4. СЕТЕВОЙ УРОВЕНЬ

Задача технологий сетевого уровня – разработка маршрутов доставки пакетов от отправителей к получателям [1, 3, 7]. Причём каждому пакету, направляющемуся по своему адресу, часто требуется преодолевать несколько различных транзитных участков между маршрутизаторами. Функции сетевого уровня более сложные, чем функции уровня передачи данных.

На сетевом уровне сосредоточивается служебная информация о топологии подсети связи (в т. ч. множестве всех маршрутизаторов и их взаимосвязях). Задача сетевого уровня – оптимальный выбор нужного пути по подсети, обеспечивая, по возможности, равномерную нагрузку на маршрутизаторы. Если отправитель и получатель находятся в различных сетях, то на сетевом уровне решаются проблемы их корректного сопряжения.

4.1. Коммутация пакетов с ожиданием

Сетевой уровень обычно реализуется в рамках определённой физической топологии (рис. 4.1) [3, 7]. Основными компонентами схемы являются устройства ввода-вывода информации пользователей и устройства связи и коммутации оператора сети. Последние представлены
маршрутизаторами, соединёнными линиями связи.

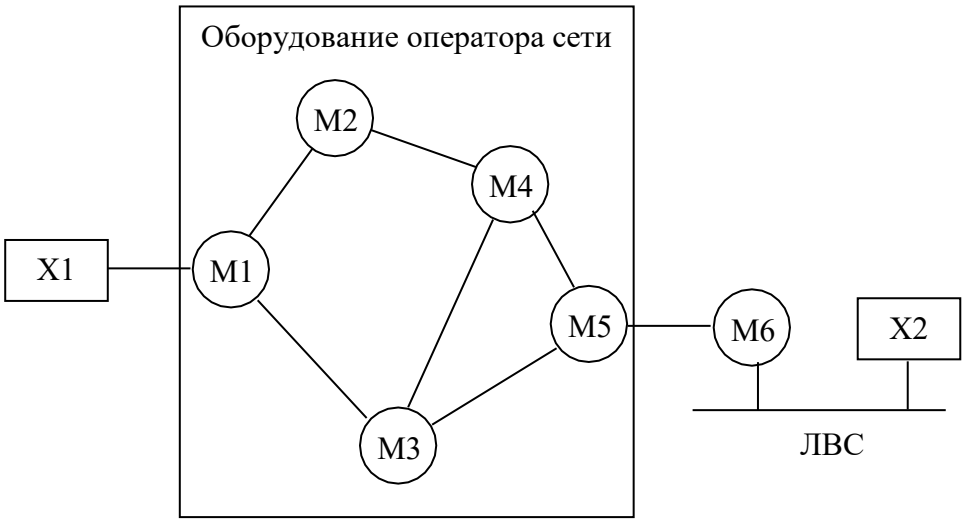


Рис. 4.1. Физическая среда функционирования сетевого уровня

Хост X1 соединён по выделенной линии с маршрутизатором M1 оператора сети. Хост X2 находится в ЛВС пользователя с маршрутизатором M6, который связан с сетью по выделенной линии. По предназначению и протоколам функционирования все маршрутизаторы идентичны.

Хост передаёт пакет на ближайший Маршрутизатор своей ЛВС или непосредственно по соединению оператора сети. Пакет записывается в буфер обмена и хранится до тех пор, пока не будет принят полностью, включая контрольную сумму. Затем он передаётся по цепочке маршрутизаторов до пункта назначения. Описанный механизм называется коммутацией пакетов с ожиданием.

4.2. Сервисы, предоставляемые транспортному уровню

Сетевой уровень предоставляет транспортному уровню сервисы в виде интерфейсов между ними. При этом должны выполняться следующие требования [3, 7]:

1) сервисы сетевого уровня не должны зависеть от технологии маршрутизатора;

2) транспортный уровень должен быть независимым от количества, типа и топологии подсетей с маршрутизаторами;

3) сетевые адреса, доступные транспортному уровню, должны использовать единую систему нумерации в ЛВС и глобальных сетях.

При разработке сети нет чёткого правила в написании детальной спецификации сервисов, которые обычно должны предоставляться транспортному уровню. В сетевых технологиях существуют противоположные точки зрения по вопросу о том, какие сервисы должен предоставлять сетевой уровень – ориентированные на соединение или не требующие соединений.

Разработчики Интернет-сообщества считают, что работа маршрутизатора заключается только в перемещении пакетов и никакие иные функции ему не нужны. Подсеть обладает априорной ненадёжностью независимо от того, как она спроектирована. Хосты должны это учитывать, обнаруживая и исправляя ошибки своими силами, а также самостоятельно управлять потоком. Поэтому сетевой сервис не должен требовать установки соединения и состоять в основном из примитивов SEND PACKET (передача пакета) и RECEME PACKET (приём пакета). Сюда нельзя включать упорядочивание пакетов и контроль потока – всё равно эти действия будет выполнять хост. От выполнения одной и той же работы дважды качество обслуживания не повысится. Кроме того, каждый пакет должен содержать полный адрес получателя, так как пересылка производится независимо от предыдущих пакетов.

Напротив, разработчики телефонных компаний полагают, что сеть должна предоставлять надёжный сервис с соединением. С точки зрения управления телефонными системами, качество обслуживания – определяющий фактор, а без установления соединения в подсети трудно добиться приемлемых результатов, особенно когда дело касается трафика реального масштаба времени – например, передачи голоса и видео.

Примеры технологий, реализующих эти крайние позиции – это Интернет и ATM. Интернет всегда предоставляет не требующие установления соединения сервисы сетевого уровня, а система ATM – ориентированные на соединения.

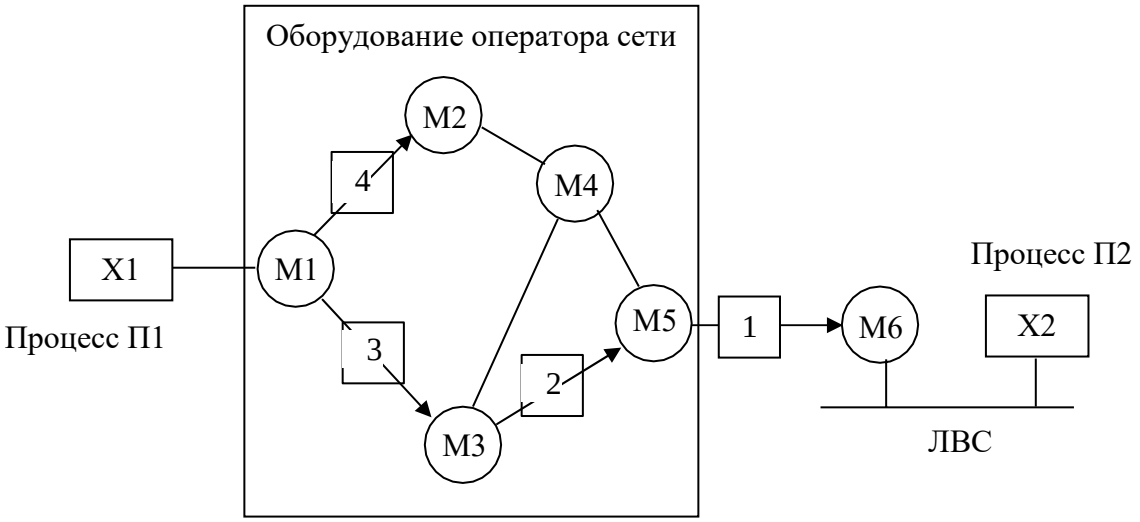
4.3. Сервис без установления соединения

Сервис без установления соединения предполагает, что пакеты поступают в подсеть по отдельности и их маршруты рассчитываются независимо. Тогда никакой предварительной настройки не требуется.

В этом случае пакеты называются дейтаграммами, а подсети – дейтаграммными [3]. При использовании сервиса с соединением путь между маршрутизаторами отправителя и получателя должен быть установлен до начала транспортировки пакетов. Такое соединение называется виртуальным каналом, а подсеть - подсетью виртуального канала.

Рассмотрим принцип работы дейтаграммной подсети (рис. 4.2) [3]. Пусть процесс П1 посылает длинное сообщение для П2. Он передаёт информацию транспортному уровню, сообщает ему, что требуется доставить данные процессу П2, выполняющемуся на хосте Х2. Код транспортного уровня исполняется на хосте Х1. Обычно он является частью операционной системы. Заголовок транспортного уровня вставляется в начало сообщения. В таком виде оно передаётся на сетевой уровень. Обычно это ещё одна процедура операционной системы.

Предположим, что сообщение в 4 раза длиннее максимального размера пакета. Тогда сетевой уровень должен разбить его на 4 пакета и послать их поочерёдно на маршрутизатор М1 с использованием протокола двухточечного соединения, например, PPP. Начинает работать оператор сети. Каждый маршрутизатор имеет свою внутреннюю таблицу, по которой он определяет дальнейший путь пакета при каждом из возможных адресов назначения. Каждая запись таблицы состоит из двух полей: пункта назначения (адресата) и выходящей линии для данного адресата. Во втором поле могут использоваться только линии, непосредственно соединённые с данным маршрутизатором. Например, на рис. 4.2 у маршрутизатора М1 имеется только две исходящие линии – к М2 и М3. Поэтому все входящие пакеты должны пересылаться на один из этих двух маршрутизаторов, даже если они не являются адресатами.



В начале

M1	-
M2	M2
M3	M3
M4	M2
M5	M3
M6	M3

Назначение Линия

В конце

M2	M2
M3	M3
M4	M2
M5	M2
M6	M3

Таблица маршрутизатора M3

M1	M4
M2	M1
M3	-
M4	M4
M5	M2
M6	M5

Таблица маршрутизатора M5

M1	M4
M2	M4
M3	M3
M4	M4
M5	-
M6	M6

Рис. 4.2. Маршрутизация в дейтаграммной подсети

Пакеты 1, 2 и 3, прибывая по сети на маршрутизатор M1, кратковременно сохраняются в буферной памяти для проверки их корректного приёма по контрольной сумме. Затем, по таблице M1 они передаются на маршрутизатор M3. Далее пакет 1 уходит на M5, откуда поступает на маршрутизатор ЛВС M6. После прибытия на M6 пакет инкапсулируется в кадр уровня передачи данных и передаётся на хост X2 по ЛВС. Пакеты 2 и 3 следуют по тому же маршруту. Пакет 4 после прибытия на M1 передаётся на маршрутизатор M2, несмотря на то, что адресом назначения является M6, как у первых трёх пакетов. По различным причинам маршрутизатор M1 может послать пакет 4 по новому маршруту. Причиной может быть затор на линии M1-M3-M5, возникший при пересылке первых трёх пакетов. Такие явления возникают при ограниченной пропускной способности каналов связи или возникновении внешних помех на линиях. В результате маршрутизатор M1 обновляет свою таблицу (см. рис. 4.2 под надписью «В кон-

це»). Решение о корректировке маршрута принимается алгоритмом маршрутизации.

4.4. Сервис с установлением соединения

Идея состоит в предотвращении выбора своего маршрута для каждого пакета (см. рис. 4.2) [3]. Вместо этого устанавливается соединение, маршрут от источника сообщения до получателя прописывается в настройках системы и хранится в специальных таблицах, встроенных в маршрутизаторы. Один и тот же маршрут используется для всего трафика, проходящего через данное соединение. Так работает телефонная система. Когда соединение разрывается, виртуальный канал также прекращается своё существование. При использовании сервиса с установлением соединения каждый пакет включает идентификатор виртуального канала (рис. 4.3).

Хост X1 устанавливает соединение с хостом X2. Это соединение запоминается и становится первой записью во всех таблицах маршрутизации. Первая строчка таблицы маршрутизатора M1 означает, что если пакет с идентификатором соединения 1 пришёл с хоста X1, то его надо направить на M3 с тем же идентификатором.

Если хост X3 захочет установить соединение с X2, то он выбирает идентификатор соединения 1 (в данном случае, у него нет выбора, это единственное существующее соединение) и просит подсеть установить виртуальный канал. Так в таблице появляется вторая запись. Здесь возникает конфликт, потому что если M1 может отличить пакеты соединения 1, пришедшие от X1, от пакетов соединения 1, пришедших с X3, то M3 такой возможности не имеет. Поэтому M1 присваивает новый идентификатор соединения исходящему трафику и тем самым создаёт второе соединение. Маршрутизаторам нужна возможность изменения идентификаторов соединения в исходящих пакетах для предотвращения подобных конфликтов. Иное название процедуры – коммутация меток.

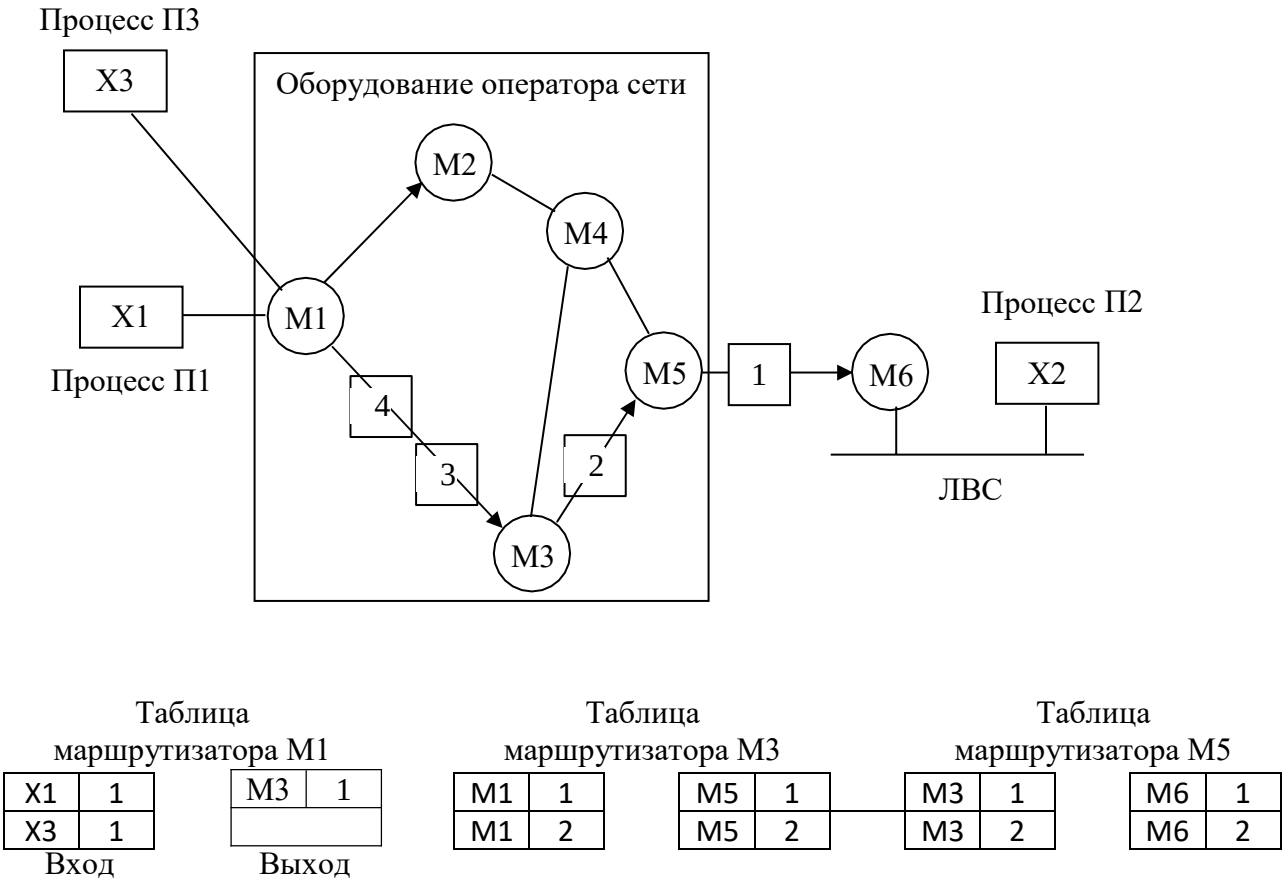


Рис. 4.3. Маршрутизация в подсети виртуального канала

4.5. Сравнение сетей с установлением логических соединений с дейтаграммными сетями

Каждая сеть обладает своими достоинствами и недостатками и используется на практике примерно в равной степени. Основные аргументы сравнения приведены в табл. 4.1 [3].

Оба подхода к созданию сетей реализуются путём компромиссов. Существует компромисс между внутренней памятью маршрутизатора и пропускной способностью. Виртуальные каналы позволяют экономить на адресах получателя, указывая вместо них короткие номера виртуальных каналов. Если размер кадра мал, то полный адрес получателя может составлять довольно существенную часть всего кадра, сильно снижая полезную пропускную способность сети. В то же время, для хранения больших таблиц с адресами в маршрутизаторе может потребоваться много памяти.

Таблица 4.1

Сравнение способа виртуальных каналов с дейтаграммным способом

Признак сравнения	Дейтаграммы	Виртуальные каналы
Установка канала	Не требуется	Требуется
Адресация	Каждый пакет содержит полный адрес отправителя и	Каждый пакет содержит короткий номер виртуального канала
Информация о состоянии	Подсеть не содержит информации о состоянии	Каждый виртуальный канал требует места в таблице подсети
Маршрутизация	Маршрут каждого канала выбирается независимо	Единый маршрут для всех пакетов выбирается при установке виртуального канала
Последствия выхода из строя маршрутизатора	Только потерянны	Все виртуальные каналы, проходящие через маршрутизатор, прекращают существование
Борьба с перегрузкой	Трудно реализуется	Легко реализуется при наличии достаточного количества буферов для каждого виртуального канала

Второй компромисс при создании сетей – между временем установления соединения и временем обработки адреса. Виртуальный канал требует затрат времени на его установку, но последующая обработка пакетов для маршрутизатора будет проще и быстрее, чем в дейтаграммной сети.

Виртуальные каналы обладают некоторыми преимуществами, помогающими им предоставлять гарантированное качество обслуживания и избегать заторов в подсети, так как ресурсы могут быть зарезервированы заранее, во время установления соединения. Когда начинают прибывать пакеты, необходимая пропускная способность и мощность маршрутизатора будут предоставлены системой. В дейтаграммной подсети предотвращение заторов реализовать значительно сложнее.

В системах обработки транзакций (например, при запросе магазина на верификацию кредитной карты) затраты времени на установление соединения и удаление виртуального канала могут сильно снизить потребительские свойства сети.

Если объём информации, передаваемой во время одного соединения, невелик, то использование виртуального канала не имеет смысла.

В данной ситуации могут оказаться полезными постоянные виртуальные каналы, установленные вручную и не разрываемые месяцами и даже годами.

Недостатком виртуальных каналов является их уязвимость в случае выхода из строя по различным причинам или временного выключения маршрутизатора. Даже если он будет быстро починен и снова включен, все виртуальные каналы, проходившие через него, будут прерваны. Если же в дейтаграммной сети маршрутизатор выйдет из строя, то будут потеряны только те пакеты, которые находились в данный момент на маршрутизаторе (а возможно, лишь некоторые из них, в зависимости от того, были они подтверждены или нет). Обрыв линии связи для виртуальных каналов является фатальным, а в дейтаграммной системе может оказаться почти незамеченным. Кроме того, дейтаграммная система позволяет соблюдать баланс между загрузкой маршрутизаторов и линий связи.

4.6. Алгоритмы маршрутизации

Основная функция сетевого уровня заключается в выборе маршрута для пакетов от начальной до конечной точки. В большинстве сетей пакетам приходится проходить через несколько маршрутизаторов. Единственным исключением здесь являются широковещательные сети, но даже в них маршрутизация является важным вопросом, если отправитель и получатель находятся в разных сетях. Алгоритмы выбора маршрутов и используемые ими структуры данных являются главной целью при проектировании сетевого уровня.

Алгоритм маршрутизации пакетов реализуется той частью программного обеспечения сетевого уровня, которая отвечает за выбор выходной линии для отправки пришедшего пакета. Если подсеть использует дейтаграммную службу, выбор маршрута для каждого пакета должен производиться заново, так как оптимальный маршрут мог измениться. Если подсеть использует виртуальные каналы, маршрут выбирается только при создании нового виртуального канала. После этого все информационные пакеты следуют по выбранному маршруту. Последний случай иначе называется сеансовой маршрутизацией, так как маршрут остаётся в силе на протяжении всего сеанса связи с пользователем.

Есть разница в понятиях маршрутизации, при которой системе приходится делать выбор определённого маршрута следования, и пересылки действием, происходящим при получении пакета. Можно представить себе маршрутизатор как устройство, в котором функционируют два процесса. Один из них обрабатывает приходящие пакеты и выбирает для них по таблице маршрутизации исходящую линию. Такой процесс называется пересылкой. Второй процесс отвечает за заполнение и обновление таблиц маршрутизации. Именно здесь в игру вступает алгоритм маршрутизации.

Вне зависимости от того, отдельно ли выбираются маршруты для каждого пакета или же только один раз – для соединения, желательно, чтобы алгоритм выбора конкретного маршрута обладал определёнными свойствами: корректностью, оптимальностью, надёжностью, устойчивостью, и простотой. Правильность и простота вряд ли требуют комментариев, а вот потребность в надёжности не столь очевидна с первого взгляда. Во время работы большой сети постоянно происходят какие-то отказы аппаратуры и изменения топологии. Алгоритм маршрутизации должен уметь справляться с изменениями топологии и трафика без необходимости прекращения всех задач на всех хостах и перезагрузки сети при каждой поломке маршрутизатора.

Алгоритм сетевой маршрутизации должен также обладать устойчивостью. Существуют алгоритмы выбора маршрута, никогда не приходящие в состояние равновесия, независимо от того, как долго они работают.

Такая цель, как оптимальность, может показаться очевидной, но здесь могут возникнуть определённые сложности. Для примера рассмотрим следующую ситуацию (рис. 4.4). Предположим, что трафик между станциями *A* и *A'*, *B* и *B'*, *C* и *C'* настолько интенсивный, что горизонтальные линии связи оказываются полностью загруженными. Чтобы максимизировать общий поток данных, трафик между станциями *X* и *X'* следовало бы совсем отключить. Однако станции *X* и *X'* тоже должны работать в сети. Очевидно, необходим компромисс между справедливым выделением трафика всем станциям и оптимальным использованием канала в глобальном смысле.

Задача оптимальности решается исходя из конкретно выбранного критерия. Можно попробовать минимизировать среднее время задержки или увеличить общую пропускную способность сети.

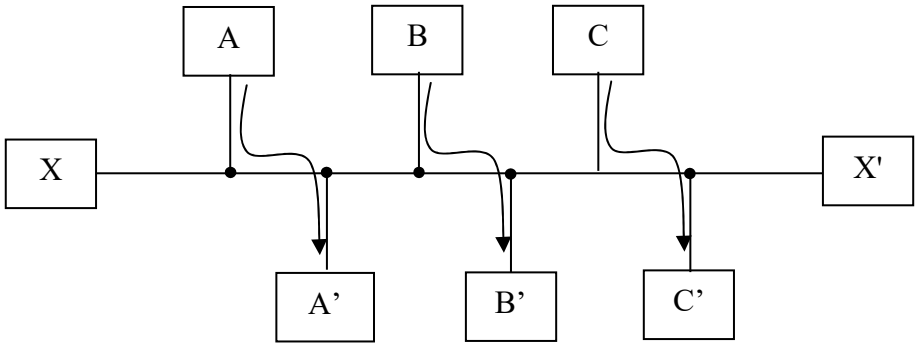


Рис. 4.4. Проблема оптимальности трафика

Однако эти цели противоречат друг другу, поскольку работа любой системы с очередями вблизи максимума производительности предполагает долгое стояние в очередях. В качестве компромисса многие сети стараются минимизировать количество пересылок для каждого пакета, поскольку при этом снижается время прохождения пакета по сети, а также нагрузка на сеть, в результате чего улучшается пропускная способность.

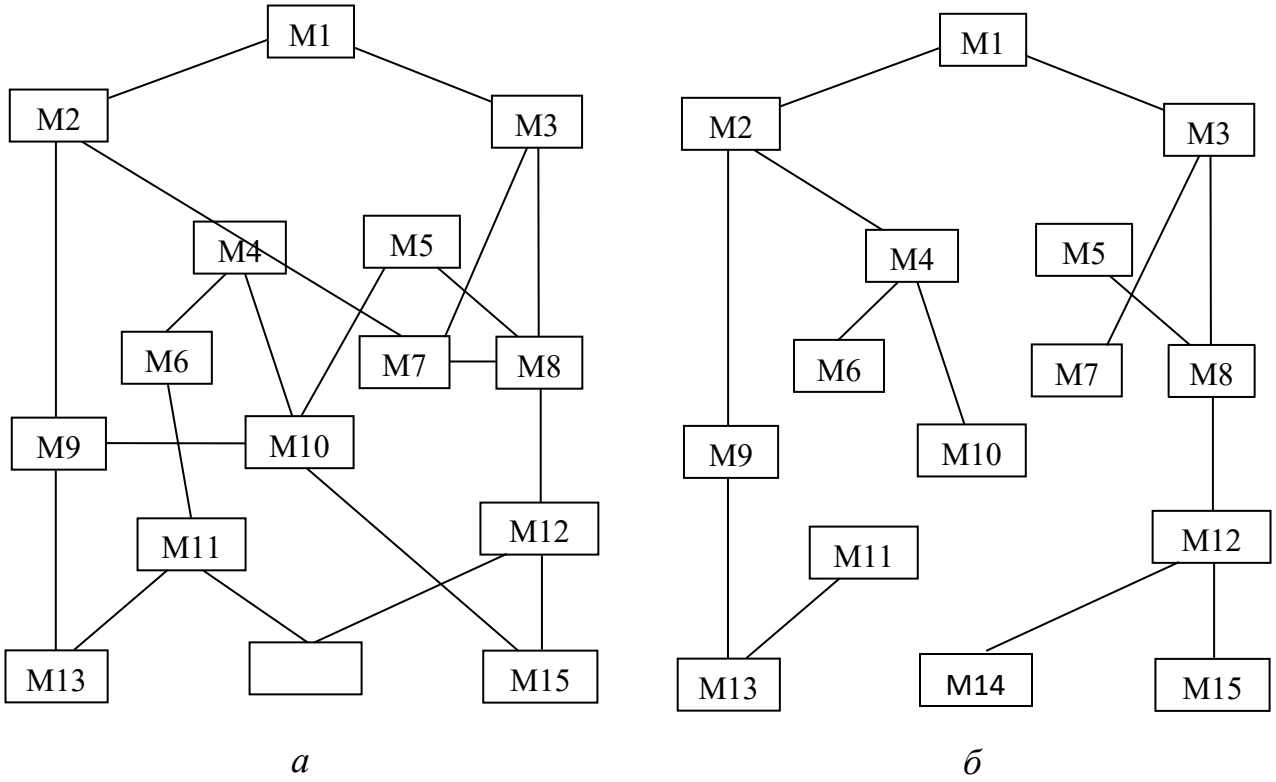
Алгоритмы выбора маршрута можно разбить на два основных класса: адаптивные и неадаптивные [3]. Неадаптивные алгоритмы не учитывают при выборе маршрута топологию и текущее состояние сети и не изменяют трафик на линиях. Вместо этого выбор маршрута для каждой пары станций производится заранее, в автономном режиме. Список маршрутов загружается в маршрутизаторы во время каждой загрузки сети. Такая процедура называется статической маршрутизацией.

Адаптивные алгоритмы, напротив, изменяют решение о выборе маршрутов при изменении топологии, а также, во многих случаях - в зависимости от загруженности линий. Адаптивные алгоритмы отличаются источниками получения информации (такие источники могут быть, например, локальными, если это соседние маршрутизаторы, либо глобальными, если это вообще все маршрутизаторы сети), моментами изменения маршрутов (например, через определённые равные интервалы времени, при изменении нагрузки или при изменении топологии) и данными, использующимися для оптимизации (расстояние, количество транзитных участков или ожидаемое время пересылки).

4.7. Принцип оптимальности маршрута

Основополагающей идеей сетевого уровня является принцип оптимальности [2]. В соответствии с этим принципом, если маршрутизатор *B* располагается на оптимальном маршруте от маршрутизатора *A* к маршрутизатору *C,* то оптимальный маршрут от маршрутизатора *B* к маршрутизатору *C* совпадёт с частью первого маршрута. Чтобы убедиться в этом, обозначим часть маршрута от маршрутизатора *A* к маршрутизатору *B* как *r*1 а остальную часть маршрута – *r*2. Если бы существовал более оптимальный маршрут от маршрутизатора *B* к маршрутизатору *C*, чем *r*2, то его можно было бы объединить с *r*1, чтобы улучшить маршрут от маршрутизатора *A* к маршрутизатору *C,* что противоречит первоначальному утверждению о том, что маршрут *r*1 - *r*2 является оптимальным.

Прямым следствием принципа оптимальности сетей является возможность рассмотрения множества оптимальных маршрутов от всех источников к приёмникам в виде дерева (рис. 4.5). Такое дерево называется входным деревом.



а *б*

Рис. 4.5. Пример топологии сети:
а – подсеть; *б* – входное дерево для маршрутизатора M1

Расстояния между отправителем и получателем сообщений измеряются количеством транзитных участков маршрута. Входное дерево

не обязательно является уникальным. У одной сети может существовать несколько входных деревьев с одинаковыми длинами путей. Цель всех алгоритмов выбора маршрутов заключается в вычислении и использовании входных деревьев для всех маршрутизаторов.

Входное дерево не содержит петель, поэтому каждый пакет будет доставлен за конечное и ограниченное число пересылок. На практике всё это не так просто. Линии связи и маршрутизаторы могут выходить из строя и снова появляться в сети во время выполнения операции, поэтому у разных маршрутизаторов могут оказаться различные представления о текущей топологии сети. Кроме того, ещё не обсуждался вопрос о том, собирает ли маршрутизатор информацию для вычисления входного дерева сам или эта информация поступает к нему каким-то другим образом. Тем не менее принцип оптимальности и входное дерево – это те точки отсчета, относительно которых можно измерять эффективность различных алгоритмов маршрутизации.

4.8. Выбор кратчайшего пути

Изучение алгоритмов выбора маршрутов начинается с широко применяемого в различных формах метода графов благодаря его простоте и понятности. Идея заключается в построении графа подсети, где каждый узел соответствует маршрутизатору, а каждая дуга – линии связи. При выборе маршрута между двумя маршрутизаторами алгоритм просто находит кратчайший путь между ними на графе (рис. 4.6) [2].

Один из способов измерения длины пути состоит в подсчёте количества транзитных участков. В таком случае пути M1-M2-M6 и M1-M2-M4 (рис. 4.6, *а*) имеют одинаковую длину. Но если измерять расстояния в километрах, то окажется, что путь M1-M2-M6 значительно длиннее пути M1-M2-M4.

Кроме количества транзитных участков и физической длины линий возможен учёт и других параметров. Например, каждой дуге графа можно поставить в соответствие среднюю длину очереди и время задержки пересылки, которые определяются каждый час с помощью передачи специального тестового пакета. В таком графе кратчайший путь определяется как самый быстрый путь, а не путь с самой короткой длиной кабеля или путь, состоящий из минимального числа отдельных отрезков кабеля.

В общем случае, параметры дуг графа являются функциями расстояния, пропускной способности, средней загруженности, стоимости связи средней длины очереди, измеренной величины задержки и других факторов. Изменяя весовую функцию, алгоритм может вычислять кратчайший путь с учётом любого количества критериев в различных комбинациях.

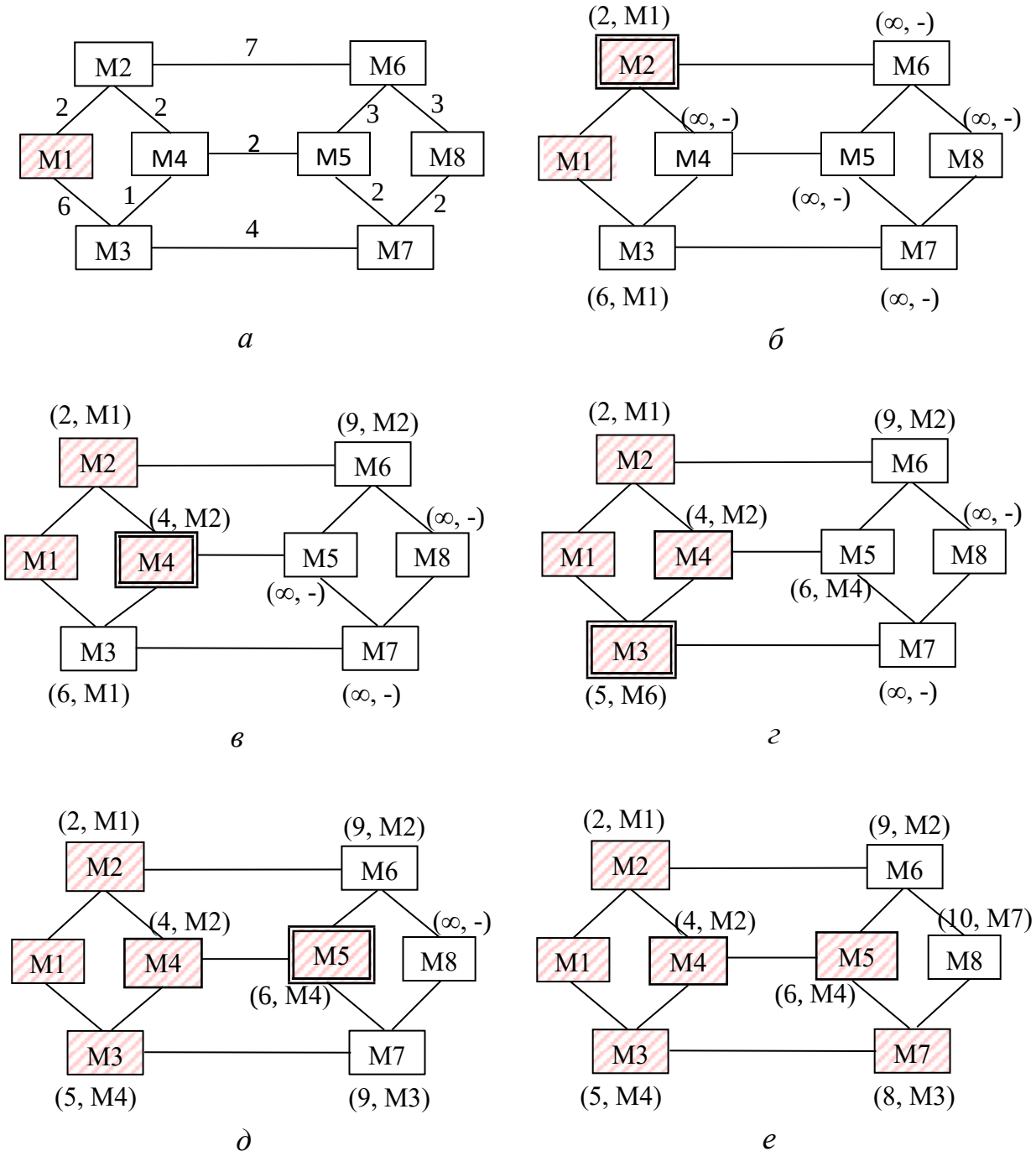


Рис. 4.6. Первые пять шагов вычисления кратчайшего пути от M1 к M8:
а – взвешенный ненаправленный граф; *б* – определение ближайшего узла; *в* – определение текущего рабочего узла; *г-е* – завершающие этапы вычисления пути; 3, 4, 5-й – шаги соответственно

Известно несколько алгоритмов вычисления кратчайшего пути между двумя узлами графа. Один из них был создан Дейкстрой (Dijkstra) в 1959 г. Каждый узел помечается (в скобках) расстоянием до него от узла отправителя по наилучшему известному пути. Вначале пути неизвестны, поэтому все узлы помечаются символом бесконечности. По мере работы алгоритма и нахождения путей отметки узлов изменяются, показывая оптимальные пути. Отметка может быть постоянной или экспериментальной. Вначале все отметки являются ориентировочными. Когда выясняется, что отметка действительно соответствует кратчайшему возможному пути, она становится постоянной и в дальнейшем не изменяется.

Чтобы показать, как работает этот алгоритм, рассмотрим взвешенный ненаправленный граф (см. рис. 4.6, *а*), где весовые коэффициенты соответствуют, например, расстояниям. Нужно найти кратчайший путь от М1 к М8. Для начала розовым пунктиром помечается узел М1 как постоянный. Затем исследуются все соседние с ним узлы, около них указывается расстояние до узла М1. Если отыскивается более короткий путь к какому-либо узлу, то вместе с указанием расстояния в отметке меняется и узел, через который прошёл более короткий путь. Таким образом позже можно восстановить весь путь. Рассмотрев все соседние с М1 узлы, помечается ближайший узел как постоянный (см. рис. 4.6, *б*). Этот узел и становится новым рабочим узлом.

Теперь можно повторить ту же процедуру с узлом М2, исследуя все его соседние узлы. Если сумма расстояния от узла М2 и значения отметки в узле М2 (расстояния от М1 до М2) оказывается меньше, чем отметка у исследуемого узла (уже найденное другим путём расстояние от М1), это значит, что найден более короткий путь, поэтому пометка узла изменяется.

После того, как все соседние с рабочим узлы исследованы и временные отметки при необходимости изменены, по всему графу ищется узел с наименьшей временной отметкой. Этот узел помечается как постоянный и становится текущим рабочим узлом.

На этапе (рис. 4.6, *в*) узел М4 отмечен как постоянный. Предположим, что существует более короткий путь, чем М1-М2-М4, например, М1-М3-М7-М5-М4. В этом случае предполагаемый маршрут исследуется и принимается решение о целесообразности его использования. По пути к М8 все варианты маршрутов просчитываются неоднократно в различных сочетаниях. При этом имеет значение текущее состояние каналов связи – их исправность и скорость передачи данных. Последняя зависит от внешних помех интенсивности трафика.

4.9. Заливка

Метод заливки (лавинной маршрутизации) – это статический алгоритм, при котором каждый приходящий пакет посылается на все исходящие линии, кроме той, по которой он пришёл [2, 3]. Очевидно, что лавинная марштуризация порождает огромное количество дублированных пакетов, даже бесконечное количество в сетях с замкнутыми контурами, если не принять специальных мер. Одна из таких мер состоит в помещении в заголовок пакета счётчика преодолённых им транзитных участков, уменьшаемого при прохождении каждого маршрутизатора. Когда значение этого счётчика падает до нуля, пакет удаляется. В идеальном случае счётчик транзитных участков должен вначале устанавливаться равным длине пути от отправителя до получателя. Если отправитель не знает расстояния до получателя, он может установить значение счётчика равным длине максимального пути в данной подсети.

Альтернативный способ ограничения количества тиражируемых пакетов заключается в их учёте при прохождении через маршрутизатор. Это позволяет не посылать их повторно. Один из методов состоит в том, что каждый маршрутизатор помещает в каждый получаемый от своих хостов пакет порядковый номер. Все маршрутизаторы ведут список маршрутизаторов-источников, в котором сохраняются все порядковые номера пакетов, которые им встречались. Если пакет от данного источника с таким порядковым номером в списке уже есть, то дальше он не распространяется и удаляется.

Чтобы предотвратить неограниченный рост размера списка, можно снабдить все списки счётчиком *k,* показывающим, что все порядковые номера, вплоть до *k*, уже встречались. И когда приходит пакет, можно очень легко проверить, не является ли он дубликатом. При положительном ответе такой пакет отвергается. Кроме того, не нужно хранить весь список до *k,* так как этот счётчик очень действенно подытоживает его.

На практике чаще всего применяется выборочная заливка. В этом алгоритме маршрутизаторы посылают пакеты не по всем линиям, а только по тем, которые идут приблизительно в нужном направлении. Вряд ли есть смысл посылать пакет, направляющийся на запад, по линии, идущей на восток, если только топология сети не представляет собой лабиринт и маршрутизатор не знает об этом.

В большинстве случаев алгоритм лавинной маршрутизации является непрактичным, но, тем не менее, иногда он применяется. Например, в военных приложениях, где большая часть маршрутизаторов в любой момент может оказаться уничтоженной, высокая надёжность алгоритма заливки является, наоборот, желательной. В распределённых базах данных иногда бывает необходимо одновременно обновить все базы данных, и в этом случае заливка также оказывается полезной. Третье применение алгоритма заливки – эталонное тестирование других алгоритмов выбора маршрутов, так как заливка всегда находит все возможные пути в сети, а следовательно, и кратчайшие. Ухудшить эталонные показатели времени доставки могут разве что накладные расходы, вызванные огромным количеством пакетов, формируемых самим алгоритмом заливки.

4.10. Маршрутизация по вектору расстояний

Современные компьютерные сети обычно используют не статические, а динамические алгоритмы маршрутизации, поскольку статические просто не принимают во внимание текущую нагрузку на сеть. Самой большой популярностью пользуются два метода: маршрутизация по вектору расстояний и маршрутизация с учётом состояния каналов [3]. В данном пункте (4.10) изложен первый, в следующем (4.11) будет изложен второй метод. Алгоритмы маршрутизации по вектору расстояний работают, опираясь на таблицы, поддерживаемые всеми маршрутизаторами и содержащие наилучшие известные пути к каждому из возможных адресатов. Для обновления данных этих таблиц производится обмен информацией с соседними маршрутизаторами.

Алгоритм маршрутизации по вектору расстояний иначе называют по именам его создателей: распределённым алгоритмом Беллмана – Форда (Bellman – Ford) и алгоритмом Форда – Фулкерсона (Ford – Fulkerson), (Bellman, 1957; Ford and Filkerson, 1962). Этот алгоритм изначально применялся в сети ARPANET и в Интернете был известен под названием RIP.

При маршрутизации по вектору расстояний таблицы, с которыми работают и которые обновляют маршрутизаторы, содержат записи о каждом маршрутизаторе подсети. Каждая запись состоит из двух частей: предпочитаемого номера линии для данного получателя и предполагаемого расстояния или времени прохождения пакета до

этого получателя. В качестве единиц измерения на практике может использоваться число транзитных участков, миллисекунды, число пакетов, ожидающих в очереди в данном направлении, или ещё что-нибудь подобное.

Предполагается, что маршрутизаторам известно расстояние до каждого из соседей. Если в качестве единицы измерения используется число транзитных участков, то расстояние равно одному транзитному участку. Если же дистанция измеряется временем задержки, то маршрутизатор может измерить его с помощью специального пакета ECHO (эхо), в который получатель помещает время получения и который отправляет обратно как можно быстрее.

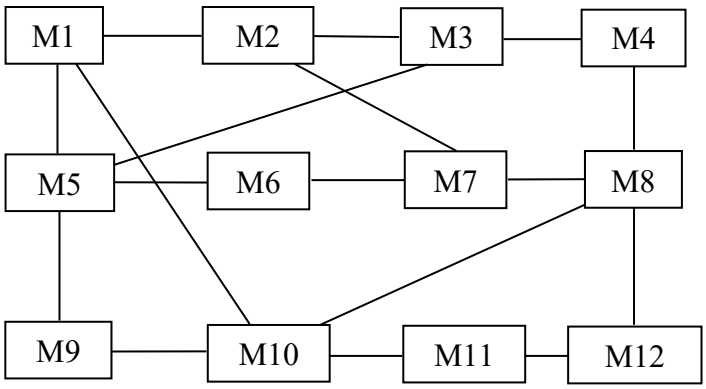
Предположим, что в качестве единицы измерения используется время задержки, и этот параметр относительно каждого из соседей известен маршрутизатору. Через каждые *T* миллисекунды все маршрутизаторы сети посылают своим соседям список с приблизительными задержками для каждого получателя. Они, в свою очередь, также получают подобный список от всех своих соседей. Допустим, одна из таких таблиц пришла от соседа M1, и в ней указывается, что время распространения от маршрутизатора M1 до маршрутизатора M_i равно T_i. Если маршрутизатор знает, что время пересылки до маршрутизатора M1 равно *tn,* тогда задержка при передаче пакета маршрутизатору M_i через маршрутизатор M составит $M_t + m$. Выполнив такие расчёты для всех своих соседей, маршрутизатор может выбрать наилучшие пути и поместить соответствующие записи в новую таблицу. Старая таблица в расчётах не используется.

Процесс обновления таблицы проиллюстрирован на рис. 4.7. Сверху показана подсеть (рис. 4.7, *а*). Первые четыре столбца (рис. 4.7, *б*) показывают векторы задержек, полученные маршрутизатором M10 от своих соседей.

Маршрутизатор M1 считает, что время пересылки от него до маршрутизатора M2 равно 12 мс, 25 мс – до маршрутизатора M3, 40 мс – до M4 и т. д. Предположим, что маршрутизатор измерил или оценил задержки до своих соседей M1, M9, M8 и M11 как 8, 10, 12 и 6 мс соответственно.

Теперь рассмотрим, как M10 рассчитывает свой новый маршрут к маршрутизатору M7. Он знает, что задержка до M1 составляет 8 мс, а M1 думает, что от него до M7 данные дойдут за 18 мс. Таким образом, M10 знает, что если он станет отправлять пакеты для M7 через M1, то задержка составит 26 мс. Аналогично он вычисляет

значения задержек для маршрутов от него до M7, проходящих через остальных его соседей (M9, M8 и M11), и получает соответственно 41 = 31 + 10, 18 = 6 + 12 и 37 = 31 + 6.



а

	M1	M9	M8	M11	Новая расчётная задержка для M10	
					T	Линия
M1	0	24	20	21	6	M1
M2	12	36	31	28	20	M1
M3	25	18	19	36	28	M9
M4	40	27	8	24	20	M8
M5	14	7	30	22	17	M9
M6	23	20	19	40	30	M9
M7	18	31	6	31	18	M8
M8	17	20	0	19	12	M8
M9	21	0	14	22	10	M9
M10	9	11	7	10	0	-
M11	24	22	22	0	6	M11
M12	29	33	9	9	15	M12
	Задержка M10-M1 = 8	Задержка M10-M9 = 10	Задержка M10-M8 = 12	Задержка M10-M11 = 6	Новая таблица маршрутов для M10	

Векторы, полученные от четырёх соседей M10

б

Рис. 4.7. Построение таблицы маршрутизации:
а – подсеть; *б* – полученные от M1, M9, M8, M11 векторы и новая таблица маршрутов для M10

Лучшим значением очевидно является 18, поэтому именно оно помещается в таблицу в запись для получателя M7. Вместе

с числом 18 туда же помещается обозначение линии, по которой проходит самый короткий маршрут до M7, то есть M8. Данный метод повторяется для всех остальных адресатов, и при этом получается новая таблица, показанная в виде правого столбца на рис. 4.7, *б*.

4.11. Маршрутизация с учётом состояния линий

Маршрутизация на основе векторов расстояний использовалась в сети ARPANET вплоть до 1979 г., после чего её сменил алгоритм маршрутизации с учётом состояния линий. Отказаться от прежнего алгоритма пришлось по двум причинам. Во-первых, старый алгоритм при выборе пути не учитывал пропускную способность линий. Пока все линии имели пропускную способность 56 кбит/с, в её учёте не было необходимости. Однако стали появляться линии со скоростью 230 кбит/с, а затем и 1,544 Мбит/с, и не принимать во внимание пропускную способность стало невозможно. Конечно, можно было ввести пропускную способность в качестве множителя для единицы измерения, но имелась ещё и другая проблема, заключавшаяся в том, что алгоритм слишком долго приходил к устойчивому состоянию (непредсказуемость событий обрыва и восстановления каналов между маршрутизаторами, а также временных рамок протекания этих процессов и оперативного доведения этой информации до маршрутизаторов). Поэтому старый алгоритм был полностью заменён новым алгоритмом, который называется сейчас маршрутизацией с учётом состояния линий [3]. Варианты этого алгоритма широко применяются в наши дни.

В основе алгоритма лежит простая идея, которую можно изложить в *пяти требованиях к маршрутизатору.* Каждый маршрутизатор должен:

1. Обнаруживать своих соседей и узнавать их сетевые адреса.
2. Измерять задержку или стоимость связи с каждым из своих соседей.
3. Создавать пакет, содержащий всю собранную информацию.
4. Посылать созданный пакет всем маршрутизаторам.
5. Вычислять кратчайший путь ко всем маршрутизаторам.

В результате, каждому маршрутизатору высылаются полная топология и все измеренные значения задержек.

4.12. Объединение различных сетей

До сих пор неявно предполагалось наличие единой однородной сети, в которой каждая машина использует один и тот же протокол на всех уровнях. Но существует множество различных сетей, включая локальные, региональные и глобальные. Наличие различных сетей всегда приводит к возникновению различных протоколов [3].

Есть основание предполагать, что разнообразие сетей (а следовательно, и протоколов) будет оставаться всегда по следующим причинам. Прежде всего, установленная база существующих сетей уже достаточно велика и продолжает расти. Почти все персональные компьютеры используют протокол TCP/IP. Во многих больших компаниях ещё остались мейнфреймы, использующие протоколы SNA фирмы «IBM». Существенная доля телефонных сетей ориентирована на ATM. Локальные сети персональных компьютеров все еще пользуются протоколами Novell NCP/IPX или AppleTalk. Наконец, беспроводные сети – эта бурно развивающаяся сегодня область – внедряют свои протоколы. Такая тенденция будет сохраняться в ближайшие годы благодаря наличию большого количества существующих сетей и ещё благодаря тому, что некоторые производители считают, что возможность клиента легко переходить в системы других производителей не в их интересах.

Во-вторых, по мере того как компьютеры и сети становятся всё дешевле, уровень принятия решения о выборе той или иной технологии всё опускается и опускается, и теперь уже этим занимаются организации, желающие установить у себя сеть. Многие компании придерживаются политики разграничения полномочий в зависимости от стоимости технологий. При существенном удешевлении систем передачи с набором различных протоколов решение о выборе системы будет приниматься на уровне руководителей среднего и низшего звена без согласования с вышестоящими руководителями. Такая политика может легко привести к тому, что в различных отделах одной и той же организации будут установлены не согласованные между собой рабочие станции, ориентированные на несовместимые друг с другом протоколы.

В-третьих, различные сети (например, ATM и беспроводные сети) основаны на принципиально разных технологиях, поэтому вряд ли

стоит удивляться, что с появлением нового оборудования появится и новое программное обеспечение для него. Например, средний дом сейчас напоминает офис, каким он был десять лет назад: жилище битком набито разнообразными компьютерами (стационарными, ноутбуками и карманными), не соединёнными друг с другом. В будущем, возможно, будет нормой объединять в единую сеть телефон, телевизор и другую бытовую технику, так, чтобы ею можно было управлять дистанционно. Появление этой новой технологии, несомненно, повлечёт создание новых протоколов.

Рассмотрим следующий пример взаимодействия нескольких различных сетей (рис. 4.26). Показана корпоративная сеть, части которой находятся далеко друг от друга и соединены глобальной сетью ATM. В одной из частей для объединения Ethernet, беспроводной локальной сети 802.11 и мейнфреймовой сети SNA корпоративного центра данных используется оптическая магистраль FDDI.

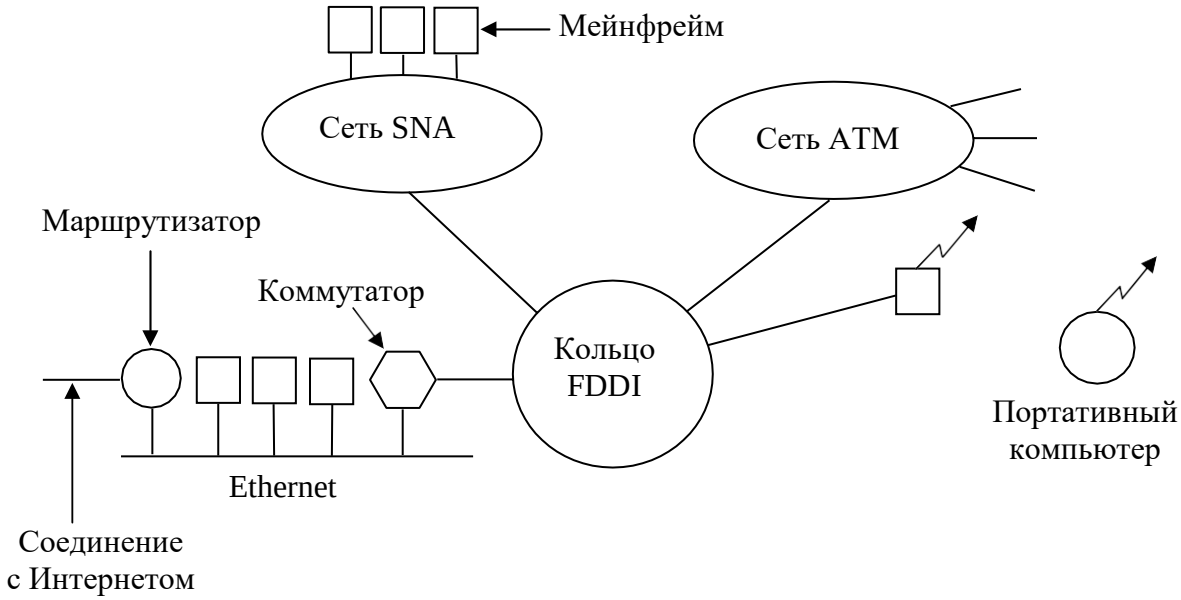


Рис. 4.26. Набор объединённых сетей

Целью объединения этих сетей является предоставление пользователям возможности общаться с пользователями любой другой из этих сетей, а также получать доступ к данным всех частей сети. Для реализации этих возможностей требуется посылать пакеты из одной сети в другую. Поскольку сети зачастую различаются довольно сильно, это действие осуществить не так уж просто.

Сети могут отличаться друг от друга довольно сильно и по разным параметрам. Некоторые из параметров, такие, как методы модуляции

или форматы кадров, нас сейчас не интересуют, поскольку они относятся к физическому уровню и уровню передачи данных. На сетевом уровне могут быть различия по следующим параметрам:

- предлагаемому сервису (ориентированные на соединение или не требующие соединения);
- протоколам (IP, IPX, SNA, ATM, MPLS, AppleTalk и др.);
- адресации (плоская (802) или иерархическая (IP));
- многоадресной рассылке (присутствует или отсутствует, а также широковещание);
- размеру пакета (у каждой сети есть свой максимум);
- качеству обслуживания (много разновидностей);
- обработке ошибок (надёжная, упорядоченная и хаотичная доставка);
- управлению потоком (скользящее окно, управление скоростью, другое или никакого);
- борьбе с перегрузкой («дырявое ведро», «маркерное ведро», сдерживающие пакеты, нерегулярное раннее обнаружение);
- безопасности (правила секретности, шифрование, кодирование, аутентификация);
- параметрам (различные тайм-ауты, спецификация потока);
- тарификации (по времени соединения, за пакет, побайтно или никак).

Именно сглаживание этих различий делает обеспечение работы объединённой сети значительно более сложным делом, чем обеспечение работы одной сети.

Когда пакетам данных приходится пересекать несколько сетей, отличных от исходной сети, может возникнуть много проблем, связанных с интерфейсами между сетями. Во-первых, когда пакеты из ориентированной на соединение сети должны пересечь не требующую соединений сеть, их порядок может нарушиться, причём для отправителя это может оказаться неожиданностью, а получатель может оказаться не подготовленным к такому событию. Часто будет требоваться преобразование протоколов, что может быть непросто, если необходимая функциональность не может быть выражена. Также понадобится преобразование форматов адресов, что может потребовать создания некой разновидности системы каталогов. Передача многоадресных пакетов через сеть, не поддерживающую многоадресную рассылку, потребует формирования отдельных пакетов для каждого адресата.

Различия в максимальном размере пакетов в разных сетях составляют главную проблему. Как передать 8000-байтовый пакет по сети, в которой максимальный размер пакета равен 1500 байтам? При передаче пакета с обязательствами доставки в реальном масштабе времени по сети, не предоставляющей каких-либо гарантий работы в реальном времени, возникает проблема разницы в качестве обслуживания.

Методы обработки ошибок, управления потоком и борьбы с перегрузкой часто различаются в разных сетях. Если отправитель и получатель ожидают, что все пакеты будут доставлены без ошибок и с сохранением порядка, а сеть просто игнорирует пакеты, когда ей угрожает перегрузка, или пакеты, направляясь различными путями, приходят к получателю совсем не в том порядке, в каком они были отправлены, то многие приложения просто не смогут работать в таких условиях. Различия в механизмах безопасности, установке параметров, правилах тарификации и даже в законах, охраняющих тайну переписки, могут послужить причиной многих проблем.

4.13. Способы объединения сетей

Сети могут объединяться с помощью разных устройств [3]. На физическом уровне сети объединяются повторителями или концентраторами, которые просто переносят биты из одной сети в другую такую же сеть. Чаще всего это аналоговые устройства, не имеющие отношения к цифровым протоколам (регенераторы сигналов).

На канальном уровне объединение идёт с помощью мостов и коммутаторов. Они могут принимать кадры, анализировать их MAC-адреса, направлять их в другие сети, осуществляя по ходу дела минимальные преобразования протоколов, например, из Ethernet в FDDI или в 802.11.

На сетевом уровне есть маршрутизаторы, соединяющие две сети. Если сетевые уровни у них разные, маршрутизатор может обеспечить перевод пакета из одного формата в другой, хотя такие преобразования сейчас выполняются всё реже. Маршрутизатор может поддерживать несколько сетевых протоколов, тогда он называется мультипротокольным маршрутизатором.

На транспортном уровне существуют транспортные шлюзы, предоставляющие интерфейсы для соединений своего уровня. Транс-

портный шлюз позволяет, к примеру, передавать пакеты из сети TCP в сеть SNA (протоколы транспортного уровня у них различаются), склеивая одно соединение с другим.

Наконец, на прикладном уровне шлюзы осуществляют преобразование семантики сообщений. Например, шлюзы между электронной почтой Интернета (RFC 822) и электронной почтой X.400 должны анализировать содержимое сообщений и изменять различные поля электронного конверта.

На рис. 4.27 показано отличие объединения на сетевом уровне от объединения на уровне передачи данных (канальном). Источник *S* пытается послать пакет приёмнику *D*. Эти две машины работают в разных сетях Ethernet, соединённых коммутатором. Источник *S* вставляет пакет в кадр и отправляет его. Кадр прибывает на коммутатор, который по MAC-адресу определяет, что его надо переслать в ЛВС-2. Коммутатор просто снимает кадр с ЛВС-1 и передаёт его в ЛВС-2.

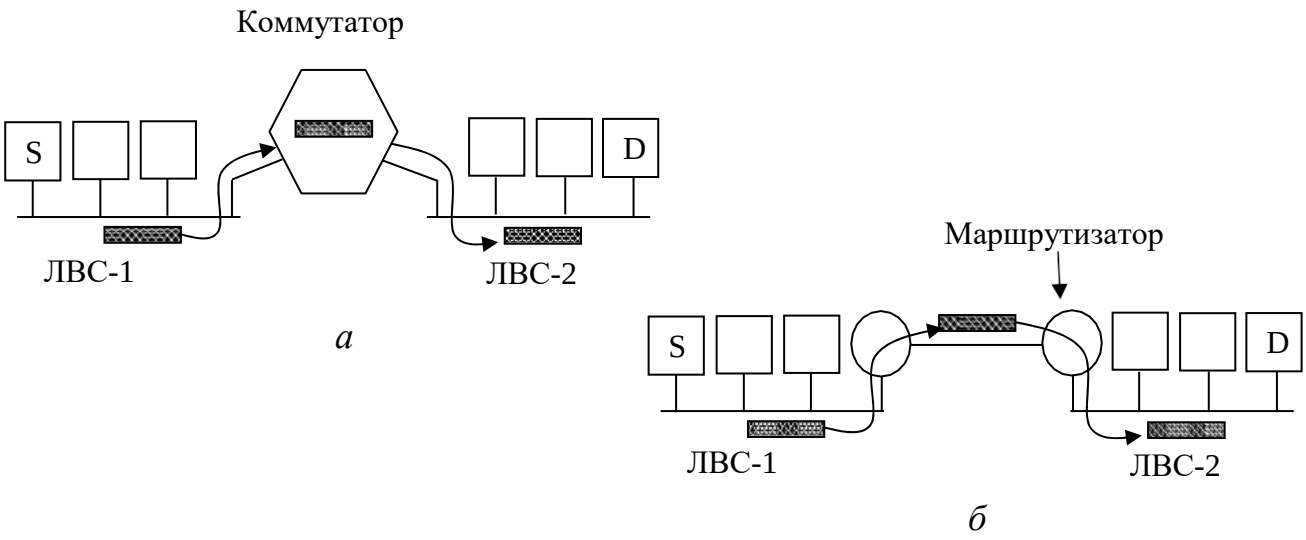


Рис. 4.27. Объединение сетей:
а – коммутатором; *б* – маршрутизатором

Теперь рассмотрим ту же ситуацию, но с применением другого сетевого оборудования. Допустим, две сети Ethernet объединены не коммутатором, а парой маршрутизаторов. Маршрутизаторы между собой соединены двухточечной линией, которая может представлять собой, например, выделенную линию длиной в тысячи километров. Что в данном случае будет происходить с кадром? Он принимается маршрутизатором, из его поля данных извлекается пакет. Далее маршрутизатор анализирует содержащийся в пакете адрес (например,

IP-адрес). Этот адрес нужно отыскать в таблице маршрутизации. В соответствии с ним принимается решение об отправке пакета (возможно, упакованного в кадр нового вида – это зависит от протокола, используемого линией) на удалённый маршрутизатор. На противоположном конце пакет вставляется в поле данных кадра Ethernet и помещается в ЛВС-2.

В чём заключается основная разница между процессами коммутации (установки моста) и маршрутизации? Коммутатор (мост) пересылает весь пакет, обосновывая своё решение значением MAC-адреса. При применении маршрутизатора пакет извлекается из кадра, и для принятия решения используется адрес, содержащийся именно в пакете. Коммутаторы не обязаны вникать в подробности устройства протокола сетевого уровня, с помощью которого производится маршрутизация, а маршрутизаторы работают именно по этому протоколу.

Выводы по главе 4

На сетевом уровне решается задача разработки маршрутов доставки пакетов от отправителя к получателю. Здесь требуется создание алгоритмов оптимальной маршрутизации с использованием многих транзитных участков сети. Другая задача уровня – корректное сопряжение пользователей, находящихся в различных сетях.

На этом уровне основным действующим звеном сети выступает маршрутизатор, связанный каналами с такими же устройствами. Стоит задача эффективного управления этим звеном.

Сетевой уровень предоставляет транспортному уровню сервисы с установлением и без установления соединения. Причём основная функция сетевого уровня заключается в выборе маршрута для пакетов от начальной до конечной точки. В большинстве сетей пакетам приходится проходить через несколько маршрутизаторов. Алгоритм сетевой маршрутизации должен отвечать требованиям оптимальности с точки зрения скорости доставки сообщений, равномерности загруз-

ки каналов, гибкости реакции на изменения трафика во времени, а также обладать устойчивостью.

Задача оптимальности решается исходя из конкретной обстановки по составу и поведению сети и выбранного критерия. Сетевые алгоритмы выбора маршрута могут быть адаптивными и неадаптивными. Первые изменяют решение о выборе маршрутов при изменении топологии сети и также часто – в зависимости от загруженности линий. Вторые не учитывают при выборе маршрута топологию и текущее состояние сети и не изменяют трафик на линиях. Выбор маршрута для каждой пары пользователей производится заранее, в автономном режиме (статическая маршрутизация).

Метод лавинной маршрутизации позволяет наилучшим образом обеспечить гарантированность доставки пакетов сообщения адресату. Но часто данный метод не является оптимальным, поскольку происходит лавинообразное размножение пакетов на параллельных ветвях сети. Сеть перегружается циркулирующей информацией с многократным дублированием. И даже после фактического получения адресатом всего сообщения в сети продолжают перемещаться пакеты данных, создавая перегрузку и высокую вероятность ошибочного повторного приёма информации (одна из разновидностей сетевых атак). Следовательно, необходим контроль прохождения информации через каждый маршрутизатор путём регистрации номеров (или иных меток) пакетов, что требует наличия в нём элементов памяти. Для устранения упомянутых недостатков применяется выборочная заливка в конкретных приложениях.

Маршрутизация по вектору расстояний и с учётом состояния линии позволяет оптимально решать задачу доставки сообщений. Но требуется применять «интеллектуальные» сетевые маршрутизаторы, способные самостоятельно, по заданной программе, анализировать ситуацию в каналах связи, поддерживать взаимодействие с соседями (другими маршрутизаторами) и принимать решение о переключениях или блокировании сетевого трафика. Для взаимодействия с соседями маршрутизатор должен создавать пакеты служебной информации, где содержится информация о необходимом оповещении и постановка текущих задач по обработке конкретного пакета.

В больших, сложных, распределённых сетях целесообразно вводить иерархическую маршрутизацию, чтобы избежать проблем, связанных с увеличением памяти для таблиц маршрутов и времени обработки всей этой служебной информации процессором. В определён-

ный момент сеть может вырасти до таких размеров, при которых перестанет быть возможным хранение на маршрутизаторах записи обо всех остальных маршрутизаторах. Поэтому в больших сетях маршрутизация пакетов должна осуществляться иерархически. Маршрутизаторы разбиваются на отдельные регионы, которые определяются им как зоны ответственности. В своём регионе каждый маршрутизатор хранит в памяти все детали подсети, освобождаясь от необходимости знать подробности топологии соседних зон.

Многоадресная рассылка предполагает выбор оптимальных, наиболее коротких путей и информирование об этом всех маршрутизаторов, которые предполагается задействовать на данной траектории. Если участники информационного обмена находятся в движении, это ведёт к дополнительным трудностям маршрутизации. Мобильные хосты привносят новое усложнение в и без того непростое дело выбора маршрутов в различных вычислительных сетях – чтобы направить пакет к мобильному хосту, его нужно сначала найти. Здесь актуален вопрос о средствах и возможностях привязки мобильных абонентов к точкам доступа стационарной сети и, разумеется, о механизме их адресации. Следует учитывать также множество дестабилизирующих факторов, связанных с различными условиями и физическими средами распространения информационных сигналов.

Для выбора оптимального маршрута пакетов узлы коммутации направляют запросы по сети широковещательным способом. Запросы должны иметь стандартизированный формат, чтобы они были понятны устройствам различных подсетей. Готовые к работе узлы (маршрутизаторы) должны сообщать в ответ свои состояния и наличие связей. Для принятия решения важен временно́й интервал исполнения следующей части цикла управления: запрос – сбор информации – ответ. Одновременно должно быть ограничено время жизни каждого такого пакета служебной информации, поскольку она объективно быстро устаревает.

Производительность сети может снижаться из-за перегрузки – превышения количеством одновременно циркулирующих пакетов некоего порогового уровня. Наличие перегрузки означает, что нагрузка временно превысила возможности ресурсов данной части системы. В особо напряжённые моменты может иметь место ситуация полной блокировки сети по причине невозможности обеспечения информационного обмена. Известно, что одновременное обращение нескольких объектов к одному ресурсу ведёт к конфликтным ситуациям. То-

гда в сети начинается не только задержка, но и потеря пакетов дан- ных. Когда число пакетов достигает максимального уровня, производительность сети начинает снижаться. При очень высоком уровне трафика производительность сети падает до совсем низкого уровня и практически никакие пакеты не доставляются. Причины перегрузки: ненормативное (превышающее расчётное) увеличение трафика, недостаточная память для буферизации пакетов, недостатки программного обеспечения, не предусматривающего ограничение времени жизни пакетов и тайм-ауты на их повторную передачу, медленные процессо- ры, некорректная работа интерфейсов, которые не способны согласо- вать скорости и форматы взаимодействия различных элементов сети.

В главе рассмотрены различные принципы, стратегии, методы и алгоритмы борьбы с перегрузками. Здесь следует отметить, что применение комбинаций этих методов и алгоритмов может дать значительно больший эффект, чем применение только одного из них. В любом случае, решение принимается исходя из принципов целесообразности и разумной достаточности.

Вопросы для самопроверки

1. Задачи, решаемые на сетевом уровне ВОС.
2. Виды коммутации пакетов и их характеристики.
3. В чём заключается понятие оптимальности маршрута?
4. В чем состоит основное различие между ЛВС и ГВС?
5. Принципы выбора кратчайшего пути пакета.
6. Что такое вектор расстояний?
7. Поясните смысл понятия «идентификатор соединений».
8. Базовые требования к маршрутизаторам.
9. Что такое иерархическая маршрутизация?
10. Как организуется широковещательная и многоадресная рассылка?
11. Опишите процедуру регистрации в сети для мобильных объектов.
12. Форматы пакетов запроса маршрута и наличия маршрута.
13. Причины возникновения перегрузки в сети.
14. Перечислите методы борьбы с перегрузками.
15. Алгоритмы «дырявого ведра» и «маркерного ведра».
16. Как происходит диспетчеризация пакетов?
17. Проблемы взаимодействия сетей при их объединении.
18. Необходимость и способы объединения сетей.