятного перехода в нежелательную устойчивую точку систему можно запустить заново, сбросив задолженные пакеты. Однако, вместо того чтобы продолжать анализ этой довольно несовершенной системы, мы обратимся к модификациям синхронной АЛОНА, в которых обеспечивается устойчивость.

4.2.3. Адаптивная синхронная АЛОНА

Один простой способ достижения устойчивости теперь почти очевиден. Вероятность $P_{\text{усп}}$ приближенно равна $G(n) e^{-G(n)}$; это выражение достигает максимума при $G(n) = 1$. Таким образом, желательно динамически изменять q_r , чтобы поддерживать интенсивность попыток $G(n)$ равной 1. Трудность здесь состоит в том, что n неизвестно узлам и может быть оценено только на основе информации, идущей по обратной связи. Существует много способов оценки n или соответствующего значения q_r (см., например, [110, 199]). Все они, по существу, увеличивают q_r , когда появляется пустое окно, и уменьшают q_r , когда возникает конфликт.

Устойчивость и максимальная скорость передачи

Понятие устойчивости необходимо несколько уточнить, прежде чем продолжать рассмотрение. Синхронная АЛОНА была названа неустойчивой в предыдущем подразделе на основе изучения модели, иллюстрируемой рис. 4.4. Однако если предполагается отсутствие буферизации, то система имеет вполне определенное стационарное поведение при любой скорости поступления. Вместе с тем если предполагается бесконечное число узлов, то стационарного распределения не существует и средняя задержка растет неограниченно по мере продолжения работы системы. Если предполагается отсутствие буферизации, то система сбрасывает большое число поступающих пакетов и имеет очень большую, но конечную задержку; однако если предполагается бесконечное число узлов, то ни один поступающий пакет не сбрасывается, но задержка становится бесконечной.

В дальнейшем мы будем использовать предположение бб о бесконечном числе узлов и считать систему с множественным доступом устойчивой при заданной скорости поступления, если

средняя задержка пакета (предел или среднего по времени или среднего по ансамблю, когда время стремится к бесконечности) конечна. Обычная синхронная АЛОНА неустойчива в этом смысле при любой скорости поступления, большей нуля. Заметим, что если система устойчива, то при достаточно большом, но конечном числе узлов m система (в предположении, что каждому поступающему пакету соответствует новый виртуальный узел) имеет меньшую среднюю задержку, чем при ВУ, так как задержка при ВУ с фиксированной общей скоростью поступления увеличивается линейно с ростом m .

Максимальная скорость устойчивой передачи определяется как наименьшая верхняя граница скоростей поступления, при которых система устойчива. Например, максимальная скорость устойчивой передачи обычной синхронной АЛОНА равна нулю. Наша цель, для достижения которой были введены эти определения, состоит в том, чтобы изучить алгоритмы множественного доступа, которые не требуют знания числа узлов m и обеспечивают малую задержку (при заданной λ), не зависящую от m . В дальнейшем некоторое внимание будет уделено модификациям, в которых непосредственно используется знание числа узлов.

Возвращаясь к адаптации синхронной АЛОНА, заметим, что если оценка задолженности точна и $G(n)$ поддерживается равной оптимальному значению 1, то (согласно пуассоновскому приближению) пустые окна возникают с вероятностью $1/e \approx 0,368$, успешные передачи — с вероятностью $1/e$ и конфликты — с вероятностью $1 - 2/e \approx 0,264$. Таким образом, правило изменения q , должно допускать число конфликтов, меньшее чем число пустых окон. Максимальная скорость устойчивой передачи такой системы равна в лучшем случае $1/e$. Чтобы увидеть это, обратим внимание на то, что, когда задолженность велика, пуассоновское приближение становится более точным, скорость успешных передач стремится к $1/e$ и, следовательно, снос положителен при $\lambda > 1/e$. Важно заметить, что это утверждение основывается на том, что во всех узлах с задолженностью используется одна и та же вероятность повторной передачи. Мы увидим в разд. 4.3, что, если в узлах учитывается как их собственная история повторных передач, так и история канала при принятии решения о передаче, становятся возможными максимальные скорости устойчивой передачи, большие чем $1/e$.

Псевдобайесовский алгоритм

Псевдобайесовский алгоритм Ривеста [199] представляет особенно простой и эффективный способ адаптации АЛОНА. Этот алгоритм, по существу, совпадает с более ранним независимо разработанным алгоритмом Михайлова [174], но байесовская

интерпретация Ривеста проще для понимания. Алгоритм отличается от синхронной АЛОНА тем, что новые поступающие пакеты становятся задолженными сразу в момент поступления. В следующем окне они передаются не с вероятностью 1, а с вероятностью q_r , т. е. так же, как пакеты, уже попавшие в конфликт. Таким образом, если в начале окна число задолженных пакетов равно n (с учетом только что поступивших), то интенсивность попыток $G(n) = nq_r$, а вероятность успешной передачи равна $nq_r(1 - q_r)^{n-1}$. Для нестабилизированной АЛОНА эта модификация не имела бы большого смысла, так как q_r должна быть относительно малой и новые поступающие пакеты приобрели бы дополнительную ненужную задержку. В случае адаптивной АЛОНА, однако, q_r может повышаться до 1, когда оцениваемая задолженность незначительна, так что новые поступающие пакеты задерживаются только тогда, когда оценка показывает, что система уже перегружена. В задаче 4.6 показывается, что эта модификация увеличивает вероятность успешной передачи, если задолженность оценивается точно.

Алгоритм работает, производя вычисление оценки \hat{n} задолженности n в начале каждого окна. Каждый задолженный пакет при этом передается независимо с вероятностью $q_r(\hat{n}) = \min\{1, 1/\hat{n}\}$. Операция взятия минимума ограничивает q_r сверху единицей и благодаря этому ограничению интенсивность попыток $G = nq_r$ стремится достичь уровня 1. При каждом k оценка задолженности в начале окна $k + 1$ обновляется исходя из оценки задолженности и сигнала обратной связи в окне k в соответствии с правилом

$$\hat{n}_{k+1} = \begin{cases} \max\{\lambda, \hat{n}_k + \lambda - 1\} & \text{(для пустого или успешного окна),} \\ \hat{n}_k + \lambda + (e - 2)^{-1} & \text{(для конфликтного окна).} \end{cases} \quad (4.7)$$

Добавлением λ к предыдущей задолженности учитываются новые поступающие пакеты. Операция взятия максимума обеспечивает, чтобы оценка никогда не была меньше вклада новых пакетов. После успешных передач 1 вычитается из предыдущей задолженности, чтобы учесть успешный уход пакета из системы. Наконец, вычитание 1 из предыдущей задолженности после пустых окон и добавление $(e - 2)^{-1}$ после конфликтов приводят к тому, что \hat{n} уменьшается, когда появляется слишком много пустых окон, и \hat{n} увеличивается, когда возникает слишком много конфликтов. При большой задолженности и $\hat{n} = n$ каждый из n задолженных пакетов независимо передается в окне с вероятностью $q_r = 1/n$. Следовательно, $G(n)$ равна 1 и, согласно пуассоновскому приближению, пустые окна появляются с вероятностью $1/e$, а конфликты с вероятностью $(e - 2)/e$, так что уменьшение \hat{n}

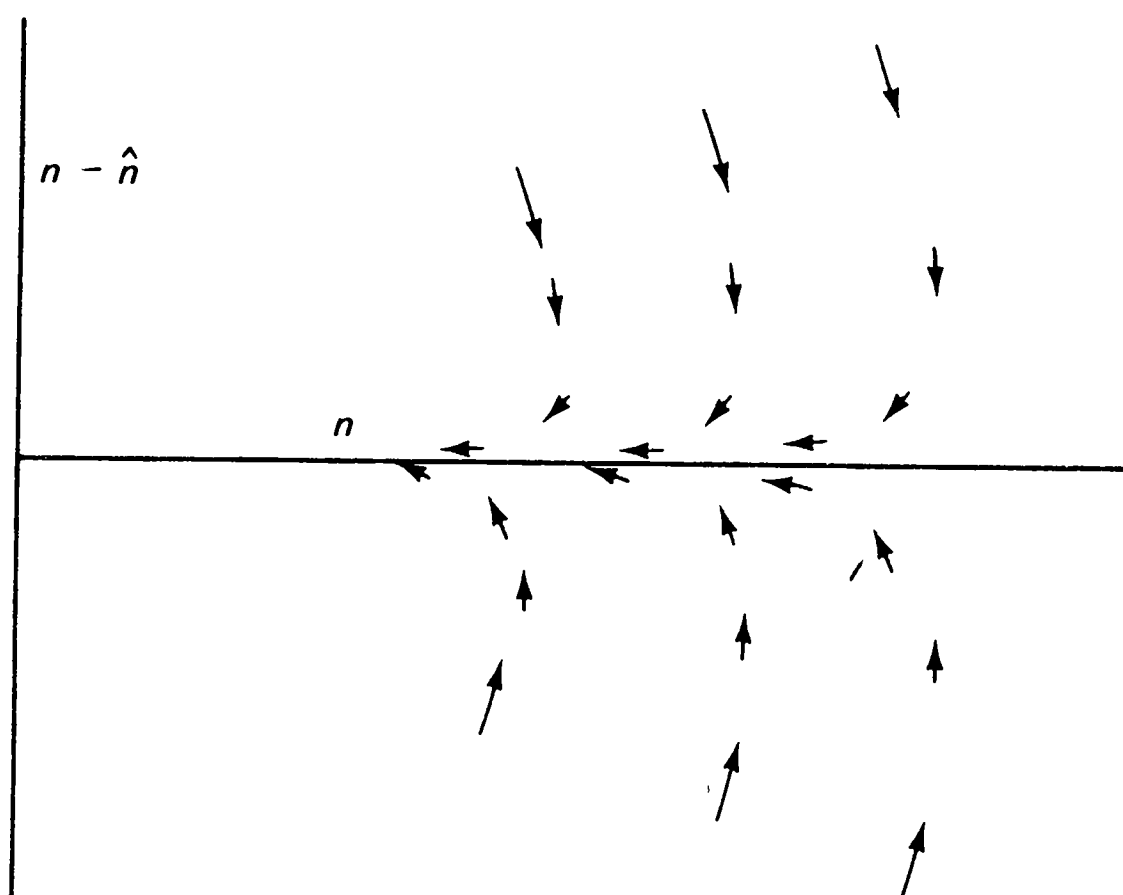


Рис. 4.5. Снос n и $n - \hat{n}$ в случае псевдобайесовского адаптивного алгоритма. Когда абсолютная величина $n - \hat{n}$ велика, она стремится к нулю быстрее чем $1/n$.

на 1 после пустых окон и увеличение \hat{n} на $(e - 2)^{-1}$ после конфликтов поддерживает в среднем равновесие между n и \hat{n} .

В задаче 4.9 показывается, что если априорное распределение вероятностей величины n_k является пуассоновским с параметром $\hat{n}_k \geq 1$, то, при условии пустого окна k или успешной передачи в окне k , распределение вероятностей величины n_{k+1} является пуассоновским с параметром $\hat{n}_k + \lambda - 1$. При условии конфликта результирующее распределение n_{k+1} не совсем пуассоновское, но оно приемлемо аппроксимируется пуассоновским с параметром \hat{n}_{k+1} . По этой причине алгоритм называется псевдобайесовским.

Рис. 4.5 помогает до некоторой степени понять эвристически, почему алгоритм устойчив при $\lambda < 1/e$. Состояние системы характеризуется величинами n и \hat{n} и на рисунке показан средний снос этих величин за одно окно. Если n и \hat{n} велики и равны, то средний снос каждой равен выражению $\lambda - 1/e$, которое отрицательно. Если же абсолютная величина $n - \hat{n}$ велика, то средний снос n положителен, но среднее уменьшение $|n - \hat{n}|$ значительно больше. Таким образом, если система стартует из некоторой произвольной точки (n, \hat{n}) плоскости, n может увеличиваться в среднем в течение некоторого числа окон, но в конце концов n и \hat{n} сблизятся и тогда n будет в среднем уменьшаться.

В приложениях скорость поступления λ обычно неизвестна и является медленно меняющейся величиной. Следовательно, алгоритм должен или оценивать λ по средней частоте успешных передач или полагать ее равной некоторому фиксированному значению. Михайлов [137] и Цициклис [235] показали, что если

в алгоритме скорость поступления пакетов фиксирована и равна $1/e$, то система устойчива при всех фактических значениях λ , таких, что $\lambda < 1/e$. Никаких строго доказанных результатов не было получено относительно поведения алгоритма в случае, когда используется динамическая оценка λ .

Приближенный анализ задержки

Точный анализ средней задержки этого алгоритма даже при известной величине λ , по-видимому, очень труден и полезнее выполнить анализ приближенной модели. Предположим, что величина λ известна и вероятность успешной передачи $P_{\text{усп}}$ равна $1/e$, когда задолженность n равна 2 или более, и что $P_{\text{усп}} = 1$ при $n = 1$. Эта модель приемлема при очень малых λ , так как конфликты возникают очень редко и q_r равна, как правило, 1. Она также приемлема при больших $\lambda < 1/e$, поскольку n обычно велико и $\hat{n} \approx n$.

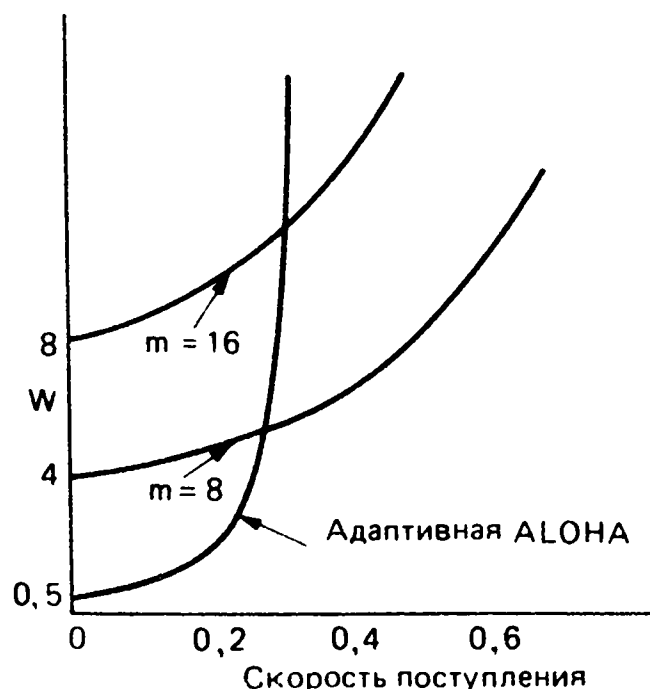
Пусть W_i обозначает время, прошедшее от момента поступления i -го пакета до начала i -й успешной передачи. Заметим, что если дисциплина обслуживания в системе была бы «первый пришел—первый обслуживается», то W_i было бы тогда временем ожидания для i -го поступившего пакета. Применяя ту же аргументацию, что и в разд. 3.5, получаем, что усреднение W_i по всем i дает среднее время ожидания W . Пусть n_i обозначет число задолженных пакетов в момент времени, непосредственно предшествующий моменту поступления i -го пакета; в величину n_i не дает вклада пакет, находящийся в процессе успешной передачи, но дают вклады пакеты, вступившие в конфликт. W_i можно представить следующим образом:

$$W_i = R_i + \sum_{j=1}^{n_i} t_j + y_i. \quad (4.8)$$

Величина R_i — это остаточное время до начала следующего окна, а t_1 (при $n_i > 0$) есть последующий отрезок времени, заканчивающийся вместе с окончанием следующей успешной передачи. Аналогично t_j , $1 < j \leq n_i$ есть интервал времени от конца $(j-1)$ -й последовательной успешной передачи до конца j -й последовательной успешной передачи. После того как пройдут n_i успешных передач, начинается остающийся интервал y_i , который заканчивается в момент начала следующей успешной передачи, т. е. успешной передачи с номером i .

Обратим внимание на то, что для каждого интервала t_j задолженность составляет не менее двух пакетов, учитывая новый i -й по порядку поступления пакет и n_i пакетов, уже являющихся задолженными. Следовательно, каждое окно является успешным с вероятностью $1/e$ и средняя длина каждого t_j равна e . Далее за-

Рис. 4.6. Сравнение среднего времени ожидания W , измеряемого в числе окон, от момента поступления до начала успешной передачи для адаптивной АЛОНА и ВУ с $m = 8$ и $m = 16$. При малых скоростях поступления задержка адаптивной АЛОНА немного больше времени ожидания начала следующего окна, а когда скорость поступления приближается к $1/e$, задержка растет неограниченно.



метим, что формула Литтла устанавливает связь между средними величинами W_i и n_i (где W_i — время ожидания без учета успешной передачи, а n_i — число пакетов в системе без учета текущей успешной передачи). Наконец, среднее значение R_i равно $1/2$ (время измеряется в окнах). Итак, переходя к средним величинам в (4.8) и усредняя по i , получаем

$$W = 1/2 + \lambda e W + E\{y\}. \quad (4.9)$$

Теперь рассмотрим систему в момент окончания окна, в котором произошла $(i - 1)$ -я успешная передача, а также поступил i -й пакет. Если задолженность равна 1 в этой точке (т. е. только i -й пакет находится в системе), то y_i равно 0. В противном случае при $n > 1$ имеем $E\{y_i\} = e - 1$. Пусть p_n обозначает стационарную вероятность того, что задолженность на границе окна равна n . Далее, поскольку пакет всегда передается успешно, если в начале окна состояние 1, то p_1 равно доле окон, в которых состояние равно 1 и пакет успешно передается. Так как общая доля окон с успешными передачами равна λ , то доля пакетов, переданных из состояния 1, равна p_1/λ , а доля пакетов, переданных из состояний с большими номерами, равна $1 - p_1/\lambda$. Отсюда следует, что

$$E\{y\} = \frac{(e - 1)(\lambda - p_1)}{\lambda}. \quad (4.10)$$

В заключение мы должны определить p_1 . Из приведенного рассуждения следует, что $\lambda = p_1 + (1 - p_0 - p_1)/e$. Перейти в состояние 0 на границе окна можно только в случае, если ни один пакет не поступил в течение предыдущего окна и предыдущее состояние было 0 или 1. Следовательно, $p_0 = (p_0 + p_1)e^{-\lambda}$. Решая эти уравнения относительно p_1 , получаем

$$p_1 = \frac{(1 - \lambda e)(e^\lambda - 1)}{[1 - (e - 1)(e^\lambda - 1)]}. \quad (4.11)$$

Из (4.9)—(4.11) следует

$$W = \frac{e - 1/2}{1 - \lambda e} - \frac{(e^\lambda - 1)(e - 1)}{\lambda [1 - (e - 1)(e^\lambda - 1)]}. \quad (4.12)$$

Величина W , полученная из этой формулы, достаточно близка к значению задержки псевдобайесовского алгоритма, полученной посредством моделирования. На рис. 4.6 показана $E\{W\}$ для этой приближенной модели и даны для сравнения времена ожидания для ВУ с 8 и 16 узлами. Довольно удивительно, что задержка мала, даже когда скорость поступления относительно близка к $1/e$. По-видимому, предположение о быстрой обратной связи не является необходимым для того, чтобы адаптивные стратегии этого типа приводили к устойчивости системы; обоснование этого состоит в том, что задержка обратной связи наверняка сделает оценку \hat{n} менее точной, но отношение n/\hat{n} тем не менее должно оставаться близким к 1 при больших n .

Замедление по двоичной экспоненте

В радиосетях, которые будут рассматриваться в разд. 4.6, и в некоторых других системах с множественным доступом предположение об обратной связи с сигналами 0, 1, e нереально. В некоторых системах узел получает сигнал обратной связи, извещающий только о том, успешно или нет были переданы его собственные пакеты; он не получает никаких сигналов обратной связи в окнах, в которых не передает. Такая ограниченная обратная связь достаточна для синхронной системы АЛОНА, но не достаточна для оценки задолженности при псевдобайесовской стратегии. Замедление по двоичной экспоненте [172] является стабилизирующей стратегией, применяемой в системе ETHERNET, в которой используется только эта более ограниченная форма обратной связи.

Такая стратегия очень проста; если пакет безуспешно передавался i раз, то вероятность передачи в последующих окнах предполагается равной $q_r = 2^{-i}$ (или для передачи равновероятно выбирается одно из 2^i окон, следующих за i -й безуспешной попыткой). При первоначальном поступлении в систему пакет сразу передается в следующем окне.

Некоторое обоснование этой стратегии состоит в том, что, когда пакет впервые поступает (при такой ограниченной обратной связи), в узле ничего не известно о задолженности, так что разумно сразу передавать. В случае последовательности конфликтов любая разумная оценка задолженности увеличилась бы, вызывая уменьшение текущего значения q_r . При этом ситуация усложняется из-за того, что при уменьшении q_r узел получает меньше информации по обратной связи о величине задолженности в рас-

чете на одно окно, и, следовательно, чтобы действовать осторожнее, разумно увеличивать оценку задолженности во все большей степени после каждого очередного конфликта.

К сожалению, эта стратегия неустойчива (т. е. имеет бесконечную среднюю задержку при $m = \infty$) при любой скорости поступления λ , большей чем 0 [5]. Неизвестно, может ли какая-либо стратегия обеспечивать устойчивость при этом типе ограниченной обратной связи. В задаче 4.10, однако, предполагается еще одна стратегия, которую можно использовать при такой ограниченной обратной связи и конечном числе узлов с неограниченными буферами; эта стратегия дает конечную среднюю задержку при любом значении λ , меньшем чем 1. К сожалению, такая высокая скорость достигается за счет чрезмерно большой задержки.

4.2.4. Несинхронная АЛОНА

Несинхронная или чистая АЛОНА [2] является предшественником синхронной АЛОНА. В этой системе каждый узел, получив новый пакет, передает его немедленно, а не ждет начала окна. Окна никак не используются в чистой АЛОНА, так что мы временно не будем считать систему синхронной. Если пакет попадает в конфликт, то он передается повторно с предварительной случайной задержкой. Предположим, что если времена передач двух пакетов в какой-либо мере перекрываются, то ЦИП в этих пакетах покажут на наличие ошибки и потребуются повторная передача. Мы предполагаем, что приемник ретранслирует в широкопередаточном режиме принятое наложение сигналов (или что во всех узлах известно о наложении сигналов), так что в каждом узле по прошествии заданного времени распространения можно определить, были ли правильно приняты переданные им пакеты или нет. Таким образом, мы имеем тот же вид ограниченной обратной связи, который обсуждался в последнем подразделе. Можно рассматривать другие виды обратной связи, и в задаче 4.11 исследуются специфические последствия других предположений о виде обратной связи.

На рис. 4.7 показано, что если один пакет начинает передаваться в момент времени t и все пакеты имеют единичную длину, то любая другая передача, начинающаяся в промежутке от $t - 1$ до $t + 1$, приводит к конфликту. Для простоты рассмотрим бесконечное число узлов (т. е. предположение 6б) и обозначим через n число задолженных пакетов в заданный момент времени.

Рассмотрим последовательность следующих друг за другом попыток передать пакет по каналу. При некотором заданном значении i обозначим через τ_i длительность интервала между моментами начала i -й и $(i + 1)$ -й попыток передач. Попытка с номером i будет успешной, если как τ_i , так и τ_{i-1} превышают 1 (предпола-

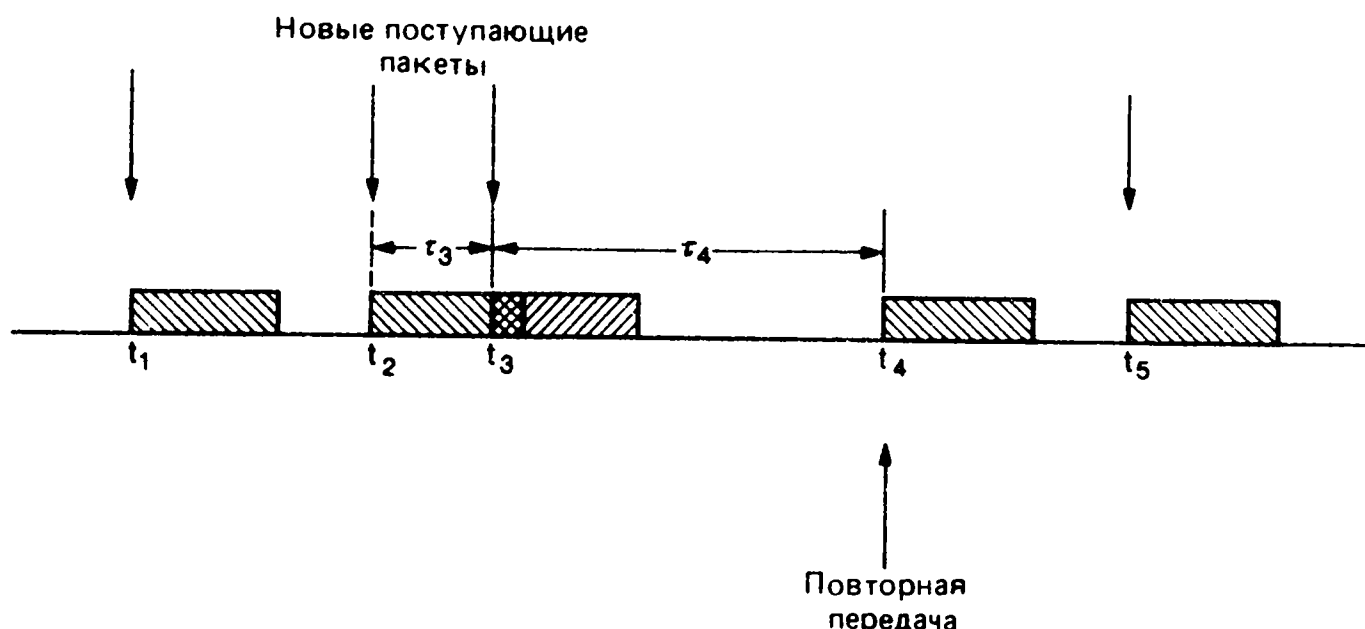


Рис. 4.7. Несинхронная АЛОНА. Новые поступающие пакеты передаются немедленно, а неуспешные передачи повторяются спустя случайное время. Время передачи пакета равно единице и две передачи приводят к возникновению конфликта, если интервал между их началами меньше одной единицы.

гается, что длина пакета единичная). Если в каждом интервале задана задолженность, то эти интервалы независимы. Тогда, предполагая, что в каждом интервале задолженность равна n , мы имеем

$$P_{\text{усп}} = e^{-2G(n)}. \quad (4.13)$$

Поскольку предпринимаемые передачи возникают с интенсивностью $G(n)$, скорость передачи (т. е. среднее число успешных передач в единицу времени) как функция n имеет вид

$$\text{Скорость}(n) = G(n) e^{-2G(n)}. \quad (4.14)$$

Рисунок 4.8 иллюстрирует этот результат. Ситуация весьма аналогична той, что характерна для синхронной АЛОНА, за исключением того, что максимум скорости передачи равен $1/(2e)$ и достигается при $G(n) = 1/2$. Приведенный анализ является приближенным в том смысле, что равенство (4.13) предполагает, что задолженность одинакова в интервалах, окружающих заданную попытку передачи, тогда как в соответствии с нашим определением задолженности она уменьшается на единицу всякий раз, когда начинается передача задолженного пакета, и увеличивается на единицу всякий раз, когда обнаруживается наложение пакетов. При малых x (т. е. большом среднем времени задержки перед попыткой передачи) этот эффект довольно незначителен.

Из рис. 4.8 видно, что системе чистой АЛОНА свойственна та же проблема устойчивости, что и системе синхронной АЛОНА. Из-за ограниченности обратной связи, которая здесь предполагается, очень трудно достичь устойчивости или анализировать ее (так же как и в случае синхронной АЛОНА). Относительно устой-

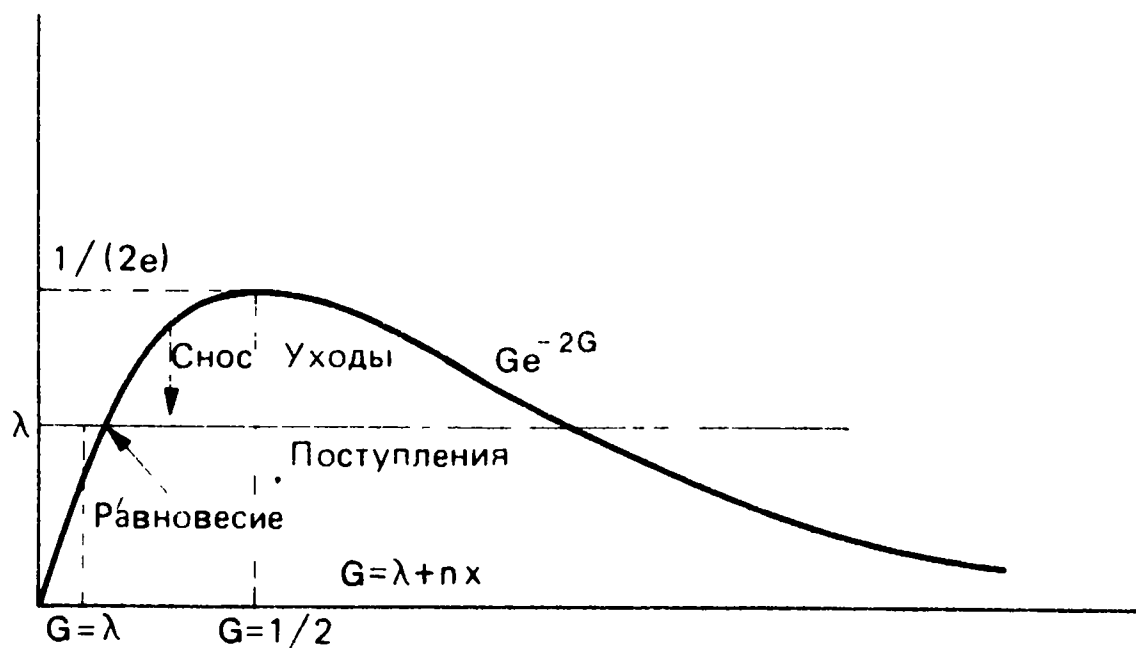


Рис. 4.8. Скорость передачи системы ALOHA как функция интенсивности попыток передач G . Успешно переданные пакеты уходят из системы со скоростью Ge^{-2G} , а поступление пакетов происходит со скоростью λ , что ведет к гипотетическому равновесию в указанной на рисунке точке.

чивости чистой ALOHA известно немного, но если λ очень мало и средняя задержка повторной передачи очень большая, можно ожидать, что система будет работать в течение длительных периодов без значительного накопления задолженных пакетов.

Одно из главных преимуществ чистой ALOHA состоит в том, что ее можно использовать, когда пакеты имеют различную длину, тогда как при синхронной ALOHA длинные пакеты должны разбиваться на части, соответствующие окнам, а короткие пакеты должны дополняться до длины окна. Это компенсирует в некоторой степени снижение скорости передачи у чистой ALOHA и делает алгоритм менее сложным. Анализ следствий, обусловленных переменной длиной пакетов, приводится в работах [69, 209].

4.3. Алгоритмы разбиения

Мы выяснили, что синхронная ALOHA требует некоторых усилий для поддержания устойчивости и ее скорость, по существу, ограничена величиной $1/e$. Теперь обратимся к более тонким методам разрешения конфликтов, которые обеспечивают устойчивость без применения каких-либо сложных процедур оценки и к тому же увеличивают диапазон доступных скоростей. Чтобы дать интуитивное представление о том, как это можно сделать, заметим, что при малой интенсивности попыток передач в возникшем конфликте с наибольшей вероятностью участвуют только два пакета. Если запретить передачу пакетов до тех пор, пока конфликт не разрешится, то каждый из вступивших в конфликт пакетов мог бы независимо передаться в следующем окне с вероятностью $1/2$. Это приводило бы к успешной передаче какого-либо из пакетов с вероятностью $1/2$, а другой мог бы затем пере-

даться в следующем окне. В противоположном случае с вероятностью $1/2$ возникает другой конфликт или пустое окно. В этом случае каждый из двух пакетов мог бы снова независимо передаваться в следующем окне с вероятностью $1/2$ и т. д. до тех пор, пока не возникнет успешная передача, за которой последовала бы передача оставшегося пакета.

При такой стратегии двум пакетам потребуется два окна с вероятностью $1/2$, три окна с вероятностью $1/4$ и i окон с вероятностью $2^{-(i-1)}$. Среднее число окон, необходимых для передачи этих двух пакетов, теперь можно вычислить и получить величину три, что дает скорость передачи $2/3$ для периода, в течение которого разрешается конфликт.

Существуют различные способы, используя которые в узлах, попавших в конфликт, можно делать выбор между тем, передавать или нет в последующих окнах. В каждом узле можно просто подбрасывать симметричную монету, чтобы делать выбор. При альтернативном варианте (который будет описан в дальнейшем) в каждом узле можно использовать момент прихода вступившего в конфликт пакета. Наконец, при конечном числе узлов, каждый из которых имеет однозначный идентификатор (номер), представляющий последовательность битов, в узле можно последовательно выбирать биты его идентификатора для того, чтобы производить последовательность выборов. Последний вариант обладает тем преимуществом, что число окон, необходимых для разрешения конфликта, ограничено, поскольку идентификаторы каждой пары узлов должны иметь хотя бы по одному несовпадающему биту. Все эти варианты имеют общую особенность, состоящую в том, что множество вступивших в конфликт узлов разбивается на подмножества, причем узлы одного из них передают в следующем окне. Если конфликт не разрешился (например, из-за того, что каждый вступивший в конфликт узел находится в одном и том же множестве), то производится дальнейшее разбиение на подмножества. Мы называем алгоритмы этого типа *алгоритмами разбиения*. В дальнейшем при изучении этих алгоритмов мы имеем в виду модель с синхронным каналом, пуассоновским поступающим потоком, конфликтами или успешным приемом, быстрой обратной связью с сигналами $0, 1, e$, повторной передачей при попадании в конфликт и бесконечным числом узлов, т. е. предположения 1—6б подразд. 4.2.1.

4.3.1. Древовидные алгоритмы

Первыми по моменту появления алгоритмами разбиения являются алгоритмы с древовидной структурой [42, 113, 241]. Когда возникает конфликт, например в k -м окне, узлы, не участвующие в конфликте, переходят в состояние ожидания, а те,

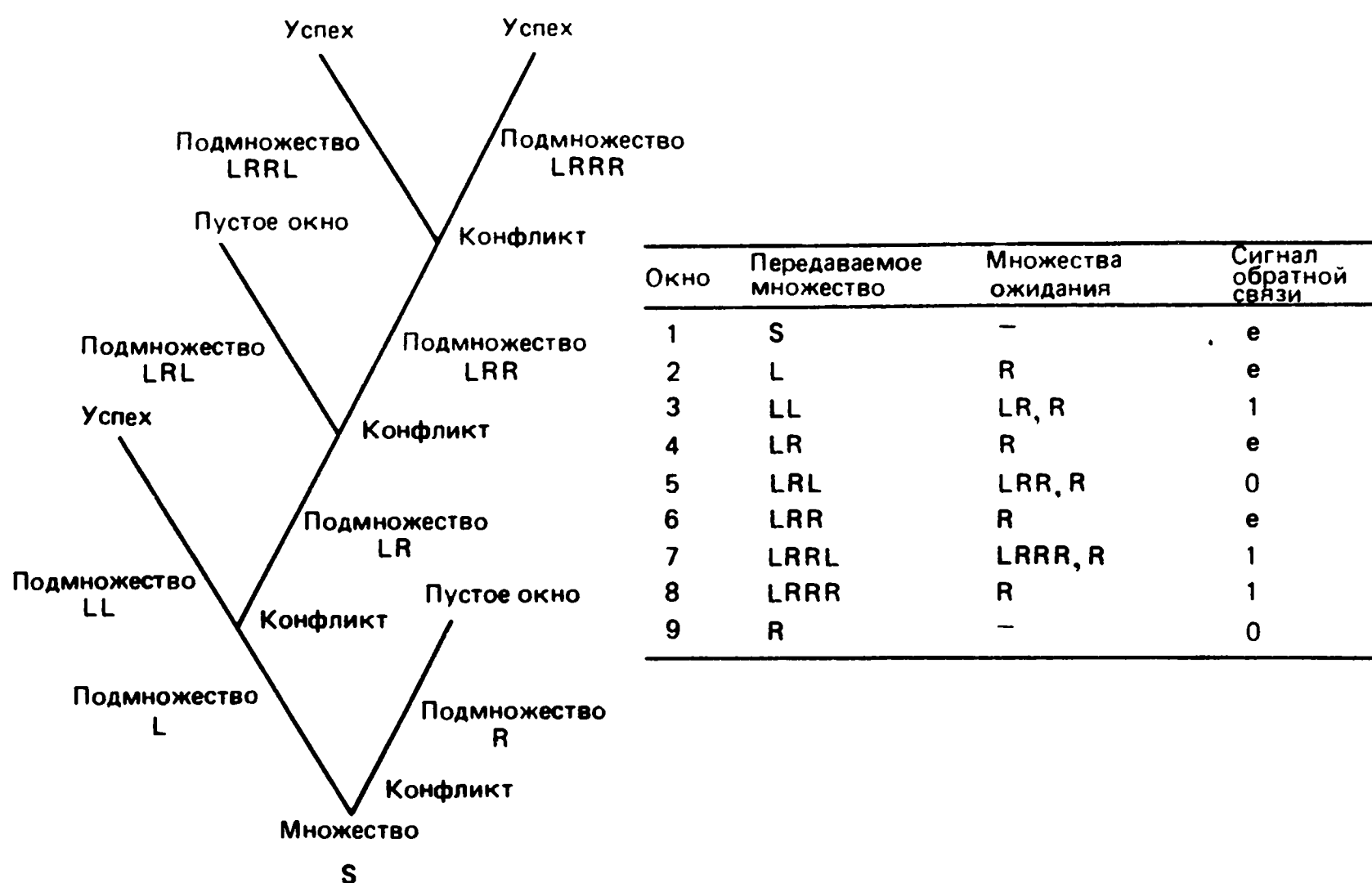


Рис. 4.9. Древоподобный алгоритм. После конфликта все новые поступающие пакеты переходят в состояние ожидания, а все вступившие в конфликт узлы разделяются на подмножества. Каждый последующий конфликт, созданный подмножеством узлов, вызывает новое разбиение этого подмножества на меньшие подмножества, в то время как другие узлы ждут.

которые вступили в конфликт, разбиваются на два подмножества (например, посредством подбрасывания монеты в каждом из узлов). Первое подмножество узлов передает в окне $k + 1$ и, если это окно пустое или в нем была успешная передача, то второе подмножество передает в окне $k + 2$ (рис. 4.9). В противном случае, когда еще один конфликт возникает в окне $k + 1$, первое из двух подмножеств разбивается еще раз и второе подмножество узлов ожидает разрешения этого конфликта.

Корневое двоичное дерево на рис. 4.9 отображает последовательность пустых окон, успешных передач и конфликтов, появляющихся при одной из возможных последовательностей таких разбиений. S обозначает множество пакетов, вступивших в первоначальный конфликт, а L (левое) и R (правое) обозначают два подмножества, на которые разбивается S . Аналогично LL и LR обозначают два подмножества, на которые разбивается L после того, как узлы из L вступили в конфликт. Каждой вершине дерева соответствует подмножество (быть может, пустое) задолженных пакетов. Из вершин, подмножества которых содержат два или более пакетов, идут вверх две ветви, что соответствует разбиению подмножества на два новых подмножества; вершины, соответствующие пустым подмножествам или подмножествам с одним пакетом, являются концевыми вершинами дерева.

Множество пакетов, соответствующих корневой вершине S , передается первым и после передачи подмножества, соответствующего какой-либо неконцевой вершине, передается подмножество, соответствующее вершине левой ветви, и все исходящие из него подмножества, а затем подмножества правой ветви. Так как мы предположили, что обратная связь быстрая, то должно быть ясно, что каждый узел может в принципе строить это дерево по мере того, как поступают сигналы 0, 1, e обратной связи; каждый узел может хранить в памяти путь следования по дереву своего пакета и, следовательно, каждый узел может определить момент передачи своего задолженного пакета.

Описанный порядок передачи легко организовать с помощью стека [243]. Когда возникает конфликт, образовавшее его подмножество узлов разбивается и каждое получающееся подмножество поступает в стек (т. е. каждая ячейка стека является подмножеством узлов); затем верхнее подмножество стека (т. е. последнее поступившее в стек подмножество) выбирается из стека и передается. В каждом окне список ожидающих подмножеств на рис. 4.9, рассматриваемый слева направо, соответствует последовательности ячеек стека, рассматриваемых начиная с верхней ячейки. Заметим, что в узле с задолженным пакетом можно определять момент передачи посредством счетчика, состояние которого совпадает с уровнем стека, на котором находится текущее подмножество, содержащее пакет. Когда пакет вступает в конфликт, в счетчик записывается 0 или 1 в зависимости от того, в какое подмножество попадает пакет. Когда счетчик переходит в состояние 0, пакет передается, а если счетчик в ненулевом состоянии, его значение увеличивается на 1 после каждого конфликта и уменьшается на 1 после каждой успешной передачи или пустого окна.

Остается одна проблема, связанная с этим древовидным алгоритмом, которая состоит в том, что делать с новыми поступающими пакетами, которые приходят во время разрешения конфликта. Период разрешения конфликта (ПРК) заканчивается по определению в момент окончания успешной передачи или в конце пустого окна, причем когда в стеке не остается ячеек с задолженными пакетами (т. е. в конце окна 9 на рис. 4.9). В этот момент времени начинается новый ПРК между пакетами, которые поступили во время предыдущего ПРК. Если предыдущему ПРК потребовалось большое число окон, что маловероятно, то новых ожидающих пакетов будет много, они образуют конфликт и будут продолжать образовывать конфликты до тех пор, пока подмножества не станут достаточно малыми в этом новом ПРК. Решение этой проблемы следующее: в конце ПРК множество узлов с новыми пакетами сразу разбивается на j подмножеств, где j выбирается таким, чтобы среднее число пакетов в подмножестве

было немного больше 1 (немного больше из-за того, что после конфликта скорость передачи временно повышается). Эти новые подмножества помещаются затем в стек, и начинается новый ПРК.

По существу, древовидный алгоритм теперь полностью определен. Каждый узел с пакетом, участвующим в текущем ПРК, следит за своим положением в стеке, как описано выше. Во всех узлах ведется учет числа пакетов в стеке и числа окон, прошедших с момента окончания предыдущего ПРК. После завершения этого ПРК в каждом узле определяется среднее число ожидающих новых пакетов, новое число j подмножеств, а в тех узлах, которые имеют новые ожидающие пакеты, случайно выбирается одно из этих j подмножеств и устанавливается счетчик в состояние, соответствующее положению в стеке.

Максимальная скорость передачи, которую обеспечивает этот алгоритм при оптимальном выборе j как функции среднего числа ожидающих пакетов, равна 0,43 пакета за окно [42]; мы опускаем весь анализ, поскольку далее опишем некоторые простые способы повышения скорости передачи.

Модификации древовидного алгоритма

Рассмотрим сначала ситуацию, показанную на рис. 4.10. Здесь в окнах 4 и 5 за конфликтом следует пустое окно; это означает, что все пакеты, участвующие в конфликте, перешли во второе подмножество. Древовидный алгоритм должен просто передавать второе подмножество, порождая неизбежный конфликт. Модификация состоит в том, что передача этого второго подмножества не производится, оно разбивается на два подмножества и передается первое из них. Аналогично если пустое окно появляется снова, то второе подмножество разбивается снова перед передачей и т. д.

Эту модификацию можно выразить в терминах стека и реализовать, используя счетчики, так же как исходный древовидный алгоритм. Каждый узел должен содержать дополнительную двоичную переменную состояния, которая принимает значение 1, если при некотором $i \geq 1$ последние i окон содержат конфликт, за которым следует $i - 1$ пустых окон; во всех остальных случаях переменная состояния принимает значение 0. Если в текущем окне обратная связь поставляет сигнал 0 и переменная состояния имеет значение 1, то значение этой переменной остается без изменения, а верхнее подмножество стека разбивается на два подмножества, которые помещаются в стек вместо исходного верхнего подмножества.

Использование этой модификации повышает максимальную скорость передачи до 0,46 пакетов за окно [169]. При использовании этой модификации на практике возникает незначительная проблема, связанная с тем, что если пустое окно ошибочно вос-

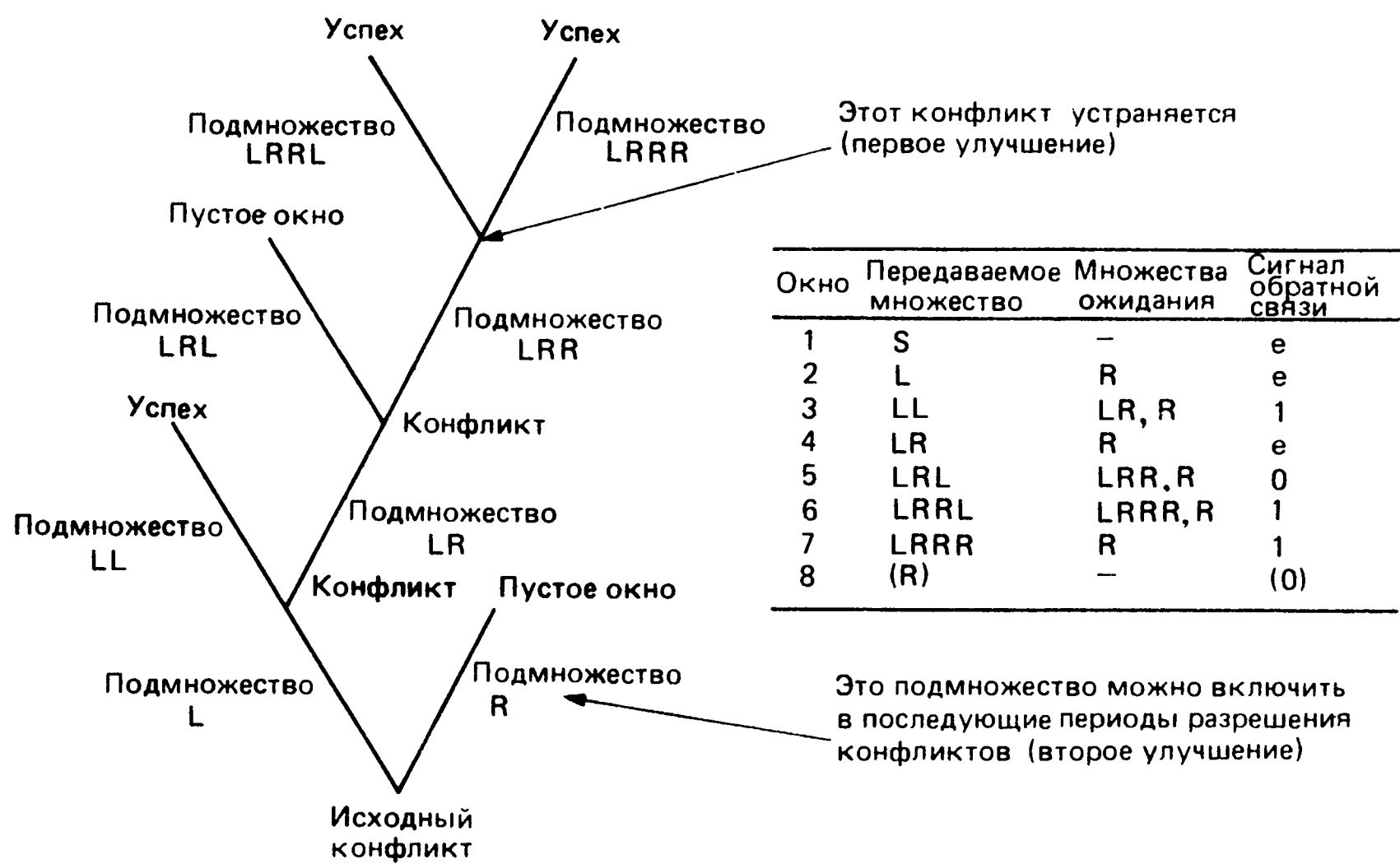


Рис. 4.10. Улучшения древовидного алгоритма. Подмножество LRR можно разбить без предварительной передачи его пакетов, поскольку сигналы обратной связи показывают, что оно содержит два пакета или более. Кроме того, пакеты подмножества R лучше объединить с новыми поступающими пакетами, так как число пакетов в нем является пуассоновской величиной с нежелательно малым средним значением.

принимается приемником как конфликт, то алгоритм продолжает разбиения до бесконечности, никогда более не совершая ни одной успешной передачи. Таким образом, чтобы алгоритм использовать на практике, его нужно усовершенствовать так, чтобы после некоторого числа h пустых окон, сопровождаемых разбиениями, передавалось следующее подмножество стека без его предварительного разбиения; если обратная связь достаточно надежная, то h может быть довольно большим, в противном случае h должно быть малым.

Следующая модификация древовидного алгоритма не только повышает скорость передачи, но и сильно упрощает анализ. Рассмотрим, что происходит при древовидном алгоритме в случае, когда за некоторым конфликтом следует еще один конфликт (см. окна 1 и 2 на рис. 4.10). Пусть x — число пакетов, образовавших первый конфликт, а x_L и x_R — числа пакетов в появившихся после деления подмножествах; таким образом, $x = x_L + x_R$. Предположим, что априори, т. е. до того как стало известно о возникновении конфликта, величина x является пуассоновской случайной величиной. Мысленно представим, что, перед тем как стало известно о конфликте, произведено разбиение этих x пакетов, например посредством подбрасывания монеты, на два

подмножества, содержащих x_L и x_R пакетов соответственно. Тогда априори x_L и x_R являются независимыми пуассоновскими случайными величинами, средние значения которых равны половине среднего значения x . Однако если известно о возникновении двух конфликтов, то x_L и x_R являются независимыми пуассоновскими величинами, на которые наложены условия $x_L + x_R \geq 2$ и $x_L \geq 2$. Из второго условия следует первое, поэтому первое можно опустить; это означает, что x_R при условии $x \geq 2$ продолжает оставаться пуассоновской со своим исходным безусловным распределением вероятностей. Задача 4.17 дает это утверждение в численной и менее абстрактной форме. Следовательно, вместо того чтобы отводить окно этому второму подмножеству, которое имеет нежелательно малое среднее число пакетов, лучше считать второе подмножество присоединенным к ждущим новым пакетам, которые еще не попадали в конфликт.

Когда эта идея используется для построения реального алгоритма, в результате получается алгоритм разбиения с передачей в порядке поступления (ППП), который является темой обсуждения в следующем подразделе. Перед этим обсуждением опишем несколько вариантов древовидного алгоритма.

Варианты древовидного алгоритма

Древовидный алгоритм, описанный выше, требует, чтобы все узлы следили за обратной связью канала и определяли, когда заканчивается каждый период разрешения конфликта. Это является недостатком в случае, когда приемники узлов отключаются, если нет пакетов для передачи. Один из способов избавления от этого недостатка, сохраняющий другие свойства древовидного алгоритма, состоит в том, что новые поступающие пакеты просто присоединяются к подмножеству узлов, находящихся в верхней ячейке стека [243, 249]. Таким образом, только те узлы, которые имеют задолженность в данный момент, должны следить за обратной связью канала. Подобные алгоритмы называются *неблокированными стек-алгоритмами*, что указывает на то, что новые поступающие пакеты не блокируются до окончания текущего периода разрешения конфликта. В противоположность этому древовидный алгоритм часто называется *блокированным стек-алгоритмом*.

В случае древовидного алгоритма новые поступающие пакеты разбиваются на переменное число подмножеств в начале каждого периода разрешения конфликта, а подмножества затем разбиваются на два подмножества после каждого конфликта. Однако в случае неблокированного стек-алгоритма новые поступающие пакеты постоянно добавляются к подмножеству в верхней ячейке стека и, следовательно, конфликты образуются в среднем не-

сколько большим числом пакетов. Из-за относительно большей вероятности образования конфликта тремя или более пакетами большей максимальной скорости передачи можно достичь, если разбивать множество вступивших в конфликт узлов на три, а не на два подмножества. Достижимая таким образом максимальная скорость передачи равна 0,40 [166].

4.3.2. Алгоритм разбиения в передаче в порядке поступления

Согласно второй модификации древовидного алгоритма, описанной выше, второе подмножество, участвующее в конфликте, рассматривается как никогда ранее не передававшееся, если первое подмножество содержит два или более пакетов. Вспомним также, что множество вступивших в конфликт пакетов можно разбить на два подмножества рядом способов, например подбрасывая монету, используя идентификаторы узлов или моменты поступления. В данном случае наиболее просто разбивать, используя момент поступления. Применяя этот подход, получаем, что каждое подмножество состоит из всех пакетов, которые поступили в течение некоторого заданного интервала времени; при возникновении конфликта этот интервал разбивается на два более коротких интервала. Передавая в первую очередь пакеты с более раннего интервала, алгоритм в итоге будет передавать успешно пакеты в порядке их поступления.

В каждый целочисленный момент времени k в окне k (т. е. в интервале от момента k до момента $k + 1$) должно передаваться множество всех пакетов, прибывших во время некоторого более раннего интервала, например от момента $T(k)$ до $T(k) + \alpha(k)$. Этот интервал называется *назначенным интервалом* для окна k (рис. 4.11). Можно считать, что пакеты, поступившие после $T(k) + \alpha(k)$, находятся в очереди, а те, что поступили между моментами $T(k)$ и $T(k) + \alpha(k)$, находятся на обслуживании. Особенность этой очереди в том, что число пакетов в ней неизвестно, хотя все узлы следят за тем, с какого назначенного интервала обслуживаются (т. е. передаются) пакеты.

Алгоритм разбиения с ППП определяется набором правил, который используется в узлах для вычисления $T(k)$ и $\alpha(k)$ для каждого последующего k на основе сигнала обратной связи в предыдущем окне. Эти правила являются модифицированными правилами ранее рассмотренного древовидного алгоритма для случая, когда при разбиении множеств пакетов используются моменты поступления.

На рис. 4.11 иллюстрируются эти правила. Когда возникает конфликт, например в окне k , назначенный интервал разбивается на два равных подынтервала и левый подынтервал L (т. е. тот, в котором пакеты ожидают дольше всех остальных) становится

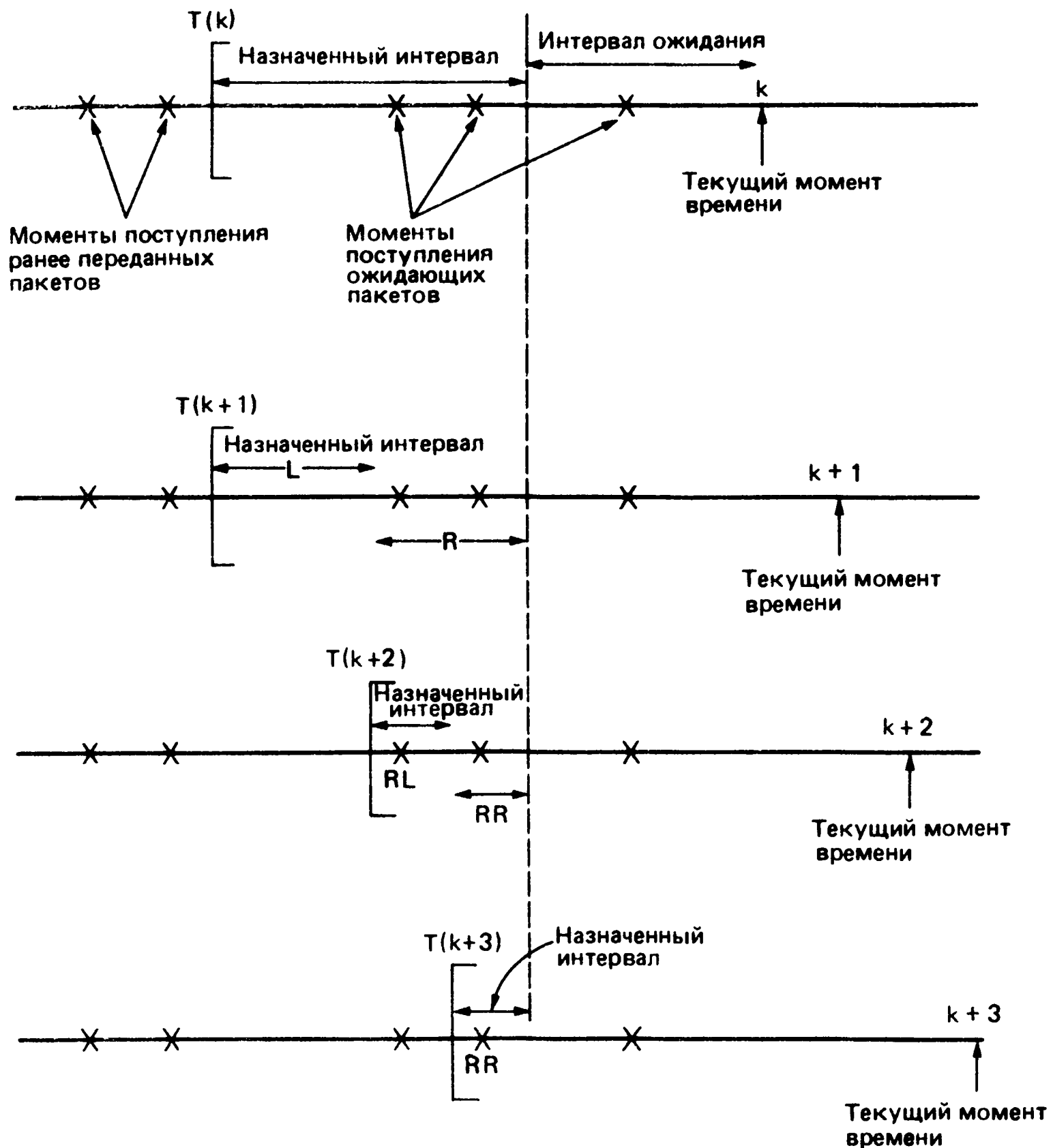


Рис. 4.11. Алгоритм разбиения с ППП. Пакеты передаются в порядке поступления. В случае возникновения конфликтов назначенный интервал, породивший конфликт, разбивается на два подынтервала, причем левому из них (т. е. содержащему ранее поступившие пакеты) назначается передавать в первую очередь.

назначенным интервалом для окна $k + 1$. Таким образом, $T(k + 1) = T(k)$ (т. е. левая граница не изменяется), а $\alpha(k + 1) = \alpha(k)/2$. Когда пустое окно, такое, как, например, окно $k + 1$, следует за конфликтом, применяется первая модификация для древовидного алгоритма. Известно, что предыдущий правый интервал R содержит два или более пакетов и поэтому он сразу разбивается, причем интервал RL становится назначенным интервалом для окна $k + 2$. Таким образом, $T(k + 2) = T(k + 1) + \alpha(k + 1)$, а $\alpha(k + 2) = \alpha(k + 1)/2$. В итоге успешные передачи происходят в окнах $k + 2$ и $k + 3$, завершая ПРК.

Теперь рассмотрим пример, иллюстрируемый на рис. 4.12. За конфликтом в окне k следует еще один конфликт в окне $k + 1$.

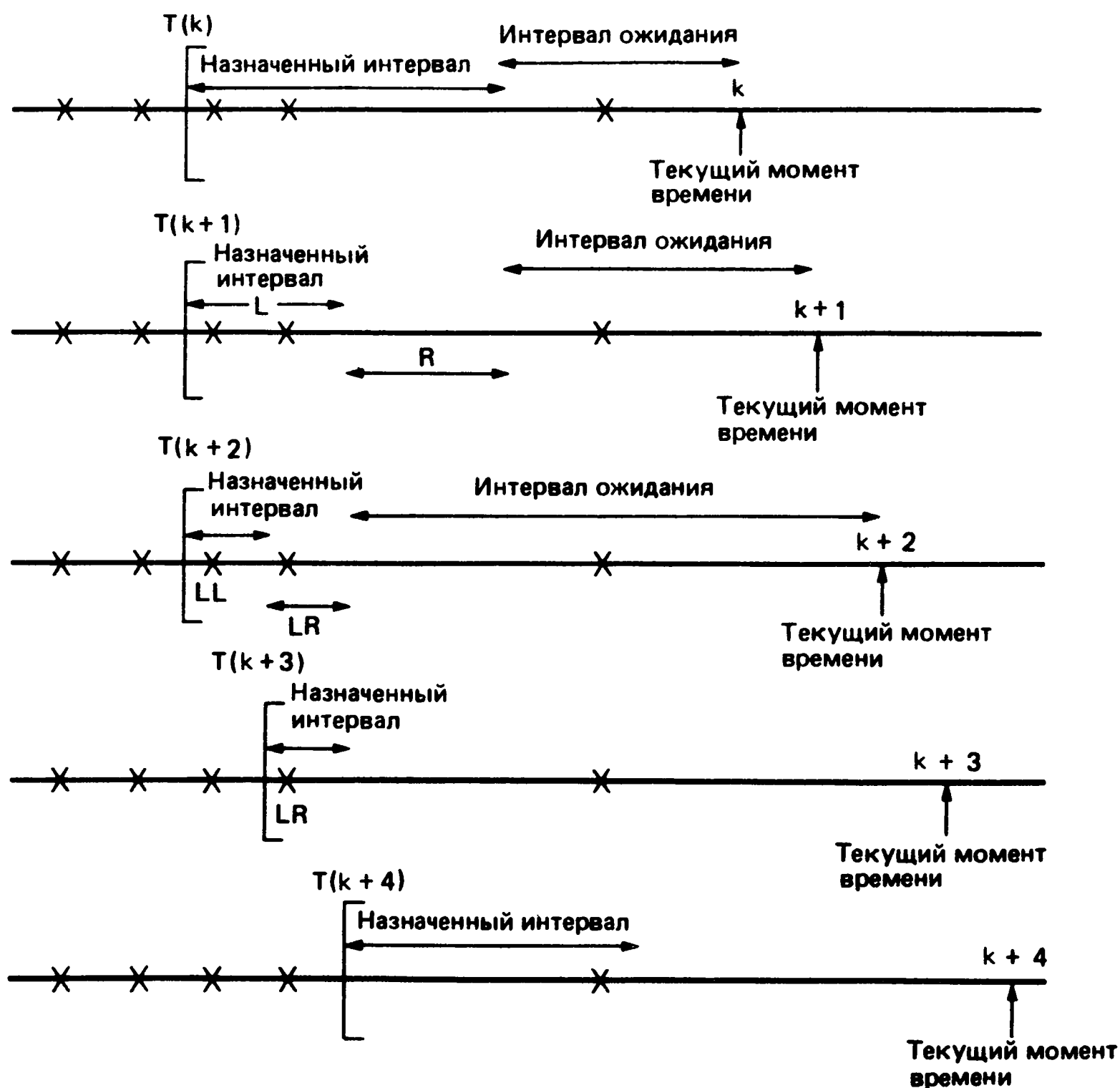


Рис. 4.12. Алгоритм разбиения с ППП. Когда за конфликтом следует еще один конфликт, интервал справа от того, в котором возник этот второй конфликт, снова становится частью ожидающего интервала. ПРК завершается в окне $k + 3$, и начинается новый ПРК с назначенным интервалом фиксированной длины.

Здесь применяется вторая модификация древовидного алгоритма. Так как интервал L содержит два или более пакетов, то конфликт в окне k ничего не говорит об интервале R и можно считать, что R никогда ранее не был частью назначенного интервала. Как показано на рисунке для окна $k + 2$, это само по себе является упрощением. Интервал L разбивается, причем LL становится следующим назначенным интервалом, а LR — ожидающим; алгоритм просто забывает об R . Когда пакеты с LL и LR успешно передаются в окнах $k + 2$ и $k + 3$, ПРК заканчивается.

При древовидном алгоритме в конце ПРК все ожидающие пакеты разбиваются на некоторое число подмножеств. Поскольку разбиение выполняется путем назначения временных интервалов, удобнее просто выбрать новый назначенный интервал некоторой

определенной длины, обозначаемой через α_0 , о которой будет идти речь в дальнейшем. Заметим, что в этот новый интервал для окна $k + 4$ входит старый интервал R , который теперь уже теряет свою «индивидуальность» и не считается отдельным интервалом.

В терминах дерева, образованного ожидающими множествами, следствием того, что правый интервал теряет индивидуальность всякий раз, когда соответствующий левый интервал разбивается, является отсечение дерева так, чтобы оно никогда не имело более двух наиболее удаленных от корня вершин (соответственно в терминах стека это означает, что стек никогда не помнит содержимого более чем двух верхних ячеек). Всякий раз, когда назначенным интервалом является левое подмножество разбиения, имеется соответствующее правое подмножество, пакеты которого, возможно, нужно будет передать впоследствии. Наоборот, когда назначенным интервалом является правое подмножество, не имеется других ожидающих интервалов. Таким образом, во время работы этого алгоритма в узлах нужно помнить только местонахождение назначенного интервала и то, что он левый или правый. Принято считать, что начальный интервал ПРК является правым интервалом. Теперь можно описать алгоритм, который строго выполняется каждым узлом сети. Алгоритм задает назначенный интервал (т. е. $T(k)$ и $\alpha(k)$) и статус ($\sigma = L$ или $\sigma = R$) для окна k на основе сигнала обратной связи, назначенного интервала и статуса, относящихся к окну $k - 1$.

Если сигнал обратной связи равен e , то

$$\begin{aligned} T(k) &= T(k - 1), \\ \alpha(k) &= \alpha(k - 1)/2, \\ \sigma(k) &= L. \end{aligned} \quad (4.15)$$

Если сигнал обратной связи равен 1 и $\sigma(k - 1) = L$, то

$$\begin{aligned} T(k) &= T(k - 1) + \alpha(k - 1), \\ \alpha(k) &= \alpha(k - 1), \\ \sigma(k) &= R. \end{aligned} \quad (4.16)$$

Если сигнал обратной связи равен 0 и $\sigma(k - 1) = L$, то

$$\begin{aligned} T(k) &= T(k - 1) + \alpha(k - 1), \\ \alpha(k) &= \alpha(k - 1)/2, \\ \sigma(k) &= L. \end{aligned} \quad (4.17)$$

Если сигнал обратной связи равен 0 или 1 и $\sigma(k - 1) = R$, то

$$\begin{aligned} T(k) &= T(k - 1) + \alpha(k - 1), \\ \alpha(k) &= \min [\alpha_0, k - T(k)], \\ \sigma(k) &= R. \end{aligned} \quad (4.18)$$

Последняя инструкция (4.18) используется в конце разрешения конфликта или тогда, когда конфликты не возникают. Длина нового назначенного интервала в этом случае является некоторой константой α_0 , которую можно выбирать так, чтобы или минимизировать задержку при заданной скорости поступления, или максимизировать скорость устойчивой передачи. Естественно, при иссякании очереди назначенный интервал не может содержать пакеты, которые еще не поступили, так что длина интервала не превышает $k - T(k)$, что обеспечивается операцией взятия минимума в (4.18).

Анализ алгоритма разбиения с ППП

Рисунок 4.13 помогает представить течение периода разрешения конфликта; мы увидим в дальнейшем, что эту диаграмму можно интерпретировать как цепь Маркова. Левый узел диаграммы соответствует начальному окну ПРК; узел разделен на две части для того, чтобы наглядно выделить моменты начала и конца ПРК. Если появляется пустое окно или имеет место успешная передача, то ПРК сразу заканчивается и начинается новый ПРК в следующем окне. Однако если возникает конфликт, то совершается переход в узел $(L, 1)$; L обозначает статус, а 1 указывает на то, что произошло одно разбиение назначенного интервала.

Каждое последующее пустое окно или конфликт в левом назначенном интервале приводит к дополнительному разбиению и более короткому левому назначенному интервалу, что соответствует переходу на диаграмме из (L, i) в $(L, i + 1)$, где i — число разбиений исходного назначенного интервала. Успешная передача в левом интервале приводит к проверке правого интервала, причем без дополнительного разбиения, что соответствует пере-

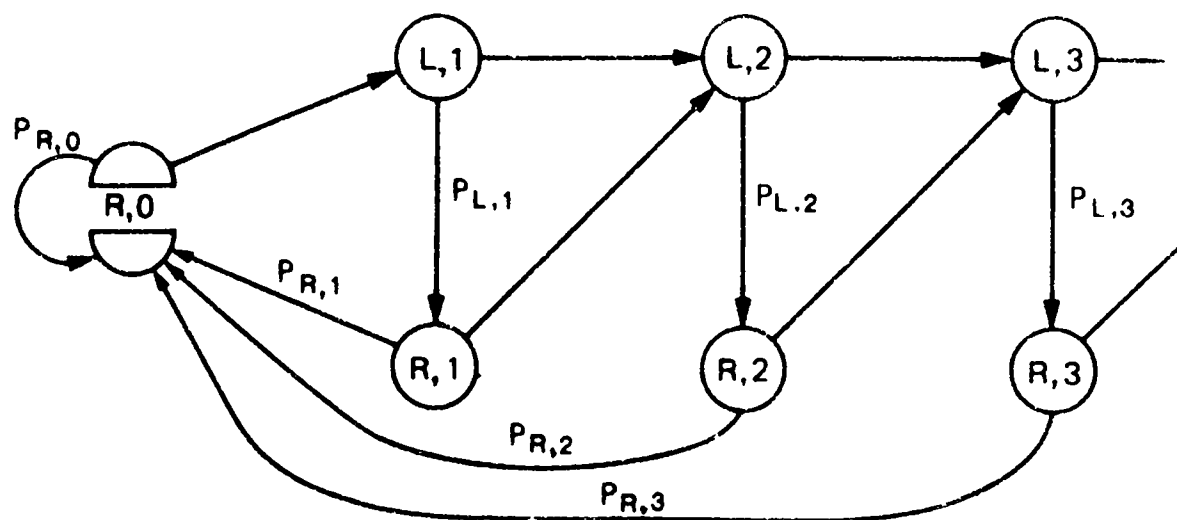


Рис. 4.13. Цепь Маркова для алгоритма разбиения с ППП. Переходы в верхние состояния происходят после разбиения интервала и соответствуют передаче пакетов из левой части этого интервала. Переходы в нижние состояния происходят после успешной передачи в левой части интервала и нижние состояния соответствуют передаче пакетов из правой части. Переходы сверху вниз и из нижних состояний назад в $R, 0$ соответствуют успешным передачам.

ходу из (L, i) в (R, i) . Успешная передача в правом интервале завершает ПРК, что отображается переходом назад в $(R, 0)$, тогда как конфликт вызывает новое разбиение и переход из (R, i) в $(L, i + 1)$.

Теперь проведем анализ ПРК. Предположим, что длина исходного назначенного интервала равна α_0 . Каждое разбиение назначенного интервала уменьшает его длину в 2 раза и, следовательно, узлы (L, i) и (R, i) на диаграмме соответствуют назначенным интервалам длины $2^{-i}\alpha_0$. Так как мы предполагаем, что поступающий поток пакетов является пуассоновским с параметром λ , то число пакетов в исходном назначенном интервале является пуассоновской случайной величиной со средним $\lambda\alpha_0$. Аналогично априорные распределения вероятностей числа пакетов на непересекающихся подынтервалах являются независимыми и пуассоновскими. Определим G_i как априорное среднее число пакетов в интервале, который получается после i разбиений

$$G_i = 2^{-i}\lambda\alpha_0. \quad (4.19)$$

Теперь вычислим вероятности переходов на рис. 4.13 и покажем, что они составляют цепь Маркова, т. е. что каждый переход не зависит от пути, который привел в данный узел. Заметим, что мы интересуемся (сейчас) только одним периодом разрешения конфликта; мы считаем верхнюю половину узла $(R, 0)$ начальным состоянием, а нижнюю половину — конечным состоянием. Начинаем слева и продвигаемся направо. $P_{R,0}$ обозначает вероятность пустого окна или успешной передачи (т. е. нулевого числа пакетов или одного пакета) в первом окне. Так как число пакетов на исходном назначенном интервале имеет пуассоновское распределение со средним G_0 , то вероятность нулевого числа пакетов или одного пакета равна

$$P_{R,0} = (1 + G_0) e^{-G_0}. \quad (4.20)$$

Теперь рассмотрим состояние $(L, 1)$. Переход в это состояние происходит после конфликта в состоянии $(R, 0)$, который возникает с вероятностью $1 - P_{R,0}$. Пусть x_L обозначает число пакетов в новом назначенном интервале L (т. е. в левой половине исходного интервала), а x_R — число пакетов в новом назначенном интервале R (в правой половине исходного интервала). Априори x_L и x_R — это независимые пуассоновские случайные величины со средним G_1 каждая. Условием перехода в состояние $(L, 1)$ является $x_L + x_R \geq 2$. Вероятность успешной передачи $P_{L,1}$ равна, таким образом, вероятности того, что $x_L = 1$, при условии, что $x_L + x_R \geq 2$. Замечая, что как $x_L = 1$, так и $x_L + x_R \geq 2$ только тогда, когда $x_L = 1$ и $x_R \geq 1$, имеем

$$P_{L,1} = \frac{P\{x_L = 1\} P\{x_R \geq 1\}}{P\{x_L + x_R \geq 2\}} = \frac{G_1 e^{-G_1} [1 - e^{-G_1}]}{1 - (1 + G_0) e^{-G_0}}. \quad (4.21)$$

Состояние $(R, 1)$ посещается, если только предыдущий переход происходит, т. е. если $x_L = 1$ и $x_R \geq 1$. Следовательно, вероятность успешной передачи $P_{R,1}$ в состоянии $(R, 1)$ равна

$$P_{R,1} = \frac{P\{x_R = 1\}}{P\{x_R \geq 1\}} = \frac{G_1 e^{-G_1}}{1 - e^{-G_1}}. \quad (4.22)$$

Теперь покажем, что равенства (4.21) и (4.22) обобщаются для вычисления $P_{L,i}$ и $P_{R,i}$ при всех $i \geq 1$, т. е.

$$P_{L,i} = \frac{G_i e^{-G_i} (1 - e^{-G_i})}{1 - (1 + G_{i-1}) e^{-G_{i-1}}}, \quad (4.23)$$

$$P_{R,i} = \frac{G_i e^{-G_i}}{1 - e^{-G_i}}. \quad (4.24)$$

Рассмотрим состояние $(L, 2)$. Его можно посетить после конфликта в состоянии $(L, 1)$, пустого окна в $(L, 1)$ или конфликта в $(R, 1)$. В первом случае интервал L разбивается на LL и LR , и LL становится новым назначенным интервалом. Во втором и третьем случаях R разбивается на RL и RR , причем RL становится новым назначенным интервалом. Имея в виду первый случай, обозначим через x_{LL} и x_{LR} числа пакетов в LL и LR соответственно. Априори они являются независимыми пуассоновскими случайными величинами со средним G_2 каждая. Конфликт в $(L, 1)$ означает, что $x_L + x_R \geq 2$ и $x_L \geq 2$, что эквивалентно единственному условию $x_L \geq 2$. Ситуация, таким образом, аналогична той, когда вычислялась $P_{L,1}$, за исключением того, что интервалы короче в два раза, так что $P_{L,2}$ в (4.23) в этом случае вычисляется правильно.

Теперь рассмотрим второй случай, когда в $(L, 1)$ появляется пустое окно. Это означает, что $x_L + x_R \geq 2$ и $x_L = 0$, что эквивалентно $x_R \geq 2$ и $x_L = 0$. В этом случае $P_{L,2}$ равна вероятности того, что RL , левая половина R , содержит один пакет, при условии, что R содержит по крайней мере два; опять видим, что равенство (4.23) справедливо. Наконец, в случае конфликта в $(R, 1)$ имеем $x_L + x_R \geq 2$, $x_L = 1$, и $x_R \geq 2$ или, что эквивалентно, $x_L = 1$, $x_R \geq 2$; равенство (4.23) опять применимо для $(L, 2)$. Таким образом, марковское свойство выполняется для $(L, 2)$.

Вообще независимо от того, как осуществляется переход в $(L, 2)$, данный интервал L или R , который проверялся перед переходом в $(L, 2)$, является интервалом длины $\alpha_0/2$ с пуассоновским потоком, на который накладывается условие, что данный интервал содержит два или более пакетов. Если успех возникает в левой половине, то число пакетов в правой половине имеет условное пуассоновское распределение вероятностей с условием, что

это число не меньше одного пакета, откуда следует выражение (4.24) для $P_{R,2}$. Это рассуждение повторяется для $i = 3, 4, \dots$ (что можно сделать более формально, применяя индукцию). Таким образом, на рис. 4.13 показана цепь Маркова и равенства (4.20), (4.23) и (4.24) задают вероятности переходов.

Анализ этой цепи особенно прост по той причине, что ни одно состояние не посещается более одного раза за время до возврата в $(R, 0)$. Вероятности $p(L, i)$ и $p(R, i)$ того, что (L, i) и (R, i) соответственно посещаются до возвращения в $(R, 0)$, можно вычислить итерацией, начиная с исходного состояния $(R, 0)$:

$$p(L, 1) = 1 - P_{R,0}, \quad (4.25)$$

$$p(R, i) = P_{L,i} p(L, i), \quad i \geq 1, \quad (4.26)$$

$$p(L, i+1) = (1 - P_{L,i}) p(L, i) + (1 - P_{R,i}) p(R, i), \quad i \geq 1. \quad (4.27)$$

Пусть K обозначает число окон в ПРК; таким образом, K — число последовательно посещаемых состояний, включая исходное состояние $(R, 0)$, до возвращения в $(R, 0)$

$$E\{K\} = 1 + \sum_{i=1}^{\infty} [p(L, i) + p(R, i)]. \quad (4.28)$$

Мы также должны оценить величину изменения $T(k)$ за время одного РПК. Так как предполагается, что исходный интервал имеет длину α_0 , то это изменение не превышает α_0 , но если в левых интервалах возникают конфликты, то соответствующие правые интервалы возвращаются к ожидающему интервалу и изменение меньше α_0 . Пусть f обозначает долю α_0 , которая вернулась за время ПРК, так что $\alpha_0(1 - f)$ является изменением $T(k)$. Вероятность конфликта в состоянии (L, i) равна вероятности того, что левая половина интервала в состоянии (L, i) содержит не менее двух пакетов при условии, что правый и левый интервалы вместе содержат не менее двух; т. е.

$$P\{e | (L, i)\} = \frac{1 - (1 + G_i) e^{-G_i}}{1 - (1 + G_{i-1}) e^{-G_{i-1}}}. \quad (4.29)$$

Доля исходного интервала, вернувшаяся после такого конфликта, равна 2^{-i} , так что среднее значение f равно

$$E\{f\} = \sum_{i=1}^{\infty} p(L, i) P\{e | (L, i)\} 2^{-i}. \quad (4.30)$$

Заметим, что $E\{f\}$ и $E\{K\}$ — функции только $G_i = \lambda \alpha_0 2^{-i}$, где $i \geq 1$, и, следовательно, — функции только произведения $\lambda \alpha_0$. При увеличении i вероятность $P_{L,i}$ стремится к $1/2$, и, та-

ким образом, $p(L, i)$ и $p(R, i)$ стремятся к нулю с ростом i как 2^{-i} . Таким образом, зависимость $E\{f\}$ и $E\{K\}$ от $\lambda\alpha_0$ легко оценить численно.

Наконец, определим снос $D = k - T(k)$ как среднее изменение задолженности на временной оси за время ПРК (снова предполагая, что исходный назначенный интервал имеет длину α_0). Снос равен среднему числу окон в ПРК минус среднее изменение $T(k)$, т. е.

$$D = E\{K\} - \alpha_0 (1 - E\{f\}). \quad (4.31)$$

Снос отрицателен, если $E\{K\} < \alpha_0 (1 - E\{f\})$ или, что эквивалентно, если

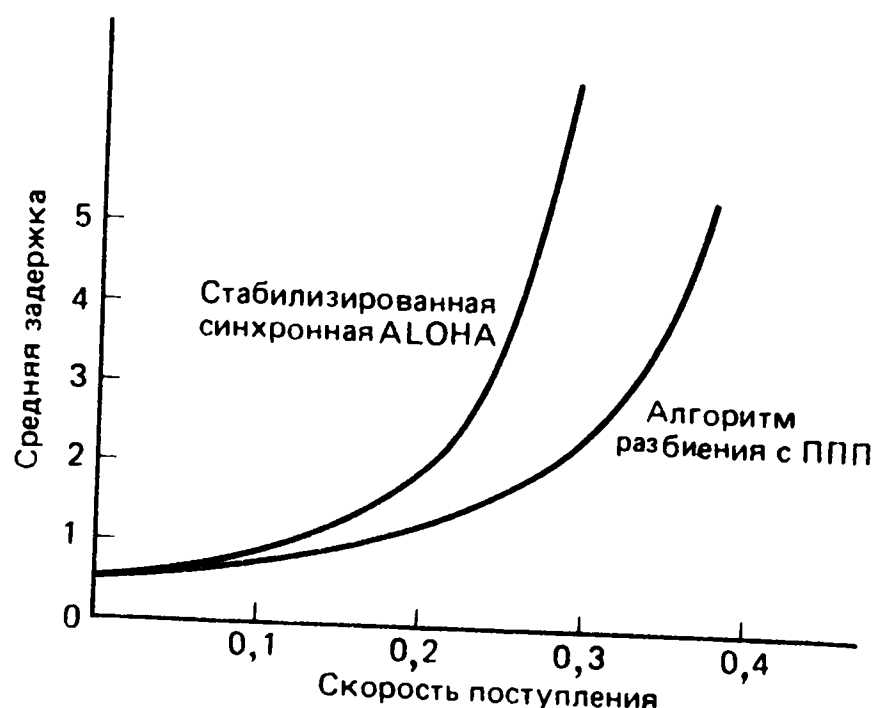
$$\lambda < \frac{\lambda\alpha_0 (1 - E\{f\})}{E\{K\}}. \quad (4.32)$$

Численно найдено, что максимум правой части (4.32) как функции $\lambda\alpha_0$ равен 0,4871 и достигается при $\lambda\alpha_0 = 1,266$. Выражение $\lambda\alpha_0$ определяет среднее число пакетов, находящихся в исходном назначенном интервале; оно несколько больше единицы (при которой максимизируется вероятность успеха в первом окне ПРК), так как вероятность успеха увеличивается непосредственно после конфликта. Если α_0 полагается равным 2,6 (т. е. величине, равной $1,266/0,4871$), то соотношение (4.32) выполняется при всех $\lambda < 0,4871$. Таким образом, средняя задолженность на временной оси уменьшается (всякий раз, когда ранее она была больше α_0) и мы заключаем¹⁾, что алгоритм устойчив при $\lambda < 0,4871$.

Анализировать среднюю задержку намного труднее, чем максимальную скорость устойчивой передачи. Сложные верхняя и нижняя границы были построены в работах [119, 240]; они хорошо согласуются с результатами моделирования. На рис. 4.14 показан график этой задержки и для сравнения дан соответствующий график стабилизированной синхронной АЛОНА.

¹⁾ Математикам желательно представить более строгое доказательство устойчивости. Определим период занятости как последовательность ПРК, начинающуюся таким ПРК, у которого задолженность $k - T(k) < \alpha_0$, и заканчивающуюся в момент начала следующего ПРК с $k - T(k) < \alpha_0$. Последовательность задолженностей в моменты начала ПРК, составляющих период занятости, представляет случайное блуждание, у которого приращения (за исключением первого) имеют одинаковые распределения вероятностей с отрицательными средними значениями при $\lambda < 0,4871$. Поскольку $p(L, i)$ стремятся к 0 по экспоненциальному закону с ростом i , то приращения имеют производящую функцию моментов и из тождества Вальда следует, что число ПРК в периоде занятости N также имеет производящую функцию моментов. Поскольку число окон в периоде занятости не более $N\alpha_0$, то число окон имеет производящую функцию моментов; отсюда следует, что средняя задержка пакета конечна.

Рис. 4.14. Сравнение средней задержки стабилизированной синхронной АЛОНА и алгоритма разбиения с ППП как функции скорости поступления пакетов. У первого алгоритма задержка растет неограниченно, когда скорость поступления приближается к $1/e$, а у второго — когда скорость поступления приближается к 0,4871.



Модификации алгоритма разбиения с ППП

Разбиение интервала на подынтервалы равной длины не является оптимальным, так как в этом случае максимальная скорость устойчивой передачи немного меньше оптимального значения. Когда каждый интервал разбивается на подынтервалы оптимальной длины, максимальная скорость передачи повышается до 0,4878 [176, 242]. Другая модификация, приводящая к еще меньшему повышению на $3,6 \times 10^{-7}$, состоит в том, что в состоянии (R, i) при больших i некоторая часть ожидающего интервала добавляется к правому интервалу [248]. Хотя эти модификации не имеют практического значения, они представляют теоретический интерес при определении оптимальной максимальной скорости устойчивой передачи.

Максимальная скорость устойчивой передачи, которую можно достичь в модели с предположениями 1—6б, в настоящее время неизвестна. Большое число исследований посвящено отысканию верхних границ скорости передачи; наилучшая такая граница равна 0,587 [175]. Таким образом, максимум скоростей устойчивой передачи всех алгоритмов лежит между 0,4878 и 0,587.

Эти вопросы о максимальной скорости передачи сильно зависят от предположений 1 — 6б. Как было показано, при любом конечном множестве m узлов ВУ позволяет тривиально достичь любой скорости передачи вплоть до одного пакета в окно. Эта поразительная разница между конечным и бесконечным m кажется парадоксальной до тех пор, пока мы не узнаем, что при ВУ средняя задержка (при данной интенсивности λ) увеличивается линейно с ростом m , тогда как алгоритмы в предположении, что $m = \infty$, имеют ограниченную задержку независимо от m .

Мы увидим также в следующих двух разделах, что значительно большей, чем 0,4878, скорости передачи можно достичь, если изменить предположение о синхронности и воспользоваться обратной связью в случаях, когда канал пуст или в нем возник конф-

ликт. Наконец, довольно удивительно то, что если обратную связь усилить, с тем чтобы сообщалось число пакетов в каждом конфликте, то максимальная скорость устойчивой передачи снова увеличится до 1 [190]. К сожалению, этот тип обратной связи трудно реализовать на практике и даже если бы такая обратная связь существовала, неизвестен ни один алгоритм, обеспечивающий такую высокую скорость передачи.

Практические детали

Алгоритму разбиения с ППП свойственна та же проблема записывания, что и первой модификации древовидного алгоритма; записывание возникает, когда сигнал обратной связи в пустом окне ошибочно воспринимается как сигнал о конфликте. Как и ранее, это записывание устраняется введением в алгоритм максимального числа h последовательных повторений инструкции (4.17). При появлении $(h + 1)$ -го последовательного пустого окна после конфликта выполняется инструкция (4.16).

Кроме того, предполагается, что в узлах время поступления пакета может измеряться с неограниченной точностью. На практике времена поступления измеряются с точностью до конечного числа битов и, если это необходимо для разбиения, каждый узел может генерировать дополнительные биты с помощью псевдослучайного генератора чисел ¹⁾, ²⁾.

Алгоритм разбиения с передачей в обратном порядке (ПОП)

Алгоритм разбиения с ППП требует, чтобы все узлы постоянно следили за обратной связью канала. Недавно разработанная модификация позволяет узлам следить за обратной связью только после того, как они получают пакет для передачи [93, 123]. Идея состоит в том, что пакеты в основном передаются в соответствии с дисциплиной передачи в обратном порядке (ПОП); значит, самым последним поступившим пакетам нет необходимости знать длину очереди, так как они имеют самый высокий приоритет передачи.

Рисунок 4.15 иллюстрирует работу этой разновидности алгоритма. Новые поступившие пакеты находятся в состоянии предварительного ожидания до тех пор, пока они не получают количество информации обратной связи, достаточное для обнаружения конца ПРК; затем они присоединяются к ожидающему множеству узлов. Окончание ПРК можно определить по сигналу обратной

¹⁾ При такой псевдослучайной генерации нарушается принцип ППП. — *Прим. ред.*

²⁾ Среди практических деталей следует также отметить, что оптимальное значение α_0 зависит от λ . — *Прим. ред.*

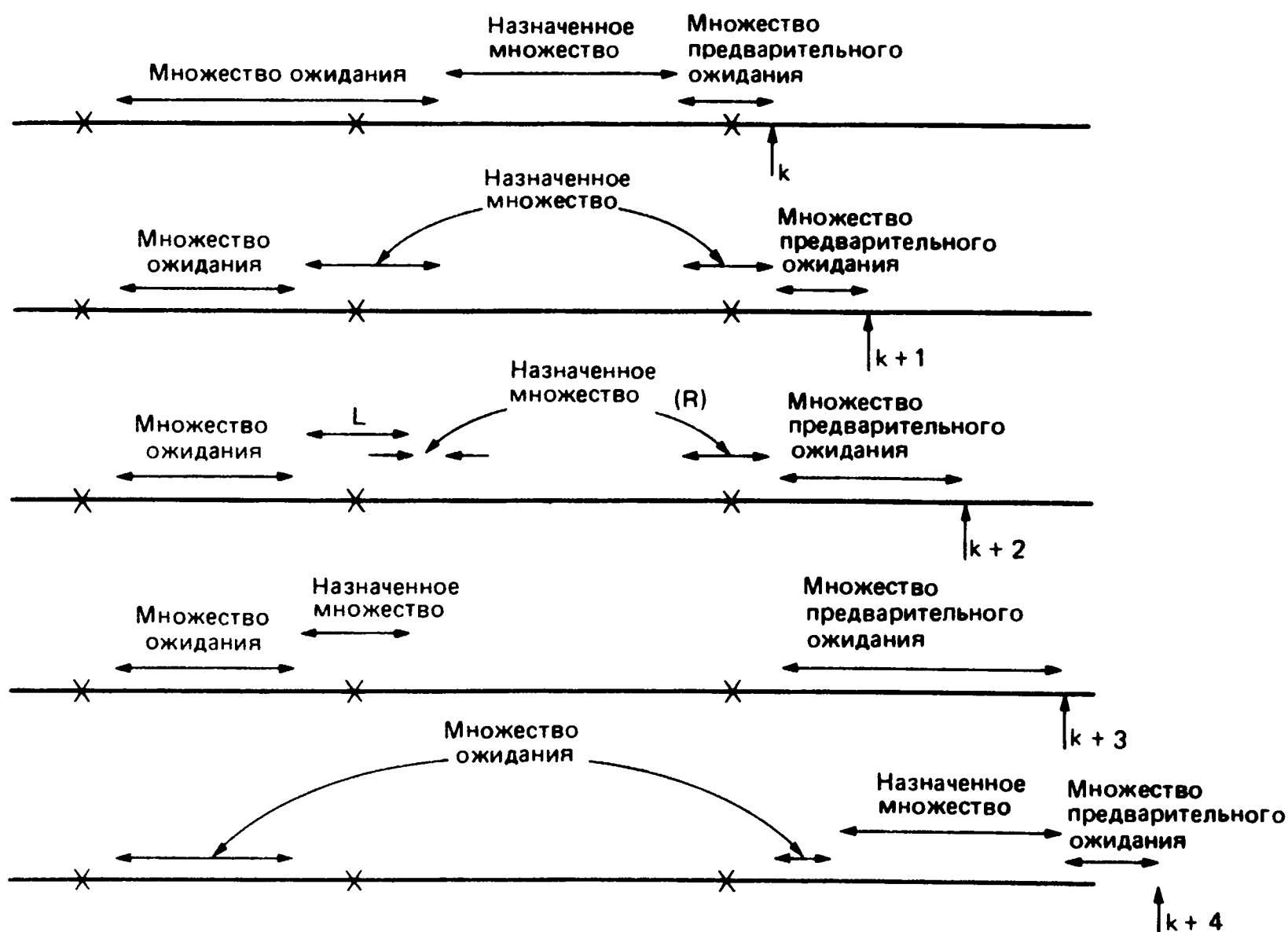


Рис. 4.15. Алгоритм разбиения с ПОП. Поступающие пакеты ожидают начала нового ПРК, но назначение производится в соответствии с дисциплиной «последний поступил, первый передается». Назначенные множества и ожидающие множества могут состоять из более чем одного интервала.

связи 1, за которым следует сигнал обратной связи 0 или 1. Если используется модификация, предназначенная для практического применения, согласно которой конструкция (4.17) может последовательно повторяться не более h раз, последовательное появление h нулей, за которыми следует 1 или 0, также означает окончание ПРК.

После того как ожидающее множество дополнилось справа интервалом времени, который, быть может, содержит узлы, обнаружившие конец ПРК, выбирается новое назначенное множество в правом конце ожидающего множества (таким образом, в него входит весь новый интервал или его часть). Как показано на рисунке, ожидающее множество, а следовательно, и назначенное множество, может состоять из нескольких отдельных интервалов.

Узлы, имеющие задолженность, после присоединения к ожидающему множеству следят за расстоянием между моментом поступления их пакета и правым краем ожидающего множества. При определении этого расстояния учитываются только интервалы, про которые не было вынесено решение, и (левые и правые) интервалы, которые не получали назначения. Узлы, принадлежащие назначенному множеству, аналогичным образом следят

за их расстояниями до правого края назначенного множества. Когда возникает конфликт, назначенное множество разбивается так же, как и ранее, но здесь следующее назначение получает правая половина. В случае конфликта в правой половине соответствующая левая половина присоединяется к правому краю ожидающего множества, увеличивая расстояние от правого края предыдущих ожидающих узлов. В конце ПРК новый поступивший интервал, для которого это событие является обнаруживаемым, сначала добавляется к правому краю ожидающего множества, а затем новое назначенное множество отсекается справа. В узлах неизвестно, где находится левый край ожидающего множества и, поэтому назначенное множество всегда имеет длину α_0 (без учета пробелов) и, возможно, фиктивный участок слева.

Когда задолженность велика, алгоритм разбиения с ПОП имеет такой же снос, как и алгоритм дробления с ППП, и, следовательно, он устойчив в том же диапазоне λ . При увеличении h (где h — допустимое число повторений инструкции (4.17) алгоритма) верхняя граница этого диапазона, как ранее, стремится к 0,4871. Средняя задержка несколько больше, чем при ППП, в основном из-за времени, которое пакеты проводят в состоянии предварительного ожидания.

Обратная связь с задержкой

Предположим, что сигнал обратной связи k -го окна принимается узлами в интервале времени от начала окна $k + j - 1$ до начала окна $k + j$ при некотором фиксированном $j \geq 1$. Представим, что канал совместно используется на основе временного уплотнения j различными версиями алгоритма дробления с ППП, за исключением того, что все j алгоритмов имеют общее ожидающее множество. Каждый узел следит за состоянием каждого из j текущих ПРК и длиной ожидающего множества. В конце ПРК одной из j версий назначенное множество длины α_0 (или меньше, если ожидающее множество меньше) отсекается слева от ожидающего множества, чтобы начать следующий ПРК этой версии. Поскольку j версий имеют различные задержки, то пакеты в этом случае передаются не в порядке их поступления.

Когда правое подмножество возвращается к ожидаемому множеству из-за конфликта в левом подмножестве в одной из j версий, оно присоединяется к левому краю ожидающего множества. При этом ожидающее множество можно расчленить на отдельные интервалы, но узлы должны следить только за длиной множества без учета пробелов.

Анализ, аналогичный проведенному ранее, показывает, что максимальная скорость устойчивой передачи при любом конечном j по-прежнему равна 0,4871. Средняя задержка больше, чем у

алгоритма разбиения с ППП, потому что требуется дополнительное время для разрешения конфликтов. Заметим, что, по существу, задержку можно считать состоящей из двух частей: 1) задержки при разрешении конфликтов, которая увеличивается почти линейно с ростом j , и 2) времени ожидания в ожидающем множестве, которое слабо зависит от j . При большом j и малой λ отсюда следует, что α_0 нужно уменьшать, уменьшая таким образом частоту конфликтов за счет некоторого увеличения длины ожидающего множества.

Алгоритм разбиения в порядке кругового опроса

Последний вариант алгоритма разбиения с ППП [117] можно применять тогда, когда имеется конечное множество идентифицируемых узлов, пронумерованных числами от 1 до m . Будем считать, что узлы образуют логическое кольцо, в котором узел $i + 1$ следует за узлом i , $1 \leq i < m$, а узел 1 следует за узлом m . Для формирования назначенных множеств не используются моменты поступления пакетов; вместо этого назначенное множество составляется из множества смежных узлов, имеющих, например, номера от i до j в кольце. После завершения ПРК алгоритм назначает следующее множество узлов, следующих друг за другом в кольце. Длины назначенных множеств, инициирующих ПРК, изменяются через время обхода кольца, с тем чтобы при слабой нагрузке исходное назначенное множество содержало, как правило, все узлы, а при сильной нагрузке исходные назначенные множества сокращались, как правило, до одного узла, что эквивалентно ВУ.

4.4. Прослушивание несущей

Во многих системах с множественным доступом, таких как локальные сети, узел может услышать, передают ли другие узлы, через время распространения и обнаружения, которое очень мало по сравнению с временем передачи пакета. Время обнаружения — это время, необходимое приемнику для определения того, осуществляется ли в данный момент времени передача каким-либо другим узлом. Это время несколько отличается, во-первых, от времени обнаружения начала новой передачи, во-вторых, от времени синхронизации приема новой передачи, и, в-третьих, от времени обнаружения конца старой передачи. Мы не учитываем эти и другие тонкости физического уровня в дальнейшем и просто рассматриваем передающую среду как прерывисто-синхронный битовый тракт с множественным доступом, пустые периоды которого можно отличать (с задержкой) от периодов передачи пакетов.

Если узлы могут быстро обнаруживать пустые периоды, то естественно после обнаружения быстро прервать пустой период

и разрешить узлам начинать передачу пакетов. Этот метод называется множественным доступом с прослушиванием несущей (МДПН) [149]; при этом необязательно подразумевается использование несущей, имеется в виду способность быстро обнаруживать пустые периоды.

Пусть β — время распространения и обнаружения (единицей измерения времени считается время передачи пакета), необходимое для того, чтобы все источники обнаружили пустой канал после того, как передача закончилась. Таким образом, если τ обозначает это время в секундах, C — скорость передачи по каналу в битах в секунду, а L — среднее число битов в пакете данных, то

$$\beta = \frac{\tau C}{L}. \quad (4.33)$$

Мы увидим, что характеристики МДПН ухудшаются с ростом β и, следовательно, ухудшаются также при увеличении скорости передачи по каналу и при уменьшении длины пакета.

Рассмотрим синхронную систему, в которой окно, в котором ничего не передается, заканчивается через β единиц времени и начинается новое окно. Это предположение о делении пустых периодов на окна длины β не реально, но создает простую модель, позволяющую многое понять. Таким образом, мы отказываемся от нашего предыдущего предположения о том, что временная ось разбита на окна равной длины. Мы также отказываемся от предположения, что все пакеты данных имеют одинаковую длину, хотя мы по-прежнему сохраняем временную нормировку, при которой среднее время передачи пакета равно 1. Вместо предположения о быстрой обратной связи мы предполагаем, что сигналы 0, 1, с обратной связью поступают с задержкой, не превышающей β , как указывалось выше. В целях упрощения мы продолжаем считать, что множество узлов бесконечно и поступающие потоки — пуассоновские с суммарной интенсивностью λ . Мы сначала модифицируем синхронную АЛОНА для этой новой модели, затем рассмотрим несинхронные системы и алгоритмы разбиения с ППП.

4.4.1. Синхронная АЛОНА с МДПН

Главное отличие синхронной АЛОНА с МДПН от обычной синхронной АЛОНА состоит в том, что пустые окна при МДПН имеют длительность β . Другое отличие заключается в том, что если пакет поступает в узел во время текущей успешной передачи, то он считается задолженным и его передача начинается с вероятностью q , после каждого последующего пустого окна; пакеты, поступившие во время пустого окна, передаются, как обычно, в следующем окне. Этот метод в работе [149] был назван ненастойчивым МДПН, чтобы отличить его от двух других методов: настоячивого МДПН и P -настоячивого МДПН. При настоячи-

вом МДПН передача всех поступающих во время занятого окна пакетов откладывается до окончания этого окна, что вызывает конфликт с относительно большой вероятностью. При P -настойчивом МДПН вступившие в конфликт пакеты и новые пакеты, ожидающие конца занятого периода, передаются с различными вероятностями. Везде, кроме отдельных замечаний, мы не будем рассматривать эти методы, потому что они не имеют существенных преимуществ по сравнению с ненастойчивым МДПН.

Для анализа АЛОНА с МДПН мы можем снова воспользоваться цепью Маркова, у которой число n задолженных пакетов является состоянием, а переходы между состояниями происходят в моменты окончания пустых окон. Заметим, что за каждым занятым окном (с успешной передачей или конфликтом) должно следовать пустое окно, так как узлам разрешается начать передачу только после обнаружения пустого окна. Для простоты мы предполагаем, что все пакеты данных имеют единичную длину. Обобщение на пакеты с произвольной длиной не представляет, однако, трудности и изучается в задаче 4.21. Отрезок времени между последовательными переходами из одного состояния в другое равен или β (в случае пустого окна) или $1 + \beta$ (в случае занятого окна с последующим пустым). Вместо того чтобы вычислять переходные вероятности, которые не очень способствуют пониманию, мы просто видоизменим определение сноса, данное в (4.4), чтобы оно соответствовало этой новой модели. После перехода в состояние n (в конце пустого окна) вероятность того, что никто не будет передавать в следующем окне (и следовательно, вероятность пустого окна), равна $e^{-\lambda\beta} (1 - q_r)^n$. Первый множитель является вероятностью того, что ни один пакет не поступает в предыдущем пустом окне, а второй — вероятностью того, что не передает ни один задолженный узел. Таким образом, среднее время одного пребывания в состоянии n равно $\beta + [1 - e^{-\lambda\beta} (1 - q_r)^n]$. Аналогично среднее число пакетов, поступивших за время одного пребывания в состоянии n , равно

$$E \{\text{число поступлений}\} = \lambda [\beta + 1 - e^{-\lambda\beta} (1 - q_r)^n]. \quad (4.34)$$

Среднее число ушедших из системы пакетов за время одного пребывания в состоянии n — это просто вероятность успешной передачи; в случае $q_r < 1$ она дается выражением

$$P_{\text{усп}} = [\lambda\beta + q_r n / (1 - q_r)] e^{-\lambda\beta} (1 - q_r)^n. \quad (4.35)$$

Снос в состоянии n определяется как разность между средним числом поступивших пакетов и средним числом ушедших пакетов за время одного пребывания в состоянии n

$$D_n = \lambda [\beta + 1 - e^{-\lambda\beta} (1 - q_r)^n] - [\lambda\beta + q_r n / (1 - q_r)] e^{-\lambda\beta} (1 - q_r)^n. \quad (4.36)$$

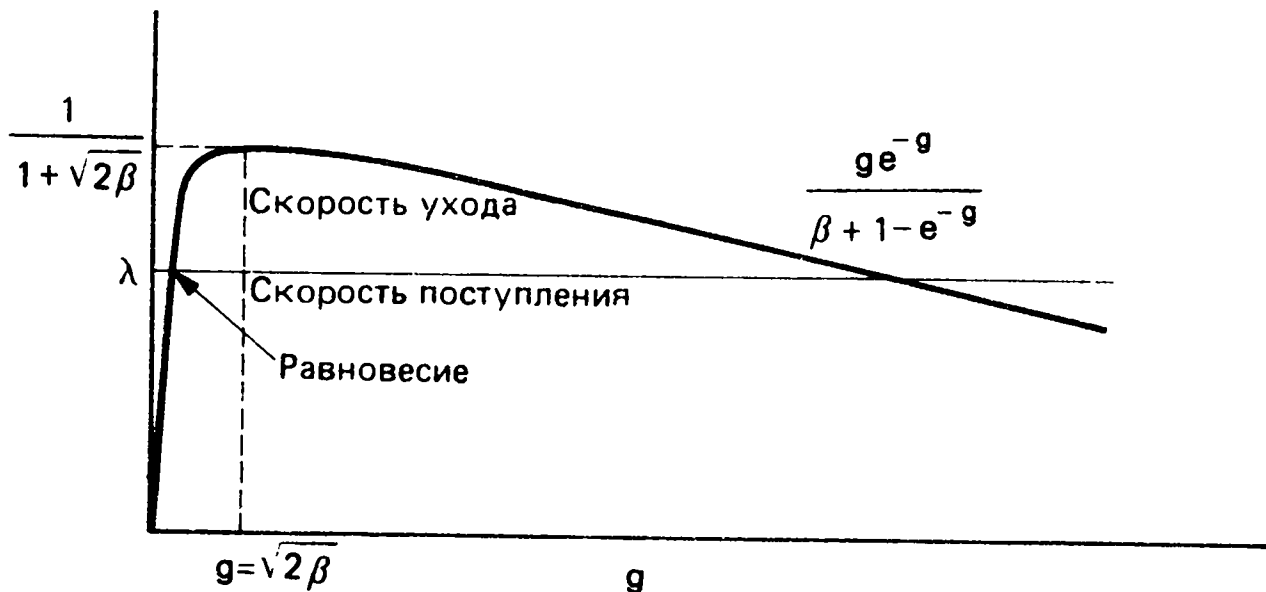


Рис. 4.16. Скорость ухода пакетов в единицу времени для синхронной АЛОНА с МДПН как функция интенсивности попыток передач g , выраженной в пакетах на одно пустое окно. Если β — отношение длины пустого окна к длине окна с передачей данных — мало, то максимальная скорость ухода равна $1/(1 + \sqrt{2\beta})$.

При малой q_r можно воспользоваться приближением $(1 - q_r)^{n-1} \approx (1 - q_r)^n \approx e^{-nq_r}$ и D_n можно представить как

$$D_n \approx \lambda (\beta + 1 - e^{-g(n)}) - g(n) e^{-g(n)}, \quad \text{где} \quad (4.37)$$

$$g(n) = \lambda\beta + q_r n \quad (4.38)$$

есть среднее число попыток передач в состоянии n . Из формулы (4.37) следует, что снос в состоянии n отрицателен, если

$$\lambda < \frac{g(n) e^{-g(n)}}{\beta + 1 - e^{-g(n)}}. \quad (4.39)$$

Числитель правой части (4.39) — это среднее число ушедших из системы пакетов за время пребывания в некотором состоянии, а знаменатель — среднее время пребывания в некотором состоянии; таким образом, отношение можно интерпретировать как скорость ухода пакетов (т. е. среднее число пакетов, уходящих из системы за единицу времени) в состоянии n . На рис. 4.16 показано это отношение как функция $g(n) = \lambda\beta + q_r n$. При малых β эта функция имеет максимум, который равен примерно $1/(1 + \sqrt{2\beta})$ и достигается при $g(n) = \sqrt{2\beta}$. В этом можно убедиться, заменяя $eg(n)$ на приближение $1 + g(n) + g^2(n)/2$, которое справедливо при малом $g(n)$. Чтобы интуитивно понять, почему скорость ухода максимизируется при $g = \sqrt{2\beta}$, заметим, что при малом β одно пустое окно занимает очень мало времени и значительная доля времени приходится на конфликт. При $g = \sqrt{2\beta}$ пустые окна возникают настолько чаще конфликтов, что средние суммарные доли времени, затрачиваемые на них, одинаковы.

Из рис. 4.16 также видно, что системе АЛОНА с МДПН свойственна та же проблема устойчивости, что и обычной синхронной АЛОНА. При фиксированной q_r интенсивность $g(n)$ растет с ро-

стом задолженности n , и, когда n становится слишком большой, скорость ухода будет меньше скорости поступления, что ведет к дальнейшему увеличению задолженности. С практической точки зрения, однако, проблема устойчивости не так важна при МДПН, как при обычной АЛОНА. Заметим, что β/q_r дает среднее число пустых окон, которое узел с задолженностью должен пропустить перед попыткой передать пакет, и при малой величине β и небольшой λ вероятность q_r можно сделать достаточно малой, не вызывая этим заметного увеличения задержки. Это означает, что переход в неустойчивый режим происходит при очень большой задолженности, и поэтому проблемой неустойчивости можно просто пренебречь.

Метод P -настойчивый МДПН, при котором пакеты передаются после пустых окон с вероятностью p , если это новые поступившие пакеты, и с намного меньшей вероятностью q_r , если они попадали в конфликты, представляет собой примитивный путь получения незначительной дополнительной защиты от неустойчивости. В следующем разделе исследуется более фундаментальный способ стабилизации.

4.4.2. Псевдобайесовская стабилизация системы АЛОНА с МДПН

Будем считать, что все пакеты становятся задолженными сразу в момент входа в систему. В конце каждого пустого окна каждый задолженный пакет передается независимо с вероятностью q_r , которая меняется в зависимости от оценки задолженности канала \hat{n} . В состоянии n среднее число пакетов, передаваемых в конце пустого окна, равно $g(n) = nq_r$. Так как мы видели, что скорость ухода пакетов (выраженная в числе пакетов в единицу времени) максимизируется (при малых β и q_r) при $g(n) = \sqrt{2\beta}$, мы выбираем q_r при заданной оценке задолженности \hat{n} следующим образом:

$$q_r(\hat{n}) = \min \left[\frac{\sqrt{2\beta}}{\hat{n}}, \sqrt{2\beta} \right]. \quad (4.40)$$

Операция взятия минимума не позволяет $q_r(\hat{n})$ становиться слишком большой, когда \hat{n} мала; мы не можем рассчитывать на то, что n/\hat{n} близко к 1, когда задолженность мала, и желательно не допускать возникновения большого числа конфликтов в этом случае. Соответствующее правило для вычисления оценки задолженности (снова предполагается, что пакеты имеют единичную длину) имеет вид

$$\begin{aligned} \hat{n}_{k+1} &= \\ &= \begin{cases} \hat{n}_k [1 - q_r(\hat{n}_k)] + \lambda\beta & \text{для пустого окна,} \\ \hat{n}_k [1 - q_r(\hat{n}_k)] + \lambda(1 + \beta) & \text{для окна с успешной передачей,} \\ \hat{n}_k + 2 + \lambda(1 + \beta) & \text{для окна с конфликтом.} \end{cases} \end{aligned} \quad (4.41)$$

Обоснованием этого правила является то, что если априорное распределение n_k пуассоновское со средним \hat{n}_k , то при условии пустого окна апостериорное распределение n_k — пуассоновское со средним $\hat{n}_k [1 - q_r(\hat{n}_k)]$ (см. задачу 4.20). Учитывая, что поток новых пакетов, поступающих в пустом окне длительности β , является пуассоновским, получаем, что распределение n_{k+1} — пуассоновское со средним \hat{n}_{k+1} , как показано выше. Аналогично при условии успешной передачи апостериорное распределение $n_k - 1$ (вычитается успешно переданный пакет) — пуассоновское со средним $\hat{n}_k [1 - q_r(\hat{n}_k)]$. Учитывая, что поток новых пакетов, поступающих в окне с успешной передачей и последующем пустом окне, является пуассоновским, получаем, что n_{k+1} имеет пуассоновское распределение со средним \hat{n}_{k+1} , как показано выше. Наконец, если возникает конфликт, то апостериорное распределение n_k не совсем пуассоновское, но оно хорошо аппроксимируется пуассоновским со средним $\hat{n}_k + 2$. Добавляя $\lambda(1 + \beta)$ для учета новых поступающих пакетов, получаем последнее выражение в (4.41).

Заметим, что, когда n_k и \hat{n}_k малы, q_r велика и новые поступающие пакеты вряд ли вообще задерживаются. Когда $\hat{n}_k \approx n_k$ и величина \hat{n}_k велика, скорость ухода примерно равна $1/(1 + \sqrt{2\beta})$, так что при $\lambda < 1/(1 + \sqrt{2\beta})$ скорость ухода превышает скорость поступления и задолженность в среднем уменьшается. Наконец, если величина $|n_k - \hat{n}_k|$ велика, среднее изменение задолженности может быть положительным, но среднее изменение $|n_k - \hat{n}_k|$ отрицательным; рис. 4.5 опять показывает качественную картину средних изменений величин n_k и $n_k - \hat{n}_k$.

Теперь сделаем приближенный анализ задержки этой стратегии (и других подобных стабилизирующих стратегий), используя тот же метод анализа, что и в подразд. 4.2.3. Пусть W_i обозначает время, прошедшее от момента поступления i -го пакета до начала i -й успешной передачи. Усреднение W_i по всем i дает среднее время ожидания W . Пусть n обозначает число задолженных пакетов непосредственно перед i -м поступлением пакета без учета пакета, находящегося в данный момент в процессе успешной передачи. Тогда

$$W_i = R_i + \sum_{j=1}^{n_i} t_j + y_i, \quad (4.42)$$

где R_i — остаточное время до следующего изменения состояния, t_j ($1 \leq j \leq n_i$) — последовательность интервалов, такая что моменты окончания каждой из n_i успешных передач являются концом одного интервала и началом следующего, а y_i — конечной интервал времени до начала i -й успешной передачи.

Задолженность равна по крайней мере 1 на всех интервалах между переходами в другое состояние, принадлежащих периоду времени, входящему в правую часть равенства (4.42), и можно воспользоваться упрощающим приближением, состоящим в том, что число попыток передач в каждом из этих интервалов является пуассоновской величиной с параметром g . В дальнейшем мы положим $g = \sqrt{2\beta}$, но в данный момент параметр g может быть произвольным. Это приближение несколько отличается от того, которое использовано в подразд. 4.2.3, где мы предполагали, что, когда задолженность равна 1, всегда возникает успешная передача; различие обусловлено равенством (4.40), в соответствии с которым q_r остается малой. Среднее значение каждого t_j определяется выражением

$$E\{t\} = e^{-g}(\beta + E\{t\}) + ge^{-g}(1 + \beta) + \\ + [1 - (1 + g)e^{-g}](1 + \beta + E\{t\}). \quad (4.43)$$

Первый член соответствует отсутствию передачи в первом интервале между переходами в другое состояние; это событие появляется с вероятностью e^{-g} , занимает отрезок времени β и требует в дальнейшем время $E\{t\}$ для осуществления успешной передачи. Следующие два члена соответствуют аналогичным образом успешной передаче и конфликту соответственно. Разрешая относительно $E\{t\}$, получаем

$$E\{t\} = \frac{\beta + 1 - e^{-g}}{ge^{-g}}. \quad (4.44)$$

Заметим, что это равно, как и следовало ожидать, обратной величине среднего числа пакетов, уходящих из системы за единицу времени (см. (4.39)). Таким образом, $E\{t\}$ минимизируется при $g = \sqrt{2\beta}$. Усредняя по i и используя в (4.42) теорему Литтла, получаем

$$W(1 - \lambda E\{t\}) = E\{R\} + E\{y\}. \quad (4.45)$$

Среднее остаточное время можно приближенно оценить, замечая, что система проводит долю времени $\lambda(1 + \beta)$ в интервалах с успешной передачей. Среднее остаточное время при поступлении пакета в этих интервалах равно $(1 + \beta)/2$. Доля времени, затрачиваемая на интервалы с конфликтами, пренебрежимо мала (при малом β) по сравнению с той, что приходится на успешные передачи, а остаточное время при поступлении в пустых интервалах незначительно. Таким образом,

$$E\{R\} \approx \lambda(1 + \beta)^2/2. \quad (4.46)$$

Наконец, $E\{y\}$ — это просто $E\{t\}$, уменьшенное на длину интервала с успешной передачей, так что $E\{y\} = E\{t\} - (1 + \beta)$. Подстановка этих выражений в (4.45) дает

$$W \approx \frac{\lambda(1 + \beta)^2 + 2[E\{t\} - (1 + \beta)]}{2[1 - \lambda E\{t\}]} \quad (4.47)$$

Минимум этого выражения по всем g достигается при минимальном $E\{t\}$ и мы уже знаем, что этот последний минимум (при малом β) равен $1 + \sqrt{2\beta}$ и достигается при $g = \sqrt{2\beta}$. После подстановки этого значения получаем

$$W \approx \frac{\lambda + 2\sqrt{2\beta}}{2[1 - \lambda(1 + \sqrt{2\beta})]} \quad (4.48)$$

Обратим внимание на сходство этого выражения с (3.45) из гл. 3, которое определяет время ожидания в системе $M/D/1$. Стабилизация системы АЛОНА с МДПН позволяет изменять q_r в зависимости от задолженности так, чтобы поддерживать скорость ухода пакетов из системы, близкой к $1/(1 + \sqrt{2\beta})$, всегда, когда имеется задолженность.

4.4.3. Несинхронная АЛОНА с МДПН

При рассмотрении синхронной АЛОНА с МДПН мы предполагали, что все узлы синхронизированы, так что передачи начинаются только во время пустых периодов в моменты, кратные β . Здесь мы избавляемся от этого ограничения и предполагаем, что, когда пакет поступает, его передача начинается немедленно, если прослушивание канала показывает, что он свободен. Если при прослушивании канала обнаруживается, что он занят или если передача приводит к конфликту, то пакет считается задолженным. Повторные попытки передачи каждого задолженного пакета предпринимаются в случайно выбранные моменты времени, разделенные независимыми, экспоненциально распределенными случайными задержками τ , имеющими плотность распределения вероятностей $\lambda e^{-\lambda\tau}$. Если канал свободен в один из таких моментов, то пакет передается, и эта процедура повторяется до тех пор, пока передача не будет успешной. Мы продолжаем считать, что время распространения и обнаружения равно β , так что если одна передача начинается в момент t , то другой узел обнаружит занятость канала только в момент $t + \beta$, что приводит к возможности возникновения конфликтов.

Рассмотрим пустой период, в начале которого задолженность была равна n . Время до начала первой передачи (предполагается, что $m = \infty$) является экспоненциально распределенной случайной величиной со средним значением

$$G(n) = \lambda + nx. \quad (4.49)$$

Заметим, что $G(n)$ — это интенсивность попыток, выраженная в пакетах в единицу времени, тогда как $g(n)$ в последнем разделе выражалась в пакетах на пустое окно. После начала этой первой передачи задолженность равна или n (если начал передаваться новый поступивший пакет), или $n - 1$ (если начал передаваться задолженный пакет). Таким образом, время, прошедшее от первого начала передачи до того момента, когда другой узел с новым поступившим или задолженным пакетом начинает прослушивать канал, является экспоненциально распределенной случайной величиной со средним $G(n)$ или $G(n - 1)$. Конфликт возникает, если следующее прослушивание начинается в пределах интервала времени β . Следовательно, вероятность того, что этот период занятости будет конфликтным, равна $1 - e^{-\beta G(n)}$ или $1 - e^{-\beta G(n-1)}$. Разница между этими величинами мала, если мало значение βx , и мы не обращаем на нее внимания в дальнейшем. Таким образом, мы оцениваем вероятность успешной передачи, следующей за пустым периодом, выражением $e^{-\beta G(n)}$.

Среднее время от начала одного пустого периода до начала следующего равно $1/G(n) + (1 + \beta)$; первый член — это среднее время до начала первой передачи, а второй член $(1 + \beta)$ — это время, в течение которого произойдет первая передача и в узлах будет обнаружено, что канал опять свободен. Если возникает конфликт, то появляется небольшое добавочное время, меньшее чем β , до тех пор, пока пакеты, вступившие в конфликт, не будут прослушиваться; мы не учитываем этот вклад, поскольку он незначителен даже по сравнению с β , которое незначительно. Скорость ухода во время периодов, когда задолженность равна n , определяется тогда выражением

$$\text{Скорость ухода } (n) = \frac{e^{-\beta G(n)}}{1/G(n) + (1 + \beta)}. \quad (4.50)$$

При малом β максимальное значение этой скорости ухода примерно равно величине $1/(1 + 2\sqrt{\beta})$, которая достигается при $G(n) \approx \beta^{-1/2} - 1/2$. Эта максимальная скорость ухода немного меньше, чем в синхронном случае (см. равенство (4.39)); причина связана с теми периодами, когда МДПН не используется — при заданной интенсивности попыток передачи конфликты возникают в несинхронной системе с несколько большей вероятностью, чем в синхронной системе. При МДПН с малым β эта убыль скорости ухода, однако, очень мала. Более значительно то, что в синхронной системе β должно быть значительно больше, чем в несинхронной системе, чтобы компенсировать неточности синхронизации и учесть время распространения в наихудшем случае. Таким образом, несинхронная АЛОНА является, по-видимому, естественным вариантом для МДПН.

Несинхронной АЛОНА с МДПН свойственны те же проблемы устойчивости, что и всем системам АЛОНА, но ее можно стабилизировать таким же образом, как синхронную АЛОНА с МДПН. Детали даются в задаче 4.22.

4.4.4. Алгоритм разбиения с ППП и МДПН

Теперь исследуем, можно ли достичь более высоких скоростей передач или меньших задержек, используя алгоритмы разбиения с МДПН. Мы увидим, что можно добиться относительно малого улучшения, но интересно узнать почему. Мы возвращаемся к предположению о пустых окнах длительности β и предположим, что по обратной связи поступают сигналы 0, 1, e . В конце каждой успешной передачи или конфликта появляется пустое окно, необходимое для того, чтобы сигнал обратной связи поступил в узлы. Таким образом, мы считаем, что успешные передачи и конфликты имеют длительность $1 + \beta$, инструкции алгоритма отрабатываются в конце каждого такого удлиненного окна с успешной передачей или конфликтом, а также в конце каждого обычного пустого окна.

Тот же алгоритм, что описывается соотношениями (4.15)–(4.18), можно использовать, хотя длину исходного интервала для ПРК следует изменить. Более того, как мы скоро увидим, интервалы с конфликтами не следует разбивать на равные подынтервалы. Так как конфликты занимают намного больше времени, чем пустые окна, базовый назначенный интервал α_0 , должен быть мал. Это означает в свою очередь, что конфликтов с более чем двумя пакетами немного и, таким образом, анализ проще, чем раньше.

Сначала вычислим среднюю длину ПРК и среднее число успешных передач в ПРК. Пусть $g = \lambda \alpha_0$ обозначает среднее число пакетов, поступивших на начальный назначенный интервал длины α_0 . С вероятностью e^{-g} исходный интервал не содержит пакеты, что приводит к времени разрешения конфликта, равному β , и отсутствию успешной передачи. С вероятностью ge^{-g} в начале ПРК возникает успешная передача, что приводит к времени разрешения конфликта, равному $1 + \beta$. Наконец, с вероятностью $(g^2/2)e^{-g}$ возникает конфликт, что приводит к времени разрешения конфликта, равному $1 + \beta + T$ при некотором T , которое будет вычислено позже; поскольку конфликтов с более чем двумя пакетами немного, мы предполагаем, что во время каждого ПРК с конфликтом происходят две успешные передачи. Таким образом, получаем

$$E \{\text{длина ПРК}\} \approx \beta e^{-g} + (1 + \beta) ge^{-g} + (1 + \beta + T) (g^2/2) e^{-g}, \quad (4.51)$$

$$E \{\text{число пакетов в ПРК}\} \approx ge^{-g} + 2 (g^2/2) e^{-g}. \quad (4.52)$$

Как и ранее, максимальная скорость устойчивой передачи при заданном g равна

$$\lambda_{\max} = \frac{E \{\text{число пакетов в ПРК}\}}{E \{\text{длина ПРК}\}} \approx \frac{g + g^2}{\beta + g(1 + \beta) + (g^2/2)(1 + \beta + T)}. \quad (4.53)$$

Теперь можно максимизировать правую часть (4.53) по g (т. е. по α_0). В предельном случае $\beta \rightarrow 0$ получаем асимптотические выражения

$$g \approx \sqrt{2\beta/(T-1)}, \quad (4.54)$$

$$\lambda_{\max} \approx \frac{1}{1 + \sqrt{2\beta(T-1)}}. \quad (4.55)$$

В заключение нужно вычислить T , время разрешения конфликта, после того как он возник. Пусть x обозначает долю интервала, которая отводится первому подынтервалу при разбиении интервала; оптимальное x выберем позже. Первое окно, после того как конфликт обнаружен, является пустым, содержащим успешную передачу или конфликтным с вероятностями $(1-x)^2$, $2x(1-x)$ или x^2 соответственно. Средняя длина в каждом из этих случаев равна $\beta + T$, $2(1 + \beta)$ и $1 + \beta + T$. Таким образом,

$$T \approx (1-x)^2(\beta + T) + 4x(1-x)(1 + \beta) + x^2(1 + \beta + T), \quad (4.56)$$

откуда T можно выразить через x .

Приравнивая производную dT/dx к нулю, мы находим прямым вычислением, что T минимально при

$$x = \sqrt{\beta + \beta^2} - \beta. \quad (4.57)$$

Результирующее значение T при малом β равно $T \approx 2 + \sqrt{\beta}$. Подставив это выражение в (4.55), получаем

$$\lambda_{\max} \approx \frac{1}{(1 + \sqrt{2\beta})}. \quad (4.58)$$

При малом β алгоритм разбиения с ППП имеет такую же максимальную скорость передачи, как и синхронная АЛОНА. Это не удивительно, потому что в модели без МДПН основным преимуществом алгоритма с ППП является его эффективность при разрешении конфликтов, а в модели с МДПН конфликты возникают редко. Когда конфликты действительно возникают, они разрешаются при обеих стратегиях посредством повторной передачи с малой вероятностью. Довольно удивительно на первый взгляд то, что если мы используем алгоритм с ППП с равными подынтервалами (т. е. $x = 1/2$), то обнаруживаем, что скорость передачи ограничена величиной $1/(1 + \sqrt{3\beta})$. Это падение скорости обусловлено существенным увеличением числа конфликтов.

4.5. Резервирование при множественном доступе

Теперь рассмотрим очень простой способ увеличения скорости передачи каналов с множественным доступом, который, вероятно, стал уже очевидным. Если пакеты с передаваемыми данными имеют большую длину, то зачем использовать эти длинные окна времени на то, чтобы не посылать ничего или посылать пакеты, вступающие в конфликт? Было бы намного эффективнее посылать очень короткие пакеты или в режиме состязаний или в режиме ВУ и использовать эти короткие пакеты для резервирования более длинных окон, предназначенных для бесконфликтной передачи фактических данных. Таким образом, все пустые окна или конфликтные окна становятся короткими, что приводит к повышению общей эффективности. Существует много различных систем, которые работают таким образом, и наша основная цель состоит не в исследовании второстепенных различий между ними, а в выяснении того, что все они, по существу, одинаковы.

Сначала исследуем в некоторой степени «каноническую» систему резервирования [238]. Предположим, что для передачи каждого пакета данных требуется одна единица времени и для передачи каждого резервирующего пакета нужно $v \ll 1$ единиц времени. Формат резервирующего пакета неважен; он должен содержать информацию, достаточную для выполнения резервирования. Например, в случае быстрой обратной связи, указывающей на отсутствие передачи, успех или конфликт, резервирующий пакет не должен содержать какую-либо дополнительную информацию, кроме той, что этот пакет существует. После успешной передачи резервирующего пакета последующий отрезок времени единичной длины или некоторой заранее определенной длины может автоматически отводиться для передачи соответствующего пакета данных. Для передачи резервирующих пакетов может использоваться любая стратегия, в том числе временное уплотнение, синхронная АЛОНА или алгоритм разбиения.

Можно легко определить максимальную скорость передачи S , выраженную в пакетах данных за единицу времени, которая достигается в такой системе. Пусть S_r обозначает максимальную скорость передачи, выраженную в успешно переданных резервирующих пакетах за окно резервирования алгоритма, который используется резервирующими пакетами (т. е. $1/e$ в случае синхронной АЛОНА, $0,478$ в случае алгоритма разбиения или 1 в случае ВУ). Тогда время, затрачиваемое на резервирование, усредненное по большому числу актов резервирования, стремится к v/S_r и, кроме того, одна единица времени необходима для передачи каждого пакета данных. Таким образом, общее время, затрачиваемое на пакет данных, стремится к $1 + v/S_r$, и мы видим, что

$$S = \frac{1}{(1 + v/S_r)} . \quad (4.59)$$

В этом выражении резервирующий пакет служит только для выполнения резервирования и не содержит никаких данных. Как мы увидим, во многих системах резервирующий пакет содержит некоторые данные; следовательно, если на передачу данных требуется единица времени, то достаточно передавать резервирующий пакет длины v , за которым следует передача оставшихся данных в течение времени $1 - v$. В этом случае максимальная скорость передачи определяется выражением

$$S = \frac{1}{[1 + v(1/S_r - 1)]}. \quad (4.60)$$

Например, в случае синхронной ALOHA максимальная скорость передачи равна $S = 1/[1 + v(e - 1)]$. Очевидно, что если v мало, например порядка 0,01, то максимальная скорость передачи близка к 1 и не зависит сильно от алгоритма разрешения конфликтов или от того, содержит ли резервирующий пакет часть данных. В дальнейшем мы увидим, что локальные сети типа ETHERNET [172] можно описывать моделью, которая почти полностью совпадает с приведенной, и значение $v = 0,01$ является типичным. Таким образом, пропускная способность подобных сетей может достигать очень высоких значений, причем при этом не требуется использования очень сложных алгоритмов разрешения конфликтов.

4.5.1. Спутниковые системы резервирования

Одна из простейших систем резервирования, предназначенная в наибольшей степени для сетей со спутниковой связью [130], имеет кадровую структуру, показанную на рис. 4.17. Последовательности окон, отведенных для передачи данных, предшествует период резервирования, состоящий из m окон резервирования, по одному окну резервирования на каждый узел. Пусть v обозначает длину окна резервирования; здесь, как обычно, за единицу измерения времени принята средняя длина пакета данных. Следовательно, v равно отношению числа битов, используемых для выполнения резервирования (включая защитное пространство и служебную информацию), к среднему числу битов в пакете данных. Период резервирования в каждом кадре имеет полную длину $A = mv$. Минимальная длина кадра устанавливается так, чтобы превышать время распространения до спутника и обратно 2β , так что в окнах резервирования текущего кадра производится распределение окон передачи данных следующего кадра (см. рис. 4.17).

Кадр растягивается и имеет длину, которая больше минимальной, в случае когда необходимо выполнить все резервирования. Заметим, что использование ВУ для резервирования имеет здесь гораздо больше смысла, чем в случае, когда обычное ВУ приме-

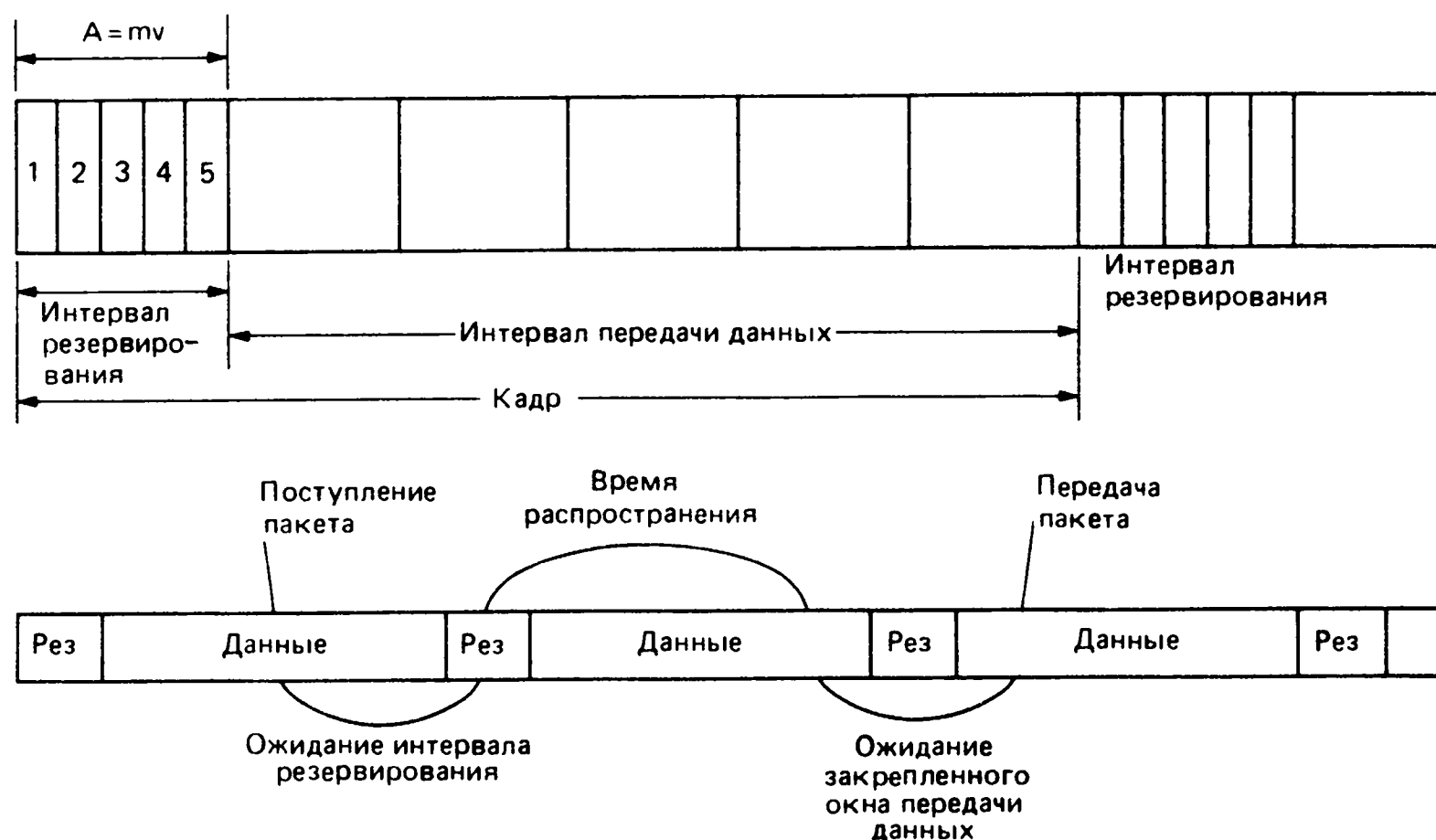


Рис. 4.17. Спутниковая система резервирования с использованием ВУ для выполнения резервирования. Пакеты, поступившие во время одного кадра, передаются в следующем кадре.

няется для передачи данных. Одна из причин состоит в том, что окна резервирования короткие и поэтому тратится мало времени на источник, которому нечего передавать. Другая причина при использовании спутника заключается в том, что если допустить возникновение конфликтов в окнах резервирования, то из-за задержки распространения по каналу они будут разрешаться довольно медленно. Из равенства (4.59) следует, что максимальная скорость передачи подобной системы равна $1/(1 + v)$, когда предполагается, что каждый резервирующий пакет может только выполнить резервирование для одного пакета данных. Вместе с тем если допустить, что резервирующий пакет может выполнять кратные резервирования для своего источника, то нетрудно увидеть, что максимальная скорость передачи может приближаться к 1 сколь угодно близко, поскольку при сильной нагрузке кадры становятся очень длинными и редкие периоды резервирования занимают очень мало времени.

В предположении, что каждый резервирующий пакет может выполнять резервирование для большого числа пакетов данных, эта система становится очень похожей на систему резервирования с одним пользователем, исследованную в подразд. 3.5.2 гл. 3. Основное отличие состоит в том, что из-за задержки распространения резервирования, выполняемые на одном интервале резервирования, относятся к окнам передачи данных, идущих за следующим интервалом резервирования. Таким образом, если предполагать, что поступающий поток пакетов пуассоновский и не

учитывать ограничение на минимальную длину кадра, то среднее время ожидания в очереди i -го пакета равно

$$E\{W_i\} = E\{R_i\} + E\{N_i\}/\mu + 2A. \quad (4.61)$$

Это выражение совпадает с равенством (3.60) гл. 3, за исключением того, что последний член равен $2A$. Здесь $A = tv$ является длиной интервала резервирования; она умножается на 2, так как пакет перед передачей должен ожидать прохождения двух интервалов резервирования. Используя тот же метод анализа, который описан в гл. 3, получаем, что среднее время ожидания в очереди равно

$$W = \frac{\lambda \bar{X}^2}{2(1-\lambda)} + \frac{A}{2} + \frac{2A}{1-\lambda}. \quad (4.62)$$

В этом методе анализа допускается, что пакеты данных могут иметь произвольное распределение длины со среднеквадратичным значением \bar{X}^2 (это требует, конечно, чтобы резервирующие пакеты содержали информацию о длине), но мы нормировали время так, что $\bar{X} = 1/\mu = 1$; следовательно, $\rho = \lambda$. Заметим, что в пределе, когда v стремится к 0, равенство (4.62) выражает, как и предполагалось, время ожидания в системе $M/G/1$. Кроме того, W остается, как мы и предсказывали, конечным при $\lambda < 1$.

К сожалению, этот анализ не учитывает то условие, что длина каждого кадра не может быть меньше времени передачи по каналу туда и обратно 2β . Поскольку обычно 2β во много раз больше длины периода резервирования A , мы видим, что W , определяемое (4.26), больше чем 2β , только когда λ очень близка к 1. Так как каждый пакет должен испытывать задержку по крайней мере на время 2β для того, чтобы резервирование выполнилось, мы заключаем, что равенство (4.62) не дает хорошей оценки задержки, за исключением, возможно, случая, когда λ очень близка к 1.

Вместо того чтобы сделать этот анализ более реалистичным, мы обратим внимание на то, что модель с переменной длиной кадра имеет некоторые нежелательные свойства. Во-первых, если некоторые узлы воспримут с ошибкой информацию резервирования, в этих узлах нельзя будет определить начало следующего периода резервирования; разработка распределенного алгоритма, который сохранял бы синхронизацию узлов в периодах резервирования при наличии ошибок, является непростой задачей. Во-вторых, система не вполне справедлива в том смысле, что очень активные узлы могут резервировать передачу многих пакетов в кадре, что сделает кадры длинными и будет значительно препятствовать передачам менее активных пользователей.

По этим причинам желательно поддерживать постоянную длину кадра. Узлы могут по-прежнему делать кратные резерви-

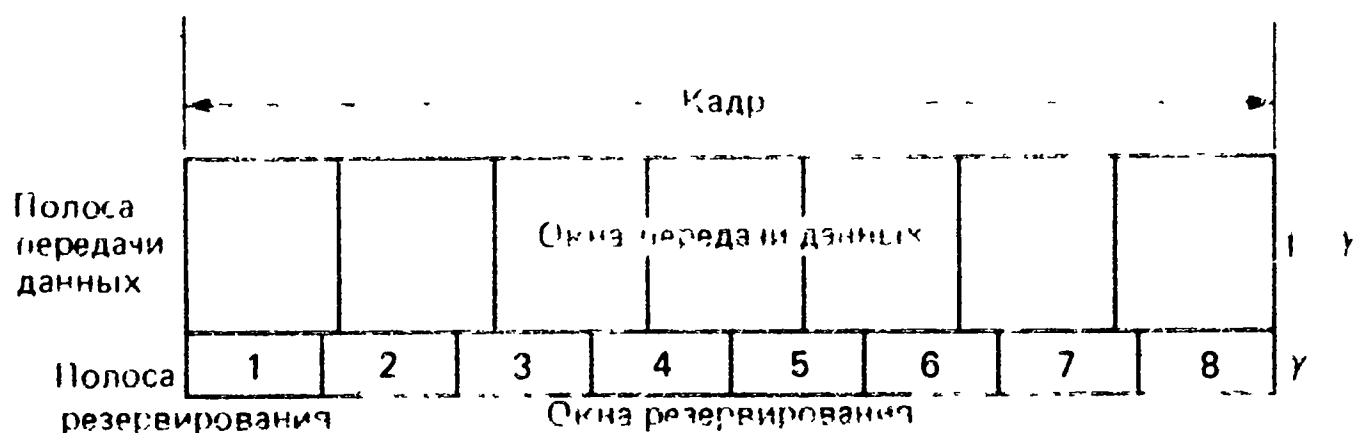


Рис 4.18. Система резервирования с отдельной полосой частот для выполнения резервирований. Резервирование производится в соответствии с ВУ в полосе резервирования.

рования в одном окне резервирования; это желательно в том случае, когда только несколько узлов активны. При фиксированной длине кадра, однако, иногда необходимо переносить передачу пакетов с резервированием в одном кадре в следующий кадр.

Заметим, что все узлы узнают обо всех резервированиях спустя время 2β , в течение которого происходит распространение резервирующих пакетов. Таким образом, концептуально мы имеем общую очередь резервирующих пакетов и узлы могут совместно передавать в соответствии с любой желаемой дисциплиной обслуживания, такой как первый поступил — первый передается, круговой опрос или некоторая приоритетная схема. Если передача пакета происходит в каждом отведенном для данных окне, для которого очередь резервирующих пакетов не является пустой, и если дисциплина не зависит от длины пакета, то средняя задержка не зависит от дисциплины обслуживания.

Вычисление средней задержки в этой системе представляет довольно сложную задачу, но если ввести небольшую модификацию, показанную на рис. 4.18, то легко выполнить приближенный анализ, дающий хорошую точность. Предположим, что для выполнения резервирования отводится доля γ доступной полосы пропускания и передача в этой доле полосы пропускания производится на основе ВУ, причем таким образом, что каждый узел передает один резервирующий пакет в каждом из периодов, длина каждого из которых равна времени распространения туда и обратно 2β . Так как v — отношение длины резервирующего пакета к длине пакета данных, то $\gamma = mv/2\beta$. Поступивший пакет ждет в среднем $2\beta/2$ единиц времени начала передачи своего резервирующего пакета, затем ждет $2\beta/m$ единиц, пока передача резервирующего пакета закончится, и, наконец, 2β единиц, пока резервирующий пакет не будет принят. Таким образом, в среднем через $2\beta (3/2 + 1/m)$ единиц времени пакет присоединится к общей очереди.

Процесс поступления резервирующих пакетов в общую очередь приближенно пуассоновский (т. е. число пакетов, поступивших во время окна резервирования, не зависит от числа пакетов в других окнах и имеет пуассоновское распределение). Поскольку пакет находится в общей очереди, его время обслуживания равно $X/(1 - \gamma)$, где X — время передачи пакета в случае, когда используется вся полоса пропускания. Таким образом, общая очередь обслуживается, как в системе $M/G/1$ с $\rho = \lambda(1 - \gamma)$. Общая задержка, состоящая из суммы среднего времени присоединения к общей очереди (т. е. $2\beta(3/2 + 1/m)$) и времени ожидания в общей очереди системы $M/G/1$, равна

$$W = 3\beta + \frac{2\beta}{m} + \frac{\lambda \overline{X^2}}{2(1 - \gamma - \lambda)(1 - \gamma)}. \quad (4.63)$$

Мы видим, что эта стратегия позволяет достичь, по существу, безупречного распределения за счет задержки при выполнении резервирования. При малых λ эта задержка кажется избыточной, поскольку в каждом кадре остается нераспределенным, как правило, ряд окон передачи данных. Это наводит на мысль об использовании нераспределенных окон кадра в режиме состязаний [254]. Для пакетов, передаваемых в нераспределенных окнах, выполняется резервирование в следующем интервале резервирования. Считается, что если пакет передается успешно во время состязаний, то его резервирование аннулируется. Этот метод приводит к очень малой задержке при слабой нагрузке (так как не тратится время распространения туда и обратно на резервирование) и к весьма высокой эффективности (определяемой равенством (4.63)) при сильной нагрузке.

Во всех упомянутых стратегиях используется ВУ для выполнения резервирований, что затрудняет введение новых источников или удаление старых источников в системе. К тому же с ростом m увеличивается γ и растет доля полосы пропускания, используемая для резервирования. Это приводит к предложению использовать окна резервирования в режиме разрешения конфликтов [130]. Анализировать такую систему трудно, но ясно, что при большом m и малой λ задержку можно уменьшить, используя намного меньше чем m мини-окон в резервирующей части кадра.

Другой более простой в некоторой степени метод резервирования состоит в том, что первому пакету сообщения разрешается производить резервирование для последующих пакетов. При подходе, рассмотренном при получении (4.60), мы считали первый пакет резервирующим пакетом, а все сообщение — пакетом данных. Таким образом, равенство (4.60) определяет максимальную скорость передачи в такой схеме, если положить ν равной обратной величине среднего числа пакетов в сообщении. Подходящая длина

Окно 1	Окно 2	Окно 3	Окно 4	Окно 5	Окно 6	
15	Пустое окно	3	20	Конфликт	2	Кадр 1
15	7	3	Пустое окно	9	2	Кадр 2
Пустое окно	7	3	Конфликт	9	Пустое окно	Кадр 3
18	7	3	Конфликт	9	6	Кадр 4
18	7	3	15	9	6	Кадр 5

Рис. 4.19. Метод резервирования с короткими пакетами и многопакетными сообщениями. После того как узел захватит окно кадра, он удерживает это окно до окончания передачи сообщения.

пакета в такой схеме получается при установлении компромисса между неэффективностью резервирования при длинных пакетах и неэффективностью УЛПД при коротких пакетах.

При использовании в спутниковых сетях эти схемы обычно дополняются кадровой структурой (рис. 4.19), при которой кадр состоит из фиксированного числа окон передачи пакетов [51]. В кадр вводится столько окон передачи пакетов, сколько необходимо, чтобы длина кадра превысила время передачи туда и обратно.

Когда источник успешно передает пакет в одном из этих окон, он может автоматически резервировать соответствующее окно очередного кадра и каждого последующего кадра до тех пор, пока не будет передано его сообщение. Это можно выполнять или с помощью поля в заголовке пакета, где указывается на то, что должен следовать еще один пакет, или, более просто и менее эффективно, посредством автоматического резервирования этого окна в каждом последующем кадре до тех пор, пока окно не окажется пустым в первый раз. После окончания периода резервирования данное окно кадра становится доступным для состязаний. Заметим, что алгоритм состязаний должен применяться с некоторой осторожностью. Недостаточно просто разрешить всем ожидающим источникам использовать первое пустое окно после периода резервирования, так как это привело бы к недопустимо высокой вероятности конфликта.

Другая разновидность этой схемы состоит в том, что каждый источник имеет собственное определенное окно в кадре [33]. Когда источник не использует свое собственное окно, другие

источники могут захватить его в режиме состязаний, но когда источник хочет вернуть свое собственное окно, он просто передает пакет в этом окне, и если возникает конфликт, то другим источникам запрещается использовать это окно в следующем кадре, что позволяет владельцу захватить его. Эта разновидность схемы в некоторой степени более справедлива, чем предыдущая, но задержки у нее больше как из-за большого числа окон в кадре в случае, когда число пользователей велико, так и из-за менее эффективного разрешения конфликтов. Сохраняется также проблема удаления и введения новых источников в систему.

4.5.2. Локальные сети; МДПН/ОК и ETHERNET

В последнем подразделе рассматривались спутниковые сети, в которых время передачи туда и обратно играет важную роль в любой схеме резервирования. В локальных сетях, по крайней мере при нынешнем уровне развития техники, время передачи туда и обратно составляет очень малую долю длительности передачи пакета. Поэтому, вместо того чтобы выполнять резервирование некоторого числа окон в далеком будущем, резервирования можно делать на ближайшее будущее. Концептуально в этом нет большой разницы. Это просто означает, что в спутниковых сетях задержки передачи должны быть больше, так как сигналы обратной связи при любой стратегии разрешения конфликтов всегда задерживаются на время распространения туда и обратно. Однако с технической точки зрения это различие более важно, поскольку физические свойства среды, используемой в локальных сетях, могут упростить реализацию методов резервирования.

В сети ETHERNET [172] используется это упрощение и широко применяемый в локальных сетях метод доступа. Ряд узлов подсоединяется к общему кабелю, так что когда один узел передает пакет (а другие молчат), то все остальные узлы прослушивают этот пакет. Кроме того, так же как при прослушивании несущей, узел может прослушивать кабель перед тем, как начать передачу (т. е. может различать состояния шины e , 1 и отсутствие передачи). Наконец, благодаря физическим свойствам кабеля узел способен прослушивать кабель во время передачи. Таким образом, если два узла начинают передавать почти одновременно, то они вскоре обнаруживают наличие конфликта и оба прекращают передачу. Этот метод называется МДПН с обнаружением конфликтов (МДПН/ОК). Вместе с тем если один узел начинает передавать, а другие узлы не передают во время распространения сигнала первого узла по кабелю, то первый узел гарантированно заканчивает передачу своего пакета, не вступая в конфликты. Таким образом, можно считать, что первая часть пакета выполняет резервирование для оставшейся части.

Синхронный МДПН/ОК

Наиболее удобно описывать ETHERNET с помощью окон и мини-окон. Мини-окна имеют длину β , которая обозначает время, необходимое для того, чтобы сигнал распространился от одного конца кабеля до другого и был прослушан. Если все узлы действуют синхронно на мини-окнах этой длины и только один узел передает в мини-окне, то все остальные узлы обнаруживают передачу и не будут использовать последующие мини-окна до тех пор, пока не завершится передача всего пакета. Если более чем один узел передает в мини-окне, то каждый передающий узел обнаружит это к концу мини-окна и прекратит передачу. Таким образом, мини-окна используются в режиме состязаний и, когда в мини-окне имеет место успешная передача, эффективно резервируется канал для завершения передачи пакета.

МДПН/ОК можно анализировать с помощью цепи Маркова таким же способом, что МДПН в системе АЛОНА. Мы предполагаем, что каждый узел с задолженностью передает после каждого пустого окна с вероятностью q_r и что число узлов, передающих после пустого окна, имеет пуассоновское распределение с параметром $g(n) = \lambda + q_r n$ ¹⁾. Рассмотрим переходы из одного состояния в другое в моменты окончания пустых окон; если никто не передает, то следующее пустое окно заканчивается через время β . Если передает один узел, то следующее пустое окно заканчивается спустя время $1 + \beta$. Можно рассмотреть пакеты с произвольной длиной, но, чтобы точно соответствовать модели с пустыми окнами, нужно считать, что длины пакетов кратны длине пустого окна; как и ранее, мы предполагаем, что средняя длина пакета равна 1. Наконец, если возникает конфликт, то следующее пустое окно заканчивается через время 2β ; другими словами, узлы должны после конфликта прослушать пустое окно, чтобы узнать о том, что передавать не опасно.

Средняя длина интервала между переходами из одного состояния в другое равна в таком случае β плюс единица, умноженные на вероятность успешной передачи, плюс β , умноженная на вероятность конфликта:

$$E \{\text{длина интервала}\} = \beta + g(n) e^{-g(n)} + \beta [1 - (1 + g(n)) e^{-g(n)}]. \quad (4.64)$$

Среднее число пакетов, поступивших между переходами из одного состояния в другое, равно λ , умноженной на длину этого интервала, так что снос в состоянии n равен $\lambda E \{\text{длина интервала}\} - P_{\text{усп}}$. Вероятность успешной передачи равна $g(n) e^{-g(n)}$, так

¹⁾ Последнее предположение противоречит ранее сделанному предположению о пуассоновском потоке поступления новых пакетов. — *Прим. ред.*

что аналогично неравенству (4.39) снос в состоянии n отрицателен, если

$$\lambda < \frac{g(n) e^{-g(n)}}{\beta + g(n) e^{-g(n)} + \beta [1 - (1 + g(n)) e^{-g(n)}]}. \quad (4.65)$$

Правая часть (4.65) интерпретируется как скорость ухода в состояние n . Эта величина достигает максимума по $g(n)$ в точке $g(n) = 0,77$ и максимальное значение правой части равно $1/(1 + 3,31\beta)$. Таким образом, если МДПН/ОК стабилизирован (это можно сделать, например, с помощью псевдобайесовского метода), то максимальная величина λ , при которой система устойчива, определяется неравенством

$$\lambda < \frac{1}{1 + 3,31\beta}. \quad (4.66)$$

Средняя задержка МДПН/ОК в описанной синхронной модели и при наличии идеальной стабилизации вычисляется так же, как задержка МДПН (см. задачу 4.24). Результат при малом β и среднем квадрате длины пакета \bar{X}^2 имеет вид

$$W \approx \frac{\lambda \bar{X}^2 + \beta (4,62 + 2\lambda)}{2 [1 - \lambda (1 + 3,31\beta)]}. \quad (4.67)$$

Константа 3,31 в выражении для максимальной скорости передачи (4.66) зависит от детальных предположений относительно системы. Когда делаются различные предположения, получаются различные значения (см., например, [157]). Если β очень мало, что типично для ETHERNET, то величина этой константы не имеет большого значения. Однако следует отметить, что несинхронная версия МДПН/ОК представляется значительно более разумной по сравнению с синхронной версией как из-за трудности синхронизации в коротких мини-окнах, так и из-за преимуществ использования меньших значений (по сравнению с максимальным значением) времени распространения, когда это возможно.

Несинхронный МДПН/ОК

На рис. 4.20 иллюстрируется причина того, что анализ несинхронной системы является довольно сложным. Предположим, что узел, находящийся на одном конце кабеля, начинает передачу и затем спустя почти β единиц времени начинает передачу узел, находящийся на другом конце. Второй узел прекращает передачу почти сразу после того, как начинает прослушивать передачу первого узла, но тем не менее вносит ошибки в первый пакет и вынуждает первый узел остановить передачу спустя еще β единиц времени. Наконец, еще β единиц времени проходит, пока затухнут сигналы на другом конце линии. Другая сложность

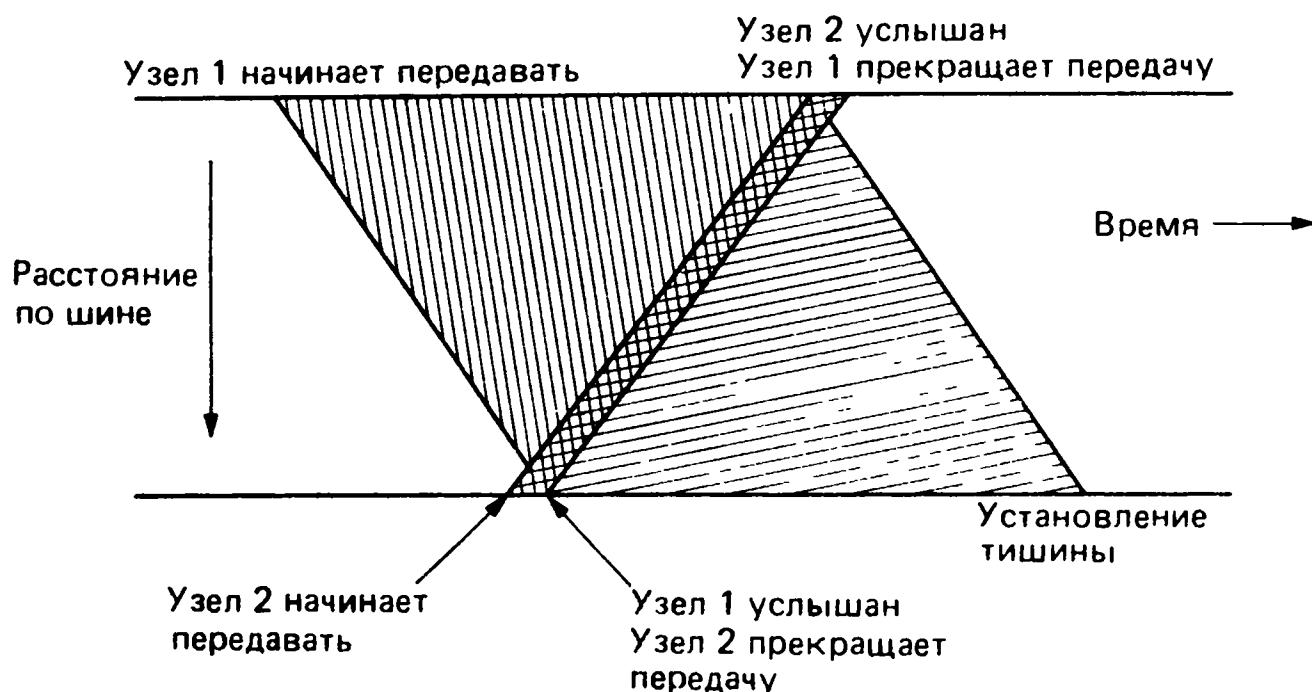


Рис. 4.20. Обнаружение конфликта. Узел 2 начинает передавать почти через β единиц после начала передачи узла 1; узел 2 тут же прекращает передачу, но узел 1 продолжает передачу еще почти β единиц перед тем, как прекратить ее.

связана с тем, что узлы, которые расположены на кабеле ближе, обнаруживают конфликты быстрее, чем те, которые расположены дальше. В результате максимальная скорость передачи, достижимая в сети ETHERNET, зависит от расположения узлов на кабеле и точное вычисление ее очень сложно.

Однако, чтобы построить верную, но заниженную границу максимальной скорости передачи, можно найти границы для всех параметров, существенных при рассмотрении интервала от конца одной передачи (как успешной, так и прерванной) до конца следующей. Предположим, что каждый узел инициирует передачи в соответствии с независимым пуассоновским процессом всякий раз, когда он обнаруживает, что канал пустой; обозначим через G интенсивность суммарного пуассоновского процесса. Все узлы обнаруживают начало пустого периода не более чем через β единиц после окончания передачи, а среднее время до начала следующей передачи не превышает $1/G$. Следующий пакет вступит в конфликт с некоторым начавшим передаваться позднее пакетом с вероятностью, не превышающей $1 - e^{-\beta G}$, и вступившие в конфликт пакеты прервут передачу максимум через 2β единиц. Однако передача пакета будет успешной с вероятностью, не меньшей $e^{-\beta G}$, и на это будет затрачена 1 единица времени. Скорость ухода S при заданной G равна вероятности успешной передачи, деленной на среднюю длину интервала с успешной передачей или конфликтом, так что

$$S > \frac{e^{-\beta G}}{\beta + 1/G + 2\beta (1 - e^{-\beta G}) + e^{-\beta G}}. \quad (4.68)$$

Оптимизируя правую часть (4.68) по G , находим, что максимум достигается при $\beta G = (\sqrt{13} - 1)/6 = 0,43$; соответствующее максимальное значение равно

$$S > \frac{1}{1 + 6,2\beta}. \quad (4.69)$$

Этот анализ приводит к слишком заниженному результату, но если β мало, то можно достичь скоростей передачи, очень близких к 1, и разница между (4.66) и (4.69) становится небольшой. Заметим, что максимальная скорость устойчивой передачи стремится к 1 при уменьшении β ; при МДПН/ОК это выражается константой, умноженной на β , а при МДПН — константой, умноженной на $\sqrt{\beta}$. Причина этого различия в том, что при МДПН/ОК на конфликты тратится не очень много времени и поэтому можно использовать намного более высокие интенсивности попыток передать пакет. По этой же причине настойчивый МДПН (при котором новые пакеты, поступающие во время окна передачи данных, передаются сразу после окончания этого окна) работает приемлемо в модели с МДПН/ОК, но довольно плохо в модели с МДПН.

МДПН/ОК (и МДПН) становятся в возрастающей степени неэффективными при увеличении длины шины, повышении скорости передачи данных и уменьшении длины пакета данных. Чтобы увидеть это, вспомним, что β измеряется в длинах пакета данных. Таким образом, если τ — время распространения (и время приема) в секундах, C — скорость передачи данных шины и L — средняя длина пакета, то $\beta = \tau G/L$. Ни МДПН/ОК, ни МДПН не представляют приемлемый выбор для системы, у которой β составляет несколько десятых или более.

Стандарты IEEE 802

Институт инженеров по электротехнике и радиоэлектронике (IEEE) разработал ряд стандартов 802, которые предназначены для ЛС (локальных сетей). Стандарты разделяются на пять частей, которые имеют номера от 802.1 до 802.5. Стандарт 802.1 относится к интерфейсу протоколов ЛС с более высокими уровнями. 802.2 является стандартом управления линией передачи данных, аналогичным HDLC, который рассматривался в гл. 2. Наконец, 802.3 — 802.5 являются стандартами управления доступом к среде (УДС), которые называются МДПН/ОК, система с передачей маркера по шине и система с передачей маркера по кольцу соответственно. Стандарт 802.3, по существу, совпадает с методом доступа сети ETHERNET, использующей несинхронный настойчивый МДПН/ОК с замедлением по двоичной экспоненте. Мы обсудим кратко стандарты 802.5 и 802.4 в следующих двух подразделах.

4.5.3 Локальные сети. Кольца с передачей маркера

Кольцевые сети с передачей маркера [66, 67] представляют другой распространенный подход к управлению доступом в локальных сетях. В таких сетях узлы логически расположены так, что они образуют кольцо, в котором каждый узел передает следующему узлу на кольце (рис. 4.21). Обычно каждый узел просто ретранслирует принятый от предыдущего узла поток битов следующему узлу. Он делает это по крайней мере с однократной задержкой, позволяющей узлу прочитать и регенерировать поступающую двоичную цифру перед ее передачей следующему узлу. Естественно, когда узел передает свой собственный пакет следующему узлу, он должен сбрасывать то, что принимается. Для правильной работы системы следует обеспечить, чтобы то, что принимается и сбрасывается, не являлось пакетом, который еще не достиг своего пункта назначения. Концептуально мы представляем, что в сети существует маркер, который переходит от узла к узлу. Всякий раз, когда узел получает маркер, ему разрешается передать пакет и после передачи пакета маркер посылается следующему узлу. Узлы, которым нечего передавать, обязаны отправлять маркер, не задерживая его.

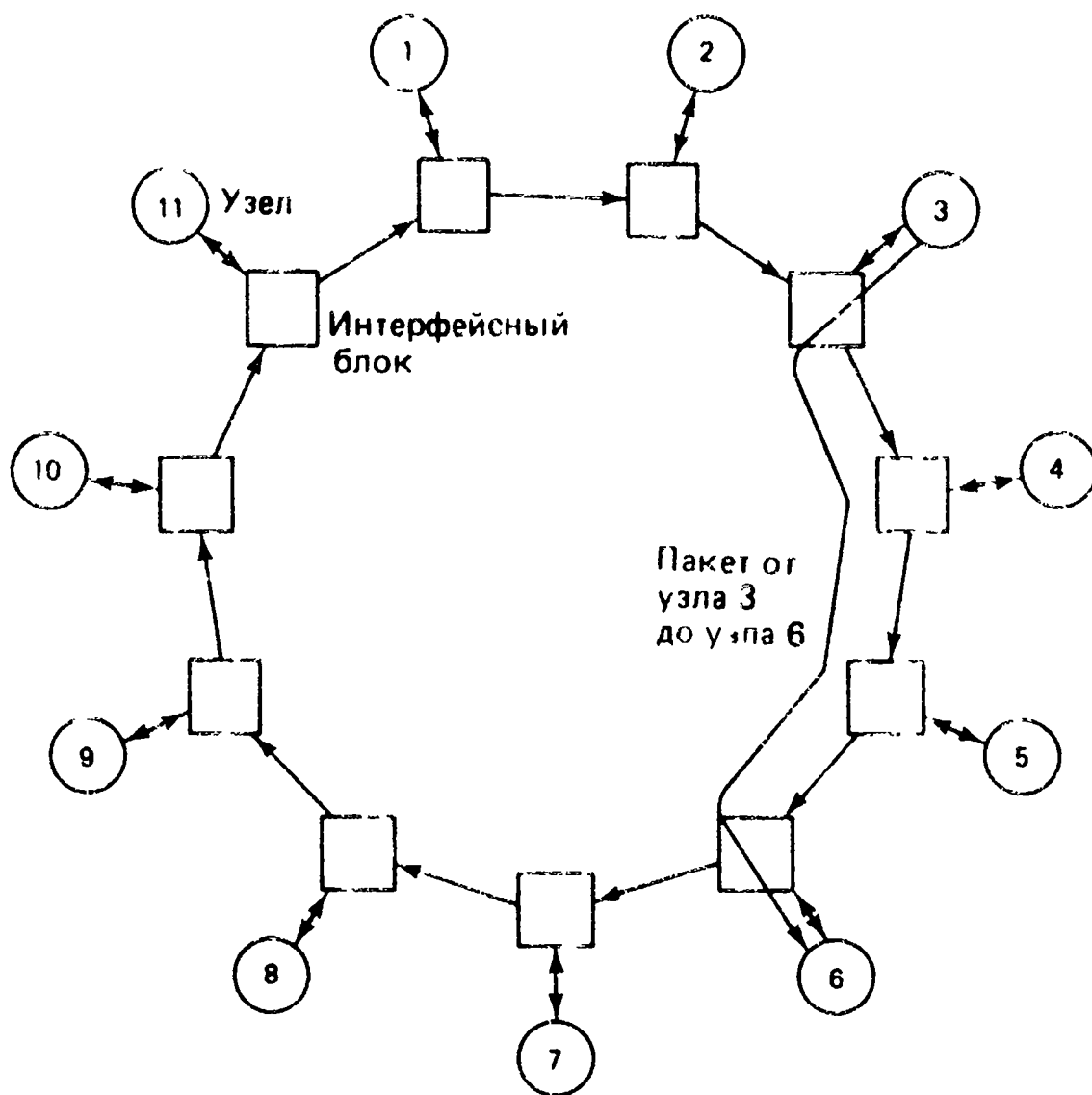


Рис. 4.21. Кольцевая сеть. Данные перемещаются по кольцу в одном направлении. Каждый узел или ретранслирует с однократной задержкой принимаемый поток битов следующему узлу или передает свой пакет, сбрасывая принимаемый поток битов.

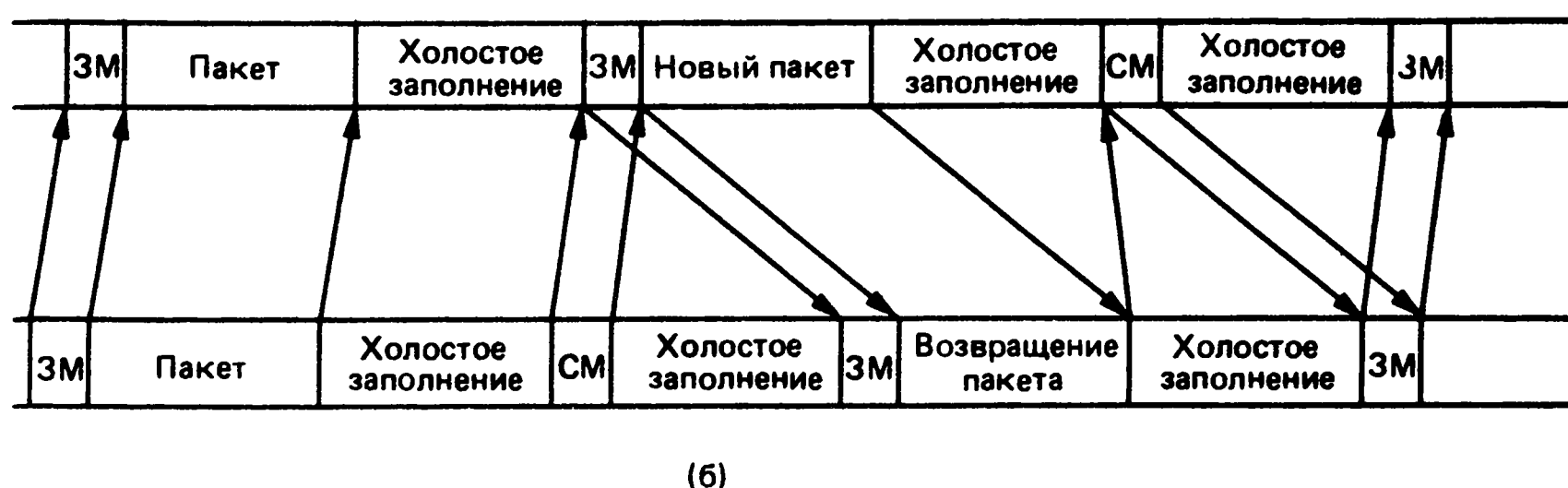
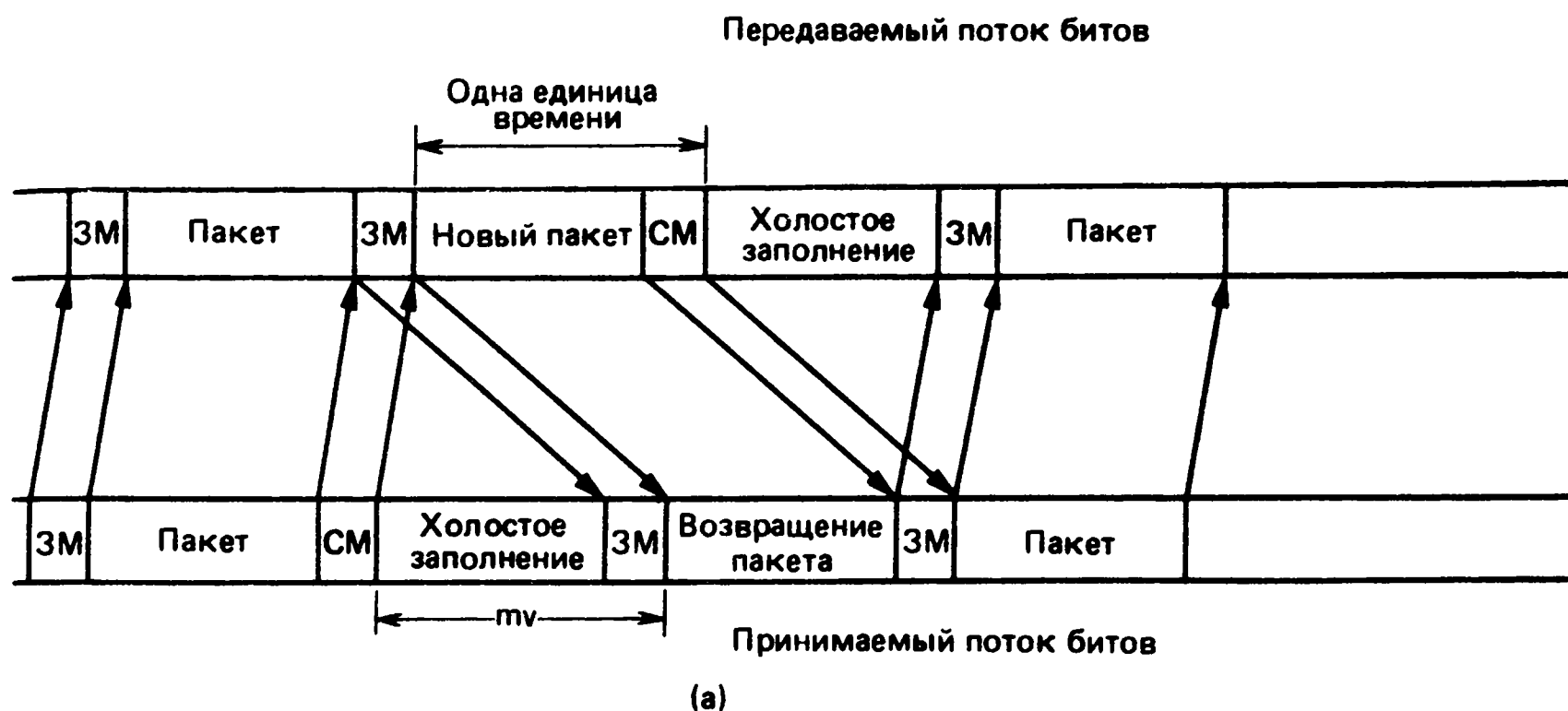


Рис. 4.22. Передаваемый и принимаемый потоки битов на одном из кольцевых интерфейсных блоков кольцевой сети с передачей маркера. Интерфейсный блок передает то, что он принимает, с одноканальной задержкой до тех пор, пока не получит свободный маркер (СМ). Он превращает свободный маркер в занятый маркер (ЗМ) и затем передает свой пакет. На части (а) за пакетом следует свободный маркер, тогда как на части (б) интерфейс блок ждет возвращения прошедшего по кольцу пакета, а затем передает свободный маркер. В каждом случае после свободного маркера передается холостое заполнение до тех пор, пока не вернется маркер (занятый или свободный), прошедший по кольцу, а затем блок возвращается к ретрансляции того, что принимается, с одноканальной задержкой.

Когда мы рассматриваем свойства, которыми должен обладать маркер, мы видим, что они, по существу, совпадают со свойствами флагов, изучавшихся при обсуждении УЛПД, и что эти флаги можно использовать как маркеры и как средство обозначения конца пакета, т. е. всякий раз, когда узел, находящийся в состоянии передачи пакета, заканчивает эту передачу, он обычно помещает маркер или флаг, например последовательность 01111110, в конце пакета. Когда следующий узел прочитывает этот маркер, он просто отправляет маркер дальше, если у него нет пакетов для передачи, но если у него есть пакет для передачи, то он инвертирует последний бит маркера, превращая маркер в последовательность 01111111. Этот модифицированный маркер 01111111

обычно называется *занятым маркером*, а исходная последовательность 01111110 — *свободным маркером*. Вслед за занятым маркером узел передает свой собственный пакет. При передаче пакетов данных используется битовая вставка, а именно вставка 0 после 011111, для того чтобы не допустить появления обоих маркеров в потоке данных. Таким образом, каждый узел может разделить принимаемый поток на пакеты, выделяя свободные и занятые маркеры, а свободный маркер осуществляет пересылку права на передачу от одного узла к другому.

Посмотрим более внимательно на то, как пакеты перемещаются по кольцу. Допустим, например, что в момент времени 0 узел 1 принимает свободный маркер, инвертирует последний бит, формируя занятый маркер, и затем начинает передавать пакет (рис. 4.22, а). Каждый последующий узел кольца задерживает этот поток битов на один бит и ретранслирует его следующему узлу. Получатель, которому предназначен этот пакет, считывает пакет в память своего узла и, кроме того, ретранслирует его дальше по кольцу.

Спустя время круговой передачи поток битов возвращается к отправителю, т. е. в узел 1 в нашем примере. Время круговой передачи определяется как сумма времени распространения сигнала по кольцевому каналу и времени передачи t бит, где t — число узлов. Предположим, что длина пакета больше, чем время круговой передачи (выраженное в битах); тогда начальная часть поступающего пакета автоматически удаляется узлом 1, поскольку узел 1 продолжает передавать последующую часть пакета. Когда узел 1 завершает передачу пакета, он добавляет свободный маркер и затем посылает холостое заполнение, тогда как оставшаяся часть только что переданного пакета возвращается в узел 1, пройдя по кольцу. После возвращения последнего бита пакета узел 1 начинает ретранслировать то, что к нему поступает, с однокбитовой задержкой. Если некоторый другой узел уже имел пакет для передачи, то узел 1 сначала ретранслирует занятый маркер, а затем этот пакет; если ни один узел не имел пакета для передачи, то свободный маркер ретранслируется узлом 1 и продолжает циркулировать до тех пор, пока какой-либо узел не будет иметь пакета для передачи.

Так как все узлы следуют этой стратегии, то после того, как свободный маркер поступает в узел 1 как часть принимаемого потока битов, за ним должно следовать холостое заполнение. Это холостое заполнение сохраняется до тех пор, пока узел, посылающий это холостое заполнение, не начнет ретранслировать занятый маркер, посланный узлом 1 (см. рис. 4.22). Таким образом, занятые маркеры всегда сопровождаются пакетами, а свободные маркеры всегда сопровождаются холостым заполнением, которое дополняет их так, чтобы заполнить все кольцо.

Заметим, что время круговой передачи должно быть больше длины маркера или по крайней мере равным ей; в противном случае узел, завершая передачу пакета и посылая свободный маркер, будет сбрасывать начальную часть маркера, когда он возвратится в узел, пройдя по кольцу. Одна из точек зрения на это состоит в том, что память кольца (т. е. длина распространения в битах плюс число узлов) должна быть достаточной для запоминания маркера. Так как узел, передающий пакет, также считает его перед тем, как удалить из сети, то в передающем узле можно с помощью побитовой проверки или вычислением ЦИП убедиться в том, что пакет принят безошибочно. Принимающий узел может также выставлять бит в заданную позицию в конце пакета в случае положительного результата вычисления ЦИП. На самом деле ни один из этих методов не обеспечивает надежной защиты, так как ошибка может появиться на участке от принимающего узла до его кольцевого интерфейса или при передаче того бита, который выставлен принимающим узлом.

Существует много разновидностей кольцевой сети. Каждому узлу можно разрешить передавать только один пакет всякий раз, когда он получает маркер, или узел может очищать свою очередь ожидающих пакетов перед тем, как отправить маркер следующему узлу. В кольцевой сети, кроме того, можно довольно легко установить приоритеты, вводя поле приоритетов в фиксированной позиции в конце свободного или занятого маркера, любой узел может изменить значения битов в этом поле, чтобы присвоить пакету высокий приоритет. Другие узлы с менее приоритетным трафиком могут ретранслировать свободные маркеры, а не использовать их с тем, чтобы узлы с высокоприоритетными пакетами передавали в первую очередь. Очевидно, что, когда узел передаст свои высокоприоритетные пакеты, ему следует уменьшить значение приоритетного поля.

Другая разновидность кольцевой сети связана с использованием ARQ. Если узел передает свободный маркер вслед за пакетом, то этот узел не будет в дальнейшем иметь контроль над каналом, если пакет не был доставлен безошибочно. Альтернативное решение заключается в том, что узел посылает холостое заполнение после завершения передачи пакета до тех пор, пока не установит, был ли пакет принят безошибочно (см. рис. 4.22, б). Если прием безошибочный, то передается свободный маркер; в противном случае посылается занятый маркер, за которым следует повторная передача. Как показано на рисунке, каждому занятому или свободному маркеру предшествует холостое заполнение, длина которого равна времени кругового распространения. Свободные маркеры также сопровождаются холостым заполнением. Это альтернативное решение упрощает в некоторой степени наблюдение за тем, что происходит в кольце (так как не более

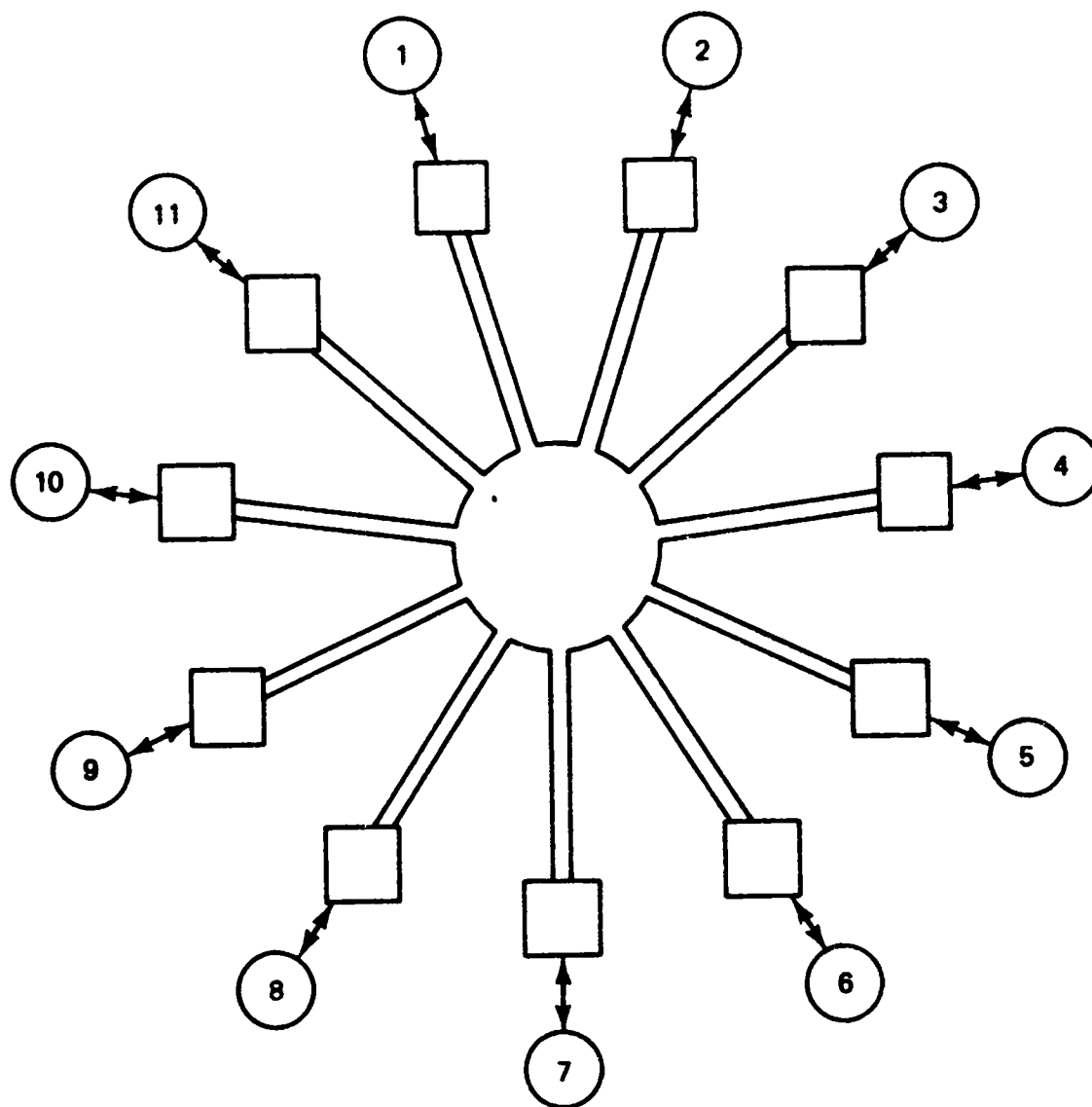


Рис. 4.23. Кольцевая сеть звездообразной конфигурации. Узлы можно обходить или подключать, находясь в центральном пункте.

чем один пакет активен в каждый момент времени), но оно понижает эффективность и увеличивает задержку особенно в больших кольцах.

Еще одна разновидность кольцевой сети связана с физическим расположением кольца. Если образующий кольцо кабель установить в соответствии со звездообразной конфигурацией, как показано на рис. 4.23, то появится ряд преимуществ. Во-первых, недостаток кольцевой сети заключается в том, что каждый узел требует активного интерфейса, в котором каждый бит считывается и передается дальше. Если интерфейс выходит из строя, то все кольцо повреждается. Соединяя все интерфейсы в общем месте, легче найти поврежденный интерфейс и обойти его в центральном пункте. Если интерфейс тоже располагается в центральном пункте (чего обычно не бывает), то время распространения по кольцу существенно уменьшается.

Наиболее важные разновидности кольцевых сетей связаны с восстановлением поврежденного или занятого маркера. Если свободный маркер разрушается шумом или несколько свободных маркеров возникает, или же занятый маркер возникает и циркулирует до бесконечности, то система выходит из строя. Одно очевидное решение этой проблемы заключается в том, что назначается специальный узел, ответственный за восстановление уте-

рянного свободного маркера или удаление ложных маркеров; это довольно сложно из-за возможности повреждения специального узла или его отключения от сети.

Стандарт IEEE 802.5 кольцевой сети с передачей маркера

Более распространен метод восстановления маркера, используемый в стандарте IEEE 802.5, когда каждый узел должен обнаружить потерю маркера или существование нескольких маркеров после некоторого тайм-аута. Если после окончания тайм-аута узел имеет пакет для передачи, то он передает занятый маркер, пакет и затем свободный маркер, одновременно освобождая кольцо от всех остальных маркеров. В случае успеха кольцо становится снова работоспособным; в случае неудачи, обусловленной тем, что два или более узла пытались исправить ситуацию одновременно, вступившие в конфликт узлы производят новые попытки спустя случайное время¹⁾.

Кроме того, в стандарте IEEE 802.5 применяется звездообразная конфигурация и отправление маркера откладывается до тех пор, пока текущий пакет не получит подтверждение о приеме. Этот стандарт предусматривает не 8-битовый, а 24-битовый маркер и тщательно разработанные процедуры восстановления работоспособности при многих возможных сбоях.

Стандарт кольцевой сети с передачей маркера, обеспечивающей скорость 4 Мбит/с, был реализован посредством нескольких СБИС; сложность этих схем при простой концепции кольца с передачей маркера поистине удивительна.

Средняя задержка в кольцевой сети с передачей маркера

Исследуем среднюю задержку в кольце с передачей маркера. Предположим сначала, что каждый узел, получив свободный маркер, очищает свою очередь перед тем, как отправить свободный маркер следующему узлу. Допустим, что имеется m узлов, в каждый из которых поступает независимый пуассоновский поток с интенсивностью λ/m . Пусть v обозначает среднее время распространения от одного узла до следующего плюс задержка ретрансляции (обычно один бит или несколько битов) в узле²⁾. Из центрального пункта проследим за перемещением свободного маркера по кольцу. Пусть $\bar{X} = 1$ обозначает среднее время перес-

¹⁾ Это означает, что здесь используется процедура случайного множественного доступа. — Прим. ред.

²⁾ Видимо, считается, что v не среднее время, а неслучайное время распространения плюс задержка ретрансляции. Иначе в (4.70) добавляется член $\sigma_v^2/2\bar{V}$. — Прим. ред.

дачи пакета (в том числе его занятого маркера). Таким образом, $\rho = \lambda$. Средняя задержка W здесь такая же, как в системе массового обслуживания с очищением; задержка определяется равенством (3.69) из гл. 3

$$W = \frac{\lambda \overline{X^2}}{2(1-\lambda)} + \frac{(m-\lambda)v}{2(1-\lambda)}. \quad (4.70)$$

Заметим, что первый член определяет обычное время ожидания в системе $M/G/1$. Второй член дает задержку при нулевой нагрузке, а именно $mv/2$. Обратим внимание на то, что mv — это время передачи по всему кольцевому каналу, т. е. β плюс m , умноженное на задержку одного или нескольких битов в интерфейсе каждого узла. Если β мало по сравнению с временем передачи пакета и λ относительно велика, то W очень близка к задержке системы $M/G/1$.

Теперь рассмотрим ситуацию, когда узел может передать не более одного пакета с каждым свободным маркером. В этом случае мы имеем систему с ограниченным обслуживанием и частичным отсечением из подразд. 3.5.2. Задержка определяется равенством (3.76)

$$W = \frac{\lambda \overline{X^2} + (m' + \lambda)v}{2(1-\lambda-\lambda v)}. \quad (4.71)$$

В этом случае мы замечаем, что максимальная скорость устойчивой передачи уменьшилась до $1/(1+v)$. В приведенном выше анализе считалось, что маркер входит в состав управляющей части пакета, и включали в v только время распространения плюс задержку бита в интерфейсе узла. Это не вопрос терминологии, а необходимое условие использования результатов анализа очередей из гл. 3. Согласно этому анализу, v — задержка, вносимая узлом даже тогда, когда узлу нечего передавать; эта задержка определяется не всем маркером, а одним или несколькими битами переходной задержки узла. В задаче 4.27 модифицируется (4.71) для случая, когда передающий узел ожидает возвращения пакета (как в стандарте IEEE 802.5) перед тем, как отправить свободный маркер. По существу, это добавляет время круговой передачи (т. е. mv) к времени передачи каждого пакета, и максимальная скорость передачи уменьшается до $1/(1+mv)$. Это может привести к значительному падению скорости, если mv велико.

Сравнивая кольцевую сеть с передачей маркера и систему МДПН/ОК, заметим, что если время распространения и обнаружения мало по сравнению с временем передачи пакета, то максимальные скорости передачи обеих систем близки к одному пакету за единицу времени. В кольцевой сети с передачей маркера не возникает проблем устойчивости, тогда как МДПН/ОК не имеет сложностей, связанных с потерей маркеров, и дает немного мень-

шую задержку при очень слабой нагрузке. Оба метода хорошо разработаны и выбор между ними можно делать на основе стоимости, достижимой надежности и личного предпочтения. Если время распространения велико ($\beta > 1$), то МДПН/ОК теряет свое преимущество перед простым разрешением конфликтов и версия IEEE 802.5 кольца с передачей маркера имеет плохие характеристики (так как $tv > \beta$). Вместе с тем у кольца с передачей маркера, в котором узлы отправляют свободный маркер сразу после окончания передачи пакета, ухудшения характеристик не происходит (так как максимальная скорость передачи равна $1/(1 + v)$ и v может быть намного меньше β , если t велико).

Кольца с тактированным доступом и кольца с регистрами

Предположим, что нагрузка на кольцо равномерно распределена между различными парами источник—пункт назначения; тогда пакет нужно передавать в среднем только по половине линий кольца. Так как в кольце с передачей маркера пакет передается по каждой линии, половина пропускной способности системы потенциально не используется. Поэтому понятно, что другая стратегия управления могла бы увеличить пропускную способность вдвое.

Кольца с тактированным доступом и кольца с регистрами позволяют достичь более высокой пропускной способности, по крайней мере в принципе. Кольцо с тактированным доступом наиболее удобно представлять как конвейерную ленту окон для передачи пакетов; кольцо расширяется регистрами сдвигов в узлах, чтобы получить нужное число окон. Когда узел имеет пакет для передачи, он находит свободное окно на конвейерной ленте и помещает пакет в это окно, помечая, что окно заполнено. Когда узел назначения обнаруживает этот пакет, он удаляет его и помечает, что окно опять свободно.

Недостаток кольца с тактированным доступом состоит в том, что все пакеты должны иметь равную длину (в отличие от колец с передачей маркера и МДПН/ОК). Другим недостатком является значительная задержка (обусловленная длиной конвейерной ленты) даже при слабой нагрузке. Это можно компенсировать, делая пакеты очень короткими, но дополнительная управляющая информация УЛПД, вызванная укорочением пакетов, приводит к потере большей части потенциального выигрыша в скорости передачи. Наконец, при выполнении ARQ общепринято оставлять пакеты в их окнах до тех пор, пока они не вернутся к посылавшему узлу; это автоматически аннулирует потенциальное удвоение пропускной способности передачи.

Кольцо с регистрами обеспечивает передачу кольцевого трафика с промежуточным накоплением в буферах узлов. Каждый

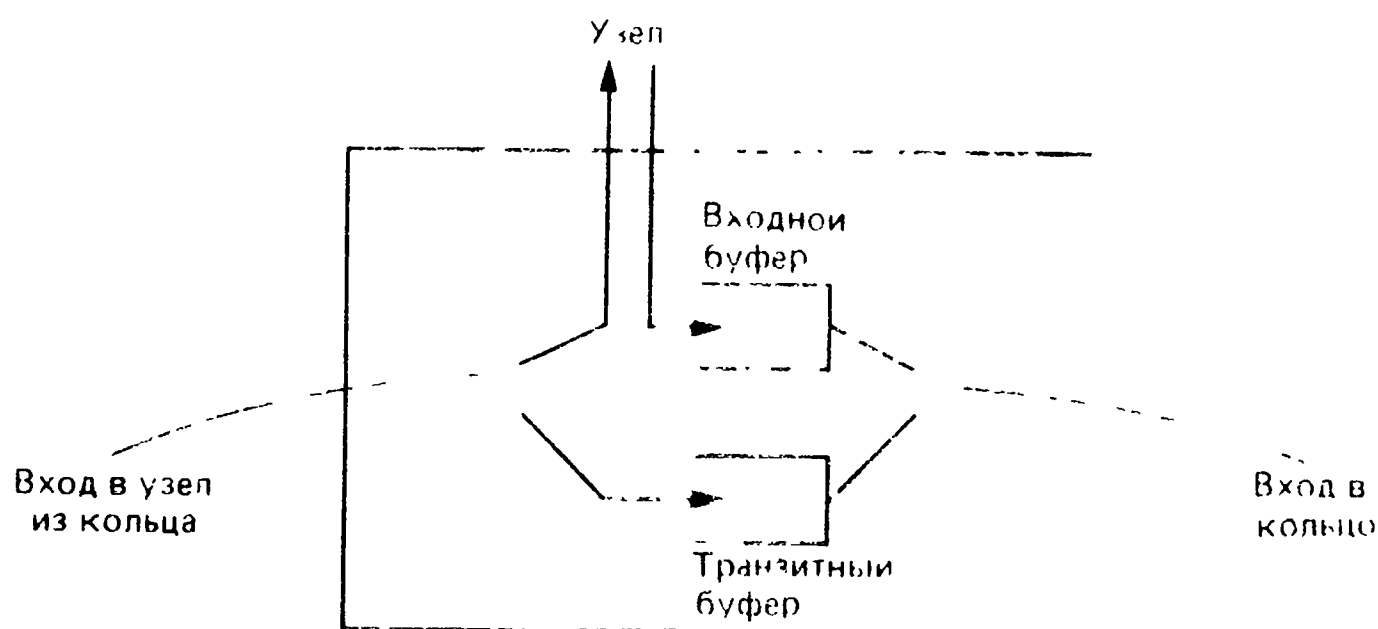


Рис. 4.24. Интерфейс для кольца с регистрами. В кольцо поступает содержимое входного или транзитного буферов. Входные пакеты из кольца направляются в транзитный буфер или узел, если они ему адресованы.

Узел имеет буфер для транзитного трафика и буфер для новых поступающих пакетов (рис. 4.24). Когда новый пакет передается, входящий трафик кольца, который необходимо транзитом передать дальше, поступает в транзитный буфер. Когда транзитный трафик передается, транзитный буфер постепенно освобождается, если на вход узла из кольца поступает холостое заполнение или пакеты, предназначенные для данного узла. Новым пакетам не разрешается дальнейшая передача, если в транзитном буфере недостаточно места для того, чтобы запоминать поступающий из кольца трафик.

Кольцо с регистрами обладает способностью обеспечения более высоких скоростей передачи и ненамного большей задержки, чем в кольце с передачей маркера. Главный недостаток такого кольца состоит в том, что оно не обеспечивает справедливого распределения и гарантированного доступа, такого как в кольце с передачей маркера в круговом порядке. Кольцо с передачей маркера намного популярнее на практике, вероятно, из-за того, что максимальная скорость передачи не является определяющим фактором для большинства локальных сетей.

15.4. Локальные сети. Шины с передачей маркера и метод опроса

Основная идея кольца с передачей маркера широко используется в других сетях связи. Идея состоит в том, что существует *m* упорядоченных узлов, которые обслуживаются в круговом порядке. Различия между этими системами состоят в том, как один узел узнает, что предыдущий закончил обслуживание (или отказался от предложенного ему обслуживания). Другими словами,

какой механизм выполняет роль маркера и какова задержка ν при передаче виртуального маркера от одного узла к следующему?

Система с опросом, краткое обсуждение которой проведено в подразд. 4.1.2, является широко распространенным примером такой системы. Запросы, посылаемые центральным узлом каждому из вторичных узлов, играют роль маркеров. Задержка передачи маркера в системе с опросом довольно велика, поскольку она определяется сначала передачей от опрашиваемого узла к центральному узлу, а затем передачей нового запроса от центрального узла к следующему вторичному узлу.

Централизованный опрос является одним из способов избавления от этой двойной задержки. Центральная станция опрашивает (т. е. передает маркер) первый вторичный узел; каждый вторичный узел после использования канала передает маркер следующему вторичному узлу. Если порядок опроса узлов отражает их расположение на многоточечной телефонной линии или шине, то задержка передачи маркера уменьшается еще больше.

Шину с передачей маркера можно реализовать на физической шине того же типа, что используется для системы МДПН/ОК. Узлы упорядочиваются так, чтобы образовалось логическое кольцо; когда узел заканчивает передачу своего пакета или пакетов, он посылает маркер следующему по порядку узлу, отдавая ему право передавать следующим. Узел, которому нечего передавать, посылает маркер следующему узлу. Нетрудно увидеть, что в концептуальном плане шина с передачей маркера, по существу, аналогична системе с централизованным опросом. Термин «опрос» больше принято использовать для систем с центральным узлом, а термин «шина с передачей маркера» — для полностью распределенных систем. Равенства (4.70) и (4.71) определяют задержку в этих системах для случаев, когда по приходу запроса передаются все пакеты из очереди или один пакет соответственно. Оба равенства получены в предположении, что во все узлы поступает одинаковый средний трафик. Параметр ν в этих равенствах можно интерпретировать в общем случае как время от момента прихода маркера в один узел пустой системы до момента прихода в следующий узел, усредненное по узлам.

Вспомним, что в кольце с передачей маркера время ν составляется из времени распространения от узла к узлу и задержки порядка одного или нескольких битов в узле; инициирование передачи маркера в узле начинается до того, как маркер будет полностью принят от предыдущего узла. Здесь длина маркера (которая может быть значительной) входит в ν , поскольку маркер должен быть полностью принят и декодирован до того, как начнет действовать следующий узел. Таким образом, в пределе при нулевой нагрузке шине с передачей маркера свойственна большая средняя задержка, чем кольцу с передачей маркера.

Характеристики шин с передачей маркера и систем с опросом в решающей мере зависят от параметра v (в том смысле, что, как следует из (4.71), максимальная скорость передачи равна $1/(1 + v)$, а средняя задержка в пределе при $\lambda \rightarrow 0$ составляет $mv/2$). Предполагая, что узлы пронумерованы независимо от положения на шине, разумно считать, что среднее время распространения равно $\beta/3$ (см. задачу 4.29), где $\beta = \tau C/L$ является нормированным временем распространения от одного конца шины до другого (т. е. временем, представленным как доля средней длины пакета). Здесь τ — время распространения в секундах, C — скорость передачи битов по каналу, а L — средняя длина пакета. Аналогично нормированное среднее время передачи маркера равно k/L , где k обозначает длину маркера. Таким образом,

$$v = \frac{\tau C}{3L} + \frac{k}{L} + \delta, \quad (4.72)$$

где δ — нормированное время обнаружения маркера приемником. В предыдущих рассмотрении мы считали δ частью времени распространения β , но здесь мы учитываем это время отдельно.

Величина τC равна числу битов, одновременно перемещающихся по шине. Если $\tau C/3$ мало по сравнению с k , то улучшение характеристик алгоритма определяется уменьшением длины маркера. Наоборот, если $\tau C/3$ велико, то величина k не является определяющей, а улучшения зависят от уменьшения времени распространения. Очевидный способ ослабления влияния времени распространения состоит в том, что узлы нумеруются последовательно от одного конца шины до другого. Если это сделано, то сумма времен распространения равна 2β ; она состоит из совокупного времени распространения β при движении по шине в направлении возрастания номеров при опросе всех узлов, а также и еще одного β , необходимого для возврата в начало шины. Таким образом, средняя величина v равна

$$v = \frac{2\tau C}{mL} + \frac{k}{L} + \delta. \quad (4.73)$$

Это основной способ сокращения времени распространения, но он затрудняет подключение новых узлов к шине. Заметим, что времена распространения много меньше среднего значения, когда процесс резервирования идет от начала шины до ее конца, а затем наступает длинный интервал резервирования, равный β , необходимый для возврата из конца шины в ее начало. Вспомним, что, как найдено в подразд. 5.2 гл. 3, равенства (4.70) и (4.71) верны, если используется среднее значение v .

Стандарт IEEE 802.4 шины с передачей маркера

Стандарт IEEE 802.4 соответствует, по существу, описанной выше системе. Кратко рассмотрим некоторые его особенности.

Чтобы разрешить включение новых узлов в логическое кольцо передачи маркера, каждый узел, уже принадлежащий кольцу, периодически посылает специальный контрольный пакет, который приглашает ожидающие узлы присоединиться. Все ожидающие узлы посылают ответ, и если их оказывается больше одного, то используется алгоритм разбиения для выделения одного из них. Новый узел включается в логическое кольцо после приглашающего узла и в дальнейшем адресует маркер узлу, который прежде следовал за приглашающим узлом. Старый узел может отключиться от логического кольца, послав контрольный пакет своему предшественнику; в контрольном пакете указывается, что предшественник должен посылать последующие маркеры узлу, который следует за отключающимся узлом. Наконец, восстановление работоспособности при возникновении сбоя производится, по существу, отключением всех узлов, последующим выделением одного узла на основе состязаний и, наконец, подключением новых узлов вслед за выделенным в соответствии с указанной процедурой.

Неявные маркеры и МДПН/ИК

Рассмотрим вопрос о том, как можно уменьшить длину маркера. Общий подход состоит в том, что маркер заменяется на неявный маркер, который обозначается пустым состоянием канала. Двумя наиболее известными системами, использующими этот подход, являются BRAM [44] и MSAP [148]. В этих схемах после окончания передачи пакета узел замолкает. Следующий по порядку узел, обнаружив пустой канал, начинает передачу, если он имеет пакет, или продолжает молчать в противном случае. Последующие узлы ждут последовательно все большие интервалы времени после обнаружения пустого состояния перед тем, как начать передачу, что дает каждой более ранней станции возможность передачи пакетов. Эти схемы часто называют МДПН с избежанием конфликтов (МДПН/ИК).

Чтобы понять, как долго узел должен слушать пустой канал перед тем, как начать передачу, рассмотрим наихудший случай, когда узел, заканчивающий передачу пакета, находится на одном конце шины, второй — на противоположном конце и третий узел — опять на первом конце. Тогда третий узел обнаружит, что канал пуст, почти немедленно после окончания передачи, но второй узел обнаружит это событие только спустя $\beta + \delta$ единиц и третий узел узнает, собирается ли передавать первый узел, только спустя дополнительно еще $\beta + \delta$. Таким образом, третий узел должен ждать $2(\beta + \delta)$ перед тем, как начать передачу. Рассуждая аналогичным образом, получаем, что каждый последующий узел обязан ждать дополнительно в течение времени $2(\beta + \delta)$.

Таким образом, в этой схеме явный маркер длины k/L заменен на неявный маркер длины $2(\beta + \delta) = 2(\tau C/L + \delta)$. Такая схема поэтому предпочтительнее в тех ситуациях, когда τC и δ малы.

Если узлы упорядочены на шине, а задержки известны и учитываются алгоритмом, то длины неявных маркеров значительно сокращаются так же, как в (4.73), но сложность системы значительно возрастает. Существует много разновидностей этой системы, различающихся способами преодоления трудностей поддержания синхронизации во время длинных пустых периодов и исправления ошибок (см. [71], где дается критический обзор результатов).

4.5.5. Высокоскоростные локальные сети

Рост требований к передаче данных наряду с появлением передающих сред с высокой скоростью передачи данных, таких, как оптическое волокно, коаксиальный кабель и CATV (кабельное телевидение), приводит к использованию локальных сетей, обеспечивающих более высокую скорость передачи. Мы определим здесь высокоскоростную локальную сеть как такую, у которой β превышает 1. Вспомним, что β — это отношение времени распространения к среднему времени передачи пакета, так что $\beta > 1$ означает, что передатчик закончит передачу пакета до того, как удаленный приемник начнет прослушивать его. Поскольку $\beta = \tau C/L$, увеличение времени распространения τ и скорости передачи данных C , а также уменьшение средней длины пакета L способствуют тому, чтобы сеть стала высокоскоростной. Кроме того, раньше возникал большой интерес к использованию методов локальных сетей на больших площадях покрытия, таких, как площадь города; большие площади могут привести к тому, что сеть станет высокоскоростной даже при умеренной скорости передачи данных.

Мы уже видели, какое воздействие оказывает β на характеристики локальных сетей. Алгоритм МДПН/ОК хуже работает при увеличении β и становится бессмысленным при $\beta > 1$. Если β мало по отношению к числу узлов m , то, согласно (4.71), кольцо с передачей маркера обеспечивает большую максимальную скорость передачи и малую задержку при малых λ . Заметим, что в (4.71) предполагается, что каждый узел передает свободный маркер сразу после завершения своей передачи. Если узел ожидает, пока его пакет распространится по кольцу, а затем освобождает маркер (как в стандарте IEEE 802.5), то не более одного пакета может быть передано за каждые $1 + \beta$ единиц времени и максимальная скорость передачи уменьшается как $1/\beta$.

Скорость передачи шины с передачей явного и неявного маркера также уменьшается как $1/\beta$, если узлы нумеруются случай-

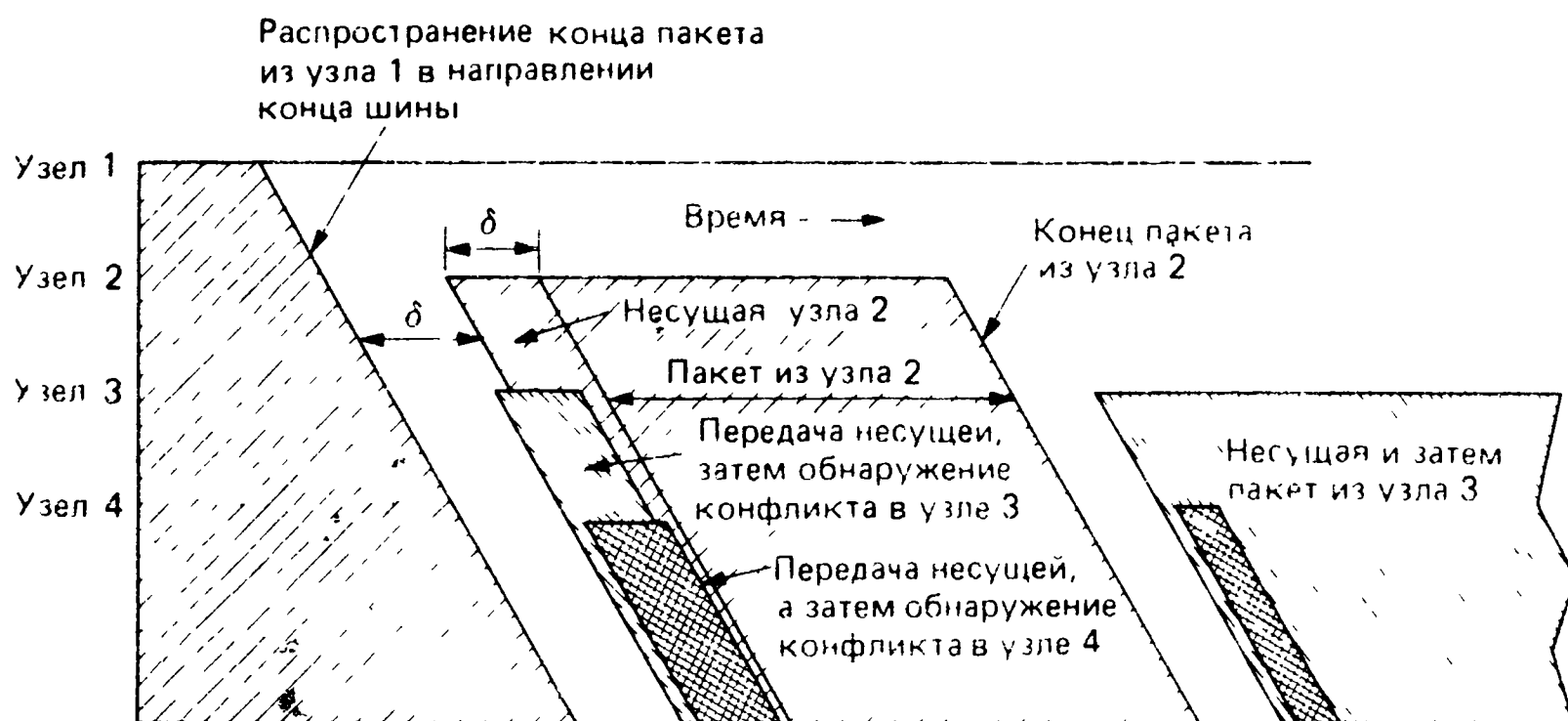


Рис. 4.25. Однонаправленная шина с неявными маркерами. Каждый узел с трафиком передает несущую, когда услышит, что шина пуста; затем прекращает передачу, если услышит несущую одного из узлов, расположенных выше по течению потока данных.

ным образом. Если узлы пронумерованы в порядке расположения на физической шине, то, как следует из (4.73), максимальная скорость передачи существенно уменьшается только тогда, когда величина β составляет существенную часть числа узлов m . Если, однако, после каждого пакета производится ожидание квитанции (подтверждения), то это преимущество в скорости теряется.

Один из наиболее привлекательных вариантов высокоскоростных локальных сетей состоит в использовании шин, в которых сигналы распространяются только в одном направлении. Оптические волокна, естественно, обладают этим свойством и кабельная техника хорошо разработана для передачи в одном направлении.

Имея однонаправленную шину, можно найти такое сочетание способов использования неявных маркеров и обнаружения конфликтов, что будет почти исключена задержка передачи маркера. Чтобы увидеть, как это сделать, рассмотрим рис. 4.25 и на время не будем принимать во внимание, как пакеты достигают своих мест назначения. Предположим, что узел в левом конце шины только что закончил передачу. Каждый последующий узел, который имеет пакет для передачи, начинает передавать несущую, как только он обнаруживает, что канал пуст.

Если δ — время, необходимое узлу для обнаружения пустого состояния и начала передачи несущей, то, как видно из рисунка, если два узла имеют пакеты для передачи после завершения передачи данного пакета, второй узел начнет передавать несущую спустя время δ после того, как конец предыдущего пакета пройдет этот узел. Несущая, переданная первым узлом, достигнет второго узла, по существу, в то время, когда второй узел

начнет передавать несущую. Так как второй узел может за дополнительное время δ обнаружить, передает ли первый узел несущую, то второй узел (и все последующие узлы) может закончить передачу несущей после этой задержки δ . Таким образом, если узел начинает передавать свой пакет после передачи несущей в течение времени δ , то он уверен, что ни один предыдущий узел не может находиться в состоянии передачи, а все последующие узлы прекратят передачу до того, как начнет поступать информация пакета.

Мы видим, что неявным маркером является конец передачи по линии. Время ν , в течение которого этот маркер перемещается от одного узла к следующему, равно времени распространения от одного узла к следующему. Мы считаем, что в длину пакета входят δ единиц времени, необходимых для обнаружения конца предыдущей передачи, и δ единиц времени для передачи несущей перед пакетом. В некотором смысле это является идеальной системой с резервированием на основе ВУ. Время выполнения резервирования включает только время распространения (неявный маркер имеет, по существу, нулевую длину), и пакеты передаются сразу после завершения резервирования.

Система EXPRESSNET

Приведенное рассмотрение не затрагивало вопросы о том, как принимать переданные пакеты и как возвращаться к началу шины, а также повторно запускать процесс после того, как все узлы получают возможность передачи. Сначала покажем, как эти проблемы решаются в системе EXPRESSNET [228], а затем кратко опишем несколько альтернативных подходов.

В системе EXPRESSNET используется однонаправленная шина, сложенная дважды, как показано на рис. 4.26. Пакеты передаются по первой части шины так, как объяснялось выше, а узлы считывают пакеты, когда они перемещаются по третьей части шины. Когда на третьей части шины обнаруживается интервал молчания длиннее δ , становится ясно, что все узлы получили возможность передачи и наступило время запуска нового цикла. Таким образом, все узлы, имеющие пакеты для передачи, снова начинают передавать несущую. Однако из-за двойного перегиба шины узлы на левом конце шины обнаружат пустое состояние раньше тех, которые находятся на правом конце, и сигналы несущих опять наложатся, начиная следующий цикл, такой, как показан на рис. 4.25.

Описание этой системы еще не завершено, так как имеется трудность в случае, если ни один узел не имеет пакета для передачи в цикле. Эта трудность устраняется тем, что все узлы как с пакетами, так и без них передают несущую в течение времени δ ,

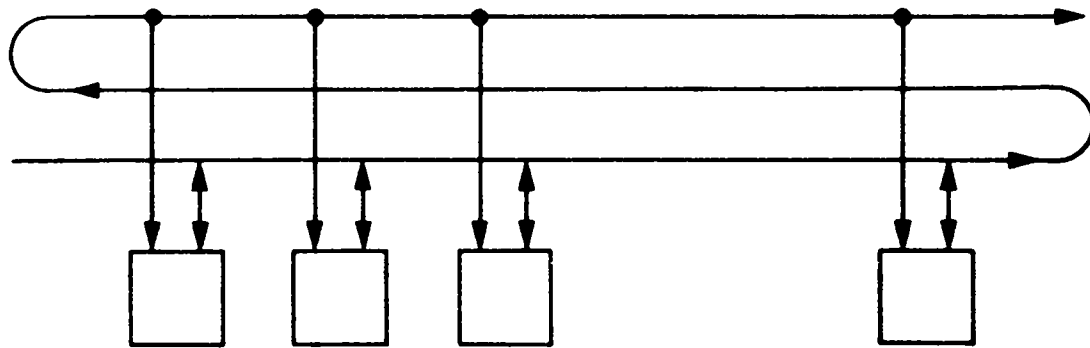


Рис. 4.26. Конфигурация шины сети EXPRESSNET. Каждый узел передает в нижнюю часть шины и слушает как нижнюю, так и верхнюю части. Конфликт разрешается в нижней части, причем приоритет отдается самому левому узлу. Прием и обнаружение конца цикла осуществляются в верхней части.

когда они услышат появление пустого интервала (длина которого больше δ) на третьей части шины. Все узлы с пакетами должны в таком случае удвоить время передачи несущей. Этот дополнительный период длины δ необходим для обнаружения пустого состояния на первой части шины, если ни один из предыдущих узлов не имеет пакета для передачи. Дополнительная передача несущей всеми узлами позволяет сохранять синхронизацию системы во время пустых периодов.

Поскольку время распространения от момента, когда узел начинает передачу, до момента, когда сигнал приходит в соответствующую точку третьей части шины, равно 2β и поскольку передача несущей и ее обнаружение занимают время 2δ , мы видим, что средний интервал резервирования равен

$$\sigma = 2(\beta + \delta)/m. \quad (4.74)$$

Равенства (4.70) и (4.71) и в этом случае определяют среднюю задержку для многопакетного и однопакетного резервирования соответственно.

Существует много возможных вариантов того, как использовать эту однонаправленную шину. Вместо конфигурации с двумя перегибами можно использовать сложенную вдвое шину, считывая пакеты во время их обратного движения по шине и затем используя специальный узел на левом конце шины для запуска новых циклов. Можно также использовать две шины, по одной на каждое направление. В таком случае узел может посылать трафик узлам, находящимся справа от него, по шине с правым направлением, а узлам, находящимся слева от него, по шине с левым направлением. Специальные узлы на каждом конце двух шин могут затем обмениваться информацией для запуска новых циклов в каждом направлении. В других вариантах применяется отдельный управляющий провод или отдельная частотная полоса вместо неявных резервирующих маркеров. В [71] проводится сопоставление большого числа этих возможных подходов.

Сеть HOMENETS

Предыдущие методы построения высокоскоростных локальных сетей основывались на использовании однонаправленной шины. Вместе с тем системы CATV (кабельное телевидение) обычно имеют структуру дерева; при этом сигналы, переданные в корне дерева, распространяются широковещательным образом по всему дереву. При использовании таких сетей для передачи данных обычно выделяется полоса частот для передачи от концевых вершин дерева (листьев) к его корню. Когда пакеты от различных листьев направляются внутрь к корню, возникает обычная задача разрешения конфликтов, но здесь нельзя применить метод с использованием неявных резервирующих маркеров с учетом положения на шине.

Сеть HOMENETS [168] представляет интересный подход к преодолению трудностей, связанных с большой величиной β в такой сети. Идея состоит в том, что сеть разбивается на подсети, называемые домовыми (homenets). Каждая домовая сеть формирует свое собственное поддерево на общем дереве сети, как показано на рис. 4.27, и каждая домовая сеть имеет свои собственные полосы частот, одну для передачи, направленной внутрь к корню, а другую для передачи наружу. Эта стратегия решает проблему большого β в двух отношениях. Во-первых, время распространения в пределах домовой сети намного меньше, чем во всей сети, и, во-вторых, использование ограниченной полосы пропускания уменьшает скорость передачи данных в домовой сети. Благодаря такому двукратному уменьшению величины β становится разумным использование МДПН/ОК в домовой сети.

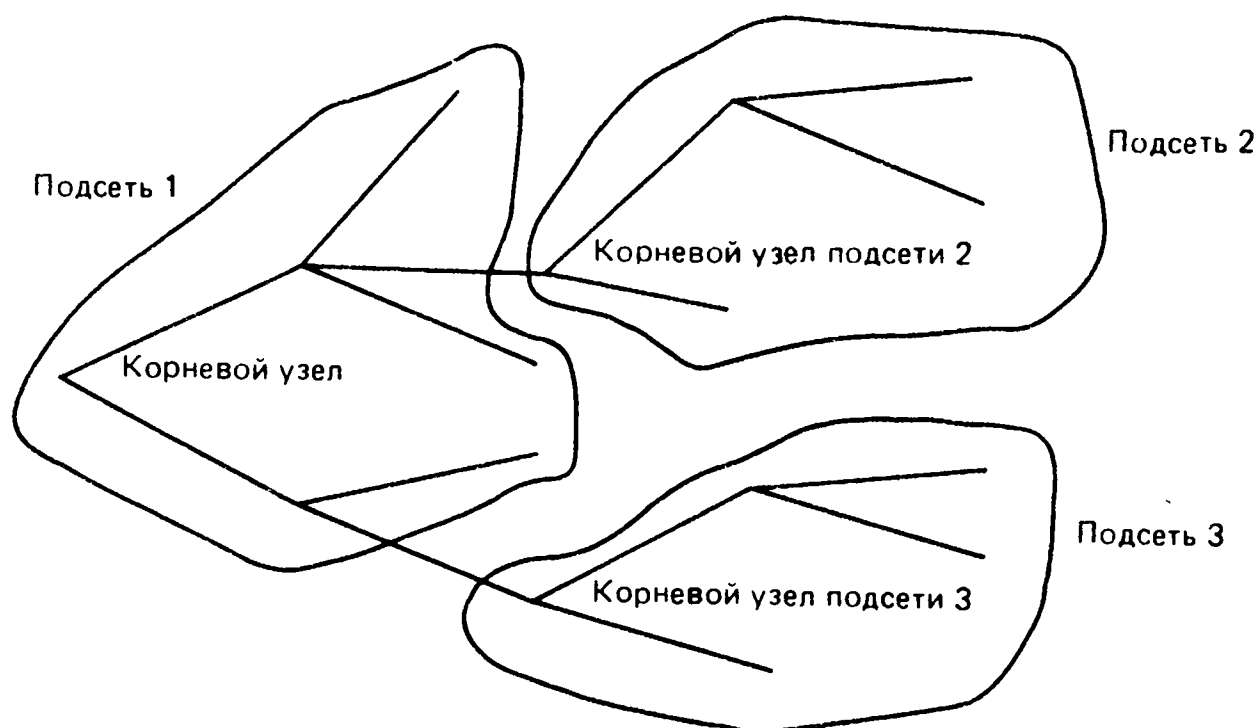


Рис. 4.27. Сеть CATV (кабельное телевидение) разделена на подсети в соответствии со стратегией домовых сетей. Каждая подсеть имеет одну полосу частот для передачи внутрь и одну для передачи наружу и использует МДПН/ОК для разрешения конфликтов в подсети.

Пока остаются без ответа два вопроса. Во-первых, так как узлы передают пакеты внутрь, в направлении корня, то как слышат конфликты другие внешние узлы этой же домовой сети? Во-вторых, как принимаются пакеты узлами, находящимися вне домовой сети? Решение первого вопроса состоит в том, что корневой узел данной домовой сети принимает сигнал во входной полосе частот домовой сети и передает его в направлении корня всей сети, преобразует его частоту и затем передает в широковещательном режиме в выходной полосе частот домовой сети. Это позволяет узлам обнаруживать конфликты за время распространения по домовой сети. Решение второго вопроса состоит в том, что сигналы всех входных частотных полос направляются к корню дерева, где они преобразуются по частоте и передаются в широковещательном режиме в выходных частотных полосах по всей сети. Корневой узел каждой домовой сети отфильтровывает затем сигнал на своей выходной частоте (поскольку этот сигнал является просто задержанной копией того, что он уже передал наружу) и направляет сигналы других полос в свою домовую сеть.

Дополнительным преимуществом этой стратегии является то, что узлам нужно только передавать и принимать в этих ограниченных частотных полосах. Когда между двумя узлами устанавливается сеанс, приемник каждого узла должен знать выходную полосу частот домовой сети отправителя, а затем он просто принимает нужные пакеты, передаваемые сигналом этой полосы.

4.5.6. Обобщенный опрос и алгоритмы разбиения

Последний подраздел был посвящен системам с множественным доступом и с большим временем распространения. Здесь мы рассмотрим противоположный случай, когда $\beta \ll 1$. Примером являются системы с очень короткими шинами, такие как в задней части мультимикропроцессорной системы. Другой пример представляют системы с опросом, использующие многоточечные телефонные линии; скорости передачи данных в них малы, вследствие чего время передачи пакетов много больше, чем время распространения.

В этих случаях, как видно из (4.72) и (4.73), время выполнения резервирования узлом u увеличивается линейно с ростом времени обнаружения δ и времени передачи маркера (явного или неявного). Из выражений для средней задержки (4.70) и (4.71) следует, что задержка равна $tu/2$ при нулевой нагрузке; более того, если tu велико по сравнению с временем передачи пакета, то задержка увеличивается по крайней мере линейно с ростом tu при всех устойчивых нагрузках. Таким образом, мы хотим сократить tu , т. е. время, затрачиваемое на резервирование в течение цикла.

В целях упрощения будем моделировать канал с множественным доступом (по крайней мере для резервирования) как двоич-

ный канал «или» с синхронизацией по битам, т. е. в каждом периоде передачи бита выходной сигнал канала равен 1, если входной сигнал для одного или более узлов равен 1; в противном случае выходной сигнал равен 0. Эта модель применима, если для резервирования используется отдельный управляющий провод; она также применима, если в узлах используются короткие импульсы несущей для запросов на резервирование (т. е. наличие несущей может быть обнаружено, а число узлов, передающих несущую, нет).

Эта модель в значительной мере подобна исходной модели синхронного множественного доступа, описанной в разд. 4.2. Отличие состоит в том, что длина окна сократилась до периода одного бита; другое отличие — вместо обратной связи с сигналами 0,1, e имеем теперь обратную связь с нулевым или положительным значениями. Мы рассматриваем окна как окна резервирования и считаем (как в приведенных выше примерах), что пакеты могут передаваться со скоростями, большими чем один бит за окно резервирования. Интерес представляет вопрос о том, как много окон резервирования необходимо для выполнения резервирования.

Простейшей стратегией в этой модели является выполнение резервирований посредством ВУ в окнах резервирования. При сильной нагрузке большинство узлов имеет пакет для передачи к моменту наступления каждого своего окна и почти каждое окно резервирования используется для выполнения резервирования. В пределе, когда нагрузка стремится к нулю, поступающий пакет должен ждать в среднем $m/2$ окон резервирования; это та ситуация, которой мы хотим избежать.

При слабой нагрузке более совершенной стратегией является логарифмический поиск узла с пакетом [113, 183]. Предположим, что узлы имеют номера от 0 до $m - 1$, и пусть n_1, \dots, n_k обозначает двоичное представление номера некоторого заданного узла, где $k = \lceil \log_2 m \rceil$.

Алгоритм работает последовательными периодами разрешения конфликтов (ПРК) при отыскании узла с наименьшим номером, который имеет пакет для передачи. В первом окне ПРК все активные узлы (т. е. все узлы с ожидающими передачи пакетами) передают 1, а все остальные узлы передают 0 (т. е. молчат). Если выходной сигнал канала, обозначаемый через y_0 , равен 0, то ПРК заканчивается и начинается новый ПРК в следующем окне для выполнения резервирований для всех пакетов, которые могли поступить за это время. Вместе с тем если $y_0 = 1$, то один узел или более имеют пакеты и начинается логарифмический поиск.

Если $y_0 = 1$, то все активные узлы с $n_1 = 0$ передают 1 в следующем окне. Если y_1 — выходной сигнал канала в этом окне — равен 1, то это означает, что двоичное представление наименьшего

номера узла начинается с 0; в противном случае оно начинается с 1. В первом случае все активные станции с $n_1 = 1$ становятся неактивными и ожидают конца ПРК, когда они становятся опять активными. В последнем случае все активные узлы имеют $n_1 = 1$ и остаются активными.

Общее правило на $(i + 1)$ -м шаге ПРК, $1 \leq i \leq k$ (предполагается, что $y_0 = 1$), состоит в том, что все активные узлы с $n_i = 0$ передают 1; если выходной сигнал канала y_i в конце окна равен 1, то все узлы с $n_i = 1$ становятся неактивными. На рис. 4.28 показан пример работы такой стратегии. В конце $(k + 1)$ -го окна активен только узел с наименьшим номером и двоичное представление номера этого узла совпадает с побитовым дополнением к y_1, \dots, y_k .

Чтобы сохранить справедливость и обслуживать узлы в порядке кругового опроса, узлы необходимо перенумеровывать после выполнения каждого резервирования. Если резервирование выполнено для узла n , например, то из текущего номера каждого узла вычитается $n + 1$ по модулю m . Таким образом, номер узла на единицу меньше расстояния кругового обхода до последнего переданного узла; понятно, что это правило можно модифицировать, вводя приоритеты любым желаемым способом.

Предполагая, что пакет передается сразу после выполнения резервирования и что следующий ПРК начинается после этой передачи, легко найти среднюю задержку этого алгоритма. Каждый ПРК, который начинается в момент времени, когда система занята, длится $k + 1$ окон резервирования. Таким образом, мы можем считать, что время передачи пакета просто удлинилось на эти $k + 1$ окон. Если система пуста в начале ПРК, то ПРК длится одно окно резервирования и в этом случае можно считать, что обслуживающий прибор простаивает в течение одного окна резервирования. Равенство (3.55) из гл. 3 определяет задержку этой системы.

Сравнивая эту стратегию резервирования на основе логарифмического поиска с резервированием на основе ВУ, видим, что логарифмический поиск сокращает задержку с $m/2$ до $k + 1$ окон резервирования на один пакет при слабой нагрузке, но увеличивает задержку с 1 до $k + 1$ окон резервирования на пакет при сильной нагрузке. Возникает очевидный вопрос о том, при каком сочетании этих стратегий суммируются их преимущества. Ответ следует из рассмотрения этой задачи как задачи кодирования источника, что в большой степени аналогично вопросу обнаружения границ кадра из гл. 2. Использование ВУ соответствует использованию унарного кода при кодировании длин кадров. Логарифмический поиск, применяемый здесь, соответствует использованию обычного двоичного представления. В гл. 2 отмечалось преимущество перехода к комбинированному унарно-двоичному пред-

Узлы								Выходной сигнал	
000	001	010	011	100	101	110	111	y	
0	0	1	1	0	0	1	0	1	Окно 1
0	0	1	1	0	0	0	0	1	Окно 2
0	0	0	0	0	0	0	0	0	Окно 3
0	0	1	0	0	0	0	0	1	Окно 4

Рис. 4.28. Логарифмический поиск активного узла с наименьшим номером. Узлы 010, 011 и 110 имеют пакеты. В конце окна 2 узел 110 становится неактивным; в конце окна 4 узел 011 становится неактивным. Дополнения выходных сигналов канала в окнах 2—4 составляют 010.

ставлению. В данном контексте это означает, что в каждом ПРК должно проверяться ограниченное число узлов, например 2^j первых узлов. Если выходной сигнал равен 1, то узел с наименьшим номером в этом множестве находится, как и ранее, за j дополнительных окон. Если выходной сигнал 0, то 2^j вычитается из номера каждого узла и начинается следующий ПРК. В задаче 4.30 нужно найти среднее число окон резервирования на пакет и оптимальное значение j при нескольких различных предположениях.

Обратим внимание на подобие этих алгоритмов алгоритмам разбиения из разд. 4.3. В их основе лежат одни и те же идеи и единственное отличие состоит в том, что успех невозможно здесь отличить от конфликта; поэтому, если даже только один узел активного подмножества содержит пакет, подмножество должно разбиваться до тех пор, пока останется только один узел.

4.6. Пакетные радиосети

В подразд. 4.1.4 кратко представлены пакетные радиосети как сети с множественным доступом, в которых не все узлы могут слышать передачи всех остальных узлов. Эта особенность характерна как для радиосвязи в пределах прямой видимости в УВЧ-диапазоне (300—3000 МГц), так и для связи в ВЧ-диапазоне (3—30 МГц), не ограниченной пределами прямой видимости. Нас интересуют здесь не физические характеристики ширококвотельной радиосреды, а влияние частичной связности на методы множественного доступа.

Топологию радиосети можно описать графом; пример такого графа приведен на рис. 4.29. Граф $G = (N, L)$ содержит множество узлов N и множество линий L . Каждая линия из L соответствует упорядоченной паре узлов (i, j) и указывает на то, что передачи узла i могут прослушиваться узлом j . В некоторых ситуациях узел j способен слышать i , но i не может слышать j . В таких случаях $(i, j) \in L$, но $(j, i) \notin L$. Такой асимметрии нет на рис. 4.29,

где каждое ребро означает две линии, по одной на каждое направление.

Наше предположение о связи в такой среде с множественным доступом состоит в том, что если узел i передает пакет, то этот пакет будет безошибочно принят узлом j тогда и только тогда, когда

1) существует линия связи от i к j , т. е. $(i, j) \in L$;

2) никакой узел k , для которого $(k, j) \in L$, не передает в то время, когда передает i ;

3) узел j не передает во время передачи i .

Таким образом, если в сети на рис. 4.29 узлы 1 и 3 передают одновременно, то узел 2 безошибочно примет пакет из узла 1, а узел 4 безошибочно примет пакет из узла 3. Вместе с тем если узлы 2 и 3 передают одновременно, то каждый из узлов 1 и 4 обнаружит конфликт, а узел 5 безошибочно примет пакет из узла 3.

Этот пример показывает, что наличие большого числа линий в графе не обязательно является благоприятным фактором. Большое число линий увеличивает число пар узлов, которые могут связываться непосредственно, но также увеличивает вероятность конфликтов. Дальнейшее исследование этого обменного соотношения проводится в подразд. 4.6.3.

Интересный вопрос, который можно сейчас поставить, состоит в том, какой величины трафик может быть передан в такой сети. Определим *свободное от конфликтов множество* как множество линий, которые могут передавать пакеты одновременно, не вызывая конфликты на приемных концах линий. Например, $(1, 2)$, $(3, 4)$ и $(2, 1)$, $(5, 3)$ являются свободными от конфликтов множествами; кроме того, пустое множество и любое множество, содержащее единственную линию, являются множествами, свободными от конфликтов. Удобно пронумеровать линии некоторым произвольным образом и представить каждое свободное от конфликтов множество как вектор, составленный из нулей и единиц,

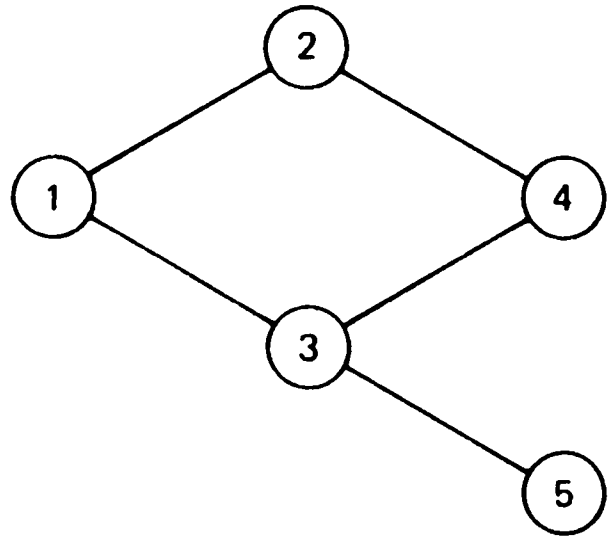


Рис. 4.29. Пакетная радиосеть; два узла, находящиеся на концах ребра, могут слышать передачи друг друга.

(1,2)	(2,1)	(1,3)	(3,1)	(2,4)	(4,2)	(3,4)	(4,3)	(3,5)	(5,3)
1	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	1	0	0
0	1	0	0	0	0	0	0	0	1

который называется *свободным от конфликтов вектором* (СКВ). Компонента СКВ с номером l равна 1, если только l -я линия принадлежит соответствующему свободному от конфликтов множеству. Ниже перечислены некоторые СКВ для графа на рис. 4.29.

4.6.1. ВУ в пакетных радиосетях

Простой способ использования пакетной радиосети состоит в том, что выбирается определенная совокупность свободных от конфликтов множеств, которые используются циклически на основе ВУ, т. е. в i -м окне цикла ВУ все линии из i -го свободного от конфликтов множества могут передавать пакеты. При такой основанной на ВУ стратегии конфликты не возникают и доля времени, в течение которой определенная линия может передавать пакеты, равна доле тех свободных от конфликтов множеств совокупности, которые содержат эту линию.

Более формально, если x_1, \dots, x_J обозначают СКВ, соответствующие совокупности J свободных от конфликтов множеств, то вектор $f = (\sum_j x_j)/J$ определяет долю времени, в течение которой может использоваться каждая линия. В несколько более общем случае кадр ВУ может быть расширен и каждое свободное от конфликтов множество может использоваться в кадре некоторое произвольное число раз. Если α_j обозначает долю окон кадра, использующих j -е свободное от конфликтов множество, то

$$f = \sum_j \alpha_j x_j \quad (4.75)$$

определяет долевое использование каждой линии. Вектор, имеющий представление $\sum_j \alpha_j x_j$, в котором $\sum_j \alpha_j = 1$ и $\alpha_j \geq 0$ для $0 \leq j \leq J$, называется выпуклой комбинацией векторов x_1, \dots, x_J . Только что мы увидели, что для любой выпуклой комбинации СКВ можно получить сколь угодно точное приближение вектором долевого использования линий при ВУ.

Предположим, что вместо ВУ, рассмотренного выше, мы используем в сети некоторый метод с разрешением конфликтов. В любой заданный момент времени вектор линий, которые успешно передают пакеты, является СКВ. Усредняя по времени этот вектор успешных передач по линиям, получаем вектор, у которого l -я компонента является долей времени, в течение которой l -я линия передает пакеты успешно. Он также является выпуклой комбинацией СКВ. Таким образом, мы видим, что любая скорость передачи по линиям, обеспечиваемая методами с разрешением конфликтов, достигается также при ВУ.

Недостатком ВУ является то, что в слабо нагруженной сети задержки больше, чем это необходимо. Он не так серьезен, как

в случае, когда все узлы соединены с общим приемником, поскольку если все узлы имеют малое число входящих линий, то много линий могут передавать одновременно и ожидание окна из цикла ВУ уменьшается.

Более серьезная проблема, свойственная ВУ, заключается в том, что в пакетной радиосети узлы обычно мобильны и поэтому топология сети постоянно меняется. Это означает, что свободные от конфликтов множества меняются, что требует частого обновления циклического порядка ВУ. Это трудная проблема, так как даже для статичной сети задача определения того, является ли потенциальный вектор использования линий выпуклой комбинацией СКВ, относится к классу сложных задач, известных как *NP-полные* [9]. Введение в теорию *NP*-полных задач имеется в работе [188]; для наших целей достаточно считать, что это просто означает, что в наихудшем случае вычислительные затраты на решение задачи увеличиваются очень быстро с ростом числа линий в сети. Основная причина этой трудности состоит в том, что число различных свободных от конфликтов множеств с ростом числа линий в сети увеличивается экспоненциально.

Частотное уплотнение можно использовать в пакетных радиосетях во многом так же, как ВУ. Все линии свободного от конфликтов множества могут использовать одновременно одну и ту же полосу частот, так что в принципе линии могут передавать такой же трафик, что и при ВУ. Этот подход используется в сотовых радиосетях для мобильной речевой связи. Площадь, покрываемая такой сетью, разбивается на большое число локальных площадей, называемых ячейками; при этом каждая ячейка имеет множество частотных полос, предназначенных для использования в пределах ячейки. Множество частотных полос, используемых одной ячейкой, может использоваться также другими ячейками, которые достаточно удалены друг от друга, чтобы избежать интерференции. Это обеспечивает возможность простого выбора свободных от конфликтов множеств. Принцип разбиения на ячейки можно сочетать с использованием ВУ.

Выше не рассматривался вопрос о том, как выбирать маршрут для пакетов, движущихся от источника до пункта назначения. Когда используется представленная структура на основе ВУ или ЧУ, каждая линия сети может передавать пакеты с определенной скоростью, и взаимодействие линий по совместному использованию ресурсов устраняется. Таким образом, задача маршрутизации, по существу, совпадает с аналогичной задачей для традиционных сетей с назначенными линиями между узлами; эта задача исследуется в гл. 5. Однако мы видим, что, когда используются стратегии с разрешением конфликтов, маршрутизация является значительно более сложной задачей по сравнению со случаем традиционных сетей.

4.6.2. Разрешение конфликтов в пакетных радиосетях

Разрешение конфликтов в пакетных радиосетях использовать несколько сложнее, чем в изученной ранее системе с единственным приемником. Первое усложнение связано с получением информации обратной связи. Предположим, например, что линии (2, 4) и (3, 5) графа на рис. 4.29 передают пакеты в данном окне. Тогда узел 4 воспримет это как конфликт, а узел 5 примет пакет безошибочно. Если узлы 4 и 5 пошлют сигналы обратной связи, то в узле 3 возникнет конфликт этих сигналов. Вторая проблема состоит в том, что если узел обнаруживает конфликт, то он не знает, сколько пакетов было ему адресовано. По этим двум причинам нельзя предполагать существование совершенной обратной связи с сигналами 0, 1, e , которая рассматривалась ранее. Отсюда следует, что алгоритмы разбиения, описанные в разд. 4.3, невозможно использовать, а методы стабилизации из подразд. 4.2.3 нужно основательно пересмотреть.

К счастью, синхронная и несинхронная системы АЛОНА по-прежнему применимы и некоторые идеи прослушивания несущей и резервирования могут также использоваться. Начнем с анализа того, как синхронная АЛОНА может действовать в такой окружающей обстановке. Когда узел, не имеющий задолженности, получает пакет для передачи (новый пакет, вошедший в сеть, или транзитный пакет, который необходимо отправить другому узлу), он посылает пакет в следующем окне. Если квитанция (подтверждение) правильного приема не поступает в течение некоторого периода времени (тайм-аута), то узел будет иметь задолженность, и пакет передается повторно после случайной задержки. Наконец, узел с задолженностью не будет иметь задолженность, когда все его пакеты успешно переданы и получили положительные квитанции.

Существует ряд способов передачи квитанции передающему узлу. Простейший способ состоит в том, что если узел i посылает узлу j пакет, который должен быть передан некоторому другому узлу k , то i слушает передачи j , и если он услышит передачу для узла k , то это служит квитанцией передачи (i, j) . Этот способ является несовершенным в двух отношениях. Во-первых, требуется некоторый другой метод квитирования пакетов, для которых узел j является конечным. Во-вторых, предположим, что узел j успешно ретранслировал пакет узлу k , но i не удалось прослушать передачу из-за конфликта. В этом случае производится ненужная повторная передача от i к j и, кроме того, требуется некоторый способ квитирования для j , так как j уже отправил пакет для k . Другой подход, который можно использовать совместно с описанными выше неявными квитанциями, состоит в том, что в каждом узле явные квитанции для нескольких пос-

ледних пакетов, которые уже приняты, помещаются в каждый отправляемый пакет. Этот подход требует от узла передачи фиктивного пакета, содержащего информацию квитирования, если узел не имеет данных для передачи в течение некоторого периода. Третий подход, который, по-видимому, несколько уступает предыдущему, состоит в том, что в конце каждого окна выделяется время для передачи явных квитанций пакетов, переданных в этом окне.

Проанализируем работу сильно нагруженной сети, использующей синхронную АЛОНА. В частности, предположим, что все узлы всегда имеют задолженность и пакеты для передачи по всем выходящим линиям. Можно предположить, что емкость буферов в узлах для хранения задолженных пакетов не ограничена, но в данный момент нас не интересует вопрос о задержке. Это предположение о постоянной задолженности сильно отличается от наших предположений в разд. 4.2, но причины этого будут рассмотрены в дальнейшем. Для любых узлов i и j обозначим через q_{ij} вероятность того, что узел i передает пакет узлу j в любом заданном окне, и пусть Q_i есть вероятность того, что узел i передает какому-либо узлу. Таким образом,

$$Q_i = \sum_j q_{ij}. \quad (4.76)$$

Чтобы упростить обозначения, будем считать, что q_{ij} равна нулю, если (i, j) не принадлежит множеству линий L . Пусть p_{ij} обозначает вероятность того, что передача по (i, j) успешна. При условии сильной нагрузки каждый узел передает или не передает в окне независимо от всех остальных узлов. Поскольку p_{ij} — вероятность того, что ни один узел из окружения узла j , включая узел j , не передает, имеем

$$p_{ij} = (1 - Q_j) \prod_{\substack{k: (k, j) \in L \\ k \neq i}} (1 - Q_k). \quad (4.77)$$

Наконец, интенсивность f_{ij} успешных передач пакетов в течение окна, т. е. скорость передачи, для линии (i, j) равна

$$f_{ij} = q_{ij} p_{ij}. \quad (4.78)$$

Равенства (4.76)—(4.78) дают выражение для скорости передачи линии через интенсивности попыток q_{ij} при условии сильной нагрузки. Однако больший интерес представляют интенсивности попыток q_{ij} , которые приводят к желаемому множеству скоростей передач (если это множество скоростей передач достижимо).

Эту задачу можно решить с помощью итеративного подхода. Для простоты обозначений введем вектор q , компонентами которого являются интенсивности попыток q_{ij} , векторы p и f , компонентами которых являются p_{ij} и f_{ij} соответственно, и вектор Q

с компонентами Q_i . При заданном желаемом векторе скоростей f начинаем с исходного вектора q^0 , составленного из нулей. Затем используем (4.76) и (4.77) для вычисления Q^0 и p^0 (таким образом, Q^0 — вектор, составленный из нулей, а p^0 — вектор, составленный из единиц). Затем используется (4.78) для получения следующего итеративного значения q , т. е. компоненты q^1 определяются соотношением

$$q_{ij}^1 = f_{ij}/p_{ij}^0. \quad (4.79)$$

На каждом следующем шаге итерации Q^n вычисляется по формуле (4.76) с использованием q^n , p^n — по формуле (4.77) с использованием Q^n и затем q^{n+1} — по формуле (4.78) с использованием p^n . Заметим, что $q^1 \geq q^0$ (т. е. каждая компонента q^1 не меньше соответствующей компоненты q^0). Таким образом, $Q^1 \geq Q^0$, а $p^1 \leq p^0$. Из (4.78) следует, что $q^2 \geq q^1$. Продолжая рассуждать таким образом, убеждаемся, что пока ни одна компонента Q не превышает 1, q не уменьшается на каждом последующем шаге итерации, а p не увеличивается. Отсюда следует, что или некоторая компонента Q должна превысить 1 на некотором шаге итерации, или q должна стремиться к пределу, скажем q^* , и в пределе равенства (4.76) и (4.78) одновременно удовлетворяются при подстановке результирующих Q^* и p^* .

Теперь покажем, что если (4.76)—(4.78) имеют некоторое другое решение, например q' , Q' , p' (удовлетворяющее, конечно, неравенствам $q' \geq 0$, $Q' \leq 1$), то $q' \geq q^*$, $Q' \geq Q^*$, а $p' \leq p^*$. Чтобы понять это, заметим, что $q^0 \leq q'$, $Q^0 \leq Q'$, $p^0 \geq p'$. Из формулы (4.78) в таком случае следует, что $q^1 \leq q'$. Продолжая рассуждать таким образом на следующих шагах итерации, получаем $q^n \leq q'$, $Q^n \leq Q'$ и $p^n \geq p'$ для любого n и убеждаемся в справедливости утверждения, переходя к пределу. Эти рассуждения показывают также, что если некоторая компонента Q^n превышает 1 при некотором n , то уравнения (4.76)—(4.78) не имеют решения, т. е. данный вектор f недостижим.

Далее предположим, что мы знаем входные потоки в сеть и маршруты, по которым передаются пакеты сеансов, так что в принципе можно определить стационарные скорости передачи f'_{ij} , с которыми линии должны обрабатывать трафик. Нам хотелось бы выбрать скорости передачи всех линий при условии сильной нагрузки так, чтобы они превышали эти стационарные скорости передачи и задолженности не росли неограниченно. Один из подходов сводится к отысканию наибольшего числа $\beta > 1$, для которого $f = \beta f'$ достижим при условии сильной нагрузки. При заданном наибольшем f и соответствующих интенсивностях попыток q можно обслужить очередь задолженных пакетов.

Здесь существует одно затруднение, состоящее в том, что если некоторые узлы имеют задолженность, а другие нет, то узлы

без задолженности не выбирают моменты передачи независимо. Таким образом, можно установить, что в определенных ситуациях некоторые узлы с задолженностью действуют намного хуже, когда другие узлы не имеют задолженностей, по сравнению с тем, когда все узлы имеют задолженность. В задаче 4.23 приводится пример такого явления. Один способ избежать это затруднение состоит в том, что новые пакеты в узле сразу присоединяются к задолженным вместо того, чтобы передаваться в следующем окне. Это, конечно, увеличивает задержку при слабой нагрузке. Другой подход — это жить в опасности и надеяться на лучшее. Во всяком случае, в пакетных радиосетях приходится действовать в некоторой степени таким образом, поскольку при изменяющейся топологии невозможно точно контролировать интенсивности попыток.

Одна из причин того, что внимание концентрируется на случае сильной нагрузки, состоит в том, что число входных линий каждого узла обычно мало в пакетных радиосетях и, таким образом, интенсивности попыток могут быть умеренно высокими даже при условии сильной нагрузки. Вместе с тем в случае единственного приемника число узлов, как правило, намного больше и, таким образом, интенсивности попыток, соответствующие сильной нагрузке, приводят, как правило, к большим задержкам. Другая причина заключается в том, что стабилизация в данном случае представляет намного более трудную задачу, чем в случае единственного приемника; узел не может обеспечить передачу собственных пакетов, регулируя свои интенсивности попыток, поскольку другие узлы могут вызывать переполнение; при этом никакого переполнения они не испытывают (см. задачу 4.32).

4.6.3. Радиусы передачи в пакетной радиосети

В предыдущих подразделах мы считали, что множество линий в пакетной радиосети является заданным. Однако понятно, что если узел увеличивает мощность своего передатчика, то его передача будет слышна большему множеству узлов. Следующие качественные рассуждения показывают, что желательно удерживать мощность на относительно низком уровне, чтобы каждый узел имел довольно малое множество входящих и выходящих линий. Для простоты предположим, что у нас есть симметричная сеть, в которой каждый узел имеет n входящих и n выходящих линий. Далее допустим, что каждая линия имеет одинаковые требования по передаче трафика. Нетрудно убедиться в том, что соотношения (4.76)—(4.78) удовлетворяются при одинаковой интенсивности попыток q в каждой линии. Каждая Q_i равна тогда nq , каждая p_{ij} определяется выражением $(1 - nq)^n$ и, наконец,

$$\bar{f} = q (1 - nq)^n. \quad (4.80)$$

Легко проверить, что f максимальна при $q = 1/[n(n+1)]$, максимальная величина f приближенно равна $1/(en^2)$. Каждый узел тогда передает успешно пакеты со скоростью $1/en$. Если в сети имеется m узлов и среднее число линий, составляющих путь от источника до пункта назначения, равно J , то скорость, с которой сеть может доставлять пакеты, равна $m/(Jen)$ пакетов за окно.

Теперь посмотрим, к чему приводит изменение радиуса передачи R , определяющего зону, в которой узел может быть услышан. Число узлов в зоне радиуса R данного узла будет меняться примерно как R^2 , так что скорость, с которой отдельный узел успешно передает пакеты, будет уменьшаться как $1/R^2$. Вместе с тем при увеличении R маршрутизация будет, по-видимому, изменяться так, чтобы передавать пакеты как можно дальше в направлении пункта назначения по каждой линии пути. Таким образом, мы ожидаем, что число линий, составляющих путь, уменьшается как $1/R$. Таким образом, если J пропорционально $1/R$, а n пропорционально R^2 , то скорость, с которой сеть может доставлять пакеты, пропорциональна $1/R$; это показывает, что R должен быть очень малым.

В приведенном очень грубом анализе не учитываются два важных фактора. Во-первых, когда R и n велики, пакет может приближаться к своему пункту назначения почти на расстояние R при передаче по каждой линии хорошо выбранного пути, так что J в основном пропорционально $1/R$ в этой области. Однако, когда R становится меньше, пути становятся очень извилистыми и, таким образом, J уменьшается с ростом R много быстрее, чем $1/R$ при малых R . Во-вторых, когда R очень мал, связность у сети не очень хорошая и некоторым линиям приходится, возможно, передавать очень большой трафик. Это приводит к заключению, что R должен быть малым, но не очень малым, так чтобы n было значительно больше чем 1. Такаги и Клейнрок [225] провели намного более точный анализ (хотя некоторые предположения вызывают вопросы) и пришли к заключению, что радиус должен быть таким, чтобы n равнялось примерно 8.

4.6.4. Прослушивание несущей и сигналы занятости

В разд. 4.4 было показано, что прослушивание несущей приводит к значительному улучшению характеристик передачи по сравнению с синхронной АЛОНА в ситуации, когда каждый из узлов может прослушивать передачи всех других узлов и время распространения мало. В радиосетях с передачей в пределах прямой видимости время распространения обычно мало по сравнению с временем передачи пакетов, так что есть основания исследовать, насколько эффективно здесь прослушивание несущей. К сожалению, если узел i передает узлу j и узел k хочет передать j ,

то нет уверенности в том, что k может услышать i . Узлы i и k может разделять некоторое препятствие или каждый из них может находиться вне зоны слышимости другого. Таким образом, прослушивание несущей будет предотвращать возникновение некоторых конфликтов, но не может предотвратить другие конфликты. Прослушивание несущей затрудняется еще тем, что отсутствует постоянная синхронизация по окнам, и, таким образом, прослушивание несущей теряет некоторые из преимуществ, которые имеет синхронная АЛОНА по отношению к простой АЛОНА. Наконец, радиопередача подвержена воздействиям замирания и флуктуирующего шума, так что существование другого передающего узла, даже находящегося в зоне прослушивания, трудно обнаружить за короткое время. По этим причинам прослушивание несущей не очень эффективно в пакетных радиосетях.

Введение сигнала занятости [217, 231] позволяет улучшить характеристики пакетной радиосети с прослушиванием несущей. Всякий раз, когда некоторый узел обнаруживает передаваемый пакет, он начинает посылать сигнал, называемый сигналом занятости, в выделенной полосе частот. Таким образом, когда узел i начинает передавать пакет узлу j , последний (наряду со всеми другими узлами, которые могут слышать i) начинает посылать сигнал занятости. Все узлы, которые могут слышать j , не будут, таким образом, передавать; итак, если множество узлов, которые могут слышать передачу j , совпадает с множеством узлов, которое может слышать j , узел j не будет испытывать конфликтов.

С использованием сигнала занятости связана проблема, состоящая в том, что, когда узел i начинает передавать пакет, все узлы, находящиеся в зоне i , начинают посылать сигналы занятости и, таким образом, каждому узлу, находящемуся вблизи любого узла из зоны i , запрещается передавать. Применяя очень грубый метод анализа из последнего подраздела и предполагая радиус передачи равным R , мы видим, что, когда узел начинает передавать, большинству узлов из зоны радиуса $2R$ запрещается передавать. Обычно число таких узлов примерно в четыре раза больше числа узлов, находящихся в зоне радиуса R принимающего узла и составляющих множество узлов, которым следует запретить передачу. Таким образом, с точки зрения скорости передачи это не многообещающий подход.

При другой разновидности использования сигнала занятости узел посылает сигнал занятости только после того, как примет адресную часть пакета и определит, что он сам является предназначенным получателем. Это сильно увеличивает β — время, в течение которого другой узел может начать передачу, перед тем как услышит сигнал занятости.

Нетрудно понять, что пакетные радиосети — это область, в которой больше вопросов, чем ответов, в отношении как жела-

тельной структуры, так и анализа. Вопросы модуляции и обнаружения пакетов еще больше усложняют ситуацию. В военных применениях часто желательно использовать методы передачи с расширенным спектром. Одно из следствий этого заключается в том, что если принимаются сразу два пакета, то приемник часто может захватить передачу одного, а другой пакет действует только как широкополосный шум. Если каждый приемник использует свой код передачи с расширенным спектром, то ситуация еще больше улучшается, поскольку приемник может отыскивать только свою последовательность и, таким образом, отделять пакеты, одновременно передаваемые другим приемникам. К сожалению, затухание передачи в пределах прямой видимости часто довольно сильное, так что посторонние пакеты, поступающие в узел, могут иметь намного более высокий уровень мощности, чем желаемые пакеты, и поэтому по-прежнему приводить к появлению конфликта.

4.7. Заключение

Центральная задача связи в системах с множественным доступом состоит в организации совместного использования канала связи множеством узлов, каждый из которых имеет случайные требования по обслуживанию. Эта задача возникает в локальных сетях, сетях городского масштаба, спутниковых сетях и различных видах радиосетей.

Разрешение конфликтов представляет собой один из подходов к организации совместного использования канала. Алгоритмы разрешения конфликтов позволяют получить малую задержку при большом числе слабо нагруженных узлов, но устойчивости должно уделяться особое внимание. Взаимосвязанные вопросы устойчивости, скорости передачи и задержки можно исследовать наиболее глубоко, если предполагается бесконечное число узлов. Это предположение позволяет изучить разрешение конфликтов без дополнительного усложнения, обусловленного очередями в узлах. Вводя это предположение, мы установили, что скорости передачи вплоть до $1/e$ пакетов за окно можно достичь при использовании стабилизированной синхронной АЛОНА, а скорости передачи вплоть до 0,487 — при использовании алгоритмов разбиения.

Резервирование представляет собой другой основной подход к совместному использованию канала в режиме множественного доступа. Канал можно резервировать в соответствии с заранее выбранным методом распределения (например, ВУ или ЧУ) или динамически. Динамическое резервирование делится в свою очередь на резервирование с использованием разрешения конфликтов и резервирование с использованием ВУ (или кругового опроса). Система с МДПН/ОК (т. е. ETHERNET) является известным примером использования разрешения конфликтов для

выполнения (неявных) резервирований. Кольца с передачей маркера, шины с передачей маркера и их усовершенствованные разновидности являются примерами использования кругового опроса для выполнения резервирований.

Существует большое число различных способов использования специфических характеристик конкретных сред с множественным доступом для выполнения резервирования в порядке кругового опроса. Одни из них для выполнения резервирования требуют время, пропорциональное β (времени распространения), а другие — пропорциональное β/m , где m — число узлов. Последние способы особенно подходят для использования в высокоскоростных системах с $\beta > 1$.

Пакетные радиосистемы занимают промежуточное положение между системами с множественным доступом, в которых все узлы совместно используют общую среду (и, таким образом, не требуется явная маршрутизация), и системами с передачей от точки к точке (в которых требуется не совместное использование среды в режиме множественного доступа, а маршрутизация). Исследование радиосетей находится все еще в стадии формирования и фрагментарного изучения.

4.8. Замечания, источники и дополнительная литература

Раздел 4.2. Сеть ALONA впервые описана в работе [2], а ее синхронный усовершенствованный вариант в [200]. Проблема устойчивости обсуждалась в [41, 155, 173]. Метод передачи с замедлением по двоичной экспоненте был разработан в [172]. Современные методы стабилизации рассмотрены в [110, 199].

Раздел 4.3. Первые древовидные алгоритмы описаны в [42, 113, 241]. В работе [169] представлены усовершенствованные и простые методы анализа. Алгоритм разбиения с ППП разработан в [88] и независимо в [242]. Верхние границы максимальной скорости передачи (в модели с предположениями 1—6б из подразд. 4.2.1) получены в [175, 190]. Мартовский номер журнала IEEE Transactions on Information Theory посвящен системам связи с множественным доступом; в его статьях дано описание текущего состояния дел в области алгоритмов разбиения.

Раздел 4.4. Классическими работами по МДПН являются [149] и [230].

Раздел 4.5. Литературы по локальным сетям и спутниковым сетям более чем достаточно. В работе [220] представлено много практических деталей. Хорошая библиография по спутниковым сетям имеется в [33, 51, 130, 254]. Что касается локальных сетей, то [172] является первой работой по сети ETHERNET (см. также

[239]), а [66] и [67] — первые работы по кольцевым сетям. Работы [50, 153] содержат хорошие обзоры. В [71] сравниваются методы с опросом и передачей явных и неявных маркеров в системах с общей шиной.

Раздел 4.6. В работе [138] представлен хороший обзор пакетных радиосетей. Методы использования сигналов занятости описаны в [217, 231]. Вопросы выбора радиуса передачи обсуждаются в [225].

Задачи

- 4.1. (а) Убедитесь в том, что стационарные вероятности p_n цепи Маркова, показанной на рис. 4.3, являются решением уравнений

$$p_n = \sum_{i=0}^{n+1} p_i P_{in}, \quad \sum_{n=0}^m p_n = 1.$$

- (б) Для $n < m$ используйте а), чтобы выразить p_{n+1} через p_0, p_1, \dots, p_n .
 (в) Выразите p_1 через p_0 , а затем p_2 через p_0 .
 (г) При $m = 2$ найдите выражение для p_0 , зависящее от переходных вероятностей.

- 4.2. (а) Покажите, что $P_{\text{усп}}$ из (4.5) можно представить в следующем виде:

$$P_{\text{усп}} = \left[\frac{(m-n)q_a}{1-q_a} + \frac{nq_r}{1-q_r} \right] (1-q_a)^{m-n} (1-q_r)^n.$$

- (б) Используя приближение $(1-x)^y \approx e^{-xy}$ для малых x , покажите, что при малых q_a и q_r

$$P_{\text{усп}} \approx G(n) e^{-G(n)},$$

где $G(n) = (m-n)q_a + nq_r$.

- (в) Заметим, что $(1-x)^y = e^{y \ln(1-x)}$. Разложите $\ln(1-x)$ в степенной ряд и покажите, что

$$\frac{(1-x)^y}{e^{-xy}} = \exp \left(-\frac{x^2 y}{2} - \frac{x^3 y}{3} \dots \right).$$

Покажите, что это отношение близко к 1, когда $x \ll 1$ и $x^2 y \ll 1$.

- 4.3. (а) Изобразите графики зависимостей, показанных на рис. 4.4, в случае когда $q_r = 1/m$, а $q_a = 1/me$.

- (б) Найдите скорость ухода (т. е. $P_{\text{усп}}$) в случае всеобщей задолженности $n = m$.

- (в) Обратите внимание на то, что в этом случае не существует неустойчивого равновесия или нежелательной устойчивой точки, и покажите (графически), что это справедливо при любом значении q_a .

- (г) Найдите численно (используя $q_a = 1/me$) значение G , при котором система находится в устойчивой точке.

- (д) Найдите n/m в устойчивой точке. Заметьте, что это равно доле поступающих пакетов, которые не принимаются системой в этой типичной точке.

- 4.4. Рассмотрите идеализированную синхронную модель множественного доступа из подразд. 4.2.1 без предположения об отсутствии буферизации. Пусть n_k обозначает число задолженных узлов в начале k -го окна, а \bar{n} — среднее значение n_k по всем k . Заметим, что \bar{n} будет зависеть от конкретного способа разрешения конфликтов, но мы считаем здесь, что \bar{n} задано.

- (а) Найдите выражение для среднего числа *принятых* системой в течение окна новых пакетов \bar{N}_a , зависящее от \bar{n} , m и q_a , где m — число узлов, q_a — скорость поступления новых пакетов на узел.
- (б) Найдите выражение для средней скорости ухода за время окна $\bar{P}_{\text{усп}}$, зависящее от \bar{n} , m и q_a . *Указание:* как величина N_a связана с $\bar{P}_{\text{усп}}$? Вспомните, что обе величины являются средними по времени.
- (в) Найдите среднее число пакетов в системе $\bar{N}_{\text{сист}}$ непосредственно после начала окна (это число складывается из задолженных и принятых новых пакетов).
- (г) Найдите среднюю задержку принятого пакета, которая определяется как время, прошедшее от момента поступления пакета в начале окна до момента завершения его успешной передачи в конце окна. *Указание:* используйте теорему Литтла; полезно, быть может, еще раз нарисовать диаграмму, используемую при доказательстве теоремы Литтла.
- (д) Предположим теперь, что стратегия разрешения конфликтов модифицирована и средняя задолженность \bar{n} уменьшилась ($\bar{n}' < \bar{n}$). Покажите, что \bar{N}_a увеличивается, $\bar{P}_{\text{усп}}$ увеличивается, $\bar{N}_{\text{сист}}$ уменьшается и T уменьшается. Заметим, что это означает, что улучшение одного из этих параметров системы влечет за собой улучшение всех остальных.

4.5. Предположим для простоты, что каждый переданный пакет в синхронной системе ALOHA успешно передается с некоторой фиксированной вероятностью p . Предполагается, что новые пакеты поступают в начале окна и сразу передаются. Если пакет передан неуспешно, то он передается повторно с вероятностью q_r в каждом последующем окне до тех пор, пока не будет успешно принят.

- (а) Найдите среднюю задержку T , определяемую как время, прошедшее от момента поступления пакета до момента завершения его успешной передачи. *Указание:* если задано, что пакет еще не передан успешно, то какова вероятность того, что он будет успешно передан в i -м окне ($i > 1$) после момента поступления?
- (б) Предположим, что число узлов m велико и что q_a и q_r малы. Покажите, что в состоянии n вероятность p того, что заданная передача пакета будет успешной, приближенно равна $p = e^{-G(n)}$, где $G(n) = (m - n)q_a + nq_r$.
- (в) Теперь рассмотрим состояние устойчивого равновесия n^* системы, в котором $G = G(n^*)$; $Ge^{-G} = (m - n^*)q_a$. Подставьте выражение из б) в выражение для T из а), используя $n = n^*$, и покажите, что

$$T = 1 + \frac{n^*}{q_a(m - n^*)}.$$

(Заметим, что если предполагается, что n^* равно \bar{n} из задачи 4.4, то это выражение дает то же значение T , что получено там.)

- (г) Найдите численное значение T в случае, когда $q_a m = 0,3$, а $q_r m = 1$; покажите, что $n^* \approx m/8$ (это соответствует потере 1/8 доли поступающего трафика) и что $T \approx m/2$; это примерно совпадает с задержкой ВУ.

4.6. (а) Рассмотрите $P_{\text{усп}}$, которое точно определяется в (4.5). При заданных q_a , m , n покажите, что значение q_r , которое максимизирует $P_{\text{усп}}$, удовлетворяет уравнению

$$\frac{1}{1 - q_r} - \frac{q_a(m - n)}{1 - q_a} - \frac{q_r n}{1 - q_r} = 0.$$

- (б) Будем считать значение q_r , которое удовлетворяет указанному уравнению, функцией q_a , например $q_r(q_a)$. Покажите, что $q_r(q_a) > q_a$ (предположите, что $q_a < 1/m$).

- (в) Возьмите полную производную от $P_{\text{усп}}$ по q_a , используя $q_r(q_a)$ для q_r , и покажите, что эта производная отрицательна. *Указание:* вспомните, что $\partial P_{\text{усп}}/\partial q_r$ равна 0 при $q_r = q_r(q_a)$ и сравните $\partial P_{\text{усп}}/\partial q_a$ с $\partial P_{\text{усп}}/\partial q_r$.
- (г) Покажите, что если q_r имеет значение, максимизирующее $P_{\text{усп}}$ и $q_r < 1$, то значение $P_{\text{усп}}$ в случае, когда новые пакеты сразу становятся задолженными, больше значения $P_{\text{усп}}$ в случае, когда новые пакеты сразу передаются. *Указание:* в случае немедленного перехода в задолженное состояние узел, бывший ранее незадолженным, передает с вероятностью $q_a q_r < q_a$.
- 4.7.** Рассмотрите синхронную систему ALOHA с совершенным захватом, т. е. если более чем один пакет передается в окне, то приемник захватывает одну из передач и принимает ее безошибочно; обратная связь немедленно сообщает каждому передающему узлу о том, какой узел принят успешно, и пакеты, переданные неуспешно, передаются повторно в дальнейшем.
- (а) Приведите убедительный аргумент того, что средняя задержка минимизируется, если делаются попытки передать все ожидающие пакеты в каждом окне.
- (б) Найдите среднюю задержку в системе, предполагая, что потоки поступающих пакетов пуассоновские с общей интенсивностью λ . *Указание:* рассмотрите пример из подразд. 3.5.1.
- (в) Теперь предположим, что обратная связь имеет задержку, и если пакет передается неуспешно в окне, то он передается повторно не в первом последующем окне, а в k -м последующем окне. Найдите в этом случае среднюю задержку как функцию k . *Указание:* рассмотрите систему как совокупность k подсистем, таких что i -я подсистема обрабатывает пакеты, поступившие в окнах с номерами j , такими, что $j \bmod k = i$.
- 4.8.** Рассмотрите синхронную систему, в которой все узлы имеют бесконечно большие буферы и все новые поступающие пакеты (образующие в каждом узле пуассоновский поток с интенсивностью λ/m) принимаются системой, но сразу становятся задолженными, а не передаются в следующем окне. Когда узел имеет один пакет или более, он независимо передает один пакет в каждом окне с вероятностью q_r . Пусть любая заданная передача приводит к успеху с вероятностью p .
- (а) Покажите, что среднее время, прошедшее от начала окна с задолженностью до конца первой успешной передачи в заданном узле, равно $1/(pq_r)$. Покажите, что второй момент этого времени равен $(2 - pq_r)/(pq_r)^2$.
- (б) Заметим, что предположение о постоянной вероятности успешной передачи позволяет рассматривать каждый узел независимо от других узлов. Пусть λ/m — интенсивность пуассоновского входного потока в узле; используя результаты относительно времени обслуживания из а), покажите, что средняя задержка равна
- $$T = \frac{1}{q_r p (1 - \rho)} + \frac{1 - 2\rho}{2(1 - \rho)}, \quad \rho = \frac{\lambda}{m p q_r}.$$
- (в) Предположим, что $\rho = 1$ (это минимизирует T и соответствует случаю очень слабой нагрузки). Найдите T при $q_r = 1/m$; заметьте, что она примерно в два раза больше задержки ВУ, если m велико.
- 4.9.** Предположим, что число пакетов n в синхронной системе ALOHA в заданный момент времени является пуассоновской случайной величиной со средним $\lambda \geq 1$. Предположим, что каждый пакет независимо передается в следующем окне с вероятностью $1/\lambda$.
- (а) Найдите вероятность того, что это окно будет пустым.
- (б) Покажите, что апостериорная вероятность того, что в системе было n пакетов при условии, что появилось пустое окно, имеет пуассоновское распределение со средним $\lambda - 1$.

- (в) Найдите вероятность того, что в этом окне будет успешная передача.
 (г) Покажите, что апостериорная вероятность того, что в системе было $n + 1$ пакетов при условии, что возникла успешная передача, равна $e^{-(n+1)} (n+1)^n / n!$ (т. е. число оставшихся пакетов имеет пуассоновское распределение со средним $n + 1$).

4.10. Рассмотрите синхронную систему ALOHA, удовлетворяющую условиям 1—6а из подразд. 4.2.1, за исключением того, что каждый узел имеет неограниченный буфер для хранения всех поступающих пакетов до их передачи и узлы получают сигналы быстрой обратной связи, сообщающей только о том, были ли переданы успешно их собственные пакеты. Каждый узел находится в одном из двух различных состояний. В состоянии 1 узел передает (с вероятностью 1) в каждом окне пакеты, которые не были успешно переданы, до тех пор пока не очистится его буфер с пакетами, ожидающими передачи; в этот момент узел переходит в состояние 2. В состоянии 2 узел передает в каждом окне фиктивный пакет с вероятностью q_r , до тех пор, пока не произойдет успешная передача, после чего он возвращается в состояние 1 (фиктивные пакеты используются только для упрощения математического анализа). Пусть в момент начала работы системы все узлы находятся в состоянии 2. В каждый узел поступает пуассоновский поток новых пакетов с интенсивностью λ/m .

- (а) Объясните, почему в каждый момент времени не более одного узла может находиться в состоянии 1.
 (б) При условии что пакет находится в состоянии 1, найдите вероятность его успешной передачи p_1 . Получите среднее время \bar{x} между успешными передачами и второй момент \bar{x}^2 этого времени. *Указание:* рассмотрите пример ARQ из подразд. 3.5.1 гл. 3 при $N = 1$.
 (в) При условии что все узлы находятся в состоянии 2, найдите вероятность p_2 того, что некоторый фиктивный пакет успешно передается в заданном окне. Найдите среднее время \bar{v} до завершения такой успешной передачи и его второй момент \bar{v}^2 .
 (г) Будем считать интервалы времени, когда все узлы находятся в состоянии 2, интервалами резервирования. Покажите, что среднее время, которое пакет должен провести в очереди до первой попытки передачи, равно

$$W = \frac{R + E\{S\} \bar{v}}{1 - \rho}, \quad \rho = \lambda \bar{x},$$

где R — среднее оставшееся время, необходимое для завершения обслуживания в состоянии 1 или для окончания интервала резервирования, а S — число полных интервалов резервирования, проходящих за время, пока узел, в который поступил пакет, не перейдет в состояние 1.

- (д) Покажите, что

$$W = \frac{\lambda (2 - p_1)}{2p_1^2 (1 - \rho)} + \frac{2 - p_2}{2p_2} + \frac{m - 1}{p_2 (1 - \rho)}.$$

Покажите, что W — конечная величина, если $q_r < (1 - \lambda)/(m - 1)$.

4.11. Рассмотрите довольно нереальное предположение об обратной связи для несинхронной ALOHA, при которой всем узлам через τ единиц времени после начала каждой передачи сообщается, была ли эта передача успешной. Таким образом, в случае конфликта в каждом узле известно, сколько пакетов вступило в конфликт и в каждом узле, вступившем в конфликт, известно, сколько других узлов начали передачи перед ним. Пусть каждая передача длится одну единицу времени и $m = \infty$. Рассмотрим стратегию повторения передачи, при которой первый узел, вступивший в конфликт, ждет одну единицу времени после получения сигнала обратной связи,

известившего о конфликте, и затем передает свой пакет. Последующие узлы, вступившие в конфликт, передают повторно через одну единицу времени после начала повторной передачи предыдущего узла. Все новые пакеты, поступающие в систему во время этих повторных передач, ждут окончания повторных передач. После завершения повторных передач каждый узел с задолженностью выбирает момент начала своей передачи с помощью равномерного распределения на следующей единице времени. Все новые пакеты, поступившие после окончания упомянутых выше повторных передач, начинают передаваться немедленно.

(а) Постройте приближенную модель описанной системы с помощью системы резервирования с интервалами резервирования длины $1 + \tau$ (заметим, что это является приближением в том смысле, что успешные передачи будут иногда возникать в интервалах резервирования, но приближенная модель становится более точной по мере увеличения нагрузки). Найдите среднюю задержку пакета в этой приближенной модели (предполагая, что поток новых пакетов пуассоновский с интенсивностью λ).

(б) Покажите, что эта задержка остается конечной при всех $\lambda < 1$.

4.12. Эта задача показывает, что максимальная скорость передачи несинхронной АЛОНА может быть повышена до e^{-1} за счет очень большого увеличения задержки. Рассмотрим конечное, но большое число узлов m с неограниченными буферами в каждом узле. Каждый узел ждет до тех пор, пока не накопится k пакетов, и затем передает их один за другим в течение следующих k единиц времени. Пакеты, вступившие в конфликты, вместе с новыми пакетами передаются затем повторно, спустя случайное время, причем опять серией из k пакетов. Предположим, что начала передач всех узлов в совокупности образуют пуассоновский процесс с параметром G (т. е. оставим в стороне вопросы устойчивости).

(а) Покажите, что вероятность успешной передачи j -го пакета в серии из k пакетов равна $e^{-G(k+1)}$. *Указание:* рассмотрите интервалы между началом данной серии, началом предыдущей серии и началом последующей серии.

(б) Покажите, что скорость передачи равна $kGe^{-G(k+1)}$, и найдите максимальную скорость передачи, оптимизируя по G .

4.13. (а) Рассмотрите ПРК, который получается при поступлении сигналов обратной связи $e, 0, e, e, 1, 1, 0$, когда используется древовидный алгоритм, иллюстрируемый на рис. 4.9. Изобразите аналогичный рисунок для этой последовательности сигналов обратной связи.

(б) Какого конфликта или конфликтов можно было бы избежать, если бы использовалась первая модификация древовидного алгоритма?

(в) Какой была бы последовательность сигналов обратной связи в этом ПРК, если бы использовались обе модификации древовидного алгоритма?

4.14. Рассмотрите древовидный алгоритм, иллюстрируемый на рис. 4.9. При условии что k конфликтов возникло в течение ПРК, определите число окон, необходимых для ПРК. Проверьте свой ответ на конкретном примере рис. 4.9. *Указание 1:* обратите внимание, что каждый конфликт соответствует неконцевому узлу (не листу) корневого дерева. Рассмотрите построение любого заданного дерева из его корня, при котором листья последовательно заменяются на внутренние узлы с двумя идущими вверх ребрами. *Указание 2:* при другом подходе рассмотрите, что происходит в стеке при каждом конфликте, пустом или успешном окне.

4.15. Рассмотрите древовидный алгоритм, иллюстрируемый на рис. 4.9. Предположим, что после каждого конфликта каждый пакет, вступивший в конфликт, подбрасывает симметричную монету, чтобы определить, в левое или правое подмножество ему переходить.

- (а) При условии что в конфликт вступило k пакетов, найдите вероятность того, что i пакетов перейдут в левое подмножество.
 (б) Пусть A_k обозначает среднее число окон, необходимых ПРК, в первом окне которого передавалось k пакетов. Заметим, что $A_0 = A_1 = 1$. Покажите, что при $k \geq 2$

$$A_k = 1 + \sum_{i=0}^k \binom{k}{i} 2^{-k} (A_i + A_{k-i}).$$

- (в) Упростите ваш результат из (б), представив его в следующем виде:

$$A_k = c_{kk} + \sum_{i=0}^{k-1} c_{ik} A_i$$

и найдите коэффициенты c_{ik} . Вычислите значения A_2 и A_3 . Чтобы ознакомиться с другими результатами относительно A_k при больших k и с использованием A_k при оценке максимальной скорости передачи, обратитесь к работе [169].

- 4.16. (а) Рассмотрите первую модификацию древовидного алгоритма, иллюстрируемую на рис. 4.10. Предположим, что каждый пакет, вступивший в конфликт, подбрасывает симметричную монету перед тем, как присоединиться к левому или правому подмножеству. Пусть B_k обозначает среднее число окон в ПРК, в первом окне которого передавалось k пакетов; заметим, что $B_0 = B_1 = 1$. Покажите, что при $k \geq 2$

$$B_k = 1 + \sum_{i=1}^k \binom{k}{i} 2^{-k} (B_i + B_{k-i}) + 2^{-k} B_k.$$

- (б) Упростите ваш результат, приведя его к виду

$$B_k = C'_{kk} + \sum_{i=1}^{k-1} C'_{ik} B_i,$$

и оцените значения констант C'_{ik} . Найдите числовые значения B_2 и B_3 . (См. [169].)

- 4.17. Пусть X_L и X_R обозначают независимые пуассоновские случайные величины, каждая из которых имеет среднее G . Используйте определение условной вероятности, чтобы оценить следующие величины:

- (а) $P \{X_L = 0 \mid X_L + X_R \geq 2\}$,
 (б) $P \{X_L = 1 \mid X_L + X_R \geq 2\}$,
 (в) $P \{X_L \geq 2 \mid X_L + X_R \geq 2\}$,
 (г) $P \{X_R = 1 \mid X_L = 1, X_L + X_R \geq 2\}$,
 (д) $P \{X_R = i \mid X_L = 0, X_L + X_R \geq 2\}$, $i \geq 2$,
 (е) $P \{X_R = i \mid X_L \geq 2, X_L + X_R \geq 2\}$.

- 4.18. Пусть в момент времени k алгоритм разбиения с ППП назначает новый интервал от $T(k)$ до $T(k) + \alpha_0$. Предположим, что этот интервал содержит множество пакетов, поступающих в следующие моменты времени: $T(k) + 0,1\alpha_0$, $T(k) + 0,6\alpha_0$, $T(k) + 0,7\alpha_0$ и $T(k) + 0,8\alpha_0$.

- (а) Найдите назначенные интервалы для каждого окна ПРК.
 (б) Укажите, какие инструкции из набора (4.15)—(4.18) используются при определении каждого из этих назначенных интервалов.
 (в) Укажите путь на диаграмме, показанной на рис. 4.13, соответствующий этой последовательности событий.

- 4.19. (а) Покажите, что \bar{n} — среднее число пакетов, успешно переданных за время ПРК при алгоритме разбиения с ППП, определяется выражением

$$\bar{n} = 1 - e^{-G_0} + \sum_{i=1}^{\infty} p(R, i).$$

(Предполагается, что исходный назначенный интервал имеет длину α_0 , а $G_0 = \lambda \alpha_0$.)

- (б) Покажите, что

$$\bar{n} = \lambda \alpha_0 (1 - E\{f\}),$$

где $E\{f\}$ обозначает долю интервала длины α_0 , снова ставшей частью ожидающего интервала за время ПРК. (Это дает другой способ вычисления $E\{f\}$.)

- 4.20. Чтобы обосновать соотношения (4.41), покажите, что если n_k является пуассоновской случайной величиной со средним \hat{n}_k , то апостериорное распределение n_k при условии, что задано пустое окно, является пуассоновским со средним $\hat{n}_k [1 - q_r(\hat{n}_k)]$. Покажите, что апостериорное распределение $n_k - 1$ при условии, что произошла успешная передача, является пуассоновским со средним $\hat{n}_k [1 - q_r(\hat{n}_k)]$.

- 4.21. *Синхронная АЛОНА с переменной длиной пакета.* Пусть время передачи пакета является случайной величиной X ; в соответствии с предположением о синхронности допустим, что X — дискретная величина, значения которой кратны β . Будем считать, что все передачи — независимые и одинаково распределенные (НОР) — случайные величины со средним $\bar{X} = 1$.

- (а) Пусть Y обозначает более длинную передачу среди двух НОР передач X_1 и X_2 (т. е. $Y = \max(X_1, X_2)$). Покажите, что среднее значение Y удовлетворяет неравенству $\bar{Y} \leq 2\bar{X}$. Покажите, что если X принимает значение β с большой вероятностью, а $k\beta$ (при большом k) — с малой вероятностью, то неравенство переходит в равенство.

- (б) Покажите, что среднее время пребывания в состоянии при условии, что возник конфликт между двумя пакетами, не больше чем $2 + \beta$.

- (в) Пусть $g(n) = \lambda\beta + q_r n$ обозначает среднее число попыток передач после перехода в состояние n и предположим, что это число попыток является пуассоновской случайной величиной со средним $g(n)$. Покажите, что среднее время пребывания в состоянии n не более

$$\beta e^{-g(n)} + (1 + \beta) g(n) e^{-g(n)} + (1 + \beta/2) g^2(n) e^{-g(n)}.$$

Конфликтами между более чем двумя пакетами, можно пренебречь.

- (г) Найдите нижнюю границу среднего числа уходов пакетов из системы за единичное время в состоянии n (см. равенство (4.39)).

- (д) Покажите, что эта нижняя граница максимальна (при малом β) при $g(n) \approx \sqrt{\beta}$; соответствующая скорость передачи примерно равна $1 - 2\sqrt{\beta}$.

- 4.22. *Псевдобайесовская стабилизация несинхронного МДПН.* Пусть в конце передачи пакета число задолженных пакетов n в системе является пуассоновской случайной величиной со средним \hat{n} . Предположим, что во время пустого периода, заканчивающегося началом следующей передачи, предпринимаются попытки передачи с интенсивностью x , а каждый новый пакет передается сразу после поступления. Таким образом, если число задолженных пакетов равно n , то время до начала следующей передачи имеет плотность распределения вероятностей $p(\tau | n) = (\lambda + xn) e^{-(\lambda + xn)\tau}$.

- (а) Найдите безусловное распределение вероятностей $p(\tau)$.

- (б) Найдите апостериорную вероятность $p(n, b | \tau)$ того, что задолженных пакетов было n и один из них начал передаваться первым при условии, что передача началась через время τ .

- (в) Найдите апостериорную вероятность $p(n, a | \tau)$ того, что задолженных пакетов было n и начал передаваться первым новый пакет при условии, что эта передача началась через время τ .
- (г) Пусть n' обозначает число задолженных пакетов в системе после начала следующей передачи (без учета пакета, находящегося в процессе передачи); т. е. $n' = n - 1$, если начал передаваться задолженный пакет, и $n' = n$, если начал передаваться новый пакет. Покажите, что если задано τ , то n' имеет пуассоновское распределение со средним $\hat{n}' = \hat{n}e^{-\tau x}$. Это означает, что псевдобайесовское правило обновления оценки задолженности (в предположении, что время передачи пакета — единичное) состоит в том, что задолженность оценивается в конце $(k+1)$ -й передачи на основе оценки в конце k -й передачи и длины пустого периода τ_k между передачами следующим образом:

$$\hat{n}_{k+1} = \begin{cases} \hat{n}_k e^{-\tau_k x_k} + \lambda(1 + \beta), & \text{успешная передача,} \\ \hat{n}_k e^{-\tau_k x_k} + 2 + \lambda(1 + \beta), & \text{конфликт,} \end{cases}$$

$$x_k = \beta^{-1/2} \min \left(\frac{1}{\hat{n}_k}, 1 \right).$$

- 4.23. Дайте интуитивное объяснение того, почему при малом β максимальная скорость передачи синхронной АЛОНА с МДПН и алгоритма разбиения с ППП и МДПН примерно одинакова. Покажите, что оптимальное среднее число пакетов, переданных после перехода в другое состояние, для системы АЛОНА такое же, как в начале ПРК для алгоритма разбиения с ППП. Обратите внимание на то, что после конфликта эти средние величины двух систем немного различаются, но различие незначительно, так как конфликты редки.
- 4.24. *Задержка идеальной синхронной системы с МДПН/ОК.* Предположим, что всегда, когда имеется положительная задолженность, число пакетов, передаваемых в интервале, имеет пуассоновское распределение со средним g .
- (а) Начните с равенства (4.42), которое справедливо для МДПН/ОК, так же как для МДПН. Покажите, что для МДПН/ОК

$$E\{t\} = \frac{\beta e^{-g} + (1 + \beta) g e^{-g} + 2\beta [1 - (1 + g) e^{-g}]}{g e^{-g}}.$$

Покажите, что $E\{t\}$ минимизируется по $g > 0$, когда $g = 0,77$, и

$$\min_g E\{t\} = 1 + 3,31\beta.$$

- (б) Покажите, что при этом g и единичной средней длине пакета

$$W = \frac{\bar{R} + \bar{y}}{1 - \lambda(1 + 3,31\beta)}.$$

- (в) Оцените \bar{R} и \bar{y} и проверьте равенство (4.67) при малом β .
- (г) Обсудите предположение о пуассоновском распределении числа пакетов, передаваемых в каждом интервале и рассмотрите, в частности, случай, когда задолженность равна 1.
- 4.25. Покажите, что для несинхронной системы с МДПН/ОК максимальная длина интервала времени, в течение которого передающий узел может слышать конфликт, равна 2β (обратите внимание на рис. 4.20 на то, что время, прошедшее от начала конфликтного события в одном узле до его окончания в другом узле, не превышает 3β).

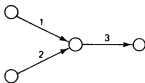


Рис. 4.30.

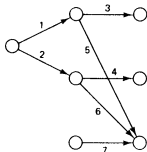


Рис. 4.31.

- 4.26. Рассмотрите несинхронную систему с МДПН/ОК, в которой время пространства пренебрежимо мало по сравнению с временем β , необходимым узлу для обнаружения того, что канал пуст или занят. Пусть передача каждого пакета занимает одну единицу времени. Предположим, что через β единиц времени после окончания успешной передачи или конфликта все узлы с задолженностью после случайной задержки пытаются передать и суммарный процесс моментов иницирования является пуассоновским с интенсивностью G (вплоть до момента, когда истекает β единиц времени после первого иницирования). Предположим для простоты, что каждый конфликт длится β единиц времени.
- Найдите вероятность того, что первое иницирование передачи после того, как обнаружено пустое состояние канала, является успешным.
 - Найдите среднее время от одного обнаружения пустого состояния до следующего.
 - Найдите скорость передачи (при сделанных предположениях).
 - Выполните численную оптимизацию скорости передачи по G .
- 4.27. Модифицируйте равенство (4.71) для случая, когда после передачи пакета узел ждет возвращения пакета перед тем, как передать свободный маркер. *Указание:* рассматривайте дополнительную задержку как увеличение времени передачи пакета.
- 4.28. Покажите на примере, что в кольце с регистрами может оказаться так, что узлу приходится ждать произвольно долгое время, чтобы передать пакет, если его транзитный буфер полный.
- 4.29. Предположим, что два узла случайно расположены на шине, т. е. каждый размещается независимо от другого и положение каждого из них имеет равномерное распределение на длине шины. Предполагая, что длина шины единичная, покажите, что среднее расстояние между узлами равно $1/3$.
- 4.30. С целью анализа обобщенного опроса из подразд. 4.5.6 предположим, что каждый узел имеет пакет для передачи с вероятностью q .
- Найдите вероятность $P\{i\}$ того, что узел i , $i \geq 0$ является узлом с наименьшим номером, имеющим пакет для передачи.
 - Предположим, что в ходе ПРК в каждый момент времени тестируются только первые 2^j узлов с наименьшими номерами при некотором j . Представим номер узла с пакетом, у которого этот номер наименьший, в виде $i = k2^j + r$, где k — целое, а $0 \leq r < 2^j$. Покажите, что при заданном i число окон резервирования, необходимое для отыскания i , равно $k + 1 + j$.
 - Предположим, что общее число узлов бесконечно и будем использовать для упомянутой выше величины k приближенную оценку $i2^{-j}$. Найдите

среднее число окон резервирования, необходимых для отыскания узла с пакетом, имеющего наименьший номер i .

- (г) Вычислите целое число j , которое минимизирует ваш ответ в п. (в). Указание: найдите наибольшее значение j , для которого среднее число окон резервирования меньше чем среднее число для $j+1$.

4.31. Рассмотрите простую пакетную радиосеть, показанную на рис. 4.30, и пусть f_l обозначает скорость передачи по линии l ($l = 1, 2, 3$); передачи по линиям производятся только в указанных направлениях.

- (а) Заметим, что в каждом свободном от конфликтов множестве может находиться не более одной линии. Используя обобщенное ВУ (как в равенстве (4.75)), покажите, что можно достичь любого множества скоростей передач, удовлетворяющих

$$f_1 + f_2 + f_3 \leq 1, \quad f_l \geq 0 \quad (l = 1, 2, 3).$$

- (б) Пусть $f_1 = f_2 = \bar{f}$ и $f_3 = 2\bar{f}$ при некотором \bar{f} . Используйте соотношения (4.77) и (4.78), чтобы установить связь между f_1, f_2, f_3 и интенсивностями

попыток q_1, q_2, q_3 для синхронной АЛОНА. Покажите, что $q_1 = q_2$ и $q_3 = 2\bar{f}$.

- (в) Найдите максимальное достижимое значение \bar{f} из (б).

4.32. Рассмотрите пакетную радиосеть, показанную на рис. 4.31, в которой передачи по линиям производятся только в указанных направлениях. Линии 5 и 6 не передают трафика; они служат для того, чтобы вызывать конфликты на линии 7, когда пакеты передаются по линиям 3 и 4 соответственно.

- (а) Покажите, что скорости передачи $f_1 = f_2 = f_3 = f_4 = 1/3$, $f_7 = 4/9$ достижимы при условии сильной нагрузки и независимых попыток передач по каждой линии. Найдите соответствующие интенсивности попыток и интенсивности успешных передач из формул (4.77) и (4.78).

- (б) Теперь предположим, что линии 3 и 4 используются для передачи поступающего трафика и передача по ним всегда осуществляется в окне, которое следует за успешным поступлением пакета. Покажите, что при $f_1 = f_2 = f_3 = f_4 = 1/3$ скорость передачи f_7 ограничена сверху величиной $1/3$.

- (в) Предположим, наконец, что попытки передач пакетов по линиям 1 и 2 осуществляются попеременно, а линии 3 и 4 действуют так же, как в (б). Покажите, что скорости передачи $f_1 = f_2 = f_3 = f_4 = 1/2$ достижимы, но f_7 тогда должна быть равна 0.

5. Маршрутизация в сетях передачи данных

5.1. Введение

Под алгоритмом маршрутизации мы часто подразумеваем протокол сетевого уровня, который управляет пакетами при их движении по подсети связи до требуемого места назначения. Моменты времени, когда принимаются решения о выборе маршрута, зависят от того, использует ли сеть дейтаграммную передачу или виртуальные соединения. В дейтаграммной сети два последовательных пакета одной и той же пары пользователей могут проходить по разным маршрутам и выбирать маршрут необходимо индивидуально для каждого пакета (рис. 5.1). В сети с виртуальными соединениями маршрут выбирается при установлении каждого виртуального соединения. Алгоритм маршрутизации используется для выбора пути по сети для данного виртуального соединения. Все пакеты виртуального соединения последовательно используют этот путь вплоть до момента, когда либо данное виртуальное соединение заканчивает свое существование, либо когда для данного соединения по каким-либо причинам выбирается другой маршрут (рис. 5.2).

Обычно для выбора маршрута используется довольно сложный набор алгоритмов, которые работают более или менее независимо, хотя и обмениваются информацией. Его сложность обусловлена рядом причин. Во-первых, маршрутизация требует координации работы всех узлов подсети, а не только одной пары модулей, как, например, в протоколах уровней линии передачи данных и транспортного. Во-вторых, система маршрутизации должна справляться с выходами из строя линий или узлов путем перенаправления трафика и обновления баз данных, используемых системой. В-третьих, для достижения наилучших характеристик алгоритм маршрутизации может изменить маршруты, когда некоторые области сети становятся перегруженными.

Основное внимание будет уделено двум аспектам задачи маршрутизации. Первый касается выбора маршрутов для достижения наилучших характеристик. В подразд. 5.2.3—5.2.5 рассматриваются алгоритмы кратчайшего пути, которые широко используются на практике. В разд. 5.5—5.7 описываются более сложные алгоритмы, позволяющие достигать характеристики, близкие к оптимальным. Второй аспект задачи маршрутизации, которому

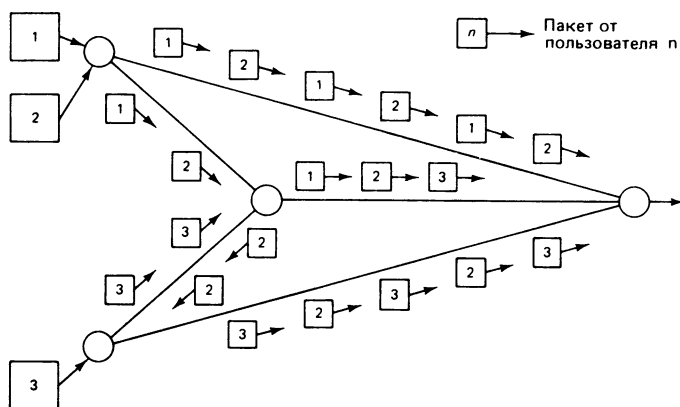


Рис. 5.1. Маршрутизация в дейтаграммной сети. Два пакета одной и той же пары пользователей могут проходить по разным маршрутам. Решение о выборе маршрута требуется принимать для каждого отдельного пакета.

будет уделено внимание, состоит в распространении между всеми узлами сети информации, необходимой для выбора маршрутов (включая информацию о выходе из строя и восстановлении линий и узлов). Этот аспект, рассматриваемый в разд. 5.3, имеет ряд тонкостей, которые не до конца оценены в этой области.

В разд. 5.1 сначала приводятся сведения, необходимые для перехода к главным результатам. Затем без математики описываются главные цели маршрутизации и дается обзор используемых

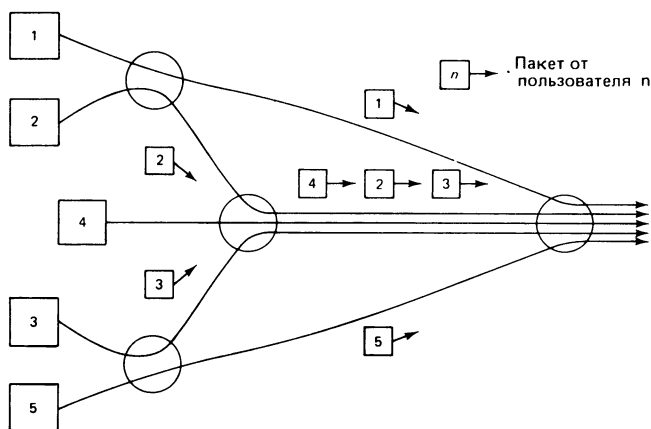


Рис. 5.2. Маршрутизация в сети с виртуальными цепями. Все пакеты каждой виртуальной цепи используют один и тот же путь. Решение о выборе маршрута требуется принимать только при установлении виртуальной цепи.

сейчас на практике методов маршрутизации. В подразд. 5.2.1—5.2.3 представлены некоторые основные обозначения и результаты теории графов, касающиеся в основном кратчайших путей и остовных деревьев минимального веса. В разд. 5.4 материал теории графов используется для описания методов построения топологий сетей. И наконец, в разд. 5.8 дается обзор системы маршрутизации сети CODEX и показана ее связь с алгоритмом оптимальной маршрутизации из разд. 5.7.

5.1.1. Основные понятия, используемые при маршрутизации

Двумя главными функциями, которые выполняет алгоритм маршрутизации, являются выбор маршрутов для различных пар отправитель — адресат и обеспечение правильной доставки сообщений их адресатам после того, как выбраны маршруты. Вторая функция обеспечивается путем использования различных протоколов и структур данных (называемых маршрутными таблицами), некоторые из которых будут описаны при обзоре существующих сетей связи в подразд. 5.1.2. Основное внимание будет уделено первой функции (выбору маршрутов) и тому, как это влияет на характеристики сети.

Существуют две основные характеристики, на которые существенное влияние оказывает алгоритм маршрутизации — *пропускная способность* (количество обслуживания) и *средняя задержка пакета* (качество обслуживания). Маршрутизация взаимодействует с управлением потоками в определении характеристик посредством механизма обратной связи, показанного на рис. 5.3. Когда трафик, поступающий в подсеть от внешних источников, относительно мал, он полностью будет принят сетью и тогда

Пропускная способность = Поступающая нагрузка.

Когда поступающая нагрузка чрезмерно большая, часть этой нагрузки будет отвергаться алгоритмом управления потоками и тогда

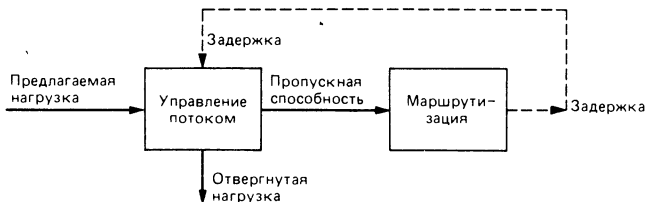


Рис. 5.3. Взаимосвязь между маршрутизацией и управлением потоком. Когда хорошая маршрутизация сохраняет задержку малой, управление потоком позволяет принимать в сеть больше трафика.

Пропускная способность =
Поступающая нагрузка —
Отвергнутая нагрузка

Трафик, принятый в сеть, будет иметь среднюю задержку пакетов, зависящую от того, какие маршруты были выбраны алгоритмом маршрутизации. Однако на пропускную способность существенное влияние оказывает также алгоритм маршрутизации (хотя, может быть, и не непосредственно), так как алгоритм управления потоками обычно действует на основе поддержания баланса между пропускной способностью и средней задержкой, например, поступающая нагрузка не принимается, как только задержка становится слишком большой. Поэтому *если алгоритму маршрутизации удастся более успешно поддерживать малую задержку, то алгоритм управления потоками разрешает принимать в сеть больше трафика*. Хотя точный баланс между задержкой и пропускной способностью устанавливается алгоритмом управления потоками, хорошая маршрутизация в условиях большого предлагаемого трафика дает более предпочтительную кривую задержка — пропускная способность, по которой действует алгоритм управления потоками (рис. 5.4).

Следующие примеры иллюстрирует вышесказанное.

Пример 1

На рис. 5.5 изображена сеть, все линии которой имеют пропускную способность 10 единиц. (Единицы, в которых измеряются пропускная способность линии и интенсивность нагрузки на ней, несущественны и не указываются.) Имеются только один адресат (узел 6) и два отправителя (узлы 1 и 2). Предлагаемая нагрузка от каждого из узлов 1 и 2 к узлу 6 равна 5 единицам. Здесь предлагаемая нагрузка является небольшой и ее можно легко разместить по крайнему слева

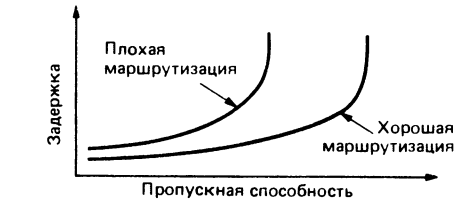
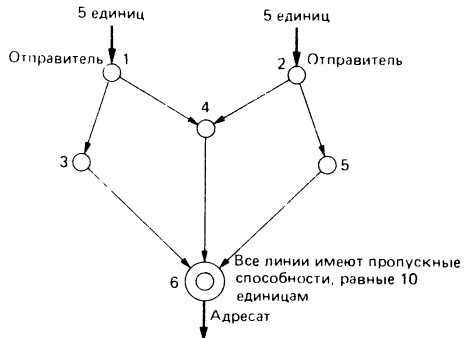


Рис. 5.4. Кривые зависимости задержки от пропускной способности для хорошей и плохой маршрутизации.

Рис. 5.5. Сеть для примера 1. Все линии имеют пропускные способности, равные 10 единицам. Если весь трафик направляется по средней линии (4,6), то возникает перегрузка. Если вместо этого используются пути (1 → 3 → 6) и (2 → 5 → 6), то средняя задержка мала.



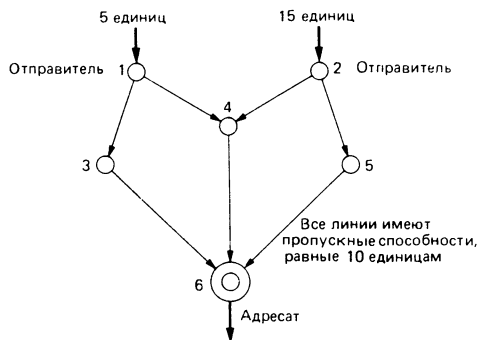


Рис. 5.6. Сеть для примера 2. Все линии имеют пропускные способности, равные 10 единицам. Входной трафик может быть обслужен, если при маршрутизации использовано несколько путей, но не менее 5 единиц трафика должны быть отвергнуты, если при маршрутизации используются однопутевые варианты.

(1 → 3 → 6) и крайнему справа (2 → 5 → 6) путям, обеспечив при этом малую задержку. Если же вместо этого использовать маршруты 1 → 4 → 6 и 2 → 4 → 6, то поток по линии (4, 6) будет иметь интенсивность, равную пропускной способности линии, и в результате возникнут очень большие задержки.

Пример 2

Для той же сети предположим, что интенсивности предлагаемой нагрузки от узлов 1 и 2 равны 5 и 15 единицам соответственно (рис. 5.6). Если нагрузку от узла 2 направить к адресату только по одному пути, то не менее 5 единиц предлагаемой нагрузки будет отвергнуто, так как все линии имеют пропускные способности, равные 10. Таким образом, суммарная пропускная способность сети будет не более 15 единиц. Вместе с тем предположим, что трафик, возникающий в узле 2, поровну делится между двумя путями 2 → 4 → 6 и 2 → 5 → 6, а трафик, возникающий в узле 1, направляется по пути 1 → 3 → 6. Тогда интенсивность проходящего трафика по каждой линии не превышает 75 % ее пропускной способности, задержка пакета будет относительно небольшой и никакая предлагаемая нагрузка не будет отвергнута (при хорошей схеме управления потоками). Рассуждая аналогично, можно увидеть, что если обе предлагаемые в узлах 1 и 2 нагрузки достаточно большие, то максимальная суммарная нагрузка, которую эта сеть способна обслужить, находится между 10 и 30 единицами в зависимости от схемы маршрутизации. Этот пример показывает, что для достижения большой пропускной способности трафик между некоторыми парами отправитель—адресат, видимо, должен быть поделен между несколькими маршрутами.

В заключение подведем итоги. *Если предлагаемая нагрузка велика, то хороший выбор маршрутов увеличивает пропускную способность сети при том же значении средней задержки пакета, а если предлагаемая нагрузка мала, то он уменьшает среднюю задержку пакета.* Более того, желательно, чтобы алгоритм маршрутизации работал так, чтобы задержка была минимальной для любого уровня пропускной способности. Это позволяет четко сформулировать математическую задачу и затем исследовать ее аналитически.

5.1.2. Обзор применяемых методов маршрутизации

Целью этого раздела являются ознакомление со способами маршрутизации, применяемыми в настоящее время на практике, классификация различных схем и подготовка основы для последующего математического анализа.

Существует несколько способов классификации алгоритмов маршрутизации. Один из них состоит в разделении всех алгоритмов на *централизованные* и *распределенные*. В централизованных алгоритмах выбор всех маршрутов осуществляется в центральном узле, а в распределенных — в узлах сети. При этом узлы обмениваются информацией, если это необходимо. Заметим, однако, что часто такая классификация относится к реализации, а не к математическому описанию алгоритма. Вполне возможно, что на некотором уровне математической абстракции распределенный и централизованные алгоритмы становятся эквивалентными.

Другая классификация алгоритмов маршрутизации основана на том, изменяются ли маршруты в зависимости от интенсивностей входных потоков. В *статических* алгоритмах маршрутизации путь, используемый каждой парой отправитель — адресат, фиксирован и не зависит от колебаний трафика. Он может меняться только в случае выхода из строя какого-либо узла или линии. При использовании алгоритмов этого типа не может достигаться большая пропускная способность при значительной вариации входного трафика. Такой способ маршрутизации рекомендуется применять либо для очень простых сетей, либо когда эффективность работы сети не существенна. Для большинства основных сетей с коммутацией пакетов используются разновидности *адаптивной* маршрутизации, при которой пути от отправителей к адресатам для нового трафика изменяются в ответ на перегрузки. Дело в том, что в результате изменения статистики входного трафика на отдельных участках сети могут возникать перегрузки. В этом случае алгоритм маршрутизации должен пытаться изменить маршруты и направить потоки в обход мест скопления пакетов.

Применяется много алгоритмов маршрутизации, различающихся степенью сложности и эффективности. Это объясняется отчасти историческими причинами, а отчасти разнообразием предназначений различных сетей. В этом разделе будут описаны алгоритмы маршрутизации трех сетей (ARPANET, TYMNET и SNA), которые достаточно полно отражают современный уровень развития сетей связи. Алгоритм маршрутизации сети Codex будет описан в разд. 5.8; этот алгоритм легче понять, предварительно ознакомившись в разд. 5.5—5.7 с методами оптимальной маршрутизации.

Маршрутизация в сети ARPANET

Алгоритм маршрутизации сети ARPANET, реализованный в 1969 г., сыграл важную историческую роль в развитии всего направления. С одной стороны, это был претенциозный, распре-

деленный, адаптивный алгоритм, стимулировавший немало исследований не только в области маршрутизации, но и вообще в области распределенных методов вычислений. С другой стороны, этот алгоритм имел несколько принципиальных недостатков, которые в конце концов были устранены в 1979 г., когда он был заменен на новую версию. К сожалению, к тому времени первоначальный алгоритм уже стал применяться в нескольких других сетях связи.

Обе версии алгоритма ARPANET основаны на понятии *кратчайшего пути* между двумя узлами. Каждой линии связи приписывается некоторое положительное число, называемое его *длиной*. Линия может иметь различную длину в зависимости от того, в каком направлении ведется передача. Каждый путь (т. е. последовательность линий) между двумя узлами имеет длину, равную сумме длин его линий (более подробно это описано в разд. 5.2). Алгоритм ARPANET стремится проложить маршрут пакета по пути наименьшей длины (или кратчайшему пути) между узлом-отправителем и узлом-адресатом данного пакета. Идея здесь состоит в том, что если длина каждой линии означает степень ее загруженности, то кратчайший путь будет состоять из небольшого числа слабонагруженных линий и поэтому этот путь будет более предпочтителен при выборе маршрута.

В обоих алгоритмах ARPANET длина каждой линии зависит от некоторым образом измеряемой нагрузки, поступающей в эту линию, и эта длина периодически обновляется. Таким образом, алгоритмы ARPANET являются адаптивными, и так как в сети ARPANET используются дейтаграммы, то два последовательных пакета одного и того же потока могут следовать разными маршрутами. Это имеет два нежелательных эффекта. Во-первых, пакеты могут прибывать в узлы-адресаты в другом порядке и поэтому после прибытия должны быть снова упорядочены. (Это может, например, произойти из-за того, что в протоколе передачи данных по линии в сети ARPANET используются восемь логических каналов, см. обсуждение в гл. 2.) Во-вторых, алгоритмы ARPANET имеют тенденцию к колебаниям. Это явление более подробно объясняется в подразд. 5.2.5, но главной причиной колебаний является то, что, выбирая маршруты, проходящие через какую-либо одну область сети, мы тем самым увеличиваем нагрузку и, следовательно, длины соответствующих линий. В результате при следующей корректировке маршрутов алгоритм выберет маршруты, проходящие через другие области сети. Это в свою очередь делает первоначальную область привлекательной при последующих корректировках маршрутов и, таким образом, возникают колебания. Эффект обратной связи между длиной линии и обновлениями маршрутов был главным недостатком первоначального алгоритма ARPANET, создававшим трудности

в течение нескольких лет, что привело в итоге к замене алгоритма. Последний алгоритм также имеет тенденцию к колебаниям, но не столь сильную, как первый (см. подразд. 5.2.5).

В первоначальном алгоритме ARPANET соседние узлы обмениваются оценками кратчайших расстояний до любого узла каждые 625 мс. Алгоритм обновления кратчайших расстояний от данного узла до каждого узла-адресата основывается на методе Беллмана — Форда, рассматриваемого в подразд. 5.2.3 и 5.2.4. Длина каждой линии равна числу пакетов, находящихся в очереди на передачу по этой линии в момент, когда производится корректировка. Таким образом, длина линии меняется очень быстро, отражая как случайные флуктуации потока, так и эффект обновления маршрутов. Для ослабления колебаний к длинам линий добавляются большие положительные константы. К сожалению, это уменьшает чувствительность алгоритма к перегрузкам.

В последней версии алгоритма ARPANET [181] длина каждой линии вычисляется путем запоминания величины задержки каждого пакета, прошедшего по этой линии. Длина каждой линии периодически каждые 10с обновляется и выбирается равной средней задержке пакетов, переданных по этой линии в течение предыдущего 10-секундного интервала. Задержка пакета на линии определяется как интервал времени между моментом поступления пакета во входной узел линии и моментом передачи его в выходной узел линии (время распространения сигнала также учитывается). Каждый узел следит за длинами всех уходящих от него линий и распространяет значения этих длин по всей сети по крайней мере не реже одного раза в течение 60с, используя для этого лавинный алгоритм (см. подразд. 5.3.1). В каждом узле после получения новых значений длин линий пересчитываются кратчайшие пути от этого узла до всех узлов-адресатов с использованием конечно-разностного варианта алгоритма Дейкстры (см. подразд. 5.2.3). Этот алгоритм является асинхронным в том смысле, что сообщения о новых значениях длин линий не передаются и не принимаются одновременно во всех узлах. Оказывается, что это приводит к улучшению свойств устойчивости алгоритма.

Заметим, что алгоритм не вычисляет маршрутные пути для других узлов, хотя информации, имеющейся в каждом узле (топология сети и длины всех линий), вполне достаточно для вычисления кратчайших путей от каждого узла-отправителя до каждого узла-адресата. В действительности маршрутные таблицы узлов сети ARPANET указывают только первую уходящую линию кратчайшего пути от данного узла до каждого узла-адресата (рис. 5.7). Когда узел получает новый пакет, он читает адрес пакета, смотрит в свою маршрутную таблицу и затем помещает пакет в очередь для передачи по соответствующей уходящей от

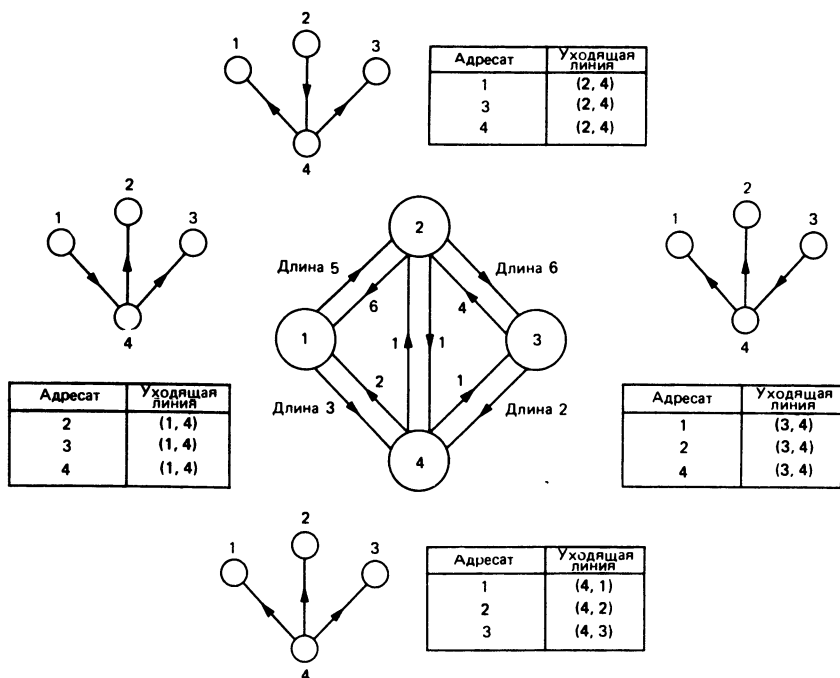


Рис. 5.7. Маршрутные таблицы в узлах в алгоритме ARPANET. Каждый узел вычисляет кратчайший путь от себя до каждого узла-адресата и вносит первую линию на этом пути в маршрутную таблицу.

него линии. Если бы во всех узлах, через которые пройдет пакет, составлялись маршрутные таблицы с использованием одной и той же информации о длинах линий, то легко заметить, что тогда этот пакет прошел бы по кратчайшему пути. В действительности возможна такая ситуация, когда пакет, путешествуя по сети, случайно пройдет по петле из-за постоянных изменений в маршрутных таблицах. Однако такие эффекты возникают редко и если уж пакет прошел один раз по петле, то маловероятно, что такое путешествие повторится.

Замена существующего алгоритма маршрутизации сети ARPANET намечена на 1987 г. Некоторые детали нового алгоритма еще не разработаны окончательно и могут быть изменены. Главной чертой нового алгоритма является использование нескольких путей для одной и той же пары отправитель — адресат. Кроме того, входной поток делится на несколько классов обслуживания и маршруты для каждого класса выбираются независимо от других. Пути, выбранные для какой-либо пары отправитель — адресат и определенного класса обслуживания, обновляются

довольно редко (каждые 5—15 минут). Это делается на основе вычисления кратчайших путей, а длина каждой линии зависит от класса обслуживания, характеристик линии и интенсивности суммарного потока, прошедшего по линии в течение предыдущего временного интервала. Маршруты также обновляются сразу после выхода из строя какого-либо узла или линии. Для каждого класса обслуживания и любой пары отправитель — адресат доля входного потока, проходящего по тому или иному пути, корректируется довольно часто (каждые 10—30 секунд) на основе информации о перегрузках, собранной в течение предыдущего временного интервала. В результате использования кратных путей новый алгоритм имеет большую пропускную способность по сравнению с существующим; в этом алгоритме могут использоваться методы оптимальной маршрутизации, рассматриваемые в разд. 5.5—5.7.

Маршрутизация в сети TUMNET

Алгоритм маршрутизации сети TUMNET, первоначально реализованный в 1971 г., основан на методе кратчайшего пути и является адаптивным, так же как и алгоритмы ARPANET. Однако способы реализации идеи адаптивного выбора кратчайшего пути сильно различаются в этих двух сетях.

В сети TUMNET алгоритм маршрутизации является централизованным и осуществляется в некотором специальном узле, называемом супервизором. В сети TUMNET используются виртуальные цепи и, следовательно, решение о выборе маршрутов принимается только в момент установления виртуальной цепи. Супервизор, получив запрос на установление виртуальной цепи, назначает кратчайший путь, соединяющий узел-отправитель с узлом-адресатом данной виртуальной цепи. Супервизор следит за текущими значениями длин всех линий и вычисляет все кратчайшие пути. Длина каждой линии зависит не только от нагрузки на этой линии, но и от других факторов, таких как тип линии и тип виртуальной цепи, для которой выбирается маршрут [246]. Хотя этот алгоритм можно классифицировать как адаптивный, однако у него нет такой тенденции к колебаниям, как у алгоритмов ARPANET. Это обусловлено в первую очередь тем, что используются виртуальные цепи, а не дейтаграммы (подробнее см. в подразд. 5.2.5).

Супервизор после того, как выбрал путь для данной виртуальной цепи, информирует все узлы на этом пути и необходимая информация вносится во все узловые маршрутные таблицы. Структура этих таблиц показана на рис. 5.8. Основным является то, что виртуальной цепи назначается номер канала (или порт) на каждой линии, по которой цепь проходит. В маршрутной

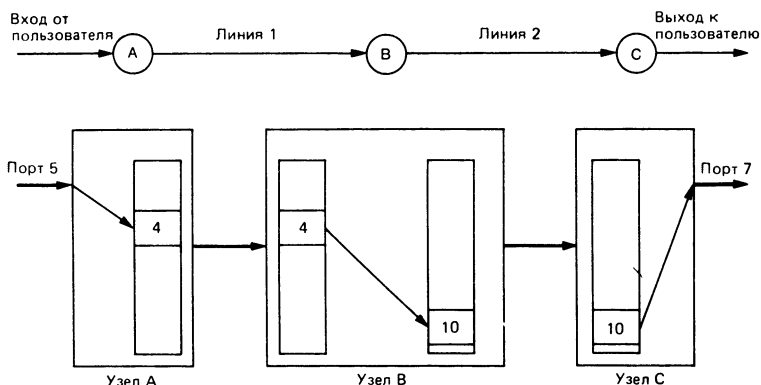


Рис. 5.8. Структура маршрутных таблиц сети TYMNET. Для виртуальной цепи на пути *ABC* маршрутная таблица в узле *A* ставит в соответствие входному порту 5 канал 4 на линии 1. В узле *B* входному каналу 4 ставится в соответствие входящий канал 10 на линии 2. В узле *C* входному каналу 10 ставится в соответствие порт 7.

таблице каждого узла для каждой виртуальной цепи установлено соответствие между номером канала на приходящей к узлу линии и номером канала на уходящей от узла линии. В первоначальном варианте алгоритма (называемом теперь TYMNET I) в супервизоре хранятся образцы маршрутных таблиц всех узлов и он непосредственно читает и обновляет эти таблицы. В существующем варианте (TYMNET II) узлы сами следят за состоянием своих таблиц. Супервизор устанавливает новую виртуальную цепь путем посылки пакета-«иголки» узлу-отправителю. Пакет-иглолка содержит маршрутную информацию и сам прокладывает путь по сети, устанавливая тем самым виртуальную цепь, а за ним, как нить за иголкой, следуют информационные пакеты пользователя. Маршрутная информация включает последовательность номеров узлов маршрутного пути и некоторые флаги, указывающие класс цепи. Когда иглолка поступает в очередной узел TYMNET II, ее содержимое читается. Если цепь оканчивается в этом узле, то она приспосабливается к порту, соединенному с соответствующим внешним пунктом. В противном случае проверяется следующий узел на маршруте. Если он неизвестен (вследствие недавнего выхода из строя линии или какой-либо другой ошибки), то цепь втягивается назад к своему узлу-отправителю. Если все в порядке, то маршрутные таблицы обновляются и иглолка переходит к следующему узлу.

Алгоритм TYMNET хорошо показал себя в работе в течение нескольких лет. Несколько его идей применены в сети Codex, которая имеет похожую структуру (см. разд. 5.8). **Алгоритм**

Codex, однако, является распределенным и более сложным. Слабым местом алгоритма TYMNET является его уязвимость к ошибкам и выходам из строя супервизора. Для повышения надежности в сети TYMNET используются узлы, дублирующие работу супервизора. В распределенном варианте, используемом в сети Codex, длины всех линий передаются в широковещательном режиме всем узлам (как в сети ARPANET) и каждый узел сам должен отвечать за маршрутизацию цепей, возникающих или заканчивающихся в этом узле.

Маршрутизация в архитектуре SNA

Маршрутизация в архитектуре SNA, разработанной фирмой IBM, несколько необычна, так как право выбора маршрута частично предоставляется самому пользователю. SNA устанавливает некоторые рамки, в пределах которых можно осуществить выбор среди ряда алгоритмов маршрутизации. Точнее говоря, для каждой пары отправитель — адресат SNA дает несколько путей, по которым может быть установлена виртуальная цепь. Правило выбора путей для виртуальных цепей должно удовлетворять некоторым ограничениям, но в остальном остается произвольным. Это упрощенное описание маршрутизации в SNA дано с использованием общей терминологии путей и виртуальных цепей, принятой в этой книге. В SNA используется другая терминология которую мы сейчас приведем.

Архитектура SNA не согласуется полностью с семиуровневой архитектурой OSI, используемой в качестве нашей основной модели. Уровень, который в архитектуре SNA ближе всего соответствует сетевому уровню в архитектуре OSI, называется уровнем управления путями; он обеспечивает установление виртуальной цепи для следующего более высокого уровня, называемого уровнем управления передачей. Уровень управления путями имеет три функции: управление группой передачи; управление явными маршрутами и управление виртуальными маршрутами. Первая функция ближе всего соответствует уровню управления линией передачи данных в модели OSI, а вторая и третья соответствуют функциям маршрутизации и управления потоками сетевого уровня в модели OSI.

Группой передачи в терминологии SNA называется набор из одной или нескольких физических линий связи между соседними узлами сети. Благодаря функции управления группы передачи эта группа представляется для более высоких уровней как одна физическая линия. Может быть несколько групп передачи, соединяющих пару узлов. В соответствии с протоколом группы передачи приходящие пакеты ставятся в очередь, посылаются по следующей имеющейся в распоряжении физической линии

и затем упорядочиваются, если в процессе передачи их порядок нарушается. Восстановление порядка происходит в каждом узле в отличие от сети ARPANET, в которой пакеты упорядочиваются только в узле-адресате.

Под явным маршрутом в терминологии SNA подразумевается то, что мы называем путем по подсети. Таким образом, явный маршрут это просто последовательность групп передачи, создающая физический путь от узла-отправителя к узлу-адресату. Каждой паре отправитель — адресат предоставляется несколько явных маршрутов (не более восьми в SNA 4.2), которые могут быть изменены только в результате супервизорного управления. Метод выбора набора явных маршрутов остается за администрацией сети. Каждый узел запоминает в своих маршрутных таблицах номер следующего узла для каждого явного маршрута и каждый пакет содержит номер своего явного маршрута. Поэтому, когда пакет прибывает в узел, следующий узел на его пути (если таковой имеется) определяется просто путем просмотра маршрутной таблицы.

Виртуальным маршрутом в терминологии SNA, по существу, является то, что мы называем виртуальной цепью. Разговор между парой пользователей (или сеанс) использует только один виртуальный маршрут. Однако один виртуальный маршрут может иметь несколько сеансов. Каждый виртуальный маршрут принадлежит некоторому приоритетному классу (или классу обслуживания). Любая группа передачи может обслуживать до трех приоритетных классов, в каждом из которых не более восьми виртуальных маршрутов, так что она обслуживает максимум 24 виртуальных маршрута. Когда пользователь через входной узел подсети делает запрос на сеанс, класс обслуживания определяется на основе того, какой из входных узлов пытается установить сеанс по одному из своих виртуальных маршрутов. Если все линии связи данной группы передачи выходят из строя, то все сеансы, использующие эту группу передачи, должны выбрать себе новый маршрут. Если ни одного другого маршрута найти не удастся, то сеанс обрывается. Когда устанавливаются новые группы передачи, все узлы уведомляются об этом посредством специальных управляющих пакетов и, таким образом, в каждом узле имеется копия топологии сети и известно, какие из явных маршрутов доступны, а какие — нет.

5.2. Сетевые алгоритмы и выбор кратчайшего пути

Методы маршрутизации используют ряд простых графовых алгоритмов. Рассмотрим, например, задачу нахождения кратчайшего пути, в которой известны длины всех линий сети и требуется найти путь, соединяющий два данных узла и имеющий минималь-

ную суммарную длину. Если длина линии некоторым образом отражает степень ее нагруженности, то поиск кратчайшего пути эквивалентен поиску наименее нагруженного пути между двумя узлами, а это имеет отношение к задаче маршрутизации (см. также предыдущее описание алгоритмов ARPANET и TYMNET). В этом разделе мы подробно рассмотрим как задачу выбора кратчайшего пути и ее роль в маршрутизации, так и другие близкие задачи, представляющие самостоятельный интерес. Начнем же мы с того, что введем некоторые основные графовые обозначения.

5.2.1. Неориентированные графы

Мы определяем *граф* $G = (\mathcal{N}, \mathcal{A})$ как конечное непустое множество *узлов* \mathcal{N} и некоторый набор \mathcal{A} пар различных узлов из \mathcal{N} . Каждая пара узлов в \mathcal{A} называется *дугой* ¹⁾. Несколько примеров графов показаны на рис. 5.9. Узлы часто называются вершинами, а дуги часто — ребрами, ветвями, звеньями или линиями. Главная цель формального определения $G = (\mathcal{N}, \mathcal{A})$ состоит в подчеркивании того, что местоположение узлов и конкретная форма дуг при изображении графа на бумаге совершенно несущественны. Если n_1 и n_2 являются узлами графа, а (n_1, n_2) — дугой, то говорят, что эта дуга *инцидентна* n_1 и n_2 . Некоторые авторы определяют графы так, что дугам разрешается иметь оба конца инцидентными одному и тому же узлу, но мы умышленно не допускаем такие петли. Мы также не допускаем кратные дуги между одной и той же парой узлов.

Переходом ²⁾ по графу G называется последовательность узлов (n_1, n_2, \dots, n_l) , таких, что все пары $(n_1, n_2), (n_2, n_3), \dots, (n_{l-1}, n_l)$ являются дугами графа G . Переход, в котором нет повторяющихся узлов, называется *путем*. Переход (n_1, \dots, n_l) , в котором $n_1 = n_l$,

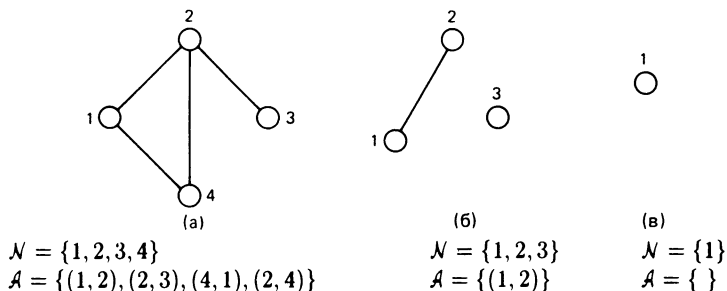


Рис. 5.9. Три примера графов с множеством узлов \mathcal{N} и множеством дуг \mathcal{A} .

¹⁾ Дугой в математической теории графов называется ориентированная дуга (см. 5.2.3). — *Прим. ред.*

²⁾ Переход в математической теории графов называется маршрутом. — *Прим. ред.*

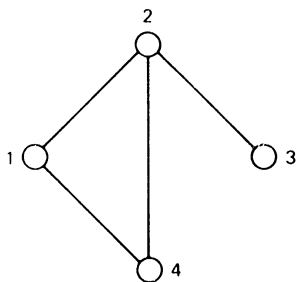


Рис. 5.10. Иллюстрация переходов, путей и циклов. Последовательности (1, 4, 2, 3), (1, 4, 2, 1), (1, 4, 2, 1, 4, 1), (2, 3, 2) и (2) являются переходами. Каждая из последовательностей (1, 4, 2, 3) и (2) является путем, а (1, 4, 2, 1) — циклом. Заметим, что (2) и (2, 3, 2) не рассматриваются как циклы.

$l > 3$ и нет повторяющихся узлов, кроме $n_1 = n_l$, называется *циклом*. Эти определения проиллюстрированы на рис. 5.10.

Будем говорить, что граф *связный*, если для каждого узла i существует путь ($i = n_1, n_2, \dots, n_l = j$) к любому другому узлу j . Заметим, что на рис. 5.9 графы (а) и (в) связные, а граф (б) не связный. Мы обнаруживаем отсутствие связности графа, если видим, что два подмножества узлов не соединены между собой дугами. Это наблюдение можно доказать и поэтому оно сформулировано в виде следующей леммы, доказательство которой остается читателю в качестве упражнения.

Лемма. Пусть G — связный граф, а S — любое непустое неполное подмножество узлов N . Тогда существует по крайней мере одна дуга (i, j) , такая, что $i \in S$ и $j \notin S$.

Будем говорить, что $G' = (N', A')$ является *подграфом* графа $G = (N, A)$, если G' сам является графом, $N' \subset N$ и $A' \subset A$. Например, последние три графа на рис. 5.11 являются подграфами первого.

Деревом называется связный граф, не имеющий циклов. *Остовным деревом* графа G называется подграф графа G , который является деревом и содержит все узлы графа G . Например, на рис. 5.11 подграфы (б) и (в) являются остовными деревьями графа (а). Следующий простой алгоритм строит остовное дерево произвольного графа $G = (N, A)$:

1. Пусть n — произвольный узел из N . Положим $N' = \{n\}$, $A' = \{\}$.

2. Если $N' = N$, то стоп ($G' = (N', A')$ — остовное дерево); в противном случае переход к шагу 3.

3. Пусть дуга $(i, j) \in A$, такая, что $i \in N'$, $j \in N - N'$. Обновляем A' и N' :

$$N' := N' \cup \{j\},$$

$$A' := A' \cup \{i, j\}.$$

Переход к шагу 2.

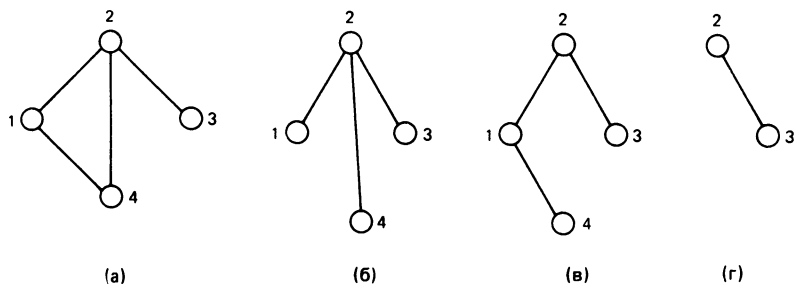


Рис. 5.11. Граф (а) и три подграфа (б), (в) и (г). Подграфы (б) и (в) являются основными деревьями.

Чтобы понять, почему этот алгоритм работает, заметим, что переход к шагу 3 возможен только тогда, когда \mathcal{N}' не содержит все множество \mathcal{N}^o , а в этом случае предыдущая лемма гарантирует существование дуги (i, j) . Докажем по индукции, что после каждого выполнения шага 3 граф $G' = (\mathcal{N}', \mathcal{A}')$ является деревом. Первоначально $G' = (\{n\}, \{ \})$ тривиально является деревом. Предположим теперь, что перед очередным обновлением на шаге 3 граф $G' = (\mathcal{N}', \mathcal{A}')$ является деревом. Следовательно, существует путь между любой парой узлов из \mathcal{N}' , использующий дуги из \mathcal{A}' . После добавления узла j и дуги (i, j) каждый узел имеет путь до j , состоящий из пути до i и дуги (i, j) , и аналогично j имеет путь к любому другому узлу. Наконец, узел j не может принадлежать никакому циклу, так как (i, j) является единственной дугой из G' , инцидентной j , а циклов, не содержащих j , нет по предположению индукции. На рис. 5.12 показан граф G' после каждого выполнения шага 3 для одного из возможных вариантов выбора дуг.

Заметим, что алгоритм начинает с подграфа, состоящего из одного узла и нуля дуг, и добавляет при каждом выполнении шага 3 один узел и одну дугу. Это означает, что остовное дерево

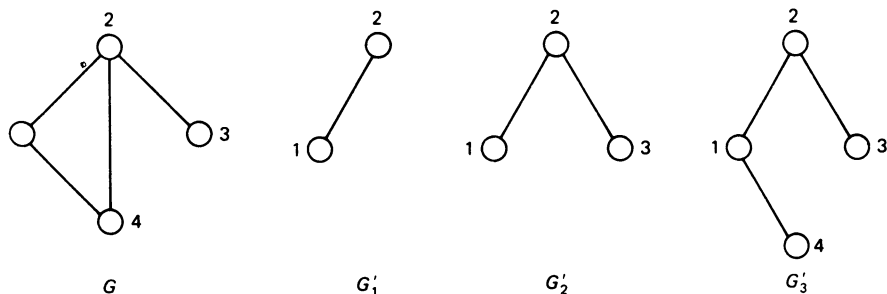


Рис. 5.12. Алгоритм построения остовного дерева данного графа G .

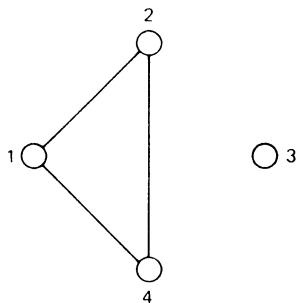


Рис. 5.13. Граф с $A = N - 1$, который содержит цикл, но является несвязным.

G' , построенное алгоритмом, всегда имеет N узлов и $(N - 1)$ дугу, где N — число узлов в графе G . Так как G' есть подграф графа G , то число дуг A в G должно удовлетворять неравенству $A \geq N - 1$, которое справедливо для любого связного графа G . Предположим теперь, что $A = N - 1$. Это означает, что алгоритм использует все дуги графа G при построении остовного дерева G' , так что $G' = G$, и, следовательно, сам граф G должен быть деревом. И наконец, если $A \geq N$, то G содержит по крайней мере одну дугу (i, j) , не вошедшую в остовное дерево G' , построенное алгоритмом.

Пусть (j, \dots, i) — путь от j к i в графе G' ; тогда видно, что (i, j, \dots, i) является циклом в G и, следовательно, граф G не может быть деревом. Следующее утверждение суммирует эти результаты:

Утверждение. Пусть G — связный граф с N узлами и A дугами. Тогда

1. G содержит остовное дерево.
2. $A \geq N - 1$.
3. G является деревом тогда и только тогда, когда $A = N - 1$.

На рис. 5.13 показано, почему связность является необходимым условием выполнения последней части утверждения.

5.2.2. Остовные деревья минимального веса

Рассмотрим задачу, представляющую интерес для сетей передачи данных; эта задача приводит к распределенному алгоритмическому решению. Пусть задан связный граф $G = (\mathcal{N}, \mathcal{A})$ с N узлами и A дугами и весами w_{ij} для каждой дуги $(i, j) \in \mathcal{A}$. Задача состоит в построении остовного дерева с минимальной суммой весов дуг. Такое дерево называется *остовным деревом минимального веса* (МОД для краткости). МОД может оказаться полезным при распространении сообщений, адресованных всем узлам сети, например управляющих сообщений от узла-супервизора. В таком контексте вес дуги означает стоимость передачи сообщения по этой дуге в любом направлении, а вес всего остовного дерева — стоимость передачи сообщения всем узлам остовного дерева.

Любое поддерево (т. е. подграф, являющийся деревом) МОДа будет называться *фрагментом*. Заметим, что узел может рас-

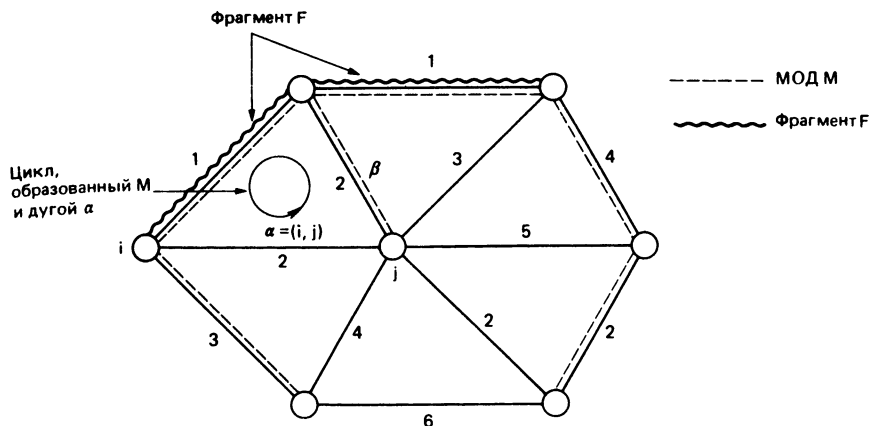


Рис. 5.14. Доказательство утверждения 1. Цифры рядом с дугами представляют собой веса дуг. F — фрагмент, который является поддеревом МОД M . Пусть α — уходящая от F дуга, имеющая минимальный вес и не принадлежащая M . Пусть $\beta \neq \alpha$ — уходящая от F дуга, которая одновременно принадлежит и M и циклу, образованному α и M . Удаление β из M и добавление α на ее место образует другое МОД M' . Если к F добавить α , то образуется фрагмент.

смагиваться как фрагмент. Дуга, имеющая один узел во фрагменте, а другой узел — вне этого фрагмента, называется дугой, уходящей от фрагмента. Следующее утверждение имеет большое значение для МОД-алгоритмов.

Утверждение 1. Пусть F — некоторый заданный фрагмент, а $\alpha = (i, j)$ — дуга минимального веса, уходящая от фрагмента F , причем узел j не входит в F . Тогда если к F добавить дугу α и узел j , то получится фрагмент.

Доказательство. Через M обозначим МОД, для которого F является поддеревом. Если дуга α принадлежит M , то все в порядке; предположим теперь обратное. Тогда существует цикл, сформированный дугой α и дугами из M . Так как узел j не принадлежит F , то должна найтись дуга $\beta \neq \alpha$, принадлежащая одновременно циклу и M и которая является уходящей от F (рис. 5.14). Если удалить β и добавить α , то в результате получим подграф M' с $(N - 1)$ дугами, в котором не будет циклов, и поэтому он должен быть остовным деревом. Так как вес α не больше веса β , то M' должен быть МОД, и поэтому после добавления к F дуги α и узла j получится фрагмент, что и требовалось доказать.

Утверждение 1 можно использовать в качестве основы для алгоритмов построения МОД. Идея состоит в том, чтобы начать

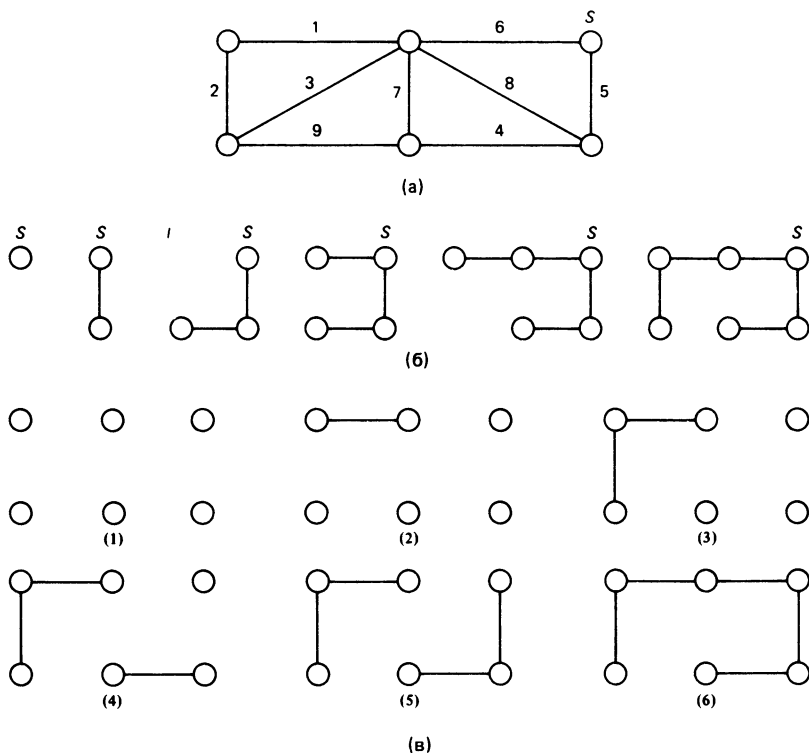


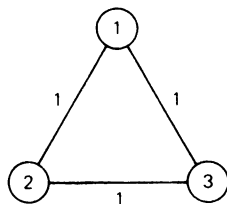
Рис. 5.15. Построение остова минимального веса.

a — граф с указанными весами дуг; *b* — последовательные итерации алгоритма Прим — Дijkstra. Исходный фрагмент состоит из единственного узла, помеченного *S*. К этому фрагменту последовательно добавляются уходящие дуги минимального веса. *в* — последовательные итерации алгоритма Крускала. Алгоритм начинает со всех узлов в качестве фрагментов. На каждой итерации добавляется дуга, имеющая минимальный вес среди всех дуг, уходящих от какого-либо одного из фрагментов.

с одного или нескольких разьединенных фрагментов и затем постепенно увеличивать или соединять их путем добавления уходящих дуг минимального веса. Алгоритм Прим — Дijkstra начинает с одного произвольного узла, выбранного в качестве фрагмента, и затем увеличивает этот фрагмент путем последовательного добавления уходящих дуг минимального веса. Алгоритм Крускала начинает одновременно со всех узлов, которые рассматриваются в качестве одноузловых фрагментов, и затем последовательно соединяет какие-либо два фрагмента дугой, имеющей минимальный вес среди дуг, добавление которых не создает цикла (рис. 5.15). Оба алгоритма заканчивают работу после $N - 1$ итерации.

Алгоритм Крускала создает одновременно несколько фрагментов, которые в итоге объединяются в МОД, однако на каждом

Рис. 5.16. Пример, когда распределенный алгоритм построения МОД в графе, в котором веса некоторых дуг могут совпадать, приводит к неправильному результату. Узлы 1, 2 и 3 могут одновременно добавить к своим фрагментам дуги (1, 2), (2, 3) и (3, 1) соответственно, в результате чего образуется цикл.



шаге к имеющемуся набору фрагментов добавляется только одна дуга. Утверждение 1 подсказывает возможность одновременного добавления к каждому фрагменту уходящих дуг минимального веса методом распределенного алгоритма. Это можно делать, если МОД единственно.

Распределенный алгоритм построения МОД в графе с единственным МОД работает следующим образом. Он начинает с некоторого набора фрагментов (ими могут быть, например, узлы). Каждый фрагмент отыскивает уходящую от него дугу минимального веса, добавляет ее к себе и информирует об этом фрагмент, лежащий на другом конце этой дуги. Можно доказать, что если дуга, соединяющая на очередном шаге два фрагмента, действительно является уходящей дугой минимального веса для одного из фрагментов, то после каждого шага алгоритма результатом будет набор действительно фрагментов МОД и циклы никогда не возникнут. Более того, процесс присоединения новых дуг будет продолжаться до тех пор, пока не останется ни одной уходящей дуги и будет только один фрагмент, по необходимости МОД. Поэтому алгоритм не может остановиться, не найдя МОД. Вообще говоря, для того чтобы алгоритм работал правильно, не обязательно, чтобы процедура присоединения дуг была синхронной для всех фрагментов. По существу, нужна только схема, согласно которой узлы и дуги фрагмента определяют уходящую дугу минимального веса. Это можно сделать разными способами, но желательно создать такую схему, которая бы выполняла это с минимальными ресурсами связи. Более детально этот вопрос рассмотрен в работе [96]. В [122] описан случай, когда в разных направлениях дуга имеет разный вес.

Для того чтобы увидеть, какие трудности могут возникнуть в случае, когда МОД не единственно, рассмотрим треугольную сеть на рис. 5.16, в которой все дуговые веса равны единице. Если алгоритм начнет с трех узлов, выбранных в качестве фрагментов, и позволит каждому фрагменту добавлять к себе любую уходящую дугу минимального веса, то будет возможность одновременного добавления дуг (1, 2), (2, 3) и (3, 1) к узлам 1, 2 и 3 соответственно, что приведет к циклу.

Следующее утверждение подсказывает выход из ситуации, когда МОД не единственно.

Утверждение 2. Если все дуговые веса различны, то существует только одно МОД.

Доказательство. Предположим, что существуют два различных МОД, обозначенных через M и M' . Пусть α — дуга, имеющая минимальный вес среди дуг, принадлежащих либо M , либо M' , но не обоим сразу. Будем, например, считать, что α принадлежит M . Рассмотрим граф, образованный объединением M' и α . Он должен содержать цикл (так как $\alpha \notin M'$), и по крайней мере одна дуга этого цикла, например β , не принадлежит M (иначе M содержало бы цикл). Так как вес α строго меньше веса β , то удаление β из M' и добавление на его место α даст в результате остовное дерево, вес которого строго меньше веса M' . Приходим к противоречию, так как M' по предположению является МОД.

Рассмотрим теперь задачу построения МОД в случае, когда веса некоторых дуг могут совпадать. Неопределенность в выборе дуг одинакового веса можно устранить, используя номера узлов на их концах, т. е. в случае, когда веса дуг совпадают, предпочтение отдается дуге, уходящей от узла с меньшим номером, а если дуги уходят от одного и того же узла, то предпочтение отдается дуге, приходящей к узлу с меньшим номером. Таким образом, применяя, если это необходимо, только что описанную схему, можно считать без потери общности, что все веса дуг различны и МОД единственно.

5.2.3. Алгоритм отыскания кратчайшего пути

В дальнейшем нас будет интересовать также график, проходящий по дугам сети. Это заставляет различать направление потока и, следовательно, дать ориентацию самим дугам.

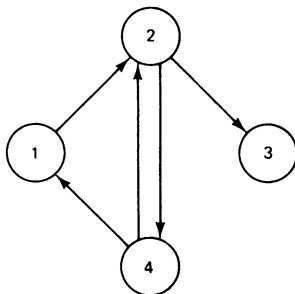
Ориентированный граф или *диграф* $G = (\mathcal{N}, \mathcal{A})$ — это конечное непустое множество узлов \mathcal{N} и набор \mathcal{A} упорядоченных пар различных узлов из \mathcal{N} ; каждая упорядоченная пара узлов из \mathcal{A} называется *ориентированной дугой* (или просто дугой). Графически диграф изображается, так же как и граф, но дуга представляется в виде стрелки, идущей от первого узла упорядоченной пары ко второму узлу (рис. 5.17). Заметим, что на рис. 5.17 дуги $(2, 4)$ и $(4, 2)$ различны.

Каждому диграфу $G = (\mathcal{N}, \mathcal{A})$ соответствует ассоциированный (неориентированный) граф $G' = (\mathcal{N}', \mathcal{A}')$, где $\mathcal{N}' = \mathcal{N}$ и $(i, j) \in \mathcal{A}'$, если либо $(i, j) \in \mathcal{A}$, либо $(j, i) \in \mathcal{A}$, либо одновременно и то и другое. Будем говорить, что (n_1, n_2, \dots, n_l) — переход, путь или цикл в диграфе, если это переход, путь или цикл в ассоциированном графе. Кроме того, (n_1, n_2, \dots, n_l) называется *ориентированным переходом* в диграфе G , если (n_i, n_{i+1}) является ориентированной дугой в G для всех $1 \leq i \leq l-1$. *Ориентированный*

Рис. 5.17. Изображение ориентированного графа.

$$\mathcal{N} = \{1, 2, 3, 4\},$$

$$\mathcal{A} = \{(1, 2), (2, 3), (2, 4), (4, 2), (4, 1)\}$$



путь — это ориентированный переход с неповторяющимися узлами, а *ориентированный цикл* — это ориентированный переход (n_1, \dots, n_l) с неповторяющимися узлами, такой, что $n_1 = n_l$ и $l > 2$. Заметим, что (n_1, n_2, n_1) является ориентированным циклом, если (n_1, n_2) и (n_2, n_1) являются ориентированными дугами, но (n_1, n_2, n_1) не может быть неориентированным циклом, если (n_1, n_2) — неориентированная дуга.

Диграф называется *сильно связным*, если для любой пары узлов i и j существует ориентированный путь $(i = n_1, n_2, \dots, n_l = j)$ от i к j . Диграф является *связным*, если связным является ассоциированный граф. На рис. 5.18 первый граф является связным, но не сильно связным, так как нет ориентированного пути от 3 к 2. Второй граф является сильно связным.

Рассмотрим ориентированный граф $G = (\mathcal{N}, \mathcal{A})$ с числом узлов N и числом дуг A , в котором каждой дуге (i, j) приписано некоторое действительное число d_{ij} , называемое иногда длиной или расстоянием дуги. *Длина* любого ориентированного пути $p = (i, j, k, \dots, l, m)$ определяется как $d_{ij} + d_{jk} + \dots + d_{lm}$. Длина ориентированного перехода или цикла определяется аналогично. Для любых двух узлов i и m графа задача кратчайшего пути состоит в отыскании такого пути от i к m , который бы имел минимальную длину (т. е. был кратчайшим).

Задача кратчайшего пути возникает в чрезвычайно большом числе приложений. Если d_{ij} — стоимость использования данной линии (i, j) в сети передачи данных, то кратчайший путь от i к m будет маршрутом, по которому данные будут передаваться с наименьшими затратами. Таким образом, если стоимость использования линии равна средней задержке пакета при прохождении по этой линии, то маршрут с наименьшими затратами будет маршрутом с наименьшей задержкой. К сожалению, в сетях передачи данных средняя задержка в линии зависит от интенсивности нагрузки на этой линии, которая в свою очередь зависит от тех маршрутов, которые выбрал алгоритм маршрутизации. Из-за этого эффекта обратной связи задача маршрутизации на-

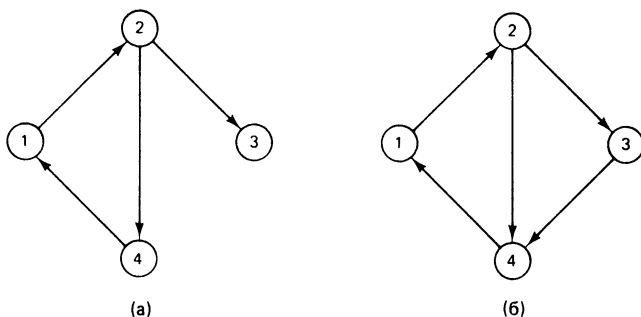


Рис. 5.18. Оба диграфа a и b — связные; b является сильно связным, но a не является сильно связным, так как нет ориентированного пути от 3 к 2.

много сложнее задачи выбора кратчайшего пути, однако задача отыскания кратчайшего пути все же является составной частью задачи маршрутизации во всех постановках, которые будут рассматриваться. Приведем другой пример. Если p_{ij} — вероятность того, что дуга (i, j) находится в сети в рабочем состоянии и состояние каждой дуги не зависит от состояния остальных дуг, то поиск кратчайшего пути между i и m в сети с дугowymi расстояниями, равными $-\ln p_{ij}$, эквивалентен поиску наиболее надежного пути от i до m .

Задача отыскания кратчайшего пути возникает также в сетевом планировании, используемом руководством предприятий при выполнении сложных проектов. Узлы сети соответствуют этапам (подзадачам), а дуга от этапа i к этапу j указывает на то, что выполнение этапа j зависит от выполнения этапа i . Если t_{ij} — время, необходимое для выполнения этапа j , после того как выполнен этап i , то в качестве расстояния для (i, j) берется $d_{ij} = -t_{ij}$. Кратчайший путь от начала выполнения всего проекта до его завершения требует наибольшего времени, и кратчайший путь указывает те критические этапы, время выполнения которых, по существу, определяет время выполнения всего проекта. Еще одним примером являются некоторые задачи динамического программирования, которые можно рассматривать как задачи отыскания кратчайшего пути [28]. И наконец, многие более сложные задачи теории графов требуют решения задач отыскания кратчайшего пути.

Ниже будут рассмотрены три стандартных алгоритма решения задачи отыскания кратчайшего пути — алгоритм Беллмана — Форда, алгоритм Дijkstra и алгоритм Флойда — Уоршела. Первые два алгоритма находят кратчайшие пути от данного узла-источника ко всем другим узлам (или, эквивалентно, от всех

узлов к данному узлу-адресату), а третий алгоритм находит кратчайшие пути от всех узлов ко всем другим узлам. Оказывается, если необходимо найти только один кратчайший путь от одного узла i до другого узла j , то неизвестно, можно ли это сделать без того, чтобы, по существу, не отыскать кратчайшие пути от всех узлов k j или, наоборот, от i ко всем другим узлам. Далее будут рассмотрены распределенные алгоритмы решения задачи отыскания кратчайшего пути, но их легче понять, если сначала ознакомиться с централизованными алгоритмами решения этой задачи.

Алгоритм Беллмана—Форда

Предположим, что узел 1 является узлом-источником и требуется найти длины кратчайших путей от узла 1 до каждого другого узла графа. Для этого алгоритма дуговые расстояния могут быть как положительными, так и отрицательными, но мы предположим, что нет циклов отрицательной длины (это предположение будет обсуждаться позднее более подробно). Для упрощения обозначений положим $\alpha_{ij} = \infty$, если в графе отсутствует дуга (i, j) . Основная идея алгоритма Беллмана—Форда состоит в том, чтобы сначала найти длины кратчайших путей, при условии, что пути содержат не более одной дуги, затем длины кратчайших путей при условии, что пути содержат не более двух дуг и т. д. Кратчайший путь при условии, что путь содержит не более h дуг, в дальнейшем будет называться кратчайшим ($\leq h$) путем.

Пусть $D_i^{(h)}$ — длина кратчайшего ($\leq h$) пути от узла 1 до узла i . Будем считать, что $D_i^{(h)} = 0$ для всех h . Алгоритм Беллмана—Форда состоит в следующем.

Вначале

$$D_i^{(0)} = \infty \quad \text{для всех } i \neq 1. \quad (5.1)$$

При каждом последующем $h \geq 0$

$$D_i^{(h+1)} = \min_j [D_j^{(h)} + d_{ji}] \quad \text{для всех } i \neq 1. \quad (5.2)$$

Рис. 5.19 иллюстрирует работу алгоритма. Для доказательства того, что этот алгоритм приводит к правильному решению, заметим сначала, что (5.1) и (5.2) дают $D_i^{(1)} = d_{1i}$ для всех $i \neq 1$ и они действительно являются длинами кратчайших (≤ 1) путей. Далее проведем доказательство индукцией по h , предполагая для данного h , что $D_i^{(h)}$ являются длинами кратчайших ($\leq h$) путей для всех $i \neq 1$, и доказывая, что равенство (5.2) дает длину кратчайшего ($\leq h + 1$) пути от узла 1 до каждого узла $i \neq 1$. Сначала покажем, что левая часть (5.2) больше или равна правой части, а затем докажем противоположное неравенство. Предположим, что $(1, \dots, m, k, i)$ — кратчайший ($\leq h + 1$) путь от 1

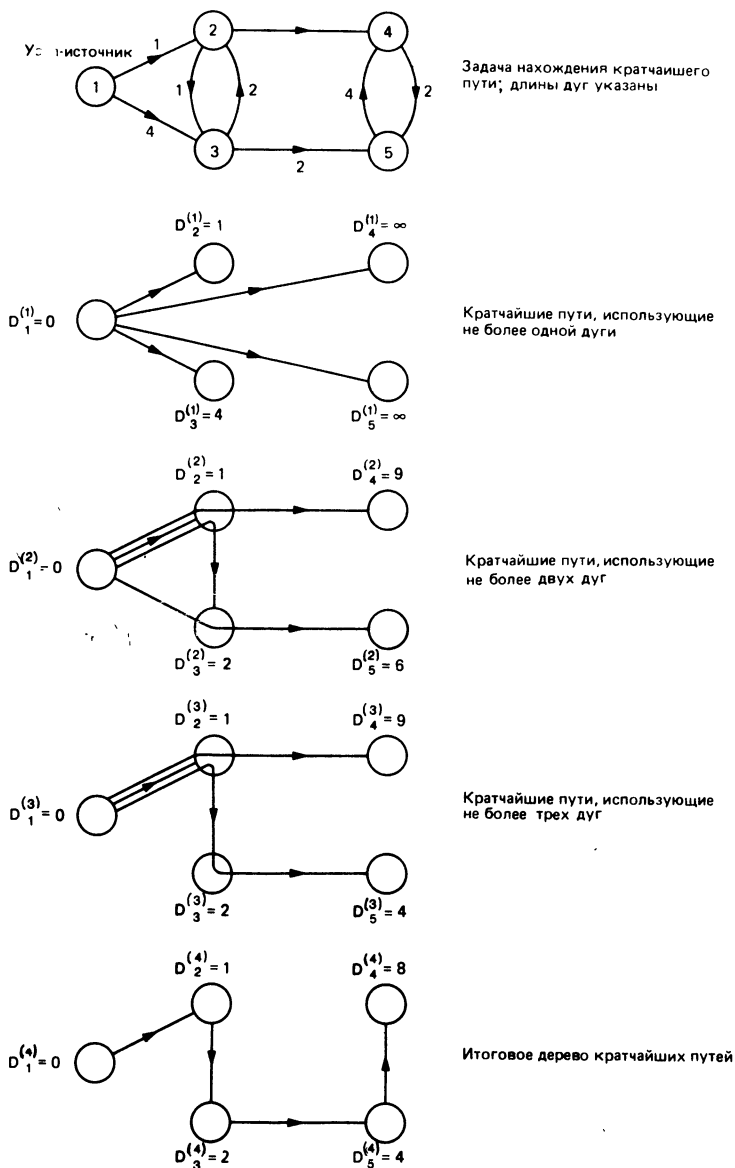


Рис. 5.19. Последовательные итерации в методе Беллмана—Форда.

до i . Тогда его длина равна длине пути $(1, \dots, m, k)$ плюс d_{ki} . Так как $(1, \dots, m, k)$ содержит не более h дуг, то

$$D_i^{(h+1)} \geq D_k^{(h)} + d_{ki} \geq \min_j [D_j^{(h)} + d_{ji}].$$

Для доказательства обратного неравенства предположим, что минимум в правой части (5.2) достигается при $j = k$ и что $(1, \dots, m, k)$ является кратчайшим ($\leq h$) путем, длина которого по предположению равна $D_k^{(h)}$. Тогда длина пути $(1, \dots, m, k, i)$ равна правой части (5.2). Поэтому если $(1, \dots, m, k, i)$ — некоторый путь, то

$$D_i^{(h+1)} \leq D_k^{(h)} + d_{ki} = \min_j [D_j^{(h)} + d_{ji}].$$

И наконец, если $(1, \dots, m, k, i)$ не является путем (т. е. если он содержит цикл), то i должен лежать на пути $(1, \dots, m, k)$, а так как все циклы имеют неотрицательную длину по предположению, то, следовательно, предыдущее неравенство сохраняется, что и доказывает окончательно равенство (5.2).

Путь может содержать не более $N - 1$ дуг (где N — число узлов) и поэтому, если нет циклов отрицательной длины, то $D_i^{(N-1)}$ будет длиной кратчайшего пути от 1 до i . Также легко заметить, что если $D_i^{(h+1)} = D_i^{(h)}$ для всех i и определенного h , то последующие итерации с большими h не изменят длин кратчайших путей и $D_i^{(h)}$ будет длиной кратчайшего пути для каждого i .

Число итераций алгоритма в худшем случае равно $N - 1$, каждая итерация должна быть проведена для $N - 1$ узла, а для каждого узла минимизация осуществляется самое большее по $N - 1$ переменной. Таким образом, в худшем случае объем вычислений растет как N^3 , что записывается в виде $O(N^3)$. Обозначение $O(p(N))$, где $p(N)$ — многочлен от N , используется для зависящего от N числа, которое меньше $cp(N)$ для всех N , где c — некоторая константа, не зависящая от N . Действительно, более тщательный подсчет показывает, что объем вычислений равен $O(mA)$, где A — число дуг, а m — число итераций (m также равно максимальному числу дуг, содержащихся в кратчайшем пути).

Теперь более внимательно исследуем влияние циклов отрицательной длины. Рассмотрим граф, изображенный на рис. 5.20. Длина кратчайшего перехода от узла 1 к узлу 2 среди всех переходов, содержащих не более трех дуг, равна -1 и соответствует переходу $(1, 2, 3, 2)$; длина кратчайшего пути, однако, равна $+1$ для пути $(1, 2)$. В действительности алгоритм Беллмана — Форда отыскивает длину кратчайшего ($\leq h + 1$) перехода от 1 к i при условии, что узел 1 не повторяется в переходе. Если существуют циклы отрицательной длины, не содержащие узла 1, то длины более длинных переходов становятся все меньше и меньше и алго-

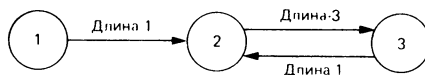


Рис. 5.20. Граф с отрицательным циклом. Длина кратчайшего пути от 1 до 2 равна 1. Алгоритм Беллмана—Форда дает $D_2^{(2)} = 1$ и $D_2^{(3)} = -1$, что указывает на существование отрицательного цикла.

ритм Беллмана — Форда никогда не остановится. Существуют ли отрицательные циклы (не содержащие узла 1), можно определить, сравнивая $D_i^{(N)}$ и $D_i^{(N-1)}$ для каждого i . Если для всех i имеет место равенство, то отсюда следует, что $D_i^{(h)} = D_i^{(N-1)}$ для всех i и $h \geq N$, и, следовательно, длины кратчайших путей найдены и отрицательных циклов нет (кроме, может быть, циклов, содержащих узел 1). Если для какого-то i имеет место неравенство, то это означает, что существует переход от 1 к i с N дугами, проходящий лишь однажды через узел 1 и имеющий длину, меньше чем любой переход от 1 к i с числом дуг, меньшим или равным $(N - 1)$. Поэтому этот переход должен содержать цикл отрицательной длины. В примере на рис. 5.20 можно проверить, что $D_2^{(2)} = 1$, а $D_2^{(3)} = -1$, и это указывает на существование цикла отрицательной длины.

Пусть D_i — длина кратчайшего пути от узла 1 к узлу i и $D_1 = 0$. Если нет циклов отрицательной длины, то (5.2) сойдется к истинному значению на шаге $h = N - 1$ и тогда его можно будет записать в виде

$$D_i = \min_j [D_j + d_{ji}] \quad \text{для всех } i \neq 1, \quad (5.3a)$$

$$D_1 = 0. \quad (5.3b)$$

Это уравнение Беллмана; оно означает, что длина кратчайшего пути от узла 1 к i равна сумме длины пути к узлу, предшествующему узлу i (на кратчайшем пути) и расстояния на последней дуге пути. Из решения этого уравнения (которое можно получить, используя алгоритм Беллмана—Форда) легко найти кратчайшие пути (а не длины кратчайших путей), учитывая, что все циклы, не содержащие узел 1, имеют положительную длину. Чтобы это сделать, нужно отобрать для каждого $i \neq 1$ по одной дуге (j, i) , на которой достигается минимум в (5.3a), и рассмотреть подграф, состоящий из отобранных $(N - 1)$ дуг. Чтобы найти кратчайший путь к произвольному узлу i , следует начать с узла i и идти в обратном направлении по соответствующим дугам получившегося подграфа до тех пор, пока не придем в узел 1. Заметим, что при

этом ни один узел не будет посещен дважды до прихода в узел 1, так как получившийся в противном случае цикл имел бы (на основании уравнения (5.3)) нулевую длину. Поскольку рассматриваемый подграф соединяет каждый узел с узлом 1 и имеет $(N-1)$ дугу, то он должен быть остовным деревом. Назовем этот подграф *остовным деревом кратчайших путей* и заметим, что оно имеет особую структуру; у него имеется корень (узел 1) и каждая дуга дерева ориентирована в обратном от корня направлении. В задаче 5.7 указывается способ построения такого же дерева в случае, когда имеются циклы нулевой длины. В задаче 5.4 выясняется разница между остовным деревом кратчайших путей и остовным деревом минимального веса.

Используя предыдущую конструкцию, можно доказать, что *если нет циклов нулевой (или отрицательной) длины, то уравнение Беллмана (5.3) имеет единственное решение*. Действительно, предположим, что \tilde{D}_i , $i = 1, \dots, N$, является решением (5.3) с $\tilde{D}_1 = 0$. Тогда, используя предыдущую конструкцию, можно показать, что D_i является длиной соответствующего пути от узла 1 до узла i . Поэтому $\tilde{D}_i \geq D_i$, где D_i — истинное минимальное расстояние для всех i . Чтобы доказать неравенство в обратную сторону, рассмотрим алгоритм Беллмана—Форда с двумя разными начальными условиями. Первым начальным условием является $D_i^{(0)} = \infty$ для $i \neq 1$ и $D_1^{(0)} = 0$. Для этого начального условия истинные минимальные расстояния D_i получаются после $N-1$ итерации. Вторым начальным условием является $D_i^{(0)} = \tilde{D}_i$ для всех i ; в этом случае \tilde{D}_i будет получаться после каждой итерации (так как \tilde{D}_i является решением уравнения Беллмана). Так как второе начальное условие меньше или равно первому для любого i , то из (5.2) следует, что $\tilde{D}_i \leq D_i$ для всех i . Следовательно, $\tilde{D}_i = D_i$ и только решения уравнения Беллмана могут быть истинными кратчайшими расстояниями D_i . Следовательно, легко понять, что *если есть циклы нулевой длины, не содержащие узел 1, то уравнение Беллмана всегда имеет неединственное решение* (хотя и в этом случае алгоритм Беллмана—Форда сходится к истинным кратчайшим расстояниям). Этот факт рассматривается в задаче 5.8.

Отчасти популярность алгоритма Беллмана—Форда объясняется тем, что в случае, когда длины всех дуг положительны, начальные условия $D_i^{(0)}$ для $i \neq 1$ могут быть любыми неотрицательными числами и итерации (5.2) могут выполняться параллельно для разных узлов, по существу, в произвольном порядке, что имеет большое значение для приложений с распределенными алгоритмами. Анализ алгоритмов такого типа будет дан в следующем разделе.

Алгоритм Дijkstra

Этот алгоритм требует, чтобы длины всех дуг были положительны (к счастью, это выполняется в сетях передачи данных). Объем вычислений в худшем случае для этого алгоритма значительно меньше, чем у алгоритма Беллмана—Форда. Основная идея алгоритма состоит в том, чтобы отыскивать кратчайшие пути в порядке возрастания длины пути. Кратчайшим среди всех кратчайших путей от узла 1 является путь, состоящий из одной дуги, соединяющей узел 1 с ближайшим соседним узлом, так как любой путь, состоящий из нескольких дуг, будет всегда длиннее длины первой дуги вследствие предположения о положительности всех дуговых длин. Следующим кратчайшим среди кратчайших путей должен быть либо путь из одной дуги к следующему ближайшему соседу узла 1, либо кратчайший путь из двух дуг, проходящий через узел, выбранный на первом шаге, и т. д. Чтобы формально описать эту процедуру в виде алгоритма, будем считать, что каждый узел i имеет метку D_i , означающую оценку длины кратчайшего пути от узла 1. Когда оценка становится неизменной, будем считать, что узел *окончательно помечен*, и множество окончательно помеченных узлов обозначим через P . Узел, который будет добавлен на очередном шаге к P , является ближайшим к узлу 1 среди всех узлов, еще не вошедших в P .

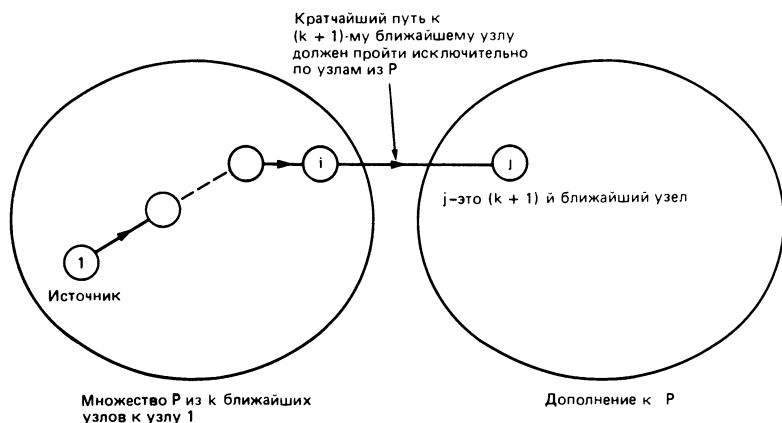
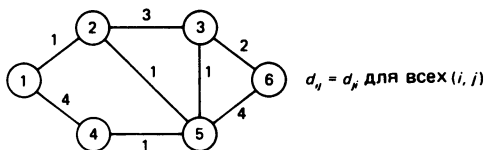
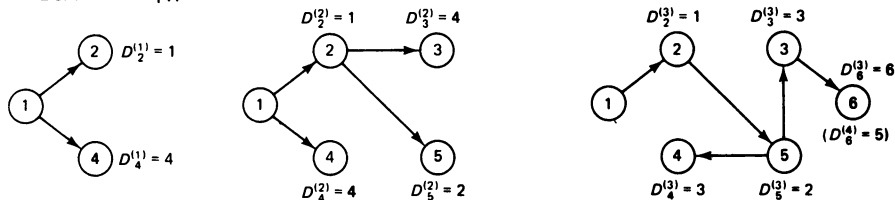


Рис. 5.21. Основная идея алгоритма Дijkstra. На k -м шаге имеется множество P из k ближайших узлов к узлу 1, D_i — кратчайшее расстояние от узла 1 до каждого узла i из P . Среди всех путей, соединяющих узел 1 с каким-либо узлом не из P , самый кратчайший путь должен пройти по узлам из P (так как $d_{ij} > 0$). Поэтому $(k+1)$ -й ближайший узел и соответствующее кратчайшее расстояние получаются минимизацией по $j \notin P$ величины $\min_{i \in P} \{D_i + d_{ij}\}$. Эти вычисления можно провести эффективно, как обсуждается в тексте, в результате чего вычислительная сложность будет порядка $O(N^3)$.



Беллман-Форд



Дейкстра

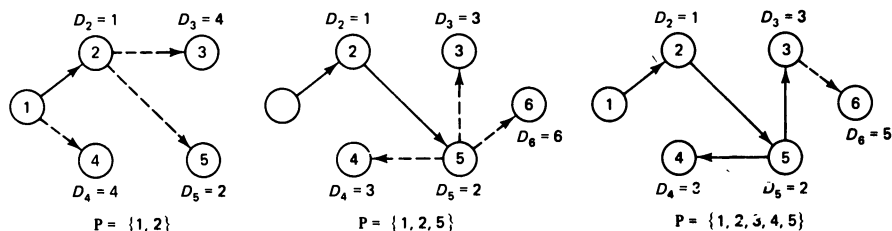


Рис. 5.22. Пример использования алгоритмов Беллмана—Форда и Дейкстра.

Рис. 5.21 иллюстрирует главную идею. Формально алгоритм работает следующим образом.

Сначала $P = \{1\}$, $D_1 = 0$ и $D_j = d_{1j}$ для $j \neq 1$.

Шаг 1 (поиск следующего ближайшего узла). Найти $i \notin P$, такой, что

$$D_i = \min_{i \notin P} D_j.$$

Положить $P := P \cup \{i\}$. Если P содержит все узлы, то на этом работа алгоритма заканчивается.

Шаг 2 (обновление меток.) Для всех $j \notin P$ положить

$$D_j := \min [D_j, D_i + d_{ij}].$$

Перейти к шагу 1.

Работа этого алгоритма, а также алгоритма Беллмана—Форда показана на рис. 5.22. Для доказательства того, что этот алгоритм приводит к правильному результату, дадим интерпретацию оценкам длины пути D_i для i , не входящих в P . Мы утверждаем, что D_i (в начале каждой итерации шага 1) — кратчайшее расстояние от узла 1 до i среди путей, для которых все узлы, кроме i , содержатся в множестве P . Вначале это имеет место и, как легко заметить, на шаге 2 алгоритма сохраняется это свойство для каждого нового узла, добавляемого к P . Оставшуюся часть докажем по индукции. Можно заметить, что если каждый узел из P по крайней мере также близок к узлу 1, как каждый узел не из P , то следующий ближайший узел не из P должен иметь кратчайший путь, для которого все узлы, кроме последнего, лежат в P ; именно этот узел будет выбран на шаге 1.

Так как число операций, выполняемых алгоритмом Дijkstra на каждом шаге, пропорционально N , а шаги итерируются $N - 1$ раз, то объем вычислений в худшем случае равен $O(N^2)$, а не $O(N^3)$, как у алгоритма Беллмана—Форда. Однако существует много задач, в которых $A \ll N^2$ и алгоритм Беллмана—Форда заканчивается после очень малого числа итераций ($m \ll N$); в этом случае объем вычислений $O(mA)$ может быть меньше чем $O(N^2)$ у алгоритма Дijkstra.

Алгоритм Флойда—Уоршела

Этот алгоритм в отличие от предыдущих двух находит кратчайшие пути сразу для всех пар узлов. Как и в алгоритме Беллмана—Форда, дуговые расстояния могут быть как положительными, так и отрицательными, но также не должно быть циклов отрицательной длины. Во всех трех алгоритмах окончательное решение находится методом итераций, но в каждом алгоритме итерируются разные величины. Если в алгоритме Беллмана—Форда итерируется число дуг в пути, а в алгоритме Дijkstra — длина пути, то в алгоритме Флойда—Уоршела итерируется множество узлов, которые допускается иметь в качестве промежуточных узлов на путях. Как и оба других алгоритма, алгоритм Флойда—Уоршела начинает с расстояний из одной дуги (т. е. без промежуточных узлов), выбранных в качестве исходных оценок для длин кратчайших путей. Затем вычисляются кратчайшие пути с тем ограничением, что промежуточным узлом может быть только узел 1, затем с ограничением, что промежуточными узлами могут быть только узлы 1 и 2 и т. д.

Для более строгого описания алгоритма обозначим через $D_{ij}^{(n)}$ длину кратчайшего пути от узла i к узлу j при ограничении, что только узлы 1, 2, ..., n могут использоваться в качестве промежуточных узлов на пути. Алгоритм при этом работает следующим образом.

Сначала

$$D_{ij}^{(0)} = d_{ij} \quad \text{для всех } i, j, i \neq j. \quad (5.4)$$

Для $n = 0, 1, \dots, N - 1$,

$$D_{ij}^{(n+1)} = \min [D_{ij}^{(n)}, D_{i(n+1)}^{(n)} + D_{(n+1)j}^{(n)}] \quad \text{для всех } i \neq j. \quad (5.5)$$

Доказательство того, что этот алгоритм приводит к правильному результату, будем опять проводить по индукции. При $n = 0$ начальными условиями, очевидно, являются длины кратчайших путей при ограничении, что пути не содержат промежуточных узлов. Теперь предположим, что для данного n величины $D_{ij}^{(n)}$ в алгоритме являются длинами кратчайших путей, для которых в качестве промежуточных узлов используются узлы $1, \dots, n$. Тогда кратчайший путь от i к j , имеющий в качестве промежуточных узлы $1, \dots, (n + 1)$, либо содержит узел $(n + 1)$, либо нет. В первом случае кратчайший путь от i к j при указанном ограничении идет сначала от i к $(n + 1)$, а затем от $(n + 1)$ к j и имеет длину, представленную последним членом в (5.5). Во втором случае кратчайший путь при указанном ограничении будет таким же, как путь, использующий узлы от 1 до n в качестве промежуточных, и поэтому его длина представлена первым членом под знаком минимума в (5.5).

Так как каждый из N шагов проводится для каждой пары узлов, то объем вычислений для алгоритма Флойда—Уоршела равен $O(N^3)$, т. е. такой же, как и у алгоритма Дijkstra, повторенного для всех узлов, выбранных в качестве источника.

5.2.4. Распределенный асинхронный алгоритм Беллмана—Форда

Рассмотрим алгоритм маршрутизации, который каждому пакету назначает маршрут по кратчайшему пути от узла-отправителя пакета до узла-адресата. Длины линий могут меняться либо в результате выхода из строя или, наоборот, ввода в строй линии, либо в результате изменения картины нагрузок в сети. Поэтому возникает необходимость обновления кратчайших путей в ответ на такие изменения. Несколько таких алгоритмов описано в подразд. 5.1.2, когда шла речь о сетях ARPANET и Tymnet.

В этом подразделе мы рассмотрим алгоритм, аналогичный первоначальному алгоритму ARPANET (1969 г.). Он также близок алгоритму, используемому в настоящее время в DNA (архитектуре сетей передачи данных фирмы DEC) [252]. Идея состоит в том, чтобы для вычислений кратчайших расстояний от каждого узла до каждого другого узла использовать распределенный вариант алгоритма Беллмана—Форда. Интересной особенностью этого алгоритма является то, что для его работы в узлах сети необходимо хранить очень мало информации. В самом деле, узлу

не нужны подробные сведения о топологии всей сети, вполне достаточно знать длины уходящих от него линий и номера узлов, находящихся в сети.

Будем считать, что длина d_{ij} каждой линии (i, j) положительна. Сеть всегда остается сильно связной, и если существует линия (i, j) , то существует также линия (j, i) . Следовало бы также иметь в виду, что длины d_{ij} могут изменяться во времени. Однако в этом анализе пока будем предполагать, что длины d_{ij} фиксированы, но начальные условия для алгоритма могут быть произвольными. Эти предположения делают модель адекватной ситуации, когда длины линий остаются фиксированными после некоторого момента t_0 , а все изменения в сети произошли до t_0 .

Основное внимание будет уделено кратчайшему расстоянию D_i от каждого узла i до общего для всех узла-адресата, которым для конкретности будет узел 1. (На практике для каждого узла-адресата алгоритм нужно выполнить отдельно.) Из уравнения Беллмана имеем

$$D_i = \min_{j \in N(i)} [d_{ij} + D_j], \quad i \neq 1, \quad (5.6)$$

$$D_1 = 0, \quad (5.7)$$

где $N(i)$ означает имеющееся в настоящее время множество соседей узла i , т. е. узлов, соединенных с узлом i идущей к нему линией. Эти уравнения эквивалентны уравнениям, приведенным в предыдущем подразделе, но слегка от них отличаются. Там задача состояла в отыскании кратчайших расстояний от данного узла 1 до всех остальных узлов, а здесь — в отыскании кратчайших расстояний до узла 1 от всех остальных узлов. Одну задачу можно получить из другой простой заменой ориентации каждой линии на противоположную без изменения ее длины.

Алгоритм Беллмана—Форда задается уравнениями

$$D_i^{(h+1)} = \min_{j \in N(i)} [d_{ij} + D_j^{(h)}], \quad i \neq 1, \quad (5.8)$$

$$D_1^{(h+1)} = 0. \quad (5.9)$$

В предыдущем подразделе была доказана сходимость к истинным кратчайшим расстояниям для начальных условий

$$D_i^{(0)} = \infty, \quad i \neq 1, \quad (5.10)$$

$$D_1^{(0)} = 0. \quad (5.11)$$

Этот алгоритм хорошо подходит для распределенных вычислений, так как итерации (5.8) можно вычислять в каждом узле параллельно с любым другим узлом. Можно, например, сделать так, чтобы все узлы одновременно выполняли (5.8), обменивались результатами своих вычислений со своими соседями и затем снова выполняли

(5.8) с индексом h , увеличенным на единицу. Если начальные условия выбрать в виде (5.10) и (5.11), то алгоритм закончит работу после не более $N-1$ итераций (где N — число узлов) и в результате каждый узел i будет знать не только свое кратчайшее расстояние D_i , но и уходящую от него линию, лежащую на кратчайшем пути к узлу 1.

К сожалению, реализовать алгоритм в таком синхронном виде не просто. Возникают трудности двух типов. Во-первых, необходимо, чтобы все узлы смогли договориться начать запуск алгоритма одновременно. Во-вторых, необходимо уметь останавливать работу алгоритма и запустить сначала, если по ходу его выполнения у какой-либо линии изменился статус или длина. Хотя все эти трудности можно преодолеть успешно [213], в результате алгоритм становится намного сложнее метода Беллмана—Форда, задаваемого уравнениями (5.8) и (5.9).

Более простой альтернативой является использование такого варианта алгоритма Беллмана—Форда, в котором работа всех узлов не должна быть синхронизирована и который не начинает работу с начальных условий (5.10) и (5.11). Это сделает ненужным как протокол инициирования, так и протокол повторного запуска алгоритма. Алгоритм работает не регламентированно, выполняя время от времени в каждом узле $i \neq 1$ итерацию

$$D_i := \min_{j \in N(i)} [d_{ij} + D_j] \quad (5.12)$$

и используя последнюю оценку D_j , полученную от соседей $j \in N(i)$, и последние длины и статусы уходящих от узла i линий. Алгоритм требует также, чтобы каждый узел i время от времени передавал всем соседям свою последнюю оценку D_i . Однако не нужна синхронизация всех узлов ни в выполнении итераций (5.12), ни в передаче сообщений. Кроме того, не делается никаких предположений относительно начальных значений D_j , $j \in N(i)$, имеющих в каждом узле i . Единственное требование состоит в том, чтобы узел i в конце концов выполнил итерацию (5.12), если она приведет к изменению значения D_i , и передал результат вычислений соседям. Таким образом, рассматривается полностью асинхронный режим работы.

Оказывается, что, работая в асинхронном режиме, алгоритм по-прежнему приводит к правильному результату. Будет показано, что если все изменения длин линий произошли до какого-либо момента t_0 и после этого никакие изменения не происходили, то в течение конечного периода времени после t_0 асинхронный алгоритм найдет правильное кратчайшее расстояние для каждого узла i . Оценками кратчайших длин в момент t_0 могут быть произвольные неотрицательные числа, и поэтому нет необходимости

возобновлять работу алгоритма после каждого изменения статуса или длины какой-либо линии.

Первоначальный алгоритм ARPANET 1969 г. основан на итерациях (5.12) и реализован в асинхронном варианте, очень похожем на описанную выше схему. Соседние узлы обмениваются своими текущими оценками кратчайших расстояний D_j , необходимыми для равенства (5.12), каждые 625 мс, но этот обмен не синхронизирован по всей сети. Более того, алгоритм не начинает работу снова из-за выхода из строя какой-либо линии или изменения ее длины. Главным различием между алгоритмом ARPANET и алгоритмом, анализируемым в этом подразделе, является то, что длины линий d_{ij} в алгоритме ARPANET меняются очень часто и поэтому окончательный устойчивый режим, предполагаемый в нашем анализе, достигается редко.

Теперь определим формально распределенный асинхронный алгоритм Беллмана—Форда и докажем его состоятельность. В каждый момент времени t в каждом узле $i \neq 1$ имеются

$D_i^1(t)$ — оценка кратчайшего расстояния до узла 1 для каждого соседнего узла $j \in N(i)$, которая была передана узлу i в последний раз,

$D_i(t)$ — оценка кратчайшего расстояния до узла 1 узла i , которая была вычислена в соответствии с итерацией Беллмана—Форда в последний раз.

Оценки расстояний для узла-адресата 1 полагаются равными нулю и, таким образом,

$$D_1(t) = 0 \quad \text{для всех } t \geq t_0,$$

$$D_i^1(t) = 0 \quad \text{для всех } t \geq t_0 \text{ и } i, \text{ для которого } 1 \in N(i).$$

В каждом узле i имеются также длины линий d_{ij} для всех $j \in N(i)$, которые предполагаются положительными и не меняющимися во времени после момента t_0 . Будем предполагать, что оценки расстояний не меняются, за исключением некоторых моментов $t_0, t_1, t_2, \dots, t_{m+1} \geq t_m$, для всех m и $t_m \rightarrow 0$ при $m \rightarrow \infty$, в которые в каждом процессоре $i \neq 1$ происходит одно из трех событий:

1. Узел i обновляет $D_i(t)$ по формуле

$$D_i(t) := \min_{j \in N(i)} [d_{ij} + D_j^1(t)]$$

и оставляет оценки $D_j^1(t)$, $j \in N(i)$, неизменными.

2. Узел i получает от одного или большего числа соседей $j \in N(i)$ значения D_j , вычисленные в узле j в какой-либо предыдущий момент времени, обновляет оценки $D_i^1(t)$ и оставляет все остальные оценки неизменными.

3. Узел i свободен и в этом случае все оценки, имеющиеся в узле i , остаются неизменными.

Пусть T^i — множество моментов времени, в которые в узле i происходят обновления по типу 1, а T_j^i — множество моментов времени, в которые узел i получает сообщения от узла j по типу 2. Будем предполагать следующее.

Предположение 1. Узлы никогда не прекратят обновлять свои собственные оценки и получать сообщения от всех своих соседей, т. е. T^i и T_j^i состоят из бесконечного числа элементов для всех $i \neq 1$ и $j \in N(i)$.

Предположение 2. Все имеющиеся в узлах начальные оценки $D_i(t_0)$ и $D_j^i(t_0)$, $j \in N(i)$ являются неотрицательными. Кроме того, все оценки, переданные узлам их соседями до начального момента t_0 , но полученные после момента t_0 , являются неотрицательными.

Предположение 3. Система обязательно очистится от устаревшей информации о расстояниях, т. е. для любого момента $\bar{t} \geq t_0$ найдется такой момент $\tilde{t} > \bar{t}$, после которого любой узел i уже не получит оценку D_j , если она была вычислена в соседнем узле $j \in N(i)$ до момента \bar{t} .

Следующее утверждение показывает, что оценки $D_i(t)$ сходятся к истинным кратчайшим расстояниям в течение конечного периода времени. Доказательство (которое можно было бы опустить без потери непрерывности изложения) интересно тем, что оно служит моделью для доказательства правильной сходимости некоторых других асинхронных распределенных алгоритмов (см. доказательство сходимости алгоритма из подразд. 5.3.3 и работы [22, 26]).

Утверждение. Существует момент времени t_m , такой, что

$$D_i(t) = D_i \quad \text{для всех } t \geq t_m, \quad i = 1, \dots, N,$$

где D_i — правильное кратчайшее расстояние от узла i до узла-адресата 1.

Доказательство. Идея доказательства состоит в том, чтобы для каждого узла i определить две последовательности $\{\underline{D}_i^k\}$ и $\{\bar{D}_i^k\}$, которые удовлетворяют неравенствам

$$\underline{D}_i^k \leq \underline{D}_i^{k+1} \leq D_i \leq \bar{D}_i^{k+1} \leq \bar{D}_i^k \quad (5.13)$$

и

$$\underline{D}_i^k = D_i = \bar{D}_i^k \quad \text{для достаточно большого } k. \quad (5.14)$$

Эти последовательности получаются алгоритмом Беллмана—Форда, начинающимся при двух разных начальных условиях. Затем

показывается, что для каждых k и i все оценки $D_i(t)$ удовлетворяют неравенствам

$$\underline{D}_i^k \leq D_i(t) \leq \bar{D}_i^k \quad \text{для всех достаточно больших } t. \quad (5.15)$$

Ключевую роль в доказательстве играет свойство монотонности итераций Беллмана—Форда. Это свойство состоит в том, что если для некоторых скалярных величин \bar{D}_j и \tilde{D}_j выполняется неравенство

$$\bar{D}_j \geq \tilde{D}_j \quad \text{для всех } j \in N(i),$$

то знак неравенства сохранится и после итерации, т. е.

$$\min_{j \in N(i)} [d_{ij} + \bar{D}_j] \geq \min_{j \in N(i)} [d_{ij} + \tilde{D}_j].$$

Из этого неравенства следует, что если D_i^k являются последовательностями, порожденными итерациями Беллмана—Форда (5.8) и (5.9), начинающимися при некотором начальном условии D_i^0 , $i = 1, \dots, N$ и справедливо соотношение $D_i^1 \geq D_i^0$ для каждого i , то $D_i^{k+1} \geq D_i^k$ для всех i и k . Аналогично если $D_i^1 \leq D_i^0$ для каждого i , то $D_i^{k+1} \leq D_i^k$ для всех i и k .

Рассмотрим алгоритм Беллмана—Форда, определенный равенствами (5.8) и (5.9). Пусть \bar{D}_i^k , $i = 1, \dots, N$, — k -я итерация этого алгоритма с начальным условием

$$D_i^{(0)} = \infty, \quad i \neq 1, \quad (5.16)$$

$$D_1^{(0)} = 0, \quad (5.17)$$

а D_i^k , $i = 1, \dots, N$ — k -я итерация с начальным условием $D_i^{(0)} = 0$, $i = 1, 2, \dots, N$.

Лемма. Определенные выше последовательности $\{D_i^k\}$ и $\{\bar{D}_i^k\}$ удовлетворяют соотношениям (5.13) и (5.14).

Доказательство. Соотношение (5.13) доказывается по индукции с помощью свойства монотонности итераций Беллмана—Форда и выбора указанных выше начальных условий. Для доказательства (5.14) заметим сначала, что из анализа сходимости алгоритма Беллмана—Форда, выполненного в предыдущем разделе, следует, что для всех i

$$\bar{D}_i^k = D_i, \quad k \geq N - 1. \quad (5.18)$$

Поэтому остается доказать только равенство $\underline{D}_i^k = D_i$ для достаточно большого k . Для этого, используя индукцию, можно доказать сначала, что \underline{D}_i^k для любого k является суммой длин не более чем k линий. Далее, используя неравенство $\underline{D}_i^k \leq D_i$, можно

показать, что для любого k величина D_i^k не может быть суммой длин более чем

$$\frac{\max_i D_i}{\min_{(i, j)} d_{ij}} \quad (5.19)$$

линий. Поэтому число всевозможных значений \underline{D}_i^k для $i = 1, \dots, N$ и $k = 0, 1, \dots$ конечно. Так как \underline{D}_i^k — монотонно неубывающая по k последовательность для всех i , отсюда следует, что для некоторого \bar{k} имеем $\underline{D}_i^k = \underline{D}_i^{k+1}$ для всех $i = 1, \dots, N$. Поэтому скалярные величины $\underline{D}_i^{\bar{k}}$ удовлетворяют уравнению Беллмана, имеющему в качестве своего единственного решения кратчайшие расстояния D_i . Следовательно,

$$\underline{D}_i^k = D_i \quad \text{для всех } i = 1, \dots, N, \quad k \geq \bar{k}. \quad (5.20)$$

Равенства (5.18) и (5.20) дают (5.14), что и доказывает лемму.

Завершая доказательство утверждения, покажем по индукции, что для любого k существует такой момент времени t_k , в котором для всех $t \geq t(k)$

$$\underline{D}_i^k \leq D_i(t) \leq \bar{D}_i^k, \quad i = 1, 2, \dots, N, \quad (5.21)$$

$$\underline{D}_j^k \leq D_j^i(t) \leq \bar{D}_j^k, \quad j \in N(i), \quad i = 1, 2, \dots, N \quad (5.22)$$

и для всех $t \in T_i^t$, $t \geq t(k)$

$$\underline{D}_i^k \leq D_i[\tau_j^i(t)] \leq \bar{D}_j^k, \quad j \in N(i), \quad i = 1, 2, \dots, N, \quad (5.23)$$

где $\tau_j^i(t) \leq t$ — последний момент, в который оценка $D_j^i(t)$, имеющаяся в узле i в момент t , вычислялась в узле j с использованием равенства (5.12). (Более формально $\tau_j^i(t)$ можно определить как максимальный, но меньший чем t момент времени в T^I , такой, что $D_j[\tau_j^i(t)] = D_j^i(i)$.)

Действительно, предположение индукции при $k = 0$ (для $t(0) = t_0$) выполняется благодаря предположению о неотрицательности начальных оценок, находящихся либо в узлах, либо в процессе передачи к узлам (предположение 2). Считая теперь, что предположение индукции выполняется для данного k , покажем, что найдется момент $t(k+1)$ с требуемыми свойствами. Действительно, из соотношения (5.22) и монотонности итерации Беллмана—Форда имеем для любого $t \in T^i$, $t \geq t(k)$ (т. е. для момента t , в который происходит обновление $D_i(t)$ посредством итерации Беллмана—Форда)

$$\underline{D}_i^{k+1} \leq D_i(t) \leq \bar{D}_i^{k+1}.$$

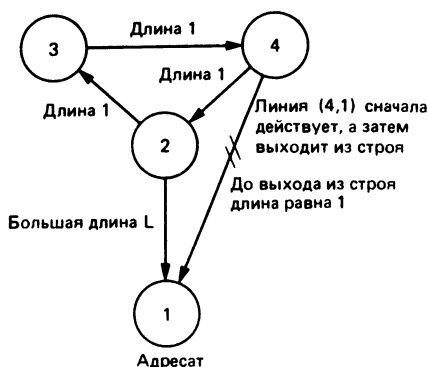


Рис. 5.23. Пример, в котором число итераций (синхронного) алгоритма Беллмана—Форда оказывается слишком большим. Предположим, что начальными условиями являются кратчайшие расстояния от узлов 2, 3 и 4 до узла 1, до того как линия (4, 1) вышла из строя ($D_2 = 3$, $D_3 = 2$, $D_4 = 1$). После того как линия (4, 1) вышла из строя, потребуется примерно L итераций, до того как узел 2 поймет, что его кратчайшим путем к узлу 1 является прямая линия (2, 1). Это пример так называемого феномена плохих новостей, когда алгоритм медленно реагирует на внезапное увеличение длины одной или нескольких линий.

Поэтому если $t'(k)$ — минимальный момент времени $t \in T^i$ и $t \geq t(k)$, то

$$\underline{D}_i^{k+1} \leq D_i(t) \leq \overline{D}_i^{k+1} \quad \text{для всех } t \geq t'(k), \quad i = 1, \dots, N. \quad (5.24)$$

Так как $\tau_j^i(t) \rightarrow \infty$ при $t \rightarrow \infty$ (в соответствии с предположениями 1 и 3), можно выбрать момент времени $t(k+1) \geq t'(k)$, такой, что $\tau_j^i(t) \geq t'(k)$ для всех $i, j \in N(i)$ и $t \geq t(k+1)$. Тогда с учетом (5.24) имеем, что для всех $t \geq t(k+1)$

$$\underline{D}_j^{k+1} \leq D_j^i(t) \leq \overline{D}_j^{k+1}, \quad j \in N(i), \quad i = 1, 2, \dots, N \quad (5.25)$$

и для всех $t \in T_j^i$, $t \geq t(k+1)$

$$\underline{D}_j^{k+1} \leq D_j[\tau_j^i(t)] \leq \overline{D}_j^{k+1}, \quad j \in N(i), \quad i = 1, 2, \dots, N. \quad (5.26)$$

Это завершает индукцию и доказательство утверждения.

В предыдущем анализе предполагалось, что сеть связна. Если это не так, то анализ применим к части сети, которая связна и содержит узел 1. Для узла i , принадлежащего другой, не содержащей узел 1 части сети, и имеющего хотя бы одного соседа (чтобы можно было действовать согласно алгоритму), можно показать, что $D_i(t) \rightarrow \infty$ при $t \rightarrow \infty$. Это свойство узел может использовать для идентификации узлов-адресатов, с которыми он не связан.

Завершим этот раздел обсуждением двух недостатков асинхронного метода Беллмана—Форда. Первый состоит в том, что в некоторых ситуациях алгоритм может потребовать слишком много итераций до завершения работы (см. пример на рис. 5.23). Это происходит не в результате асинхронности алгоритма, а из-за

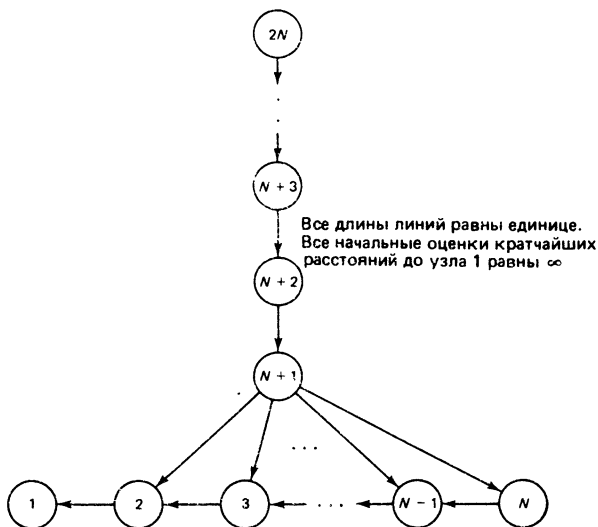


Рис. 5.24. Пример, в котором асинхронный вариант алгоритма Беллмана—Форда требует намного больше передач сообщений, чем синхронный вариант. Начальные оценки кратчайших расстояний для всех узлов равны ∞ . Рассмотрим следующую последовательность событий, когда все сообщения принимаются с нулевой задержкой.

1. Узел 2 обновляет свое кратчайшее расстояние и передает результат узлу 3. Узел 3 обновляет и передает результат узлу 4... Узел $N - 1$ обновляет и передает результат узлу N . Узел N обновляет и передает результат узлу $N + 1$. (Это составит $N - 1$ сообщение.)

2. Узел $N + 1$ обновляет и передает результат узлу $N + 2$... Узел $2N - 1$ обновляет и передает результат узлу $2N$. Узел $2N$ обновляет. (Это составит $N - 1$ сообщение.)

3. Для $i = N - 1, N - 2, \dots, 2$ и именно в таком порядке узел i передает свой результат обновления узлу $N + 1$, и последовательность событий с номером 2 повторяется. (Это составит $N(N - 2)$ сообщений.) Суммарное число сообщений равно $N^2 - 2$. Если бы алгоритм выполнялся в синхронном варианте, то потребовалось бы только $3(N - 1) - 1$ сообщений. Такое расхождение объясняется тем, что в асинхронном варианте много несущественной информации прибывает в узел $N + 1$ слишком рано и это приводит к тому, что много ненужных сообщений передается по пути от $N + 1$ до $2N$.

произвольности выбора начальных условий (на это указывает рассуждение, предшествующее выражению (5.19)). Вторым недостатком, продемонстрированный на рис. 5.24, состоит в том, что при неблагоприятном стечении обстоятельств алгоритм может потребовать слишком много передач сообщений. В настоящее время не известно, в какой степени эти недостатки влияют на усредненные характеристики алгоритма (пример на рис. 5.24 требует маловероятной последовательности событий).

5.2.5. Адаптивная маршрутизация, основанная на кратчайших путях

Возможность использования в качестве длин линий некоторых величин, которые бы отражали степень нагруженности линии в недавнем прошлом, обсуждалась в предыдущих разделах. Идея состоит в том, что если более нагруженной линии приписывать большую длину, то в результате алгоритм отыскания кратчайшего пути не будет стремиться использовать эту линию в качестве части маршрутного пути. Это выглядит заманчиво, но после некоторых раздумий появляется тревога по поводу возможного появления колебаний. Далее мы увидим, что такая возможность особенно опасна в дейтаграммных сетях.

Вопросы устойчивости в дейтаграммных сетях

Для того чтобы привести простой пример колебаний, рассмотрим дейтаграммную сеть и предположим, что есть два пути, по которым узел-отправитель может направить трафик узлу-получателю. Выбор данного пути при маршрутизации повлечет за собой увеличение длины этого пути и в следующий период времени другой путь станет более предпочтительным при выборе маршрута, в результате чего возникнут колебания. Оказывается, что тип колебаний может оказаться гораздо хуже, как показывает следующий пример.

Пример 1

Рассмотрим сеть с 16 узлами, показанную на рис. 5.25, в которой узел 16 является единственным узлом-адресатом. Пусть интенсивность входного трафика (единицы данных/сек) в каждом из узлов $i = 1, \dots, 7, 9, \dots, 15$ равна единице, а интенсивность входного трафика в узле 8 равна $\epsilon > 0$, где ϵ очень мало. Предположим, что длина линии (i, j) равна

$$d_{ij} = F_{ij},$$

где F_{ij} — интенсивность поступающего в линию трафика, включая входной и транзитный трафики. Предположим, что все узлы вычисляют свои кратчайшие пути до узла-адресата каждые T секунд, используя в качестве длин линий интенсивности F_{ij} , измеренные в течение предыдущих T секунд, а затем направляют свой трафик по кратчайшему пути в течение последующих T секунд. Пусть вначале узлы 1—7 выбрали маршруты по часовой стрелке, а узлы 8—15 — против часовой стрелки. Это достаточно хорошая маршрутизация, сбалансированно распределяющая входной трафик по двум направлениям. На рис. 5.26 показаны интенсивности поступающих в линию трафиков для первоначальной и последующих маршрутизаций, основанных на отыскании кратчайших путей. После трех обновлений кратчайших путей алгоритм приводит к устойчивому колебательному режиму, в котором весь трафик поочередно направляется то по часовой стрелке, то против нее; смена направления происходит при каждом очередном обновлении.

Разумеется, такая маршрутизация — наилучшая среди всех возможных.

Причиной плохой маршрутизации в предыдущем примере является тот факт, что интенсивности поступающих в линии нагрузок зависят от выбранной маршрутизации, которая в свою очередь зависит от интенсивностей проходящих по линиям трафиков, в результате чего возникает эффект обратной связи. Это похоже на систему с обратной связью из теории управления и поэтому здесь можно использовать соответствующие методы [24]. Можно показать, что рассмотренный выше тип неустойчивости появляется в основном тогда, когда длина d_{ij} линии (i, j) возрастает непрерывно и монотонно с ростом проходящей по линии нагрузки F_{ij} и $d_{ij} = 0$, если $F_{ij} = 0$. Эти колебания можно погасить путем добавления положительной константы к длине линии так, чтобы $d_{ij} = \alpha > 0$, когда $F_{ij} = 0$. Скалярная величина α (длина линии при нулевой нагрузке) называется коэффициентом смещения. Колебания в первоначальном алгоритме ARPANET, о котором речь шла в подразд. 5.1.2, гасились с помощью довольно большого коэффициента смещения. Заметим, что если α — достаточно большое число, то соответствующие кратчайшие пути будут путями с наименьшим числом линий до узла-адресата. В результате маршрутизация станет статической, у которой не может быть никаких колебаний указанного выше типа, но она также будет полностью нечувствительна к возникающим перегрузкам. В существующем алгоритме ARPANET, о котором речь шла в подразд. 5.1.2, коэффициент α , по существу, равен сумме среднего времени передачи пакета, задержки на обработку пакета и задержки при распространении сигнала по линии. Эта величина представляется подходящей, учитывая, что сеть ARPANET обычно работает в условиях малой нагрузки. Второй способ погашения колебаний состоит во введении механизма усреднения длин линий в течение временного интервала, охватывающего более одного обновления кратчайших путей. Это делает алгоритм более устойчивым, но зато приходится расплачиваться уменьшением быстроты реакции алгоритма на возникающие перегрузки. Оказывается, что асинхронное обновление кратчайших путей приводит в ре-

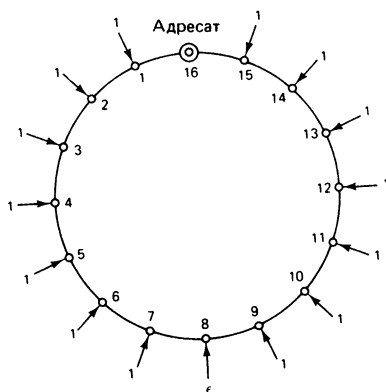
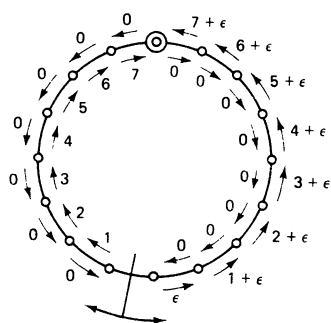
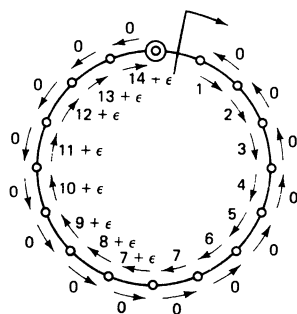


Рис. 5.25. Кольцевая сеть с шестнадцатью узлами из примера 1. Узел 16 является единственным адресатом.

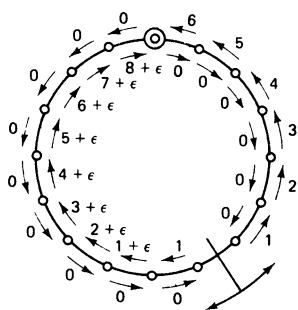
является в основном тогда, когда длина d_{ij} линии (i, j) возрастает непрерывно и монотонно с ростом проходящей по линии нагрузки F_{ij} и $d_{ij} = 0$, если $F_{ij} = 0$. Эти колебания можно погасить путем добавления положительной константы к длине линии так, чтобы $d_{ij} = \alpha > 0$, когда $F_{ij} = 0$. Скалярная величина α (длина линии при нулевой нагрузке) называется коэффициентом смещения. Колебания в первоначальном алгоритме ARPANET, о котором речь шла в подразд. 5.1.2, гасились с помощью довольно большого коэффициента смещения. Заметим, что если α — достаточно большое число, то соответствующие кратчайшие пути будут путями с наименьшим числом линий до узла-адресата. В результате маршрутизация станет статической, у которой не может быть никаких колебаний указанного выше типа, но она также будет полностью нечувствительна к возникающим перегрузкам. В существующем алгоритме ARPANET, о котором речь шла в подразд. 5.1.2, коэффициент α , по существу, равен сумме среднего времени передачи пакета, задержки на обработку пакета и задержки при распространении сигнала по линии. Эта величина представляется подходящей, учитывая, что сеть ARPANET обычно работает в условиях малой нагрузки. Второй способ погашения колебаний состоит во введении механизма усреднения длин линий в течение временного интервала, охватывающего более одного обновления кратчайших путей. Это делает алгоритм более устойчивым, но зато приходится расплачиваться уменьшением быстроты реакции алгоритма на возникающие перегрузки. Оказывается, что асинхронное обновление кратчайших путей приводит в ре-



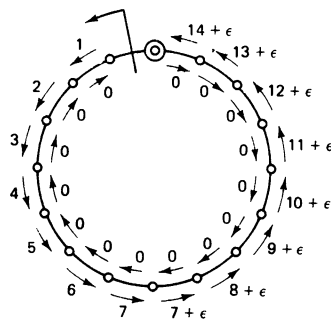
Первая
маршрутизация



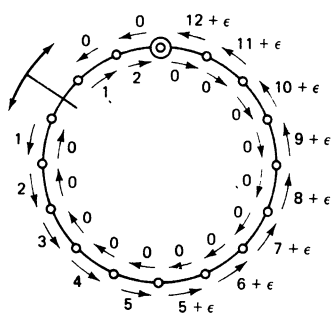
Четвертая
маршрутизация



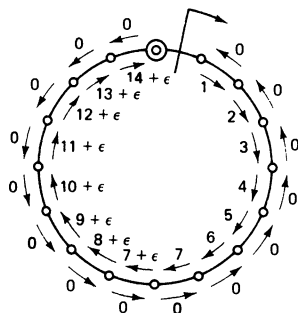
Вторая
маршрутизация



Пятая
маршрутизация



Третья
маршрутизация



Шестая
маршрутизация

Рис. 5.26. Колебания в кольцевой сети в случае, когда длины линий d_{ij} равны интенсивностям поступающего в линии трафика F_{ij} . Каждый узел посылает одну единицу входного трафика узлу-адресату, за исключением среднего узла 8, который посылает $\varepsilon > 0$, где ε очень мало. Числа рядом с линиями означают

зультате к некоторому усреднению их длин, что благотворно сказывается на устойчивости. (Более подробно это описано в работах [20, 23].)

Вопросы устойчивости в сетях с виртуальными цепями

Колебания, о которых речь шла ранее, в основном ассоциируются с дейтаграммными сетями. Мы покажем, что в сетях с виртуальными цепями колебания не столь сильны. Основной особенностью дейтаграммной сети является то, что каждый пакет пары пользователей не должен следовать по тому же пути, что и предыдущий пакет. Поэтому время, в течение которого пара отправитель—адресат будет использовать данный путь после того, как он был выбран при очередном обновлении маршрутов, очень мало. В результате дейтаграммная сеть реагирует очень быстро на обновление кратчайших путей и почти мгновенно перенаправляет весь трафик по новым кратчайшим путям.

Совершенно другая ситуация имеет место в сетях с виртуальными цепями, в которых каждому сеансу приписывается фиксированный путь в момент, когда он впервые установлен. Средняя продолжительность жизни виртуальной цепи часто оказывается больше периода между обновлениями кратчайшего пути. В результате реакция сети на обновление кратчайших путей оказывается более плавной, так как на старых сеансах продолжает использоваться выбранные ранее пути и только новым сеансам назначаются кратчайшие пути, вычисленные в последний момент.

Следующий пример демонстрирует эти явления.

Пример 2

Рассмотрим простую сеть, состоящую из двух линий, одного узла-отправителя и одного узла-адресата, изображенную на рис. 5.27, а. Предположим, что интенсивность (скорость) поступающей нагрузки равна r бит/с и обе линии имеют одинаковую пропускную способность C . «Оптимальный» алгоритм маршрутизации должен разделить в каком-то смысле входную нагрузку r поровну между двумя линиями и поэтому пропускная способность сети может доходить до $2C$.

Посмотрим, как работает в этой сети какой-либо типичный адаптивный алгоритм, основанный на отыскании кратчайших путей. Алгоритм делит времен-

интенсивности проходящего по линии потока в каждом из двух направлений. В качестве примера вычислений кратчайших путей рассмотрим первую итерацию. Средний узел 8 вычисляет длину пути по часовой стрелке $28 (= 0 + 1 + 2 + \dots + 7)$ и длину пути против часовой стрелки $28 + 8\epsilon (= \epsilon + 1 + \epsilon + 2 + \dots + \epsilon + 7 + \epsilon)$ и направляет свой трафик на кратчайший (по часовой стрелке) путь при второй маршрутизации. Соответствующими числами для узла 9 будут 28 и $28 + 7\epsilon$ и поэтому узел 9 также направляет свой трафик по часовой стрелке. Все остальные узлы обнаруживают, что путь, использованный ими при первой маршрутизации, является кратчайшим и поэтому они не будут направлять свой трафик на другой путь.

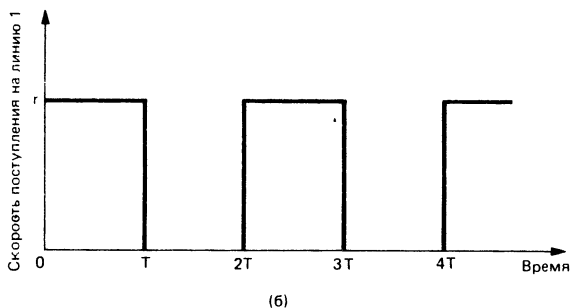
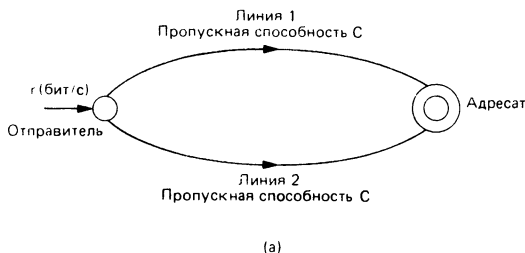


Рис. 5.27.

a — сеть с двумя линиями из примера 2. $б$ — интенсивность поступающего в линию 1 трафика в примере 2, в этом примере используется правило кратчайшего пути в дейтаграммном случае. По существу, только один путь используется при маршрутизации в любой момент времени, если период между обновлениями кратчайших путей значительно больше, чем время, необходимое для уменьшения длины очереди ожидающих пакетов, накопившихся к моменту обновления.

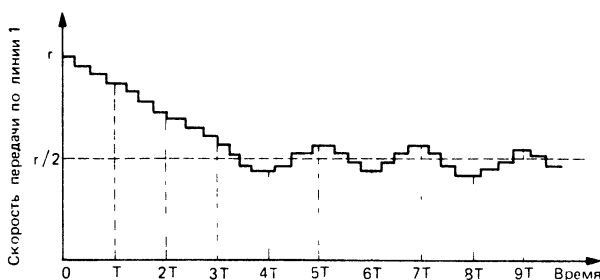
ную ось на T -секундные интервалы, измеряет средние скорости (бит/с) поступающей нагрузки для обеих линий и направляет весь трафик (как в дейтаграммном режиме, так и в режиме с виртуальными цепями), возникающий в течение всего T -секундного интервала, по линии, имевшей наименьшую нагрузку в предыдущем интервале.

Если сеть работает в дейтаграммном режиме и T намного больше среднего времени уменьшения очереди пакетов, ожидающих передачи в момент очередного обновления маршрутов, то каждая линия, чередуя временные интервалы, либо ничего не будет пропускать, либо будет пропускать весь входной трафик интенсивности r , как показано на рис. 5.27, б.

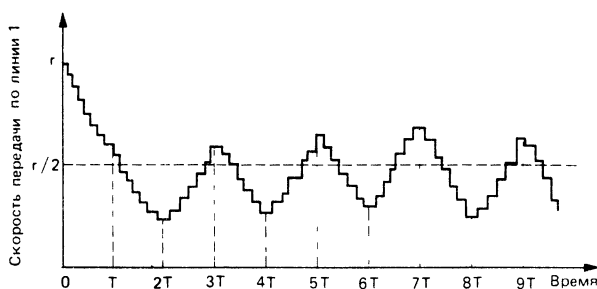
Теперь рассмотрим случай, когда сеть работает в режиме с виртуальными цепями. Пусть моменты возникновения виртуальных цепей образуют пуассоновский процесс интенсивности λ возникновения в секунду и каждая виртуальная цепь использует линию, которая была ей предоставлена алгоритмом маршрутизации, в течение всего своего существования, продолжительность которого распределена экспоненциально со средним $1/\mu$ секунд. Поэтому согласно результатам относительно системы $M/M/\infty$ (подразд. 3.4.2) число существующих в данный момент виртуальных цепей имеет пуассоновское распределение со средним λ/μ . Если γ — средняя скорость передачи (бит/с) в виртуальной цепи, то $r = (\lambda/\mu) \gamma$, или

$$\gamma = \frac{r\mu}{\lambda}. \quad (5.27)$$

Предположим, что интервал между обновлениями кратчайших путей \bar{T} мал по сравнению со средней продолжительностью жизни виртуальных цепей $1/\mu$.



(а)



(б)

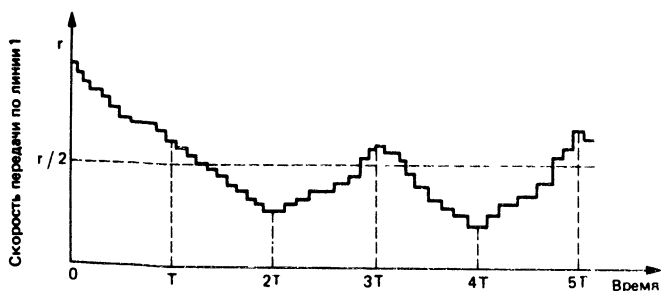
Рис. 5.28. а, б. Интенсивность поступающего в линию 1 трафика в примере 2, когда используются виртуальные цепи.

(а) — виртуальные цепи длятся долго, а кратчайшие пути обновляются часто (μT мало);
(б) — виртуальные цепи длятся мало, а кратчайшие пути обновляются часто (μT принимает среднее значение).

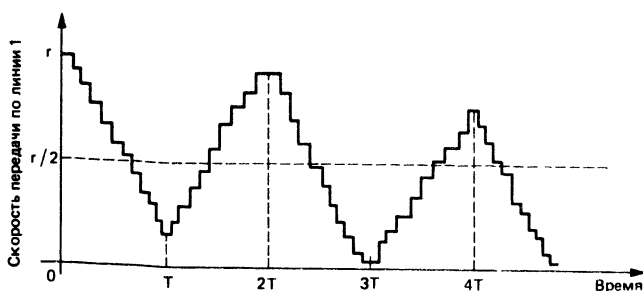
Тогда доля виртуальных цепей, которые кончились в течение T -секундного интервала, от всех цепей, существовавших в начале данного интервала на каждой линии, приблизительно равна μT . В среднем λT виртуальных цепей добавятся в линии, которая передавала этот трафик в течение предыдущего интервала. Новые цепи увеличат интенсивность нагрузки на $\gamma \lambda T$ бит/с или с учетом (5.27) на $r \mu T$ бит/с. Поэтому средние интенсивности x_1^k и x_2^k (бит/с) нагрузок в этих двух линиях в k -м временном интервале будут приблизительно меняться в соответствии с равенством

$$x_i^{k+1} = \begin{cases} (1 - \mu T) x_i^k + r \mu T, & \text{если } i \text{ — кратчайший путь,} \\ & \text{т. е. } x_i^k = \min \{x_1^k, x_2^k\}, \\ (1 - \mu T) x_i^k & \text{в противном случае.} \end{cases} \quad (5.28)$$

На рис. 5.28 показаны примеры изменений средней нагрузки в линии 1. Из равенства (5.28) видно, что при $k \rightarrow \infty$ средние интенсивности x_1 и x_2 в пределе будут колебаться около $r/2$ с размахом колебаний, примерно равным $r \mu T$. Поэтому если средняя продолжительность жизни виртуальной цепи велика по сравнению с интервалом между обновлениями кратчайших путей ($\mu T \ll 1$), то алгоритм маршрутизации работает почти оптимально, разделяя входной



(в)



(г)

Рис. 5.28 в, г. Интенсивность поступающего в линию 1 трафика в примере 2, когда используются виртуальные цепи.

(в) — виртуальные цепи длятся долго, а кратчайшие пути обновляются редко (μT принимает среднее значение); (г) — виртуальные цепи длятся мало, а кратчайшие пути обновляются редко (μT велико). В пределе при $\mu \rightarrow \infty$ получается деятиграммный случай.

трафик почти поровну между двумя линиями. Наоборот, если произведение μT велико, то из предыдущего анализа видно, что амплитуда колебаний в этом случае будет также велика. На рис. 5.28 показана зависимость между амплитудой колебаний, μ и T .

Предыдущий пример иллюстрирует результат, который был получен аналитически для сетей с виртуальными цепями общего вида [79]. В работе [79] было также показано, что средняя продолжительность жизни виртуальной цепи является основным параметром, определяющим характеристики адаптивной маршрутизации кратчайших путей. Если она очень большая, то скорость сходимости алгоритма будет медленной, так как виртуальные цепи, которые ошибочно выбрали перегруженные пути, должны будут долго их использовать. Если она очень мала, то интервалы между обновлениями кратчайших путей должны быть также малы для того, чтобы колебания относительно оптимальных значений были малы (см. рис. 5.28). К сожалению, существует естественный пре-

дел, меньше которого интервалы между обновлениями быть не могут, так как необходимо некоторое время для того, чтобы достаточно точно измерить текущие значения длин линий, и, кроме этого, частые обновления маршрутов требуют увеличения объема передаваемой по сети служебной информации. Если бы виртуальные цепи могли изменять маршруты в течение своей жизни, то никаких проблем с большой длительностью виртуальных цепей не было бы. В сети Codex, описанной в разд. 5.8, такое изменение маршрутов допускается и в результате ее алгоритм маршрутизации оказывается более эффективным, чем рассмотренный в этом разделе.

5.3. Распространение информации, необходимой при маршрутизации; обращение с линиями, выходящими из строя

Задача, с которой часто приходится сталкиваться при маршрутизации, состоит в передаче управляющей информации от мест в сети, где она собирается, туда, где она необходима. Эта задача усложняется, если линии могут выходить из строя. В этом разделе объясняется существо возникающих при этом трудностей и рассматриваются пути их устранения. Здесь также приведены примеры распределенных алгоритмов.

Один практический пример распространения маршрутной информации уже рассматривался в подразд. 5.1.2, когда шла речь о последнем алгоритме ARPANET. В этом примере длины всех линий периодически сообщались всем узлам, которые затем вычисляли кратчайшие пути и после этого обновляли свои маршрутные таблицы. В некоторых других случаях возникают ситуации, когда необходимо предупредить более высокие уровни о появившихся изменениях в топологии или изменить структуры данных, на которые могут повлиять изменения в сетевой топологии, например остовное дерево, используемое для передачи сообщений, адресованных всем узлам. Термин *сетевая топология* будет часто использоваться в этом разделе. Он является синонимом списка узлов и линий сети со статусами (линия действует или вышла из строя) каждой линии.

Надежное получение маршрутной информации в местах, где она необходима, в условиях когда линии могут выходить из строя, сопряжено с рядом тонкостей, которые еще до конца не оценены. Вот некоторые из них.

1. Информация о топологических изменениях должна передаваться по линиям, которые могут выходить из строя. В некоторых сетях необходимо уметь поддерживать их функционирование, даже когда некоторые части сети становятся несвязанными друг с другом. В качестве примера рассмотрим *централизованный* алгоритм,

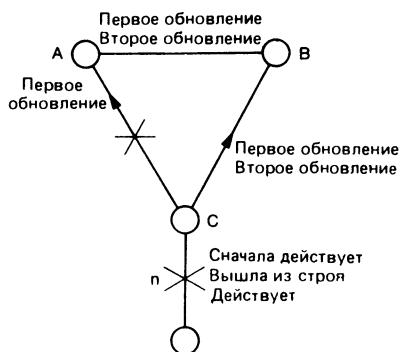


Рис. 5.29. Пример, в котором лавинный алгоритм не работает. Сначала линия n действует, затем выходит из строя, а потом снова действует. Предположим, что два обновляющих сообщения проходят по пути CBA быстрее, чем первое обновляющее сообщение проходит по линии CA . Допустим также, что линия CA выходит из строя после первого обновляющего сообщения, но до того как по нему пройдет второе обновляющее сообщение. Тогда последнее полученное A сообщение содержит информацию, что линия n вышла из строя, в то время как в действительности

линия действует. Трудность здесь состоит в том, что устаревшая информация ошибочно считается новой.

в котором вся информация, касающаяся топологических изменений, передается в специальный узел, называемый центром управления сети (ЦУС). ЦУС хранит в памяти маршрутные таблицы всей системы и обновляет их после получения новой информации о топологических изменениях, используя некоторый алгоритм, который сейчас не важен. После этого он приступает к передаче всей информации, необходимой для выполнения локальных операций по маршрутизации, всем узлам, которым эта информация адресована. Помимо основных задач сбора и распределения информации по линиям, которые могут выходить из строя, перед алгоритмом стоит еще одна проблема — возможность выхода из строя ЦУС или потери его связи с какой-либо частью сети. В одних случаях удастся за счет избыточности сделать вероятность появления таких ситуаций очень малой. Однако в других случаях эти трудности легко преодолеть не удастся и тогда, возможно, стоит вообще отказаться от централизованного подхода и использовать распределенный алгоритм.

2. Приходится иметь дело с многократными топологическими изменениями (такими как смена статуса линии с действующей на вышедшую из строя и обратно в течение короткого времени) и поэтому возникает проблема различения старой и новой информации об изменениях. В качестве примера рассмотрим метод распространения среди всех узлов информации о топологических изменениях, называемый *лавинным алгоритмом*. Суть этого метода состоит в том, что каждый узел следит за статусом всех уходящих от него линий и после обнаружения изменения статуса какой-либо линии посылает пакет всем своим соседям, сообщая им об этом изменении. Соседи пересылают этот пакет своим соседям и т. д. Необходимо предпринять некоторые меры, для того

чтобы этот обновляющий пакет не циркулировал по сети бесконечно долго; этот вопрос будет рассмотрен позднее. Лавинный алгоритм работает, когда имеется только одно топологическое изменение, и может оказаться непригодным, когда возможны многократные изменения, как показано в примере на рис. 5.29. Способы преодоления этой трудности будут рассмотрены в этом разделе; будет показано, что иногда простые очевидные решения могут содержать замаскированные ошибки.

3. Обновляющая информация о топологических изменениях используется некоторым алгоритмом, вычисляющим новые маршрутные пути (например, алгоритмом отыскания кратчайшего пути). Вполне возможно, однако, что обновляющая информация о топологических изменениях поступит во время работы алгоритма. Тогда алгоритм должен суметь либо в ходе своей работы учитывать эти изменения, либо остановиться и заново начать действовать после получения новых данных. Выполнение того или другого может оказаться нетривиальным, особенно если алгоритм является распределенным.

4. В результате восстановления какой-либо линии две несвязные части сети могут снова оказаться связными. Каждая из частей, возможно, имеет устаревшую информацию о топологии другой части. Алгоритм должен гарантировать, что в конечном итоге обе части согласуются и им станет известна правильная топология сети.

Рассмотренные выше трудности возникают также при распространении информации, касающейся степени загруженности каждой линии. Важное отличие, однако, состоит в том, что неправильная информация о перегрузках обычно имеет менее серьезные последствия, чем неправильная информация о топологии. Если в первом случае худшее, что можно ожидать, это наилучший выбор маршрутов, то во втором случае можно выбрать вообще несуществующие маршруты. Поэтому если алгоритм распространяет информацию о нагрузках, а не о топологии, то можно позволить допустить небольшой уровень ошибок, если это приведет к существенному упрощению алгоритма или уменьшит объем передач по сети. Заметим, однако, что в некоторых методах, таких как лавинный алгоритм в сети ARPANET, о котором речь пойдет далее, информация о нагрузках содержит сведения о топологических изменениях и один и тот же алгоритм используется для распространения обоих типов информации.

К вопросу о распространении обновляющей информации о топологии следует отметить, что невозможно сделать так, чтобы каждый узел в любой момент времени знал истинную топологию сети. Поэтому лучшее, что можно ожидать от алгоритма, — это что он сможет успешно справляться с любым конечным числом топологических изменений в течение конечного времени. Под этим

мы имеем в виду, что если до какого-либо момента времени произошло конечное число изменений и после этого никаких изменений не было, то все узлы каждой связной области сети должны узнать правильный статус каждой линии данной области в течение конечного интервала времени.

При обсуждении свойств различных схем в описанном выше смысле мы будем предполагать следующее.

1. По линиям сети сообщения передаются без ошибок и с сохранением очередности. Кроме того, в узлах сообщения хранятся в памяти без искажений.

2. Выход из строя линии обнаруживается в обоих ее конечных узлах, хотя не обязательно одновременно. Под этим мы имеем в виду, что если какой-либо концевой узел объявляет, что линия вышла из строя, то второй концевой узел также объявляет, что она вышла из строя, до того как первый концевой узел объявит, что данная линия снова вошла в строй.

3. Имеется некоторый протокол линии передачи данных для регистрации на концах вышедшей из строя линии момента времени, когда данная линия станет действовать снова. Если один из концевых узлов объявляет, что данная линия вошла в строй, то в течение конечного интервала времени либо противоположный концевой узел также объявит, что эта линия вошла в строй, либо первый узел объявит, что эта линия снова вышла из строя.

4. Узел может выйти из строя и в этом случае каждая из прилегающих к нему линий объявляется вышедшей из строя; это объявляется узлом на противоположном конце линии в течение конечного интервала времени.

Сделанные предположения создают основу для обсуждения и анализа конкретных схем, но они не всегда удовлетворяются на практике. Например, в редких случаях предположение 1 может нарушаться вследствие того, что поврежденные кадры данных могут быть не замечены тестом обнаружения ошибок на уровне управления линией передачи данных или из-за того, что пакет может измениться в памяти узла в результате аппаратных неисправностей. Таким образом, в дополнение к анализу обычного случая, когда эти предположения выполняются, необходимо также учитывать последствия исключительного случая, когда эти предположения нарушаются. Следует также помнить, что обновление топологии является алгоритмом низкого уровня, на котором основывается правильная работа других алгоритмов.

Заметим, что не предполагается, что сеть все время остается связной. Алгоритм обновления топологии вместе с некоторым протоколом восстановления линий должен быть способен восстановить работоспособность сети из состояния, в котором все линии вышли из строя.

5.3.1. Лавинный алгоритм сети ARPANET

Лавинный алгоритм ранее был описан как алгоритм, в котором узел распространяет среди всех узлов обновляющее сообщение о топологии путем рассылки этого сообщения всем своим соседям, которые в свою очередь пересылают это сообщение своим соседям, и т. д. Обновляющие сообщения могут содержать не только информацию о статусе линии, но и любую другую информацию, используемую при маршрутизации. Например, в сети ARPANET каждое сообщение, возникающее в узле, содержит значения усредненной по времени задержки пакета для каждой линии, уходящей от данного узла. Временное усреднение происходит в течение каждого 10-секундного интервала. Интервалы между рассылками узлам обновляющих сообщений в ARPANET могут быть от 10 до 60 с в зависимости от того, сколь сильно изменились значения средних задержек на какой-либо одной линии. Кроме того, сообщение рассылается сразу же после обнаружения изменения статуса одной из уходящих от узла линий.

Серьезная проблема, с которой приходится сталкиваться в некоторых вариантах лавинного алгоритма, состоит в том, что он может потребовать очень большое число передач сообщений. Пример на рис. 5.30 демонстрирует это и показывает, что необходимо хранить достаточно информации как в обновляющих сообщениях, так и в узлах сети, для того чтобы каждое сообщение было передано каждым узлом лишь конечное число раз (желательно только один раз). В сети ARPANET все обновляющие сообщения для каждого узла отмечаются *порядковыми номерами*. Когда узел j получает сообщение, возникшее в некотором узле i , он выясняет, является ли его порядковый номер больше порядкового номера последнего сообщения, полученного от i . Если да, то это сообщение вместе с его порядковым номером записывается в память и затем его содержание пересылается всем соседям j , за исключением того узла, от которого это сообщение получено. В противном случае это сообщение сбрасывается. Считая, что поле для порядкового номера достаточно большое, можно быть уверенным в том, что переполнение поля порядкового номера никогда не произойдет в нормальных условиях. (При 48-битовом поле и одном обновлении в миллисекунду потребуется более 5000 лет, чтобы переполнение произошло.)

Использование порядковых номеров гарантирует, что каждое обновляющее сообщение о топологии будет передаваться каждым узлом своим соседям не более одного раза. Оно также позволяет различать новую и старую информации, о которых шла речь ранее в примере на рис. 5.29. Тем не менее использование порядковых номеров создает другие более тонкие проблемы в исключительных

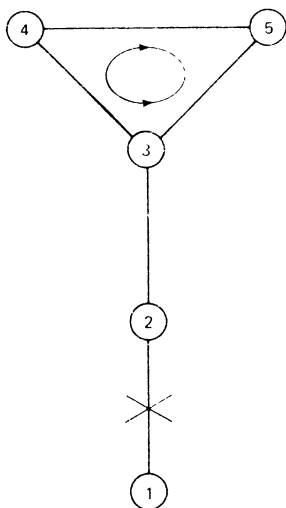


Рис. 5.30. Пример варианта лавинного алгоритма, в котором передача сообщений никогда не кончается. Правило состоит в том, что узел, получив сообщение, ретранслирует его всем своим соседним узлам, за исключением того, от которого это сообщение было получено. Сообщение о выходе из строя линии (1, 2) передается узлу 3 и затем оно начинает бесконечно долго циркулировать по петле (3, 4, 5) в обоих направлениях. Такой вариант лавинного алгоритма работает только тогда, когда сеть не имеет циклов.

ситуациях, таких как потеря связности сети или неисправная работа оборудования. Например, если приходится снова запускать в ход всю сеть или какую-либо ее часть (возможно, вследствие выхода из строя одного или нескольких узлов), то, может быть, придется снова установить некоторые порядковые номера рав-

ными нулю, так как узел может забыть порядковый номер, который он использовал в прошлом. Рассмотрим период времени, когда две части сети являются несвязными. В течение этого периода каждая часть сети не сможет следить за топологическими изменениями, происходящими в другой части. Если порядковые номера снова устанавливаются на нуль в одной из двух частей, пока они были несвязны, то для обеих частей может не оказаться другого способа определения правильной топологии после воссоединения кроме как положиться на механизм порядковых номеров. Еще одна проблема обусловлена тем, что порядковые номера могут искажаться либо в памяти узла вследствие каких-либо неисправностей, либо из-за необнаруженной ошибки при передаче. Предположим, например, что обновляющее сообщение, возникшее в узле i , достигнув узла j , случайно изменило свой порядковый номер на более высокий и затем распространилось по сети. Тогда ошибочный порядковый номер будет доминировать и если отсутствует какой-либо механизм коррекции, то все обновляющие сообщения из узла i будут игнорироваться до тех пор, пока их (правильные) порядковые номера не превысят ошибочный. Аналогично узел может отослать сообщение, порядковый номер которого случайно в результате ошибки оказался максимальным из возможных, и тогда, начиная с этого момента, все последующие сообщения об изменениях в этом узле будут игнорироваться сетью.

В сети ARPANET для преодоления проблем, связанных с порядковыми номерами, используются два правила.

1. Каждое обновляющее сообщение содержит *поле возраста*, в котором указывается время, в течение которого данное сообщение циркулирует по сети. Каждый узел, который сообщение посещает, запоминает момент прибытия этого сообщения и перед отправкой его своим соседям прибавляет в поле возраста то время, которое провело сообщение в этом узле (включая время его передачи и время распространения). (В действительности возраст сообщения в сети ARPANET увеличивается несколько иначе; см. [189].) Таким образом, узел в любой момент времени может вычислить возраст всех сообщений, хранящихся в его памяти. Сообщение, чей возраст превышает заранее установленный предел, дальше не передается. В противном случае сообщение передается всем соседним узлам, за исключением того узла, от которого это сообщение было получено.

2. Каждый узел должен *периодически* повторять передачу обновляющих сообщений в добавление к тем сообщениям, которые передаются в момент обнаружения изменения статуса линии (таких сообщений появляется не менее одного от каждого узла в течение каждых 60 секунд).

Используется также правило, касающееся поля возраста, состоящее в том, что независимо от порядковых номеров устаревшее сообщение всегда перекрывается тем, которое еще не устарело. Неустаревшее сообщение перекрывается только сообщением, порядковый номер которого больше. Это правило гарантирует, что на искаженную или неправильную информацию с большим порядковым номером не будут полагаться слишком долго. Использование периодического повторения передач обновляющих сообщений гарантирует, что после воссоединения двух частей сети информация о текущем состоянии сети станет известна спустя некоторый фиксированный интервал времени. Разумеется, периодическое повторение передач влечет за собой значительное увеличение объема передаваемой по сети информации, что является главным недостатком схемы ARPANET. Однако этот нежелательный эффект уменьшается из-за того, что в обновляющих пакетах содержится также другая, используемая при маршрутизации информация, а именно значения средних задержек пакетов в уходящих от узла линиях; эти задержки относятся к интервалу после предыдущего обновления.

5.3.2. Лавинный алгоритм без периодических обновлений

Можно организовать правильную работу схемы лавинной маршрутизации с порядковыми номерами без использования поля возраста или периодических обновлений. Следующая простая схема (предложенная П. Умбле) повышает надежность основной

идеи использования порядковых номеров посредством введения некоторого механизма борьбы с последствиями выхода из строя узлов и ошибок при передаче. Она требует, однако, чтобы поле порядковых номеров было достаточно большим, т. е. никогда не происходило его переполнение. Идея состоит в изменении лавинного алгоритма таким образом, чтобы удобнее было преодолевать трудности, возникающие после воссоединения несвязных компонент сети. Мы кратко опишем необходимые модификации, оставляя читателю самому восполнить некоторые детали. Наше внимание будет сосредоточено на случае распространения топологической информации, хотя подобные схемы можно разработать и для распространения любой другой маршрутной информации.

Как и раньше, будем считать, что узел передает своим соседям сообщение о статусах всех уходящих от него линий сразу после обнаружения изменения статуса какой-либо из уходящих от него линий. Сообщение маркируется порядковым номером, который либо равен нулю, либо на единицу больше последнего порядкового номера, использованного узлом в прошлом. Однако существует ограничение, состоящее в том, что нулевой порядковый номер разрешается использовать только тогда, когда узел только что восстановился после поломки (определяемой как ситуация, в которой все смежные с узлом линии вышли из строя и узел занимается восстановлением одной или нескольких из этих линий). Как и раньше, каждый узел, генерирующий обновляющие сообщения, имеет свои порядковые номера.

Первая модификация лавинного алгоритма касается восстановления вышедших из строя линий. Когда это происходит, концевые узлы линии, помимо обычных передач сообщений об обновлениях по всем уходящим от них линиям, должны обмениваться имеющимися у них представлениями о топологии сети. Под этим мы подразумеваем то, что они должны были переслать друг другу все хранящиеся у них в памяти обновляющие сообщения, генерированные другими узлами, вместе с их порядковыми номерами. Эти сообщения затем распространяются по сети в соответствии с правилами модифицированного лавинного алгоритма, описанного ниже. Эти правила гарантируют, что последняя обновляющая информация обязательно достигнет узлов в несвязных частях сети после их воссоединения. Они также устраняют трудности, возникающие в ситуациях, когда восстановленный после поломки узел забыл порядковый номер, который он использовал в прошлом. Такой узел после восстановления должен использовать нулевой порядковый номер и затем благодаря обмену топологиями со своими соседями, который всегда имеет место после восстановления смежных с ним линий, ему становится известен самый большой из порядковых номеров, хранящихся в памяти других узлов. Узел может затем увеличить этот самый большой порядковый номер

и передавать в сеть с помощью лавинного алгоритма новые обновляющие сообщения.

Чтобы схема работала правильно, в лавинном алгоритме изменено правило, когда обновляющее сообщение сбрасывается. Для этого для каждого узла i упорядочим обновляющие сообщения о топологии, генерируемые в узле i . О двух таких сообщениях A и B мы говорим, что $A > B$, если порядковый номер A больше чем B или если порядковые номера у A и B совпадают, но содержимое A больше содержимого B в соответствии с некоторым лексикографическим правилом (например, если содержимое A , интерпретируемое как двоичное число, больше чем аналогично интерпретируемое содержимое B). Таким образом, любые два сообщения A и B , генерированные в одном и том же узле, можно сравнить между собой, чтобы определить $A > B$, $B > A$ или $A = B$. Последнее соотношение имеет место только тогда, когда и порядковые номера, и содержимое A и B идентичны.

Предположим, что узел j получил обновляющее сообщение A , генерированное в узле i , а в это время в памяти узла j хранится генерированное в узле i сообщение B . Сообщение A сбрасывается, если $A < B$ или $A = B$. Если $A > B$, то лавинный алгоритм руководствуется теперь следующим правилом.

1. Если $j \neq i$, то узел j записывает A вместо B в своей памяти и пересылает копию A по всем своим смежным линиям, за исключением той, по которой пришло A .

2. Если $j = i$ (т. е. узел i получил сообщение, которое он сам генерировал ранее), то узел i посылает всем своим соседям обновляющее сообщение, содержащее текущие статусы всех своих уходящих линий вместе с порядковым номером, превышающим порядковый номер A на 1.

Для того чтобы увидеть, почему необходимо сравнивать содержимое A и B , в случае, когда их порядковые номера совпадают, рассмотрим следующий пример. Пусть три узла 1, 2 и 3 соединены в тандем двумя (неориентированными) линиями (1,2) и (2,3). Предположим, что вначале обе линии функционируют исправно и сообщения, хранящиеся в памяти всех узлов, содержат правильную информацию о статусе линий и имеют порядковые номера, равные нулю. Рассмотрим сценарий, согласно которому выходит из строя линия (2,3), затем линия (1,2), а после этого линия (2,3) восстанавливается и узел 2 снова устанавливает свой порядковый номер равным нулю. В этих условиях узлы 2 и 3 обмениваются своими (противоречивыми) сведениями о статусе ориентированных линий (1,2) и (2,1), но если бы использовался прежний лавинный алгоритм, то оба узла сбросили бы эти обновляющие сообщения, полученные друг от друга, так как у этих сообщений нулевые порядковые номера, т. е. такие же, как у сообщений, хранящихся в их памяти. Однако если следовать моди-

фицированному лавинному алгоритму, описанному выше, то видно, что (правильные) сведения узла 2, касающиеся линии (2,1), либо сразу (в зависимости от того, какой лексикографический порядок был установлен) вытеснят неправильную информацию из памяти узла 3, либо после того, как узел 2 пошлет обновляющее сообщение с порядковым номером 1. Заметим, что узел 3 может проработать до очередной поломки, располагая неверной информацией о статусе линии (1,2). Это может произойти в случае если все это время узлы 1 и 3 не будут соединены путем из действующих линий. По этой же причине, однако, то, что в узле 3 хранится неверная информация об уходящей от узла 1 линии, не имеет значения.

Рассмотренный в этом разделе алгоритм имеет недостаток, который присущ, по-видимому, всем вероятным алгоритмам, которые не используют периодические обновления, а именно: он чувствителен к ошибкам, возникающим в памяти и при передаче. Если, например, порядковый номер в результате ошибки при передаче или хранении в памяти изменился и стал очень большим, то может случиться так, что все последующие генерируемые узлом сообщения будут игнорироваться до тех пор, пока их порядковые номера не превысят ошибочный номер, хранящийся в памяти некоторых узлов. Оказывается, что этот недостаток можно устранить путем модификации обсуждавшегося выше лавинного алгоритма. Суть модификации состоит в том, что если узел j получает A , а в его памяти хранится B , такое, что $A < B$, то сообщение A , как и раньше, сбрасывается, но в дополнение к этому узел посылает сообщение B обратно тому соседу, от которого было получено сообщение A . (Сосед затем разошлет B дальше по сети.) Предположим теперь, что узел разошлет обновляющее сообщение с порядковым номером, меньшим, чем порядковый номер, хранящийся в некотором другом узле. Тогда благодаря только что описанной модификации, если два узла можно соединить путем из действующих линий, то больший порядковый номер (скажем, k) обязательно вернется в узел, сгенерировавший его, который сможет затем сгенерировать обновляющее сообщение с порядковым номером $k + 1$ в соответствии с описанным выше правилом 2. Однако попрежнему остается проблема, связанная с возможностью переполнения поля порядкового номера (например, если k из рассмотренного только что примера является максимальным числом, которое можно записать в поле порядкового номера). Остается и проблема, связанная с искажением обновляющих сообщений (а не только его порядкового номера) в результате ошибок при передаче или хранении в памяти. По-видимому, простым способом решения этой проблемы является применение специально предназначенных для устранения подобных ошибок схем, использующих поле возраста и периодические обновления (см. задачу 5.13).

5.3.3. Распространение топологии без использования порядковых номеров

В этом подразделе рассматривается алгоритм, аналогичный лавинному, который называется топологическим алгоритмом кратчайшего пути (ТАКП). Этот алгоритм в отличие от предыдущих схем не использует порядковые номера и, следовательно, не имеет проблем, связанных с переполнением и восстановлением порядковых номеров. ТАКП решает проблему различения старой и новой информации довольно интересным способом. Всегда, когда в узел поступает от двух разных соседей противоречивая информация о статусе какой-либо линии, он устраняет это противоречие тем, что отдает предпочтение той информации, которую считает наиболее надежной. Степень надежности имеющейся информации рассматривается в момент поступления новой информации, и имеющаяся информация сбрасывается либо когда ее вытесняет новая информация, либо когда она признается ненадежной. Это неформальное описание вскоре будет уточнено, но, по существу, надежность информации измеряется числом линий до места ее генерирования и вычисляется методом, аналогичным вычислению кратчайшего пути. В течение всего этого подраздела будут делаться те же, что и для лавинного алгоритма, предположения о сохранении порядка при передаче по линиям, отсутствии ошибок при передачах и о том, что узлы сами могут обнаружить изменение статуса любой смежной ему линии.

Данные, хранящиеся в памяти каждого узла i , в ТАКП имеют следующую структуру (рис. 5.31).

1. *Основная топологическая таблица T_i* , в которой узел i хранит статус каждой линии, который он сам считает наиболее правдоподобным. Это официальная таблица, используемая алгоритмом маршрутизации в данном узле. Основные топологические таблицы разных узлов могут содержать разные записи в различные моменты времени. Целью ТАКП является то, чтобы все эти таблицы содержали согласованную информацию спустя конечное время после последнего изменения топологии.

2. *Портовые топологические таблицы T_j^i* . Такая таблица имеется для каждого соседнего узла j . В T_j^i узел i записывает последнюю информацию о статусе каждой линии сети, которую он получает от узла j .

Алгоритм состоит из пяти простых правил.

Правила обмена информацией.

1. Как только изменяется какая-либо запись о статусе линии в основной топологической таблице узла, новая запись сразу же передается по всем смежным с данным узлом действующим линиям.

2. Когда вышедшая из строя линия начинает снова функционировать, каждый из ее концевых узлов сразу же передает все со-

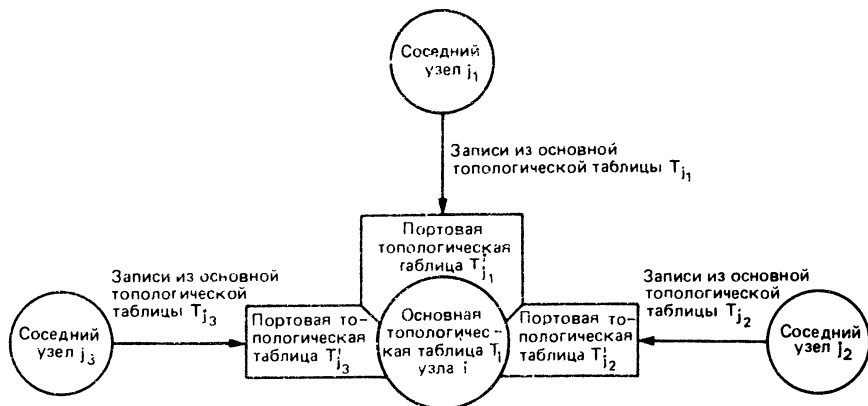


Рис. 5.31. Структурами данных в ТАКП в узле i являются основная топологическая таблица T_i и портовые топологические таблицы $T_{j_1}^i$, $T_{j_2}^i$ и $T_{j_3}^i$. Изменения в основных топологических таблицах соседних узлов j_1 , j_2 и j_3 переносятся с некоторой задержкой в соответствующие портовые таблицы узла i , за исключением тех изменений, которые относятся к линии, смежной с i . Такие изменения вносятся непосредственно в основную и портовые топологические таблицы. Как только какая-либо смежная вышедшая из строя линия начинает функционировать, узел i сразу же передает по этой линии содержимое своей основной топологической таблицы. И наконец, когда вносится изменение в портовую топологическую таблицу, содержимое основной топологической таблицы пересматривается в соответствии с алгоритмом обновления основной топологической таблицы.

содержимое своей основной топологической таблицы узлу, находящемуся на противоположном конце этой линии. После того, как эта таблица принята, противоположный узел записывает новый статус этой линии в основную и портовые топологические таблицы.

Правила обновления топологических таблиц

3. Как только какая-либо смежная с узлом линия выходит из строя, статус о поломке линии вносится в основную и портовые топологические таблицы узла.

4. Узел, получив сообщение об изменении статуса линии от своего соседа, сразу же вносит это сообщение в портовую топологическую таблицу, соответствующую этому соседу.

5. Как только изменяется какая-либо запись либо в основной топологии T_i вследствие изменения статуса смежной линии, либо в портовой топологии $T_{j_1}^i$ вследствие получения новой информации от соседнего узла j , узел i сразу же обновляет свою основную топологическую таблицу, используя для этого следующий алгоритм. (Предполагается, что, если во время выполнения алгоритма из-

меняется статус смежной линии или приходит новое сообщение от соседнего узла, работа алгоритма прекращается и затем он начинает выполняться с самого начала.) Идея алгоритма состоит в том, чтобы узел i считал достоверной ту информацию о статусе любой линии l , которая была получена от соседнего узла, лежащего на кратчайшем пути от узла i до линии l . При вычислении кратчайшего пути считается, что длина действующей линии равна единице, а длина вышедшей из строя линии равна бесконечности.

Алгоритм обновления основной топологической таблицы в узле i .

Алгоритм состоит в выполнении итераций, аналогичных итерациям алгоритма отыскания кратчайшего пути Дijkstra, в котором все линии имеют единичную длину (подразд. 5.2.3). В начале k -й итерации имеется множество узлов P_k . Каждый узел в P_k содержит метку, означающую номер одного из соседних к i узлов. Узлы в P_k — это узлы, которых можно достичь из i , проходя по пути, состоящему из не более чем k линий, которые считаются действующими на основании содержимого основной топологической таблицы узла i в момент начала k -й итерации. Метка узла $m \in P_k$ означает номер соседнего к i узла, лежащего на пути от i к m , состоящем из минимального числа (действующих) линий. В частности, P_1 состоит из узлов, соединенных с i действующей линией; каждый узел в P_1 помечен номером. В течение k -й итерации либо в P_k добавляются какие-либо новые узлы и образуется P_{k+1} , либо алгоритм на этом заканчивает свою работу. Одновременно с этим статусы всех линий, содержащих хотя бы один концевой узел из P_k , но не содержащих ни одного из своих концевых узлов из P_{k-1} , т. е. линий из множества

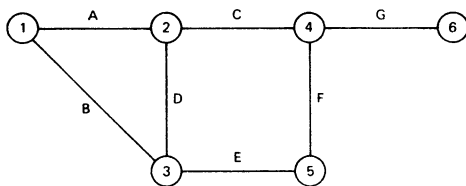
$$L_k = \{(m, n) \mid m \notin P_{k-1}, n \notin P_{k-1},$$

m или n (или оба) принадлежат $P_k\}$,

записываются в основную топологическую таблицу T_i . (Для полноты обозначений положим $P_0 = \{i\}$ в уравнении для L_1 .) Суть k -й итерации состоит в следующем.

Шаг 1. Для каждой линии $(m, n) \in L_k$ делаем следующее. Пусть (без ограничения общности) m — концевой узел, принадлежащий P_k , а j — метка для узла m . Берем копию статуса (m, n) из портовой топологической таблицы T_j^i и записываем в основную топологическую таблицу T_i . Если это статус действующей линии и $n \notin P_k$, то приписываем узлу n метку j .

Шаг 2. Пусть M_k — множество узлов, которым приписаны метки на шаге 1. Если M_k пусто, то на этом алгоритм работу заканчивает. В противном случае полагаем $P_{k+1} = P_k \cup M_k$ и переходим к $(k+1)$ -й итерации.



(a)

$$P_1 = \{2, 3\}, \quad P_2 = \{2, 3, 5\}, \quad P_3 = \{2, 3, 5, 4\}, \quad P_4 = \{2, 3, 5, 4, 6\}$$

$$L_1 = \{D, C, E\}, \quad L_2 = \{F\}, \quad L_3 = \{G\}$$

(б)

Линия	Портовая топологическая таблица T_2^1	Портовая топологическая таблица T_3^1	Рассто- яние	Доверяет узлу	Финальная основная то- пологическая таблица узла 1
A	—	—	0	—	U
B	—	—	0	—	U
C	D	U	1	2	D
D	U	U	1	2,3	U
E	U	U	1	3	U
F	U	U	2	3	U
G	D	U	3	3	U

(в)

Рис. 5.32. Иллюстрация алгоритма обновления основной топологической таблицы в узле 1 сети (а). Множества узлов P_k и множества линий L_k представлены в (б). Содержимое портовых топологических таблиц T_2^1 и T_3^1 в узле 1 приведено в таблице (в), где U означает «действует», D — «вышла из строя». Имеется противоречивая информация о статусах линий C и G . Узел 1 доверяет узлу 2 о статусе линии C , так как узел 2 находится ближе к этой линии. Узел 1 доверяет узлу 3 о статусе линии G , так как узел 3 находится ближе к этой линии (учитывая, что линия C объявлена вышедшей из строя узлом 1).

Работа этого алгоритма проиллюстрирована на рис. 5.32. Так как каждая линия обрабатывается только один раз на шаге 1 алгоритма, ясно, что объем вычислений пропорционален числу линий. Непосредственно проверяются следующие свойства алгоритма обновления основной топологии.

1. Множество P_k для $k \geq 1$ является множеством узлов m , таких, что в окончательно полученной топологии T_i существует путь не более чем из k действующих линий, соединяющий m с i ,
2. Множество L_k , $k \geq 1$, является множеством линий l , таких.

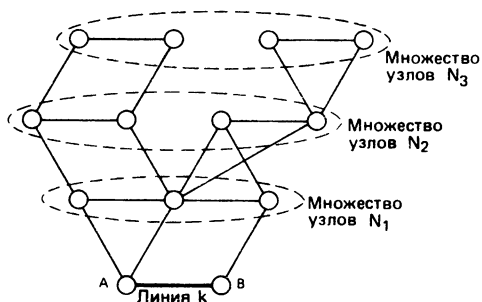


Рис. 5.33. Работа ТАКП для единственного изменения статуса линии. Информация об изменении статуса распространяется среди узлов множества N_1 (на расстоянии одной линии от A или B или от обоих сразу), затем среди узлов из N_2 (на расстоянии двух линий от A или B или от обоих сразу), затем среди N_3 и т. д. В частности, когда меняется статус линии k , это изменение вносится в топологические таблицы узлов A и B и передается узлам из N_1 . Оно вносится в портовые топологические таблицы T_A^i и (или) T_B^i узлов i из N_1 . Так как расстояние до линии k через узлы $j \neq A, B$ больше нуля, то новый статус линии k обязательно будет внесен в основные топологические таблицы каждого узла из N_1 . Затем он будет передан узлам из N_2 и внесен в соответствующие портовые топологические таблицы и т. д. Необходимое число передач сообщений не превысит $2L$, где L — число неориентированных линий в сети.

что в окончательно полученной топологии T_i все пути из действующих линий, соединяющие один из концевых узлов линии l с узлом i , имеют не менее k линий и существует один такой путь с k линиями.

3. Окончательная запись в T_i для линии $l \in L_k$ совпадает с записью из портовой топологии соседнего узла, лежащего на пути из k действующих линий, соединяющем i с одним из концевых узлов линии l . Если существуют два соседа с противоречащими записями, то алгоритм произвольным образом выбирает запись одного из них.

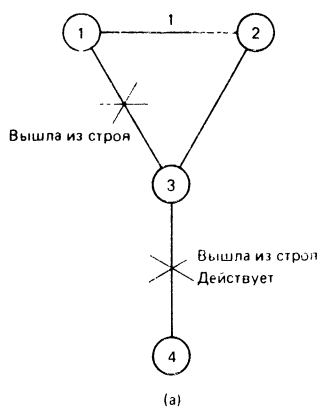
4. Когда алгоритм обновления основной топологии выполняется в узле i в ответ на выход из строя смежной линии (i, j) , информация, содержащаяся в портовой топологической таблице T_j^i , фактически игнорируется. Это происходит вследствие того, что узел j не включен в множество P_1 и поэтому ни одному из узлов не будет присвоена метка j в течение выполнения алгоритма.

Работа ТАКП для случая одного изменения статуса линий показана на рис. 5.33. В этом случае алгоритм работает, грубо говоря, как лавинный, но нет необходимости в нумерации обновляющих сообщений для того, чтобы ограничить число передач сообщений. Рис. 5.34 иллюстрирует работу ТАКП для случая многократных изменений статусов линий и показывает, каким образом ТАКП справляется с задачей различения новой и устаревшей информации (см. рис. 5.29). Основная идея здесь состоит

Рис. 5.34. Работа ТАКП по сценарию изменения топологии из рис. 5.29. Этот пример демонстрирует, каким образом ТАКП справляется с ситуациями, в которых лавинный алгоритм не работает.

(а) Пример сети с четырьмя линиями и четырьмя узлами. Сначала все линии действуют. Происходят три изменения: (1) линия (3, 4) выходит из строя, (2) линия (3, 4) начинает функционировать и (3) линия (1, 3) выходит из строя. (б) Предполагаемая последовательность событий из передач и приемов сообщений в ТАКП. Предполагается, что обновление топологических таблиц происходит мгновенно вслед за событием. (в) Содержимое основных и портовых топологических таблиц каждого узла после каждого события. U и D означают «действует» и «вышла из строя» соответственно. Например, содержимое таблицы $UUUD$ означает, что линии (1, 2), (2, 3) и (1, 3) действуют, а линия (3, 4) вышла из строя.

Содержимое таблиц, изменившееся в результате события, выделено жирным шрифтом. Заметим, что последнее событие меняет в T_1 запись не только о линии (1, 3), но также запись о линии (3, 4). Также после последнего события содержимое портовой топологической таблицы T_3^j не верно, но это не существенно, так как соединяющая линия (1, 3) вышла из строя.



в том, что, когда линия (i, j) выходит из строя, вся информация, приходящая по этой линии, по существу, игнорируется, как это обсуждалось ранее.

Теперь представим доказательство корректности ТАКП. В любой момент времени истинную топологическую таблицу (которую знает лишь всеведущий наблюдатель) обозначим через T^* . Мы говорим, что в этот момент *линия l соединена с узлом i* , если существует путь, соединяющий i с одним из концевых узлов линии l и состоящий из линий, которые являются действующими в соответствии с T^* . В данный момент узел j является *действующим соседом* узла i , если линия (i, j) является действующей в соответствии с T^* . В данный момент *узел i знает правильную топологию*, если основная топологическая таблица T_i согласуется с T^* относительно статусов всех линий, соединенных с i . Предположим, что в некоторый начальный момент времени t_s каждый узел знает правильную топологию (например, в момент перезапуска сети, когда все линии не функционировали) и все действующие портовые топологии T_i^j полностью согласуются с соответствующими основными топологиями T_j . Пусть после этого момента несколько линий изменили свой статус. Но существует такой момент времени t_0 , после которого ни одна из линий не меняет свой статус и концевые узлы каждой линии знают правильный статус линии. Мы покажем следующее.

Номер события	1	2	3
Событие	Линия (3, 4) выходит из строя. Сообщение передается от 3 к 1 и 2. Сообщение получено в 2	Сообщение передается от 2 к 1. Сообщение получено в 1	Линия (3, 4) начинает функционировать. Сообщение передается от 3 к 1 и 2. Сообщение получено в 2
Номер события	4	5	6
Событие	Сообщение передается от 2 к 1. Сообщение получено в 1	Приходит сообщение от 3 к 1, отосланное в событии 1	Линия (1, 3) выходит из строя. Сообщения от 1 и 3 отсылаются к 2 и 4. Сообщения получены в 2 и 4

(б)

Номер события	1	2	3	4	5	6
Таблицы T_1 в узле 1 T_1^1 T_1^2	UUUU UUUU UUUU	UUUU UUUD UUUU	UUUU UUUD UUUU	UUUU UUUU UUUU	UUUD UUUU UUUD	UUUD UUUD UUDD
Таблицы T_2 в узле 2 T_2^1 T_2^2	UUUD UUUU UUUD	UUUD UUUU UUUD	UUUU UUUU UUUU	UUUU UUUU UUUU	UUUU UUUU UUUU	UUUD UUUD UUUD
Таблицы T_3 в узле 3 T_3^1 T_3^2 T_3^4	UUUD UUUD UUUD UUUD	UUUD UUUD UUUD UUUD	UUUU UUUU UUUU UUUU	UUUU UUUU UUUU UUUU	UUUU UUUU UUUU UUUU	UUUD UUUD UUUD UUUD
Таблицы T_4 в узле 4 T_4^1 T_4^2	UUUD UUUD	UUUD UUUD	UUUU UUUU	UUUU UUUU	UUUU UUUU	UUUD UUUD

(в)

Рис. 5.34 (продолжение).

Утверждение. ТАКП работает правильно в том смысле, что при сделанных предположениях обязательно найдется такой момент времени $t_f \geq t_0$, после которого каждый узел будет знать правильную топологию для всех $t \geq t_f$.

Доказательство. Далее будем говорить, что линия l находится на расстоянии n от i , если в графе, которой задается T^* , кратчайший путь от i до ближайшего конечного узла линии l состоит из n

линий. Покажем по индукции, что для любого целого $n \geq 0$ найдется такой момент времени $t_n \geq t_0$, после которого для каждого узла i таблица T_i согласуется с T^* по статусам всех линий, находящихся на расстоянии не более чем n от i . Предположение индукции, очевидно, выполняется при $n = 0$, так как каждый узел i знает правильные статусы всех смежных с ним линий и хранит их в своей основной топологической таблице T_i . Сначала докажем следующую лемму.

Лемма. Будем считать, что предположение индукции выполняется для момента t_n . Тогда обязательно найдется момент времени $t'_{n+1} \geq t_n$, после которого портовая топологическая таблица T'_j для каждого действующего соседа j узла i согласуется с T^* по статусам линий, находящихся на расстоянии не более чем n от j .

Доказательство. Пусть после t_n пройдет достаточное количество времени, для того чтобы все сообщения из j и i , отправленные до момента t_n , успели прибыть в i . В соответствии с правилами 1, 2 и 4 алгоритма T'_j согласуется с T_j относительно статусов всех линий, несмежных с i . Следовательно, в силу предположения индукции T'_j согласуется с T^* по статусам всех линий, находящихся на расстоянии не более чем n от j . Это доказывает лемму.

Для завершения доказательства утверждения осталось показать, что найдется такой момент времени $t_{n+1} \geq t'_{n+1}$, после которого для любого $t \geq t_{n+1}$ и каждого узла i таблица T_i согласуется с T^* по статусам всех линий l , находящихся на расстоянии $(n + 1)$ от i . Рассмотрим первый момент времени, когда линия l просматривается алгоритмом обновления топологии, после того как все портовые топологии T'_j уже согласованы с T^* по статусам всех линий, находящихся на расстоянии не более чем n от j , в соответствии с предыдущей леммой. В этот момент линия l будет принадлежать множеству L_{n+1} , а ближайший к узлу i концевой узел (узлы) линии l будет принадлежать множеству P_{n+1} и иметь метку, относящуюся к одному из действующих соседей i (скажем, метку j), находящемуся на расстоянии n от i . В соответствии с леммой портовая топологическая таблица T'_j согласуется с T^* по статусам всех линий, находящихся на расстоянии n от j . Поэтому запись о линии l в T'_j , которая затем будет скопирована в T_i при просмотре линии l , будет согласована с соответствующей записью из T^* . Так как утверждение леммы выполняется для всех t после t'_{n+1} , то запись о линии l в T_i будет также верной во все последующие моменты времени, в которые линия l будет просматриваться алгоритмом обновления основной топологии. На этом доказательство заканчивается.

Заметим, что приведенная выше лемма утверждает, что действующие портовые топологические таблицы в конце концов также

будут согласованы с соответствующими основными топологическими таблицами. Отсюда следует, что начальные условия, для которых была доказана корректность алгоритма, восстановятся после достаточно большого периода времени, в течение которого не происходило никаких топологических изменений.

Основным преимуществом ТАКП по сравнению с алгоритмом лавинной маршрутизации ARPANET является то, что он не использует порядковые номера и, следовательно, не имеет проблем, связанных с переполнением; он также не использует поле возраста, размер которого зависит от сети и изменяется по мере расширения сети. Кроме того, ТАКП действует в ответ на появление случайных событий и поэтому не создает больших дополнительных нагрузок в сети, вызванных регулярными периодическими обновлениями передачами, как в алгоритме ARPANET.

Рассмотрим теперь, как будет вести ТАКП, если могут возникать ошибки при передаче или хранении в памяти узлов. Как и все другие алгоритмы, приводящиеся в действие случайными событиями, ТАКП не способен исправлять эти ошибки. Без периодического повторения передач необнаруженная ошибка может сколь угодно долго находиться в сети неисправленной. Следовательно, в ситуациях, когда к надежности предъявляются очень большие требования, узел должен периодически повторять передачу своей основной топологической таблицы каждому из своих соседей. Это приводит к утрате у ТАКП свойства действовать в ответ на появление случайных событий, но зато сохраняет другие желательные характеристики и позволит устранять в базах данных необнаруженные ошибки.

Наконец, заметим, что ТАКП можно использовать для распространения по сети информации не только о статусе линий. Например, при использовании алгоритма адаптивной маршрутизации может возникнуть необходимость в распространении по сети значений средних задержек или интенсивностей приходящего в каждую линию трафика. Для этого нужно, чтобы в основную и портовые топологические таблицы ТАКП в дополнение к статусам линий входила дополнительная, возможно, не двоичная информация. Если эта дополнительная информация зависит от ориентации линии, то следует незначительно модифицировать ТАКП так, чтобы в топологических таблицах содержались отдельные записи для каждого направления линии. Такая модификация рассматривается в задаче 5.14.

5.4. Потокосовые модели, оптимальная маршрутизация, построение топологии

Для того чтобы оценить характеристики алгоритма маршрутизации, необходимо количественно определить понятие перегрузки в сети. В этом разделе формулируются некоторые модели (назы-

ваемые в дальнейшем *потокowymi моделями*), основанные на интенсивностях трафика, поступающего в линии сети. Эти модели будут использоваться при постановке задачи оптимальной маршрутизации, которая является темой разделов 5.5—5.7. Мы также будем использовать потокowe модели при обсуждении построения различных частей сетевой топологии в подразд. 5.4.1—5.4.3.

Перегрузки в сетях передачи данных количественно можно определить с использованием статистик входных процессов, описывающих очереди в сети. Эти статистики определяют распределение длины очереди и время ожидания пакета в каждой линии. Очевидно, что при хорошей маршрутизации должны быть малы средние значения и дисперсии задержки пакета в каждой очереди. К сожалению, обычно бывает трудно представить количественный параметр в виде единственного функционала, подходящего для оптимизации. Главной причиной этого является то, что, как видно из гл. 3, как правило, нет явного аналитического выражения для средних значений или дисперсий длин очередей в сетях передачи данных.

Удобной, хотя и не совсем совершенной альтернативой является измерение нагрузок в линиях с использованием *среднего* трафика, проходящего по линии. Точнее говоря, мы предполагаем, что статистика потока, поступающего в любую линию (i, j) , меняется только из-за обновлений маршрутов, и под нагрузкой в линии (i, j) мы будем понимать интенсивность поступающего трафика F_{ij} . Будем называть F_{ij} *потокom*, проходящим по линии (i, j) , и измерять его в единицах данных в секунду, где единицами данных могут быть биты, пакеты, сообщения и т. д. Иногда имеет смысл измерять поток в единицах, которые, как предполагается, прямо пропорциональны единицам данных в секунду, например, в числе виртуальных цепей, проходящих по данной линии.

В потокowych моделях делается неявное предположение, что статистика трафика, поступающего в сеть, не меняется во времени. Такое допущение является разумным, когда эта статистика меняется очень медленно по сравнению со средним временем, необходимым для уменьшения очередей в сети, и когда потоки в линиях измеряются путем временного усреднения. Типичным примером сети, удовлетворяющей таким условиям, является сеть, в которой каждая пара отправитель—адресат предназначена для большого числа пользователей, а интенсивность трафика у каждого из этих пользователей мала по сравнению с суммарной интенсивностью (см. подразд. 3.4.2).

Выражение вида

$$\sum_{(i,j)} D_{ij}(F_{ij}), \quad (5.29)$$

где каждая функция D_{ij} является монотонно возрастающей, часто выбирают в качестве стоимостной функции при оптимизации. Часто используется следующая формула:

$$D_{ij}(F_{ij}) = \frac{F_{ij}}{C_{ij} - F_{ij}} + d_{ij}F_{ij}, \quad (5.30)$$

где C_{ij} — пропускная способность линии (i, j) , измеряемая в тех же единицах, что и F_{ij} , а d_{ij} — задержка из-за обработки и распространения. При этом выражение (5.29) означает среднее число пакетов в системе в предположении, что каждая очередь описывается моделью $M/M/1$; это соответствует клейнроковской аппроксимации независимостью и теореме Джексона, обсуждавшихся в разд. 3.6 и 3.8. Хотя это предположение обычно не выполняется на практике, выражения (5.29) и (5.30) очень полезны, так как они качественно отражают факт возникновения перегрузки, когда поток F_{ij} приближается по своему значению к соответствующей пропускной способности линии C_{ij} . Другой стоимостной функцией с аналогичными качественными свойствами является

$$\max_{(i, l)} \left\{ \frac{F_{ij}}{C_{ij}} \right\}, \quad (5.31)$$

т. е. максимум коэффициента использования линии. Числовые результаты [247] показывают, что обычно нет большого различия между случаями использования стоимостных функций (5.29)—(5.30) или (5.31) при оптимизации маршрутизации. Это показывает, что следует выбирать ту стоимостную функцию, которую легче оптимизировать. В дальнейшем мы сосредоточим внимание на стоимостной функции вида (5.29). Анализ и численные методы из разд. 5.5—5.7 применимы к стоимостным функциям более общего вида (см. задачу 5.27).

Теперь сформулируем задачу оптимальной маршрутизации. Для каждой пары $\omega = (i, j)$ различных узлов i и j (также называемой парой отправитель—адресат или ОА-пара) входной процесс поступающих пакетов предполагается стационарным и имеет интенсивность r_ω ¹⁾. Таким образом, r_ω — интенсивность входного трафика (измеряемая в единицах данных в секунду), поступающего в сеть в узле i и адресованного узлу j . Цель маршрутизации состоит в том, чтобы трафик интенсивности r_ω разделить между несколькими путями от отправителя к адресату так, чтобы

¹⁾ Иногда удобно расширить понятие ОА-пары и рассматривать ее как класс пользователей, которые используют один и тот же набор путей. Это позволяет моделировать ситуации, когда пользователи делятся на несколько классов по приоритетам. В этом контексте можно допустить, что несколько ОА-пар имеют одну и ту же пару узлов отправитель—адресат. Последующий анализ и алгоритмы почти дословно применимы в этом более общем случае. Однако для простоты последующего изложения будем предполагать, что каждой паре узлов соответствует не более одной ОА-пары.

как показано на рис. 5.35. Суммарный поток F_{ij} по линии (i, j) равен сумме путевых потоков, проходящих по этой линии,

$$F_{ij} = \sum_{\substack{\text{по всем путям } p, \\ \text{содержащим } (i, j)}} x_p$$

Рассмотрим стоимостную функцию вида

$$\sum_{(i, j)} D_{ij}(F_{ij}) \quad (5.32)$$

и задачу нахождения путевых потоков $\{x_p\}$, которые минимизируют эту стоимостную функцию при вышеуказанных ограничениях.

Если в стоимостной функции (5.32) заменить суммарные потоки F_{ij} на путевые, то задачу можно сформулировать в следующем виде:

$$\begin{aligned} &\text{минимизировать} \quad \sum_{(i, j)} D_{ij} \left[\sum_{\substack{\text{по всем путям } p, \\ \text{содержащим } (i, j)}} x_p \right] \\ &\text{при ограничениях} \quad \sum_{p \in P_w} x_p = r_w \quad \text{для всех } w \in W, \\ &\quad \quad \quad x_p \geq 0 \quad \text{для всех } p \in P_w, w \in W. \end{aligned} \quad (5.33)$$

Таким образом, задача сформулирована в терминах неизвестных путевых потоков $\{x_p \mid p \in P_w, w \in W\}$. Это и является основной задачей оптимальной маршрутизации, которая будет рассмотрена. Описание оптимального решения этой задачи дано в разд. 5.5, где показано, что оно выражается в терминах кратчайших путей. Алгоритмы нахождения решения представлены в разд. 5.6 и 5.7.

Только что сформулированная задача оптимальной маршрутизации поддается аналитическому исследованию и распределенному численному решению. Однако она имеет некоторые ограничения, которые стоит объяснить. Основное ограничение касается выбора стоимостной функции (5.29) в качестве меры. Этот выбор основан на гипотезе, что достаточно хорошую маршрутизацию можно сделать, оптимизируя средние уровни проходящего по линиям трафика и не обращая внимания на другие вероятностные характеристики трафика. Таким образом, стоимостная функция (5.29) не чувствительна к нежелательным явлениям, связанным с большой дисперсией и корреляциями интервалов между моментами поступления пакетов и моментами передачи. В качестве иллюстрации этого рассмотрим пример из разд. 3.6, в котором узел A посылает пуассоновский трафик узлу B по двум линиям с одинаковыми пропускными способностями. Мы сравнивали два способа разделения трафика между двумя линиями: с использованием

рандомизации и на основе измерения. Был сделан вывод: измерения лучше рандомизации по средней задержке пакета. Однако рандомизация и измерения дают одинаковые значения стоимостной функции (5.29), так как приводят к одинаковым потокам. Дело в том, что задержка на каждой линии зависит от второго и более высоких моментов входного процесса, а стоимостная функция (5.29) отражает зависимость только от первого момента.

Так как при измерениях необходимо следить за текущими значениями длин очередей, то предыдущий пример показывает, что улучшить маршрутизацию можно, используя информацию о длинах очередей. Вот еще один пример, иллюстрирующий то же самое.

Пример 1

Рассмотрим сеть, изображенную на рисунке 5.36. Имеются три пары отправитель—адресат, для каждой из которых интенсивность поступающего трафика равна 1 единице. ОА-пары (2, 4) и (3, 4) направляют свой трафик только по линиям (2, 4) и (3, 4) соответственно. Так как все линии имеют пропускную способность 2 единицы, то оба пути (1, 2, 4) и (1, 3, 4) одинаково привлекательны для трафика ОА-пары (1, 4). Оптимальный алгоритм маршрутизации, основанный на усредненных интенсивностях проходящего по линиям трафика, поочередно посылает пакеты ОА-пары (1, 4) по этим двум путям, в результате чего полный поток по линиям (2, 4) и (3, 4) станет равным 1,5 единицы. Сеть будет работать достаточно хорошо, если трафик ОА-пар (2, 4) и (3, 4) близок к пуассоновскому. Рассмотрим, однако, другую ситуацию, в которой трафик поочередно в течение достаточно длинных временных интервалов имеет интенсивность то 2 единицы,

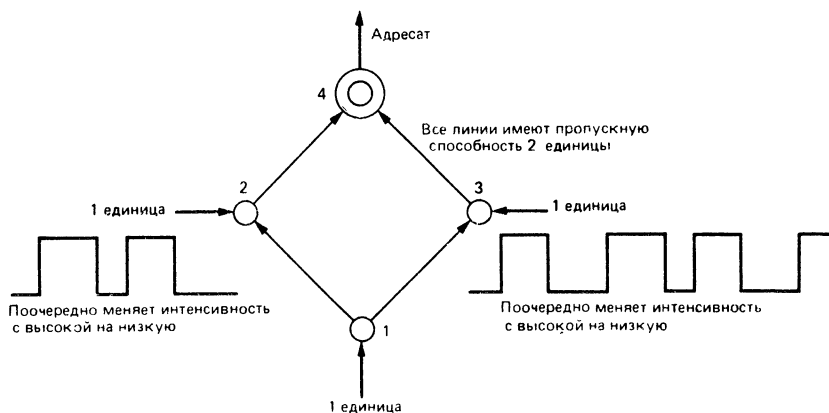


Рис. 5.36. Сеть из примера 1, для которой маршрутизация, основанная на информации о длине очереди, более предпочтительна. Интенсивность трафика, возникающего в узлах 2 и 3, поочередно меняется с 2 единиц до 0 единиц и обратно в течение продолжительных временных интервалов. При хорошей маршрутизации собирается информация о размерах очередей в линиях (2, 4) и (3, 4) и трафик, возникающий в узле 1, направляется по пути с наименьшей длиной очереди.

то 0 единиц. Тогда очереди на линиях (2, 4) и (3, 4) будут то увеличиваться (не обязательно одновременно) в течение одних длинных интервалов, то уменьшаться в течение других. Предположим теперь, что принята более динамичная форма маршрутизации, при которой узел 1 постоянно имеет информацию о текущих длинах очередей на линиях (2, 4) и (3, 4) и направляет свои пакеты по пути с наименьшей очередью. Задержка в очередях в этом случае станет значительно меньше.

К сожалению, в больших сетях практически невозможно сделать так, чтобы все узлы были информированы о длинах очередей. Даже если возникающие при этом дополнительные нагрузки не превышают допустимых пределов, задержка при передаче узлами информации о длинах очередей может привести к тому, что эта информация станет уже устаревшей. В настоящее время неизвестно, как провести эффективную и реализуемую на практике маршрутизацию, основанную на информации о длинах очередей, и поэтому в дальнейшем мы не будем касаться этой темы. В качестве интересной, но не апробированной альтернативы см. задачу 5.36.

В остальной части этого раздела рассматриваются использование сетевых алгоритмов, потоковых моделей и задачи оптимальной маршрутизации в контексте построения топологии сети. Читатель может пропустить этот материал без ущерба для понимания следующего.

5.4.1. Обзор задач построения топологии

В этом подразделе мы обсудим алгоритмы построения топологии сети передачи данных. Суть рассматриваемой задачи состоит в том, что при заданных потоках нужно создать такую сеть, которая смогла бы их обслуживать за минимальную стоимость и при соблюдении некоторых требований к характеристикам. Обычно общая постановка задачи состоит в следующем.

Пусть задано:

1) Географическое местоположение множества устройств, которые нужно соединить друг с другом. Для простоты эти устройства будем называть *терминалами*.

2) Матрица трафика, указывающая интенсивности входных трафиков от каждого терминала к каждому другому терминалу. Требуется построить (рис. 5.37):

1. *Топологию подсети связи* для обслуживания трафика, возникающего у терминалов. Это включает выбор местоположения узлов, линий и пропускной способности каждой линии.

2. *Локальную сеть доступа*, т. е. набор линий связи, соединяющих терминалы с входными узлами подсети.

При построении сети следует стремиться к тому, чтобы

1) средняя задержка пакета или сообщения была меньше некоторого заданного уровня (при заданном номинальном трафике и выбранном типе алгоритма маршрутизации);

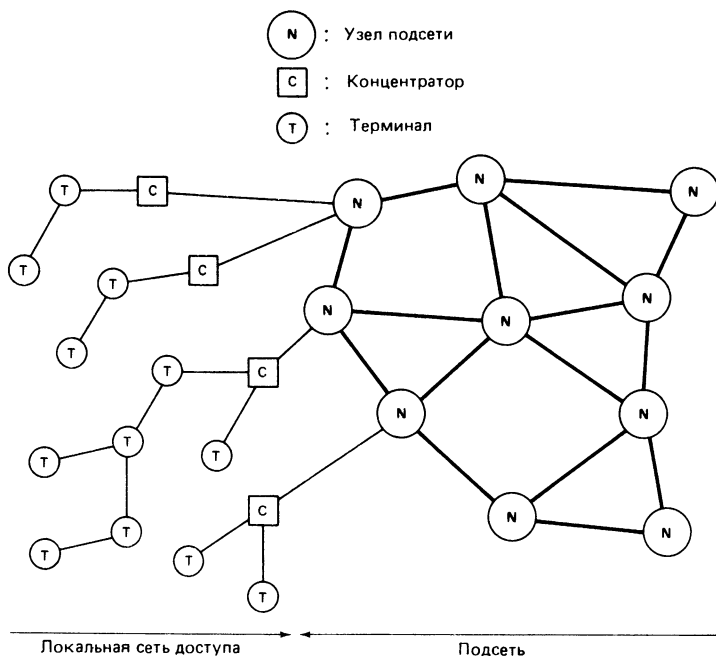


Рис. 5.37. Основными вопросами задачи построения топологии являются построение локальной сети доступа и подсети связи.

2) сеть удовлетворяла некоторым требованиям по надежности, гарантируя целостность сетевого обслуживания несмотря на ряд выходов из строя линий и узлов.

3) Минимизировать некоторую функцию от капитальных и эксплуатационных затрат при выполнении пунктов 1 и 2.

Очевидно, что, вообще говоря, такая постановка задачи является слишком широкой, ее трудно сформулировать точно, не говоря уже о ее решении. Однако во многих случаях ситуация несколько упрощается. Например, возможно, что часть этой задачи уже решена, так как некоторые участки локальной сети доступа и (или) подсети связи уже созданы. Кроме того, может вполне естественным выглядеть расчленение этой задачи на отдельные части, например, возможно, что задачу создания подсети связи естественно решать отдельно от задачи создания локальной сети доступа. В действительности это будет предполагаться ниже. Наконец, редко приходится иметь дело с четко поставленными задачами, которые можно решить точно и в разумные сроки. Обычно приходится довольствоваться приближенным решением,

полученным эвристическим методом, сочетающим теорию, метод проб и ошибок и здравый смысл.

В следующем подразделе рассматривается задача создания подсети связи в предположении, что локальная сеть доступа уже создана и, следовательно, матрица входного трафика, задающая поток для каждой пары узлов подсети, известна. Далее в подразд. 5.4.3 рассматривается задача создания локальной сети доступа.

5.4.2. Задача создания подсети связи

Пусть известны местоположения узлов подсети и интенсивности входных потоков для каждой пары этих узлов; требуется выбрать пропускные способности и потоки для каждой линии так, чтобы с минимальными затратами были удовлетворены некоторые требования по задержке и надежности. За исключением самых простых случаев, это оказывается сложной комбинаторной задачей. Чтобы показать, какие могут возникнуть трудности, рассмотрим упрощенный вариант этой задачи, в котором требуется выбрать пропускные способности линий так, чтобы минимизировать линейную стоимость. Пусть имеется ограничение по задержке, но никаких требований по надежности нет. Заметим, что, приписывая линии нулевую пропускную способность, мы тем самым устраним эту линию. Таким образом, задача выбора пропускных способностей включает как частный случай задачу выбора пар узлов подсети, которые следует непосредственно соединить друг с другом линией связи.

Задача выбора пропускных способностей

Задача состоит в том, чтобы для каждой линии (i, j) выбрать пропускную способность C_{ij} так, чтобы линейная стоимость

$$\sum_{(i, j)} p_{ij} C_{ij} \quad (5.34)$$

была минимальной (где p_{ij} — известная положительная цена единицы пропускной способности) при условии, что средняя задержка пакета не должна превышать фиксированное значение T .

Поток по каждой линии (i, j) , обозначаемый через F_{ij} , выражается в тех же единицах, что и пропускная способность. Будем исходить из модели $M/M/1$, основанной на клейнроковской аппроксимации независимостью, в результате чего ограничение на среднюю задержку можно представить в виде

$$\frac{1}{\gamma} \sum_{(i, j)} \frac{F_{ij}}{C_{ij} - F_{ij}} \leq T, \quad (5.35)$$

где γ — интенсивность суммарного потока, поступающего в сеть. Мы предполагаем, что интенсивности входных потоков для каждой пары отправитель—адресат известны и γ является их суммой. Потоки по линиям F_{ij} зависят от известных входных потоков и от выбранной схемы маршрутизации. Сначала будем предполагать, что маршрутизация и, следовательно, потоки F_{ij} известны, а затем посмотрим, что произойдет, если это предположение ослабить.

Когда потоки F_{ij} известны, рассматриваемая задача сводится к минимизации линейной стоимости (5.34) по пропускным способностям C_{ij} , удовлетворяющим ограничению (5.35); интуитивно ясно, что в точке оптимума это ограничение превратится в равенство. Введем множитель Лагранжа β и составим функцию Лагранжа

$$L = \sum_{(i, j)} \left(p_{ij} C_{ij} + \frac{\beta}{\gamma} \frac{F_{ij}}{C_{ij} - F_{ij}} \right).$$

В соответствии с методом множителей Лагранжа нужно приравнять нулю частные производные $\partial L / \partial C_{ij}$

$$\frac{\partial L}{\partial C_{ij}} = p_{ij} - \frac{\beta F_{ij}}{\gamma (C_{ij} - F_{ij})^2} = 0.$$

Из этого уравнения находим C_{ij}

$$C_{ij} = F_{ij} + \sqrt{\frac{\beta F_{ij}}{\gamma p_{ij}}}; \quad (5.36)$$

подставив это значение в равенство-ограничение, получим

$$T = \frac{1}{\gamma} \sum_{(i, j)} \frac{F_{ij}}{C_{ij} - F_{ij}} = \sum_{(i, j)} \sqrt{\frac{p_{ij} F_{ij}}{\beta \gamma}}.$$

Отсюда находим

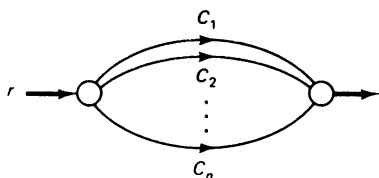
$$\sqrt{\beta} = \frac{1}{T} \sum_{(i, j)} \sqrt{\frac{p_{ij} F_{ij}}{\gamma}}, \quad (5.37)$$

после подстановки этого выражения в равенство (5.36) получаем оптимальное решение ¹⁾

$$C_{ij} = F_{ij} + \frac{1}{T} \sqrt{\frac{F_{ij}}{\gamma p_{ij}}} \sum_{(m, n)} \sqrt{\frac{p_{mn} F_{mn}}{\gamma}}.$$

¹⁾ Строго говоря, в этом месте надо доказать, что пропускные способности C_{ij} из (5.38) являются оптимальными. Читатель может сам проверить это, показав сначала, что выражение для средней задержки (5.35) является выпуклым и дифференцируемым по всем C_{ij} в области $C_{ij} > F_{ij}$, и затем доказав, что равенство $\partial L / \partial C_{ij} = 0$ представляет собой достаточное условие оптимальности и удовлетворяется при положительном множителе Лагранжа β вида (5.37) и при пропускных способностях C_{ij} вида (5.38) (см. [158], с. 84 или [201], с. 283).

Рис. 5.38. Минимизация суммарной пропускной способности $C_1 + \dots + C_n$ при ограничении на среднюю задержку приводит в результате к сети с минимальной связностью (все линии, кроме одной, удалены).



Это решение после преобразований можно переписать в виде

$$C_{ij} = F_{ij} \left(1 + \frac{1}{\gamma T} \frac{\sum_{(m,n)} \sqrt{p_{mn} F_{mn}}}{\sqrt{p_{ij} F_{ij}}} \right). \quad (5.38)$$

И наконец, после подстановки (5.38) в (5.34), получаем

$$\text{Оптимальная стоимость} = \sum_{(i,j)} p_{ij} F_{ij} + \frac{1}{\gamma T} \left(\sum_{(i,j)} \sqrt{p_{ij} F_{ij}} \right)^2. \quad (5.39)$$

Теперь рассмотрим задачу оптимизации стоимости сети одновременно и по пропускным способностям C_{ij} , и по потокам F_{ij} (или, что одно и то же, по способу маршрутизации). Эту задачу можно решить, минимизируя сначала стоимость (5.39) по F_{ij} , а затем находя оптимальные пропускные способности с помощью выражения (5.38). К сожалению, оказывается, что стоимость (5.39) имеет много локальных минимумов и найти ее глобальный минимум очень трудно. Кроме того, характерной особенностью этих локальных минимумов является то, что в них многие потоки F_{ij} и соответствующие пропускные способности C_{ij} равны нулю. Получающиеся в результате сети имеют малую связность (число линий мало, а их пропускные способности велики), что приводит к нарушению требований по надежности. Суть такого явления можно понять, рассмотрев простую сеть на рис. 5.38, содержащую два узла и n линий. Предположим, что необходимо выбрать пропускные способности C_1, \dots, C_n так, чтобы сумма $C_1 + \dots + C_n$ была минимальна и удовлетворялось ограничение на среднюю задержку, после того как входной поток λ будет поделен среди n линий. Из сравнения вероятностного и временного уплотнений (разд. 3.1 и 3.3) мы знаем, что задержка увеличивается при делении линии на несколько меньших линий, каждая из которых обслуживает долю всего трафика. Поэтому очевидно, что оптимальным решением будет топология с минимальной связностью, в которой все линии будут устранены, за исключением одной, имеющей пропускную способность, достаточную для того, чтобы было удовлетворено ограничение на среднюю задержку. Не имеет значения, какие из линий следует устранить,

и поэтому эта задача имеет n локальных минимумов. Каждый из них оказывается также глобальным минимумом, но это совпадение является случайным и обусловлено чрезвычайной простотой этого примера.

Из предыдущего обсуждения можно сделать вывод, что одномерная оптимизация по потокам на линиях и пропускным способностям является сложной комбинаторной задачей. Более того, даже если эта задача будет решена, полученная в результате топология сети будет иметь тенденцию концентрировать пропускную способность на малом числе больших линий, что, возможно, приведет к нарушению ограничений по надежности. Предыдущий анализ также является малоприменимым из-за того, что на практике стоимости пропускных способностей не являются линейными. Кроме того, значения пропускных способностей некоммутируемых линий обычно выбираются из дискретного, конечного набора значений. К тому же пропускная способность линии обычно одинакова для обоих направлений, а это не всегда имеет место, если их выбирать согласно (5.38). Эти практические ограничения еще больше усугубляют комбинаторный характер задачи и приводят к тому, что точное решение получить почти невозможно. В результате после учета всех этих ограничений единственно, что остается делать с этой задачей, так это использовать эвристические методы, к описанию которых мы переходим.

Эвристические методы выбора пропускных способностей

Обычно при использовании этих методов начинают с какой-то конкретной топологии сети, а затем последовательно подправляют эту топологию путем изменения пропускной способности какой-либо одной или одновременно нескольких линий. Таким образом, вблизи существующей топологии отыскивается другая топология, которая удовлетворяет заданным ограничениям и имеет меньшую стоимость. Заметим, что термин «топология» означает в данном случае перечень пропускных способностей всех возможных линий сети. В частности, если пропускная способность линии равна нулю, это указывает на то, что фактически линии нет. Обычно предполагается следующее:

1. Известны узлы сети и интенсивности входных потоков для каждой пары узлов.

2. Выбрана модель маршрутизации, позволяющая находить потоки F_{ij} для всех линий (i, j) при заданных пропускных способностях C_{ij} . Чаще всего потоки по линиям определяются исходя из модели оптимальной маршрутизации вида (5.33). Например, F_{ij} можно определить, минимизируя среднюю задержку пакета

$$D = \frac{1}{\gamma} \sum_{(i, j)} \left(\frac{F_{ij}}{C_{ij} - F_{ij}} + d_{ij} F_{ij} \right), \quad (5.40)$$

вычисляемую на основе аппроксимационной модели $M/M/1$ (см. (5.30)), где γ — интенсивность суммарного входного потока, а C_{ij} и d_{ij} — соответственно пропускная способность и задержка из-за обработки и распространения сигнала на линии (i, j) . В разд. 5.6 и 5.7 будут описаны несколько алгоритмов, которые можно использовать для этой цели.

3. Есть ограничение на максимальную задержку. Обычно требуется, чтобы выбранные пропускные способности C_{ij} и потоки F_{ij} (определяемые алгоритмом маршрутизации, согласно п. 2) были такими, чтобы задержка, вычисляемая по формуле (5.40) (или по какой-либо другой аналогичной формуле), была меньше некоторого, заранее выбранного порога.

4. Существует требование по надежности. Например, часто требуется, чтобы сеть была 2-связной, т. е. чтобы после выхода из строя одного любого узла все остальные узлы оставались связными. И вообще может потребоваться, чтобы после выхода из строя $(k - 1)$ узла все остальные узлы оставались связными — такую сеть называют k -связной. Далее более подробно будут рассмотрены вопросы, связанные с оценкой надежности сети.

5. Существует критерий стоимости, в соответствии с которым сравниваются различные топологии.

Цель состоит в поиске топологии, которая удовлетворяла бы требованиям по задержке и надежности в соответствии с пп. 3 и 4 и имела бы как можно меньшую стоимость в соответствии с п. 5. Мы опишем итеративный эвристический метод решения этой задачи. К началу очередной итерации в распоряжении имеются *лучшая текущая топология* и *проверяемая топология*. Первая топология удовлетворяет требованиям по задержке и надежности и является наилучшей среди всех рассмотренных до сих пор топологий, а вторая топология — это та, которая будет рассматриваться в данной итерации. Предполагается, что в самом начале эти топологии выбирались некоторым специальным образом, например число линий в сети бралось достаточно большим для того, чтобы требования по задержке и надежности заведомо удовлетворялись. Шаги итерации состоят в следующем:

Шаг 1 (вычисление потоков). Вычислить потоки F_{ij} для проверяемой топологии, используя для этого некоторый алгоритм маршрутизации в соответствии с п. 2.

Шаг 2 (проверка ограничения по задержке). Вычислить среднюю задержку пакета D по имеющейся формуле (например, по формуле (5.40)) для проверяемой топологии. Если

$$D \leq T,$$

где T — заданный порог, то перейти к шагу 3, в противном случае перейти к шагу 5.

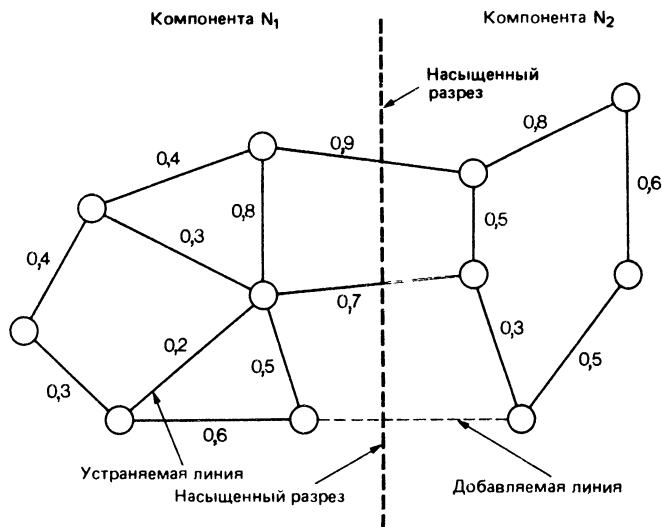


Рис. 5.39. Иллюстрация метода насыщенного разреза, который можно использовать при эвристической замене ветвей. Число рядом с каждой дугой означает степень нагруженности данной линии. Одна за другой наиболее нагруженные линии временно удаляются до тех пор, пока сеть не распадется на две части N_1 и N_2 . Затем добавляется линия, соединяющая какой-либо узел из N_1 с каким-либо узлом из N_2 , а какая-либо слабо нагруженная линия удаляется.

Шаг 3 (проверка требования по надежности). Проверить, удовлетворяет ли проверяемая топология требованию по надежности. (Методы выполнения этого шага будут вскоре обсуждаться.) Если это требование не удовлетворяется, то перейти к шагу 5, в противном случае перейти к шагу 4.

Шаг 4 (проверка снижения стоимости). Если стоимость проверяемой топологии меньше стоимости лучшей текущей топологии, то заменить лучшую текущую топологию на проверяемую.

Шаг 5 (создание новой проверяемой топологии). Используя какой-либо эвристический алгоритм, изменить одну или несколько пропускных способностей лучшей текущей топологии так, чтобы в результате получилась проверяемая топология, которая ранее не рассматривалась. Перейти к шагу 1.

Заметим, что проверяемая топология выбирается в качестве лучшей текущей на шаге 4 только тогда, когда она удовлетворяет требованиям по задержке и надежности на шагах 2 и 3 и когда она уменьшает стоимость (шаг 4). Алгоритм останавливается, когда не удастся создать новую проверяемую топологию или когда дальнейшего существенного улучшения ожидать не приходится. Естественно, что нет никакой гарантии того, что оконча-

тельное решение окажется оптимальным. Можно попытаться добиться лучшего результата, повторив алгоритм с иной начальной топологией. Другая возможность состоит в модификации шага 4 таким образом, чтобы иногда проверяемая топология случайно выбиралась в качестве лучшей текущей даже тогда, когда ее стоимость оказывается больше, чем найденная до сих пор. Схемы такого типа, известные как *алгоритмы отпускания*, стали популярны в последнее время [139]. Можно показать, что при достаточно широких предположениях глобальное оптимальное решение можно достичь, используя для этого алгоритм отпускания [109, 234]. Однако время вычислений в этом случае может оказаться очень большим. Такой подход еще не применялся в контексте создания топологии сети.

Существует ряд эвристических правил, предложенных в литературе, для создания новых проверяемых топологий (шаг 5). Одним из таких правил является просто уменьшение пропускной способности той линии, которая кажется недогруженной (малые значения F_{ij}/C_{ij} и F_{ji}/C_{ji}), или вообще устранение этой линии. Другим правилом является увеличение пропускной способности какой-либо перегруженной линии в случае, когда не выполняется ограничение на задержку. Комбинацией этих правил является *эвристический метод замены ветвей*, при котором одна линия удаляется и вместо нее добавляется другая линия. Полезным при выборе линий, которые следует удалять и добавлять, является так называемый *метод насыщенного разреза*. Идея метода состоит в отыскании такого разбиения (или разреза) узлов на два подмножества N_1 и N_2 , для которого линии, соединяющие N_1 и N_2 , более всего нагружены. Кажется естественным, что добавление какой-либо линии, соединяющей узел из N_1 с узлом из N_2 , поможет понизить уровень нагруженности этого разреза. Метод работает следующим образом (рис. 5.39):

1. Подготовить список всех ненаправленных линий (i, j), рассортированных в порядке уменьшения степени нагруженности, численно выражаемой как $\max \{F_{ij}/C_{ij}, F_{ji}/C_{ji}\}$.

2. Найти линию k , такую, что а) если удалить все линии, находящиеся в списке выше k , то сеть останется связной и б) если удалить k вместе со всеми линиями, находящимися выше ее в списке, то сеть распадается на две несвязные части N_1 и N_2 .

3. Удалить из сети наименее нагруженную линию и вместо нее добавить новую линию, соединяющую узел из N_1 с узлом из N_2 .

Существует несколько вариантов рассмотренной выше схемы. Например, при выборе удаляемой линии можно учитывать стоимость пропускной способности линии. Нет необходимости убеждать в том, что априорные сведения о сети и стоимости пропускной способности линии можно эффективно использовать в этих

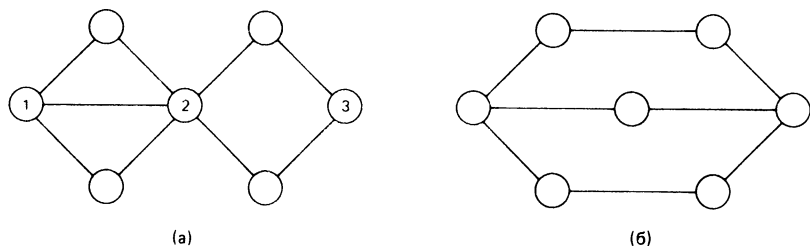


Рис. 5.40. Иллюстрация определения k -связности. В графе (а) узлы 1 и 2 являются 6-связными, узлы 2 и 3 являются 2-связными, а узлы 1 и 3 являются 1-связными. Граф (а) является 1-связным. Граф (б) является 2-связным.

эвристических алгоритмах. Кроме того, можно сочетать сразу несколько алгоритмов для более эффективного решения поставленной задачи. Теперь обратим внимание на методы оценки надежности сети.

Вопросы надежности сети

Мы уже упоминали, что самым распространенным требованием по надежности сети является то, чтобы она была k -связной. Определим теперь k -связность более формально.

Будем называть два узла i и j в неориентированном графе k -связными, если имеется путь, соединяющий i и j в каждом подграфе, полученном удалением из графа $(k - 1)$ узлов, отличных от i и j , вместе с их смежными дугами. Легко заметить, что это эквивалентно требованию, чтобы либо i и j были соединены дугой, либо существовало не менее k соединяющих i и j непересекающихся в узлах путей, т. е. таких, чтобы любые два из этих путей не имели общих узлов, кроме i и j . Будем называть граф k -связным, если каждая пара узлов в графе является k -связной. Эти определения проиллюстрированы на рис. 5.40. Представляет интерес понятие *дуговой связности*, определяемое как минимальное число дуг, которые следует удалить для того, чтобы граф оказался несвязным. Нижняя граница k для дуговой связности может быть установлена путем проверки k -(узловой) связности для расширенного графа, в котором каждая дуга (i, j) заменена на две дуги (i, n) и (n, j) , где n — новый узел. Это становится очевидным, если заметить, что выход из строя узла n в расширенном графе соответствует выходу из строя дуги (i, j) в первоначальном графе.

Можно вычислить число непересекающихся в узлах путей, соединяющих два узла, путем решения соответствующей задачи о максимальном потоке. Эта классическая комбинаторная задача

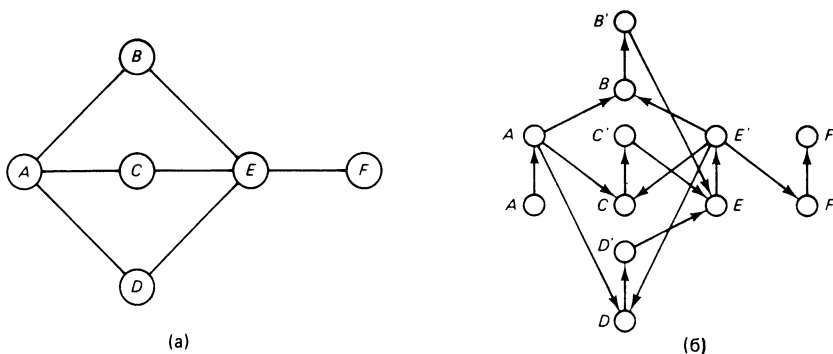


Рис. 5.41. Иллюстрация теста k -связности для тех, кто знаком с задачей о максимальном потоке. Для того чтобы найти число путей, не имеющих общих узлов, от узла A до узла F в графе (а) нужно проделать следующее: (1) Заменить каждый узел i двумя узлами i и i' и ориентированной дугой (i, i') . (2) Заменить каждую дугу (i, j) графа (а) двумя ориентированными дугами (i', j) и (j', i) . (3) Удалить все дуги, приходящие к A и уходящие от F' , в результате чего получится граф (б). (4) Приписать бесконечную пропускную способность дугам (A, A') и (F, F') и единичную пропускную способность всем остальным дугам. Тогда максимальный поток, который можно направить от узла A к узлу F в графе (б), равен числу путей, не имеющих общих узлов, соединяющих узлы A и F в графе (а).

подробно исследована во многих работах [159, 188]. Здесь она не будет обсуждаться, так как эта тема несколько выходит за рамки данного раздела. Читатель, который уже знаком с этой задачей, может увидеть на рис. 5.41 ее связь с задачей k -связности двух узлов.

Одним из способов проверки k -связности графа является проверка k -связности каждой пары узлов. Существуют, однако, более эффективные методы. В частности, метод Клейтмана [142] состоит в следующем.

Выбирается произвольный узел n_0 и проверяется k -связность этого узла с каждым другим узлом. Затем из графа удаляется узел n_0 вместе со всеми смежными дугами, выбирается какой-либо другой узел n_1 и проверяется $(k-1)$ -связность этого узла с каждым из остальных узлов. Так продолжается либо до того момента, как будет проверено, что узел n_{k-1} 1-связан с каждым из оставшихся узлов, либо окажется, что некоторый узел n_i , $i = 0, 1, \dots, k-1$, не является $(k-i)$ -связным с одним из оставшихся узлов. В последнем случае очевидно, что граф не является k -связным, так как если узел n_i не является $(k-i)$ -связным с некоторым другим из оставшихся узлов, то, следовательно, можно найти $(k-i-1)$ узлов, таких, что их удаление вместе с удалением узлов n_0, \dots, n_{i-1} делает граф несвязным. Этот процесс показан на рис. 5.42.

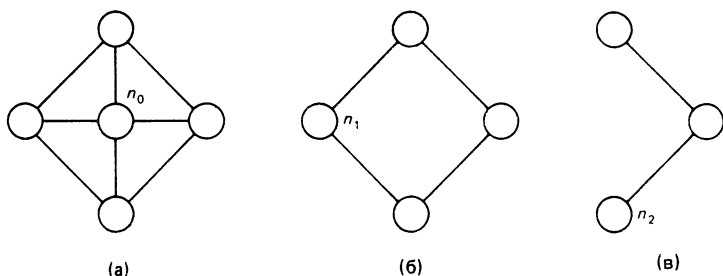


Рис. 5.42. Проверка 3-связности графа (а) алгоритмом Клейтмана. Выбирается произвольный узел n_0 и проверяется 3-связность между n_0 и каждым другим узлом. Затем удаляется узел n_0 и смежные с ним дуги, в результате чего получается граф (б). В (б) выбирается произвольный узел n_1 и проверяется 2-связность между n_1 и каждым другим узлом в (б). Затем удаляется узел n_1 и смежные с ним дуги, в результате чего получается граф (в). Так как граф (в) связан, то отсюда следует, что первоначальный граф (а) является 3-связным.

Для доказательства того, что алгоритм Клейтмана приводит к правильному результату, необходимо показать, что если удалось найти узлы n_0, \dots, n_{i-1} , как описано выше, то граф обязательно будет k -связным. Докажем это от противного. Если граф не является k -связным, то должны существовать узлы i и j и такое множество $(k-1)$ узлов \mathcal{F}_0 ($i \notin \mathcal{F}_0, j \notin \mathcal{F}_0$), что после удаления \mathcal{F}_0 не останется ни одного пути, соединяющего i и j . Тогда n_0 обязательно принадлежит \mathcal{F}_0 потому, что если бы $n_0 \notin \mathcal{F}_0$, то после удаления множества \mathcal{F}_0 все же остались бы пути P_i и P_j , соединяющие n_0, i и j , так как n_0 является k -связным с каждым другим узлом. Следовательно, путь от i до j можно было бы построить, используя сначала путь P_i от i до n_0 , а затем путь P_j от n_0 до j . Далее рассмотрим множества $\mathcal{F}_1 = \mathcal{F}_0 - \{n_0\}$, $\mathcal{F}_2 = \mathcal{F}_1 - \{n_1\}$, \dots . Аналогично n_1 должен принадлежать \mathcal{F}_1 , n_2 должен принадлежать \mathcal{F}_2 и т. д. Если это рассуждение использовать $(k-1)$ раз, то увидим, что $\mathcal{F}_0 = \{n_0, n_1, \dots, n_{k-2}\}$. Поэтому после удаления \mathcal{F}_0 из графа узел n_{k-1} останется нетронутым и, следовательно, в соответствии с определением n_{k-1} будут существовать пути от n_{k-1} до i и j . Эти пути можно использовать для соединения i и j , что противоречит предположению о том, что после удаления \mathcal{F}_0 узлы i и j станут несвязными.

Суммарное число проверок пар узлов на связность в алгоритме Клейтмана равно $\sum_{i=1}^k (N-i) = kN - k(k+1)/2$, где N — число узлов. Существует другой алгоритм, разработанный Ивеном, требующий примерно N проверок на связность. Более подробно читатель может ознакомиться с этим алгоритмом в работе [64].

Предыдущая постановка задачи о надежности ориентирована на наихудший случай, когда выходит из строя самая нежелательная комбинация из k узлов. При другой постановке задачи всем элементам сети (узлам и линиям) приписываются вероятности выхода из строя и оценивается вероятность того, что данная пара узлов окажется несвязной. Для того чтобы представить, какие трудности могут возникнуть при таком подходе, предположим, что имеется всего n ненадежных элементов в сети. Тогда число возможных различных комбинаций из не вышедших из строя элементов равно 2^n . Пусть s_k означает k -ю комбинацию, а p_k — вероятность ее возникновения. Тогда вероятность того, что сеть останется связной, равна

$$P_C = \sum_{s_k \in C} p_k, \quad (5.41)$$

где C — множество комбинаций из не вышедших из строя элементов, для которых сеть останется связной. Таким образом, при вычислении P_C необходимо явно или неявно пронумеровать элементы множества C . Как правило, на такую процедуру при вычислении на ЭВМ уходит очень много времени. Тем не менее, используя некоторые искусственные приемы (например, аппроксимацию или отбрасывание несущественных членов), иногда удается вычислить вероятность P_C для некоторых реальных сетей [13, 17]. При другом подходе оценивается нижняя граница для P_C путем упорядочения всех комбинаций из не вышедших из строя элементов по степени их правдоподобия и затем суммирования p_k по подмножеству наиболее вероятных комбинаций из C [156, 163]. Аналогичный подход, примененный к дополнению C , дает верхнюю границу для P_C .

Построение топологии остовного дерева

Для некоторых сетей, для которых вопросы надежности не столь существенны, может вполне подойти топология остовного дерева. Использование такой топологии согласуется с идеей сосредоточения всей пропускной способности лишь на небольшом числе линий для того, чтобы уменьшить среднюю задержку (см. предыдущее обсуждение задачи выбора пропускной способности).

Можно построить топологию остовного дерева, приписав веса w_{ij} каждой потенциально возможной линии (i, j) и используя алгоритм построения остовного дерева минимального веса (МОД) из подразд. 5.2.2. Некоторые трудности могут появиться, если существует ограничение на максимальный трафик, допустимый для передачи по одной линии. В этом случае возникает *задача построения МОД с ограничением*, в которой имеется матрица входного трафика от каждого узла к каждому другому узлу и требуется построить МОД, в котором поток по каждой линии не

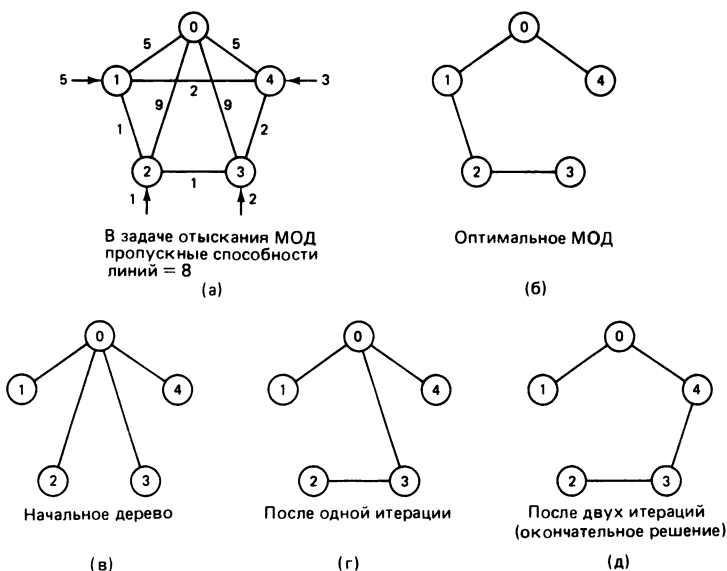


Рис. 5.43. Иллюстрация эвристического алгоритма Эссу—Вильямса для построения МОД с ограничением. Условие задачи представлено в (а). Узел 0 является центральным узлом. Веса линий указаны рядом с линиями. Входные потоки от узлов 1, 2, 3 и 4 к центральному узлу указаны рядом со стрелками. Ограничение состоит в том, чтобы поток по каждой линии был не более 8. Алгоритм заканчивается после двух итераций и в результате получается дерево (д), имеющее суммарный вес 13. Конец наступает вследствие того, что, когда линия (1, 0) или (4, 0) удаляется и добавляется какая-либо линия, не смежная с узлом 0, нарушается ограничение на пропускную способность какой-либо линии (поток по линии ≤ 8). Оптимальное дерево, показанное в (б), имеет суммарный вес 12.

превышает некоторый, заранее установленный предел. Вследствие этого ограничения задача приобретает комбинаторный характер и для ее решения обычно применяются эвристические методы.

Одним из таких методов является модификация алгоритмов Крускала или Прим — Дijkstra, описанных в подразд. 5.2.2, при которой на каждой итерации при добавлении новой линии к имеющимся фрагментам делается проверка выполнения ограничения для потоков, проходящих по линиям имеющихся фрагментов. Если хотя бы для одного потока это ограничение не выполняется, то данная линия не добавляется и рассматривается другая линия. Этот эвристический метод можно применять в комбинации с методом эвристического обмена ветвей, который обсуждался ранее в связи с общей задачей построения подсети.

Для решения частного случая задачи построения МОД с ограничением может быть применен алгоритм *Эссау—Вильямса* [63]. В этом случае имеется центральный узел 0 и N других узлов. Весь трафик должен проходить через центральный узел и поэтому можно считать, что весь входной трафик поступает от нецентральных узлов к центральному и наоборот. Эта задача возникает также в контексте задачи построения локальной сети доступа; в этом случае роль центрального узла играет концентратор трафика.

Алгоритм Эссау—Вильямса является, по существу, методом эвристического обмена ветвей. Он начинается с остоного дерева, в котором центральный узел непосредственно связан с каждым из N других узлов (предполагается, что такое решение является допустимым). При каждой очередной итерации из имеющегося остоного дерева удаляется линия $(i, 0)$, соединяющая некоторый узел i с центральным узлом 0, а вместо нее добавляется некоторая линия (i, j) . Эти линии выбираются таким образом, чтобы

- 1) не образовался цикл;
- 2) было удовлетворено ограничение на пропускные способности для всех линий нового остоного дерева;
- 3) экономия $w_{i0} - w_{ij}$ в весе линии, получающаяся при замене $(i, 0)$ на (i, j) , была бы положительной и максимальной среди всех узлов i и j , удовлетворяющих требованиям 1 и 2.

Алгоритм останавливается, когда не удается найти такие узлы i и j , которые бы удовлетворяли требованиям 1—3 при замене $(i, 0)$ на (i, j) . Работа этого алгоритма проиллюстрирована на рис. 5.43.

5.4.3. Задача построения локальной сети доступа

Здесь предполагается, что подсеть связи уже имеется и требуется построить сеть, которая бы подсоединила к подсети ряд терминалов с известными входными потоками. Эта задача часто возникает при создании сети с иерархической архитектурой, при которой группа терминалов подсоединяется, возможно, посредством локальных сетей, к различным концентраторам, которые в свою очередь подсоединяются к концентраторам более высокого уровня и т. д. Трудно рекомендовать какую-либо общую стратегию построения такой сети без знания конкретной практической ситуации. Однако существует несколько часто возникающих подзадач. Мы обсудим одну из таких подзадач, известную под названием *задача выбора местоположения концентратора*.

В этой задаче имеется n источников, расположенных в известных географических точках. Источником может быть, например, шлюз локальной сети, которая соединяет несколько терминалов в пределах области, где расположены средства пользователей или

Даже с учетом того, что переменные x_{ij} могут принимать только значения 0 или 1, рассматриваемая задача не является по-настоящему сложной комбинаторной задачей. Если целочисленное ограничение

$$x_{ij} = 0 \text{ или } 1$$

заменить на нецелочисленное ограничение

$$0 \leq x_{ij} \leq 1, \quad (5.45)$$

то задача превратится в транспортную задачу линейного программирования, для решения которой разработаны очень эффективные алгоритмы, такие как симплекс-метод. Можно показать, что решение, полученное симплекс-методом, будет целочисленным, т. е. x_{ij} будут равны 0 или 1 (см. [54] и [188]). Поэтому решение задачи без учета ограничения целочисленности автоматически приведет к оптимальному целочисленному решению. Кроме симплекс-метода существуют также другие методы решения этой задачи [27, 74, 202].

Теперь рассмотрим более сложную задачу, в которой местоположения концентраторов не фиксированы, а, наоборот, являются предметом оптимизации. Имеется m потенциальных мест для расположения концентраторов и определена цена b_j размещения концентратора в точке j . Тогда стоимость становится равной

$$\text{Стоимость} = \sum_{i=1}^n \sum_{j=1}^m a_{ij} x_{ij} + \sum_{j=1}^m b_j y_j, \quad (5.46)$$

где

$$y_j = \begin{cases} 1, & \text{если концентратор расположен в точке } j, \\ 0 & \text{в противном случае.} \end{cases}$$

Ограничение (5.44) теперь состоит в том, чтобы

$$\sum_{i=1}^n x_{ij} \leq K_j y_j \quad \text{для всех концентраторов } j. \quad (5.47)$$

Таким образом, задача состоит в минимизации стоимости (5.46) при ограничениях (5.43) и (5.47) и требовании, чтобы x_{ij} и y_j были равны либо нулю, либо единице.

Эта задача рассматривалась в литературе по исследованию операций, где она известна под названием *задачи о размещении складов*. Это сложная комбинаторная задача, которую можно решить обычно лишь приблизительно. Предложен ряд точных и эвристических методов решения этой задачи. Более подробно с ними можно ознакомиться в литературе [4, 6, 45, 134, 140, 197].

5.5. Характерные особенности оптимальной маршрутизации

Мы теперь вернемся к задаче оптимальной маршрутизации, сформулированной в начале разд. 5.4. Главной целью этого раздела является описание интересных характерных особенностей оптимальной маршрутизации. Будет показано, что при оптимальной маршрутизации трафик направляется по тем путям, которые являются кратчайшими по отношению к некоторым «длинам» линий, зависящим от потоков, проходящих по этим линиям. Эта характерная особенность используется при разработке алгоритмов, которые будут рассмотрены в разделах 5.6 и 5.7.

Вспомним вид стоимостной функции

$$\sum_{(i,j)} D_{ij}(F_{ij}). \quad (5.48)$$

Здесь F_{ij} — суммарный поток (в единицах данных в секунду), проходящий по линии (i, j) :

$$F_{ij} = \sum_{\substack{\text{по всем путям } p, \\ \text{содержащим } (i,j)}} x_p, \quad (5.49)$$

где x_p — поток (в единицах данных в секунду), проходящий по пути p . Для каждой ОА-пары w существуют ограничения

$$\sum_{p \in P_w} x_p = r_w, \quad (5.50)$$

$$x_p \geq 0 \quad \text{для всех } p \in P_w, \quad (5.51)$$

где r_w — известная интенсивность входного трафика ОА-пары w (в единицах данных в секунду), а P_w — множество ориентированных путей для w . Задача, которая формулируется с использованием неизвестного вектора путевых потоков $x = \{x_p \mid p \in P_w, w \in W\}$, записывается в виде

$$\text{минимизировать } \sum_{(i,j)} D_{ij} \left[\sum_{\substack{\text{по всем путям } p, \\ \text{содержащим } (i,j)}} x_p \right],$$

$$\text{при ограничениях } \sum_{p \in P_w} x_p = r_w \quad \text{для всех } w \in W, \quad (5.52)$$

$$x_p \geq 0 \quad \text{для всех } p \in P_w, w \in W.$$

Далее мы будем описывать оптимальную маршрутизацию с использованием производных функций D_{ij} . Предполагается, что каждая D_{ij} является дважды дифференцируемой функцией от F_{ij} и определена в полуинтервале $[0, C_{ij})$, где значение C_{ij} равно либо некоторой положительной константе (в типичных случаях это пропускная способность линии), либо бесконечности. Предполагается, что первая и вторая производные D_{ij} , обозначаемые

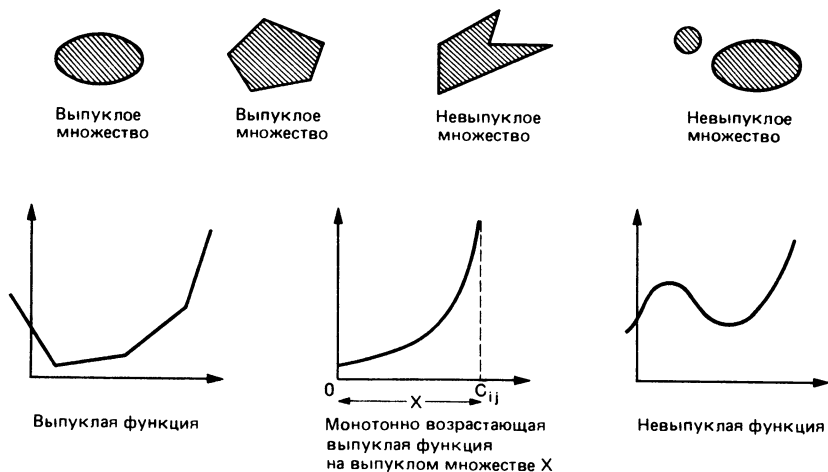


Рис. 5.45. Выпуклые и невыпуклые множества и функции.

соответственно через D'_{ij} и D''_{ij} , строго положительны для всех F_{ij} из $[0, C_{ij}]$. Отсюда, в частности, следует, что D_{ij} является выпуклой¹⁾, монотонно возрастающей функцией от F_{ij} на всем полуинтервале $[0, C_{ij}]$ (рис. 5.45). Кроме того предполагается, что $D_{ij}(F_{ij}) \rightarrow \infty$ при $F_{ij} \rightarrow C_{ij}$. Так как потоки в линиях являются линейными функциями путевых потоков, то можно непосредственно проверить, что задача (5.52), рассматриваемая как задача с переменными $\{x_p\}$, относящимися к путевым потокам, имеет выпуклую дважды дифференцируемую стоимостную функцию и выпуклое ограничительное множество.

Описание решений задачи оптимальной маршрутизации будет получено из следующего необходимого и достаточного условия оптимальности.

Лемма. Пусть f — некоторая дифференцируемая выпуклая функция n -мерного вектора $x = (x_1, \dots, x_n)$ и X — некоторое выпуклое множество векторов. Тогда $x^* \in X$ будет оптимальным решением задачи

$$\begin{aligned} &\text{минимизировать } f(x) \\ &\text{при ограничении } x \in X \end{aligned} \quad (5.53)$$

¹⁾ Говорят, что множество векторов X выпукло, если отрезок прямой, соединяющий любые два вектора из X , также содержится в X . Алгебраически это записывается в виде $[\alpha x_1 + (1 - \alpha) x_2] \in X$ для всех $x_1, x_2 \in X$ и $\alpha \in [0, 1]$. Функция f от вектора x выпукла на выпуклом множестве X , если $f[\alpha x_1 + (1 - \alpha) x_2] \leq \alpha f(x_1) + (1 - \alpha) f(x_2)$ для всех $x_1, x_2 \in X$ и $\alpha \in [0, 1]$. Эти определения продемонстрированы на рис. 5.45.

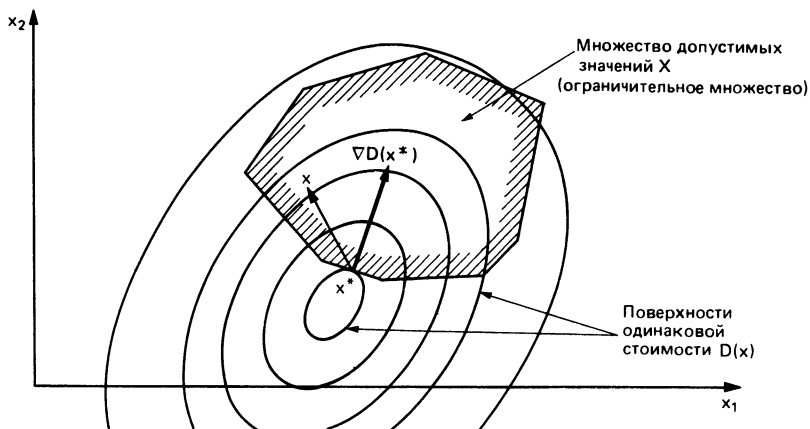


Рис. 5.46. Геометрическая интерпретация условия оптимальности (5.54). Вектор градиента, состоящий из первых частных производных

$$\nabla D(x^*) = \begin{bmatrix} \frac{\partial D(x^*)}{\partial x_1} \\ \frac{\partial D(x^*)}{\partial x_n} \end{bmatrix}$$

в точке оптимума x^* составляет угол 90° или меньше со всеми векторами $x - x^*$, где x принимает все значения из множества допустимых значений X .

тогда и только тогда, когда

$$\sum_{i=1}^n \frac{\partial f(x^*)}{\partial x_i} (x_i - x_i^*) \geq 0 \quad \text{для всех } x \in X, \quad (5.54)$$

где $\partial f(x^*)/\partial x_i$ — значение первой производной f по i -й координате x_i в точке x^* .

Доказательство. Предположим, что x^* — оптимальное решение и для каждого $x \in X$ рассмотрим функцию $g(\alpha) = f[x^* + \alpha(x - x^*)]$ скалярной переменной α . Тогда $g(\alpha)$ достигает минимума на отрезке $[0, 1]$ в точке $\alpha = 0$ и поэтому $dg(0)/d\alpha \geq 0$. Используя правило дифференцирования сложной функции, имеем

$$\frac{dg(0)}{d\alpha} = \sum_{i=1}^n \frac{\partial f(x^*)}{\partial x_i} (x_i - x_i^*),$$

и, следовательно, неравенство (5.54) доказано.

Для доказательства достаточности предположим, что неравенство (5.54) справедливо, но x^* не является оптимальным решением. Тогда мы должны прийти к противоречию. Действительно,

пусть $\bar{x} \in X$, такое, что $f(\bar{x}) < f(x^*)$, и рассмотрим функцию $g(\alpha) = f[x^* + \alpha(\bar{x} - x^*)]$. Тогда $dg(0)/d\alpha \geq 0$ (на основании (5.54)), а $f(x^*) = g(0) > g(1) = f(\bar{x})$. Можно показать (см. задачу 5.24), что эти условия противоречат выпуклости $g(\alpha)$ на $[0, 1]$ и, следовательно, выпуклости f . На этом доказательство заканчивается.

Геометрическая интерпретация этой леммы представлена на рис. 5.46. Так как и стоимостная функция и все ограничительные множества задачи минимизации (5.52) являются выпуклыми, то можно применить лемму. Пусть x будет вектором путевых потоков x_p . Для конкретности предположим, что пути p последовательно перенумерованы. Тогда если $W = \{w_1, w_2, \dots, w_m\}$ — множество ОА-пар, то соответствующими множествами путей являются

$$\begin{aligned} P_{w_1} &= \{1, 2, \dots, n_1\}, \\ P_{w_2} &= \{(n_1 + 1), \dots, n_2\}, \\ &\dots \\ P_{w_m} &= \{(n_{m-1} + 1), \dots, n_m\}, \end{aligned}$$

где n_1, n_2, \dots, n_m — некоторые целые числа, такие, что $n_1 < n_2 < \dots < n_m$. Координатами вектора путевых потоков x будут путевые потоки x_1, x_2, \dots, x_{n_m} , т. е.

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n_m} \end{bmatrix}.$$

Обозначим через $D(x)$ стоимостную функцию задачи (5.52)

$$D(x) = \sum_{(i,j)} D_{ij} \left[\sum_{\substack{\text{по всем путям } p, \\ \text{содержащим } (i,j)}} x_p \right],$$

а через $\partial D(x)/\partial x_p$ — частную производную D по x_p . Тогда

$$\frac{\partial D(x)}{\partial x_p} = \sum_{\substack{\text{по всем линиям } (i,j) \\ \text{на пути } p}} D'_{ij}, \quad (5.55)$$

где первые производные D'_{ij} берутся при суммарных потоках, соответствующих x . Из равенства (5.55) видно, что $\partial D/\partial x_p$ является длиной пути p , если длину каждой линии (i, j) положить равной первой производной D'_{ij} , взятой в x . Поэтому далее $\partial D/\partial x_p$ называется *первопроизводной длиной пути p* .

В соответствии с леммой $x^* = \{x_p^*\}$ является вектором оптимальных путевых потоков, если он удовлетворяет ограничениям задачи маршрутизации и условию (5.54). Это условие можно переписать в виде

$$\sum_{w \in W} \sum_{p \in P_w} \frac{\partial D(x^*)}{\partial x_p} (x_p - x_p^*) \geq 0 \quad (5.56)$$

для всех x_p , удовлетворяющих ограничениям

$$\sum_{p \in P_w} x_p = r_w, \quad x_p \geq 0 \quad \text{для всех } p \in P_w, \quad w \in W. \quad (5.57)$$

Условия (5.56) и (5.57) можно объединить по отношению к ОА-паре и для всех $w \in W$ записать в виде

$$\sum_{p \in P_w} \frac{\partial D(x^*)}{\partial x_p} (x_p - x_p^*) \geq 0 \quad \text{для всех } x_p \geq 0, \quad (5.58)$$

$$\text{и } p \in P_w, \text{ таких, что } \sum_{p \in P_w} x_p = r_w.$$

(Чтобы проверить это, нужно фиксировать некоторую ОА-пару w и положить $x_p = x_p^*$ в (5.56) для всех $w' \neq w$ и $p \in P_{w'}$.) Условие (5.58) эквивалентно требованию, чтобы

для всех $w \in W$ $x_p^* > 0$ только тогда, когда

$$\frac{\partial D(x^*)}{\partial x_{p'}} \geq \frac{\partial D(x^*)}{\partial x_p} \quad \text{для всех } p' \in P_w. \quad (5.59)$$

Чтобы это увидеть, рассмотрим произвольный путь p , для которого $x_p^* > 0$, и некоторый другой путь p' , и, взяв достаточно малое $\varepsilon > 0$, в неравенстве (5.58) положим $x_p = x_p^* - \varepsilon$, $x_{p'} = x_{p'}^* + \varepsilon$ и $x_{p''} = x_{p''}^*$ для всех остальных $p'' \neq p, p'$. Смысл условия (5.59) состоит в том, что набор путевых потоков оптимален тогда и только тогда, когда путевой поток положителен только для тех путей, которые имеют минимальную первопроизводную длину. Из условия (5.59) также следует, что в точке оптимума пути, по которым проходит ненулевая часть входного потока r_w ОА-пары w , должны иметь одинаковую длину (меньшую или равную длине, чем все другие пути для w).

Пример

Рассмотрим сеть, состоящую из двух линий (рис. 5.47), в которой узел 1 — это единственный отправитель, а узел 2 — единственный получатель. Известный входной поток r требуется разделить на два путевых потока x_1 и x_2 так, чтобы минимизировать стоимостную функцию, основанную на аппроксимации моделью $M/M/1$

$$D(x) = D_1(x_1) + D_2(x_2),$$

где $D_i(x_i) = x_i / (C_i - x_i)$ для $i = 1, 2$ и C_i — пропускная способность линии i . Для того чтобы эта задача имела смысл, мы должны предположить, что r меньше, чем максимальная скорость передачи $C_1 + C_2$, которую сеть может иметь.

В точке оптимума должно выполняться условие кратчайшего пути (5.59) так же, как и следующие ограничения:

$$x_1^* + x_2^* = r, \quad x_1^* \geq 0, \quad x_2^* \geq 0.$$

Предположим, что $C_1 \geq C_2$. Тогда из соображений здравого смысла оптимальный поток x_1 не может быть меньше чем x_2 (не имеет смысла посылать больше трафика по линии с меньшей пропускной способностью). Единственными возможными являются следующие.

1. $x_1^* = r$ и $x_2^* = 0$. В соответствии с (5.59) это может произойти только тогда, когда

$$\frac{dD_1(r)}{dx_1} \leq \frac{dD_2(0)}{dx_2}.$$

Отсюда (так как производная от $x/(C-x)$ равна $C/(C-x)^2$)

$$\frac{C_1}{(C_1-r)^2} \leq \frac{1}{C_2}$$

или

$$r \leq C_1 - \sqrt{C_1 C_2}. \quad (5.60)$$

2. $x_1^* > 0$ и $x_2^* > 0$. В этом случае из (5.59) следует, что длины путей 1 и 2 одинаковы, т. е.

$$\frac{dD_1(x_1^*)}{dx_1} = \frac{dD_2(x_2^*)}{dx_2}$$

или, что эквивалентно,

$$\frac{C_1}{(C_1 - x_1^*)^2} = \frac{C_2}{(C_2 - x_2^*)^2}. \quad (5.61)$$

Из этого равенства и ограничения $x_1^* + x_2^* = r$ можно найти значения x_1^* и x_2^* . Непосредственные вычисления показывают, что решением является

$$x_1^* = \frac{\sqrt{C_1} [r - (C_2 - \sqrt{C_1 C_2})]}{\sqrt{C_1} + \sqrt{C_2}},$$

$$x_2^* = \frac{\sqrt{C_2} [r - (C_1 - \sqrt{C_1 C_2})]}{\sqrt{C_1} + \sqrt{C_2}}.$$

Оптимальное решение показано на рис. 5.48 для r , принимающего значения в диапазоне $[0, C_1 + C_2]$ возможных значений. Можно заметить, что для достаточно малых r , когда имеет место равенство (5.60), используется только более высокоскоростная линия 1. Когда r превышает пороговое значение

$$C_1 - \sqrt{C_1 C_2},$$

при котором первопроизводные длины двух линий становятся равными и неравенство (5.60) переходит в равенство, линия 2 с меньшей пропускной способностью также используется. По мере возрастания r потоки на обеих линиях увеличиваются так, чтобы первопроизводные длины оставались равными в соответствии с (5.61). Такое поведение является типичным для оптимальной маршрутизации, если стоимостная функция основана на аппроксимации моделью $M/M/1$;

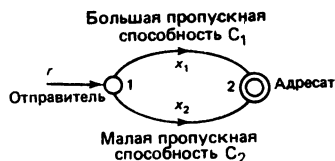


Рис. 5.47. Пример задачи маршрутизации, включающий одну ОА-пару и два пути.

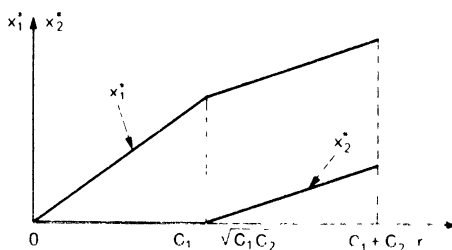


Рис. 5.48. Оптимальные путевые потоки для рассматриваемого примера маршрутизации. При небольшом входном трафике используется только одна линия с большей пропускной способностью. Когда входной трафик превышает порог C_1 --- $\sqrt{C_1 C_2}$, часть трафика направляется по линии с меньшей пропускной способностью.

при небольшом входном трафике каждая ОА-пара стремится использовать только один путь для маршрутизации (самый быстрый в терминах времени доставки пакета), а по мере увеличения входного трафика начинают использоваться дополнительные пути, для того чтобы избежать перегрузки самого быстрого пути. В более общем случае такое поведение проявляется тогда, когда стоимостные функции D_{ij} для линий обладают следующим свойством: производная $D'_{ij}(0)$ в нуле зависит только от пропускной способности линии и убывает с ростом пропускной способности (см. задачу 5.25).

Мы смогли решить предыдущий пример аналитически только в результате его чрезвычайной простоты. К сожалению, для решения более сложных задач нужно воспользоваться численными вычислениями — либо централизованными, либо распределенными. В последующих разделах рассматриваются некоторые из таких методов.

Наконец, заметим, что, несмотря на то что задача оптимальной маршрутизации была сформулирована с использованием фиксированных значений интенсивности входных потоков, ее оптимальное решение $\{x_p^*\}$ может естественным образом применяться и в случае меняющихся во времени интенсивностей входных потоков. Это можно сделать, если действовать с частями потоков

$$\xi_p = \frac{x_p^*}{r_w} \quad \text{для всех } p \in P_w,$$

и потребовать, чтобы каждая ОА-пара w делила трафик (пакеты или виртуальные цепи) между имеющимися путями в соответствии с этими долями. В сетях с виртуальными цепями, в которых узлы-отправители, как правило, имеют подробную информацию о путях, используемых при маршрутизации, легко реализовать процесс маршрутизации с учетом частей ξ_p . Узлы-отправители могут оценить интенсивности потоков, проходящих по виртуальным цепям, и выбрать маршруты для новых виртуальных цепей

таким образом, чтобы реальные доли потоков для каждого пути были как можно ближе к желаемым. В некоторых дейтаграммных сетях или в сетях, в которых узлам-отправителям неизвестна подробная информация о путях, может быть необходимо внести в маршрутную таблицу каждого узла значение маршрутной переменной $\varphi_{ik}(j)$ для каждой линии (i, k) и узла-адресата j . Эта маршрутная переменная определяется как доля всего потока, поступающего в узел i , адресованного узлу j и направленного по линии (i, k) . Математически это записывается как

$$\varphi_{ik}(j) = \frac{f_{ik}(j)}{\sum_m f_{im}(j)} \quad \text{для всех } (i, k) \text{ и } j, \quad (5.62)$$

где $f_{ik}(j)$ — поток, проходящий по линии (i, k) и адресованный узлу j . Если имеется оптимальное решение задачи маршрутизации, выраженное через путевые потоки $\{x_p^*\}$, то можно определить соответствующие ему потоки на линиях $f_{ik}(j)$ и по формуле (5.62) получить значения соответствующих маршрутных переменных $\varphi_{ik}(j)$. Непосредственная формулировка и описание распределенного алгоритма для нахождения решения оптимальной маршрутизации с использованием переменных $\varphi_{ik}(j)$ имеются в литературе [19, 29, 82, 87].

5.6. Методы допустимого направления для оптимальной маршрутизации

В предыдущем разделе было показано, что оптимальная маршрутизация достигается только тогда, когда поток каждой ОА-пары направляется по путям, имеющим минимальную первопроизводную длину (МППД-путь). Это равносильно тому, что набор путевых потоков является строго подоптимальным, если только какая-либо положительная часть потока направлена по пути, не являющемуся МППД-путем. При этом предполагается, что подоптимальная маршрутизация может быть улучшена переброской на МППД-путь потока с других путей для каждой ОА-пары. При использовании адаптивного метода отыскания кратчайшего пути для дейтаграммных сетей из подразд. 5.2.5 делается нечто в этом роде, но для каждой ОА-пары перебрасывается на кратчайший путь *весь* поток, что приводит к колебаниям. Целесообразнее перебрасывать на кратчайший путь только *часть* потока с других путей. В этом разделе рассматриваются методы, основанные на этой идее. В основном эти методы численно решают задачу оптимальной маршрутизации уменьшением стоимостной функции путем малых изменений в путевых потоках.

Для заданного вектора допустимых путевых потоков $x = \{x_p\}$ (т. е. для некоторого вектора x , удовлетворяющего огра-

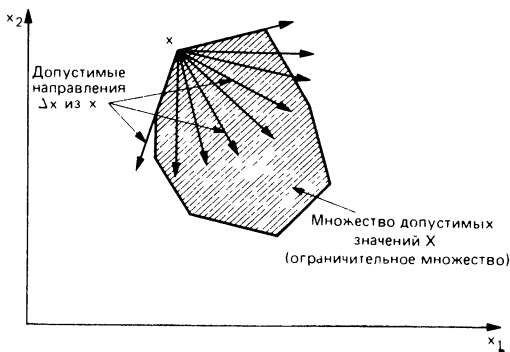


Рис. 5.49. Допустимые направления Δx из допустимой точки x . Направление Δx является допустимым, если при малом изменении x вдоль направления Δx допустимость сохраняется.

нижениям рассматриваемой задачи) рассмотрим изменение x по направлению $\Delta x = \{\Delta x_p\}$. К направлению Δx предъявляются два основных требования.

1. Первое требование состоит в том, чтобы Δx было *допустимым направлением* в том смысле, что x при малом изменении вдоль Δx оставался вектором допустимых путевых потоков (рис. 5.49). Математически это требование означает, что для некоторого $\bar{\alpha} > 0$ и всех $\alpha \in [0, \bar{\alpha}]$ вектор $x + \alpha \Delta x$ должен быть допустимым, или, что одно и то же,

$$\sum_{p \in P_w} \Delta x_p = 0 \quad \text{для всех } w \in W, \quad (5.63)$$

$$x_p + \alpha \Delta x_p \geq 0 \quad \text{для всех } \alpha \in [0, \bar{\alpha}], \quad p \in P_w, \quad w \in W. \quad (5.64)$$

Равенство (5.63) вытекает из требования допустимости

$$\sum_{p \in P_w} (x_p + \alpha \Delta x_p) = r_w$$

и того факта, что вектор x сам является допустимым; последнее означает

$$\sum_{p \in P_w} x_p = r_w.$$

Оно просто выражает тот факт, что сохранение допустимости необходимо, чтобы все увеличения потоков по каким-либо одним путям компенсировались уменьшением потоков по другим путям той же ОА-пары. Можно получить допустимое направление, например взяв любой другой допустимый вектор \bar{x} и положив

$$\Delta x = \bar{x} - x.$$

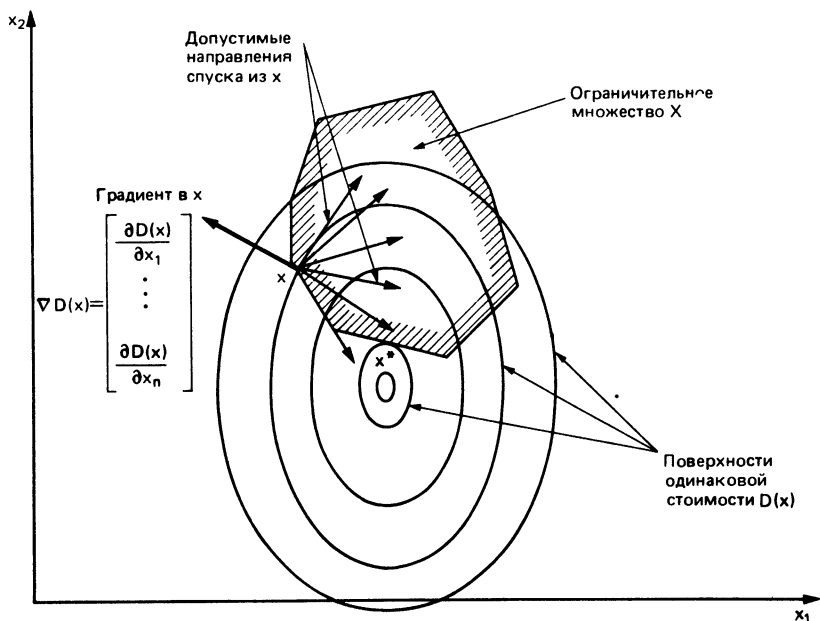


Рис. 5.50. Направление спуска из допустимой точки x образует угол больше 90° с градиентом $\nabla D(x)$. В точке оптимума x^* допустимых направлений спуска нет.

В действительности легко заметить, что с точностью до скалярного множителя всевозможные допустимые направления могут быть получены этим способом.

2. Второе требование состоит в том, чтобы Δx было *направлением спуска*, т. е. чтобы стоимостная функция убывала в направлении Δx от x (рис. 5.50). Так как градиент $\nabla D(x)$ является нормалью к поверхностям равной стоимости для стоимостной функции D , из рис. 5.50 становится ясно, что требование о спуске переходит в условие, что скалярное произведение $\nabla D(x)$ и Δx отрицательно, т. е.

$$\sum_{w \in W} \sum_{p \in P_w} \frac{\partial D(x)}{\partial x_p} \Delta x_p < 0. \quad (5.65)$$

Это условие можно проверить математически, заметив, что первая производная функции $G(\alpha) = D(x + \alpha \Delta x)$ при $\alpha = 0$ равна скалярному произведению в (5.65). Заметим, что частную производную $\partial D(x)/\partial x_p$ можно выразить в виде

$$\frac{\partial D(x)}{\partial x_p} = \sum_{\substack{\text{по всем линиям } (i, l) \\ \text{на пути } p}} D'_{il}(F_{il}),$$

и рассматривать как первопроизводную длину пути p [длина линии (i, j) равна $D'_{ij}(F_{ij})$, см. разд. 5.6]. Для того чтобы удовлетворить условию (5.65), которое часто используется в алгоритмах, можно потребовать, например, чтобы Δx удовлетворяло условию сохранения потока (5.63) и чтобы

$\Delta x_p \leq 0$ для всех неkratчайших путей p , т. е. таких путей, для которых $\partial D(x)/\partial x_p > \partial D(x)/\partial x_{\bar{p}}$ для некоторого пути \bar{p} , соответствующего той же ОА-паре;

(5.66)

$\Delta x_p < 0$ для по крайней мере одного неkratчайшего пути p . Условия (5.63) и (5.66) означают, что какая-то часть потока перебрасывается с неkratчайших путей на кратчайшие (по отношению к длинам $D'_{ij}(F_{ij})$) и никакой поток не перебрасывается с кратчайших путей на неkratчайшие. Так как $\partial D(x)/\partial x_p$ принимает наименьшие значения для тех (кратчайших) путей \bar{p} , для которых $\Delta x_{\bar{p}} > 0$ и в соответствии с (5.63) сумма Δx_p по $p \in P_w$ равна нулю, то видно, что из (5.63) и (5.66) вытекает условие спуска (5.65).

Это приводит к широкому классу итеративных алгоритмов для решения задачи оптимальной маршрутизации. Основной итерацией этих алгоритмов является

$$x := x + \alpha \Delta x,$$

где Δx — допустимое направление спуска (т. е. удовлетворяющее условиям (5.63)—(5.65) или (5.63)—(5.66)), а величина шага α выбирается таким образом, чтобы стоимостная функция убывала, т. е.

$$D(x + \alpha \Delta x) < D(x)$$

и вектор $x + \alpha \Delta x$ был допустимым. Размер шага α может выбираться для каждой итерации отдельно. Из условий оптимальности из разд. 5.5 видно, что допустимые направления спуска из точки x можно найти тогда и только тогда, когда x не является оптимальным решением. Рис. 5.51 иллюстрирует работу одного из таких алгоритмов.

5.6.1. Метод Франка — Волфа (метод девиации потока)

Здесь дается один из способов реализации идеи малых изменений вдоль допустимых направлений спуска.

Для данного вектора допустимых путевых потоков $x = \{x_p\}$ найти путь с минимальной первопроизводной длиной (МППД-путь) для каждой ОА-пары. (Значения первых производных D'_{ij} берутся, естественно, в точке x .) Пусть $\bar{x} = \{\bar{x}_p\}$ — вектор путевых потоков, который образуется, если входной поток r_w каждой ОА-пары $w \in W$ направлен по соответствующему МППД-пути.

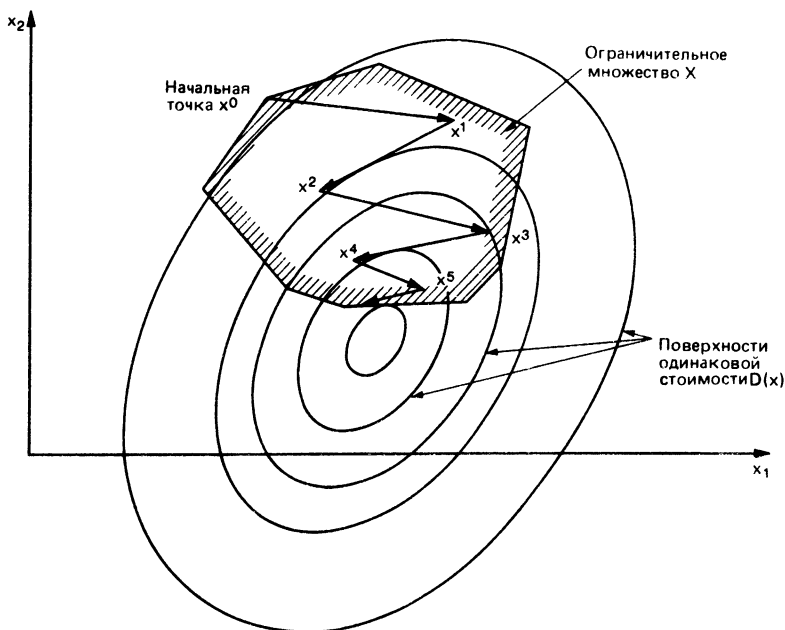


Рис. 5.51. Примерный ход итеративного метода спуска, основанного на допустимых направлениях спуска. На каждой итерации получается допустимая точка с меньшей стоимостью.

Пусть α^* — такой размер шага, который минимизирует $D[x + \alpha(\bar{x} - x)]$ по всем $\alpha \in [0, 1]$, т. е.

$$D[x + \alpha^*(\bar{x} - x)] = \min_{\alpha \in [0, 1]} D[x + \alpha(\bar{x} - x)]. \quad (5.67)$$

Тогда значения путевых потоков на следующем шаге равны

$$x_p := x_p + \alpha^*(\bar{x}_p - x_p) \quad \text{для всех } p \in P_w, \quad w \in W \quad (5.68)$$

и процесс повторяется снова.

Описанный алгоритм является частным случаем так называемого метода Франка — Волфа решения более общих задач нелинейного программирования с выпуклыми ограничительными множествами (см. [258]). Он также называется *методом девиации потока* (см. [70]); и можно показать, что в пределе он уменьшает значение стоимостной функции до минимума (см. задачу 5.31), хотя по мере приближения к оптимуму скорость сходимости становится очень малой.

Заметим, что допустимое направление, используемое в итерации (5.68), имеет вид (5.66), т. е. доля α^* каждого потока, проходящего по некратчайшему пути (пути, для которого $\bar{x}_p = 0$),



Итерация(номер) k	0	10	20	40	80	160	320
x_1	0,4	0,4593	0,4702	0,4795	0,4866	0,4917	0,4950
x_2	0,3	0,4345	0,4562	0,4717	0,4823	0,4893	0,4938
x_3	0,3	0,1061	0,0735	0,0490	0,0310	0,0189	0,0110
Стоимость	0,2945	0,2588	0,2553	0,2532	0,2518	0,2510	0,2506
Отношение ошибок $\frac{D(x^{k+1}) - D(x^*)}{D(x^k) - D(x^*)}$	0,7164	0,9231	0,9576	0,9774	0,9882	0,9939	0,9969

Рис. 5.52. Задача из примера и последовательные итерации метода Франка—Волфа. Стоймостьная функция имеет вид

$$D(x) = \frac{1}{2}(x_1^2 + x_2^2 + x_3^2) + 0,55x_3.$$

Оптимальным решением является $x^* = (1/2, 1/2, 0)$. По мере приближения к оптимальному решению метод замедляется. В пределе отношение последовательных ошибок

$$D(x^{k+1}) - D(x^*) / D(x^k) - D(x^*) \text{ стремится к единице.}$$

перебрасывается на кратчайший путь (для которого $\bar{x}_p = r_w$) для каждой ОА-пары w . Характерной особенностью здесь является то, что поток перебрасывается с кратчайших путей в *равных* пропорциях. Это отличает метод Франка — Волфа от градиентных проекционных методов, о которых речь пойдет в следующем разделе. Последние методы также перебрасывают потоки с кратчайших путей на кратчайшие пути, однако они делают это, как правило, в *неодинаковых* пропорциях.

Ниже приводится простой пример, иллюстрирующий метод Франка — Волфа.

Пример

Рассмотрим сеть, состоящую из трех линий, одного узла-отправителя и одного узла-адресата, показанную на рис. 5.52. Имеются три пути с соответствующими потоками x_1 , x_2 и x_3 , которые удовлетворяют ограничениям

$$x_1 + x_2 + x_3 = 1, \quad x_1 \geq 0, \quad x_2 \geq 0, \quad x_3 \geq 0.$$

Стоймостной функцией является

$$D(x) = \frac{1}{2}(x_1^2 + x_2^2 + 0,1x_3^2) + 0,55x_3.$$

(Линейный член $0,55x_3$ можно приписать большому времени обработки и большому времени распространения сигнала по линии 3.)

Это простая задача, которую можно решить аналитически. Можно доказать (или показать, используя условие оптимальности из разд. 5.5), что для оптимального решения $x^* = (x_1^*, x_2^*, x_3^*)$ выполняется симметрия $x_1^* = x_2^*$ и поэтому имеются две возможности: (а) $x_3^* = 0$ и $x_1^* = x_2^* = 1/2$, (б) $x_3^* = \beta > 0$ и $x_1^* = x_2^* = (1 - \beta)/2$. Случай (б) невозможен, так как согласно условию оптимальности из разд. 5.5, если $x_3^* > 0$, то длина пути 3 ($= \partial D(x^*)/\partial x_3 = 0,1\beta + 0,55$) должна быть меньше или равна длинам путей 1 и 2 ($= \partial D(x^*)/\partial x_1 = (1 - \beta)/2$), что, очевидно, не может быть. Поэтому оптимальным решением является $x^* = (1/2, 1/2, 0)$.

Рассмотрим теперь применение итерации Франка—Волфа (5.68) к вектору допустимых потоковых потоков $x = (x_1, x_2, x_3)$. Первопроизводные длины всех трех путей равны

$$\frac{\partial D(x)}{\partial x_1} = x_1, \quad \frac{\partial D(x)}{\partial x_2} = x_2, \quad \frac{\partial D(x)}{\partial x_3} = 0,1x_3 + 0,55.$$

Используя, по существу, те же доводы, что и раньше, замечаем, что кратчайшим путем будет либо 1, либо 2 в зависимости от того, что $x_1 \leq x_2$ или $x_2 < x_1$, и соответствующим потоком по кратчайшему пути будет либо $\bar{x} = (1, 0, 0)$, либо $\bar{x} = (0, 1, 0)$ соответственно. (В случае $x_1 = x_2$ предпочтение отдается пути 1.) Поэтому итерации Франка—Волфа имеют вид

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} := \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \alpha^* \begin{bmatrix} 1 - x_1 \\ -x_2 \\ -x_3 \end{bmatrix} = \begin{bmatrix} x_1 + \alpha^* (x_2 + x_3) \\ (1 - \alpha^*) x_2 \\ (1 - \alpha^*) x_3 \end{bmatrix}, \quad \text{если } x_1 \leq x_2;$$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} := \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \alpha^* \begin{bmatrix} -x_1 \\ 1 - x_2 \\ -x_3 \end{bmatrix} = \begin{bmatrix} (1 - \alpha^*) x_1 \\ x_2 + \alpha^* (x_1 + x_3) \\ (1 - \alpha^*) x_3 \end{bmatrix}, \quad \text{если } x_2 < x_1.$$

Таким образом, на каждой итерации доля α^* потока по некротчайшему пути перебрасывается на кратчайший путь.

Размер шага α^* вычисляется путем линейной минимизации на $[0, 1]$ (ср. с (5.67)). Так как $D(x)$ является квадратичной функцией, эту минимизацию можно проделать аналитически. Имеем

$$D[x + \alpha(\bar{x} - x)] = \frac{1}{2} [(x_1 + \alpha(\bar{x}_1 - x_1))^2 + [x_2 + \alpha(\bar{x}_2 - x_2)]^2 + \\ + 0,1 [x_3 + \alpha(\bar{x}_3 - x_3)]^2 + 0,55 [x_3 + \alpha(\bar{x}_3 - x_3)].$$

Дифференцируя это выражение по α и приравнявая производную нулю, получаем значение $\bar{\alpha}$, при котором достигается минимум функции $D[x + \alpha(\bar{x} - x)]$ по α без учета ограничений. Оно равно

$$\bar{\alpha} = - \frac{x_1(\bar{x}_1 - x_1) + x_2(\bar{x}_2 - x_2) + (0,1x_3 + 0,55)(\bar{x}_3 - x_3)}{(\bar{x}_1 - x_1)^2 + (\bar{x}_2 - x_2)^2 + 0,1(\bar{x}_3 - x_3)^2},$$

где $\bar{x} = (\bar{x}_1, \bar{x}_2, \bar{x}_3)$ — текущее значение вектора потока по кратчайшему пути $(1, 0, 0)$ или $(0, 1, 0)$ в зависимости от того, что $x_1 \leq x_2$ или $x_2 < x_1$. Так как $(\bar{x} - x)$ является направлением спуска, то $\bar{\alpha} \geq 0$. Оптимальное значение размера шага дает минимум с учетом ограничения, что α принадлежит $[0, 1]$ (см. (5.67)), поэтому

$$\alpha^* = \min [1, \bar{\alpha}].$$

(Можно проверить, что в этом примере $\bar{\alpha} < 1$ и, следовательно, $\alpha^* = \bar{\alpha}$.)

На рис. 5.52 и 5.53 показаны последовательные итерации этого метода. Можно заметить, что скорость сходимости замедляется вблизи оптимального ре-

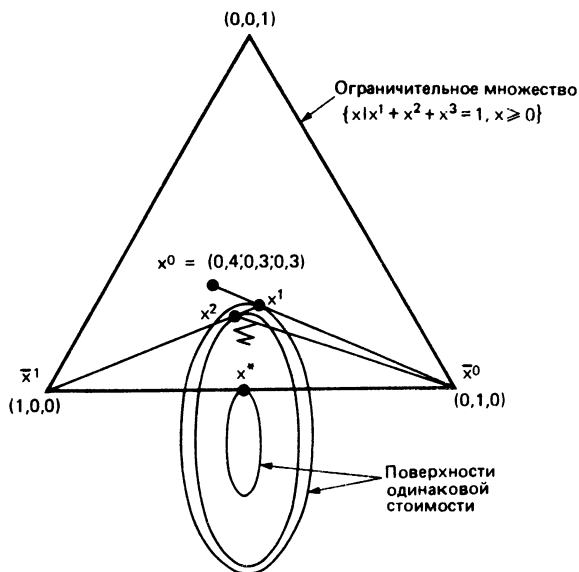


Рис. 5.53. Последовательные итерации метода Франка—Волфа в примере с двумя узлами и тремя линиями. Медленная сходимость объясняется тем, что при приближении к оптимальному решению x^* направления поиска становятся ортогональны направлению, приводящему к оптимуму.

шения. Причина состоит в том, что при приближении к x^* направления поиска становятся ортогональны направлению, ведущему от текущего значения x^k к x^* . Как показано в таблице на рис. 5.52, отношение последовательных ошибок в стоимости

$$\frac{D(x^{k+1}) - D(x^*)}{D(x^k) - D(x^*)},$$

хотя всегда и меньше единицы, но фактически стремится к единице при $k \rightarrow \infty$. Эта величина известна как *сублинейная* скорость сходимости [25]; она типична для метода Франка—Волфа [40, 58].

Для вычисления оптимального значения размера шага α^* , удовлетворяющего (5.67), требуется провести минимизацию по одной переменной на $[0, 1]$, а для этого можно использовать какой-либо из многочисленных существующих методов (см. [165]). Более простым методом является выбор размера шага α^* в (5.68) в соответствии с равенством

$$\alpha^* = \min \left[1, - \frac{\sum_{(i,j)} (\bar{F}_{ij} - F_{ij}) D'_{ij}}{\sum_{(i,j)} (\bar{F}_{ij} - F_{ij})^2 D''_{ij}} \right]. \quad (5.69)$$

Здесь $\{F_{ij}\}$ и $\{\bar{F}_{ij}\}$ являются множествами суммарных потоков, соответствующих $\{x_p\}$ и $\{\bar{x}_p\}$ соответственно, т. е. F_{ij} (или \bar{F}_{ij}) получается путем суммирования всех потоков x_p (или \bar{x}_p) по тем путям p , которые проходят по линии (i, j) . Первая и вторая производные D'_{ij} и D''_{ij} берутся в точке F_{ij} . Формула (5.69) для α^* получена на основе тейлоровской аппроксимации второго порядка $\tilde{G}(\alpha)$ для функции $G(\alpha) = D[x + \alpha(\bar{x} - x)]$ в точке окрестности $\alpha = 0$

$$\begin{aligned}\tilde{G}(\alpha) = \sum_{(i, j)} \{D_{ij}(F_{ij}) + \alpha D'_{ij}(F_{ij})(\bar{F}_{ij} - F_{ij}) + \\ + (\alpha^2/2) D''_{ij}(F_{ij})(\bar{F}_{ij} - F_{ij})^2\}\end{aligned}$$

и минимизации $\tilde{G}(\alpha)$ по α на отрезке $[0, 1]$, как в предыдущем примере.

Можно показать, что рассматриваемый алгоритм с размером шага (5.69) сходится к набору оптимальных суммарных потоков в линиях, если вначале суммарные потоки в линиях были достаточно близки к оптимальным. Оказывается, что для стоимостных функций, используемых в задачах маршрутизации (например, (5.30) из разд. 5.4), размер шага (5.69) обычно приводит к сходимости, даже если начальные суммарные потоки в линиях далеки от оптимальных. Правило выбора размера шага (5.69) является также более подходящим для распределенных реализаций, чем правило (5.67) (см. задачу 5.26). Однако даже с этим правилом рассматриваемый метод все же хуже подходит для распределенных приложений, чем проекционный метод из следующего раздела в основном из-за того, что для вычисления размера шага (5.69) необходимо, чтобы узлы сети синхронизировали свои вычисления.

Метод Франка — Волфа имеет наглядную геометрическую интерпретацию. Предположим, что имеется вектор допустимых путей потоков $x = \{x_p\}$ и нужно найти приращение путей потоков $\Delta x = \{\Delta x_p\}$, которое допустимо в смысле (5.63) и (5.64), т. е.

$$\begin{aligned}\sum_{p \in P_w} \Delta x_p = 0 \quad \text{для всех } w \in W, \\ x_p + \Delta x_p \geq 0 \quad \text{для всех } p \in P_w, \quad w \in W,\end{aligned}\tag{5.70}$$

и вдоль которого начальная скорость изменения (см. (5.65)) стоимостной функции D

$$\sum_{w \in W} \sum_{p \in P_w} \frac{\partial D(x)}{\partial x_p} \Delta x_p$$

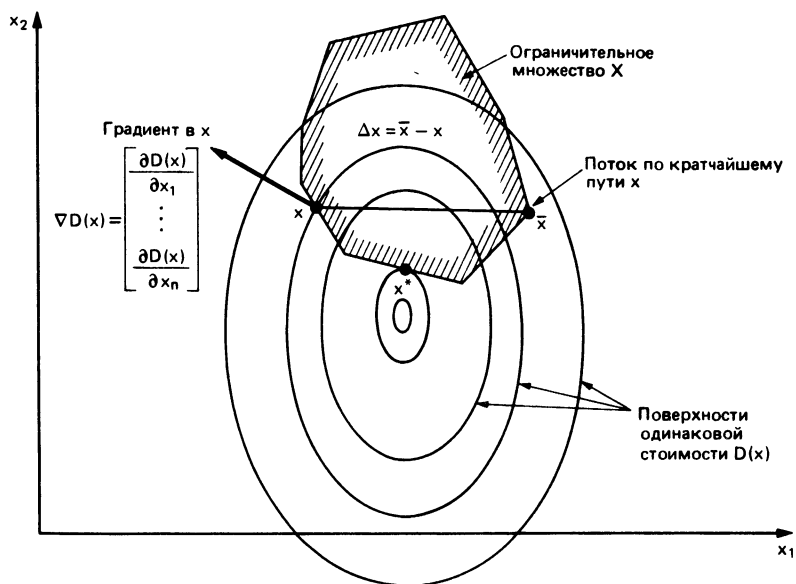


Рис. 5.54. Поиск допустимого направления спуска $\Delta x = \bar{x} - x$ из точки x в методе Франка—Волфа. Поток \bar{x} является потоком по кратчайшему пути (или крайняя точка ограничительного множества X), который лежит дальше других в направлении отрицательного градиента $-\nabla D(x)$.

имеет наибольшее по абсолютной величине отрицательное значение. Такое приращение Δx получается из решения оптимизационной задачи

минимизировать
$$\sum_{w \in W} \sum_{p \in P_w} \frac{\partial D(x)}{\partial x_p} \Delta x_p \text{ при ограничениях (5.70).}$$

Легко заметить, что $\Delta x = \bar{x} - x$ является оптимальным решением, где \bar{x} — вектор потоков по кратчайшим путям, используемый в методе Франка — Волфа. Процесс нахождения Δx можно наблюдать на рис. 5.54. Видно, что \bar{x} является вершиной многогранника векторов допустимых путевых потоков, которая лежит дальше других в направлении отрицательного градиента $-\nabla D(x)$ и поэтому минимизирует скалярное произведение $\nabla D(x)$ и Δx , удовлетворяющего ограничениям (5.70).

Последовательные итерации метода Франка — Волфа показаны на рис. 5.55. На каждой итерации вершина допустимого множества (т. е. поток по кратчайшим путям \bar{x}) получается из решения задачи нахождения кратчайших путей, а следующий вектор путевых потоков ищется на линии, соединяющей текущий вектор путевых потоков с этой вершиной. Такой поиск гаранти-

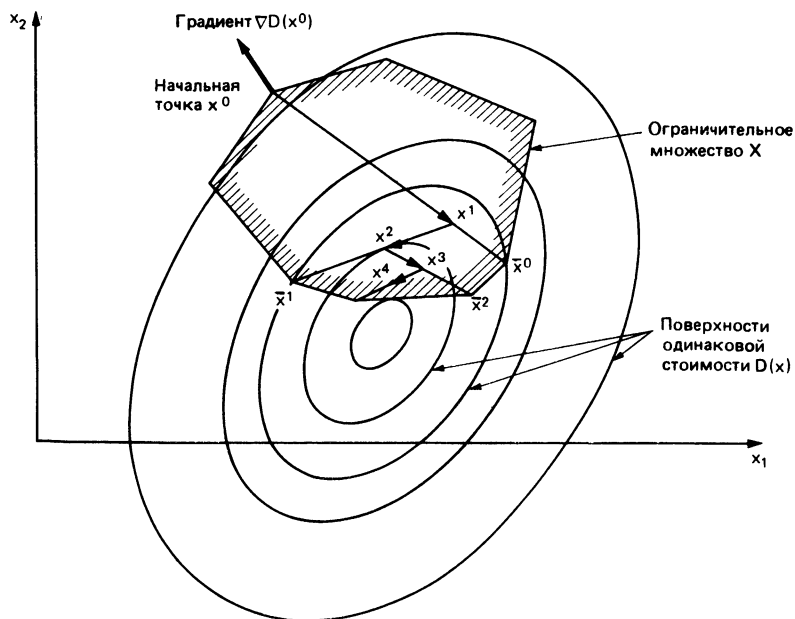


Рис. 5.55. Последовательные итерации метода Франка—Волфа. На каждой итерации находится вершина ограничительного множества (поток по кратчайшему пути). Следующий вектор путевых потоков получается из поиска на отрезке прямой, соединяющей текущий вектор путевых потоков с этой вершиной.

рует, что новый вектор путевых потоков будет иметь стоимость, меньшую, чем предыдущий.

Наконец, упомянем важную ситуацию, в которой метод Франка — Волфа имеет преимущество по сравнению с другими методами. Предположим, что нас интересуют не оптимальные путевые потоки, а только оптимальные суммарные потоки на линиях F_{ij} или только значение оптимальной стоимости. (Такие вопросы возникают при построении топологии; см. подразд. 5.4.2.) Тогда метод Франка — Волфа можно реализовать таким образом, что на каждой итерации в памяти будут храниться только текущие значения суммарных потоков на линиях и текущие кратчайшие пути для всех ОА-пар. Это можно сделать, вычисляя суммарные потоки на линиях, соответствующих вектору потоков по кратчайшим путям \bar{x} , и выполняя линейный поиск, указанный в (5.67), в пространстве суммарных потоков на линиях. Объем памяти, который необходим для такой реализации, относительно небольшой, вследствие чего эту задачу можно решать для очень больших сетей. Если, однако, кроме суммарных потоков на линиях нас

интересуют также оптимальные путевые потоки, то при использовании метода Франка — Волфа требуется объем памяти, примерно такой же, как и для других методов, включая проекционные методы, о которых речь пойдет в следующем разделе.

5.7. Проекционные методы для оптимальной маршрутизации

Рассмотрим теперь некоторый класс алгоритмов допустимого направления для решения задачи оптимальной маршрутизации; эти алгоритмы работают быстрее метода Франка — Волфа и больше приспособлены для распределенных реализаций. Они также основаны на кратчайших путях и находят путь с минимальной первопроизводной длиной для каждой ОА-пары на каждой итерации. Приращение потока вычисляется для каждого пути на основе относительных величин длин путей и иногда на основе вторых производных стоимостной функции. Если это приращение оказывается настолько большим, что путевой поток становится отрицательным, то путевой поток полагается равным нулю, т. е. он «проектируется» обратно в положительный октант. Существует несколько методов такого типа, которые представляют интерес в связи с задачей маршрутизации. Все они могут рассматриваться как версии с ограничением более общих методов оптимизации без ограничений, таких как метод наискорейшего спуска и метод Ньютона, которые более подробно описаны в литературе по нелинейному программированию (например, [25, 165, 192, 258]). Сначала мы кратко опишем эти методы в общей постановке нелинейной оптимизации, а затем рассмотрим приложение этих методов к задаче маршрутизации.

Нелинейная оптимизация без ограничений

Пусть f — дважды дифференцируемая функция n -мерного вектора $x = (x_1, \dots, x_n)$; градиентная матрица и матрица Гесса в любой точке x , обозначенные через $\nabla f(x)$ и $\nabla^2 f(x)$ соответственно, имеют вид

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix},$$

$$\nabla^2 f(x) = \begin{bmatrix} \frac{\partial^2 f(x)}{(\partial x_1)^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f(x)}{(\partial x_n)^2} \end{bmatrix}.$$

Будем предполагать, что для всех x матрица $\nabla^2 f(x)$ является неотрицательно определенной, непрерывной по x^1). Можно показать, что отсюда следует выпуклость функции f (см. [165], с. 180). Метод наискорейшего спуска для нахождения минимума функции f при отсутствии ограничений начинается с некоторой произвольной начальной точки x^0 и действует в соответствии с итерацией

$$x^{k+1} = x^k - \alpha^k \nabla f(x^k), \quad k = 0, 1, \dots, \quad (5.71)$$

где размер шага α^k является положительным числом, определяемым в соответствии с некоторым правилом. Идея здесь похожа на ту, которая обсуждалась в предыдущем разделе, а именно изменять x^k в направлении спуска. В (5.71) изменение делается в направлении отрицательного градиента, которое является направлением спуска, так как это делает отрицательным скалярное произведение с вектором градиента (кроме случая, когда $\nabla f(x^k) = 0$, при этом вектор x^k является оптимальным). Величина α^k либо выбирается путем минимизации выражения

$$f[x^k - \alpha^k \nabla f(x^k)] = \min_{\alpha > 0} f[x^k - \alpha \nabla f(x^k)], \quad (5.72)$$

либо полагается равной постоянной положительной величине $\bar{\alpha}$

$$\alpha^k \equiv \bar{\alpha} \quad \text{для всех } k. \quad (5.73)$$

Существует много результатов, относящихся к сходимости итерации (5.71) с размером шага (5.72) или (5.73). Например, если у f только одна точка минимума при отсутствии ограничений, то можно показать, что последовательность $\{x^k\}$, порожденная (5.71) и (5.72), сходится к точке минимума при любом начальном x^0 (см. [25, 165]). Точно так же при любом начальном векторе x^0 последовательность, порожденная (5.71) и (5.73), сходится к точке минимума, если $\bar{\alpha}$ достаточно мало. К сожалению, однако, скорость сходимости $\{x^k\}$ может быть достаточно малой. Можно показать ([165], с. 218), что в случае правила линейной минимизации (5.72), если f является положительно определенной квадратичной функцией, то имеет место неравенство

$$\frac{f(x^{k+1}) - f^*}{f(x^k) - f^*} \leq \left(\frac{M - m}{M + m} \right)^2, \quad (5.74)$$

¹⁾ Симметричная $n \times n$ матрица A с элементами A_{ij} называется *положительно полуопределенной*, если квадратичная форма

$$\sum_{i=1}^n \sum_{j=1}^n A_{ij} z_i z_j$$

неотрицательна для всех векторов $z = (z_1, \dots, z_n)$. Матрица A *положительно определенная*, если эта квадратичная форма положительна для всех векторов $z \neq 0$.

где $f^* = \min_x f(x)$, а M и m соответственно максимальное и минимальное собственные значения матрицы $\nabla^2 f(x)$. Кроме того, существует такая начальная точка x^0 , для которой неравенство (5.74) превращается в равенство для каждого k . Таким образом, если отношение M/m велико (в этом случае поверхности одинаковых значений функции f являются очень вытянутыми эллипсоидами), то скорость сходимости мала. Аналогичные результаты можно доказать для метода, определяемого (5.71) и (5.73), причем эти результаты качественно остаются теми же для функции f со всюду непрерывной и положительно определенной матрицей Гесса.

Скорость сходимости метода наискорейшего спуска можно улучшить, если предварительно умножить вектор градиента на соответствующим образом подобранную положительно определенную масштабную матрицу B^k , в результате чего получается итерация

$$x^{k+1} = x^k - \alpha^k B^k \nabla f(x^k), \quad k = 0, 1, \dots \quad (5.75)$$

Этот метод основан на идее изменения x^k вдоль направления спуска. (Скалярное произведение направления изменения $-B^k \nabla f(x^k)$ и градиента $\nabla f(x^k)$ отрицательно, так как B^k является положительно определенной матрицей.) С точки зрения скорости сходимости наилучший метод получается, если

$$B^k = [\nabla^2 f(x^k)]^{-1} \quad (5.76)$$

(в предположении, что матрица $\nabla^2 f(x^k)$ не вырождена). Это *метод Ньютона*, который, как можно показать, имеет очень быструю (квадратичную) скорость сходимости вблизи точки минимума, если размер шага α^k выбран равным единице. К сожалению, эта хорошая скорость сходимости достигается за счет того, что при вычислении обратной матрицы в (5.76) может потребоваться очень большой объем вычислений. Часто оказываются полезными другие способы выбора B^k , при которых аппроксимируется оптимальный выбор $[\nabla^2 f(x^k)]^{-1}$, но не требуется столь большого объема вычислений. Один из простых способов, который часто дает неплохие результаты, состоит в том, что в качестве B^k берется диагональная аппроксимация обратной матрицы Гесса, т. е.

$$B^k = \begin{bmatrix} \left[\frac{\partial^2 f(x^k)}{(\partial x_1)^2} \right]^{-1} & & & \\ & \left[\frac{\partial^2 f(x^k)}{(\partial x_2)^2} \right]^{-1} & & 0 \\ & & \dots & \\ 0 & & & \left[\frac{\partial^2 f(x^k)}{(\partial x_n)^2} \right]^{-1} \end{bmatrix}. \quad (5.77)$$

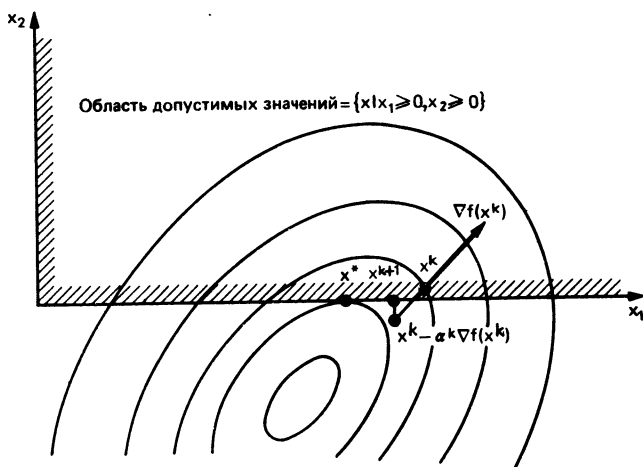


Рис. 5.56. Работа проекционного градиентного алгоритма. Делается шаг в направлении отрицательного градиента, и результат ортогонально проецируется на положительный октант. Можно показать, что шаг $(x^{k+1} - x^k)$ является направлением спуска (скалярное произведение с $\nabla f(x^k)$ будет отрицательным), и если размер шага α выбран достаточно малым, то итерация уменьшит стоимость, т. е. $f(x^{k+1}) < f(x^k)$.

При таком выборе итерацию (5.75) можно переписать в более простом виде

$$x_i^{k+1} = x_i^k - \alpha^k \left[\frac{\partial^2 f(x^k)}{(\partial x_i)^2} \right]^{-1} \frac{\partial f(x^k)}{\partial x_i}, \quad i = 1, \dots, n. \quad (5.78)$$

Нелинейная оптимизация в положительном октанте

Далее рассмотрим задачу минимизации функции f при ограничениях $x_i \geq 0$ для $i = 1, \dots, n$, т. е. задачу

минимизировать $f(x)$

при ограничениях $x \geq 0$. (5.79)

Прямым аналогом метода наискорейшего спуска будет

$$x^{k+1} = [x^k - \alpha^k \nabla f(x^k)]^+, \quad k = 0, 1, \dots, \quad (5.80)$$

где для любого вектора z мы обозначаем через $[z]^+$ проекцию z на положительный октант

$$[z]^+ = \begin{bmatrix} \max \{0, z_1\} \\ \max \{0, z_2\} \\ \vdots \\ \max \{0, z_n\} \end{bmatrix}. \quad (5.81)$$

Этот метод проиллюстрирован на рис. 5.56. Можно показать, что результаты о сходимости, упомянутые ранее в связи с (5.71) — методом наискорейшего спуска без ограничений, также верны для аналога с ограничением (5.80) (см. задачу 5.32 и [183]). То же самое верно и для метода

$$x^{k+1} = [x^k - \alpha^k B^k \nabla f(x^k)]^+,$$

где B^k — диагональная, положительно определенная масштабная матрица (см. задачу 5.33). Если B^k выбрана в виде (5.77), то получается следующая итерация (в соответствии с (5.78) и (5.81)):

$$x_i^{k+1} = \max \left\{ 0, x_i^k - \alpha^k \left[\frac{\partial^2 f(x^k)}{(\partial x_i)^2} \right]^{-1} \frac{\partial f(x^k)}{\partial x_i} \right\}, \quad i = 1, \dots, n. \quad (5.82)$$

Приложение к оптимальной маршрутизации

Рассмотрим теперь задачу оптимальной маршрутизации

$$\text{минимизировать } D(x) \triangleq \sum_{(i,j)} D_{ij}(F_{ij})$$

$$\text{при ограничениях } \sum_{p \in P_w} x_p = r_w, \quad x_p \geq 0 \quad \text{для всех } p \in P_w, w \in W, \quad (5.83)$$

где каждый суммарный поток на линии F_{ij} выражается через вектор путевых потоков $x = \{x_p\}$ в виде суммы путевых потоков, проходящих по линии (i, j) . Предположим, что после k итераций получен вектор допустимых путевых потоков $x^k = \{x_p^k\}$ и $\{F_{ij}^k\}$ представляет собой соответствующее множество суммарных потоков на линиях. Для каждой ОА-пары w пусть \bar{p}_w является МППД-путем (по отношению к текущим длинам линий $D'_{ij}(F_{ij}^k)$). Задачу (5.83) можно преобразовать (для следующей итерации) в задачу, содержащую только ограничение положительности, если выразить потоки по МППД-путям \bar{p}_w через потоки по другим путям, исключая тем самым из условий задачи ограничения вида

$$\sum_{p \in P_w} x_p = r_w.$$

Для каждой w поток $x_{\bar{p}_w}$ подставляется в стоимостную функцию $D(x)$ из равенства

$$x_{\bar{p}_w} = r_w - \sum_{\substack{p \in P_w \\ p \neq \bar{p}_w}} x_p, \quad (5.84)$$

вследствие чего получается задача вида

$$\text{минимизировать } \tilde{D}(\tilde{x})$$

при ограничениях $x_p \geq 0$ для всех $w \in W$, $p \in P_w$, $p \neq \bar{p}_w$, (5.85)

где \tilde{x} — вектор всех путевых потоков, которые не являются МППД-путями.

Теперь вычислим производные, которые понадобятся для того, чтобы применить итерацию (5.82) к задаче (5.85). Используя (5.84) и определение $\tilde{D}(\tilde{x})$, получаем

$$\frac{\partial \tilde{D}(\tilde{x}^k)}{\partial x_p} = \frac{\partial D(x^k)}{\partial x_p} - \frac{\partial D(x^k)}{\partial x_{\bar{p}_w}} \quad \text{для всех } p \in P_w, \quad p \neq \bar{p}_w, \quad w \in W. \quad (5.86)$$

В разд. 5.5 было показано, что $\partial D(x)/\partial x_p$ является первопроизводной длиной пути p , т. е.

$$\frac{\partial D(x^k)}{\partial x_p} = \sum_{\substack{\text{по всем линиям } (i, l) \\ \text{на пути } p}} D'_{il}(F^k_{il}). \quad (5.87)$$

Вторые производные получаются путем непосредственного дифференцирования выражений (5.86) и (5.87)

$$\frac{\partial^2 \tilde{D}(\tilde{x}^k)}{(\partial x_p)^2} = \sum_{(i, l) \in L_p} D''_{il}(F^k_{il}) \quad \text{для всех } w \in W, \quad p \in P_w, \quad p \neq \bar{p}_w, \quad (5.88)$$

где для каждого p

L_p — множество линий, принадлежащих либо p , либо соответствующему МППД-пути \bar{p}_w , но не обоим сразу.

Теперь получены выражения как для первых, так и для вторых производных редуцированной стоимостной функции $\tilde{D}(\tilde{x})$ и поэтому можно применять метод проекций (5.82). Итерация имеет вид (см. (5.82)—(5.86), (5.88))

$$x_p^{k+1} = \max \{0, x_p^k - \alpha^k H_p^{-1}(d_p - d_{\bar{p}_w})\} \quad \text{для всех } w \in W, \quad p \in P_w, \quad p \neq \bar{p}_w, \quad (5.89)$$

где d_p и $d_{\bar{p}_w}$ являются первопроизводными длинами путей p и \bar{p}_w ; они равны (см. (5.87))

$$d_p = \sum_{\substack{\text{по всем линиям } (i, l) \\ \text{на пути } p}} D'_{il}(F^k_{il}), \quad d_{\bar{p}_w} = \sum_{\substack{\text{по всем линиям } (i, l) \\ \text{на пути } \bar{p}_w}} D'_{il}(F^k_{il}), \quad (5.90)$$

а H_p — второпроизводная длина

$$H_p = \sum_{(i, j) \in L_p} D_{ij}''(F_{ij}^k), \quad (5.91)$$

взятая из (5.88). Размер шага α^k является некоторым положительным числом, которое можно выбирать различными способами. Например, α^k может быть константой или его можно выбирать путем какой-либо линейной минимизации. Выбор размера шага будет обсуждаться позже.

Можно сделать следующие заключения относительно итерации (5.89):

1. Так как для каждой ОА-пары $w \in W$ мы имеем $d_p \geq d_{\bar{p}_w}$ для всех $p \neq \bar{p}_w$, то отсюда следует, что все потоки по некратчайшим путям x_p ($p \neq \bar{p}_w$), которые больше нуля, станут меньше на величину, которая будет переброшена на МППД-путь \bar{p}_w . Если размер шага α^k достаточно большой, то весь поток с некратчайших путей будет переброшен на кратчайший путь. Поэтому итерацию (5.89) можно также рассматривать как обобщение метода адаптивной маршрутизации, основанного на кратчайших путях (о котором речь шла в подразд. 5.2.5) и имеющего параметры α^k , H_p и (d_p — $d_{\bar{p}_w}$), определяющие величину потока, перебрасываемого на кратчайший путь.

2. Потоки по некратчайшим путям x_p , $p \neq \bar{p}_w$, которые равны нулю, останутся равными нулю. Поэтому вычисления, указанные в (5.89), следует проводить только для тех путей, по которым проходит положительный поток.

3. Только на тех путях, по которым проходил положительный поток при начальной картине потоков или которые были МППД-путями на какой-то предыдущей итерации, может быть положительный поток в начале итерации. Это имеет большое значение, так как отсюда следует, что алгоритм стремится сохранить число путей, несущих поток, небольшим, что положительно образом сказывается на количестве вычислений и объеме памяти, необходимом для выполнения каждой итерации.

Что касается выбора размера шага α^k , то существует несколько возможностей. Можно положить α^k равным константе ($\alpha^k \equiv \alpha$ для всех k). Для такого выбора можно показать, что для любого начального множества путевых потоков существует $\bar{\alpha} > 0$, такое что если $\bar{\alpha} \in (0, \bar{\alpha}]$, то последовательность, порожденная итерацией (5.89)—(5.91), сходится к оптимальной стоимости задачи (см. задачу 5.32). Выбор величины постоянного размера шага имеет большое значение. Из опыта нелинейного программирования и анализа известно, что размер шага, равный единице, обычно хорошо подходит для метода Ньютона, а также диагональной аппроксимации метода Ньютона (см. (5.77)), в которой используется масштабирование, основанное на вторых производных [23,

165]. Опыт показал, что выбор α^k в (5.89) близким к единице обычно оказывается достаточно хорошим независимо от значений входных потоков r_w [32]. Получается даже лучше, если итерацию (5.89) выполнять *за один раз только для одной ОА-пары (или для одного отправителя)*, т. е. сначала выполнить (5.89) с $\alpha^k = 1$ для единственной ОА-пары (или отправителя), затем пересчитать значения соответствующих суммарных потоков в линиях, учитывая изменение путевых потоков этой ОА-пары, и продолжить вычисления для следующей ОА-пары, циклически обходя все ОА-пары. Разумное объяснение этого основано на том, что, отбрасывая недиагональные члены в матрице Гесса (см. (5.75) и (5.77)), мы тем самым, по существу, пренебрегаем взаимодействием между потоками различных ОА-пар. Другими словами, итерация (5.89) в какой-то степени основана на предпосылке, что каждая ОА-пара будет перестраивать свои собственные путевые потоки, в то время как другие ОА-пары сохраняют свои потоки неизменными. Выполнение итерации (5.89) за один раз только для одной ОА-пары уменьшает потенциальный нежелательный эффект, связанный с отбрасыванием недиагональных членов в матрице Гесса, и увеличивает уверенность в том, что единичный размер шага является подходящим и эффективным. При этих обстоятельствах опыт показывает, что итерация (5.89) обычно работает хорошо с единичным размером шага.

Использование постоянного размера шага хорошо подходит для распределенных приложений. Для централизованных вычислений можно выбирать α^k путем какого-либо простого линейного поиска. Например, можно начать с единичного размера шага, вычислить соответствующую стоимость, и если никакого уменьшения по сравнению с $D(x^k)$ не получится, то постепенно уменьшать размер шага до тех пор, пока уменьшение $D(x^{k+1}) < D(x^k)$ не будет получено. Нельзя показать, что такая схема сходится к оптимальному решению. Однако на практике она обычно работает хорошо. Подобные схемы, у которых теоретические свойства сходимости лучше, описаны в работах [18] и [24]. Еще один вариант одномерного поиска обсуждается в задаче 5.32.

Проекционный алгоритм обычно быстро приводит в окрестность оптимального решения. Подойдя близко к решению (насколько «близко», зависит от конкретной задачи), алгоритм замедляет работу. Его скорость около решения часто является удовлетворительной и обычно намного большей, чем в методе Франка — Волфа. На рис. 5.57 показана для сравнения работа проекционного алгоритма и метода Франка — Волфа на примере, взятом из предыдущего раздела.

Для того чтобы проекционный алгоритм сходиллся быстрее вблизи оптимального решения, необходимо учитывать недиагональные члены в матрице Гесса. Удивительно, что удается реали-

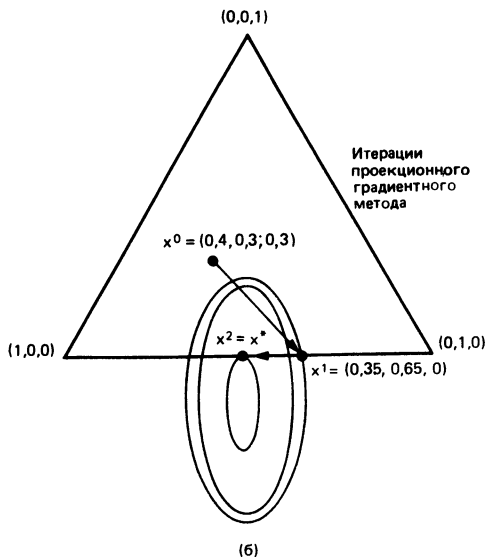
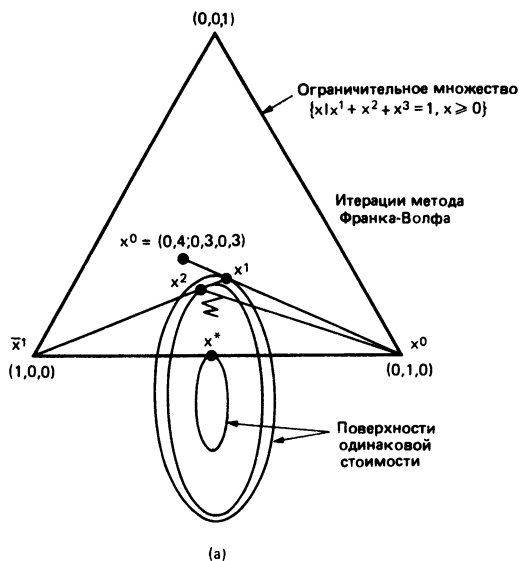


Рис. 5.57. Итерация (а) метода Франка—Волфа и (б) проекционного градиентного метода для примера из предыдущего раздела. Проекционный градиентный метод сходится намного быстрее. Он приравнивает поток x_2 правильной величине 0 за одну итерацию и (из-за того, что матрица Гесса здесь является диагональной и стоимостная функция является квадратичной) требует еще одну итерацию для окончательной сходимости.

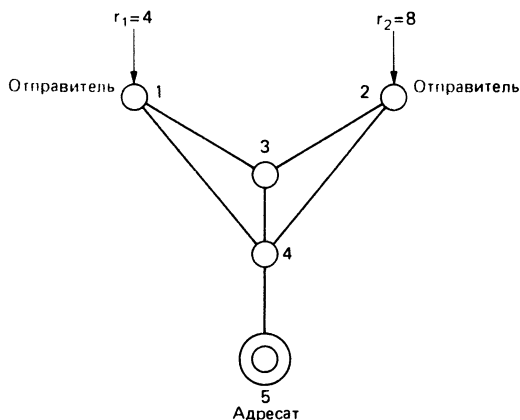


Рис. 5.58. Сеть для примера, в которой узел 5 является единственным адресатом и имеются два отправителя: узлы 1 и 2.

зовать сложные методы такого типа (см. [16]), но мы не будем вдаваться в подробности. Достаточно сказать, что эти методы основаны на более точной аппроксимации метода Ньютона для задач с ограничениями (с использованием метода сопряженных градиентов) вблизи оптимального решения. Однако вдали от решения их скорость сходимости оказывается обычно только немного выше, чем у итерации (5.89). Поэтому если важно, чтобы алгоритм как можно быстрее (за небольшое число итераций) приблизился к оптимальному решению, и не существенна скорость сходимости (как часто бывает в практических задачах маршрутизации), то обычно достаточно использовать простую итерацию (5.89).

Наконец, заметим, что итерация (5.89) хорошо подходит для распределенных приложений. Самый простой метод состоит в следующем: каждый узел i направляет всем остальным узлам текущие значения суммарных потоков F_{ij}^k по своим уходящим линиям (i, j) , используя для этого, например, лавинный алгоритм или ТАКП из подразд. 5.3.3. Затем каждый узел вычисляет МППД-пути для тех ОА-пар, для которых он сам является узлом-отправителем, и выполняет итерацию (5.89) с некоторым фиксированным размером шага. Это соответствует варианту реализации «все ОА-пары одновременно». Такой метод можно также реализовать в асинхронном, распределенном варианте, в котором вычисления и прием информации не синхронизированы в каждом узле. Эффективность метода при этих условиях работы показана в работе [233].

Мы закончим этот раздел примером приложения алгоритма (5.89)—(5.91).

Пример

Рассмотрим сеть, показанную на рис. 5.58. Имеются только две ОА-пары (1,5) и (2,5) с соответствующими входными потоками $r_1 = 4$ и $r_2 = 8$. Рассмотрим следующие два пути для каждой ОА-пары.

Пути для ОА-пары (1, 5)

$$p_1(1) = \{(1, 4), (4, 5)\},$$

$$p_2(1) = \{(1, 3), (3, 4), (4, 5)\}.$$

Пути для ОА-пары (2, 5)

$$p_1(2) = \{(2, 4), (4, 5)\},$$

$$p_2(2) = \{(2, 3), (3, 4), (4, 5)\}.$$

Рассмотрим пример задачи маршрутизации, в которой стоимостные функции для всех линий одинаковы и имеют вид

$$D_{ij}(F_{ij}) = \frac{1}{2} (F_{ij})^2 \quad \text{для всех } (i, j).$$

Рассмотрим картину начальных путевых потоков, в которой весь поток каждой ОА-пары проходит по средней линии (3, 4). Потоки, получающиеся в результате, представлены в табл. 5.1 и 5.2.

Первопроизводная длины каждой линии равна

$$D'_{ij}(F_{ij}) = F_{ij},$$

поэтому суммарные потоки в линиях, представленные в табл. 5.2, являются также длинами линий для первой итерации. Соответствующие первопроизводные длины путей представлены в табл. 5.3.

Кратчайшими путями для первой итерации будут $p_1(1)$ и $p_1(2)$ для ОА-пар (1, 5) и (2, 5) соответственно.

Конкретный вид итерации (5.89)—(5.91) покажем для первой ОА-пары. В этом случае для некратчайшего пути $p = p_2(1)$ и кратчайшего пути $\bar{p} = p_1(1)$ имеем $d_{\bar{p}} = 28$ и $d_p = 12$. Также $H_p = 3$ (каждая линия имеет вторпроизводную длину $D''_{ij} = 1$ и существуют три линии, которые принадлежат либо $p_1(1)$,

Таблица 5.1. Начальные путевые потоки для примера маршрутизации

ОА-пара	Путь	Путевой поток
(1,5)	$p_1(1)$	0
	$p_2(1)$	4
(2,5)	$p_1(2)$	0
	$p_2(2)$	8

Таблица 5.2. Начальные суммарные потоки по линиям для примера маршрутизации

Линия	Суммарный поток в линии
(1,3)	4
(1,4)	0
(2,3)	8
(2,4)	0
(3,4)	12
(4,5)	12
Другие	0

Таблица 5.3. Начальные первопроизводные длины для примера маршрутизации

ОА-пара	Путь	Первопроизводная пути
(1,5)	$p_1(1)$	12
	$p_2(1)$	28
(2,5)	$p_1(2)$	12
	$p_2(2)$	32

либо $p_2(1)$, но не обоим сразу (см. (5.88) и (5.91)). Поэтому итерация (5.89) имеет вид

$$x_p := \max \{0,4 - \alpha^k (1/3) (28 - 12)\} = \\ = \max \{0,4 - 16\alpha^k/3\} = 4 - \min \{4, 16\alpha^k/3\}$$

и
$$x_{\bar{p}} := r_1 - x_p.$$

В более общей постановке пусть $x_1(1)$, $x_2(1)$, $x_1(2)$ и $x_2(2)$ означают потоки по путям $p_1(1)$, $p_2(1)$ и $p_1(2)$, $p_2(2)$ соответственно перед началом очередной итерации. Соответствующие длины путей равны

$$d_{p_1(1)} = x_1(1) + r_1 + r_2, \\ d_{p_2(1)} = 2x_2(1) + x_2(2) + r_1 + r_2, \\ d_{p_1(2)} = x_1(2) + r_1 + r_2, \\ d_{p_2(2)} = 2x_2(2) + x_2(1) + r_1 + r_2.$$

Второпроизводная длина H_p из (5.91) равна 3. Алгоритм (5.89)—(5.91) имеет следующий вид. Для $i = 1, 2$

$$x_1(i) := \begin{cases} x_1(i) - \min [x_1(i), (\alpha^k/3) [d_{p_1(i)} - d_{p_2(i)}]], & \text{если } d_{p_1(i)} > d_{p_2(i)}, \\ x_1(i) + \min [x_2(i), (\alpha^k/3) [d_{p_2(i)} - d_{p_1(i)}]] & \end{cases}$$

в противном случае, $x_2(i) := r_i - x_1(i)$.

Заметим, что наличие линии (4, 5) никак не сказывается на виде итерации и так это и должно быть потому, что суммарный поток по линии (4, 5) всегда равен $r_1 + r_2$ независимо от маршрутизации.

В табл. 5.4 представлены последовательные значения стоимостной функции, получаемые алгоритмом для различных значений размера шага и вариантов реализации «все ОА-пары одновременно» и «одна ОА-пара за один раз». Разница между этими двумя вариантами реализации состоит в том, что в варианте «все одновременно» ОА-пары, обрабатываемые одновременно в течение итерации, используются значения потоков по линиям и путям, полученные в конце предыдущей итерации. В варианте «одна за один раз» ОА-пары обрабатываются последовательно и после каждой итерации, выполненной для одной ОА-пары, значения потоков по линиям пересчитываются с учетом результатов этой итерации и до начала выполнения следующей итерации для другой ОА-пары. Размер шага α^k выбран равным константе и принимает три возможных значения (0,5, 1 и 1,8).

Таблица 5.4. Последовательности стоимостей, генерируемые проекционным градиентным методом, для рассматриваемого примера маршрутизации при различных размерах шага

Итерация	Вариант «все одновременно»			Вариант «одна за один раз»		
	$\alpha^k \equiv 0,5$	$\alpha^k \equiv 1,0$	$\alpha^k \equiv 1,8$	$\alpha^k \equiv 0,5$	$\alpha^k \equiv 1,0$	$\alpha^k \equiv 1,8$
0	184,00	184,00	184,00	184,00	184,00	184,00
1	110,88	104,00	112,00	114,44	101,33	112,00
2	102,39	101,33	118,72	103,54	101,00	109,15
3	101,28	101,03	112,00	101,63		101,79
4	101,09	101,00	118,72	101,29		101,56
5	101,03		112,00	101,08		101,33
6	101,01		118,72	101,03		101,18
7	101,00		112,00	101,01		101,12
8			118,72	101,00		101,09
9			112,00			101,06
10			118,72			101,03

Видно, что при единичном размере шага сходимость к окрестности решения очень быстрая для обоих вариантов «все одновременно» и «одна за один раз». С ростом размера шага появляется опасность утраты свойства сходимости и обычно прежде всего перестает сходиться алгоритм в варианте «все одновременно». Это видно из таблицы, где при $\alpha^k = 1,8$ алгоритм сходится (медленно) в варианте «одна за один раз», но не сходится в варианте «все одновременно».

5.8. Маршрутизация в сети Codex

В этом разделе будет рассмотрена система маршрутизации сети, выпущенной в продажу фирмой Codex. Более подробно эта система описана в работах [118] и [124]. В сети Codex используются виртуальные цепи для трафика пользователя и дейтаграммы для системного трафика (учет, управление, маршрутная информация и т. д.). Было решено использовать дейтаграммы для системного трафика потому, что альтернативный метод, а именно установление виртуальной цепи для каждой пары узлов, представляется слишком расточительным с точки зрения ресурсов сети. Заметим, что для каждого сеанса пользователей устанавливаются две различные виртуальные цепи, по которым проходит трафик в разных направлениях, и не обязательно, чтобы они проходили по одному и тому же множеству линий. Маршрут выбирается отдельно для каждого направления.

Имеются два алгоритма выбора маршрута. Первый алгоритм, используемый для маршрутизации дейтаграмм внутреннего системного трафика, есть не что иное, как алгоритм кратчайшего пути, о котором шла речь в разд. 5.2. Кратчайшие пути вычисляются в ходе выполнения второго алгоритма, который используется при выборе маршрутов для новых виртуальных цепей и

изменении маршрутов для старых. Мы сосредоточим внимание на втором алгоритме, который намного важнее и гораздо сложнее первого.

Алгоритм маршрутизации виртуальных цепей имеет много общего с проекционным градиентным методом оптимальной маршрутизации, описанным в предыдущем разделе. Стоимостная функция D_{ij} для каждой линии (i, j) зависит от потока, проходящего по линии, а также от других параметров, таких как пропускная способность линии, задержка на обработку и распространение и значение приоритета проходящих в настоящее время по линии виртуальных цепей. Каждый узел следит за параметрами своих смежных линий и периодически распространяет значения этих параметров среди всех остальных узлов. Если все виртуальные цепи имеют одинаковый приоритет, то стоимостная функция для линии имеет вид

$$D_{ij}(F_{ij}) = \frac{F_{ij}}{C_{ij} - F_{ij}} + d_{ij}F_{ij},$$

где F_{ij} — суммарная скорость передачи данных по линии, d_{ij} — задержка на обработку и распространение, а C_{ij} — пропускная способность линии. Эта формула основана на аппроксимации задержки временем ожидания в системе $M/M/1$ (см. обсуждение в разд. 3.6 и 5.4). При наличии более одного, скажем M , уровней приоритетов для виртуальных цепей стоимостная функция для линии имеет вид

$$D_{ij}(F_{ij}^1, \dots, F_{ij}^M) = \left(\sum_{k=1}^M p_k F_{ij}^k \right) \left(\frac{1}{C_{ij} - \sum_{k=1}^M F_{ij}^k} + d_{ij} \right),$$

где F_{ij}^k — суммарный поток в линии для виртуальных цепей приоритета k , а p_k — некоторый положительный весовой множитель. Вид этого выражения мотивирован предыдущей формулой; кроме этого, нет никакого другого разумного обоснования, вытекающего из теории очередей.

В качестве маршрута для новой виртуальной цепи выбирается тот путь, который является кратчайшим для данной ОА-пары. Этот путь вычисляется в узле-адресате данной виртуальной цепи на основе топологии сети и последних полученных значений потоков и других параметров для каждой линии сети. При выборе формулы для вычисления длин линий учитывается, что интенсивность потока в виртуальной цепи может быть сравнима с суммарной интенсивностью потока на линии, по которой проходит данная виртуальная цепь. Эта используемая длина равна разнице между значениями стоимостной функции, когда данная виртуальная цепь проходит по этой линии и когда нет. Таким образом,

если предполагаемая интенсивность потока в новой виртуальной цепи равна Δ и у нее k -й приоритетный класс, то длина линии (i, j) равна

$$D_{ij}(F_{ij}^1, \dots, F_{ij}^k + \Delta, \dots, F_{ij}^M) - D_{ij}(F_{ij}^1, \dots, F_{ij}^M). \quad (5.92)$$

В результате выбор в качестве маршрута для новой виртуальной цепи кратчайшего пути по отношению к длине линий (5.92) приведет к минимальному увеличению суммарной стоимости. Если все виртуальные цепи имеют одинаковый приоритет, то длина линии (5.92) будет равна

$$D_{ij}(F_{ij} + \Delta) - D_{ij}(F_{ij}). \quad (5.93)$$

Если Δ очень мало, то длины линий (5.93) примерно равны $\Delta F'_{ij}(F_{ij})$, в любом случае (согласно теореме о среднем) они пропорциональны соответствующим производным от D_{ij} , взятым в некоторой промежуточной точке между текущим значением потока и значением потока, которое получится, если маршрут виртуальной цепи пройдет через (i, j) . Таким образом, длины линий (5.93) можно считать оценками первых производных от стоимостных функций для линий, которые используются в проекционном градиентном методе. Длины линий (5.92) также допускают аналогичную интерпретацию (см. задачу 5.27).

Изменение маршрутов для старых виртуальных цепей происходит аналогичным образом. Если какая-либо линия выходит из строя, то для виртуальных цепей, проходивших по этой линии, выбирается другой маршрут так, как если бы они были новыми. Выбор других маршрутов с намерением ослабить перегрузки осуществляется всеми узлами по ходу поступления новой информации о потоках на линиях. Каждый узел постоянно следит за теми виртуальными цепями, которые оканчиваются в этом узле. Он выбирает одну из этих цепей в качестве кандидата на смену маршрута и вычисляет длину каждой линии, равную разнице между значениями стоимостной функции, когда данная виртуальная цепь проходит по этой линии и когда нет. Затем данная виртуальная цепь направляется по кратчайшему пути, если до этого маршрут этой цепи не был кратчайшим. Важным параметром при этом является число виртуальных цепей, выбираемых в качестве кандидатов на смену маршрутов, в период между последовательными моментами поступлений сообщений о потоках на линиях. Это число соответствует размеру шага в проекционном градиентном методе, определяющему величину потока, перебрасываемого на кратчайший путь на каждой итерации. Конечно, возникает проблема, связанная с тем, что одновременная смена маршрутов у слишком большого числа виртуальных цепей в нескольких узлах, действующих не координированно, может привести к колебательному режиму, аналогичному тому, о котором речь шла

в подразд. 5.2.5. В сети Codex используется эвристическое правило, согласно которому только часть существующих виртуальных цепей (псевдослучайно) выбирается в качестве кандидатов на смену маршрутов в период между последовательными моментами поступлений сообщений о потоках на линиях.

5.9. Заключение

Маршрутизация является одной из сложнейших функций сети передачи данных, требующей согласованной работы узлов сети с помощью распределенных протоколов. Она влияет на среднюю задержку пакета и пропускную способность сети.

Наше главное внимание было сосредоточено на методах выбора маршрутов. Самый распространенный подход — маршрутизацию по кратчайшим путям — можно реализовать различными способами, как было показано на примерах алгоритмов ARPANET и TUMNET. В зависимости от способа реализации маршрутизация по кратчайшим путям может привести к низкой пропускной способности, слабой чувствительности к перегрузкам и к колебательному режиму. Эти недостатки более заметны в дейтаграммных сетях и менее заметны в сетях с виртуальными цепями. Более сложным способом является оптимальная маршрутизация, основанная на потоковых моделях. Было представлено несколько алгоритмов вычисления оптимальных маршрутов, как централизованных, так и распределенных. По мере того, как параметры входных потоков начинают быстро меняться во времени, преимущества рассмотренных методов оптимальной маршрутизации начинают исчезать. В таких случаях трудно рекомендовать какие-либо методы маршрутизации, которые были бы одновременно эффективными и практичными.

Другой интересный аспект задачи маршрутизации относится к распространению используемой при маршрутизации информации по линиям, которые могут выходить из строя. Описано несколько различных алгоритмов, основанных на идеях лавинного алгоритма.

Наконец, маршрутизацию необходимо учитывать при выборе сетевой топологии, так как метод маршрутизации определяет степень эффективности использования пропускных способностей линий. Были описаны точные и эвристические методы решения задач, возникающих при выборе топологии.

5.10. Замечания, источники и дополнительная литература

Раздел 5.1. Обзоры методов маршрутизации, включающие описания некоторых алгоритмов маршрутизации, взятых из практики, можно найти в работах [61] и [212]. Маршрутизация в сети

ARPANET описана в [171, 180, 181]. Система TYMNET рассмотрена в [10]. Маршрутизация в SNA описана в [3] и [10]. Метод выбора маршрута в сетях SNA изложен в [85]. Описание других систем маршрутизации можно найти в [219, 252].

Раздел 5.2. Имеется обширная литература по графам, остовым деревьям и кратчайшим путям, например [159, 188]. В частности, быстрые алгоритмы отыскания кратчайших путей описаны в [56, 187].

Материал по асинхронным алгоритмам кратчайшего пути взят из [22]. Для дальнейшего изучения общих асинхронных распределенных алгоритмов можно пользоваться работами [26, 38].

Более подробно вопросы устойчивости алгоритмов маршрутизации по кратчайшим путям рассмотрены в [19, 20, 23, 79].

Раздел 5.3. Трудности, возникающие при использовании лавинного алгоритма ARPANET, и некоторые методы их устранения, описаны в [189, 206]. К теме ТАКП относятся работы [59, 72, 78, 214, 218].

Раздел 5.4. Имеется обширная литература по маршрутизации на основе информации о состоянии очередей. В этих работах рассматриваются сети передачи данных, но они также относятся к задачам маршрутизации и управления в других системах. См. [65, 75, 111, 177, 205, 208, 210, 211, 224, 257].

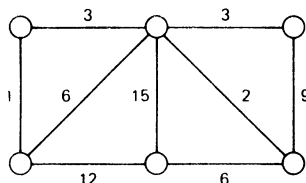
Обзоры по топологии сетей передачи данных представлены в [35, 91, 133, 170, 178]. Дальнейший материал по надежности сети содержится в [11, 12, 17, 156, 163, 193] и в источниках, на которые даются ссылки.

Раздел 5.5. Рассмотренная задача оптимальной маршрутизации является частным случаем задачи о многопродуктовом потоке в сети. Эта задача также возникает при рассмотрении транспортных сетей в постановке, очень похожей на обсуждавшуюся здесь. См. [1, 52, 53, 55, 73, 154].

Раздел 5.6. Метод Франка—Волфа [76] был предложен для решения задачи маршрутизации в [70]. В задачах 5.31 и 5.32 содержится описание свойств сходимости как этого метода, так и проекционного градиентного метода.

Раздел 5.7. Проекционный градиентный метод был применен для решения задач о многопродуктовом потоке в постановке, использующей путевые потоки, в [21]. Дальнейшее развитие метода имеется в [15, 16, 80]. Реализация проекционного градиентного метода на языке программирования FORTRAN представлена в [31]. Алгоритм оптимальной маршрутизации, описанный в [87], который регулирует доли потоков по линиям (см. конец разд. 5.5), находится в тесной связи с проекционным гра-

Рис. 5.59.



диентным методом (см. [29]). Дальнейшие развития алгоритма представлены в [19, 29, 82]. Результаты моделирования и вычислений, относящиеся к этому алгоритму и его разновидностям, можно найти в [32] и [229].

Раздел 5.8. Замечания П. Умбле, одного из ведущих создателей сети Codex, были очень полезны при подготовке этого раздела. Дополнительный материал можно найти в [124] и [118].

Задачи

- 5.1. Найдите остовое дерево минимального веса графа, изображенного на рис. 5.59, используя алгоритмы Прим—Дijkstra и Крускала.
- 5.2. Найдите дерево кратчайших путей от узла 1 до всех остальных узлов графа, изображенного на рис. 5.60, используя алгоритмы Беллмана—Форда и Дijkstra.
- 5.3. На рис. 5.61 числа, приведенные рядом с каждой линией сети, равны вероятности того, что данная линия выйдет из строя в течение времени жизни виртуальной цепи от узла A до узла B . Предполагается, что линии выходят из строя независимо друг от друга. Найдите наиболее надежный путь от A до B , т. е. путь, для которого вероятность того, что все его линии останутся неповрежденными в течение времени жизни виртуальной цепи, окажется максимальной. Чему равна эта вероятность?
- 5.4. Остовное дерево кратчайших путей (подразд. 5.2.3) отличается от остовного дерева минимального веса тем, что в первой задаче длины дуг приписываются ориентированным дугам, а во второй задаче веса дуг приписываются неориентированным дугам или, что одно и то же, веса дуг предполагаются одинаковыми для обоих направлений. Однако, даже если все дуги имеют одинаковые длины для обоих направлений, остовное дерево минимального веса (с весами дуг, равными соответствующим длинам) не обязательно будет

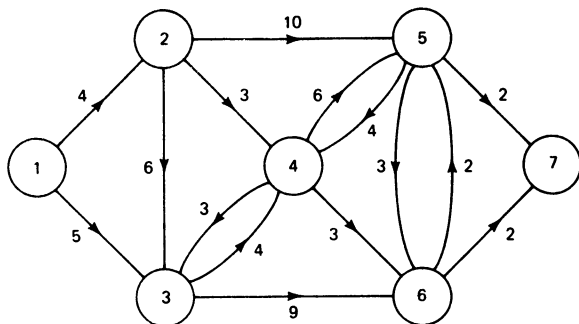


Рис. 5.60.

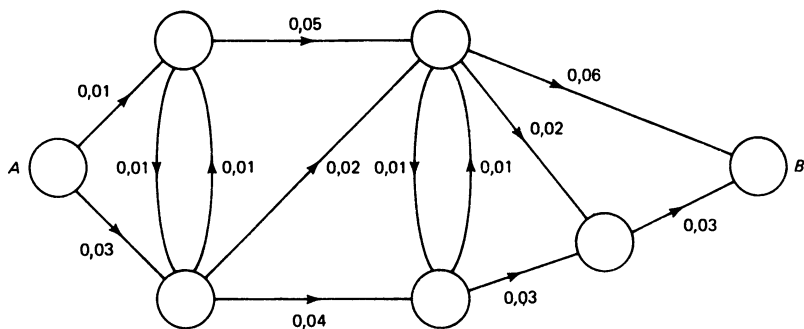


Рис. 5.61.

остовным деревом кратчайших путей. Чтобы убедиться в этом, рассмотрите сеть с тремя узлами A , B и C и тремя неориентированными дугами AB , BC , CA . Выберите вес для каждой дуги так, чтобы остовное дерево минимального веса не совпадало с деревом кратчайших путей с корневым узлом C (длина дуги в каждом из двух ее направлений равна весу дуги).

5.5. Рассмотрите пример 1 из подразд. 5.2.5.

- Повторите этот пример с $N = 6$, а не с $N = 16$. (Узлы 1, 2, 4 и 5 посылают 1 единицу к узлу 6, а узел 3 посылает ε , $0 < \varepsilon \leq 1$.)
- Повторите (а) с той разницей, что длина d_{ij} линии (i, j) равна $\alpha + F_{ij}$ (а не \bar{F}_{ij}) для $\alpha = 1$. Рассмотрите все возможные варианты начальной маршрутизации.
- Каким должно быть минимальное значение α , при котором кратчайшие пути для всех узлов, за исключением узла 3, со временем останутся неизменными независимо от выбора начальной маршрутизации?
- Повторите (а) с той разницей, что при каждой итерации, кроме первой, длина каждой линии равна среднему арифметическому интенсивностей проходящего по линии трафика, соответствующих последней и предыдущей маршрутизациям (а не интенсивностям проходящего трафика, соответствующим только последней маршрутизации).

5.6. Покажите, что (даже если существуют циклы отрицательной длины) значение $D_i^{(h)}$, порождаемое алгоритмом Беллмана—Форда, равно длине кратчайшего ($\leq h$) перехода от узла 1 до узла i при ограничении, что узел 1 не повторяется в этом переходе.

5.7. Рассмотрите алгоритм Беллмана—Форда, предполагая, что отсутствуют циклы отрицательной длины. Пусть (j_i, i) для каждого узла $i \neq 1$ — это такая дуга, что при f_i достигается минимум в уравнении

$$D_i^{(h)} = \min_j \left[D_j^{(h-1)} + d_{ji} \right],$$

где h_i равно максимальному h , такому, что $D_i^{(h)} \neq D_i^{(h-1)}$. Рассмотрите подграф, состоящий из дуг (j_i, i) , для $i \neq 1$ и покажите, что (даже если существуют циклы нулевой длины) он является остовным деревом, состоящим из кратчайших путей. *Указание:* покажите, что $h_i > h_{j_i}$.

5.8. Рассмотрите задачу нахождения кратчайшего пути, предполагая, что отсутствуют циклы отрицательной длины. Покажите, что если есть циклы нулевой длины, не содержащие узел 1, то уравнение Беллмана имеет более одного решения.

- 5.9.** Предположим, что имеется ориентированный граф, не содержащий ориентированных циклов. Заданы длины d_{ij} для каждой ориентированной дуги (i, j) и нужно вычислить кратчайшие пути от узла 1 до всех остальных узлов. Пусть существует по крайней мере один ориентированный путь от узла 1 до каждого другого узла. Покажите, что узлы 2, 3, ..., N можно перенумеровать таким образом, что дуга из i в j может существовать только тогда, когда $i < j$. Покажите, что если узлы перенумерованы таким образом, то уравнение Беллмана—Форда можно решить в худшем случае за $O(N^2)$ операций.
- 5.10.** Рассмотрите задачу нахождения кратчайшего пути от узла 1 до каждого другого узла. Предположим, что известна положительная нижняя граница для всех длин дуг. Видоизмените алгоритм Дijkstra таким образом, чтобы можно было более одного узла добавлять к множеству окончательно помеченных узлов на каждой итерации. Покажите, что этот модифицированный алгоритм будет работать корректно.
- 5.11.** Рассмотрите сеть, в которой в каждом узле известны топология сети и положительная длина каждой ориентированной дуги. В каждом узле уже вычислены кратчайшие пути от него до всех остальных узлов сети (предполагается, что граф является сильно связным). Предположим, что длина (одной) дуги (i, j) увеличивается. Покажите, каким образом следует изменить алгоритм Дijkstra для данного узла-отправителя, чтобы пересчет кратчайших путей от него до всех остальных узлов был как можно более эффективным.
- 5.12.** Обнаружение момента окончания работы распределенного алгоритма Беллмана—Форда [46]. Целью следующего алгоритма (основанного на методе Беллмана—Форда) является вычисление распределенным образом кратчайших путей от единственного узла-отправителя (узел 1) до всех остальных узлов и обнаружение в узле 1 момента, когда вычисления закончились. Предположим, что все линии (i, j) являются неориентированными, имеют неотрицательную длину d_{ij} , сохраняют очередность передающихся по ним сообщений и работают без ошибок или отказов. В каждом узле i имеется оценка D_i кратчайшего расстояния от узла 1 до него. В начале $D_i = \infty$ для всех узлов $i \neq 1$ и $D_1 = 0$. В узле 1 начинаются вычисления путем рассылки оценки $D_1 + d_{1j}$ всем соседним узлам j . Правила алгоритма для каждого узла $j \neq 1$ состоят в следующем.
- (1) Когда узел j получает оценку $D_i + d_{ij}$ от какого-либо другого узла i , то спустя некоторое, заранее не предписываемое конечное время (но до того, как будет обрабатываться следующая оценка), он делает следующее.
Если $D_j \leq D_i + d_{ij}$, то узел j посылает подтверждение АСК узлу i . Если $D_j > D_i + d_{ij}$, то узел j обновляет $D_j = D_i + d_{ij}$, объявляет узел i своим лучшим предшественником на кратчайшем пути, отсылает подтверждение АСК своему предыдущему лучшему предшественнику (если такой был) и рассылает оценку $D_j + d_{jk}$ каждому своему соседу k .
 - (2) Узел j посылает подтверждение АСК своему лучшему в настоящий момент предшественнику, как только он получает подтверждение на каждую из последних оценок, разосланных соседям.
Предполагается, что каждое подтверждение однозначно соответствует какой-либо разосланной ранее оценке и что узел 1 отвечает посредством АСК на любую полученную им оценку длины.
 - (а) Покажите, что узел 1 в конце концов получит АСК от каждого из своих соседей и в этот момент D_i будет истинным кратчайшим расстоянием для каждого узла i .
 - (б) Каковы преимущества и недостатки этого алгоритма по сравнению с распределенным асинхронным алгоритмом Беллмана—Форда из подразд. 5.2.4?

- 5.13. Рассмотрите второй лавинный алгоритм из подразд. 5.3.2. Предположим, что известна верхняя граница для времени, необходимого для того, чтобы обновляющее сообщение достигло каждого узла, связанного с узлом-отправителем. Придумайте схему, основанную на поле возраста, которую нужно добавить к алгоритму для того, чтобы он работал правильно, даже если поле порядковых номеров переполнится вследствие ошибок при передаче или при хранении в памяти. *Замечание:* пакеты, содержащие поле возраста, следует использовать лишь в исключительных ситуациях после того, как узел-отправитель обнаружит ошибку.
- 5.14. Видоизмените ТАКП таким образом, чтобы его можно было использовать для распространения по сети не только статусов, но и другой информации о линиях, зависящей от ориентации линий (см. замечание в конце подразд. 5.3.3). Информация о линии собирается узлом, от которого уходит данная линия. Проверьте работоспособность видоизмененного алгоритма. *Указание:* в существующем алгоритме добавьте основную и портовые таблицы для хранения информации об ориентированных линиях. Сформулируйте правила обновления для этих таблиц, используя надписи для узлов, предусмотренные алгоритмом обновления основной топологической таблицы из подразд. 5.3.3.
- 5.15. (а) Приведите пример, когда следующий алгоритм распространения информации о топологических изменениях приводит к ошибке. Вначале все узлы знают истинные статусы всех линий.
Правила обновления.
(1) При изменении статуса какой-либо смежной линии узел сразу же рассылает сообщение о новом статусе по всем действующим смежным линиям.
(2) Когда узел получает сообщение о несмежной линии, отличающееся от его собственного представления о статусе этой линии, он сразу же меняет статус этой линии в своей топологической таблице. Одновременно с этим он рассылает сообщение о новом статусе по всем действующим смежным линиям, *исключая* ту, по которой это сообщение пришло.
(3) Когда узел получает сообщение о смежной ему линии, отличающееся от его собственного представления о статусе этой линии, он сразу же посылает обратно истинный статус по той линии, по которой это сообщение пришло.
Указание: алгоритм может привести к ошибке даже в сетях с простой топологией, такой, как показано на рис. 5.29.
(б) Приведите пример, когда этот алгоритм приводит к ошибке, если в правиле 2 слово «исключая» заменить на «включая».
- 5.16. Опишите, какие неприятности могут произойти (если такие будут) в алгоритмах распространения топологии из разд. 5.3, если отбросить предположение о том, что сообщения, передаваемые по линии, принимаются в том же порядке, что и передаются. Рассмотрите лавинный алгоритм ARPANET, оба алгоритма из подразд. 5.3.2 и ТАКП из подразд. 5.3.3.
- 5.17. *Распределенный алгоритм вычисления числа узлов в сети.* Рассмотрите сильно связную сеть связи с N узлами и A линиями, по которым передача может вестись в обоих направлениях. Каждый узел знает себя и множество своих непосредственных соседей, но не знает топологию всей сети. Узел 1 кочет определить число узлов в сети. В качестве первого шага он запускает в ход алгоритм нахождения *ориентированного корневого дерева, корнем которого является узел 1*. Под этим мы имеем в виду дерево, каждая линия (i, j) которого ориентирована и направлена в сторону узла 1 вдоль единственного пути на дереве, приводящего из i в 1 (см. пример такого дерева на рис. 5.62).
- (а) Разработайте распределенный алгоритм, включающий обмен сообщениями между узлами, который бы создавал такое дерево. Этот алгоритм

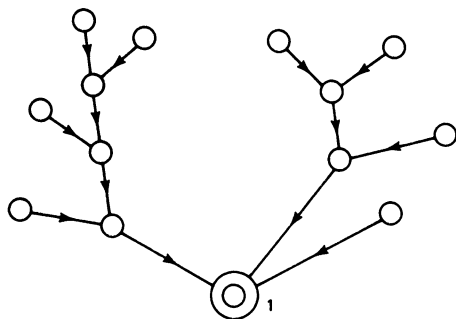


Рис. 5.62.

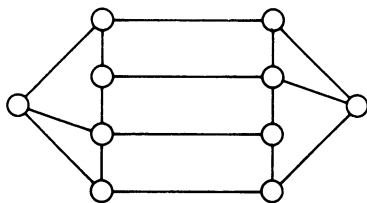


Рис. 5.63.

должен запускаться в ход узлом 1 и содержать не более $O(A)$ передач сообщений. Предполагается, что при передаче по любой линии ошибок не возникает. В конце алгоритма концевые узлы каждой линии должны знать, является ли данная линия частью дерева и если да, то они должны знать ее направление.

(б) Дополните алгоритм, разработанный в (а), другим алгоритмом, включающим не более $O(N)$ передач сообщений, благодаря которому узел 1 сможет узнать N .

(в) Предполагая, что время передачи всех сообщений одинаково и равно T , оцените сверху время, необходимое для выполнения алгоритмов из (а) и (б).

5.18. Рассмотрите задачу построения остова минимального веса. Покажите, что следующая процедура реализует алгоритм Прим—Дijkstra и требует $O(N^2)$ арифметических операций. Пусть алгоритм начинает построение с узла 1 и вначале полагает $P = \{1\}$, $T = \{\}$, $D_1 = 0$ и $D_j = w_{1j}$, $a_j = 1$ для $j \neq 1$.

Шаг 1: Найти $i \notin P$, такое, что

$$D_i = \min_{j \notin P} D_j,$$

и положить $T := T \cup \{(j, a_j)\}$, $P := P \cup \{i\}$. Если P содержит все узлы, то тогда стоп.

Шаг 2: Для всех $j \notin P$, если $w_{ij} < D_j$, положить $D_j := w_{ij}$, $a_j := i$. Перейти к шагу 1.

Замечание: обратите внимание на аналогию с алгоритмом построения кратчайших путей Дijkstra.

5.19. Используя алгоритм Клейтмана, докажите или опровергните, что сеть, показанная на рис. 5.63, является 3-связной? Чему равно максимальное k , для которого эта сеть является k -связной?

5.20. Пусть имеется алгоритм, способный находить максимальное k , для которого два узла в графе являются k -связными (например, тест, основанный на алгоритме вычисления максимального потока, показанный на рис. 5.41). Как следует видоизменить алгоритм Клейтмана, чтобы можно было находить максимальное k , для которого граф является k -связным? Примените видоизмененный алгоритм к графу из задачи 5.19 (рис. 5.63).

5.21. Используя эвристический алгоритм Эссау—Вильямса, найдите МОД с ограничением для сети, показанной на рис. 5.64. Узел 0 является центральным узлом. Веса линий указаны рядом с линиями. Входные потоки от каждого узла к концентратору указаны рядом со стрелками. Пропускные способности всех линий равны 10.

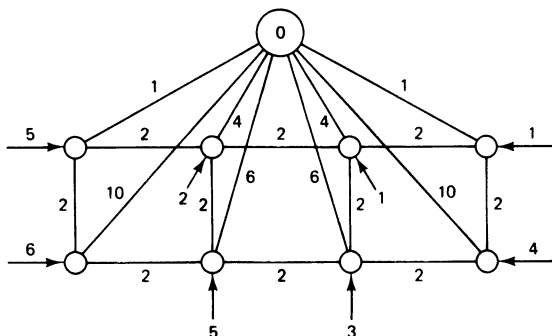


Рис. 5.64.

5.22. Рассмотрите показанную на рис. 5.65 сеть, состоящую из одной ОА-пары с известной интенсивностью входного потока r и трех линий с пропускными способностями C_1 , C_2 и C_3 соответственно. Предположив $C_1 = C_2 = C$ и $C_3 > C$, решите задачу оптимальной маршрутизации со стоимостной функцией, основанной на аппроксимации задержки моделью $M/M/1$, и r , принимающим значения между 0 и $2C + C_3$.

5.23. Рассмотрите задачу оптимальной маршрутизации для сети, состоящей из одной ОА-пары с единичной интенсивностью входного потока и трех линий, для стоимостной функции вида

$$D(x) = \frac{1}{2} [(x_1^2) + 2(x_2^2) + (x_3^2)] + 0,7x_3.$$

Математически задача состоит в том, чтобы минимизировать $D(x)$

при ограничениях $x_1 + x_2 + x_3 = 1$, $x_1, x_2, x_3 \geq 0$.

(а) Покажите, что $x_1^* = 2/3$, $x_2^* = 1/3$ и $x_3^* = 0$ является оптимальным решением.

(б) Выполните несколько итераций метода Франка—Волфа и проекционного метода, начиная из точки $1/3, 1/3, 1/3$. Проведите достаточное число итераций, для того чтобы отчетливо была видна тенденция в скорости сходимости. (При желании для этого можно использовать компьютер.) Нанесите последовательные итерации на симплекс допустимых потоков.

5.24. Пусть g является выпуклой дифференцируемой функцией одной переменной α . Предположим, что $dg(0)/(d\alpha) \geq 0$. Покажите, что $g(\alpha) \geq g(0)$ для всех $\alpha \geq 0$. Указание: покажите, что для любых двух скалярных величин α_1 и α_2 имеем

$$g(\alpha_2) \geq g(\alpha_1) + \frac{dg(\alpha_1)}{d\alpha} (\alpha_2 - \alpha_1),$$

используя для этого неравенство

$$(1 - \mu) g(\alpha_1) + \mu g(\alpha_2) \geq g[\alpha_1 + \mu(\alpha_2 - \alpha_1)], \quad \mu \in [0, 1].$$

5.25. Рассмотрите задачу оптимальной маршрутизации из разд. 5.5. Предположим, что каждая стоимостная функция для линии выбрана равной одной и той же выпуклой функции $D(F)$ потока на линии F , причем первая производная от стоимости на линии при нулевом потоке $D'(0)$ положительна.

(а) Покажите, что для достаточно малых величин r_w -интенсивностей входных потоков пар отправитель — адресат пути от отправителей к адре-

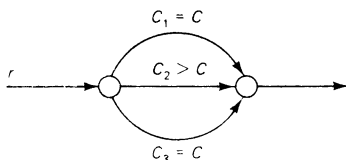


Рис. 5.65.

сатам, используемые при оптимальной маршрутизации, состоят из минимального числа линий.

- (б) Приведите пример, показывающий, что при больших значениях r_w пути, используемые при оптимальной маршрутизации, не обязательно состоят из минимального числа линий.

5.26. Рассмотрите метод Франка—Волфа с размером шага (5.69) из разд. 5.6. Опишите реализацию в виде распределенного синхронного алгоритма, включающую передачу от источников к линиям и обратно. На каждой итерации этого алгоритма каждый узел-отправитель (источник) должен вычислять размер шага (5.69) и в соответствии с ним обновлять путевые потоки, возникающие в нем.

5.27. Обобщения проекционного градиентного метода.

- (а) Покажите, каким образом условие оптимальности из разд. 5.5 и проекционный градиентный метод можно обобщить на случай, в котором стоимостная функция является произвольной выпуклой, дважды дифференцируемой функцией вектора путевых потоков x .

- (б) Рассмотрите частный случай, в котором стоимостной функцией для линии (i, j) является $D_{ij}(\tilde{F}_{ij}, F_{ij})$, где

$$\tilde{F}_{ij} = \sum_{k=1}^M p_k F_{ij}^k, \quad F_{ij} = \sum_{k=1}^M F_{ij}^k,$$

p_k — некоторый положительный скалярный взвешивающий множитель для приоритетного класса k , а F_{ij}^k — суммарный поток приоритетного класса k , проходящий по линии (i, j) (см. стоимостную функцию алгоритма Codex).

5.28. Оптимальное распространение информации по остовам деревьев. Рассмотрим задачу оптимальной маршрутизации из разд. 5.4. Предположим, что помимо обычных входных потоков ОА-пар r_w , для которых необходимо выбрать маршруты по ориентированным путям, имеется дополнительный трафик R от одного узла, скажем 1, который необходимо распространить среди всех остальных узлов по одному или нескольким ориентированным остовам деревьев, имеющим в качестве корня узел 1. Остовные деревья, которые можно использовать при маршрутизации, должны выбираться из имеющегося набора T , а часть потока R , которая должна передаваться по каждому дереву, является предметом оптимизации.

- (а) Охарактеризуйте интенсивности путевых потоков и части потока R , передаваемые по остовам деревьям из T и минимизирующие сумму стоимостей для линий, подобно тому как было представлено неравенством (5.59) из разд. 5.5.
- (б) Обобщите проекционный градиентный метод для решения этой задачи.
- (в) Обобщите ваш анализ на случай, когда есть несколько корневых узлов и имеется трафик, который нужно распространить от каждого из этих узлов среди всех остальных узлов.

Указание: помимо первопроизводных длин путей рассмотрите первопроизводные суммарные веса остовам деревьев.

- 5.29. *Оптимальная маршрутизация с виртуальными цепями, использующими одни и те же линии в обоих направлениях.* Рассмотрите задачу оптимальной маршрутизации из разд. 5.4 с дополнительным ограничением, состоящим в том, что трафик, возникающий в узле i и адресованный узлу i' , должен использовать тот же путь (в обратном направлении), что и трафик, возникающий в i' и адресованный i . В частности, предполагается, что если w и w' являются ОА-парами (i, i') и (i', i) соответственно, то существует такая константа пропорциональности $c_{ww'}$, что если $p \in P_w$ — некоторый путь и $p' \in P_{w'}$ — обратный ему путь, а x_p и $x_{p'}$ — соответствующие путевые потоки, то $x_p = c_{ww'} x_{p'}$. Сформулируйте условия, характеризующие оптимальную маршрутизацию, и подходящим образом видоизмените проекционный градиентный метод.
- 5.30. *Использование стоимостной функции, основанной на $M/M/1$, вместе с проекционным градиентным методом.* При использовании проекционного градиентного метода для минимизации стоимостной функции вида

$$\sum_{(i, j)} D_{ij}(F_{ij}) = \sum_{(i, j)} \frac{F_{ij}}{C_{ij} - F_{ij}}$$

существуют потенциальные трудности, связанные с тем, что некоторый поток по линии F_{ij} может превысить пропускную способность C_{ij} . Одним из способов избежать этого является замена D_{ij} на функцию \tilde{D}_{ij} , которая идентична D_{ij} для F_{ij} из интервала $[0, \rho C_{ij}]$, где ρ — некоторое число, меньшее единицы (скажем, $\rho = 0,99$), а для $F_{ij} > \rho C_{ij}$ является квадратичной. Квадратичная функция выбирается таким образом, чтобы значения D_{ij} и \tilde{D}_{ij} , а также их первых двух производных, совпадали в точке $F_{ij} = \rho C_{ij}$. Выведите формулу для \tilde{D}_{ij} . Покажите, что если потоки на линиях F_{ij}^* , которые минимизируют

$$\sum_{(i, j)} D_{ij}(F_{ij}),$$

приводят в результате к коэффициенту использования линий, не превышающему ρ (т. е. $F_{ij}^* \leq \rho C_{ij}$ для всех (i, j)), то F_{ij}^* также минимизирует

$$\sum_{(i, j)} \tilde{D}_{ij}(F_{ij}).$$

- 5.31. *Сходимость алгоритма Франка—Волфа.* Цель этой и следующей задачи состоит в том, чтобы помочь читателю разобраться в доказательствах сходимости метода Франка—Волфа и проекционного градиентного метода. Рассмотрите задачу

минимизировать $f(x)$
при ограничении $x \in X$,

где f — некоторая дифференцируемая функция n -мерного вектора x , а X — некоторое замкнутое, ограниченное, выпуклое множество. Предположите, что градиент f удовлетворяет неравенству

$$|\nabla f(x) - \nabla f(y)| \leq L \|x - y\| \quad \text{для всех } x, y \in X,$$

где L — некоторая положительная константа, а $\|z\|$ означает евклидову норму вектора z , т. е.

$$\|z\| = \sqrt{\sum_{i=1}^n (z_i)^2}.$$

(Можно показать, что это предположение будет выполняться, если существует $\nabla^2 f$ и он непрерывен в X .)

- (а) Покажите, что для всех векторов x , Δx и скалярных величин α , таких, что $x \in X$, $x + \Delta x \in X$, $\alpha \in [0, 1]$,

$$f(x + \alpha \Delta x) \leq f(x) + \alpha \nabla f(x)^T \Delta x + \frac{\alpha^2 L}{2} |\Delta x|^2,$$

где верхний индекс T означает транспонирование.

Указание: воспользуйтесь формулой Тейлора

$$f(x + g) = f(x) + \nabla f(x)^T g + \int_0^1 [\nabla f(x + tg) - \nabla f(x)]^T g dt.$$

- (б) Используя часть (а), покажите, что если Δx является направлением спуска из x (т. е. $\nabla f(x)^T \Delta x < 0$), то

$$\min_{\alpha \in [0, 1]} f(x + \alpha \Delta x) \leq f(x) + \delta, \quad \text{где}$$

$$\delta = \begin{cases} \frac{1}{2} \nabla f(x)^T \Delta x, & \text{если } \nabla f(x)^T \Delta x + L |\Delta x|^2 \leq 0, \\ -\frac{|\nabla f(x)^T \Delta x|^2}{2LR^2} & \text{в противном случае,} \end{cases}$$

где R — диаметр X , т. е.

$$R = \min_{x, y \in X} |x - y|.$$

Указание: проведите минимизацию по $\alpha \in [0, 1]$ в обеих частях неравенства из (а).

- (в) Рассмотрите метод Франка—Волфа

$$x^{k+1} = x^k + \alpha^k \Delta x^k,$$

где $x^0 \in X$ — исходный вектор, Δx^k — решение задачи

$$\text{минимизировать } \nabla f(x^k)^T \Delta x$$

$$\text{при ограничении } x^k + \Delta x \in X$$

и α^k — минимизирующий размер шага

$$f(x^k + \alpha^k \Delta x^k) = \min_{\alpha \in [0, 1]} f(x^k + \alpha \Delta x^k).$$

Покажите, что каждый предел x^* последовательности $\{x^k\}$ удовлетворяет условию оптимальности

$$\nabla f(x^*)^T (x - x^*) \geq 0 \quad \text{для всех } x \in X.$$

Указание: докажите, что если $\{x^k\}$ имеет предел, а δ^k соответствует x^k , как в (б), то $\delta^k \rightarrow 0$ и, следовательно, также $\nabla f(x^k)^T \Delta x^k \rightarrow 0$. Переходя к пределу в неравенстве $\nabla f(x^k)^T \Delta x \leq \nabla f(x^k)^T (x - x^k)$ для всех $x \in X$, докажите, что предельная точка x^* должна удовлетворять неравенству $0 \leq \nabla f(x^*)^T (x - x^*)$ для всех $x \in X$.

5.32. Сходимость проекционного градиентного метода. Рассмотрите задачу минимизации и предположения из задачи 5.31. Рассмотрите также итерацию

$$x^{k+1} = x^k + \alpha^k (\bar{x}^k - x^k),$$

где \bar{x}^k является проекцией вектора $x^k - s \nabla f(x^k)$ на X , где s — некоторая фиксированная положительная скалярная величина, т. е. \bar{x}^k является решением задачи

$$\begin{aligned} & \text{минимизировать} \quad |x - x^k + s \nabla f(x^k)|^2 \\ & \text{при ограничении} \quad x \in X. \end{aligned}$$

(а) Предположите, что α^k является минимизирующим размером шага

$$f[x^k + \alpha^k (\bar{x}^k - x^k)] = \min_{\alpha \in [0, 1]} f[x^k + \alpha (\bar{x}^k - x^k)],$$

и покажите, что каждая предельная точка x^* для $\{x^k\}$ удовлетворяет условию оптимальности

$$\nabla f(x^*)^T (x - x^*) \geq 0 \quad \text{для всех} \quad x \in X.$$

Указание: следуйте указанию из задачи 5.31.

Используйте следующее необходимое условие, которому удовлетворяет проекция (см. лемму из разд. 5.5):

$$[x^k - s \nabla f(x^k) - x^k]^T (x - \bar{x}^k) \leq 0 \quad \text{для всех} \quad x \in X,$$

для того чтобы показать, что $s \nabla f(x^k)^T (\bar{x}^k - x^k) \leq -|\bar{x}^k - x^k|^2$.

(б) Предположите, что $\alpha^k = 1$ для всех k . Покажите, что если $s < 2/L$, то заключение из (а) имеет место.

5.33. Проекционный градиентный метод с диагональной масштабной матрицей. Рассмотрите задачу

минимизировать $f(x)$
при ограничении $x \geq 0$
из разд. 5.7. Покажите, что итерация

$$x_i^{k+1} = \max \left\{ 0, x_i^k - \alpha^k b_i \frac{\partial f(x^k)}{\partial x_i} \right\}, \quad i = 1, \dots, n,$$

где b_i — некоторые положительные скалярные величины, эквивалентна итерации проекционного градиентного метода

$$y_i^{k+1} = \max \left\{ 0, y_i^k - \alpha^k \frac{\partial h(y^k)}{\partial y_i} \right\}, \quad i = 1, \dots, n,$$

где переменные x_i и y_i связаны соотношением $x_i = \sqrt{b_i} y_i$, а $h(y) = f(Ty)$ и T является диагональной матрицей, i -й диагональный элемент которой равен $\sqrt{b_i}$.

5.34. Маршрутизация в сетях с часто меняющейся топологией [78]. В некоторых ситуациях (например, в пакетных радиосетях с передвигающимися источниками) изменения в топологии происходят настолько часто, что алгоритмы распространения топологии оказываются в чем-то непрактичными. Алгоритмы этой задачи разработаны для того, чтобы справляться с ситуациями этого типа. Рассмотрите связный неориентированный граф со специальным узлом, называемым *адресатом*. Рассмотрите совокупность S ориентированных ациклических графов, полученных путем приписывания какого-либо одного направления каждой неориентированной линии. Говорят, что граф G из S имеет в качестве корня *адресат*, если для каждого узла су-

существует ориентированный путь в G , выходящий из этого узла и заканчивающийся в адресате. Покажите, что следующие два распределенных асинхронных алгоритма, начинающие свою работу с произвольного графа из \mathcal{C} , создадут в результате такой граф из \mathcal{C} , который имеет в качестве корня адресат.

Алгоритм А. Любой узел, кроме адресата, не имеющий уходящих от него линий, меняет ориентацию всех смежных с ним линий на противоположную. (Это повторится до тех пор, пока каждый узел, кроме адресата, не будет иметь хотя бы одну уходящую от него линию.)

Алгоритм Б. У каждого узла i кроме адресата имеется список тех его соседних узлов j , которые имеют обратную ориентацию соответствующих линий (i, j) . На каждой итерации каждый узел i , не имеющий уходящих от него линий, меняет ориентации линий (i, j) для всех j , не содержащихся в его списке, на противоположные и затем полностью опустошает список. Если таких j нет (т. е. список полон), то узел i меняет ориентации всех входящих к нему линий на противоположные и затем опустошает список. Вначале все списки пусты.

Указание: для алгоритма А припишите каждому узлу индивидуальный номер. Каждому набору узловых номеров поставьте в соответствие граф из \mathcal{C} , у которого каждая линия ориентирована от большего к меньшему номеру. Рассмотрите алгоритм, в котором каждый узел меняет ориентации линий путем изменения своего номера. Используйте аналогичную идею для алгоритма Б.

5.35. *Маршрутизация виртуальных цепей методом лавинного алгоритма.* В некоторых сетях, в которых бывает трудно уследить за изменениями в топологии (например, в пакетных радиосетях), можно использовать лавинный алгоритм для установления виртуальных цепей. Основная идея состоит в том, чтобы узел m , который хочет установить виртуальную цепь с узлом n , рассылал по сети пробный пакет методом лавинного алгоритма. Каждый транзитный узел, который пересылает дальше этот пакет, отмечает в нем свой номер-идентификатор, для того чтобы узел-адресат n , получив пробный пакет, смог узнать маршрут, по которому он прошел. Затем узел-адресат может выбрать один из возможно большего числа маршрутов, которые были доставлены копиями пробного пакета, и приступить к установлению данной виртуальной цепи. Опишите один или несколько лавинных протоколов, которые сделают этот процесс работающим. Учтите вопросы, связанные с бесконечными или слишком долгими циркуляциями по сети сообщений, с подходящей нумерацией пробных пакетов, с непредвиденными выходами из строя линий и задержками, с возможной путаницей в двух концевых узлах виртуальных цепей и т. д.

5.36. *Оптимальная динамическая маршрутизация, основанная на стратегиях вычисления отношений.* Рассмотрите задачу оптимальной маршрутизации из разд. 5.4 в дейтаграммной сети. Предположим, что мы хотим реализовать множество путейых потоков $\{x_p\}$, вычисленных на основе номинального множества интенсивностей входного трафика $\{r_w\}$. Обычное решение, обсуждавшееся в разд. 5.5, состоит в том, чтобы направлять трафик каждой ОА-пары w таким образом, чтобы подгонять действительные части пакетов, направляемых по путям $p \in P_w$, как можно ближе к желаемым x_p/r_w . При такой реализации каждая ОА-пара w учитывает изменения своего входного трафика r_w , но совсем не принимает во внимание ни изменения во входных трафиках других ОА-пар, ни длины очередей в сети. В этой задаче рассматривается следующая более динамичная реализация. Предположим, что каждая ОА-пара w вычисляет среднее число пакетов N_p , перемещающихся в настоящий момент по каждому пути $p \in P_w$, используя теорему Литтла

$$N_p = x_p T_p \quad \text{для всех} \quad p \in P_w, \quad w \in W,$$

где T_p — оцениваемая средняя пакетная задержка в оба конца на пути p (время между моментом, когда пакет вошел в сеть, и моментом, когда пришло подтверждение о том, что этот пакет достиг адресата). Узел-отправитель каждой ОА-пары следит за *реальным* числом \tilde{N}_p пакетов, находящихся на маршруте p , но для которых еще не пришло подтверждение, и выбирает маршруты для пакетов таким образом, чтобы отношения \tilde{N}_p/N_p для всех путей $p \in P_w$ были примерно одинаковы. Заметим, что эта схема чувствительна к изменениям в задержке на путях некоторой ОА-пары, вызванным статистическими флуктуациями во входном трафике и(или) результатами маршрутизации других ОА-пар. Трудности с этой схемой состоят в том, что при вычислении чисел N_p не известны задержки T_p и поэтому необходимо как-то оценивать T_p . Простейшим способом является использование значений средних задержек, измеренных в течение предыдущего временного периода. Вам предлагается завершить анализ этой схемы на следующем простом, частном примере. Предположим, что имеются только одна ОА-пара и два пути с потоками и задержками на них, обозначенными через x_1 , x_2 и $T_1(x)$, $T_2(x)$ соответственно, где $x = (x_1, x_2)$. Вид функций задержки на пути $T_1(x)$, $T_2(x)$ не известен. Пусть $x_1 + x_2 = r$ для некоторой константы r . Предположим, что мы измеряем x_1 , x_2 , $T_1(x)$, $T_2(x)$, а затем вычисляем новые номинальные путевые потоки \bar{x}_1 , \bar{x}_2 , используя для этого некоторый алгоритм, и после этого реализуем их в соответствии со схемой отношений, описанной выше.

- (а) Докажите, что реальный вектор путевых потоков $\bar{x} = (\bar{x}_1, \bar{x}_2)$ определяется из $\bar{x}_1 + \bar{x}_2 = r$ и соотношения

$$\frac{\bar{x}_1 T_1(\bar{x})}{\bar{x}_1 T_1(x)} = \frac{\bar{x}_2 T_2(\bar{x})}{\bar{x}_2 T_2(x)}.$$

- (б) Предположим, что T_1 и T_2 монотонно возрастают в том смысле, что если $z = (z_1, z_2)$ и $z' = (z'_1, z'_2)$, такие, что $z_1 > z'_1$, $z_2 < z'_2$ и $z_1 + z_2 = z'_1 + z'_2$, то $T_1(z) > T_1(z')$ и $T_2(z) < T_2(z')$. Покажите, что путевые потоки \bar{x}_1 , \bar{x}_2 из (а) лежат в интервале между x_1 и \bar{x}_1 и x_2 и \bar{x}_2 соответственно.
- (в) Рассмотрите выпуклую стоимостную функцию вида $\sum_{(i,j)} D_{ij}(F_{ij})$. При предположении из (б) покажите, что если \bar{x} приводит к меньшей стоимости, чем x , то то же самое верно и для \bar{x} .

6. Управление потоками

6.1. Введение

В большинстве сетей возникают ситуации, когда поступающая извне нагрузка больше той, которая может быть обслужена даже при оптимальной маршрутизации. Тогда, если не предпринять никаких мер по ограничению поступающего трафика, размеры очередей на наиболее нагруженных линиях будут неограниченно расти и в конце концов превысят размеры буферов в соответствующих узлах. Когда это происходит, пакеты, поступающие в узлы, для которых нет свободного места в буфере, будут сброшены и позднее переданы повторно, что приведет к пустой трате ресурса связи. В результате возникает эффект, аналогичный транспортной пробке на автострате, когда при увеличении поступающей нагрузки действительная пропускная способность сети уменьшается, а задержка пакета становится чрезвычайно большой. Таким образом, иногда необходимо не допускать в сеть какую-то часть поступающего трафика для того, чтобы избежать подобных перегрузок. Это является функцией алгоритмов управления потоками.

В этой главе мы опишем некоторые схемы, используемые в настоящее время для управления потоками, объясним их преимущества и области применения и обсудим возможные пути их улучшения. В оставшейся части этого раздела мы уточним главные цели управления потоками. В разд. 6.2 описываются оконные стратегии, которые в настоящее время являются наиболее распространенными методами управления потоком. В разд. 6.3 рассматривается управление потоком в некоторых конкретных сетях (ARPANET, TYMNET, SNA и Codex). Раздел 6.4 посвящен более сложным схемам, цель которых — улучшить оконные стратегии.

Всюду в этой главе делается акцент на управление потоком на сетевом уровне. Вопросы управления потоком на транспортном уровне рассматриваются в разд. 6.2.4.

6.1.1. Главные цели управления потоками

В основном вопрос управления потоком возникает, когда имеется ограничение на скорость передачи между двумя точками вследствие ограниченной пропускной способности линии пере-

дачи или узла обработки. Таким образом, схема управления потоком может потребоваться на участках передачи между двумя пользователями (транспортный уровень), между пользователем и входной точкой подсети (сетевой уровень) или между двумя узлами подсети (сетевой уровень). Мы будем делать акцент на вопросах управления потоком внутри подсети, так как управление потоком в других случаях во многих отношениях аналогично и часто оказывается менее сложным, чем внутри подсети.

Наше внимание будет сосредоточено на двух целях управления потоком. Во-первых, на *установлении подходящего компромисса между притеснением пользователей и удержанием средней задержки сообщения на разумном уровне*. Во-вторых, на *соблюдении справедливости по отношению ко всем пользователям, когда часть поступающего трафика не допускается в сеть*. Другая цель, доминировавшая на заре построения сетей связи, когда стоимость объема буфера была очень большой, состоит в том, чтобы не допустить уменьшения пропускной способности и тупиковой ситуации из-за переполнения буферов. Мы также считаем эту цель важной и позднее обсудим механизм уменьшения пропускной способности и средств избежания тупиковых ситуаций. Однако эта цель будет косвенно достигаться путем удержания средней задержки пакета (и, следовательно, средних размеров очередей) на низком уровне. Основная причина того, что внимание будет сосредоточено на средней задержке, а не на переполнении буфера, объясняется желанием упростить анализ и облегчить понимание принципов.

Сохранение малой задержки в подсети

Малая средняя задержка пакета, конечно, желательна с точки зрения пользователя. Однако важно понять, что управление потоком на сетевом уровне *не обязательно* уменьшает задержку для пользователей сети; оно просто перебрасывает задержку с сетевого уровня на более высокие уровни. Это означает, что путем ограничения входа в подсеть управление потоком заставляет пакеты ждать вне подсети, а не в очередях внутри ее. Таким образом, управление потоком, возможно, не окажет большую помощь пользователю, чьи пакеты вынуждены проходить очень медленно и, как оказывается в действительности, оно иногда увеличивает задержку. Польза от управления потоком в основном состоит в том, что оно может предотвратить появление катастрофических трафиковых перегрузок в подсети, способных ухудшить положение многих пользователей. Единственный способ уменьшить неудовлетворительность пользователя по поводу больших задержек состоит в гарантировании того, чтобы потребность в управлении потоком не возникала слишком часто; этого можно

достичь либо путем увеличения сетевых ресурсов связи (пропускных способностей линий и т. д.), либо путем улучшения алгоритма маршрутизации. Другой возможностью, конечно, является запрещение доступа новых пользователей к сети, если она находится в почти перегруженном состоянии, но это порой бывает трудно реализовать и вызывает неудовлетворенность другого характера.

Основной причиной, почему важно сохранять задержку малой внутри подсети, а не вне ее, является то, что при этом не тратятся ресурсы подсети на повторную передачу пакетов. Последнее происходит в двух случаях: во-первых, когда размеры очередей становятся такими, что происходит переполнение буферов в узлах и приходящие пакеты сбрасываются; во-вторых, когда подтверждения возвращаются настолько поздно, что узел-источник ошибочно думает, что некоторые пакеты потеряны, и начинает повторно передавать их. При повторных передачах тратятся впустую ресурсы сети; эти передачи существенно уменьшают пропускную способность сети и приводят к распространению перегрузок.

Заметим, однако, что сохранить сетевую задержку малой, когда предлагаемая нагрузка велика, можно только путем уменьшения числа сообщений, передаваемых по сети. Поэтому должен существовать естественный компромисс между разрешением свободного доступа пользователя к сети и сохранением задержки на достаточно низком уровне для того, чтобы повторные передачи или другие нежелательные явления не ухудшали сетевые характеристики. Несколько упрощенная рекомендация состоит в том, чтобы управление потоком вообще не использовалось, когда сетевая задержка находится ниже некоторого критического уровня, а в условиях большой нагрузки оно должно отвергать столько предлагаемой нагрузки, сколько необходимо для того, чтобы сохранить задержку на критическом уровне. К сожалению, это проще сказать, чем сделать. так как ни задержку, ни прошедший по сети трафик невозможно представить разумным образом в виде скалярных величин в контексте управления потоком.

Соблюдение справедливости

Когда часть поступающего трафика должна быть отвергнута, важно сделать это справедливо. Это не тривиальная задача, так как максимизация суммарной пропускной способности сети часто оказывается не совместимой со справедливостью. Например, рассмотрим ситуацию, представленную на рис. 6.1. Имеется $n + 1$ пользователь, каждый из которых предлагает 1 единица/с трафика на последовательность из n линий с пропускной способностью 1 единица/с. Трафик одного пользователя должен проходить по всем n линиям, а трафик остальных пользователей — только по одной линии. Максимальной пропускной способности

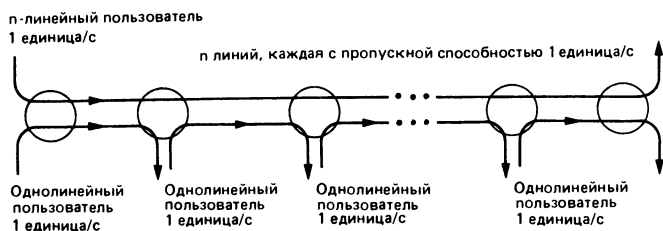


Рис. 6.1. Пример, показывающий, что максимизация пропускной способности может быть несовместима со справедливостью. Максимальной пропускной способности n единиц/с можно достигнуть, если полностью отвергнуть трафик n -линейного пользователя. Предоставляя одинаковые интенсивности $1/2$ единиц/с всем пользователям, можно достигнуть пропускной способности только $(n + 1)/2$ единиц/с.

n единиц/с можно достигнуть, если допустить весь трафик однолинейных пользователей и полностью отвергнуть трафик n -линейного пользователя. Однако если придерживаться равноправия и потребовать, чтобы для всех пользователей интенсивность принятого трафика была одинаковой, то максимальная пропускная способность составит $(n + 1)/2$ единицы/с.

Как правило, необходим компромисс между справедливостью и притеснением тех пользователей, которые больше других ответственны за перегрузки. Заметим также, что пользователи могут принадлежать разным приоритетным классам и в этом случае справедливость подразумевается только внутри приоритетного класса. Наконец, следует упомянуть, что в некоторых из предложенных в литературе схем управления потоком совершенно игнорируется справедливость. Например, есть схемы, которые, по существу, отвергают трафик от источников в узле, если состояние буферов в этом узле близко к переполнению, но в то же время продолжают принимать трафик, отосланный другими узлами. Беспокойство здесь вызывает то, что если какой-либо узел оказался перегружен трафиком, пришедшим от каких-либо других узлов, то этот приведший к перегрузкам транзитный трафик совершенно не сдерживается, тогда как источники, подключенные к этому перегруженному узлу, полностью отключаются.

Переполнение буфера

Термин «перегрузка» часто используется для обозначения явления, в котором увеличение поступающей нагрузки приводит к уменьшению пропускной способности и увеличению задержки. Перегрузка в этом смысле понятна каждому, кто знаком с поведением сети автомобильных дорог, на которых возникают транспортные пробки, когда слишком много машин одновременно оказывается на автостраде. При этом не только возрастают за-

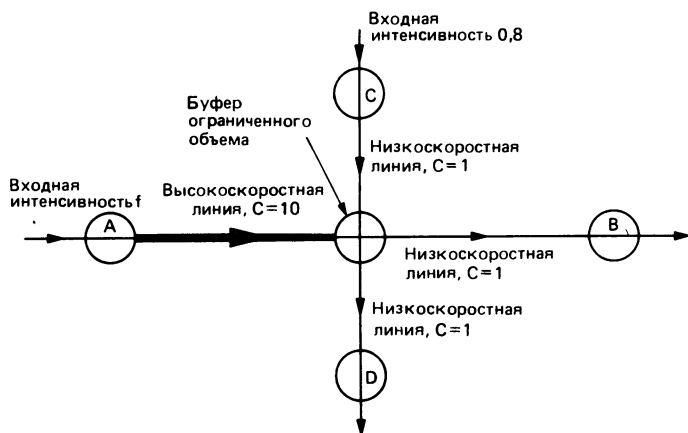
держки, но и падает пропускная способность автострады. Такое поведение явно контрастирует с поведением в моделях с очередями, рассмотренных при анализе маршрутизации; в последних уменьшение пропускной способности приводило к уменьшению задержки. Ранее уже было упомянуто, что перегрузки могут возникнуть в результате переполнения буфера. Следующий пример, заимствованный из работы [92], иллюстрирует, как это может произойти.

Пример 1

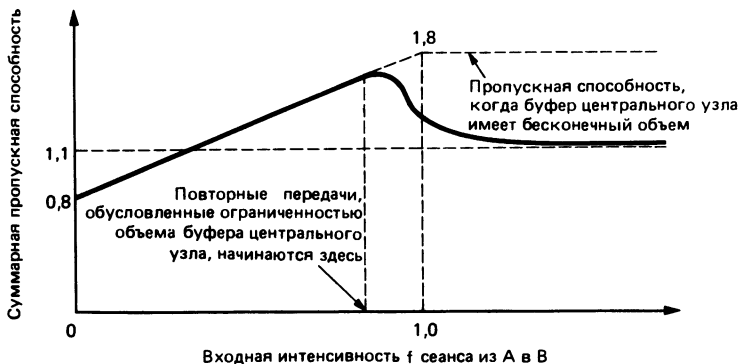
Рассмотрим сеть из пяти узлов, показанную на рис. 6.2, *а*, с двумя сеансами: один — от верхнего узла к нижнему с интенсивностью пуассоновского входного трафика 0,8, а второй — от левого узла к правому с интенсивностью пуассоновского входного трафика f . Предположим, что центральный узел имеет большой, но конечный буфер, который совместно используют эти два сеанса. Если буфер оказывается полным, то приходящий пакет отвергается и затем повторно передается узлом-отправителем. Для небольших f буфер редко заполняется и в этом случае суммарная пропускная способность системы равна $0,8 + f$. Когда f близко к единице (которая равна пропускной способности правой линии), буфер центрального узла будет почти всегда полон и в это время верхний и левый узлы будут почти все время повторно передавать пакеты, пытаясь занять место в буфере центрального узла. Так как левый узел передает в 10 раз быстрее, чем верхний узел, то поток, проходящий слева направо, будет примерно в 10 раз больше, чем поток, проходящий сверху вниз. Поскольку проходящий поток слева направо примерно равен единице (пропускной способности правой линии), то суммарная пропускная способность будет примерно равна 1,1. Более детально это показано на рис. 6.2, *б*, на котором видно, что по мере возрастания поступающей нагрузки f суммарная пропускная способность убывает до 1,1.

Этот пример также иллюстрирует, каким образом конечность размера буфера может привести к потере справедливости. Возможны ситуации, когда некоторые сеансы захватывают почти все буферы и в результате не дают другим сеансам пользоваться сетью.

Другая проблема, связанная с переполнением буфера, состоит в том, что могут возникнуть тупиковые ситуации, когда два или более узла не смогут дальше продвигать пакеты из-за того, что у всех потенциальных приемников нет свободного места в буфере. Простейшим примером этого являются два узла *A* и *B*, направляющие пакеты непосредственно друг другу, как показано на рис. 6.3, *а*. Если все буферы обоих узлов *A* и *B* заполнены пакетами, адресованными *B* и *A* соответственно, то узлы попадут в тупик и будут безуспешно постоянно повторять передачу одних и тех же пакетов из-за того, что в буферах приемников не будет места, куда можно было бы записать эти пакеты. Эта проблема может также возникнуть в более сложных ситуациях, когда более двух узлов, образующих цикл, попадут в тупик из-за того, что их буферы будут заполнены пакетами, адресованными другим узлам в цикле (см. рис. 6.3, *б*). Существуют простые способы устранения подобных тупиковых ситуаций путем распределения пакетов по приоритетным уровням и выделения дополнительного места в буфере для пакетов с более высоким приоритетом [98,



(a)



(б)

Рис. 6.2. Пример, показывающий снижение пропускной способности вследствие повторных передач, вызванных переполнением буфера. *а* — При f , приближающемся к единице, буфер в центральном узле почти всегда будет полон, в результате чего возникают повторные передачи. Так как для сеанса из *A* в *B* используется линия, в 10 раз более быстрая, чем для сеанса из *C* в *D*, то у него в 10 раз выше шансы захвата свободного буфера и передачи пакета в центральный узел. В результате пропускная способность сеанса из *A* в *B* приближается к единице, а пропускная способность сеанса из *C* в *D* приближается к 0,1. *б* — Суммарная пропускная способность как функция входной интенсивности сеанса из *A* в *B*.

196]. Обычно пакету приписывается уровень приоритета, равный числу линий, по которым он прошел в сети, как описано на рис. 6.4. Если пакетам не разрешается проходить по петлям, то можно показать, что только что описанные тупиковые ситуации не возникают

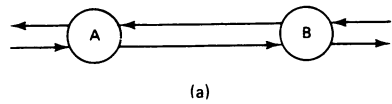
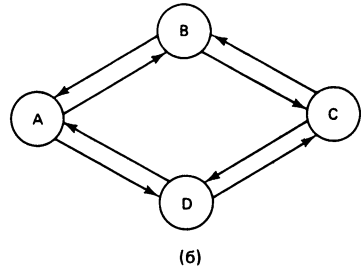


Рис. 6.3. Тупиковая ситуация, обусловленная переполнением буфера. На рис. *a* все буферы в *A* и *B* заполнены пакетами, адресованными *B* и *A* соответственно. В результате ни один пакет не может быть принят каким-либо узлом. На рис. *б* все буферы в *A*, *B*, *C* и *D* заполнены пакетами, адресованными *C*, *D*, *A* и *B* соответственно.



Несмотря на предыдущие примеры, перегрузки и переполнения буферов не являются столь же важными вопросами в управлении потоком, как задержка. При существующей технике стоимость буферов невысока и сети следует проектировать таким образом, чтобы переполнения буферов возникали редко. Другими словами, буферы должны быть настолько большими, чтобы до того, как они начнут переполняться, задержка становилась весьма существенной. Такая точка зрения не является общепринятой в области сетей в основном из-за влияния ARPANET, которая была построена до того, как стоимость буферов стала небольшой.

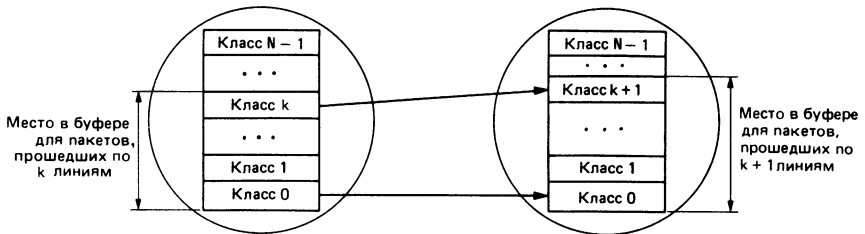


Рис. 6.4. Распределение памяти в узле по приоритетным классам для избежания тупиковых ситуаций, вызванных переполнением буферов. Пакет, прошедший по k линиям, принимается в узел только тогда, когда есть свободное место в буфере класса k или ниже, где k меняется в диапазоне от 0 до $N - 1$ (где N — число узлов). Если предположить, что пакеты, прошедшие более чем по $N - 1$ линиям, сбрасываются, так как они прошли по петле, то тупиковые ситуации никогда не возникнут. Доказательство состоит в том, чтобы показать по индукции (начиная с $k = N - 1$), что в каждом узле буфер класса k не может быть постоянно наполненным.

6.2. Оконное управление потоками

В этом разделе мы опишем наиболее часто используемый класс методов управления потоком. В подразд. 6.2.1—6.2.3 основной акцент будет сделан на управлении потоком в подсети связи. Управление потоком вне подсети кратко описано в подразд. 6.2.4.

В процессе передачи между передатчиком A и приемником B используется *оконное управление потоком*, если установлена верхняя граница на число единиц данных, которые уже были переданы передатчиком A , но о которых передатчику A не известно, что они попали к B (рис. 6.5). Верхняя граница (целое положительное число) называется *размером окна* или просто *окном*. Приемник B уведомляет передатчик A о том, что к нему попала единица данных путем отправления специального сообщения к A , называемого *разрешением* (другими названиями, используемыми в литературе, являются *подтверждение*, *резервирующее сообщение*, *квитанция* и т. д.). После получения разрешения передатчик A может отослать еще одну единицу данных к B . Таким образом, разрешение можно рассматривать как пропуск, который обязана получить единица данных, прежде чем войти в логический канал связи между A и B . Число разрешений, находящихся в использовании, не должно превышать размер окна.

Разрешения либо содержатся в специальных управляющих пакетах, либо прицепляются к обычным информационным пакетам. Их можно реализовать различными способами (см. взятые из практики примеры в разд. 6.3 и последующие обсуждения). Управление потоком используется при передаче по одной виртуальной цепи, группе виртуальных цепей (например, по всем виртуальным цепям, использующим один и тот же путь) или управлению подвергается весь поток пакетов, возникающих в одном узле и адресованных другому узлу. Передатчиком и приемником могут быть, например, два узла подсети связи или терминал пользо-

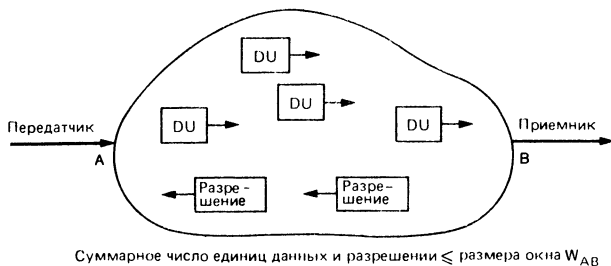


Рис. 6.5. Оконное управление потоком между передатчиком и приемником состоит в установлении верхней границы W_{AB} на число единиц данных и разрешений, находящихся в пути внутри сети.

вателя и входной узел подсети связи. Наконец, единицами данных в окне могут, например, быть сообщения, пакеты или байты.

Основная идея оконной стратегии состоит в том, чтобы интенсивность входного трафика у передатчика уменьшалась при замедлении возвращения разрешений. Следовательно, если на коммуникационном пути процесса возникают перегрузки, то сопутствующее увеличение задержки возвращения разрешений приводит к естественному замедлению интенсивности передачи данных передатчиком. Однако оконная стратегия допускает еще одну возможность, а именно приемник может умышленно задержать отправленные разрешения для того, чтобы ограничить интенсивность передачи в данном процессе. Например, приемник может сделать это для того, чтобы избежать переполнения буфера.

В последующем обсуждении мы рассмотрим две стратегии: оконное управление от конца до конца и паузовое оконное управление. Первая стратегия относится к управлению потоком между входным и выходными узлами подсети для некоторого процесса передачи, а вторая — к управлению потоком между каждой парой последовательных узлов вдоль пути виртуальной цепи.

6.2.1. Оконное управление от конца до конца

В самой распространенной версии управления потоком от конца до конца размер окна равен $W \cdot A$, где W и A — некоторые положительные числа. Каждый раз, когда новая партия из A единиц данных доходит до узла-адресата, назад к источнику отсылается разрешение на передачу новой партии из A единиц данных. В некоторых вариантах этой схемы узел-адресат отсылает разрешение для новых A единиц данных после получения только первой единицы из партии, состоящей из A единиц данных (см. описание шаговой схемы SNA в следующем разделе). Для упрощения последующего изложения мы будем считать, что $A = 1$, но наши выводы будут верны независимо от величины A . Для конкретности мы будем вести изложение в терминах пакетов, хотя окно, о котором пойдет речь, может состоять из других единиц данных, таких, как байты.

Обычно используется какая-либо схема нумерации пакетов и разрешений для того, чтобы можно было установить соответствие между разрешениями и ранее переданными пакетами. Одной из возможностей является использование скользящего оконного протокола, аналогичного тому, который применялся для управления линией передачи данных, когда пакет содержит порядковый номер и следующий ожидаемый номер. Последнее число может служить в качестве одного или нескольких разрешений для целей управления потоком (см. также обсуждения в подразд. 2.8.1 в гл. 2). Например, предположим, что узел A получил от узла B

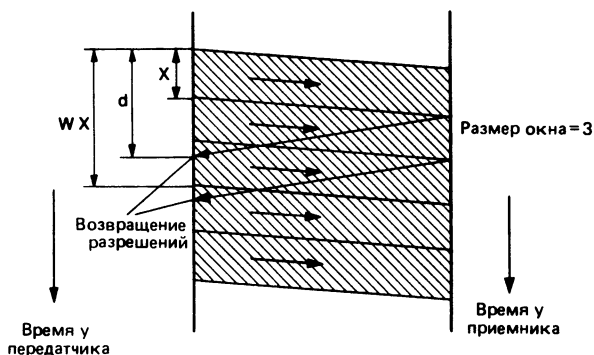


Рис. 6.6. Пример полнораспространенной передачи с размером окна $W = 3$. Задержка при передаче туда и обратно d меньше, чем время WX , необходимое для передачи всего окна из W пакетов.

пакет со следующим ожидаемым номером k . Тогда A знает, что до B дошли все пакеты, отосланные A с номерами, меньшими k , и поэтому узлу A разрешается отослать те пакеты, номера которых не превышают $k + W - 1$ и которые он еще не отослал, где W — размер окна. В такой схеме оба числа — порядковый номер и следующий ожидаемый номер — представлены по модулю m , где $m \geq W + 1$ аналогично тому, как это было сделано в системе ARQ на n шагов назад (задача 6.12). В некоторых сетях (например, ARPANET и сеть Codex (см. разд. 6.3)) эта схема используется в сочетании с протоколом подтверждения от конца до конца.

Для упрощения последующего изложения мы не будем обращать внимания на конкретный способ реализации разрешений. Будем предполагать, что узел-источник просто считает число x пакетов, которые он передал, но для которых еще не получил назад разрешение, и передает новые пакеты только тогда, когда $x < W$.

На рис. 6.6 показан поток пакетов для случая, когда задержка между отправлением пакета и возвращением разрешения меньше, чем время, необходимое для передачи всего окна из W пакетов, т. е.

$$d \leq WX,$$

где X — время передачи одного пакета. (Для упрощения последующего изложения считается, что все пакеты имеют одинаковое время передачи и одинаковую задержку возвращения разрешения.) В этом случае источник имеет возможность передавать пакеты с полной скоростью, равной $1/X$ пакетов/с, и управление потоком в данном случае оказывается неактивным.

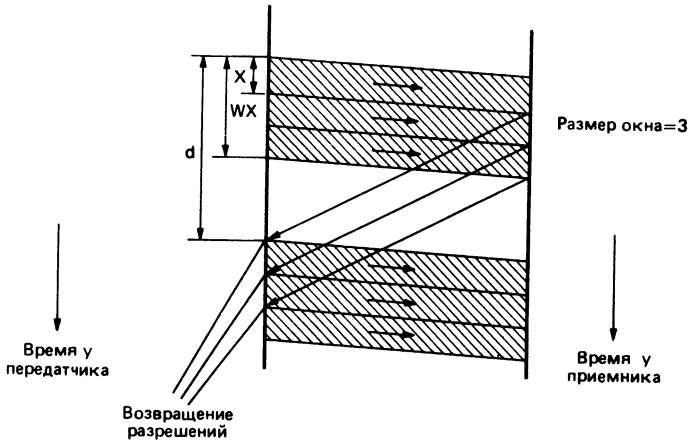


Рис. 6.7. Пример передачи с задержкой при размере окна $W = 3$. Задержка при передаче туда и обратно d больше, чем время Wx , необходимое для передачи всего окна из W пакетов. В результате оконное управление потоком становится активным, ограничивающим входную интенсивность величиной W/d .

Случай, в котором управление потоком оказывается активным, показан на рис. 6.7. Здесь

$$d > Wx$$

и задержка возвращения разрешения d оказывается настолько большой, что все W пакетов будут переданы до момента возвращения первого разрешения. Предполагая, что источник всегда имеет ожидающий в очереди пакет, скорость передачи можно считать равной W/d пакетов/с. Если подытожить результаты, приведенные на рис. 6.6 и 6.7, то видно, что максимальная скорость передачи, соответствующая задержке в оба конца d , равна

$$r = \min \left\{ \frac{1}{x}, \frac{W}{d} \right\}. \quad (6.1)$$

На рис. 6.8 иллюстрируется механизм управления потоком; скорость передачи источника уменьшается в ответ на перегрузки и присущие им большие задержки. Кроме того, оконные схемы реагируют очень быстро на перегрузки — не более чем за время передачи W пакетов. Такая быстрая реакция в сочетании с малыми дополнительными нагрузками является главным преимуществом оконных стратегий по сравнению с другими (не оконными) схемами.

Недостатки оконного управления от конца до конца

Оконное управление от конца до конца имеет некоторые недостатки, которые мы теперь обсудим. Напомним, что двумя главными целями управления потоком являются установление разум-



Рис. 6.8. Зависимость скорости передачи от величины задержки в оба конца в системе с оконным управлением потоком. Эта слишком упрощенная зависимость предполагает, что все пакеты требуют одинаковое время передачи в источнике и задержки при передаче туда и обратно для пакета и разрешения одинаковы.

ного компромисса между малой задержкой и большой пропускной способностью и соблюдение справедливости по отношению ко всем пользователям. Оказывается, что оконная стратегия от конца до конца (с фиксированным размером окна) не является полностью удовлетворительной ни в том ни в другом отношении.

Рассмотрим сначала задержку. Предположим, что имеется n процессов в сети, потоки которых активно управляются с размерами окон W_1, \dots, W_n . Тогда суммарное число пакетов в сети равно $\sum_{i=1}^n \beta_i W_i$, где множитель β_i принимает значения между 0 и 1 в зависимости от относительной величины времени возвращения разрешения. В соответствии с теоремой Литтла средняя задержка пакета равна

$$T = \frac{\sum_{i=1}^n \beta_i W_i}{\lambda},$$

где λ — пропускная способность (суммарный принятый входной трафик процессов). По мере возрастания числа процессов с активно управляемым потоком пропускная способность λ начинает подходить к ограничению, которое задается пропускной способностью линий и поэтому будет приближаться к константе. (Эта константа будет зависеть от сети, местоположения источников и адресатов и от алгоритма маршрутизации.) Поэтому задержка T будет возрастать примерно пропорционально числу процессов с активно управляемым потоком (точнее, сумме их размеров окон), как показано на рис. 6.9. Таким образом, если максимальное число процессов очень велико, то оконная схема от конца до конца может не удерживать задержку на разумном уровне и предотвратить перегрузки. Эти трудности объясняются тем, что оконная схема ущемляет

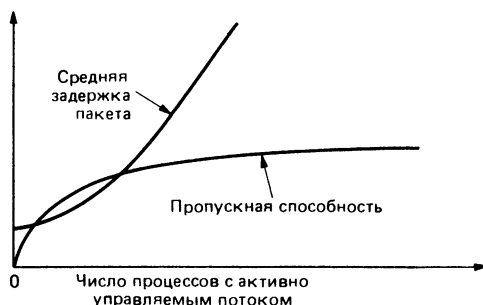


Рис. 6.9. Средняя задержка пакета и пропускная способность как функции числа процессов в сети, для которых оконное управление потоком является активным. Когда сеть сильно нагружена, средняя задержка пакета возрастает примерно линейно с ростом числа активных процессов, тогда как суммарная пропускная способность остается примерно постоянной. (При этом предполагается, что нет повторных передач, вызванных переполнением буфера и (или) большими задержками разрешений. Если такие повторные передачи имеют место, то пропускная способность может уменьшаться с ростом числа активных процессов.)

пользователей, когда задержки становятся большими, но не настолько, как это необходимо.

Можно рассмотреть использование малых размеров окон как средство борьбы с большими задержками в условиях большой нагрузки. К сожалению, существует предел, ниже которого нельзя уменьшить размеры окон без того, чтобы не ущемить пользователей в условиях малой нагрузки, в чем нет никакой необходимости. Действительно, если процесс использует путь из n линий с временем передачи пакета X на каждой линии, то время между моментом отправления пакета и возвращением разрешения будет не менее nX и оно будет значительно больше, если разрешениям не будет дан высший приоритет при их передаче по обратному каналу. Например, если разрешения на обратном пути прицепляются к пакетам, проходящим по тому же пути в обратном направлении, то время возвращения будет также не менее nX . Таким

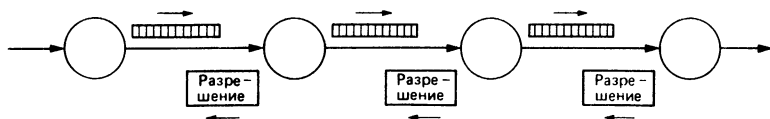


Рис. 6.10. Размер окна должен быть по крайней мере равен числу линий на пути, для того чтобы можно было достигнуть полноскоростной передачи. (Если предположить, что время передачи на всех линиях одинаково, то пакеты должны одновременно передаваться по всем линиям вдоль пути, для того чтобы не было остановок при передаче.) Если задержка разрешения сравнима с задержкой пакета в прямом направлении, то размер окна следует удвоить. Если временем распространения нельзя пренебречь, то необходим даже больший размер окна.

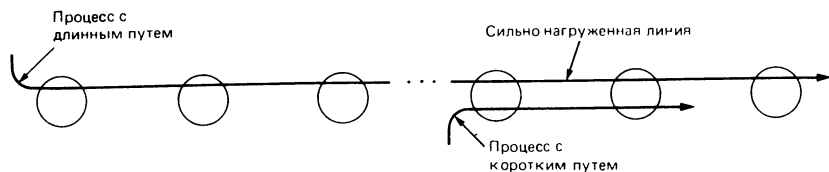


Рис. 6.11. Оконное управление от конца до конца больше подходит для процессов с длинными путями. Необходимо предоставить большое окно процессу с длинным путем, для того чтобы можно было достигнуть полноскоростной передачи. Поэтому процесс с длинным путем будет обычно иметь больше пакетов в очереди у перегруженной линии, чем процесс с коротким путем, и будет получать пропорционально больше обслуживания (предполагается, что пакеты передаются в порядке поступления).

образом, из рис. 6.8 видно, что передача с полной скоростью будет невозможна для этих процессов даже в условиях малой нагрузки, если размер окна не превышает число линий n на пути (рис. 6.10). Поэтому размеры окон обычно рекомендуется выбирать между n и $3n$. Такая рекомендация предполагает, что время передачи на каждой линии намного больше времени обработки и распространения. Если время распространения намного больше времени передачи, как, например, в спутниковых линиях, то подходящий размер окна будет намного больше.

Что действительно необходимо в оконном управлении потоком от конца до конца для достижения хорошего соотношения между задержкой и пропускной способностью, так это динамический выбор размеров окон. В условиях малой нагрузки окна должны быть большими и позволять вести беспрепятственную передачу, а в условиях большой нагрузки окна должны быть несколько «прикрыты» для того, чтобы задержка не становилась очень большой. Это не легко сделать в общем случае, но некоторые возможности будут рассмотрены в разд. 6.4.

Оконное управление от конца до конца может также быть плохим по отношению к справедливости. Ранее было показано, что подходящий размер окна для процесса должен быть пропорционален числу линий в его пути. Отсюда следует, что процессы с длинными путями могут иметь намного больше пакетов, ожидающих передачи по сильно перегруженной линии, чем процессы с короткими путями, в результате чего интенсивность проходящего трафика у этих процессов будет пропорционально больше. Типичная ситуация показана на рис. 6.11. Здесь окна всех процессов оказались заполненными пакетами, скопившимися у одной сильно перегруженной линии. Если пакеты передаются в порядке поступления, то получающаяся интенсивность передачи у каж-

дого процесса примерно пропорциональна размеру его окна и, таким образом, отдается предпочтение процессам с длинными путями.

Аспект справедливости оконного управления от конца до конца можно улучшить, если процессы, подчиненные управлению потоком и принадлежащие одному и тому же классу приоритетности, обслуживать методом кругового опроса в каждой очереди на передачу. Это, как правило, просто сделать, если процесс занимает одну виртуальную цепь. В дейтаграммной сети эффективность метода кругового опроса зависит от того, достаточно ли идентифицирующей информации несет каждый пакет для того, чтобы можно было в каждом узле поставить его в соответствие с конкретным процессом, подчиненным управлению потоком.

6.2.2. Пузловое оконное управление для виртуальных цепей

В этой стратегии имеется отдельное окно для каждой виртуальной цепи и пары смежных узлов на пути виртуальной цепи. Многое из того, что относится к оконному управлению от конца до конца, применимо также и к этой стратегии. Так как здесь путь, вдоль которого осуществляется управление потоком, эквивалентен, по существу, одной линии, то размер окна, измеряемый в пакетах, обычно равен двум или трем для наземных линий.

Сосредоточим внимание на паре последовательных узлов на пути виртуальной цепи, которые будем называть передатчиком и приемником. Основная идея пузловой схемы состоит в том, что приемник может избежать накопления большого числа пакетов в своей памяти путем уменьшения скорости, с которой он возвращает разрешения передатчику. В самой распространенной стратегии у приемника имеется буфер, в который можно записать W пакетов для каждой виртуальной цепи, и приемник возвращает разрешение передатчику только тогда, когда в его W -пакетном буфере имеется свободное место для записи еще одного пакета. Как только пакет покинет W -пакетный буфер, он либо будет отдан пользователю вне подсети, либо войдет в модуль управления линией передачи данных (УЛПД), ведущей к последующему узлу на пути виртуальной цепи.

Рассмотрим теперь взаимодействие окон вдоль трех последовательных узлов ($i - 1$, i и $i + 1$) на пути виртуальной цепи. Предположим, что W -пакетный буфер узла i полон. Тогда узел i отошлет разрешение узлу $i - 1$, как только он вручит еще один пакет модулю УЛПД на линии (i , $i + 1$), а это в свою очередь произойдет, как только узел i получит разрешение, высланное узлом ($i + 1$). Таким образом, происходит сцепление последовательных окон вдоль пути виртуальной цепи. В частности, предположим, что на какой-то линии образовалась перегрузка. Тогда W -пакет-

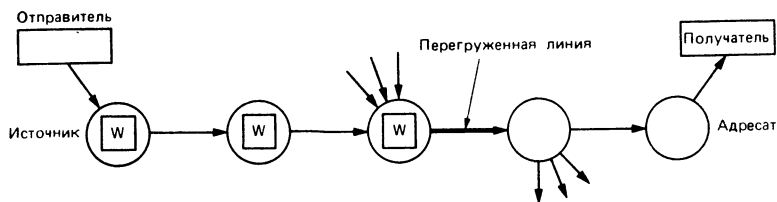


Рис. 6.12. Эффект обратного давления при поузловом управлении потоком. Каждый узел на пути виртуальной цепи может хранить не более чем W пакетов для этой виртуальной цепи. Свободные места в окнах всех предшествующих узлов, лежащих позади перегруженной линии, постепенно заполняются. В конце концов окно узла-отправителя заполнится и в этот момент передача прекратится.

ное окно в узле, от которого идет поток по этой перегруженной линии, заполнится для каждой виртуальной цепи, проходящей по этой линии. В результате W -пакетные окна узлов, лежащих выше (по течению потока) от перегруженной линии, будут постепенно заполняться, включая окна узлов-отправителей виртуальных цепей, проходящих по перегруженной линии. В это время потоки этих виртуальных цепей будут активно управляться. Явление, когда окна постепенно заполняются в направлении от точки перегрузки до узлов-отправителей виртуальных цепей, называется обратным давлением (рис. 6.12).

Одну привлекательную особенность поузлового оконного управления можно увидеть на рис. 6.12. В худшем случае, когда перегрузка образовалась на последней линии (скажем, на n -й) пути виртуальной цепи, суммарное число пакетов внутри сети для этой виртуальной цепи будет приблизительно равно nW . Если бы оконное управление потоком этой виртуальной цепи было от конца до конца, то суммарное число пакетов внутри сети было бы примерно таким же. (Предполагается, что в случае поузлового управления размер окна $W = 2$, а в случае управления от конца до конца $W \approx 2n$ на основании правила использования такого размера окна, которое бы в два раза превышало число линий на пути между передатчиком и приемником.) Важное отличие, однако, состоит в том, что в случае поузлового управления эти пакеты будут равномерно распределены вдоль пути виртуальной цепи, а в случае управления от конца до конца они будут сосредоточены у перегруженной цепи. Вследствие этого объем памяти, который необходим в каждом узле для избежания переполнения буфера, при поузловом оконном управлении может быть намного меньше, чем в случае оконного управления от конца до конца.

Равномерное распределение пакетов виртуальной цепи вдоль ее пути ослабляет проблему нарушения справедливости, которое

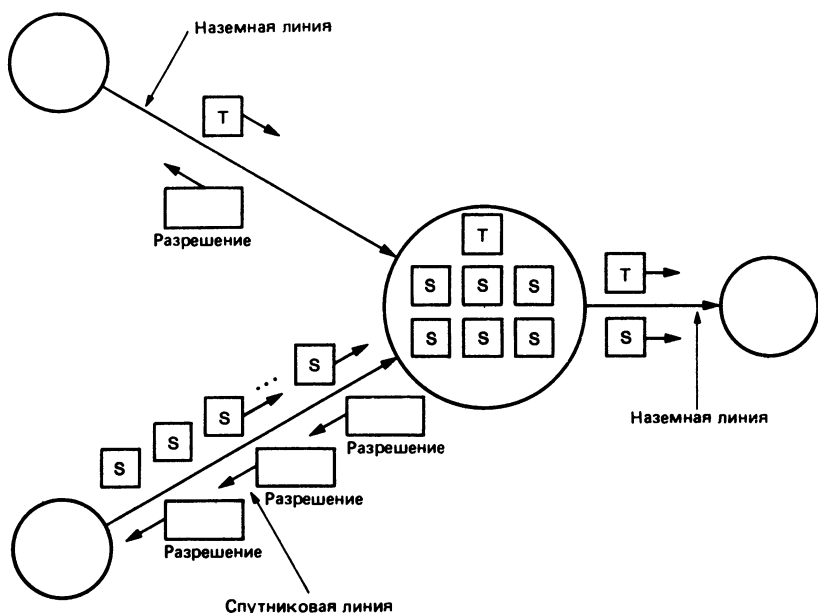


Рис. 6.13. Потенциальная задача соблюдения справедливости в узле, обслуживающем виртуальные цепи, пришедшие по спутниковой линии (большое окно), и виртуальные цепи, пришедшие по наземной линии (малое окно). Если виртуальные цепи обслуживаются в порядке «первым пришел — первым обслуживается», то виртуальные цепи с большими окнами получают лучшее обслуживание на последующей линии передачи. Эту проблему можно избежать, если обслуживать виртуальные цепи методом кругового опроса.

проявляется тогда, когда процессы с большими окнами монополизируют перегруженную линию за счет процессов с малыми окнами (см. рис. 6.11). Это, в частности, верно тогда, когда размеры окон для всех линий примерно одинаковы, как, например, тогда, когда цепи проходят только по наземным линиям. Проблема нарушения справедливости, однако, все же может возникнуть, когда используются спутниковые линии. Для таких линий из-за большого времени распространения необходимо выбирать большие размеры окон с целью получения беспрепятственной передачи в условиях малой нагрузки. Трудности возникают в узле, который обслуживает как виртуальные цепи с большими размерами окон, пришедшими по спутниковой линии, так и виртуальные цепи с малыми размерами окон, пришедшими по наземной линии (рис. 6.13). Если эти цепи выходят из узла по одной и той же линии передачи, то может быть нарушение справедливости, когда эта линия станет перегружена. Разумный способ избежать эту трудность состоит в передаче пакетов от разных виртуальных цепей методом кругового опроса (с учетом разных классов приоритетности).

6.2.3. Изаритмический метод

Изаритмический метод можно рассматривать как разновидность оконного управления потоком, в котором имеется только одно всеобщее окно для всей сети. Идея здесь состоит в том, что для ограничения суммарного числа пакетов в сети нужно иметь фиксированное число разрешений, циркулирующих по сети. Пакет входит в сеть после того, как он захватит одно из этих разрешений. Попав в свой узел-адресат, он затем отпускает разрешение. Таким образом, суммарное число пакетов в сети ограничено числом разрешений. Можно установить верхнюю границу средней задержки пакета, которая не будет зависеть от числа сеансов в сети. К сожалению, вопросы справедливости и перегрузки внутри сети зависят от того, как распределены по сети разрешения, которые при изаритмическом подходе не имеют адресов. В настоящее время не известен ни один разумный алгоритм управления местоположением разрешений и это является главной трудностью на пути практической реализации этой схемы. Существует также и другая трудность, связанная с тем, что разрешения могут исчезать вследствие различных аппаратурных неисправностей; возможно, что не существует простого способа контроля числа разрешений, циркулирующих по сети.

6.2.4. Оконное управление потоком на уровне пользователя

Многое из того, что было сказано до сих пор об оконных стратегиях, применимо к управлению потоком в сеансе между двумя пользователями либо на сетевом уровне, либо на транспортном уровне.

На рис. 6.14 показана типичная ситуация. Данные из машины *A* отсылаются пользователем входному узлу *NA* подсети, затем они направляются выходному узлу *NB* и машине *B*. Имеется управление потоком (сетевого уровня) между входным и выходным узлами *NA* и *NB* (либо от конца до конца, либо поузловое, включающее некоторую последовательность узлов). Имеется также оконное управление потоком (сетевого уровня) между машиной *A* и входным узлом *NA*, которое удерживает машину *A* от передачи узлу *NA* большого количества данных, чем то, с которым он может справиться. Точно так же имеется оконное управление потоком между выходным узлом *NB* и машиной *B*, которое удерживает *NB* от передачи машине *B* слишком большого количества данных. Таким образом, мы видим, что имеется система управления потоком сетевого уровня, простирающаяся от машины *A* до машины *B*, которая работает во многих отношениях так же, как система поузлового оконного управления потоком из подразд. 6.2.2. По существу, у нас есть путь из трех последовательных линий, в котором подсеть между узлами *NA* и *NB* мысленно представляется

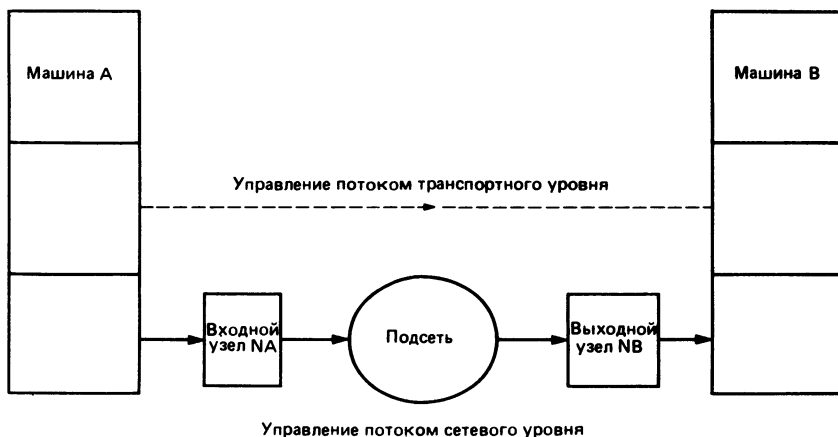


Рис. 6.14. Управление потоком от пользователя до пользователя. Пользователь отсылает данные из машины *A* другому пользователю в машину *B* через подсеть, используя входной и выходной узлы *NA* и *NB*. Мысленно можно представить, что имеется система поузлового управления потоком сетевого уровня по соединению *A—NA—NB—B*. Возможно также непосредственное управление потоком от пользователя до пользователя на транспортном уровне, например если потоки нескольких пользователей с разными требованиями к управлению потоком вместе управляются на сетевом уровне.

в виде средней линии. Аналогичная ситуация возникает, когда два пользователя общаются посредством нескольких сетей, соединенных шлюзами.

Может показаться, что только что описанной системы вполне достаточно для целей управления потоком, и во многих случаях так оно и есть. Например, в сетях с виртуальными цепями TUMNET и Codex только она и используется. Однако может потребоваться дополнительное управление потоком на транспортном уровне, посредством которого интенсивность входного трафика пользователя у машины *A* непосредственно может контролироваться приемником машины *B*. Причина состоит в том, что оконное управление потоком сетевого уровня от машины *A* до узла *NA* может совместно применяться к нескольким сеансам пар пользователей. Эти сеансы могут быть по разным причинам уплотнены в один поток трафика, но они могут иметь разные индивидуальные потребности в управлении потоком. Например, в SNA имеется алгоритм управления потоком транспортного уровня, который точно такой же, как и алгоритм сетевого уровня, за исключением того, что он принимается отдельно к каждому сеансу, а не к группе сеансов (см. описание, представленное в следующем разделе).

6.3. Обзор применяемых на практике методов управления потоками

В этом разделе описываются системы управления потоком нескольких существующих сетей и стандарта X.25. Во всех этих системах используются в том или ином виде оконные стратегии.

Управление потоком в сети ARPANET

Управление потоком в сети ARPANET основывается на оконном управлении от конца до конца. Весь поток пакетов каждой пары ЭВМ, подключенных к подсети (эти ЭВМ называются главными ЭВМ), рассматривается как нить в логическом тракте. Для каждого такого тракта имеется окно из восьми сообщений между соответствующим узлом-отправителем и узлом-адресатом подсети. Каждое сообщение состоит из одного или нескольких пакетов, но не больше, чем из восьми. Переданное сообщение содержит номер, указывающий его позицию в соответствующем окне. Узел-адресат, после того как в его распоряжение попало сообщение, посылает обратно специальный управляющий пакет (разрешение) узлу-отправителю, который в ARPANET называется RFNM (готов к следующему сообщению). RFNM также используется в качестве подтверждения от конца до конца в целях контроля ошибок. Получив RFNM, узел-отправитель освобождает место в соответствующем окне и ему разрешается передать еще одно сообщение. Если RFNM не получено после заранее установленного тайм-аута, то узел-отправитель посылает управляющий пакет с целью узнать у адресата, получил ли он соответствующее сообщение. Это исключает потери RFNM и создает механизмы для повторной передачи потерянных сообщений.

Кроме того, в сети имеется механизм, гарантирующий, что в памяти узла-адресата будет достаточно места для того, чтобы можно было вновь собрать многопакетные сообщения. (Пакеты в ARPANET могут прибывать в узел-адресат с нарушением порядка.) Каждое многопакетное сообщение должно резервировать себе достаточно места в буфере приемника для сборки до того, как оно начнет передаваться. Это делается посредством резервирующего сообщения REQALL (требование места), которое посылается узлом-отправителем узлу-адресату. Если затребованное место выдается, то узел-адресат посылает узлу-отправителю сообщение ALL (место выделено). При передаче по сети большого файла приходится передавать длинную последовательность многопакетных сообщений. В такой ситуации было бы слишком расточительно получать отдельное разрешение на каждое сообщение. Чтобы этого избежать, сообщения ALL прицепляются к возвращающимся RFNM для многопакетных сообщений и, таким образом, у всех сообщений, последующих за первым в файле, не будет задержки, связанной с резервированием места. Если зарезерви-

ванное место в буфере не используется узлом-отправителем в течение определенного тайм-аута, то оно возвращается узлу-адресату посредством специального сообщения. Однопакетные сообщения не нуждаются в предварительном резервировании до своей передачи. Если, однако, такое сообщение обнаружит буфер узла-адресата полным, то оно сбрасывается и его копия обязательно будет передана узлом-отправителем после получения сообщения, резервирующего место в буфере.

Ряд улучшений схемы ARPANET планировался для реализации в конце 1986 г. [167]. Во-первых, максимальный размер окна должен быть равен 127; это позволяет повысить эффективность работы в случае, когда используются спутниковые линии. Во-вторых, могут быть разрешены многократные независимые соединения (не более 256) между двумя ГВМ, каждое из которых имеет независимое окно; это предоставляет некоторую гибкость при размещении трафика разных приоритетов и (или) при удовлетворении разных требований по пропускной способности. В-третьих, принимаются некоторые усилия для повышения справедливости имеющегося алгоритма посредством некоторой схемы, которая пытается в каждом узле при резервировании справедливо распределить свободное место в буфере среди всех главных вычислительных машин. Наконец, описанная выше схема резервирования для многопакетных сообщений отменяется. Вместо нее узел-адресат просто резервирует свободное место для многопакетного сообщения после получения первого пакета этого сообщения. Если свободного места недостаточно, то пакет сбрасывается и повторно передается узлом-отправителем спустя тайм-аут.

Управление потоком в сети TUMNET

Управление потоком в сети TUMNET осуществляется отдельно для каждой виртуальной цепи посредством поузлового оконного управления. Имеется одно такое окно на виртуальную цепь и линию на пути этой виртуальной цепи. Каждое окно измеряется в байтах, и его размер меняется в зависимости от ожидаемой пиковой скорости передачи данных по виртуальной цепи. Управление потоком становится активным, когда наблюдается эффект обратного давления, обсуждавшегося в разд. 6.2.2. Справедливость обеспечивается путем обслуживания виртуальных цепей на линии методом кругового опроса. Это достигается путем размещения групп байтов от нескольких виртуальных цепей в кадры управления линией передачи данных. Максимальное число байтов в кадре для каждой виртуальной цепи зависит от степени перегруженности этой линии и приоритетного класса виртуальной цепи. Разрешения, используемые при управлении потоком, прицепляются к кадрам данных; в целях устранения избыточности эти разрешения сильно закодированы.

Управление потоком в архитектуре SNA

Напомним, что в подразд. 5.1.2 было сказано, что уровень в архитектуре SNA, который соответствует сетевому уровню в архитектуре BOC, называется уровнем управления путями; он включает функцию управления потоком, называемую управлением виртуальным маршрутом. Соответствующий алгоритм, называемый шаговой схемой виртуального маршрута, основывается на оконном управлении от конца до конца для каждой виртуальной цепи (или виртуального маршрута в терминологии SNA). Интересной особенностью этой схемы является то, что размер окна (измеряемый в пакетах) динамически меняется в зависимости от характеристик трафика. Минимальный размер окна обычно равен числу линий в пути, а максимальный размер окна в три раза больше. Заголовок каждого пакета содержит два бита, которые источник устанавливает равными нулю. Промежуточный узел пути, если он умеренно нагружен, устанавливает первый бит равным 1. Если узел сильно перегружен, то он устанавливает оба бита равными 1. В других случаях он не меняет биты. Получив пакет, узел-адресат смотрит на эти биты и увеличивает размер окна в случае отсутствия перегрузки, уменьшает размер окна в случае умеренной нагрузки или устанавливает размер окна равным минимальному значению в случае сильной перегрузки.

В действительности схема SNA немного отличается от оконной схемы от конца до конца, которая рассматривалась до сих пор. В основной схеме, обсуждавшейся в подразд. 6.2.1, размер окна равен $W \cdot A$ и каждое возвращающее разрешение разрешает передачу A новых пакетов. Мы сосредоточили внимание на случае $A = 1$. В SNA, однако, $W = 1$ и A (а не W) меняется между минимальным и максимальным размерами окна, как это описано выше. Кроме того, узел-адресат может отослать новое резервирующее сообщение для A пакетов после получения первого пакета из пачки с A пакетами. Таким образом, передача с максимальной скоростью в условиях малой нагрузки может вестись даже при $W = 1$.

Помимо управления виртуальным маршрутом в SNA осуществляется управление потоком транспортного уровня на сеансной основе, которое называется шаговым управлением сеансного уровня. Необходимость такого управления объясняется тем, что по виртуальному маршруту в SNA могут вестись несколько сеансов, каждый из которых может предъявлять разные требования к управлению потоком. Основная идея здесь состоит в том, чтобы не дать возможность передающему концу сеанса отсылать данные чаще, чем принимающий конец их сможет обработать. Шаговое управление сеансного уровня, по существу, является оконной схемой, в которой передающий конец может ввести в под-

сеть новый пакет только после получения разрешения (называемого в SNA шаговым откликом) от противоположного конца. Интересный момент состоит в том, что шаговые отклики могут задерживаться узлом, через который передающий конец сеанса имеет доступ к сети. Это позволяет подсети контролировать скорость, с которой она допускает в сети данные от внешних пользователей.

Управление потоком в сети Codex

В сети Codex осуществляется оконное управление от конца до конца для каждой виртуальной цепи. Размер окна, измеряемый в байтах, пропорционален числу линий на пути виртуальной цепи и номинальной скорости передачи данных в виртуальной цепи. Разрешения возвращаются вместе с подтверждениями от конца до конца, используемыми в целях контроля ошибок, так что оконная схема не требует большого дополнительного объема передач. Кадры управления линией передачи данных (УЛПД) на каждой линии формируются путем сцепления групп байтов нескольких виртуальных цепей, имеющих свой трафик в очереди. Существует максимальный размер группы для каждой виртуальной цепи. Виртуальные цепи обслуживаются на основе кругового опроса, и это создает естественный механизм соблюдения справедливости.

И кроме того, в сети Codex имеется механизм управления потоком, который не связан с оконной схемой, но играет дополнительную роль. Идея состоит в том, чтобы согласовать скорость передачи в виртуальной цепи по ее приходящей и уходящей линиям в каждом узле на ее пути в условиях большой нагрузки. Это осуществляется путем соответствующего подбора максимального числа байтов, которые виртуальная цепь может вставить в УЛПД-кадр (подробнее (см. [118])). Например, предположим, что скорость передачи в виртуальной цепи уменьшилась на какой-либо линии из-за перегрузки. Тогда узел-отправитель этой линии посылает специальное сообщение предшествующему узлу на пути виртуальной цепи, который затем уменьшает максимальное число байтов, переносимых УЛПД-кадром для этой виртуальной цепи. Одним из результатов действия такой схемы является то, что, когда начинают возникать перегрузки в направлении вниз по течению потока, окно от конца до конца виртуальной цепи не будет целиком заполняться в точке перегрузки, а будет более или менее равномерно распределено вдоль пути этой виртуальной цепи, начиная от узла источника и кончая точкой перегрузки. Это в сочетании с большим объемом памяти в узлах приводит к тому, что переполнения буферов становятся маловероятными.

Управление потоком в X.25

Как уже было упомянуто в подразд. 2.8.3, управление потоком на пакетном уровне в X.25 осуществляется посредством отдельного окна для каждой виртуальной цепи. Размер окна равен 2, но он может быть установлен равным 7 или 127. Управление потоком осуществляется в обоих направлениях, т. е. от машины пользователя (DTE) до входной точки сети (DCE) и обратно от DCE до DTE. Реализация оконной стратегии напоминает УЛПД-протоколы. Каждый пакет данных содержит трехбитовый порядковый номер и трехбитовый прицепной номер. (Если размер окна равен 127, то эти номера имеют длину семь бит.) Порядковый номер указывает позицию пакета в окне отправителя, а прицепной номер равен номеру, который отправитель ожидает получить в качестве номера следующего пакета. Таким образом, прицепной номер играет роль разрешения, позволяющего приемнику продвинуть соответствующее окно.

В протоколе X.25 имеется также еще один пункт, касающийся управления потоком между двумя DTE, поддерживающими связь посредством подсети. Формат пакета в X.25 содержит специальный бит (называемый битом D), который определяет, относится ли прицепной номер пакета, полученного DTE, к непосредственно подключенному DCE ($D = 0$) или к удаленному DTE ($D = 1$). В последнем случае прицепной номер служит в качестве разрешения для продвижения окна, используемого в целях управления потоком между двумя DTE.

6.4. Схемы управления потоком, основанные на регулировании интенсивности входного трафика

Мы видели, что одна из основных трудностей, с которыми приходится сталкиваться в процессе оконного управления потоком с фиксированными размерами окон, состоит в том, что средняя задержка пакета возрастает пропорционально числу процессов, активно управляемых по потоку. Средством борьбы с этим явлением может быть уменьшение размеров окон по мере увеличения числа таких потоков. К сожалению, не очень просто найти хорошие пути осуществления этого на практике. Поэтому мы сосредоточим внимание на более четких формулировках задачи управления потоком, в которых интенсивности входного трафика у процессов, потоки которых управляются, регулируются непосредственно в ответ на состояние трафика внутри сети.

При первом рассматриваемом подходе (подразд. 6.4.1) мы сформулируем оптимизационную задачу, которая математически выражает первую цель управления потоком — установление под-

ходящего баланса между высокой пропускной способностью и приемлемой средней задержкой пакета. При втором подходе (подразд. 6.4.2) мы сделаем акцент на соблюдении справедливости при поддержании средней задержки пакета на приемлемом уровне.

Алгоритмы, которые будут разработаны, устанавливают подходящий уровень интенсивности входного трафика для каждого процесса. Практическая реализация этих входных интенсивностей не всегда проста, в частности когда приходится иметь дело с быстро меняющимися поступающими нагрузками. В подразд. 6.4.3 мы обсудим, каким образом можно установить входные интенсивности, используя для этого оконные схемы. Размеры окон будут регулироваться в зависимости от уровня перегрузок внутри сети, благодаря чему смягчится главный недостаток управления потоком с фиксированными размерами окон.

6.4.1. Сочетание оптимальной маршрутизации и управления потоком

Мы рассмотрим возможность сочетания маршрутизации и управления потоком от конца до конца внутри подсети путем оптимального регулирования как маршрутных переменных, так и входных интенсивностей для пар отправитель—адресат (ОА-пар). Частным случаем является задача чистого управления потоком, в которой маршрутизация фиксируется, и единственными переменными, которые должны регулироваться, остаются входные интенсивности.

Исходная потоковая модель похожа на ту, которая рассматривалась в контексте оптимальной маршрутизации в разд. 5.4. Обозначим через r_w входную интенсивность ОА-пары w . В некоторых случаях r_w может измеряться в бит/с, и тогда ее оптимальное значение можно интерпретировать как требуемое значение интенсивности, усредненное по достаточно длинному интервалу времени (например, передать не более $r_w T$ бит в течение временного интервала длины T). Если r_w измеряется числом виртуальных цепей, то ее оптимальное значение можно интерпретировать как требуемое значение, которое узел-отправитель старается достигнуть путем блокировки или разрешения новых запросов на соединение, возникших во внешних пунктах. Аналогично в сетях, предназначенных для передачи речи и данных, r_w может относиться к скорости цифрового кодирования речи и непосредственно регулироваться в узлах-отправителях. Далее мы сначала сосредоточим внимание на формулировке задачи минимизации некоторой разумной стоимостной функции путем подгонки маршрутных переменных и входных интенсивностей r_w . Затем покажем, что *эта задача математически эквивалентна задаче оптимальной*

маршрутизации, рассмотренной в предыдущей главе (r_w фиксированы), и поэтому применимы представленные там условия оптимальности и алгоритмы.

Если минимизировать стоимостную функцию $\sum_{(i,j)} D_{ij}(F_{ij})$ задачи маршрутизации по путевым потокам $\{x_p\}$ и входным интенсивностям $\{r_w\}$, то можно обнаружить, что оптимальное решение будет разочаровывающим: $x_p = 0$ и $r_w = 0$ для всех p и w . Это указывает на то, что стоимостная функция должна включать штраф за то, что входные интенсивности становятся слишком малыми; это приводит к задаче

$$\begin{aligned} & \text{минимизировать} \quad \sum_{(i,j)} D_{ij}(F_{ij}) + \sum_{w \in W} e_w(r_w) \\ & \text{при ограничениях} \quad \sum_{p \in P_w} x_p = r_w \quad \text{для всех} \quad w \in W, \quad (6.2) \\ & \quad x_p \geq 0 \quad p \in P_w, \quad w \in W, \\ & \quad 0 \leq r_w \leq \bar{r}_w \quad \text{для всех} \quad w \in W. \end{aligned}$$

Здесь минимизация должна проводиться по $\{x_p\}$ и $\{r_w\}$. Данные величины \bar{r}_w означают желаемые входные интенсивности ОА-пары w , т. е. предлагаемую нагрузку для w , определяемую как входные интенсивности для w , которые будут при отсутствии управления потоками. Как и раньше, F_{ij} означает суммарный поток по линии (i, j) , т. е. сумму всех путевых потоков, проходящих по этой линии. Функции e_w имеют такой вид, как показано на рис. 6.15, и означают штраф за ущемление входной интенсивности r_w . Они являются выпуклыми, монотонно убывающими функциями на множестве положительных чисел $(0, \infty)$ и стремятся к ∞ , когда r_w стремится к нулю. Мы предполагаем, что их первые и вторые производные e'_w и e''_w существуют на $(0, \infty)$ и строго отрицательны и положительно соответствуют. Интересный класс функций e_w задается следующей формулой для их первой производной:

$$e'_w(r_w) = - \left(\frac{a_w}{r_w} \right)^{b_w}, \quad \text{где } a_w \text{ и } b_w \text{ являются заданными положительными константами.} \quad (6.3)$$

Как будет объяснено позднее в этом разделе, параметры a_w и b_w влияют на оптимальные значения входных интенсивностей r_w и на приоритет ОА-пары w соответственно.

Ценность предыдущей формулировки повысится, если мы расширим наше представление о w и будем рассматривать ее как класс пользователей, использующих один и тот же набор путей P_w . Это позволяет придавать разные приоритеты (т. е. разные функции e_w) разным классам пользователей, даже если они вместе используют одни и те же пути. Можно также рассмотреть задачу, в которой P_w состоит из единственного пути для каждой w . Это

Рис. 6.15. Типичный вид функции штрафа за ущемление входной интенсивности r_w .



задача чистого управления потоком, а именно выбор оптимальной части желаемого входного потока каждого класса пользователей, которая будет допущена в сеть.

Теперь покажем, что объединенная задача маршрутизации и управления потоком (6.2) математически эквивалентна задаче маршрутизации того типа, который рассматривался в разд. 5.4. Введем новую переменную y_w для каждого $w \in W$ посредством равенства

$$y_w = \bar{r}_w - r_w. \quad (6.4)$$

Можно рассматривать y_w как *избыточный поток* (часть \bar{r}_w , которая блокируется сетью) и считать, что этот поток проходит по *избыточной линии*, непосредственно соединяющей узел-отправитель с узлом-адресатом пары w , как показано на рис. 6.16. Если определить новую функцию F_w равенством

$$E_w(y_w) = e_w(\bar{r}_w - y_w), \quad (6.5)$$

то задачу (6.2) с учетом (6.4) можно переписать в виде

$$\text{минимизировать} \quad \sum_{(i,j)} D_{ij}(F_{ij}) + \sum_{w \in W} E_w(y_w) \quad (6.6)$$

при ограничениях $\sum_{p \in P_w} x_p + y_w = \bar{r}_w$ для всех $w \in W$,

$$x_p \geq 0 \quad \text{для всех} \quad p \in P_w, \quad w \in W,$$

$$y_w \geq 0 \quad \text{для всех} \quad w \in W.$$

Вид функции E_w из (6.5) показан на рис. 6.17. Так как $e_w(r_w) \rightarrow \infty$ при $r_w \rightarrow 0$ (т. е. назначается бесконечный штраф за полное запираение класса пользователей w), то $E_w(y_w) \rightarrow \infty$, когда избыточный поток y_w приближается к своему максимальному значению — максимальной входной интенсивности \bar{r}_w . E_w можно рассматривать как *задержку для избыточной линии*, а \bar{r}_w можно рассматривать как *пропускную способность этой линии*.

Теперь стало ясно, что задача (6.6) является задачей того же типа, что рассматривалась в разд. 5.4, и что применимы алгоритмы и условия оптимальности, описанные в разд. 5.5—5.7. В частно-

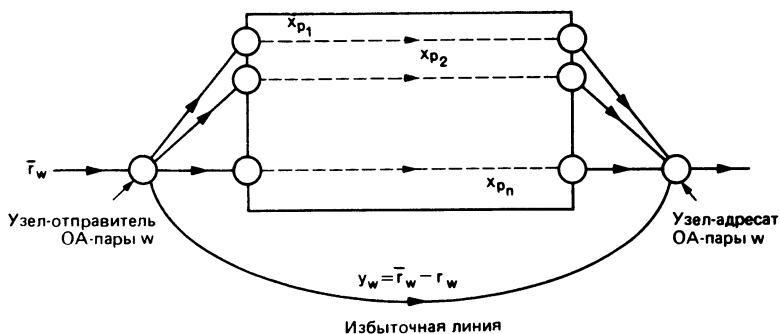


Рис. 6.16. Математическая эквивалентность задачи управления потоком задаче оптимальной маршрутизации, основанная на введении искусственной избыточной линии для каждой ОА-пары w . По избыточной линии проходит отвергнутый трафик, т. е. разница между желаемым и принятым входными потоками $\bar{r}_w - r_w$. Стоимость для избыточной линии получается из стоимости e_w за ущемление входа путем замены переменной.

сти, применение условия оптимальности, основанного на понятии кратчайшего пути, из разд. 5.5 приводит к следующему результату:

множество допустимых значений путевых потоков $\{x_p^*\}$ и входных интенсивностей $\{r_w^*\}$ является оптимальным для задачи (6.2) тогда и только тогда, когда выполняются следующие условия для всех $p \in P_w$ и $w \in W$:

$$\begin{aligned} x_p^* &> 0 \text{ только тогда, когда } d_p^* \leq d_{p'}^*, \text{ для всех } p' \in P_w, \\ d_p^* &\leq -e_w'(r_w^*), \\ r_w^* &< \bar{r}_w \text{ только тогда, когда } -e_w'(r_w^*) \leq d_p^* \text{ для всех } p \in P_w, \end{aligned} \quad (6.7a) \quad (6.7b)$$

где d_p^* — первопроизводная длина пути p [$d_p^* = \sum_{(i,j)} D_{ij}^*(F_{ij}^*)$] и F_{ij}^* — суммарный поток по линии (i, j) , соответствующий $\{x_p^*\}$.

Заметим, что условие оптимальности (6.7) зависит только от производных функций D_{ij} и e_w . Это означает, что добавление произвольных констант к D_{ij} или e_w не влияет на оптимальное решение. Заметим также, что из (6.7б) следует, что оптимальная

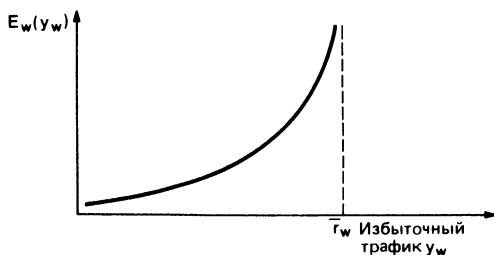
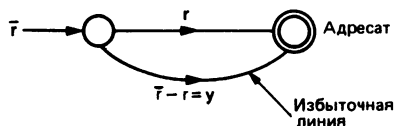


Рис. 6.17. Стоимость функция для избыточной линии. Стоимость $E_w(y_w)$ для избыточного трафика y_w равна стоимости $e_w(r_w)$ для входного трафика $r_w = \bar{r}_w - y_w$.

Рис. 6.18. Пример задачи, включающей отправителя и адресата, соединенных единственной линией.



точка r^* не зависит от \bar{r}_w , пока $\bar{r}_w > r^*$. Это является хорошим свойством стратегии управления потоком, мешающим пользователям (чей поток управляется активно) увеличить их долю ресурса путем повышения требований. Простой пример иллюстрирует условия оптимальности (6.7).

Пример 2

Рассмотрим ситуацию на рис. 6.18, когда имеется единственная линия, соединяющая отправитель и адресат. Стоимостной функцией является

$$-\frac{r}{C-r} + \frac{a}{r},$$

где первый член означает штраф за большую задержку [ср. с членом $D_{ij}(F_{ij})$ в (6.2)], а второй член — штраф за малую пропускную способность [ср. с членом $e_w(r_w)$ в (6.2)]. Константа C является пропускной способностью линии, а параметр a — некоторый положительный взвешивающий множитель. Соответствующей задачей маршрутизации будет (ср. с (6.6))

$$\text{минимизировать} \quad -\frac{r}{C-r} + \frac{a}{\bar{r}-y}$$

$$\text{при ограничениях} \quad r + y = \bar{r}, \quad r \geq 0, \quad y \geq 0,$$

где $y = \bar{r} - r$ означает величину предлагаемой нагрузки, отвергнутой управлением потока (или, что одно и то же, поток по фиктивной избыточной линии). Условия (6.7) показывают, что никакого управления потоком не будет ($y = 0$, $r = \bar{r}$), если

$$\frac{C}{(C-\bar{r})^2} < \frac{a}{\bar{r}^2},$$

т. е. если первопроизводная длина избыточной линии превышает первопроизводную длину настоящей линии. Это означает, что никакого управления потоком не будет, если

$$\bar{r} < C \frac{\sqrt{a}}{\sqrt{a} + \sqrt{C}}.$$

В соответствии с (6.7) при наличии управления потоком ($y > 0$, $r < \bar{r}$) две первопроизводные длины должны совпадать, т. е.

$$\frac{C}{(C-r)^2} = \frac{a}{(\bar{r}-y)^2}.$$

После подстановки $y = \bar{r} - r$ получаем результат

$$r = C \frac{\sqrt{a}}{\sqrt{a} + \sqrt{C}}.$$

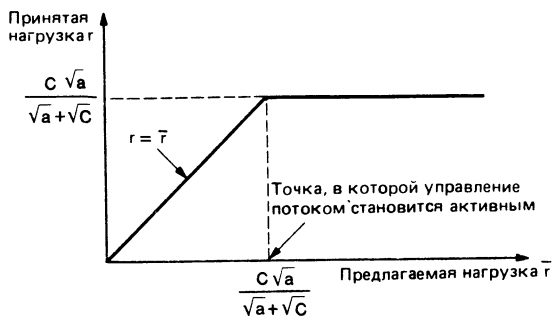


Рис. 6.19. Оптимальная принятая нагрузка как функция предлагаемой нагрузки из примера на управление потоком. Управление потоком становится активным, когда предлагаемая нагрузка превышает пороговый уровень, зависящий от весового множителя a .

Решение в виде функции предлагаемой нагрузки показано на рис. 6.19. Заметим, что пропускная способность не зависит от предлагаемой нагрузки, если она больше некоторой величины, как уже обсуждалось ранее. Максимальную пропускную способность $C\sqrt{a}/(\sqrt{a} + \sqrt{C})$ можно регулировать путем изменения параметра a ; она стремится к пропускной способности C при $a \rightarrow \infty$.

Смысл параметров a_w и b_w в стоимостной функции, определяемых формулой (ср. с (6.3))

$$e'_w(r_w) = -(a_w/r_w)^{b_w},$$

теперь может быть понят в свете условий оптимальности (6.7б). Рассмотрим два различных класса пользователей w_1 и w_2 , использующих одни и те же пути ($P_{w_1} = P_{w_2}$). Тогда из условий (6.7б) следует, что в точке оптимального решения, в которой оба класса пользователей ущемлены ($r_{w_1}^* < \bar{r}_{w_1}$, $r_{w_2}^* < \bar{r}_{w_2}$), должны выполняться равенства

$$-e'_{w_1}(r_{w_1}^*) = -e'_{w_2}(r_{w_2}^*) = \min_{p \in P_{w_1}} \{d_p^*\} = \min_{p \in P_{w_2}} \{d_p^*\}. \quad (6.8)$$

Если e'_{w_1} и e'_{w_2} определяются параметрами a_{w_1} , b_{w_1} и a_{w_2} , b_{w_2} согласно (6.3), то можно заметить следующее:

(а) Если $b_{w_1} = b_{w_2}$, то

$$\frac{r_{w_1}^*}{r_{w_2}^*} = \frac{a_{w_1}}{a_{w_2}},$$

и отсюда следует, что параметр a_w влияет на оптимальную относительную входную интенсивность класса пользователей w .

(б) Если $a_{w_1} = a_{w_2} = a$ и $b_{w_1} < b_{w_2}$ (рис. 6.20), то условие (6.8) показывает, что, когда входные потоки вынуждены быть малыми ($r_{w_1}^*$, $r_{w_2}^* < a$), классу пользователей w_2 (тому, у кото-

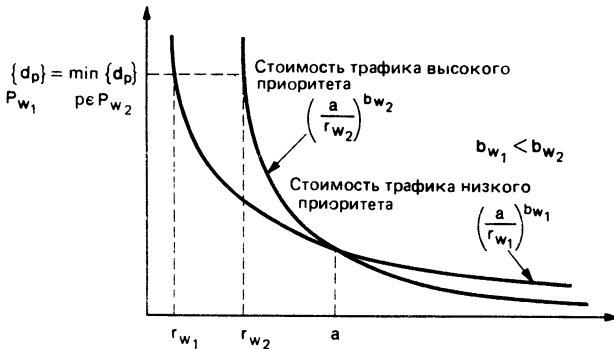


Рис. 6.20. Включение приоритетов классов пользователей в стоимостную функцию задачи управления потоком. Класс пользователей с большим b_w будет меньше ущемляться в условиях большой нагрузки.

рого параметр b_w больше) позволено допустить в сеть больше трафика. Отсюда следует, что параметр b_w влияет на относительный приоритет класса пользователей w в условиях большой нагрузки.

6.4.2. Максимальное управление потоком

Одним из наиболее сложных аспектов управления потоком является справедливое обращение со всеми пользователями в ситуации, когда необходимо не пустить часть трафика в сеть. Справедливость можно определить разными способами, но наше интуитивное понятие справедливости состоит в том, чтобы любому пользователю предоставить такое же право пользоваться сетью, как и любому другому пользователю. Рис. 6.21 проясняет некоторые неясности в этом распределении. Один сеанс проходит по тандемной последовательности всех дуг, а каждый из остальных сеансов — только по одной дуге. Желательно ограничить входные интенсивности сеансов 0, 1 и 2 величиной $1/2$ каждую, так как это поставит все эти сеансы в одинаковое положение в смысле интенсивности их трафиков. Однако было бы довольно бессмысленно ограничивать интенсивность сеанса 3 той же величиной $1/2$. Сеанс 3 лучше было бы ограничить величиной $5/2$, так как если ограничить меньшей величиной, то не будет использоваться какая-либо часть пропускной способности самой правой линии, а если ограничить большей величиной, то это будет несправедливо из-за того, что еще больше будет ограничен сеанс 0.

Этот пример наводит на мысль максимизации степени использования сети для тех пользователей, у которых она минимальная, и, таким образом, возникает понятие о *максимальном управлении*

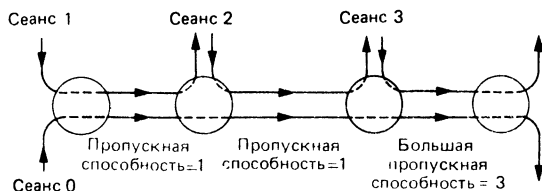


Рис. 6.21. Справедливое решение состоит в том, чтобы предоставить сеансам 0, 1 и 2 интенсивность $1/2$ каждому и сеансу 3 — интенсивность $5/2$, чтобы избежать пустой траты пропускной способности самой правой линии.

поток. После того как пользователям, которые находятся в самом затруднительном положении, будут предоставлены наилучшие из возможных условия передачи, возможно, останется достаточная свобода при выборе возможностей для остальных пользователей. Опять имеет смысл максимизация степени использования сети для тех пользователей, которые находятся в самом затруднительном положении и так далее до тех пор, пока не будут определены величины использования для всех пользователей. Другой подход к формализации интуитивного представления о справедливости, который, как оказывается, эквивалентен только что описанному, состоит в том, чтобы максимизировать степень использования для каждого пользователя i при ограничении, что увеличение использования для i -го пользователя не приведет к уменьшению степени использования для какого-либо другого пользователя, у которого она такая же, как у пользователя i или меньше.

Прежде чем определить эти максиминные степени использования сети более строго, уточним нашу модель. Пусть сеть задается ориентированным графом $G = (\mathcal{N}, \mathcal{A})$ и P означает множество сеансов, использующих эту сеть. Каждому сеансу p соответствует фиксированный путь в сети и поэтому p будет использоваться как для обозначения сеанса, так и его пути (если несколько сеансов используют один и тот же путь, то несколько индексов p означают один и тот же путь). Таким образом, в нашей модели предполагается метод маршрутизации по фиксированным путям (один путь для каждого сеанса). Входная интенсивность для сеанса p обозначается через r_p ; тогда поток на дуге a сети равен

$$F_a = \sum_{p \in P} \delta_p(a) r_p, \quad (6.9)$$

где $\delta_p(a)$ равно 1, если a лежит на пути p и равно 0 в противном случае. Обозначая пропускную способность дуги a через C_a ,

получаем следующие ограничения на вектор интенсивностей r

$$r_p \geq 0 \quad \text{для всех } p \in P, \quad (6.10a)$$

$$F_a \leq C_a \quad \text{для всех } a \in \mathcal{A}. \quad (6.10б)$$

Задача состоит в том, чтобы найти вектор интенсивностей, который был бы допустим (т. е. удовлетворял (6.10)) и был справедливым в описанном выше максиминном смысле. Теперь можно определить максиминную справедливость:

Вектор интенсивностей r является *максиминно справедливым*, если он удовлетворяет (6.10) и для каждого $p \in P$ интенсивность r_p нельзя увеличить (не нарушая при этом неравенства в (6.10)) без того, чтобы не уменьшить $r_{p'}$ для какого-либо сеанса p' , у которого $r_{p'} \leq r_p$. Более формально r является максиминно справедливым, если r удовлетворяет (6.10) и для каждого \bar{r} , удовлетворяющего (6.10), и для каждого $p \in P$; если $r_p < \bar{r}_p$, то для некоторого p' должны выполняться $r_p \geq r_{p'}$ и $r_{p'} > \bar{r}_{p'}$.

Важным свойством максиминно справедливого вектора интенсивностей является следующее: для каждого сеанса p найдется такая дуга a на пути p (называемая *узкой дугой пути* p), что, во-первых, $F_a = C_a$ и, во-вторых, r_p не меньше интенсивности любого другого сеанса, использующего эту узкую дугу. Чтобы в этом убедиться, заметим, что если бы такой дуги не было, то r_p можно было бы немного увеличить, не уменьшая при этом интенсивность какого-либо другого сеанса p' , у которого $r_{p'} \leq r_p$, что противоречило бы предположению о том, что r максиминно справедлив. Также отметим, что если это свойство удовлетворяется для всех сеансов для какого-либо вектора интенсивностей r , то r обязательно будет максиминно справедливым. Чтобы в этом убедиться, заметим, что если какая-то r_p увеличилась, то по крайней мере один сеанс, проходящий по узкой дуге пути p , должен обязательно уменьшить свой поток. На рис. 6.22 представлен пример максиминно справедливого вектора интенсивностей и там же иллюстрируется понятие узкой дуги.

Ниже представлен простой алгоритм вычисления максиминно справедливого вектора интенсивностей. Идея этого алгоритма состоит в том, чтобы начинать с вектора интенсивностей, все компоненты которого равны нулю, и одновременно увеличивать интенсивности всех путей до тех пор, пока F_a не станет равным C_a для какой-то одной дуги или нескольких дуг a . В этот момент каждый сеанс, использующий насыщенную дугу (т. е. дугу, у которой $F_a = C_a$), имеет ту же интенсивность, что и любой другой сеанс, использующий эту дугу. Таким образом, эти насыщенные дуги станут узкими дугами для всех сеансов, которые их используют.

На следующем шаге алгоритма все сеансы, которые не используют эти насыщенные дуги, одинаково увеличивают свои интенсивности до тех пор, пока не насытится какая-либо одна или не-

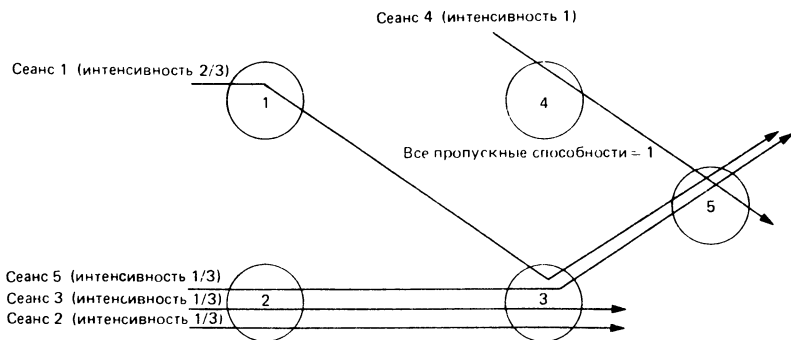


Рис. 6.22. Максимальное справедливое решение для сети из примера. Узкими дугами сеансов 1, 2, 3, 4 и 5 являются (3, 5), (2, 3), (2, 3), (4, 5) и (2, 3) соответственно. Дуга (3, 5) не является узкой дугой для сеанса 5, так как сеансы 1 и 5 совместно используют эту дугу и сеанс 1 имеет интенсивность большую, чем сеанс 5. Дуга (1, 3) не является узкой дугой никакого сеанса, так как она имеет лишнюю пропускную способность $1/3$ при таком справедливом решении.

сколько новых дуг. Заметим, что некоторые сеансы, которые используют насыщенные на предыдущем шаге дуги, также могут использовать и новые насыщенные дуги, но все остальные сеансы, использующие эти новые насыщенные дуги, будут иметь одинаковую интенсивность; эти новые насыщенные дуги станут узкими дугами для сеансов, которые не используют дуги, насыщенные на предыдущем шаге. Алгоритм продолжается шаг за шагом, всегда одинаково увеличивая интенсивности всех тех сеансов, которые не проходят по какой-либо насыщенной дуге; когда все сеансы станут проходить по крайней мере по одной насыщенной дуге, алгоритм остановится.

В алгоритме, который ниже сформулирован более точно, A^k означает множество дуг, которые не являются насыщенными к началу шага k . P^k означает множество сеансов, не проходящих по какой-либо насыщенной дуге перед началом шага k . Величина n_a^k означает число сеансов из P^k , которые используют дугу a . Заметим, что это число таких сеансов, которые будут вместе использовать еще не израсходованную пропускную способность дуги a . Наконец, \tilde{r}^k — величина приращения интенсивности, добавляемая ко всем сеансам из P^k в течение k -го шага.

Начальные условия: $k = 1$, $F_a^0 = 0$, $r_p^0 = 0$, $P^1 = P$ и $A^1 = \mathcal{A}$.

1. n_a^k : = число путей $p \in P^k$, у которых $\delta_p(a) = 1$.

2. \tilde{r}^k : = $\min_a (C_a - F_a^{k-1}) / n_a^k$.

3. r_p^k : = $\begin{cases} r_p^{k-1} + \tilde{r}^k & \text{для } p \in P^k, \\ r_p^{k-1} & \text{в противном случае.} \end{cases}$

$$4. F_a^k := \sum_{p \in P} \delta_p(a) r_p^k.$$

$$5. A^{k+1} := \{a \mid C_a - F_a^k > 0\}.$$

$$6. P^{k+1} := \{p \mid \delta_p(a) = 0, \text{ для всех } a \in A^{k+1}\}.$$

$$7. k := k + 1.$$

8. Если P^k пусто, то стоп; в противном случае перейти к 1.

На каждом шаге k одинаковые приращения добавляются к интенсивностям всех сеансов, еще не проходящих ни по одной насыщенной дуге и, таким образом, все сеансы из P^k имеют одинаковую интенсивность. Все сеансы из P^k , проходящие по дуге, которая становится насыщенной на шаге k , имеют по крайней мере не меньшую интенсивность, чем любой другой сеанс на этой дуге, и, следовательно, для них эта дуга будет узкой. Таким образом, должно быть ясно, что в результате получится максиминно справедливый вектор интенсивностей.

Пример 3

Рассмотрим задачу нахождения максиминно справедливого вектора интенсивностей для пяти сеансов и сети, показанной на рис. 6. 22. Пропускная способность каждой дуги равна единице.

Шаг 1. Все сеансы получают интенсивность, равную $1/3$. Дуга (2, 3) насыщается на этом шаге и интенсивности трех сеансов (2, 3 и 5), проходящих по ней, фиксируются на уровне $1/3$.

Шаг 2. Сеансы 1 и 4 получают дополнительное приращение интенсивности, равное $1/3$, в результате чего их интенсивности становятся равными $2/3$. Дуга (3, 5) насыщается и интенсивность сеанса 1 фиксируется на уровне $2/3$.

Шаг 3. Сеанс 4 получает дополнительное приращение интенсивности, равное $1/3$, в результате чего его интенсивность становится равной 1. Дуга (4, 5) насыщается и интенсивность сеанса 4 фиксируется на уровне 1. Так как теперь все сеансы проходят по крайней мере по одной насыщенной дуге, работа алгоритма заканчивается; окончательное максиминно справедливое решение показано на рис. 6.22.

Можно сделать несколько обобщений описанного выше основного подхода. Чтобы удержать поток на каждой дуге строго ниже пропускной способности, можно заменить C_a в алгоритме на какую-либо фиксированную часть C_a . Кроме того, можно разным трафикам приписать разные приоритеты и сделать эти приоритеты зависящими от интенсивностей трафиков. Если $b_p(r_p)$ является возрастающей функцией, означающей значение приоритета сеанса p при интенсивности r_p , то критерий максиминной справедливости можно видоизменить следующим образом: для каждого p максимизировать r_p при ограничении, что любое увеличение r_p должно вызывать уменьшение $r_{p'}$ для какого-либо p' , для которого удовлетворяется неравенство $b_{p'}(r_{p'}) \leq b_p(r_p)$. Легко видоизменить вышеописанный алгоритм для того, чтобы можно было вычислять справедливые интенсивности с такими приоритетами. Еще одно требование на эту тему состоит в том, чтобы каждая r_p

была ограничена сверху величиной $C_a - F_a$ на каждой дуге, по которой проходит путь p . Смысл этого заключается в том, чтобы всегда иметь в резерве достаточно пропускной способности, чтобы можно было добавить еще один сеанс. Проблема, однако, здесь состоит в том, что, по мере того как число сеансов, проходящих по дуге, растет, остающаяся в резерве пропускная способность падает до нуля и в результате очередь с ростом числа сеансов увеличивается точно так же, как в случае с оконным управлением. Эту трудность можно исключить, если заменить ограничение $r_p \leq C_a - F_a$ на ограничение вида $r_p \leq (C_a - F_a) q_a$, где положительный скалярный множитель q_a зависит от числа сеансов, проходящих по дуге a (см. задачу 6.16).

Распределенным алгоритмам, которые динамически регулируют интенсивности сеансов для того, чтобы соблюдалась максиминная справедливость при изменении сеансов, посвящено много работ. Типичный алгоритм [114] поможет нам понять ситуацию. В этом алгоритме v_a^k означает оценку максимально допустимой интенсивности сеанса, проходящего по дуге a на k -й итерации, F_a^k — суммарный поток по дуге a , соответствующий интенсивностям r_p^k , а n_a — число сеансов, проходящих по дуге a . Типичной итерацией алгоритма является

$$v_a^{k+1} = v_a^k + (C_a - F_a^k)/n_a,$$

$$r_p^{k+1} = \min_{\substack{\text{по дугам } a \\ \text{на пути } p}} v_a^{k+1}.$$

Каждую итерацию можно реализовать распределенно путем передачи сначала значений v_a^{k+1} от дуг a к сеансам, использующим эти дуги, а затем значений r_p^{k+1} от сеансов p ко всем дугам, по которым эти сеансы проходят. Главная проблема здесь состоит в том, что выбранные интенсивности, прежде чем начать сходиться, могут блуждать и долго находиться вдали от оптимального решения, а, кроме того, потоки по дугам иногда временно могут превышать пропускные способности. Существуют алгоритмы, которые не имеют последнего недостатка [81, 179].

Недавно был исследован другой подход к реализации максиминно справедливых интенсивностей, при котором исключается как проблема, связанная с передачей сообщений в распределенном алгоритме, так и проблема, связанная с неопределенностью в трактовании интенсивности интерактивного трафика [106]. Идея очень проста — на каждой дуге сети обслуживать разные сеансы методом кругового опроса. Это означает, что *если* сеанс всегда имеет пакет, ожидающий в узле, когда подходит его очередь на передачу по уходящей дуге, то он получает столько же обслуживания, сколько любой другой сеанс, использующий эту дугу.

Для достижения максимальной справедливости единственно, что необходимо, так это то, чтобы каждый сеанс всегда имел ожидающий пакет в очереди на передачу по своей узкой дуге. Можно показать, что в действительности при оконном паузовом управлении с достаточно большими окнами сеанс, у которого всегда есть что послать, будет всегда иметь пакет в очереди на передачу по своей узкой дуге (см. [106] и [107]).

6.4.3. Установление входных интенсивностей при изменяющихся условиях

Главная трудность при использовании способов управления потоком, основанных на регулировании входной интенсивности, состоит в том, что они не могут хорошо справляться с быстрыми изменениями предлагаемой нагрузки различных сеансов. Точной реализацией интенсивности сеанса, равной r пакетов/с, была бы передача 1 пакета каждые $1/r$ секунд. Это, однако, равносильно разнородности временного уплотнения и может привести к недопустимым задержкам, когда предлагаемая нагрузка сеансов имеет пульсирующий характер (т. е. когда малые временные интервалы с очень большой интенсивностью чередуются с большими временными интервалами полного молчания). Более подходящей реализацией является передача сразу N пакетов ($N > 1$) каждые N/r секунд. В случае пульсирующего трафика это позволяет сразу N пакетам войти в сеть без задержки и лучше подходит для динамически меняющейся нагрузки. Существует несколько вариантов этой схемы, включая следующую, которая сделана по образцу оконного управления потоком.

Право на передачу N пакетов (окно) предоставляется каждому сеансу и в узле-отправителе каждого сеанса имеется счетчик x неиспользованной части этого окна. Пакеты сеанса допускаются в сеть, пока $x > 0$. Каждый раз, когда пакет допускается в сеть, значение счетчика уменьшается на 1, а спустя N/r секунд (r — интенсивность разрешения сеансу) значение счетчика увеличивается на 1, как показано на рис. 6.23. Эта схема, называемая *управление потоком с временным окном*, очень похожа на оконное управление потоком с размером окна N за исключением того, что счетчик восстанавливается спустя N/r секунд после ввода пакета, а не спустя время передачи в оба конца, необходимое для возвращения соответствующего разрешения. (При другом варианте сначала сеансу предоставляется право на передачу N пакетов и затем счетчик устанавливается обратно на N каждые N/r секунд независимо от того, использовал ли сеанс какую-то часть окна или нет.) Таким образом, эта схема полагается на некоторый алгоритм управления потоком при определении интенсивности r и, следовательно, длины интервала, через который счетчик вос-

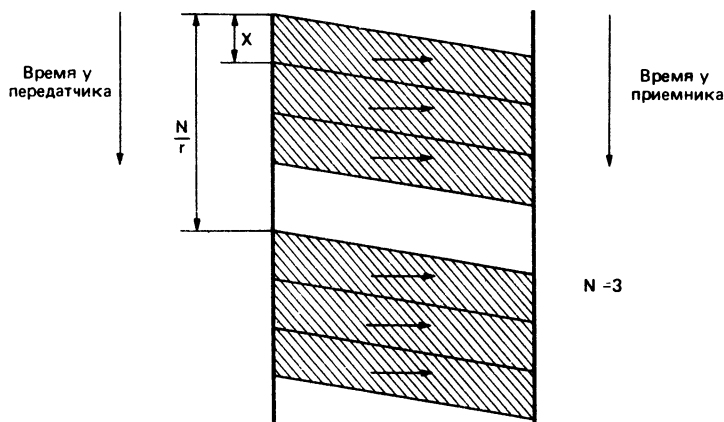


Рис. 6.23. Управление потоком с временным окном при $N = 3$. Значение счетчика пакетов, которые могут быть допущены в сеть, уменьшается, когда пакет передается, и увеличивается спустя N/r секунд.

становившаяся после ввода пакета, тогда как при обычном оконном управлении потоком этот интервал определяется в основном состоянием трафика вдоль пути сеанса. Очевидно, что обычное оконное управление потоком реагирует быстрее на перегрузки и поэтому оно лучше подходит в ситуациях, когда предлагаемая нагрузка меняется часто. В других случаях непосредственная регулировка интенсивности каким-нибудь методом, таким, например, что был описан выше, имеет преимущества более точного контроля задержки и (или) более справедливого распределения пропускной способности.

Альтернативой попытки непосредственного регулирования входной интенсивности сеанса является попытка *определения такого размера окна от конца до конца, который приведет к желаемой интенсивности*. В такой схеме желаемые входные интенсивности вычисляются посредством какого-нибудь алгоритма (подобного описанному выше) и реализуются путем выбора подходящего размера окна от конца до конца. Простейший способ состоит в вычислении размера окна W для сеанса по формуле (ср. с теоремой Литтла)

$$W = rd, \quad (6.11)$$

где

- r — желаемая входная интенсивность (в пакетах/с) для данного сеанса (вычисленная с помощью какого-нибудь алгоритма);
- d — оценка среднего времени передачи в оба конца, отнесенного к пакету этого сеанса.

Одна из трудностей с этой схемой состоит в том, что входящее в (6.11) время передачи в оба конца d не известно в момент вы-

числения нового размера окна W . Поэтому его приходится как-то оценивать, например посредством прошлых измерений, анализа или как того, так и другого. Несмотря на эту трудность, по-видимому, схемы такого типа должны хорошо работать на практике, как свидетельствуют результаты моделирования из работы [229].

6.5. Заключение

В этой главе мы определили главные цели управления потоком как поддержание внутри подсети относительно малой средней задержки и справедливости по отношению ко всем пользователям. Мы сделали обзор основных методов управления потоком и увидели, что на практике преобладают стратегии, основанные на окнах. Для этого имеются основательные причины, так как оконные стратегии сочетают низкую избыточность в нагрузке и быструю реакцию на перегрузки. Однако оконные стратегии имеют также серьезные недостатки, в частности когда реализуются схемы с фиксированным размером окна. Были описаны две схемы регулирования входной интенсивности, которые пытаются избавиться от этих недостатков. Первая схема обобщает методы оптимальной маршрутизации из гл. 5 на управление потоком и объединяет маршрутизацию и управление потоком в одном алгоритме. Во второй схеме предполагается, что применяется маршрутизация сеанса по одному фиксированному пути и внимание сосредотачивается на соблюдении справедливости при управлении потоком.

6.6. Замечания, источники и дополнительная литература

Раздел 6.1. Широкое обсуждение управления потоком можно найти в апрельском специальном номере журнала *IEEE Transactions on Communications* за 1981 год. Информационный обзор имеется в [92].

Раздел 6.2. Специфические трудности, связанные с оконным управлением потоком на спутниковых линиях, обсуждаются в работе [100].

Раздел 6.3. Управление потоком в сети ARPANET описано в нескольких источниках, например в [144] и [147]. Система управления потоком TYMNET рассмотрена в [246]. Более подробные сведения о SNA содержатся в работах [3] и [94], а о сети Codex — в [18].

Раздел 6.4. Задача, объединяющая оптимальную маршрутизацию и управление потоком, сформулирована в [84] и [97]. Дополнительный материал по этому предмету можно найти в [83] и [125]. Результаты моделирования вместе с обсуждением схем практической реализации представлены в [229]. Управление

поток, основанное на регулировании скорости кодирования цифровой речи, рассмотрено в [30]. Материал по справедливому управлению потоком основан на результатах [114]. Для ознакомления с близкими по теме работами см. [81, 106, 107, 128, 179].

Задачи

- 6.1. Рассмотрите виртуальную цепь с оконным управлением потоком, проходящую по спутниковой линии. Все пакеты имеют одинаковое время передачи, равное 5 мс. Задержка на обработку и распространение в оба конца составляет 0,5 с. Найдите нижнюю границу для размера окна этой виртуальной цепи, при котором можно было бы достигнуть максимальной скорости передачи при отсутствии другого трафика, проходящего по этой линии.
- 6.2. Предположим, что виртуальная цепь из задачи 6.1 проходит по наземной линии в дополнение к спутниковой линии. Время передачи по наземной линии равно 20 мс, а задержка на обработку и распространение пренебрежимо мала. Чему равна максимальная интенсивность передачи в пакетах/с, которую можно достигнуть в этой виртуальной цепи, если не будет управления потоком? Найдите нижнюю границу размера окна от конца до конца, которое позволит вести передачу с максимальной скоростью, если никакого другого трафика на линиях не будет. Есть ли разница между случаями, когда наземная линия будет стоять перед спутниковой линией и после нее?
- 6.3. Предположим, что используется оконное пузловое управление в системе из двух линий из задачи 6.2. Найдите нижние границы размеров окон каждой линии, необходимые для того, чтобы можно было достигнуть максимальной скорости передачи, если никакого другого трафика на линиях нет.
- 6.4. По трехузловой сети из рис. 6.24 проходит только одна виртуальная цепь от узла 1 к узлу 3 и используется оконное пузловое управление. Передача каждого пакета по линии (1, 2) занимает одну секунду, а по линии (2, 3) — две секунды; время на обработку и распространение пренебрежимо мало. Разрешения требуют одну секунду для прохождения по каждой линии. В узле 1 имеются неиспользуемые запасы пакетов. Система начинает работу в момент 0, когда имеется разрешение в узле 1, W разрешений в узле 2; очереди пакетов в узлах 2 и 3 пусты. Для $W = 1$ найдите моменты времени между 0 и 10 секундами, в которые начинается передача пакетов в узле 1 и в узле 2. Повторите то же самое для $W = 2$.
- 6.5. При обсуждении оконного пузлового управления потоком предполагалось, что узел i может посылать разрешение обратно своему предшественнику ($i - 1$), как только передаст пакет модулю УЛПД линии ($i, i + 1$). Альтернативой для узла i является возвращение разрешения, когда он получает подтверждение от УЛПД, что пакет правильно принят узлом ($i + 1$). Обсудите относительные преимущества этих двух схем. Какая схема требует больше памяти? Что произойдет, если линия ($i, i + 1$) будет спутниковой?
- 6.6. Недавно предложен другой подход к управлению потоком, который действует следующим образом. Для каждой пары отправитель — адресат (i, j) имеется соответствующий пуассоновский процесс интенсивности λ_{ij} , ко-



Рис. 6.24.

торый доставляет разрешения для пары (i, j) в узел i . Эти разрешения выстраиваются в очередь ограниченного размера для данной пары (i, j) (скажем, длина очереди не больше W разрешений). Когда пакет для узла j поступает извне в узел i сети, разрешение удаляется из очереди и пакет входит в сеть; если очередь разрешений пуста, то пакет не принимается в сеть. Если разрешение для (i, j) прибывает в момент, когда очередь полна, то это разрешение выбрасывается. Обсудите недостатки такой схемы.

- 6.7. Рассмотрите комбинированную задачу управления потоком и оптимальной маршрутизации для сети, изображенной на рис. 6.25 (см. подразд. 6.4.1). Стоимостной функцией является

$$(F_{13})^2 + (F_{23})^2 + (F_{34})^2 + a/r_1 + a/r_2,$$

где a — некоторый положительный скалярный параметр. Найдите оптимальные значения интенсивностей r_1 и r_2 для каждого значения a .

- 6.8. Рассмотрите шесть узлов, образующих кольцо и соединенных линиями $(i, i+1)$, $i = 1, 2, 3, 4, 5$ и $(6, 1)$, имеющими единичную пропускную способность и передачу по которым можно вести в обоих направлениях. Для каждого из узлов 1, 2, 3, 4 и 5 имеется два сеанса к узлу 6, один по часовой стрелке, а другой в направлении против часовой стрелки. Аналогично имеются по два сеанса от узлов 2, 3 и 4 к узлу 5 и два сеанса от узла 3 к узлу 4. Найдите максимально справедливые интенсивности для этих сеансов.

- 6.9. Предположим, что определение вектора справедливых интенсивностей из подразд. 6.4.2 видоизменено таким образом, что в дополнение к (6.10) имеется еще одно ограничение $r_p \leq b_p$, которому должна удовлетворять интенсивность сеанса p . Здесь b_p является максимальной интенсивностью, с которой сеанс p способен вести передачу. Видоизмените представленный в подразд. 6.4.2 алгоритм таким образом, чтобы можно было решить эту задачу. Найдите справедливые интенсивности для примера из подразд. 6.4.2, когда $b_p = 1$ для $p = 2, 4, 5$, и $b_p = 1/4$ для $p = 1, 3$. Указание: добавьте новую линию для каждого сеанса.

- 6.10. Цель этой задачи состоит в том, чтобы проиллюстрировать, каким образом на пропускные способности конкурирующих сеансов может влиять правило приоритетов, используемое для их обслуживания. Рассмотрите два сеанса A и B , совместно использующих первую линию L на их пути, как показано на рис. 6.26. Каждый сеанс имеет окно от конца до конца для двух пакетов. Разрешения для пакетов из A и B возвращаются спустя d_A и d_B секунд соответственно после завершения передачи по линии L . Предположим, что d_A является экспоненциально распределенной случайной величиной со средним, равным единице, тогда как $d_B = 0$ (это несколько нереалистично). Время передачи пакетов по L экспоненциально распределено со средним, равным единице. Все времена передачи пакетов и задержки разрешений являются независимыми случайными величинами. Предположим, что новый пакет из A (B) поступает в очередь на передачу по линии L сразу после того, как приходит разрешение для A (B).

а) Предположим, что пакеты передаются по L в порядке «первым пришел — первым обслуживается». Покажите, что очередь у L можно описать цепью Маркова с 10 состояниями: $BB, BBA, BAB, ABB, BBAA, BABA, BAAB, ABBA, ABAB, AABV$ (каждая буква означает пакет соответствующего сеанса). Покажите, что все состояния имеют одинаковые стационарные вероятности и стационарные пропускные способности сеансов A и B в пакетах/с равны 0,4 и 0,6 соответственно.

- б) Предположим теперь, что очередность на передачу по линии L устанавливается на основе кругового опроса. Между последовательными передачами пакетов сеанса B сеанс A передает один пакет, если у него есть хотя бы один пакет в очереди. Нарисуйте диаграмму переходов из состояния в состояние для цепи Маркова с пятью состояниями, которая моделирует очередь у L . Найдите стационарные вероятности.

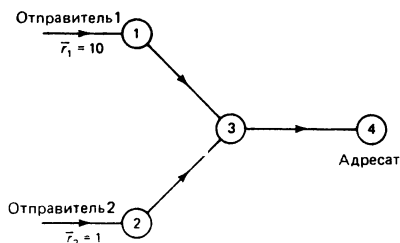


Рис. 6.25.

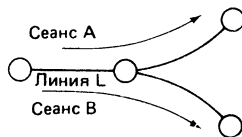


Рис. 6.26.

Чему равны стационарные пропускные способности сеансов A и B ?

- в) Наконец, предположим, что сеанс A имеет неоспоримый приоритет над сеансом B на передачу по линии L . Между последовательными передачами пакетов сеанса B сеанс A передает столько пакетов, сколько может. Сеанс B только тогда получает доступ к линии, когда A нечего посылать. Нарисуйте диаграмму переходов из состояния в состояние для цепи Маркова с пятью состояниями, которая моделирует очередь у L . Найдите стационарные вероятности состояний. Чему равны стационарные пропускные способности сеансов A и B ?

- 6.11. Рассмотрите оконное управление потоком между внешним источником и узлом, через который этот источник подсоединяется к подсети. Источник генерирует пакеты в соответствии с пуассоновским процессом интенсивности λ . Каждый новый пакет принимается узлом, если имеется разрешение. Если нет разрешения, то пакет сбрасывается, никогда после этого не возвращаясь. Как только пакет данного источника входит в модуль УЛПД для передачи по уходящей от этого узла линии, новое разрешение сразу отсылается обратно источнику. Первоначально у источника имеются два разрешения и размер окна равен 2. Предположим, что пакет в узле испытывает случайную задержку начиная с момента, когда он принимается узлом, и кончая моментом, когда он входит в модуль УЛПД. Точнее говоря, предположим, что в любой бесконечно малый интервал δ существует вероятность $\mu\delta$ того, что ожидающий пакет (или первые два ожидающих пакета) перейдет из источника в модуль УЛПД; это событие не зависит от других случайностей в системе.

- Опишите цепь Маркова для числа разрешений в источнике.
 - Найдите вероятность того, что пакет, порожденный источником, сбросится.
 - Внесите, увеличится или уменьшится вероятность в п. б), если учесть время на распространение и передачу от источника к узлу и обратно.
 - Предположим, что в источнике имеется буфер размера k для хранения пакетов, для которых не оказалось разрешений; когда приходит разрешение, один из хранящихся в буфере пакетов сразу же отсылается в узел сети. Найдите новую цепь Маркова, описывающую эту систему, и вероятность того, что новый пакет застанет буфер полным.
- 6.12. Рассмотрите сеть с оконным управлением потоком от конца до конца, применяемым к каждой виртуальной цепи. Предположим, что управление линией передачи данных выполняется идеально и пакеты внутри сети никогда не сбрасываются; таким образом, пакеты всегда прибывают в узел-адресат в том порядке, в каком они были переданы, и все пакеты обязательно прибывают туда.
- Предположим, что адресат отсылает разрешения в пакетах, возвращающихся к источнику; если в течение некоторого тайм-аута возвращающихся пакетов нет, то специальный пакет с разрешением отсылается обратно источнику. Эти разрешения представляют собой номер по мо

дулю t следующего пакета, ожидаемого адресатом. Что представляет собой ограничение на размер окна W по модулю m ?

- б) Теперь предположим, что разрешения представляют собой номер по модулю m каждого из пакетов, полученных после того, как было отправлено последнее подтверждение. Изменит ли это ваш ответ на п. а)? Дайте объяснение.
- в) Допустимо ли источнику изменять размер окна W без предварительного соглашения с адресатом? Дайте объяснение.
- г) Каким образом адресат может сделать действительный размер окна меньше размера окна, используемого источником, без предварительного соглашения с источником? (Под действительным размером окна мы подразумеваем максимальное число пакетов для пары источник—адресат, которые могут одновременно находиться в сети.)

6.13. Рассмотрите оконную поузловую схему. Намереваясь минимизировать необходимый размер буфера, проектировщики связывают окна не с виртуальными цепями, а с адресатами. Предположим, что все пути виртуальных цепей до данного адресата j используют ориентированное остовное дерево, так что каждый узел $i \neq j$ имеет только одну уходящую дугу для трафика, направленного к этому адресату. Пусть вначале каждый узел $i \neq j$ имеет два разрешения для пакетов, которые могут быть посланы по уходящей дуге к j . Каждый раз, когда узел i передает пакет для j в свой модуль УЛПД на уходящей линии, он отправляет новое разрешение для одного пакета обратно по той приходящей линии, по которой этот пакет пришел. Если этот пакет пришел от источника, подключенного к i , разрешение отправляется обратно источнику (каждый источник также вначале имеет два разрешения для данного адресата).

а) Какого размера буфер в узле i необходимо зарезервировать для пакетов, направляющихся к адресату j , для гарантии того, что каждый приходящий для j пакет можно будет поместить в буфер?

б) В чем состоят плюсы и минусы этой схемы по сравнению с обычной оконной поузловой схемой для виртуальных цепей?

6.14. Рассмотрите сеть, использующую окна поузловым способом для каждой виртуальной цепи. Опишите стратегию рассылки и использования разрешений, такую, чтобы переполнения буферов никогда не происходили независимо от того, какой объем памяти имеется в распоряжении для хранения пакетов в каждом узле и сколько виртуальных цепей используют каждую линию. *Указание:* вам следует побеспокоиться о том, что слишком много свободных разрешений может скопиться на передающем конце каждой линии.

6.15. Опишите, каким образом проекционный градиентный метод оптимальной маршрутизации может быть использован для решения распределенным способом объединенной задачи управления потоком и оптимальной маршрутизации из подразд. 6.4.1.

6.16. Другая формулировка максимно справедливого управления потоком [80, 81, 128]. Рассмотрите задачу максимно справедливого управления потоком, в которой требуется, чтобы вектор интенсивности дополнительно удовлетворял ограничению $r_p \leq (C_a - F_a) q_a$ для каждого сеанса p и дуги a , по которой проходит сеанс p . Здесь q_a — данная положительная числовая величина.

а) Покажите, что максимно справедливый вектор интенсивностей существует и единствен, и дайте алгоритм для его вычисления.

б) Покажите, что для максимно справедливого вектора интенсивностей коэффициент использования $\rho_a = F_a / C_a$ для каждой дуги a удовлетворяет неравенству

$$\rho_a \leq \frac{n_a q_a}{1 + n_a q_a},$$

где n_a — число сеансов, проходящих по дуге a .

ЛИТЕРАТУРА

1. Aashtiani H. Z., Magnanti T. L., Equilibria on a Congested Transportation Network, *SIAM J. Algeb Disc Math*, 2, 213—226, 1981.
2. Abramson N., The Aloha System — An other Alternative for Computer Communications, *Proc. Fall Joint Comput Conf, AFIPS Conf*, 37, 1970.
3. Ahuja V., Routing and Flow Control in Systems Network Architecture, *IBM Syst J*, 18, 298—314, 1979.
4. Akinc U., Khumawala B., An Efficient Branch and Bound Algorithm for the Capacitated Warehouse Location Problem, *Manage Sci*, 23, 585—594, 1977.
5. Aldous D., Ultimate Instability of Exponential Back-off Protocol for Acknowledgement-Based Transmission Control of Random Access Communication Channels, Berkeley, CA, University of California, Dept. of Statistics, 1986.
6. Alcouffe A., Muratet G., Optimum Location of Plants, *Manage Sci*, 23, 267—274, 1976.
7. Altes T., Minimum Delay Packet Length (Report LIDS-TH-1602), Cambridge, MA, MIT Laboratory for Information and Decision Systems, 1986.
8. Anagnostou M., Protonotarios E., Performance Analysis of the Selective Repeat ARQ Protocol, *IEEE Trans Commun*, COM-34, 2, 127—135.
9. Arian E., Some Complexity Results About Packet Radio Networks, *IEEE Trans Inf Theory*, 681—685, 1984.
10. Atkins J. D., Path Control, The Transport Network of SNA, *IEEE Trans Commun*, COM-28, 527—538.
11. Ball M. O., Computing Network Reliability, *Oper Res*, 27, 823—838, 1979.
12. Ball M. O., Provan J. S., Calculating Bounds on Reachability and Connectedness in Stochastic Networks, *Networks*, 13, 253—278, 1983.
13. Baratz L., Segall A., *Reliable Link Initialization Procedures* (Report RC10032), IBM Thomas J. Watson Research Center, 1983.
14. Basket F., Chandy K. M., Muntz R. R., Palacios F. G., Open, Closed, and Mixed Networks of Queues with Different Classes of Customers, *J. Assoc. Comput. Mach.*, 22, 248—260, 1975.
15. Bertsekas D. P., Gafni E., Projection Methods for Variational Inequalities with Application to the Traffic Assignment Problem. In D. C. Sorensen and R. J.-B. Wets (Eds.), *Math Prog Study* (Vol. 17, pp. 139—159), Amsterdam, North-Holland, 1982.
16. Bertsekas D. P., Gafni E. M., Projected Newton Methods and Optimization of Multicommodity Flows, *IEEE Trans Auto Contr*, AC-28, 12, 1090—1096, 1983.
17. Benjamin R., Analysis of Connection Survivability in Complex Strategic Communications Networks, *IEEE J Select Areas Commun*, SAC-4, 243—253, 1986.
18. Bertsekas D. P., On the Goldstein—Levitin—Polyak Gradient Projection Method, *IEEE Trans Auto Contr*, AC-21, 174—184, 1976.
19. Bertsekas D. P., Algorithms for Nonlinear Multicommodity Network Flow Problems. In A. Bensoussan and K. L. Lions (Eds.), *International Symposium on Systems Optimization and Analysis* (pp. 210—224), New York, Springer-Verlag, 1979.

20. Bertsekas D. P., Dynamic Models of Shortest Path Routing Algorithms for Communication Networks with Multiple Destinations, *Proc 1979 IEEE Conf Dec Contr* (Ft. Lauderdale, FL), 127—133, 1979.
21. Bertsekas D. P., A Class of Optimal Routing Algorithms for Communication Networks, *Proc 5th Int Conf Comput Commun* (Atlanta, GA), 71—76, 1980, October.
22. Bertsekas D. P., Distributed Dynamic Programming, *IEEE Trans Auto Contr*, AC-27, 610—616, 1982.
23. Bertsekas D. P., Dynamic Behavior of Shortest Path Routing Algorithms for Communication Networks, *IEEE Trans Auto Contr*, AC-27, 60—74, 1982.
24. Bertsekas D. P., Projected Newton Methods for Optimization Problems with Simple Constraints, *SIAM J Contr Optim*, 20, 221—246, 1982.
25. Bertsekas D. P., Constrained Optimization and Lagrange Multiplier Methods, New York, Academic Press, 1982.
26. Bertsekas D. P., Distributed Asynchronous Computation of Fixed Points, *Math Prog*, 27, 107—120, 1983.
27. Bertsekas D. P., A Unified Framework for Primal-Dual Methods in Minimum Cost Network Flow Problems, *Math Prog*, 32, 125—145, 1985.
28. Bertsekas D. P., Dynamic Programming: Deterministic and Stochastic Models. Englewood Cliffs, NJ, Prentice-Hall, 1987.
29. Bertsekas D. P., Gafni E. M., Gallager R. G., Second Derivative Algorithms for Minimum Delay Distributed Routing in Networks, *IEEE Trans Commun*, COM-32, 911—919, 1984.
30. Bially T., Gold B., Seneff S., A Technique for Adaptive Voice Flow Control in Integrated Packet Networks, *IEEE Trans Commun*, COM-28, 325—333, 1980.
31. Bertsekas D. P., Gendron R., Tsai W. K., Implementation of an Optimal Multicommodity Network Flow Algorithm Based on Gradient Projection and a Path Flow Formulation (Report LIDS-P-1364. Cambridge, MA, MIT Laboratory for Information and Decision Systems) 1984.
32. Bertsekas D. P., Gafni E. M., Vastola K. S., Validation of Algorithms for Optimal Routing of Flow in Networks, *Proc 1978 IEEE Conf Dec Contr* (San Diego), 1979.
33. Binder R., A Dynamic Packet Switching System for Satellite Broadcast Channels, *Proc ICC*, 41.1—41.5, 1975.
34. Blahut R. E., Theory and Practice of Error Control Codes, Reading, MA, Addison-Wesley, 1983. [Имеется перевод: Блейхут Р. Э. Теория и практика кодов; контролирующих ошибки. — М.: Мир, 1986.]
35. Boorstyn R. R., Frank H., Large-Scale Network Topological Optimization, *IEEE Trans Commun*, COM-25, 29—47, 1977.
36. Bochmann G. V., Sunshine C. A., A Survey of Formal Methods, In P. Green (Ed.), *Computer Network Architecture*, New York, Plenum, 1982.
37. Bruell S. C., Balbo G., Computational Algorithms for Closed Queueing Networks, New York, Elsevier North-Holland, 1980.
38. Bertsekas D. P., Tsitsiklis J. N., Athans M., Convergence Theories of Distributed Iterative Processes (Report LIDS-P1412), Cambridge, MA, MIT Laboratory for Information and Decision Systems, 1984.
39. Bux W., Local Area Networks: A Performance Comparison, *IEEE Trans Commun*, Com. 29, 1465—1473, 1981.
40. Cannon M. D., Cullum C. D., A Tight Upper Bound on the Rate of Convergence of the Frank-Wolfe Algorithm, *SIAM J Contr Optim*, 6, 509—516, 1968.
41. Carleial A. B., Hellman M. E., Bistable Behavior of Slotted Aloha-Type Systems, *IEEE Trans Commun*, Com. 23, 401—410, 1975.
42. Capetanakis J. I., The Multiple Access Broadcast Channel: Protocol and Capacity Considerations, PhD Diss., MIT, Dept. of Electrical Engineering and Computer Science, Also in *IEEE Trans Info Theory*, September 1979, IT-25, 505—515, 1977.

43. Carlson D. E., Bit-Oriented Data Link Control Procedures, *IEEE Trans Commun*, 455—467, 1980.
44. Chlamtac, Franta W., Levin K. D., BRAM: The Broadcast Recognizing Access Method, *IEEE Trans Commun*, COM-27, 1183—1190, 1979.
45. Cornuejols G., Fisher M. L., Nemhauser G. L., Location of Bank Accounts to Optimize Float, An Analytic Study of Exact and Approximate Algorithms, *Manage Sci*, 23, 789—810, 1977.
46. Chandy K. M., Misra J., Distributed Computation on Graphs, *Commun ACM*, 25, 833—837, 1982.
47. Clark G. C., Cain J. B., Error Correction Coding for Digital Communication, New York, Plenum, 1981. [Имеется перевод: Кларк Д., Кейн Д., Кодирование с исправлением ошибок в системах цифровой связи. — М.: Радио и связь, 1987.]
48. Cooper R. B., Queues Served in Cyclic Order: Waiting Times, *Bell Syst Tech J*, 49, 399—413, 1970.
49. Cooper R. B., Introduction in Queueing Theory (2d ed.), New York, Elsevier, North-Holland, 1981.
50. Clark D. D., Pogram K. T., Reed D. P., An Introduction to Local Area Networks, *Proc IEEE*, 1497—1517, 1978. [Имеется перевод: Кларк Д. Д., Порген К. Т., Рид Д. П. Локальные сети. — ТИИЭР, 1987, т. 66, № 11, с. 248—272.]
51. Crowther W., Rettburg R., Walden D., Ornstein S., Heart F., A System for Broadcast Communication, Reservation Aloha, *Proc 6th Hawaii Int Conf Syst Sc*, 371—374, 1973.
52. Dafermos S. C., An Extended Traffic Assignment Model with Applications to Two-Way Traffic, *Trans Sci*, 5, 366—389, 1971.
53. Dafermos S. C., Traffic Equilibrium and Variational Inequalities, *Trans Sci*, 14, 42—54, 1980.
54. Dantzig G. B., *Linear Programming and Extensions*, Princeton, NJ, Princeton University Press, 1963. [Имеется перевод: Данциг Д. Б. Линейное программирование, его применения и обобщения. — М.: Прогресс, 1966.]
55. Dembo R. S., Kliniewicz J. G., A Scaled Reduced Gradient Algorithm for Network Flow Problems with Convex Separable Costs *Math Prog Studies*, 15, 125—147, 1981.
56. Dial R., Glover F., Karney D., Klingman D., A Computational Analysis of Alternative Algorithms and Labeling Techniques for Finding Shortest Path Trees, *Networks*, 9, 215—248, 1979.
57. Disney R. L., Konig D., Queueing Networks, A Survey of Their Random Processes, *SIAM Rev.*, 27, 335—403, 1985.
58. Dunn J. C., Rates of Convergence of Coditional Gradient Algorithms Near Singular and Nonsingular Extremals, *SIAM J Contr Optim*, 17, 187—211, 1979.
59. Eisenberg M., Two Queues with Alternating Service, *SIAM J Appl Math*, 20, 287—303, 1979.
60. Ephremides A., Baker D. J., The Architectural Organization of a Mobile Radio Network Via a Distributed Algorithm, *IEEE Trans Commun*, COM-29, 1694—1701, 1981.
61. Ephremides A., The Routing Problem in Computer Networks, In I. F. Blake and H. V. Poor (Eds.), *Communication and Networks* (pp. 299—324), New York, Springer-Verlag, 1986.
62. Erlenkotter D., A Dual-Based Procedure for Uncapacitated Facility Location, *Oper Res*, 26, 992—1009, 1978.
63. Essau L. R., Williams K. C., On Teleprocessing System Design, *IBM Syst J*, 5, 142—147, 1966.
64. Even S., An Algorithm for Determining Whether the Connectivity of a Graph is at Least k, *SIAM J Comput*, 4, 393—396, 1975.
65. Ephremides A., Varaiya P., Walrand J., A Simple Dynamic Routing Problem, *IEEE Trans Auto Contr*, AC-25, 4, 690—693, 1980.

66. Farber D. J., Larson K. C., The System Architecture of the Distributed Computer System — The Communications System, Paper presented at the Symposium on Computer Networks, Polytechnic Institute of Brooklyn, New York.
67. Farmer W. D., Newhall E. E., An Experimental Distributed Switching System to Handle Bursty Computer Traffic, *Proc ACM Symp Probl Opt Data Commun Sys*, 1—33, 1969.
68. Ferguson M. J., Aminetzah Y. J., Exact Results for Nonsymmetric Token Ring Systems, *IEEE Trans Commun*, COM-33, 223—231, 1985.
69. Ferguson M. J., A Study of Unslotted Aloha with Arbitrary Message Lengths, *Proc. 4th Data Commun Symp* (Quebec, PQ, Canada), 5.20—5.25, 1975.
70. Fratta L., Gerla M., Kleinrock L., The Flow Deviation Method, An Approach to Store-and-Forward Communication Network Design, *Networks*, 3, 97—133, 1973.
71. Fine M., Tobagi F. A., Demand Assignment Multiple Access Schemes in Broadcast Bus Local Area Networks, *IEEE Trans. Comput.*, 1130—1159, 1984.
72. Finn S. G., Resynch Procedures and a Fail-Safe Network Protocol, *IEEE Trans Commun*, COM-27, 840—845, 1979.
73. Florian M., Nguyen S., A Method for Computing Network Equilibrium with Elastic Demand, *Transport Sci*, 8, 321—332, 1974.
74. Ford L. R., Jr., Fulkerson D. R., Flows in Networks, Princeton, NJ, Princeton University Press, 1962. [Имеется перевод: Форд Л. Р., Фалкерсон Д. Р. Потоки в сетях. — М.: Мир, 1966.]
75. Foschini G. J., Salz J., A Basic Dynamic Routing Problem and Diffusion, *IEEE Trans Commun*, COM-26, 3, 320—327, 1978.
76. Frank M., Wolfe P., An Algorithm for Quadratic Programming, *Naval Res Logist Quart*, 3, 149—154, 1956.
77. Fuhrmann S. W., Cooper R. B., Stochastic Decompositions in the M/G/1 Queue with Generalized Vacations, *Oper Res*, 33, 1117—1129, 1985.
78. Gafni E. M., Bertsekas D. P., Distributed Routing Algorithms for Networks with Frequently Changing Topology, *IEEE Trans Commun*, COM-29, 11—18, 1981.
79. Gafni E. M., Bertsekas D. P., Asymptotic Optimality of Shortest Path Routing (Report LIDS-P-1307), Cambridge, MA, MIT Laboratory for Information and Decision Systems, *IEEE Trans Inf Theory*, IT-33, 83—90, 1987, January.
80. Gafni E. M., Bertsekas D. P., Two-Metric Projection Methods for Constrained Optimization, *SIAM J Contr Optim*, 22, 6, 936—964, 1984.
81. Gafni E. M., Bertsekas D. P., Dynamic Control of Session Input Rates in Communication Networks, *IEEE Trans Auto Contr*, AC-29, 1009—1016, 1984.
82. Gafni E. M., Convergence of a Routing Algorithm. MS thesis, University of Illinois, Dept. of Electrical Engineering, Urbana, IL, 1979.
83. Gafni E. M., The Integration of Routing and Flow Control for Voice and Data in Integrated Packet Networks, PhD thesis, MIT, Dept. of Electrical Engineering and Computer Science, Cambridge, MA, 1982.
84. Gallager R. G., Golestani S. J., Flow Control and Routing Algorithms for Data Networks, *Proc 5th Intern Conf Commun*, 779—784, 1980.
85. Gavish B., Hantler S., An Algorithm for Optimal Route Selection in SNA Networks, *IEEE Trans Commun*, COM-31, 1154—1161, 1983.
86. Gallager R. G., Information Theory and Reliable Communications, New York, John Wiley & Sons, 1968. [Имеется перевод: Галлагер Р. Д. Теория информации и надежная связь. — М.: Сов. радио, 1974.]
87. Gallager R. G., A Minimum Delay Routing Algorithm Using Distributed Computation, *IEEE Trans Commun*, COM-23, 73—85, 1977.
88. Gallager R. G., Conflict Resolution in Random Access Broadcast Networks, *Proc. AFOSR Workshop Commun Theory Appl* (Provincetown, MA), 74—76, 1978.
89. Gallager R. G., Applications of Information Theory for Data Communication Networks, In J. Skwirzynski (Ed.), *New Concepts in Multi-user Communication*

- (Series E, No. 43), NATO Advanced Study Institutes (Sijthoff and Noordhoff), 1981.
90. Gavish B., Augmented Lagrangean Based Algorithms for Centralized Network Design, *IEEE Trans Commun*, COM-33, 1247—1257, 1985.
 91. Gerla M., Kleinrock L., On the Topological Design of Distributed Computer Networks, *IEEE Trans Commun*, COM-25, 48—60, 1977.
 92. Gerla M., Kleinrock L., Flow Control, A Comparative Survey, *IEEE Trans Commun*, COM-28, 553—574, 1980.
 93. Georgiadis L., Papantoni-Kazakos P., A 0.487 Throughput Limited Sensing Algorithm, Storrs, Ct, University of Connecticut, 1985.
 94. George F. D., Young G. E., SNA Flow Control, Architecture and Implementation, *IBM Syst J*, 21, 179—210, 1982.
 95. Goodman J., Greenberg A. G., Madras N., March P., On the Stability of Ethernet, *Proc 17th Ann HCM Symp Theory Comput*, 1985.
 96. Gallager R. G., Humblet P. A., Spira P. M., A Distributed Algorithm for Minimum-Weight Spanning Trees, *ACM Trans Prog Lang Syst*, 5, 66—77, 1983.
 97. Golestaani S. J., A Unified Theory of Flow Control and Routing on Data Communication Networks, PhD thesis, MIT, Dept. of Electrical Engineering and Computer Science, Cambridge, MA, 1980.
 98. Gopal I. S., Prevention of Store-and-Forward Deadlock in Computer Networks, *IEEE Trans Commun*, COM-33, 1258—1264, 1985.
 99. Gray J. P., Line Control Procedures, *Proc IEEE*, 1301—1312, 1972. [Имеется перевод: Грей Д. П. Линейное управление. — ТИИЭР, т. 60, № 11, с. 44—59, 1972.]
 100. Grover G. A., Bharath-Kumar K., Windows in the Sky — Flow Control in SNA Networks with Satellite Links, *IBM Syst J*, 22, 451—463, 1983.
 101. Green P. E., Computer Network Architectures and Protocols, New York, Plenum, 1982.
 102. Green P. E., Computer Communications, Milestones and Prophecies, *IEEE Commun*, 49—63, 1984.
 103. Green P. E., Protocol Conversion, *IEEE Trans Commun*, 257—268, 1986.
 104. Gross D., Harris C. M., Fundamentals of Queueing Theory (2d ed.), New York, John Wiley & Sons.
 105. Gunther K. D., Prevention of Deadlocks in Packet-Switched Data Transport Systems, *IEEE Trans Commun*, COM-29, 512—524, 1981.
 106. Hahne E. L., Gallager R. G., Round Robin Scheduling for Fair Flow Control in Data Communication Networks (Report LIDS-P-1537), Cambridge, MA, MIT Laboratory for Information and Decisions Systems.
 107. Hahne E., PhD thesis, MIT, Dept. of Electrical Engineering and Computer Science, Cambridge, MA, 1986, forthcoming.
 108. Hajek B., Birth-and-Death Processes with Phases and General Boundaries, *J Appl Prob*, 19, 488—499, 1982.
 109. Hajek B., Cooling Schedules for Optimal Annealing (preprint), Urbana-Champaign, IL, University of Illinois, Dept. of Electrical Engineering, 1985.
 110. Hajek B., van Loon T., Decentralized Dynamic Control of a Multiaccess Broadcast Channel, *IEEE Trans Auto Contr*, AC-27, 559—569, 1982.
 111. Hajek B., Ogier R. G., Optimal Dynamic Routing in Communication Networks with Continuous Traffic, *Networks*, 14, 457—487, 1984.
 112. Hashida O., Analysis of Multiqueue, *Rev. Elect. Commun. Lab.*, 20, 189—199, 1972.
 113. Hayes J., An Adaptive Technique for Local Distribution (Bell Telephone Laboratory Technical Memo TM-76-3116-1.) (Also in *IEEE Trans Commun* (1978), COM-26, 1178—1186, 1976.)
 114. Hayden H., Voice Flow Control in Integrated Packet Networks (Report LIDS-TH-1152), Cambridge, MA, MIT Laboratory for Information and Decisions Systems, 1981.

115. Hayes J. F., Modeling and Analysis of Computer Communications Networks, New York, Plenum, 1984.
116. Heyman D. P., Sobel M. J., Stochastic Models in Operations Research, Vol. 1, New York, McGraw-Hill, 1982.
117. Hluchyj M. G., Gallager R. G., Multiaccess of a Slotted Channel by Finitely Many Users, Proc Nat Telecommun Conf (New Orleans, LA), (also LIDS Report P-1131, MIT, Cambridge, MA, August 1981), 1981.
118. Humblet P. A., Soloway S. R., Steinka B., Algorithms for Data Communication Networks — Part 2, Codex Corp, 1986.
119. Huang J.-C., Berger T., Delay Analysis of the Modified 0.487 Contention Resolution Algorithm, *IEEE Trans Inf Theory*, IT-31, 264—273, 1985.
120. Humblet P. A., Mosely J., Efficient Accessing of a Multiaccess Channel, *Proc 19th Conf Dec Contr* (Albuquerque, NM), 1980.
121. Humblet P. A., Source Coding for Communication Concentrators (Report ESL-R-798), Cambridge, MA, MIT, 1978.
122. Humblet P. A., A Distributed Algorithm for Minimum Directed Spanning Trees, *IEEE Trans Commun*, COM-31, 756—762, 1983.
123. Humblet P. A., On the Throughput of Channel Access Algorithms with Limited Sensing, *IEEE Trans Commun*, COM-34, 345—347, 1986.
124. Humblet P. A., Soloway S. R., Algorithms for Data Communication Networks — Part I, Codex Corp, 1986.
125. Ibe O. C., Flow Control and Routing in an Integrated Voice and Data Communication Network, PhD thesis, MIT, Dept. of Electrical Engineering and Computer Science, Cambridge, MA, 1981.
126. IEEE Journal on Selected Areas in Communications, Special Issue on Network Performance Evaluation, SAC-4, 6, 1986, September.
127. Jackson J. R., Networks of Waiting Lines, *Oper Res.*, 5, 518—521, 1957.
128. Jaffe J. M., A Decentralized «Optimal» Multiple-User Flow Control Algorithm, *IEEE Trans Commun*, COM-29, 954—962, 1981.
129. Jaiswal N. K., Priority Queues, New York, Academic Press, 1968.
130. Jacobs I. M., Binder R., Hoversten E. V., General Purpose Packet Satellite, Networks, *Proc. IEEE*, 1448—1468, 1978. [Имеется перевод: Джейкобс Э. М., Байндер Э., Ховерстен Э. В. — Пакетные спутниковые сети общего назначения. — ТИИЭР, т. 66, № 11, с. 186—211, 1978].
131. Kaplan M., A Sufficient Condition for Nonergodicity of a Markov Chain, *IEEE Trans Inf Theory*, IT-25, 470—471, 1979.
132. Karlin S., Taylor H. M., A First Course in Stochastic Processes, New York, Academic Press, 1975.
133. Kershbaum A., Boorstyn R. R., Centralized Teleprocessing Network Design, *Networks*, 13, 279—293, 1983.
134. Keuhn A. A., Hamburger M. J., A Heuristic Program for Locating Warehouses, *Manage Sci*, 9, 643—666, 1963.
135. Keilson J., Markov-Chain Models — Rarity and Exponentiality, New York, Springer-Verlag, 1979.
136. Kelly F. P., Reversibility and Stochastic Networks, New York, John Wiley & Sons, 1979.
137. Kelly F. P., Stochastic Models of Computer Communication Systems, *J Royal Stat Soc*, B, 47, 1, 1985.
138. Kahn R. E., Gronemeyer S. A., Burchfiel J., Kungelman R. C., Advances in Packet Radio Technology, *Proc IEEE*, 1468—1469, 1978. [Имеется перевод: Кан Р. Э., Гронемейер С. А., Берчфил Дж., Кангелман Р. С. Достижения в области пакетной радиосвязи. — ТИИЭР, т. 66, № 11, с. 212—247, 1988.]
139. Kirkpatrick S., Gelatt C. D., Jr., Vecchi M. P., Optimization by Simulated Annealing, *Science*, 220, 671—680, 1983.
140. Khumawala B. M., An Efficient Branch and Bound Algorithm for the Warehouse Location Problem, *Manage Sci*, 18 : B 718—B731, 1972.

141. Kleinrock L., Communication Nets: Stochastic Message Flow and Delay, New York, McGraw-Hill, 1964. [Имеется перевод: Клейнрок Л. Коммуникационные сети. Стохастические потоки и задержки сообщений. — М: Наука, 1970.]
142. Kleitman D., Methods for Investigating the Connectivity of Large Graphs, *IEEE Trans Circ. Theory*, CT-16, 232—233.
143. Kleinrock L., Queueing Systems, Vol. 1, New York, John Wiley & Sons.
144. Kleinrock L., Queueing Systems, Vol. 2, New York, John Wiley & Sons. [Имеется перевод: Клейнрок Л. Вычислительные системы с очередями. — М.: Мир, 1979.]
145. Klessig R. W., Overview of Metropolitan Area Networks, *IEEE Commun.*, 9—15, 1986.
146. Kleinrock L., Lam S., Packet Switching in Multiaccess Broadcast Channel, Performance Evaluation, *IEEE Trans Commun*, 410—423, 1975.
147. Kleinrock L., Opterbeck H., Throughput in the ARPANET — Protocols and Measurements, *IEEE Trans Commun*, COM-25, 95—104, 1977.
148. Kleinrock L., Scholl, Packet Switching in Radio Channels, New Conflict Free Multiple Access Schemes, *IEEE Trans Commun*, 1015—1029, 1980.
149. Kleinrock L., Tobagi F. A., Packet Switching in Radio Channels, Part 1, CSMA Modes and Their Throughput-Delay Characteristics, *IEEE Trans Commun*, COM-23, 1400—1416, 1975.
150. Konheim A. G., Meister B., Waiting Lines and Times in a System with Polling, *J ACM*, 21, 470—490, 1974.
151. Kemeny J. G., Snell J. L., Knapp A. W., Denumerable Markov Chains, New York, Springer-Verlag, 1976. [Имеется перевод: Кемени Д. Д., Снелл Д., Кнепп А. Счетные цепи Маркова. — М.: Наука, 1987.]
152. Kuehn P. J., Multiqueue Systems with Nonexhaustive Cyclic Service, *Bell Syst Tech J*, 58, 671—698, 1979.
153. Kummerle K., Reiser M., Local Area Networks — An Overview, *J. Telecommun Netw*, 1, 4, 1982.
154. Lanphongpanich S., Hearn D., Simplicial Decomposition of the Asymmetric Traffic Assignment Problems, *Trans Res*, 18B, 123—133, 1984.
155. Lam S., Kleinrock L., Packet Switching in a Multiaccess Broadcast Channel: Dynamic Control Procedures, *IEEE Trans Commun*, COM-23, 891—904, 1975.
156. Lam Y. F., Li, V. O. K., An Improved Algorithm for Performance Analysis of Networks with Unreliable Components, *IEEE Trans Commun*, COM-34, 496—497, 1986.
157. Lam S., A Carrier Sense Multiple Access Protocol for Local Networks, *Computer Networks*, 4, 21—32, 1980.
158. Lasdon L. S., Optimization Theory for Large Systems, New York, Macmillan, 1970.
159. Lawler E. L., Combinatorial Optimization, Networks and Matroids, New York, Holt, Rinehart and Winston, 1976.
160. Leiner B. M., A Simple Model for Computation of Packet Radio Network Communication Performance, *IEEE Trans Commun*, COM-28, 2020—2023, 1980.
161. Limb J. O., Flores C., Description of Fasnets, or Unidirectional Local Area Communications Network, *Bell Syst Tech J*, 1982.
162. Limb J. O., Performance of Local Area Networks at High Speed, *IEEE Commun*, 41—45, 1984.
163. Li V. O. K., Silverster J. A., Performance Analysis of Networks with Unreliable Components, *IEEE Trans Commun*, COM-32, 1105—1110, 1984.
164. Little J., A Proof of the Queueing Formula $L = W$. *Oper Res. J*, 18, 172—174, 1961.
165. Luenberger D. G., Linear and Nonlinear Programming, Reading MA, Addison-Wesley, 1984.

166. Mathys P., Flajolet P., Q-ary Collision Resolution Algorithms in Random-Access Systems with Free or Blocked Channel Access, *IEEE Trans Inf Theory*, IT-31, 217—243, 1985.
167. Malis A. G., PSN End-to-End Functional Specification (RFC 979), BBN Communications Corp., Network Working Group, 1986.
168. Maxemchuck N. F., Netravali A. N., Voices and Data on a CATV Network, 1985, *IEEE J Select Areas Commun*, SAC-3, 300—311.
169. Massey J. L., Collision-Resolution Algorithms and Random Access Communications (Report UCLA-ENG-8016), Los Angeles, University of California, 1980.
170. McGregor P. V., Shen D., Network Design: An Algorithm for the Access Facility Location Problem, *IEEE Trans Commun*, COM-25, 61—73, 1977.
171. McQuillan J. M., Walden D. C., The ARPANET Design Decisions, *Networks*, 1, 1977.
172. Metcalf R. M., Boggs D. R., Ethernet: Distributed Packet Switching for Local Computer Networks, *Commun ACN*, 395—404, 1976.
173. Metcalfe R., Steady State Analysis of a Slotted and Controlled Aloha System with Blocking, Paper presented at 6th Hawaii Conference on System Science (Honolulu), 1973.
174. Михайлов В. А. Методы случайного множественного доступа. Диссертация на соискание ученой степени кандидата технических наук, 1979, Московский физико-технический институт.
175. Михайлов В. А., Цыбаков В. С. Верхняя граница для пропускной способности системы случайного множественного доступа. Пробл. передачи информ., 1981, т. 17, № 1, с. 90—95.
176. Mosely J., Humblet P. A., A Class of Efficient Contention Resolution Algorithms for Multiple Access Channels, *IEEE Trans Commun*, COM-33, 145—151, 1985.
177. Moss F. H., Segall A., An Optimal Control Approach to Dynamic Routing in Networks, *IEEE Trans Auto Contr*, AC-27, 329—339.
178. Monma C. L., Sheng D. D., Backbone Network Design and Performance Analysis: A Methodology for Packet Switching Networks, *IEEE J Select Areas Commun*, SCA-4, 946-965, 1986.
179. Mosely J., Asynchronous Distributed Flow Control Algorithms, PhD thesis, MIT, Dept. of Electrical Engineering and Computer Science, 1984.
180. McQuillan J. M., Richer I., Rosen E. C., Bertsekas D. P., ARPANET Routing Algorithm Improvements, Second Semiannual Report (Prepared for ARPA and DCA), Bolt, Beranek, and Newman, Inc., 1978.
181. McQuillan J. M., Richer I., Rosen E. C., The New Routing Algorithm for the ARPANET, *IEEE Trans Commun*, COM-28, 711—719, 1980.
182. Neuts M. F., Matrix-Geometric Solutions in Stochastic Models — An Algorithmic Approach, Baltimore, MD, The Johns Hopkins University Press, 1981.
183. Nisnevitch L., Strasbourger E., Decentralized Priority in Data Communication, *Proc 2nd Ann Symp Comput Arch*, 1974.
184. Nomura M., Tsukamoto K., Traffic Analysis of Polling Systems, *Trans Inst Elect Commun Eng* (Japan), J61-B, 600—607, 1978.
185. Nyquist H., Certain Topics in Telegraph Transmission Theory, *Trans AIEE*, 47, 617—644, 1928.
186. Pakes A. G., Some Conditions for Ergodicity and Recurrence of Markov Chains, *Op Res*, 17, 1059—1061, 1969.
187. Pape U., Implementation and Efficiency of Moore-Algorithms for the Shortest Route Problem, *Math Prog*, 7, 212—222, 1974.
188. Papadimitriou C. H., Steiglitz K., Combinatorial Optimization, Algorithms and Complexity, Englewood Cliffs, NJ, Prentice-Hall, 1982.
189. Perlman R., Fault-Tolerant Broadcast of Routing Information, *Computer Networks*, 7, 395—405 (*Proc IEEE Infocom'83*, San Diego), 1983.

190. Pippenger N., Bounds on the Performance of Protocols for a Multiple Access Broadcast Channel, *IEEE Trans Inf Theory*, IT-27, 1981.
191. Pinedo M., Wolff R. W., A Comparison Between Tandem Queues with Dependent and Independent Service Times, *Oper Res*, 30, 464—479.
192. Polak E., Computational Methods in Optimization, New York, Academic Press, 1971. [Имеется перевод: Полак Э. Численные методы оптимизации. Единый подход. — М.: Мир, 1974.]
193. Provan J. S., Ball M. O., Computing Network Reliability in Time Polynomial in the Number of Cuts, *Oper Res*, 32, 516—526, 1984.
194. Proakis J. G., Digital Communications, New York, McGraw-Hill, 1983.
195. Qureshi S., Adaptive Equalization, *Proc IEEE*, 1349—1387, 1985. [Имеется перевод: Куреши Ш. Адаптивная коррекция. — ТИИЭР, т. 7, № 9, с. 5—49, 1985.]
196. Raubold E., Haenle J., A Method of Deadlock-Free Resource Allocation and Flow Control in Packet Networks, *Proc Int Conf Comput Commun* (Toronto), 1976.
197. Ramamoorthy C. V., Wah B. W., The Isomorphism of Simple File Allocation, *IEEE Trans Comput*, C-32, 221—231, 1983.
198. Rinde J., TYMNET I: An Alternative to Packet Switching, *Proc 3rd Int Conf Comput Commun*, 1976.
199. Rivest R. L., Network Control by Bayesian Broadcast (Report MIT/LCS/TM-285), Cambridge, MA: MIT, Laboratory for Computer Science, 1985.
200. Roberts L. G., Aloha Packet System With and Without Slots and Capture, (ASS Note 8), Stanford, CA: Stanford Research Institute, Advanced Research Projects Agency, Network Information Center, 1972.
201. Rockafellar R. T., Convex Analysis, Princeton, NJ, Princeton University Press, 1970. [Имеется перевод: Рокафеллар Р. Т. Выпуклый анализ. — М.: Мир, 1973.]
202. Rockafellar R. T., Network Flows and Monotropic Programming, New York, John Wiley & Sons, 1984.
203. Ross S. M., Introduction to Probability Models, New York, Academic Press, 1980.
204. Ross S. M., Stochastic Processes, New York, John Wiley & Sons, 1983.
205. Rosberg Z., Deterministic Routing to Buffered Channels, *IEEE Trans Commun*, COM-34, 504—507, 1986.
206. Rosen E. C., Vulnerabilities of Network Control Protocols: An Example, *Comput Commun Rev*, 1981.
207. Rybczynski A., X.25 Interface and End-to-End Virtual Circuit Service Characteristics, *IEEE Trans Commun*, 500—509, 1980.
208. Sasaki G., Hajek B., Optimal Dynamic Routing in Simple Commodity Networks by Iterative Methods, *IEEE Trans Commun*, 1965 (forthcoming).
209. Sant D., Throughout of Unslotted Aloha Channels with Arbitrary Packet Interarrival Time Distribution, *IEEE Trans Commun*, COM-28, 1422—1425, 1980.
210. Sarachik P. E., Ozguner U., On Decentralized Dynamic Routing for Congested Traffic Networks, *IEEE Trans Auto Contr*, AC-27, 1233—1238, 1982.
211. Sarachik P. E., An Effective Local Dynamic Strategy to Clear Congested Multidestination Networks, *IEEE Trans Auto Contr*, AC-27, 510—513, 1982.
212. Schwartz M., Stern T. E., Routing Techniques Used in Computer Communication Networks, *IEEE Trans Commun*, COM-28, 539—552, 1980.
213. Segall A., Advances in Verifiable Fail-Safe Routing Procedures, *IEEE Trans Commun*, COM-29, 491—497, 1981.
214. Segall A., Distributed Network Protocols, *IEEE Trans Inf Theory*, IT-29, 23—34, 1983.
215. Shannon C. E., A Mathematical Theory of Communication, *Bell Syst Tech J*, 27, 379—423 (Part 1), 623—656 (Part 2), (Reprinted in book form by the University of Illinois Press, Urbana, 1949), 1948.

216. Siebert W. M., Circuits, Signals, and Systems. Cambridge, MA, MIT Press, and New York, McGraw-Hill, 1986.
217. Sidi M., Segall A., A Busy-Tone-Multiple-Access-Type Scheme for Packet-Radio Networks, In G. Payolk (Ed.), Performance of Data Communication Systems and Time Applications, (pp. 1—10), New York, North-Holland, 1981.
218. Soloway S. R., Humblor P. A., On Distributed Network Protocols for Changing Topologies, Codex Corp, 1986.
219. Sproule D. E., Mellor F., Routing, Flow and Congestion Control and the Data-pac Network, *IEEE Trans Commun*, COM-29, 386—391, 1981.
220. Stallings W., Data Computer Communications, New York, Macmillan, 1985.
221. Stuck B. W., Arthurs E., A Computer Communications Network Performance Analysis Primer, Englewood Cliffs, NJ, Prentice-Hall, 1985.
222. Stidman S., $L = W$, A Discounted Analogue and a New Proof, *Oper Res*, 20, 1115—1125, 1972.
223. Stidman S., Jr., A Last Work on $L = W$, *Oper Res*, 22, 417—421, 1974.
224. Stassinopoulos G. I., Konstantopoulos P., Optimal Congestion Control in Single Destination Networks, *IEEE Trans Commun*, COM-33, 792—800, 1985.
225. Takagi H., Kleinrock L., Throughput Delay Characteristics of Some Slotted-Aloha Packet Radio Networks, *IEEE Trans Commun*, COM-33, 1200—1207, 1985.
226. Takagi H., Analysis of Polling Systems, Cambridge, MA, MIT Press, 1986.
227. Tanenbaum A. S., Computer Networks, Englewood Cliffs, NJ, Prentice-Hall, 1981.
228. Tobagi F., Borgonovo F., Fratta L., Express-Net: A High Performance Integrated-Services Local Area Network, *IEEE J Select Areas Commun*, SAC-1, 1983.
229. Thaker G. H., Cain J. B., Interactions Between Routing and Flow Control Algorithms, *IEEE Trans Commun*, COM-34, 269—277, 1986.
230. Tobagi F. A., Random Access Techniques for Data Transmission Over Packet Switched Radio Networks. PhD thesis, UCLA, Computer Science Dept., 1974.
231. Tobagi F. A., Kleinrock L., Packet Switching in Radio Channels, Part II — The Hidden Terminal Problem in CSMA and Busy-Tone Solution, *IEEE Trans Commun*, COM-23, 1417—1433, 1975.
232. Towsley D., Wolf J. K., On the Statistical Analysis of Queue Lengths and Waiting Times for Statistical Multiplexors with ARQ Retransmission Schemes, *IEEE Trans Commun*, COM-27, 693—702, 1979.
233. Tsitsiklis J. N., Bertsekas D. P., Distributed Asynchronous Optimal Routing in Data Networks, *IEEE Trans Auto Contr*, AC-31, 325—331, 1986.
234. Tsitsiklis J. N., Markov Chains with Rare Transitions and Simulated Annealing (Report LIDS-1497), Cambridge, MA, MIT Laboratory for Information and Decision Systems, 1985.
235. Tsitsiklis J. N., Analysis of a Multiaccess Control Scheme (Report LIDS-P-1534), Cambridge, MA, MIT Laboratory for Information and Decision Systems, 1986.
236. Цыбаков Б. С., Бакиров В. Л. Устойчивость несинхронной системы Алоха. — Пробл. передачи информ., 1984, т. 20, № 1, с. 82—94.
237. Цыбаков Б. С., Бакиров В. Л. Передача пакетов в радиосетях. — Пробл. передачи информ., 1985, т. 21, № 1, с. 80—101.
238. Цыбаков Б. С., Берковский М. А. Множественный доступ с резервированием. — Пробл. передачи информ., 1980, т. 16, № 1, с. 50—76.
239. Цыбаков Б. С., Федоров С. П. Локальная сеть со стек-алгоритмом множественного доступа. — Пробл. передачи информ., 1986, т. 22, № 2, с. 49—58.
240. Цыбаков Б. С., Лиханов Н. Б. Верхняя граница для задержки в системе СМД с алгоритмом дробления. — Пробл. передачи информ., 1982, т. 18, № 4, с. 76—84.

241. Цыбаков Б. С., Михайлов В. А. Свободный синхронный доступ пакетов в широкополосный канал с обратной связью. — Пробл. передачи информ., 1978, т. 14, № 4, с. 32—59.
242. Цыбаков Б. С., Михайлов В. А. Случайный множественный доступ пакетов. Алгоритм дробления. — Пробл. передачи информ., 1980, т. 16, № 4, с. 65—79.
243. Цыбаков Б. С., Введенская Н. Д. Стек-алгоритм случайного множественного доступа. — Пробл. передачи информ., 1980, т. 16, № 3, с. 80—94.
244. Цыбаков Б. С. Случайный множественный доступ. — Препринт ИППИ АН СССР, М., 1984.
Tsybakov B. S., Survey of USSR Contributions to Random Multiple-Access Communications, *IEEE Trans Inf Theory*, 143—165 (1985).
245. Tweedie R. L., Operator-Geometric Stationary Distributions for Markov Chains with Application to Queueing Models, *Adv Appl Prob*, 14, 368—391, 1982.
246. Tymes L., Routing and Flow Control in TYMNET, *IEEE Trans Commun*, COM-29, 392—398, 1981.
247. Vastola K. S., A Numerical Study of Two Measures of Delay for Network Routing, MS thesis, University of Illinois, Dept. of Electrical Engineering, Urbana, 1979.
248. Введенская Н. Д., Пинскер М. С. Неоптимальность алгоритма дробления. В кн.: Международный семинар «Сверточные коды»; связь с многими пользователями», тезисы докладов, Сочи, 1983, с. 137—144.
249. Введенская Н. Д., Цыбаков Б. С. Задержка пакетов при стек-алгоритме множественного доступа. — Пробл. передачи информ., 1984, т. 20, № 2, с. 77—97.
250. Walrand J., Probabilistic Look at Networks of Quasi-Reversible Queues, *IEEE Trans Inf Theory*, IT-29, 825—831, 1983.
251. Walrand J., Varaiya P., Interconnections of Markov Chains and Quasi-Reversible Queueing Networks, *Stoc Proc Appl*, 10, 209—219, 1980.
252. Wecker S., The Digital Network Architecture, *IEEE Trans Commun*, COM-28, 510—526, 1980.
253. Weldon E. J., An Improved Selective Repeat ARQ Strategy, *IEEE Trans Commun*, COM-30, 480—486, 1982.
254. Wieselthier J. E., Ephremides A., A New Class of Protocols for Multiple Access in Satellite Networks, *IEEE Trans Auto Contr*, AC-25, 865—880.
255. Wolff R. W., Poisson Arrivals See Time Averages, *Oper Res*, 30, 223—231, 1982.
256. Wolff R. W., Tandem Queues with Dependent Service Times in Light Traffic, *Oper Res*, 82, 619—635, 1982.
257. Yum T. P., The Design and Analysis of a Semidynamic Deterministic Routing Rule, *IEEE Trans Commun*, COM-29, 4, 498—504, 1981.
258. Zangwill W., Nonlinear Programming, A Unified Approach, Englewood Cliffs, NJ, Prentice-Hall, 1969. [Имеется перевод: Зангвилл У. И. Нелинейное программирование. Единый подход. — М.: Сов. радио, 1973.]

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ

- Автоматический запрос на повторение (ARQ)** 48, 79—97, 240
ARQ на n шагов назад 85—88, 177
доказательство корректности 91—93
идеальный 96
эффективность 93—95, 138
в кольцевых сетях 309
пакетных радиосетях 330
сети ARPANET 83—85
с выборочным повтором 95—99, 135, 139
идеальный 96
остановкой и ожиданием 80—82
Адаптивная компенсация 56, 61, 133
Адресация 49, 120—126, 131
Алгоритм
ALOHA 245—254, 336
несинхронная 261
неустойчивое равновесие 253
синхронная 248, 249—254, 335
точка равновесия 253, 249—254
чистая 261
Беллмана—Форда 371—375, 379—388, 472
Дijkstra 376—378
древовидный 264—270
Ивена 430
Клейтмана 429—430
Крускала 366, 433
оптимальной маршрутизации 436—469
отпускания 427
псевдобайесовский 255—258
разбегания 263—283
двоичная обратная связь 324
обратная связь с задержкой 282
с передачей в порядке кругового опроса 283
обратном 280—282
поступления 270—280, 292
распределенный 38, 42—45, 48, 133
Флойда—Уоршела 378, 379
Эссау—Вильямса 433
- Американский национальный институт стандартов** 112
Амплитудная модуляция (AM) 60
Асинхронный символьный тракт 47
- Бит** 18
запроса или конца 115
Битовый тракт 32, 33, 46, 79, 97
- Взаимодействие открытых систем** 31, 45
Виртуальная цепь 120
Витая пара проводов 15
Вычислительная техника 13, 14
Вычислительные задачи реального времени 17
- Граф** 361
связный 362
сильно связный 369
- Двоичная синхронная связь** 100
Деление многочленов 75
Дерево 362
остовное 362
кратчайших путей 375
минимального веса (МОД) 364—368
- Диаграммы сигнальные** 62
Диграф 368
Динамическая маршрутизация 27, 417—419, 481
Динамическое программирование 370
- Длина**
второпроизводная 460
первопроизводная 439
сообщений 21

Длительность сеанса 21
 Добавление битов 102, 140, 307
 Дуга 361, 369

Заголовок 48, 79

Задача о
 выборе пропускных способностей 421—427
 максимальном потоке 429
 местоположении концентраторов 433
 построении МОД с ограничением 431
 размещении складов 435
 Задержка 10
 АЛОНА 259—261
 в спутниковых системах 297
 кольцевая сеть с передачей маркера 311, 312
 МДПН 288—290
 МДПН/ОК 303, 345
 на обработку 143
 передачи 143

Задолженность 246, 249, 256
 Звездобразная конфигурация 310
 Звон 59

Знак
 ETX 100
 DLE 100
 STX 100
 SYN 100

Иерархия модулей 27
 Идентификация сеансов 120—125
 Изаритмический метод 500
 Измерение 203
 Импульсная характеристика 52
 Инициирование сеансов 40
 Интерфейс RS-232-C 33
 Искажения нелинейные 65
 Исправление ошибок 38, 62, 75, 127—132

К-связность 428
 Кадр 25, 48, 64, 79
 информационный 114
 нenumерованный 115
 супервизорный 115
 типа отказ 115
 селективный отказ 115
 Кадрирование 99—112
 битовое 102

в спутниковых системах 297—298
 знаковое 100
 избыточность 104
 максимальный размер 109—112
 поля для указания длин 105
 при наличии ошибок 107—109

Канал
 микроволновый 67
 радиоканал 67, 240
 речевой 62, 64
 связи 32, 46, 49—70
 аналоговый 49
 двухточечный 49
 полосовой 59—64
 цифровой 49, 66
 спутниковый 67, 240—244, 294—298
 Коаксиальный кабель 15, 67
 Код
 ASCII 32, 42, 71, 99
 NRZ 52
 сверточный 74
 унарно-двоичный 107
 Кольцевые сети 306—314
 с передачей маркера 306—314
 регистрами 313
 тактированным доступом 313
 Команда установления режима 116
 Коммутатор 13
 Коммутация каналов 23—27
 пакетов 26
 Концентратор 10
 Конфликт 246
 Коэффициент использования 151, 162, 415

Лавинный алгоритм 396, 399—404
 Линейный код 72
 Линейные системы 61—63

Максимальное управление потоком 513—518
 Манчестерский код 59
 Маршрутизация 37, 39, 358—482
 адаптивная 353, 388—395
 асинхронная 355, 379—387
 взаимосвязь с управлением потоком 350
 виртуальных цепей 87, 120, 128, 467, 468, 480
 дейтаграммная 27, 132, 348
 колебания 354, 388—395
 маршрутные таблицы 350, 356, 358
 по кратчайшим путям 354—356, 360—395

- оптимальная 413, 436—467
- основные понятия 350—352
- распределенная 353
- в сети
 - ARPANET 353—357
 - Codex 466—468
 - SNA 359, 360
 - TYMNET 357—359
 - с часто меняющейся топологией 480
- статическая 353
- централизованная 353
- Матрица
 - Гесса 454
 - положительно определенная 455
 - положительно полуопределенная 455
- Международная организация стандартов (МОС) 31
- Международный консультативный комитет по телеграфии и телефони 112
- Межсимвольная интерференция 52, 56, 61
- Метод
 - девиации потока 446—453
 - замены ветвей 427, 432
 - наискорейшего спуска 455
 - насыщенного разряда 427
 - Ньютона 456
 - проекции градиента 454—463
 - Франка—Волфа 446—453
- Микропроцессор 13
- Многоточечные телефонные линии 240, 244, 314, 324
- Многочлен
 - порождающий 75—77
 - примитивный 77
- Множественный доступ 15, 36, 183, 240—347
 - в пакетных радиосетях 334
 - замедление по двичной экспоненте 260, 336
 - настойчивый 284
 - ненастойчивый 284
 - псевдобайесовская стабилизация 287
 - с избежанием конфликтов 317, 320, 321
 - обнаружением конфликтов 301
 - прослушиванием несущей (МДПН) 241, 280—288, 336
- Множество, свободное от конфликтов 327
- Модем 32, 46, 59—64, 98
- Модуль 27
- Модульность 27
- Модуляция 60—64
 - амплитудная 60
 - квадратурная (КАМ) 61
- Мощность сигнала 63
- Мультиплексор 10
- Надежность 17, 22, 39, 325—432
- Найквиста критерий 58
- Направление
 - допустимое 444
 - спуска 445
- Номер запроса 82
- порядковый 81
- ОА-пара см.** Пара отправитель—адресат
- Оборудование
 - обработки данных 131
 - передачи данных 33, 131
- Обратная связь
 - быстрая 246
 - с задержкой 282
- Обратное давление 498
- Окно скользящее 86, 491
- Оконное управление потоком 127, 152, 438—501
 - от конца до конца 491—497
 - пузловое 497—499
- Опрос 315
 - анализ очередей 183
 - обобщенный 323—325
 - централизованный 315
- Оптимальная маршрутизация
 - динамическая 417—419
 - формулировка 413—419
- Оптические волокна 15, 67
- Ориентированный цикл 369
- Остановка и ожидание 80—82
- Остовное дерево 362
 - минимального веса 364
- построение топологии 431, 432
- Ответ
 - готов к приему (RR) 115
 - не готов к приему (RNR) 115
- Отклонение фазы 65
- Ошибка 48, 72, 79, 133
- Пакет 19, 48
 - «вызов соединения» 133
 - ошибок 73
- Пара отправитель—адресат (ОА-пара) 415
- Перегрузка 127
- Передачная характеристика 54
- Передача

- в обратном порядке 280
 непрерывная по тракту 110
 с промежуточным накоплением 23—27
 файла 23
 Переполнение буфера 486—489
 Переход в графе 361
 ориентированный 368
 Период разрешения конфликта 266
 Подграф 362
 Подсеть 3
 Поле возраста 401
 Поллачека—Хинчина формула 173
 Полоса частот 62, 63
 Порядок сообщений 22
 Поточковые модели 413—419
 Правила доступа 40
 Предположение о
 бесконечном числе узлов 246
 квазистационарности 171
 Преобразование
 кодовое 41
 Фурье 54
 Проверка на четность 69
 по вертикали и горизонтали 71—72
 циклические избыточные про-
 верки (ЦИП) 74—78
 от конца до конца 130
 Пропускная способность
 алгоритма
 несинхронная АЛОНА 262
 с МДПН 291
 синхронная АЛОНА 250
 с МДПН 286
 разбиения 278
 с МДПН 292
 канала 63, 145
 кольцевой сети с передачей мар-
 кера 312
 МДПН/ОК 302, 304
 шины с передачей маркера и си-
 стем с опросом 316
 Протокол
 ADCCP 85, 112—120
 HDLC 48, 85, 112—120
 LAPB 112—120, 131
 SDLC 85, 112—120
 X21, 33, 131
 Процесс
 паритетный 29, 35, 39, 47
 пуассоновский 20, 155, 165, 245
 рождения и гибели 203, 231
 Процессор
 связи 13
 фронтальный 10
 Псевдобайесовский алгоритм 255—258
 АЛОНА с МДПН 287
 Путь в графе 361
 МППД-путь 443
 ориентированный 368
 Радиосети пакетные 68
 временное уплотнение 328, 329
 радиус передачи 333, 334
 разрешение конфликтов 330—333
 Разбиение на уровни 27—42
 Разделение требований 242
 Разрешение на передачу 126
 Рандомизация 203, 212
 Распределение
 нагрузки 17
 пуассоновское 157, 234
 экспоненциальное 157
 Расстояние минимальное 73
 Регенератор 66
 Регистр сдвига 75
 Режим
 асинхронного ответа 113
 нормального ответа 113
 прозрачности 100
 Резервирование в системах с
 многими пользователями 188
 множественным доступом 242, 294
 ограниченным обслуживанием 192,
 224
 одним пользователем 185
 отсечением 185
 очищением 185
 частичным отсечением 185
 Речь
 пакетированная 22
 цифровая 126
 Решение в форме произведения
 211, 213, 214
 СБИС 13
 Сборка сообщений 39
 Свертка 53
 Свободный для всех доступ 241
 Связность 428
 дуговая 428
 Сеанс 20—23
 Сетевое планирование 370
 Сеть
 глобальная 15
 городского масштаба 205, 317
 интегрального обслуживания циф-
 ровая (ЦИО) 15
 локальная 40, 240, 300—326, 336
 очередей
 аппроксимация Клейнрока 201,
 211, 415

- аппроксимация *M/M/1* 201, 211, 415, 442, 468
 замкнутая 214
 с многими классами требований 213
 тандемные линии 199
 теорема *Джексона* 201, 211—217
 ALOHA 3, 17
 ARQ 83—85
 кадрирование 101
 маршрутизация 353—357
 управление потоком 502
 CATV 322
 Codex
 кадрирование 124
 маршрутизация 359, 466
 УЛПД 505
 DECNET 13, 31, 105, 379
 ETHERNET 59, 64, 244, 260, 300—304
 EXPRESSNET 320, 321
 HOMENETS 322, 323
 SNA 31
 маршрутизация 359, 360
 протокол SDLC 85, 112—120
 управление
 виртуальными маршрутами 360
 группой передач 359
 потоком 491, 501, 504
 явными маршрутами 359
 уровень управления
 передач 359
 путями 359
 шаговая схема 491
 шаговое управление сеансного уровня 504
 TUMNET 12, 31
 кадрирование 122
 маршрутизация 357—359
 управление потоком 503
 Сжатие данных 19, 40
 Сигналы занятости 334—336
 Система
 массового обслуживания (СМО)
 замкнутая 152, 214, 226
 консервативная 198
M/D/1 174
M/C/1 173
M/C/1 с перерывами 180
M/M/1 155—158
M/M/1 с разделением обслуживания 223
M/M/m 167—169
M/M/∞ 170, 205
M/M/m/m 172, 205, 235
 обратимость времени 203
 опрос 183
 приоритет без прерывания 194
 приоритет с прерыванием 197, 222, 238
 резервирование 183
 передачи изображений 18
 разделения времени 153
 резервирования авиабилетов 16
 Скорость поступления 21, 147
 Служба речевая 16
 Снос
 алгоритм разбиения 278
 МДПН 285
 синхронная ALOHA 251
 Создание подсети связи 421
 Сообщение 18—20
 Среднее остаточное время 175
 Стек-алгоритм 266, 269
 Тайм-аут 80
 ТАКП (топологический алгоритм кратчайшего пути) 405—413
 Теорема
Берка 206
Литтла 146—150
 отсчетов 56
Шеннона 63, 66, 134
 Техника связи 14, 15
 Трейлер 48, 79
 Узел графа 361
 Уплотнение
 временное 63, 145, 162, 170, 181, 203, 242, 247
 в пакетных радиосетях 328—329
 статистическое 124, 144, 160, 169, 203
 частотное 64, 145, 162, 182, 242
 Упорядочивание пакетов 22, 84
 в системах множественного доступа 241
 Управление
 доступом к среде 36, 38, 240
 линией передачи данных (УЛПД) 34—36, 46—49, 70—120, 240
 потоком 37, 127—131
 с временным окном 519, 520
 максиминное 513—518
 на транспортном уровне 500, 501
 окно 490
 оптимальное 507—512
 регулирование входного трафика 506—520
 в сети

- ARPANET 502, 503
 Codex 505
 SNA 504
 TYMNET 503
 в сочетании с оптимальной маршрутизацией 507—512
 в спутниковых линиях 499
 цели 483—489
- Уравнение
 баланса
 глобального 160, 230, 234
 детального 203, 205, 231, 232
 частичного 233
Беллмана 374
Чепмена—Колмогорова 230
- Уровень
 представления 40
 применений 41
 сеансовый 40
 сетевой 36—40
 транспортный 39, 40, 126—132
 управления линией передачи
 данных (УЛПД) 34—36,
 46—49, 70
 физический 31—34
- Устойчивость алгоритма
 маршрутизации по кратчайшим
 путям 388—395
 МДПН 288—290
 несинхронная ALOHA 262
 разбиения 278
- Фазовая манипуляция 62
- Фильтрация 50—53
 нижних частот 57
 полосовая 59
- Флаг 100
 Фрагмент 364
- Холостое заполнение 46, 105, 307
- Цепь *Маркова*
 апериодическая 230
 вложенная 233
 неприводимая 230
 обратимость времени 203
 с дискретным временем 229—231
 непрерывным временем 233
 сеть очередей 231
 синхронная ALOHA 250
M/M/1 259
M/M/m 167
M/M/∞ 170
M/M/m/m 172
- Цикл в графе 362
- Черный ящик 27
- Шины с передачей маркера 314—316
 Шифрование 40
 Шлюз 39
 Шум 56
- Электронная почта 16
 Энтропия 106, 133
Эрланга
B-формула 173, 235
C-формула 168, 235

СОДЕРЖАНИЕ

Предисловие редактора перевода	5
Предисловие	7
1. Введение. Многоуровневая архитектура сети	10
1.1. Исторический обзор	10
1.2. Сообщения и коммутация	18
1.3. Разбиение на уровни	27
1.4. Пример задачи, связанной с использованием распределенного алгоритма	42
1.5. Замечания и дополнительная литература	45
2. Управление линией передачи данных и каналы связи	46
2.1. Обзор	46
2.2. Физический уровень. Каналы и модемы	49
2.3. Обнаружение ошибок	69
2.4. ARQ — методы повторной передачи	78
2.5. Кадрирование	99
2.6. Стандартные модули УЛПД	112
2.7. Идентификация сеансов и адресация	120
2.8. Восстановление после ошибок на сетевом и транспортном уровнях	125
2.9. Заключение	134
2.10. Замечания, источники и дополнительная литература	135
Задачи	135
3. Задержки в сетях передачи данных и математические модели	143
3.1. Введение	143
3.2. Модели теории массового обслуживания, теорема Литтла	146
3.3. Система массового обслуживания $M/M/1$	155
3.4. Системы $M/M/m$, $M/M/\infty$ и $M/M/m/m$	167
3.5. Система $M/G/1$	173
3.6. Сети линий связи	199
3.7. Обращение времени; теорема Берка	203
3.8. Сети очередей; теорема Джексона	211
3.9. Заключение	217
3.10. Замечания, источники и дополнительная литература	218
Задачи	219
Приложение А. Краткий обзор теории цепей Маркова	229
3А.1. Цепи Маркова с дискретным временем	229
3А.2. Уравнения детального баланса	231
3А.3. Уравнения частичного баланса	233
3А.4. Цепи Маркова с непрерывным временем (марковские процессы)	233
Приложение Б. Сводка результатов	235
4. Связь в системах с множественным доступом	240
4.1. Введение	240
4.2. Синхронный множественный доступ и система АЛОНА	245
4.3. Алгоритмы разбienia	263
4.4. Прослушивание несущей	283
4.5. Резервирование при множественном доступе	294

4.6. Пакетные радиосети	326
4.7. Заключение	336
4.8. Замечания, источники и дополнительная литература	337
Задачи	338
5. Маршрутизация в сетях передачи данных	348
5.1. Введение	348
5.2. Сетевые алгоритмы и выбор кратчайшего пути	360
5.3. Распространение информации, необходимой при маршрутизации; обращение с линиями, выходящими из строя	395
5.4. Потокосовые модели, оптимальная маршрутизация, построение топологии	413
5.5. Характерные особенности оптимальной маршрутизации	436
5.6. Методы допустимого направления для оптимальной маршрутизации	443
5.7. Проекционные методы для оптимальной маршрутизации	454
5.8. Маршрутизация в сети Codex	466
5.9. Заключение	469
5.10. Замечания, источники и дополнительная литература	469
Задачи	471
6. Управление потоками	483
6.1. Введение	483
6.2. Оконное управление потоком	490
6.3. Обзор применяемых на практике методов управления потоком	502
6.4. Схемы управления потоком, основанные на регулировании интенсивности входного трафика	506
6.5. Заключение	521
6.6. Замечания, источники и дополнительная литература	521
Задачи	522
Литература	526
Предметный указатель	537

Научное издание

Димитри Бертсекас, Роберт Галлагер

СЕТИ ПЕРЕДАЧИ ДАННЫХ

Заведующий редакцией д-р техн. наук А. Л. Шёрс

Зам. заведующего редакцией Э. Н. Бадиков

Ст. научный редактор Л. П. Якименко. Мл. научный редактор В. Н. Соколова

Художник В. И. Шаповалов. Художественный редактор Н. М. Иванов

Технический редактор Е. С. Потапенкова. Корректор В. И. Николаева

ИБ № 6929

Сдано в набор 16.01.89. Подписано к печати 30.08.89. Формат 60×90¹/₁₆.

Бумага кн.-ж. тип. Печать офсетная. Гарнитура литературная. Объем бум. л. 17,0.

Усл. печ. л. 34,0. Усл. кр.-отт. 34,0. Уч.-изд. л. 37,55. Изд. № 6/6142. Тираж 18 000 экз. Зак. 703. Цена 3 р.

Издательство «МИР» В/О «Совэкспорткнига» Государственного комитета СССР по делам издательств, полиграфии и книжной торговли. 129820, ГСП, Москва, И-110, 1-й Рижский пер., 2.

Типография № 6 ордена Трудового Красного Знамени издательства «Машиностроение» при Государственном комитете СССР по печати. 193144, г. Ленинград, ул. Моисеенко, 10.

Срѣдѣ