# Сети передачи данных. Методы доступа

Полагаю, что идея компьютерных сетей заимствована из практики использования магистрально модульных систем, которые служат для передачи данных от различных источников информации в ЭВМ и обратно в режиме реального времени. Такие системы стали активно развиваться с начала 60-х годов прошлого века, наиболее характерным примером может служить система САМАС, где для передачи данных от или к большому числу объектов использовалась общая аппаратная транспортная среда (шина, как правило, параллельная). Строго говоря, локальная сеть ЭВМ является частным случаем магистрально модульной системы. Другим источником вдохновения могли служить каналы связи ЭВМ с периферийными приборами (например, дисковыми запоминающими устройствами или удаленными терминалами, достаточно вспомнить канал SCSI, который был разработан в том числе и для целей сбора данных). Следует отметить принципиальное отличие между такими системами и сетями. В последнем случае данные передаются от ЭВМ к ЭВМ. Важным фактором для развития сетей явилась разработка пакетного принципа передачи данных, который был разработан еще до начала второй мировой войны. Прототипом современных сетей, возможно, явились терминальные сети крупных вычислительных центров.

Первой сетью, где применен пакетный принцип передачи данных, была **ARPANET** (1969; Advanced Research Project Agency NETwork). Эта сеть имела всего 4 узла. Примерно тогда же стали разрабатываться протоколы последовательной, синхронной (SDLC) и асинхронной (старт/стоп) передачи данных, интеллектуальные терминалы и канальные концентраторы. В рамках этих работ были разработаны устройства и программы доступа к среде с разделением по времени (**TDM** - Time Division Multiplexing). Такая схема доступа используется практически во всех многозадачных операционных системах. Все это создало предпосылки для разработки реальных сетей. Целью строительства сетей является эффективное использование ресурсов машин, объединяемых сетью, и повышение надежности системы в целом. Через сеть передаются тексты, сообщения, файлы, изображения, задания, команды, видео или акустические данные. Попытаемся ответить на вопрос, почему сети получили столь широкое распространение? Сети делают возможным:

* Доступ к общим ресурсам (быстродействующая печать, диски большой емкости, backup-системы, информационные хранилища, серверы и т.д.).
* Децентрализацию вычислительного процесса, возможность создания распределенных вычислительных систем (системы **GRID**), повышение надежности систем за счет резервирования
* Повышение скорости обмена за счет реализации нескольких параллельных информационных потоков (файлообменные сети **Р2Р**)
* Информационный обмен в издательствах, информационных агентствах, поисковых системах, между обычными людьми, разбросанными по всему миру, в системах сбора и обработки научных, метео- и геофизических данных и т.д.
* Распределенное управление
* Всемирные и локальные системы межличностного общения (почта, IСQ, SMS и пр.)

|  |
| --- |
| *Сети - это системы мультиплексирования доступа к каналам и ресурсам.* |
|  |

Это достаточно обобщенный и неполный список приложений, полный список можно было бы издать отдельным томом. Сети использованы для расчета генома человека, проверки надежности ключей шифрования. Фермы ЭВМ, объединенных сетью, используются вместо супер-ЭВМ для обработки больших объемов экспериментальных данных в физике высоких энергий, причем кластеры этих машин размещаются иногда на нескольких континентах. Локальные сети появились в поездах, лайнерах, автомобилях и жилище человека.

Сети по своей принадлежности делятся на локальные (**LAN** - Local Area Network), городские (**MAN** - Metropolitan Area Network), региональные (**WAN** - Wide Area Network) и всемирные (Интернет). В настоящее время существует огромное разнообразие сетей. Но есть у них и нечто общее. Практически все они базируются на пакетном принципе передачи данных. Пакет представляет собой последовательность нулей и единиц. Нулю и единице (часто их называют логическим нулем и логической единицей) ставится в соответствие определенный уровень амплитуды или знак перепада. В этой связи во всех сетях приходится решать проблему выделения начала и конца пакетов. Как правило, для этой цели используются уникальные последовательности бит или уникальные коды. При этом возникает проблема, когда подобные сигнатуры встречаются в теле пакета, ведь это может вызвать сбой - система по ошибке может принять такую последовательность, как сигнал прерывания обработки пакета. Задача решается с использованием esc-последовательностей или техники бит-стаффинга. Esc-последовательность представляет собой строку из двух или более символов, первый из которых, как правило, имеет код 27 (десятичный; esc). Строка имеет такой формат, который не может встретиться в пределах пакета, если же такая последовательность все же встречается, включается алгоритм, обеспечивающий ее преобразование к уникальному виду. Такая последовательность служит для подмены сигнатуры начала или конца пакета, если она встретится в его поле данных. Техника бит-стафинга предполагает добавление в определенных ситуациях нулевого или единичного бита или использование определенных комбинаций сигналов на физическом уровне. Рассмотрим пример сети ISDN, где для выделения начала и конца кадра служит байт 01111110. Если внутри кадра встретится такой байт, то он будет заменен последовательностью 011111010. Приемное устройство преобразует эту последовательность назад в 01111110. Так входная последовательность бит 010011111111111111110010 будет преобразована в

010011111**0**11111**0**11111**0**10010,

где выделенные нули являются вставленными специально, чтобы была невозможна имитация сигнатуры конца кадра. Последовательность стала длиннее. Принимающая сторона восстановит исходную строку бит.

При передаче данных нужно обеспечивать синхронизацию и стабилизацию уровня регистрации. Синхронизация между передатчиком и приемником необходима, прежде всего, потому, что не каждому биту в передаваемом коде соответствует изменение уровня сигнала. А это может приводить к тому, что одна из сторон передаст N бит, а другая при этом будет считать, что приняла N-1 или N+1 бит. Синхронизация тактовых частот передатчика и приемника чаще всего осуществляется самой кодовой последовательностью, транспортирующей данные, или независимо, как, например, это делается в сетях SDH.

Так как между передатчиком и приемником не всегда имеется гальваническая связь, средний уровень сигнала может дрейфовать в зависимости от числа предыдущих нулей или единиц. Такой дрейф может приводить к ошибкам, так как в суперпозиции с шумами или наводками может превысить уровень дискриминации, отделяющий состояния логического нуля и единицы.

|  |
| --- |
| *Практически все сети, так или иначе, решают проблему****мультиплексирования****данных (как правило, это определяет алгоритм доступа к сетевой среде).* |
|  |

В сетях для формирования виртуального канала и транспортировки данных используются схемы **с коммутацией каналов** и **с коммутацией пакетов**. Эти схемы представлены на рис. 4.1. В случае коммутации каналов сначала осуществляется формирование связи между инициатором и адресатом и только затем начинается обмен данными (или разговор при телефонном соединении). Канал может реализоваться из медных проводов, оптических волокон или с помощью радио. В качестве параметра при установлении соединения используется код места назначения, например, его телефонный номер или IP-адрес. При наборе первой цифры выбирается номер выхода первого коммутатора, при наборе второй цифры, номер выходного канала второго коммутатора и т.д. Цифры могут группироваться в блоки. Время формирования канала достигает 10 секунд и не зависит от расстояния между клиентами. Зато после сформирования канала характерная задержка передачи данных не превышает 5 мсек на 1000 км.

|  |
| --- |
| *Сформированный виртуальный канал используется только конечными клиентами и полоса пропускания канала остается постоянной в течение сессии,****неиспользованный ресурс канала пропадает****.* |
|  |

Принцип коммутации каналов применяется системами ISDN, ATM и Х.25. Это не означает, что там ни при каких обстоятельствах не используется коммутация пакетов. Приведенное выше замечание о постоянстве полосы пропускания не исключает, тем не менее, ситуаций перегрузки. Реальные коммутаторы, в отличие от нарисованных, имеют многие сотни и даже тысячи входных/выходных каналов. При поступлении данных по большому числу каналов одновременно коммутатор может не справиться с такой ситуацией. В случае коммутации пакетов один и тот же канал может использоваться большим числом пользователей одновременно. Здесь перегрузка является скорее нормой, чем исключением. В современных системах часто совмещаются методы коммутации каналов и пакетов. Ниже на рисунке представлены варианты сетей с коммутацией каналов (сверху) и пакетов (внизу).

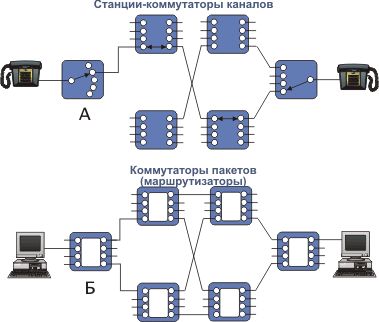


Рис. 4.1.

Помимо названных режимов существует также система ***коммутации сообщений***. Наистарейшим примером такого рода является система передачи телеграмм, где очередная станция получает телеграмму в виде перфоленты, оператор (или специализированный коммутатор) считывает ее адрес и принимает решение о том, по какому из каналов ее следует передать дальше. Такие системы называются ***“запомнить-и-переадресовать”***. В современных сетях такой принцип используется при передаче новостей (протокол **NNTP** близкий родственник **e-mail**). При передаче сообщений не регламентируется их размер, что вынуждает в точках коммутации записывать их на диски. Отсутствие такого рода ограничений приводит к тому, что канал может быть занят достаточно долго. Новое сообщение может быть послано лишь после завершения отправки предыдущего. Все эти проблемы легко разрешаются в системах с коммутацией пакетов, размер которых регламентирован. В этом методе задержка доставки пакета может варьироваться в довольно широких пределах. Более того, может меняться порядок доставляемых пакетов и кадр, посланный раньше, может быть доставлен позже, принимающая сторона должна уметь обрабатывать такие события. Пакетный способ транспортировки легко адаптируется к любым форматам, скоростям передачи и алгоритмам формирования и обработки пакетов. Здесь, как правило, исключается возможность монополизации канала каким-то пользователем, система допускает кодирование или преобразование данных в пакете. Емкость буфера определяется максимальным размером пакета. Для буферизации может использоваться оперативная память маршрутизатора. В данной схеме решение о переадресации может приниматься до того, как пакет будет принят коммутирующим устройством целиком (режим **cut-through** - коммутация на пролете).

Данный прием реализован в коммутаторах ATM, именно по этой причине там контрольное суммирование охватывает лишь заголовок, а не всю ячейку, хотя ее протяженность ограничена. Ведь именно в заголовке содержится адрес места назначения, а ошибка в нем приведет к тому, что истинный адресат не получит пакет, за то какое-то другое устройство получит пакет, ему не предназначенный.

Но даже 155 Мбит/с быстродействие АТМ-коммутаторов не избавляет их от перегрузок. Достаточно попытаться представить, что на вход 1000-канального переключателя пришло одновременно 100 ячеек, а это ординарная ситуация и здесь может не спасти даже достаточно большие объемы буферов и изощренные системы конвейерной обработки. Недаром АТМ-коммутаторы так дороги!

Разные участки маршрута могут иметь разные предельные длины пакетов (MTU) и это может требовать фрагментации получаемых пакетов. Значение MTU как правило определяется вероятностью ошибки при транспортировке (BER) или предельно допустимым временем занятости канала.

|  |
| --- |
| *Именно гибкая система фрагментации и дефрагментации пакетов сделало сети Интернет столь универсальными.* |
|  |

Как локальные, так и региональные сети могут отличаться по **топологии и методу доступа**.

**Топология**

Среди топологических схем наиболее популярными являются (см. рис. 4.1):

1. Шина
2. Звезда
3. Кольцо
4. Многокаскадные и многосвязные сети (См. раздел 4.1.10; [41\ban\_4110.doc](http://book.itep.ru/4/41/ban_4110.htm))

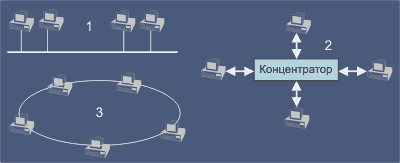


Рис. 4.2. Примеры сетевых топологий

К первым трем типам топологии относятся 99% всех локальных сетей. Наиболее популярный тип сети - Ethernet, может строиться по схемам 1 и 2. Вариант 1 наиболее дешев, так как требует по одному интерфейсу на машину и не нуждается в каком-либо дополнительном оборудовании. Сети Token Ring и FDDI используют кольцевую топологию (3 на рис. 4.2), где каждый узел должен иметь два сетевых интерфейса. Эта топология удобна для оптоволоконных каналов, где сигнал может передаваться только в одном направлении (но при наличии двух колец, как в FDDI, возможна и двунаправленная передача). Нетрудно видеть, что кольцевая топология строится из последовательности соединений точка-точка.

Используется и немалое количество других топологий, которые являются комбинациями уже названных. Примеры таких топологий представлены на рис. 4.3.

Вариант А на рис. 4.3 представляет собой схему с полным набором связей (все узлы соединены со всеми), такая схема используется только в случае, когда необходимо обеспечить высокую надежность соединений. Эта версия требует для каждого из узлов наличия n-1 интерфейсов при полном числе узлов n. Вариант Б является примером нерегулярной топологии, а вариант В - иерархический случай связи (древовидная топология).

Если топологии на рис. 4.2 чаще применимы для локальных сетей, то топологии на рис. 4.3 более типичны для региональных и глобальных сетей. Выбор топологии локальной или региональной сети существенно сказывается на ее стоимости и рабочих характеристиках. При этом важной характеристикой при однородной сети является среднее число шагов между узлами d. http://book.itep.ru/4/image113.gif, где nd - число ЭВМ на расстоянии d. n - полное число ЭВМ в сети; d - расстояние между ЭВМ. Для сети типа А на рис. 4.3 d=1. Сеть типа В характеризуется графом без циклических структур (дерево).

|  |
| --- |
| *Именно многосвязность в сочетании с динамическими протоколами маршрутизации делает Интернет достаточно надежным и устойчивым.* |
|  |

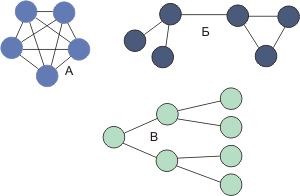


Рис. 4.3. Различные сетевые топологические схемы

Современные вычислительные системы используют и другие топологии: решетки (А), кубы (В), гипердеревья (Б), гиперкубы и т.д. (см. рис. 4.4). Но так как некоторые вычислительные системы (кластеры) базируются на сетевых технологиях, я привожу и такие примеры. В некоторых системах топология может настраиваться на решаемую задачу.

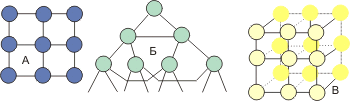


Рис. 4.4. Некоторые топологии вычислительных систем

**Метод доступа к сети**

Метод доступа определяет алгоритм, согласно которому узлы сети получают доступ к среде передачи данных, и осуществляют мультиплексирование/ демультиплексирование данных.

В 1970-х годах Норман Абрахамсон вместе с сотрудниками Гавайского университета предложил оригинальный способ решения проблемы доступа к сети, который был назван позднее ALOHA (Aloha на местном гавайском наречии означает приветствие). Этот алгоритм использовался для доступа к радио-каналу большого числа независимых пользователей. ALOHA разрешает всем пользователям передачу тогда, когда им это оказалось нужно. При этом неизбежны столкновения и искажения передаваемых данных. Алгоритм, благодаря обратной связи, позволяет отправителям узнать, были ли данные в процессе передачи искажены. Если зарегистрировано такое столкновение, все вовлеченные участники выжидают некоторое время и предпринимают повторную попытку. Время выдержки выбирается случайным образом, что делает повторные столкновения менее вероятными.

|  |
| --- |
| *Принципиальное отличие алгоритма ALOHA от CSMA/CD (используемого в Ethernet) с точки зрения столкновений заключается в том, что в первом случае столкновения детектируются на входе получателя, а во втором - на выходе отправителя.* |
|  |

Мультиплексирование можно осуществлять **по частоте** (по длине волны - **FDM**), предоставляя разным клиентам разные частотные диапазоны или **по времени (TDM)**, разрешая доступ клиентов к сетевой среде по очереди и резервируя каждому из них для работы фиксированные последовательные временные интервалы. При этом необходима синхронизация работы всех участников процесса. В последнее время стало использоваться также мультиплексирование по кодам**CDMA** (Code Division Multiple Access), где каждому участнику выделяется уникальный чип-код и допускается использование всеми клиентами всего частотного диапазона в любой момент времени.

Большая часть современных сетей базируется на алгоритме доступа CSMA/CD (carrier sensitive multiple access with collision detection), где все узлы имеют равные возможности доступа к сетевой среде, а при одновременной попытке фиксируется столкновение и сеанс передачи повторяется позднее. Здесь нет возможности приоритетного доступа и по этой причине такие сети плохо приспособлены для задач управления в реальном масштабе времени. Некоторое видоизменение алгоритма CSMA/CD (как это сделано в сетях CAN или в IBM DSDB) позволяют преодолеть эти ограничения. Доступ по схеме CSMA/CD (из-за столкновений) предполагает ограничение на минимальную длину пакета.

По существу, метод доступа CSMA/CD предполагает широковещательную передачу пакетов (не путать с широковещательной адресацией). Все рабочие станции логического сетевого сегмента воспринимают эти пакеты хотя бы частично, чтобы прочесть адресную часть. При широковещательной адресации пакеты не только считываются целиком в буфер, но и производится прерывание процессора для обработки факта прихода такого пакета. Логика поведения субъектов в сети с доступом CSMA/CD может варьироваться. Здесь существенную роль играет то, синхронизовано ли время доступа у этих субъектов. В случае Ethernet такой синхронизации нет. В общем случае при наличии синхронизации возможны следующие алгоритмы.

**А.**

1. Если канал свободен, терминал передает пакет с вероятностью 1.
2. Если канал занят, терминал ждет его освобождения, после чего производится передача.

**Б.**

1. Если канал свободен, терминал передает пакет.
2. Если канал занят, терминал определяет время следующей попытки передачи. Время этой задержки может задаваться некоторым статистическим распределением.

**В.**

1. Если канал свободен, терминал с вероятностью р передает пакет, а с вероятностью 1-р он откладывает передачу на t секунд (например, на следующий временной домен).
2. При повторении попытки при свободном канале алгоритм не изменяется.
3. Если канал занят, терминал ждет пока канал не освободится, после чего действует снова согласно алгоритму пункта 1.

Алгоритм А на первый взгляд представляется привлекательным, но в нем заложена возможность столкновений с вероятностью 100%. Алгоритмы Б и В более устойчивы в отношении этой проблемы.

|  |
| --- |
| *Эффективность алгоритма CSMA зависит от того, как быстро передающая сторона узнает о факте столкновения и прерывает передачу, ведь продолжение бессмысленно - данные уже повреждены. Это время зависит от длины сетевого сегмента и задержек в оборудовании сегмента. Удвоенное значение задержки определяет минимальную длину пакета, передаваемого в такой сети. Если пакет короче, он может быть передан так, что передающая сторона не узнает о его повреждении в результате столкновения. Для современных локальных сетей Ethernet, построенных на переключателях и полнодуплексных соединениях, эта проблема неактуальна.* |
|  |

С целью пояснения этого утверждения рассмотрим случай, когда одна из станций (1) передает пакет самой удаленной ЭВМ (2) в данном сетевом сегменте. Время распространения сигнала до этой машины пусть равно Т. Предположим также, что машина (2) попытается начать передачу как раз в момент прихода пакета от станции (1). В этом случае станция (1) узнает о столкновение лишь спустя время 2Т после начала передачи (время распространения сигнала от (1) до (2) плюс время распространения сигнала столкновения от (2) к (1)). Следует учитывать, что регистрация столкновения это аналоговый процесс и передающая станция должна “прослушивать” сигнал в кабеле в процессе передачи, сравнивая результат чтения с тем, что она передает. Важно, чтобы схема кодирования сигнала допускала детектирование столкновение. Например, сумма двух сигналов с уровнем 0 этого сделать не позволит. Можно подумать, что передача короткого пакета с искажением из-за столкновения не такая уж большая беда, проблему может решить контроль доставки и повторная передача.

|  |
| --- |
| *Следует только учесть, что повторная передача в случае зарегистрированного интерфейсом столкновения осуществляется самим интерфейсом, а повторная передача в случае контроля доставки по отклику выполняется прикладным процессом, требуя ресурсов центрального процессора рабочей станции.* |
|  |

Сопоставление эффективности использования канала для различных протоколов произвольного доступа произведено на рис. 4.5.

|  |
| --- |
| *Здесь уместно обратить внимание на то, что если загрузка канала невелика, а требуется малая задержка доступа к каналу, то лучше алгоритма ALOHA трудно что-либо придумать. Напротив, когда задержка доступа несущественна, а загрузка канала велика, то следует использовать одну из разновидностей протокола доступа CSMA или один из протоколов, исключающих возможность столкновения кадров. Такие алгоритмы описаны ниже.* |
|  |

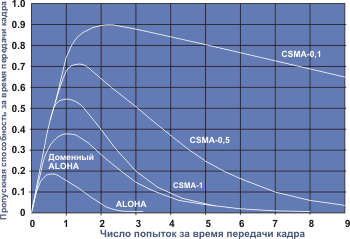


Рис. 4.5. Сравнение эффективности канала для различных протоколов произвольного доступа и разных значений **p**

Протокол доступа **CSMA** может предполагать, что когда канал оказывается свободным, а рабочая станция готова начать передачу, реальная пересылка кадра в рамках заданного временного домена происходит с определенной вероятностью **p**. С вероятностью **1-р** передача будет отложена до следующего временного домена. В следующем домене, если канал свободен с вероятностью **р** осуществится передача или будет отложена до следующего домена и так далее. Процесс продолжается до тех пор, пока кадр не будет передан. На рисунке эта вероятность **р** отмечена цифрами после аббревиатуры CSMA-. В случае Ethernet эта вероятность равна единице (CSMA-1) [46, Танненбаум, стр 302], то есть рабочая станция безусловно начнет передачу, если она к этому готова, а канал свободен. Из рисунка видно, что чем меньше эта вероятность, тем выше эффективность (среднее время доступа к сетевой среде также увеличивается). Очевидно, что все разновидности протокола доступа CSMA эффективнее протокола ALOHA в обеих его разновидностях. Связано это с тем, что ни одна станция не начинает передачу, если обнаружит, что сетевая среда занята. Существует еще одна разновидность протокола CSMA (**nonpersistent**), в которой сетевой субъект при готовности анализирует состояние сетевой среды и, если канал занят, возобновляет попытку спустя псевдослучайный интервал времени, то есть ведет себя так, будто произошло столкновение. Такой алгоритм повышает эффективность использования канала при существенном возрастании усредненной задержки доступа.

Рассмотрим, как можно избежать проблем столкновений пакетов. Пусть к сетевому сегменту подключено **N** рабочих станций. После передачи любого пакета выделяется N временных интервалов. Каждой подключенной к сетевому сегменту машине ставится в соответствие один из этих интервалов длительностью **L**. Если машина имеет данные и готова начать передачу, она записывает в это интервал бит, равный 1. По завершении этих N интервалов, рабочие станции по очереди, определяемой номером приписанного интервала, передают свои пакеты (см. рис. 4.6, N=8).

http://book.itep.ru/4/bitmap_4_4.gif

Рис. 4.6. Схема реализации протокола доступа к сети **bit-map**

В примере на рисунке 4.6 сначала право передачи получают станции 0, 2 и 6, а в следующем цикле 2 и 5 (пересылаемые пакеты окрашены в серый цвет). Если рабочая станция захочет что-то передать, когда ее интервал (домен) уже прошел, ей придется ждать следующего цикла. По существу данный алгоритм является протоколом резервирования. Машина сообщает о своих намерениях до того, как начинает что-либо передавать. Чем больше ЭВМ подключено к сетевому сегменту, тем больше временных интервалов L должно быть зарезервировано и тем ниже эффективность сети. Понятно, что эффективность растет с ростом L.

Вариация данного алгоритма доступа реализована в сетях сбора данных реального времени CAN (Controller Area Network - http://www.kvaser.se/can/protocol/index.htm - алгоритм доступа CSMA/CA - collision avoidance - с исключением столкновений). Там в указанные выше интервалы записывается код приоритета рабочей станции. Причем станции должны быть синхронизованы и начинать запись своего уникального кода одновременно, если все или часть из них готовы начать передачу. Эти сигналы на шине суммируются с помощью операции “проволочное ИЛИ” (если хотя бы один из участников выставляет логическую единицу, на шине будет низкий уровень). В процессе побитового арбитража каждый передатчик сравнивает уровень передаваемого сигнала с реальным уровнем на шине. Если эти уровни идентичны, он может продолжить, в противном случае передача прерывается и шина остается в распоряжении более приоритетного кадра. Для пояснения работы алгоритма предположим, что N=5 и одновременно пытаются начать передачу станции с кодами приоритета 10000, 10100, и 00100. Первыми будут посланы станциями биты 1, 1 и 0. Третья станция сразу выбывает из конкурса (ее бит =0). Первая и вторая продолжают передачу. На следующем такте станции передадут биты 0 и 0, что позволяет им продолжить передачу. На третьем такте они выдадут 0 и 1, что вынудит первую станцию выбыть из соревнования. Оставшаяся станция (10100) продолжит передачу кода приоритета, а затем и данных. Число рабочих станций, подключенных к сетевому, сегменту может достигать 2N-1. Необходимость синхронизации ограничивает длину сетевого сегмента или быстродействие передачи.

Базовыми характеристиками алгоритмов доступа являются задержка при малых загрузках и эффективность канала при больших загрузках. Предположим, что **k** станций борются за доступ к каналу, каждая из них имеет вероятность **р** осуществить передачу. Вероятность того, что станция получит доступ к каналу при данной попытке, равна **kp(1-p)k-1.** Чтобы найти оптимальное значение p, продифференцируем полученное выражение вероятности по **p**, а результат приравняем нулю. После преобразования получается, что оптимальным значением является **p=1/k**. Подставив это значение в выражение вероятности успешного доступа, получим:

**{(k-1)/k}k-1**

Для малого числа станций вероятность получить доступ к каналу относительно высока, но уже при k=3-5 вероятность приближается к асимптотическому значению 1/e.

**Алгоритм адаптивного обхода**

Существует еще один относительно эффективный алгоритм доступа к каналу, называемый адаптивным протоколом перебора. Рассмотрим случай, когда в сетевом сегменте имеется 8 активных станций. На базе этих станций строится дерево (см. рис. 4.7).

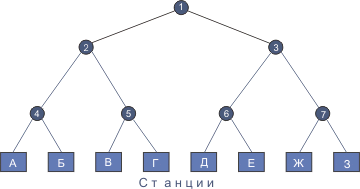


Рис. 4.7. Алгоритм адаптивного обхода

Первая ниша доступа следует за очередной успешной передачей и соответствует узлу дерева 0 (на рис. 4.7 не показан). В этом узле всем станциям предоставлена возможность доступа. Если одна из них его получила, очень хорошо. Если же произошло столкновение, то в нише узла 1 до конкурса допускаются станции, связанные с узлом 2 (А, Б, В и Г). Если одна из них получила доступ, в нише, следующей за переданным кадром, будут соревноваться станции, связанные с узлом 3 (Д, Е, Ж и З). Если же на предшествующем этапе произошло столкновение, следующая попытка предоставляется станциям, связанным узлом 4 (А и Б) и т.д.. Если столкновение произошло в нише узла 0, то сначала исследуется все дерево, чтобы выявить все готовые к передаче станции. Каждая бит-ниша ассоциируется с определенным узлом дерева. Если происходит столкновение, поиск продолжается рекурсивно для правой и левой ветвей узла. Если бит-ниша пассивна (нет заявок на передачу), или если имеется только одна станция-претендент, поиск будет прерван, так как все станции-претенденты найдены. Существует достаточно большое число вариаций данного алгоритма доступа к сети.

Следующим по популярности после csma/cd является маркерный доступ (Token Ring, Arcnet и FDDI), который более гибок и обеспечивает приоритетную иерархию обслуживания. Массовому его внедрению препятствует сложность и дороговизна. Хотя региональные сети имеют самую разнообразную топологию, практически всегда они строятся на связях точка-точка.

В таблице 4.1 представлены сводные данные по основным видам локальных сетей, используемых в настоящее время (список не является полным).

Таблица 4.1. Параметры различных локальных сетей

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Название сети** | **Топология** | **Быстро-действие Мбит/с** | **Доступ** | **Тип кабеля** | **NEXT при макс. частоте [дб]** | **Размер сети (сег-мента)** | **Макс. число узлов** |
| 10base5 | шина | 10 | CSMA/CD | RG-58 (50 Ом) |  | 500м | 1024 |
| 10base2 | шина | 10 | CSMA/CD | RG-58 (50 Ом) |  | 185 м | 90 |
| 10base-t | шина | 10 | CSMA/CD | UTP (III; 100 Ом) | 26 | 100 м | - |
| 1000base-FX | звезда | 1000 | CSMA | опто-волокно | - | 2км | - |
| 10Gbase-LR (-XL) | звезда | 10000 | CSMA | опто-волокно | - | 2км | - |
| 10broad36 | шина | 10 | CSMA/CD | RG-59 (75 Ом) |  | 3600 м | 1024 |
| 100base-tx | звезда | 100 | CSMA/CD | UTP (v; 100 Ом) | 29 | 200 м | - |
| 100base-fx | звезда | 100 | CSMA/CD | опто-волокно |  | 300 м | - |
| 100base-t4 | звезда | 100 | CSMA/CD | UTP (III; 100 Ом) | 21 | 200 м | - |
| 1base5 (starlan) | шина/ звезда | 1 | CSMA/CD | UTP (II) | 22 | 400 м | 1210 |
| IEEE 802.4 | шина | 1/5/10/20 | маркер | RG-59 (75 Ом) |  |  |  |
| Arcnet | звезда | 2,5/20 | маркер | RG-62/utp (93 Ом) |  | 600/125м | 255 |
| IEEE 802.5 | звезда | 4/16 | STP/UTP (150/120 Ом) | 22/32 | 366 м | 260 |  |
| Appletalk | шина/ звезда | 0,23 | CSMA/CD | STP/UTP (100 Ом) | 22/32 | 300/3000 м | 32 на сегмент |
| Ethertalk | шина/ звезда | 10 | CSMA/CD | STP/UTP, коакси-альный кабель |  | 500/3000 м | 254/1023 |
| ISN | звезда | 8,64 | Шина доступа | stp, опто-волокно |  | Не ограничено | 336/1920 |
| pc lan | дерево, звезда | 2 | CSMA/CD | RG-59 (75 Ом), UTP/STP | 32/22 | 2000 | 72 |
| Hyperchannel | шина | 50 | CSMA/CD | RG-59, опто-волокно |  | 3500 м | 256 |
| e-net | шина | 10 | CSMA/CD | RG-58 (50 Ом) |  | 700 м | 100 |
| G-net | шина | 1 | CSMA/CD | RG-58, RG-59 |  | 2000 м | 100 |
| FDDI | Двойное кольцо | 100 | маркер | опто-волокно |  | 100км | 1000 |
| PX-net | шина/ звезда | 1 | маркер | RG-62 (93 Ом) |  | 7000 м | 100 |
| S-net | шина/ звезда | 1 | Индивиду-альный | STP (100 Ом) | 21 | 700 м | 24 |
| wangnet | двойное дерево | 10 | CSMA/CD | RG-59 (75Ом) |  | 2800 м | 65000 |

Приведенная таблица не может претендовать на полноту. Так сюда не вошла сеть **IBM DSDB**, разработанная в начале 80-х годов. Полоса пропускания сети составляет 64 Мбит/с. Эта сеть рассчитана на обслуживания процессов реального времени. Сеть имеет топологию шины с приоритетным доступом (длина шины до 500м). Коммуникационная шина логически делится на три магистрали: сигнальная - для реализации приоритетного доступа; лексемная шина для резервирования в буфере места назначения; и, наконец, коммуникационная шина для передачи данных. Каждая из указанных магистралей использует идеологию независимых временных доменов, границы которых синхронизованы для всех трех магистралей. Схема доступа сходна с описанной для сетей CAN (см. раздел [4.1.4](http://book.itep.ru/4/41/can_414.htm)).

Существует целое семейство методов доступа, исключающих столкновение: это мультиплексирование по времени (TDM), по частоте (FDM) или длине волны (**WDMA** - Wavelength Division Multiple Access). Здесь каждому клиенту выделяется определенный временной домен или частотный диапазон. Когда наступает его временной интервал и клиент имеет кадр (или бит), предназначенный для отправки, он делает это. При этом каждый клиент ждет в среднем N/2 временных интервалов (предполагается, что работает N клиентов). При FDM передача не требует ожидания, так как каждому из участников выделяется определенный частотный диапазон, которым он может пользоваться когда угодно по своему усмотрению. Методы доступа TDM и FDM могут совмещаться. Но в обоих случаях временные интервалы или частотные диапазоны используются клиентом по мере необходимости и могут заметное время быть не заняты (простаивать). Следует также учесть, что если мы хотим обеспечить двухсторонний обмен при фиксированном частотном диапазоне, максимально возможное число участников следует уменьшить вдвое. Такие протоколы доступа часто используются в мобильной связи. Схема работы алгоритма WDMA показана на рис. 4.8.

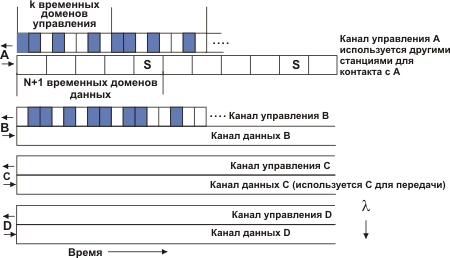


Рис. 4.8. Пояснение работы алгоритма доступа WDMA

На рисунке 4.8 представлен случай с четырьмя станциями (A, B, C и D). Каждая станция имеет два канала. Один канал (менее широкополосный) является управляющим каналом станции. Второй канал (широкополосный) служит для передачи данных. В каждом из каналов выделяются определенное число временных доменов. Будем считать, что число доменов в управляющем канале равно k, а число доменов в информационном канале - N+1, где N из них предназначены для данных, а последний служит для передачи станцией своего статуса (например, информации о свободных доменах). Все каналы синхронизуются от общих часов. Протокол поддерживает три типа трафика: постоянный поток данных с ориентацией на соединение (1), например, видео трафик; переменный поток, ориентированный на соединение (2), и поток дейтограмм, например, UDP (3). В протоколах, ориентированных на соединение, станция А прежде чем осуществить обмен с В должна послать запрос на соединение (Connection REQUEST), в свободном домене канала управления станции В. Если станция В воспринимает запрос, обмен осуществляется через канал данных станции А. Каждая станция имеет два передающих элемента и два принимающих.

1. Приемник с фиксированной длиной волны для прослушивания своего собственного канала управления.
2. Настраиваемый передатчик для посылки сообщений управляющим каналам других станций.
3. Передатчик с фиксированной длиной волны для посылки информационных кадров.
4. Настраиваемый приемник для приема данных от одного из передатчиков.

Каждая станция прослушивает свой управляющий канал с целью приема запросов и настраивается на длину определенного передатчика для получения данных. Настройка на определенную длину волны производится с помощью ячеек Фабри-Перо или интерферометров Маха-Цандера (предполагается, что мы работаем с оптическими волокнами).

Рассмотрим, как станция **А** устанавливает тип обмена **b** со станцией **В** , например, для файлового обмена. Сначала **А** настраивает свой приемник данных на частоту информационного канала **В** и ждет его статусного домена. Этот домен позволяет определить, какой из управляющих доменов свободен (на рисунке занятые домены окрашены в серый цвет). На рисунке 8 **В** имеет 8 доменов, из них 0, 3 и 5 свободны. Пусть принято решение занять запросом на соединение домен 5. Так как станция **В** постоянно прослушивает свой управляющий канал, она предоставит свободный домен для станции **А** . Это присвоение отражается в статусном домене информационного канала. Когда станция **А** обнаружит, что присвоение выполнено, можно считать что осуществлено однонаправленное соединение между**А** и **В** . Если для **А** требуется двунаправленное соединение (что реально для протокола FTP), то **В** повторит ту же процедуру со станцией **А** . Вполне возможна ситуация, когда одновременно станции **А** и **С** пытаются захватить один и тот же домен с номером 5. Ни одна из них его не получат, узнав об этом из статусного кода в домене управляющего канала **В**. После этого обе станции ждут псевдослучайное время и делают еще одну попытку.

Для осуществления пересылки файла станция **А** посылает **В** управляющее сообщение типа “загляните в домен данных 3, там кадр для вас”. Когда **В**получает управляющее сообщение, она настраивает свой приемник на выходной канал А, чтобы принять кадр данных от **А**. Станция **В** может использовать ту же схему для посылки подтверждения получения кадра, если это необходимо. Заметим, что может возникнуть проблема, если с В имеют одновременно соединения **А**и **С** и обе станции предлагают В заглянуть в информационный домен 3. Станция В воспримет одно из этих сообщений, второе буде потеряно. При постоянной скорости передачи используется модификация данного протокола. Когда станция **А** запрашивает соединение, она одновременно посылает сообщение типа “не возражаете, если я буду посылать данные в каждом домене 3?”. Если **В** способна принять данные (домен 3 не занят), устанавливается соединение с фиксированной полосой. Если это не так, **А** может попытаться использовать другой свободный временной домен. Для типа обмена 3 (дейтограммы) используется другая вариация протокола. Вместо посылки запроса соединения в найденный свободный домен управляющего канала, станция записывает сообщение “DATA FOR YOU in SLOT 3” (для вас есть данные в домене 3). Если **В** свободна во время следующего информационного домена 3, обмен будет успешным, в противном случае кадр теряется. Существует большое число различных вариантов протокола WDMA, например, с общим управляющим каналом для всех рабочих станций.

Рассмотрим случай (FDM) в предположении, что длина кадра имеет экспоненциальную плотность вероятности со средним значением **1/m** , тогда среднее время задержки **T** для канала с пропускной способностью **С** бит/сек при частоте кадров **f** сек-1 составит **T=1/(mC-f)** . Теперь предположим, что канал поделен на **N**независимых субканалов с пропускной способностью **C/N** бит/сек. Средняя загрузка каждого из субканалов станет равной **f/N** . Теперь для среднего значения задержки мы получим выражение:

http://book.itep.ru/4/formule_1.gif

Таким образом, FDM-доступ в N раз хуже, чем вариант, когда мы все кадры помещаем каким-то образом в некоторую общую очередь. Точно такая же аргументация может быть применена для метода TDM. По этой причине предпочтительнее динамические методы организации доступа к каналу (сетевой среде).

Интересной разновидностью Ethernet является широкополосная сеть типа Net/One. Она может базироваться на коаксиальном кабеле (суммарная длина до 1500 м) или на оптическом волокне (полная длина до 2500 м). Эта сеть по многим характеристикам аналогична обычному Ethernet (CSMA/CD) за исключением того, что коммуникационное оборудование передает данные на одной частоте, а принимает - на другой. Для каждого канала выделяется полоса 5 Мбит/с (полоса пропускания 6 МГц соответствует телевизионному стандарту). Предусматривается 5 передающих (59,75-89,75 МГц) и 5 принимающих (252-282 МГц) каналов для каждого из сетевых сегментов. Частота ошибок (**BER**) для сети данного типа меньше 10-8.

Другой Ethernet-совместимой сетью является Fibercom Whispernet. Сеть имеет кольцевую структуру (до 8 км), полосу пропускания 10Мбит/c, число узлов до 100 на сегменте при полном числе узлов в сети - 1024. Число кольцевых сегментов может достигать 101. Максимальное межузловое расстояние - 2км.

|  |
| --- |
| *Не следует думать, что современные сети, в частности Ethernet обладают таким уж высоким быстродействием по сравнению с “традиционными” средствами транспорта. Предположим, что товарный вагон загружен 10000 картриджами Exabyte, емкостью 10 Гбайт каждый (эта цифра может быть и больше). Вагон движется со скоростью 72 км/час. Какой информационный поток создает такой вагон? В секунду он перемещается на 20 метров, перенося 10000\*10\*109\*8 ≈ 1015 бит данных. Скорость передачи данных по скрученной паре или оптическому волокну составляет около 200000 км/c. Если измерять передающую способность в м\*бит/сек, то для 100Мбит/c Ethernet это будет 20\*1015 м\*бит/c, а для нашего вагона 20\*1015 м\*бит/сек. Таким образом наш вагон обеспечивает тот же темп переноса данных, что и Fast Ethernet.* |
|  |

Приведенное выше соображение безусловно поверхностно, так как не учитывает того факта, что мало доставить данные, надо их загрузить в ЭВМ (а вагоны машиной приниматься не могут). Но при построении сетей передачи данных и столь архаичный вариант не следует совершенно снимать со счетов.

1. В сетях типа САN узел, пытающийся получить доступ к одной из магистралей, выдает свой физический адрес и приоритет сообщения бит за битом.
2. При этом узел мониторирует состояние магистрали, проверяя, соответствует ли уровень сигнала тому, что он передает.
3. Если имеет место совпадение, передается следующий бит и осуществляется процедура пункта 2. При несовпадении узел прерывает передачу, возвращается к пункту 2 и ждет следующего цикла.

Данная схема доступа исключает столкновения, характерные для CSMA. Именно это преимущество делает сеть применимой для задач реального времени.

Существует целое семейство методов доступа, исключающих столкновение: это мультиплексирование по времени (TDM) и по частоте (FDM). Здесь каждому клиенту выделяется определенный временной домен или частотный диапазон. Когда наступает его временной интервал и клиент имеет кадр (или бит), предназначенный для отправки, он делает это. При этом каждый клиент ждет в среднем n/2 временных интервалов (предполагается, что работает n клиентов). При FDM передача не требует ожидания. Но в обоих случаях временные интервалы или частотные диапазоны используются клиентом по мере необходимости и могут заметное время быть не заняты (простаивать). Такие протоколы доступа часто используются в мобильной связи.

Примером нетрадиционного типа сети может служить Localnet 20 (Sytek). Сеть базируется на одном коаксиальном кабеле и имеет полную полосу пропускания 400 МГц. Сеть делится на 120 каналов, каждый из которых работает со скоростью 128 Кбит/с. Каналы используют две 36МГц-полосы, одна для передачи, другая для приема. Каждый из коммуникационных каналов занимает 300КГц из 36МГц. Сеть использует алгоритм доступа CSMA/CD, что позволяет подключать к одному каналу большое число сетевых устройств. Предусматривается совместимость с интерфейсом RS-232. Частота ошибок в сети не более 10-8.

Фирма Дженерал Электрик разработала сеть gm (map-шина), совместимую со стандартом IEEE 802.4. Целью разработки было обеспечение совместимости с производственным оборудованием различных компаний. Сеть рассчитана на работу со скоростями 1, 5 и 50 Мбит/с.

Традиционные сети и телекоммуникационные каналы образуют основу сети - ее физический уровень. Реальная топология сети может динамически изменяться, хотя это и происходит обычно незаметно для участников. При реализации сети используются десятки протоколов. В любых коммуникационных протоколах важное значение имеют операции, ориентированные на установление связи (connection-oriented) и операции, не требующие связи (connectionless - "бессвязные", ISO 8473). Интернет использует оба типа операций. При первом типе пользователь и сеть сначала устанавливают логическую связь и только затем начинают обмен данными. Причем между отдельными пересылаемыми блоками данных (пакетами) поддерживается некоторое взаимодействие. "Бессвязные" операции не предполагают установления какой-либо связи между пользователем и сетью (например, протокол UDP) до начала обмена. Отдельные блоки передаваемых данных в этом случае абсолютно независимы и не требуют подтверждения получения. Пакеты могут быть потеряны, задублированы или доставлены не в порядке их отправки, причем ни отправитель, ни получатель не будут об этом оповещены. Именно к этому типу относится базовый протокол Интернет - IP.

Для каждой сети характерен свой интервал размеров пакетов. Среди факторов, влияющих на выбор размеров можно выделить

* Аппаратные ограничения, например размер домена при мультиплексировании по времени.
* Операционная система, например размер буфера 512 байт.
* Протокол (например, число бит в поле длины пакета).
* Обеспечение совместимости с определенными стандартами.
* Желание уменьшить число ошибок при передаче ниже заданного уровня.
* Стремление уменьшить время занятости канала при передаче пакета.

Ниже приведены максимальные размеры пакетов (MTU) для ряда сетей

|  |  |  |
| --- | --- | --- |
| Сеть | MTU Байт | Быстродействие Мбит/с |
| IEEE 802.3 | 1500 | 10 |
| IEEE 802.4 | 8191 | 10 |
| IEEE 802.5 | 5000 | 4 |

Операции, ориентированные на установление связи (например, протокол TCP), предполагают трехстороннее соглашение между двумя пользователями и провайдером услуг. В процессе обмена они хранят необходимую информацию друг о друге, с тем, чтобы не перегружать вспомогательными данными пересылаемые пакеты. В этом режиме обмена обычно требуется подтверждение получения пакета, а при обнаружении сбоя предусматривается механизм повторной передачи поврежденного пакета. "Бессвязная" сеть более надежна, так как она может отправлять отдельные пакеты по разным маршрутам, обходя поврежденные участки. Такая сеть не зависит от протоколов, используемых в субсетях. Большинство протоколов Интернет используют именно эту схему обмена. Концептуально TCP/IP-сети предлагают три типа сервиса в порядке нарастания уровня иерархии:

1. "бессвязная" доставка пакетов;
2. надежная транспортировка информации;
3. реализация прикладных задач.

Немногие из читателей участвуют в создании региональных и тем более глобальных сетей, за то структура и принципы построения локальных сетей им, безусловно, близки. На рис. 4.9 и 4.10 приведены два варианта “ресурсных” локальных сетей (сети для коллективного использования ресурсов - памяти, процессоров, принтеров, магнитофонов и т.д.). Такие сети строятся так, чтобы пропускная способность участков, где информационные потоки суммируются, имели адекватную полосу пропускания. Эффективность сети на рис. 4.9 сильно зависит от структуры и возможностей контроллеров внешних устройств, от объема их буферной памяти. В качестве концентраторов обычно используются переключатели (switch), но могут применяться и обычные HUB или даже маршрутизаторы.

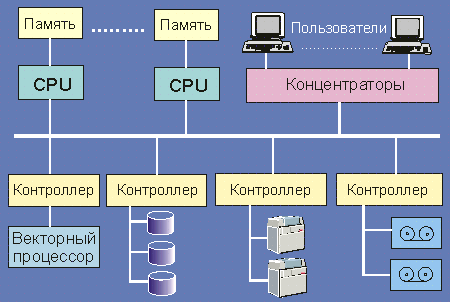


Рис. 4.9. Вариант схемы ресурсной локальной сети

Сеть, показанная на рис. 4.10, несравненно более эффективна (практически исключены столкновения и легче гарантировать определенное время доступа к ресурсу). Здесь также немало зависит от свойств контроллеров внешних ресурсов (помечены красным цветом). Но такие сети обычно более дорого реализовать.

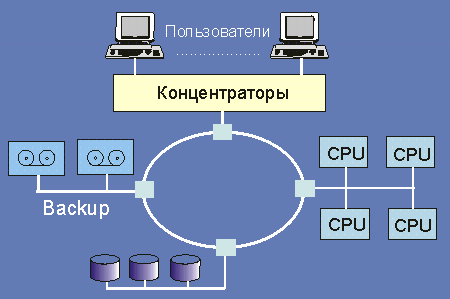


Рис. 4.10.

Для сопоставления быстродействия различных видов сетей Сталлингс (Stallings, W.: *Data and Computer Communications*, New York: Mac-Millan Publishing Company, 1985) в 1985 году разработал критерий. Критерий предполагает вычисление битовой длины BL (максимальное число бит в сегменте), которая равна произведению максимальной длины сегмента (L) на полосу пропускания (W), деленное на скорость распространения сигнала в сегменте (S):

BL = l\*W/S

Для Ethernet BL = [500(м)\*10 106(бит/c)]/2 108 (м/c) = 25 бит.

Коэффициент использования сети равен = 1/(1+), где

http://book.itep.ru/4/image118.gif. Для Ethernet при длине пакета 1500 байта  = 0,0021, что дает для эффективности использования сети 0,997. Таким образом, максимальная пропускная способность ethernet составляет 9,97 Мбит/c или 1,25 Мбайт/с. Разумеется, в этом подходе не учитываются издержки, связанные с заголовками пакетов, которые дополнительно снижают эффективность сети. Из данного рассмотрения может показаться, что чем больше пакет, тем лучше. С точки зрения пропускной способности так оно и есть. Но с увеличением длины пакета увеличивается время отклика сети. Таким образом, выбор MTU определяется реальными требованиями пользователей.

**Принципы построения сетевых программных интерфейсов**

Существует большое разнообразие сетевых интерфейсов. Их структура зависит от характера физического уровня сетевой среды, от метода доступа, от используемого набора интегральных схем и т.д.. Здесь пойдет речь о принципах построения программного интерфейса (см. также раздел 7; [..\7\soft\_7.htm](http://book.itep.ru/7/soft_7.htm)).

Существует три возможности построения интерфейса: *с базированием на памяти, с использованием прямого доступа и с применением запросов обслуживания*.

Первый вариант предполагает наличие трех компонентов: буфер сообщений, область данных для управления передачей и зона памяти для управления приемом данных. Первый из компонентов служит для формирования исходящих сообщений программного интерфейса. Должны быть приняты меры, чтобы исключить модификацию содержимого этого буфера до того, как данные будут считаны ЭВМ или интерфейсом. Проблема решается путем формирования соответствующих указателей. Управление буфером осуществляется ЭВМ или совместно ЭВМ и интерфейсом с использованием механизма семафоров.

Остальные методы связаны с использованием традиционных методов управления памятью с помощью средств операционной системы. Критической проблемой является обеспечение достаточного места в буфере для приходящих сообщений. Ведь в отсутствии памяти приходящее или записанное ранее сообщение может быть потеряно. Недостаток места для исходящих сообщений не является критическим, так как приводит обычно к задержке передачи, а не к потере сообщения.

Второй компонент интерфейса, базирующегося на использовании памяти, часто реализуется в виде так называемых буферов управления передачей (TCB). Эти буферы содержат такую информацию как положения сообщения в памяти, длина сообщения, адрес места назначения, идентификатор процесса-отправителя, приоритет сообщения, предельное значение числа попыток передачи, а также флаг, указывающий на необходимость присылки подтверждения от получателя. **TCB**(transmission control buffer) создается процессом-отправителем и передается интерфейсу, после завершения записи в буфер сообщений. Параметры TCB используются интерфейсом при организации процесса передачи сообщения.

Третий компонент интерфейса, базирующегося на использовании памяти, называется буфером управления приемом (**RCB** - Reception Control Buffer). RCB содержит в себе информацию, сходную с той, что записывается в TCB, например, идентификатор отправителя, длина сообщения, индикатор ошибки, время приема и идентификатор процесса места назначения. RCB заполняется интерфейсом при получении сообщения и передается процессору ЭВМ. Основополагающим принципом построение всех трех компонент является совместное использование памяти и гибкая система указателей. Версия программы, рассмотренная в разделе 7 ближе именно к этой схеме взаимодействия.

Во втором варианте широко используемой схемы доступа к сети (“прямой доступ”) взаимодействие ЭВМ и интерфейса строится по схеме клиент-сервер. Конкретная реализация программы в этом случае в большей степени зависит от структуры регистров физического интерфейса.

В третьем варианте сетевого программного интерфейса используются служебные запросы. Этот тип сетевого доступа удобен для коммуникационных протоколов высокого уровня, таких как команды ввода/вывода CSP-стиля (Communicating Sequential Processes) или процедуры обмена языка Ада. В этом методе накладываются определенные ограничения на реализацию нижележащих коммуникационных уровней.

Программирование для сетей существенным образом является программированием для процессов реального времени. Здесь часто можно столкнуться с тем, что одна и та же программа ведет себя по-разному в разных ситуациях. Особое внимание нужно уделять написанию многопроцессных сетевых программ, где также как в случае (см. раздел 7.1; [..\7\sock\_71.htm](http://book.itep.ru/7/sock_71.htm)) работы с соединителями могут возникать ситуации блокировок. Пример такой ситуации показан на рис. 4.11.

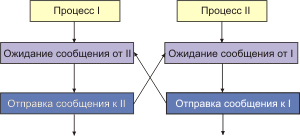


Рис. 4.11.

Когда число процессов больше, заметить запрограммированную ситуацию блокировки заметно сложнее. По этой причине необходимо предусмотреть меры препятствующие блокировке, если ожидаемое сообщение не пришло.

Одной из важнейших и достаточно трудно реализуемых функций сетевого оборудования (например, на скорости 1-10Мбит/c) является обслуживание очередей и подавление перегрузок.

**Очереди FIFO**

Очереди FIFO (First-In-First-Out) используются обычно в скоростных интерфейсах (быстродействие > 2048кбит/c). Здесь первый пришедший пакет первым и покидает очередь. Порядок следования пакетов при этом алгоритме не изменяется. Приоритетное обслуживание в этом варианте также не может быть осуществлено. Когда очередь заполнена, все последующие пакеты будут отбрасываться до тех пор, пока из очереди не будет изъят хотя бы один пакет.

**Приоритетное обслуживание очередей (PQ)**

Приоритетное обслуживание (**PQ**) является эффективной и прямой формой управления перегрузкой. PQ позволяет сетевому администратору выделить до четырех очередей в сетевом трафике. Предусмотрены очереди **высокого, среднего, нормального** и **низкого** приоритета. Маршрутизатор обрабатывает очереди строго в соответствии с их приоритетом. Пакеты из очереди с высоким приоритетом обрабатывается, пока в очереди не останется ни одного пакета, после этого начинается обработка очереди со средним приоритетом, параллельно осуществляется контроль, появления пакетов в очереди с высоким приоритетом. Пакеты из очереди с низким приоритетом обрабатываются лишь тогда, когда остальные очереди пусты. Низко приоритетный трафик при определенных обстоятельствах может быть полностью блокирован, а пакеты потеряны. Обычно PQ используется, когда приложения, критичные к задержкам, сталкиваются с проблемами. Если высокоприоритетный трафик имеет высокую интенсивность, высока вероятность того, что остальные составляющие трафика будут блокированы. Пакеты, неклассифицированные PQ, автоматически относятся к очереди с *нормальным* приоритетом. Системная очередь имеет приоритет выше высокого. По умолчанию очереди разных приоритетов имеют следующие размеры:

|  |  |
| --- | --- |
| **Приоритет** | **Длина очереди** |
| Высокий | 20 пакетов |
| Средний | 40 пакетов |
| Нормальный | 60 пакетов |
| Низкий | 80 пакетов |

**Обычное обслуживание очередей (СQ)**

Чтобы избежать жесткой регламентации системы обслуживания очередей PQ, системный администратор может выбрать стратегию **CQ** (Custom Queuing - обычное обслуживание очередей). CQ позволяет сетевому администратору реализовать приоритетное обслуживание трафика без побочных эффектов, связанных с блокировкой низкоприоритетных потоков. С помощью CQ можно сформировать 16 очередей трафика. Каждая из очередей обслуживается в карусельном режиме (round-robin). По умолчанию CQ обслуживает 1500 байт за цикл. Однако CQ не может фрагментировать пакеты. Это означает, что если CQ обрабатывает пакет длиной 1000 байт, следующий 1500-байтный пакет будет обработан вне очереди. При этом может возникнуть ситуация, когда для коротких очередей может возникнуть таймаут. CQ может конфигурироваться также как и PQ. Процесс классификации трафика приписывает пакеты одной из 16 конфигурируемых очередей, работающих в режиме "дырявого ведра". Существует очередь с номером 0 (системная очередь), которая зарезервирована для пакетов управления сетью и имеет наивысший приоритет. При больших потоках возможна потеря пакетов из-за переполнения очередей. Длина очереди может быть задана в пределах от 0 до 32767 (20 - значение по умолчанию).

**Взвешенные справедливые очереди (WFQ)**

Стратегия справедливых (взвешенных) очередей **WFQ** (Weighted Fair Queuing) используется по умолчанию для интерфейсов низкого быстродействия. WFQ делит трафик на несколько потоков, используя в качестве параметров (для IP-протокола): IP-адреса и порты получателя и отправителя, а также поле IP-заголовка ToS. Значение ToS служит для квалификации (части выделяемой полосы) потока. Для каждого из потоков формируется своя очередь. Максимально возможное число очередей равно 256. Очереди обслуживаются в соответствии с карусельным принципом (round-robin). Более высокий приоритет имеют потоки с меньшей полосой, например, telnet. По умолчанию каждая из очередей имеет емкость 64 пакета (но допускается значение и <4096 пакетов).

В сетях существует 8 уровней приоритета. Следует иметь в виду, что WFQ не поддерживается в случае туннелирования или шифрования. Субпоток с низким весом получает более высокий уровень обслуживания, чем с высоким. Не реализуется WFQ и в сетях АТМ. Когда задействованы биты ToS, WFQ реализует приоритетное обслуживание пакетов согласно значению этого кода. Весовой фактор обратно пропорционален уровню приоритета.

**Справедливые очереди базирующиеся на классах (CBWFQ)**

Дальнейшим развитием технологии WFQ является формирование классов потоков, задаваемых пользователем. Алгоритм построения очередей, базирующихся на классах, называется CBWFQ (Class-Based Weighted Fair Queuing). Алгоритм CBWFQ предоставляет механизм управления перегрузкой. Параметры, которые характеризуют класс, те же что и в случае WFQ (только вместо ToS используется приоритет). В отличии от WFQ здесь можно в широких пределах перераспределять полосу пропускания между потоками. Для выделения класса могут привлекаться ACL (Access Control List) или даже номер входного интерфейса. Каждому классу ставится в соответствие очередь. В отличие от RSVP данный алгоритм гарантирует полосу лишь в условиях перегрузки. Всего может быть определено 64 класса. Нераспределенная полоса может использоваться потоками согласно их приоритетам. Если используется резервирование CBWFQ и RSVP возможны конфликты, так как IOS маршрутизатора не проверяет баланса зарезервированной полоcы разными протоколами.

**Очереди с малой задержкой (LLQ)**

В некоторых случаях, например в случае VoIP, более важно обеспечить малую задержку, а не полосу пропускания. Для таких задач разработан алгоритм LLQ (Low Latency Queuing), который является модификацией CBWFQ . В этом алгоритме пакеты всех приоритетов кроме наивысшего вынуждены ждать, пока очередь более высокого приоритета будет опустошена. Разброс задержки в высоко приоритетном потоке может быть связан только с ожиданием завершения передачи пакета низкого приоритета, начавшейся до прихода приоритетного кадра. Такой разброс определяется диапазоном длин кадров (MTU).

Необычайно важной проблемой при построении сетей является их устойчивость при возникновении перегрузок. В Интернет для этого используется специальная опция протокола ICMP, а во Frame Relay имеются меры для преодоления перегрузок непосредственно на нижних протокольных уровнях.

**Методы работы в условиях перегрузки**

Оптимальность управления сетью в условиях перегрузок определяет эффективность использования сети. Пока субсеть загружена незначительно, число принимаемых и обрабатываемых пакетов равно числу пришедших. Однако, когда в субсеть поступает слишком много пакетов может возникнуть перегрузка и рабочие характеристики деградируют. При очень больших загрузках пропускная способность канала или сети может стать нулевой (см. [[39]](http://book.itep.ru/9/literat.htm#point39)) Такая ситуация называется коллапсом сети.

Отчасти это может быть связано с недостатком памяти для входных буферов, по этой причине некоторое увеличение памяти может помочь. Но следует помнить, что всякое лекарство хорошо в меру. Еще в 1987 году Нагле (Nagle) обнаружил, что если маршрутизатор имеет даже беспредельную память, эффект перегрузки может оказаться еще более тяжелым. Это сопряжено со временем, которые пакеты ожидают обработки. Если время ожидания в очереди превышает длительность таймаута, появятся дубликаты пакетов, что, безусловно, понижает эффективность системы. Причиной перегрузки может быть медленный процессор или недостаточная пропускная способность какого-то участка сети. Простая замена процессора или интерфейса на более быстродействующий не всегда решает проблему - чаще переносит узкое место в другую часть системы. Перегрузка, как правило, включает механизмы, усиливающие ее негативное воздействие. Так переполнение буфера приводит к потере пакетов, которые позднее должны будут переданы повторно (возможно даже несколько раз). Процессор передающей стороны получает дополнительную паразитную загрузку. Все это указывает на то, что контроль перегрузки является крайне важным процессом. Следует делать различие между контролем потока и контролем перегрузки. Под контролем потока подразумевается балансировка потока отправителя и возможности приема и обработки получателя. Этот вид контроля предполагает наличие обратной связи между получателем и отправителем. В этом процессе участвуют, как правило, только два партнера. Перегрузка же более общее явление, относящееся к сети в целом или к какой-то ее части. Например, 10 ЭВМ хотят передать одновременно какие-то файлы другим 10 ЭВМ. Конфликта потоков здесь нет, каждая из ЭВМ способна переработать поступающие данные, но сеть не может пропустить поток, генерируемый 10 сетевыми интерфейсами одновременно. Часто эти явления не так просто разделить. Отправитель может получить сообщение ICMP (quench) (в случае TCP/IP) при перегрузке получателя или при перегрузке какого-то сегмента сети.

**Алгоритм RED/WRED**

Одним из эффективных методов подавления серьезных перегрузок сети является алгоритм **RED/WRED**. Особенностью этого алгоритма является то, что решение о постановке пакета в очередь принимается по-разному, в зависимости от уровня заполнения буфера (длины очереди). Устанавливаются два порога Т1 и Т2. Пока усредненная длина очереди ниже Т1, любой входящий пакет поступает в буфер. В области между Т1 и Т2 вероятность отбрасывания пакета линейно растет от 0 до значения Pc. После достижения порога Т2 все поступающие пакеты отбрасываются. Суть алгоритма WRED заключается в том, что, отбрасывая пакеты система исключает катастрофические перегрузки (теряя немного сейчас мы избегаем больших потерь). Приводимые ниже расчетные и экспериментальные данные взяты из публикаций и диссертации Гончарова А.А..

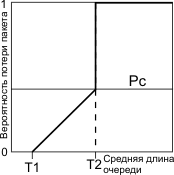


Рис. 4.12.

Усреднение длины очереди (http://book.itep.ru/4/qos6_1.files/image2.gif) производится согласно следующей формулы:

**http://book.itep.ru/4/qos6_1.files/image2.gif = (Qav × (1-2-n)) + (Q × 2-n),     (1)**

где n - экспоненциальный весовой фактор, конфигурируемый пользователем, QAV – предшествующее значение усредненной длины очереди, Q – текущее значение длины очереди. Введем обозначение qw=(2-n). Именное значение длины очереди, вычисленное по формуле (1), сравнивается с величинами Т1 и Т2. При малом значении qw процесс WRED не сразу начнет отбрасывать пакеты при перегрузке, зато продолжит отбрасывание, даже когда перегрузки уже нет (очередь сократилась ниже минимального порога)

Усреднение длины очереди является важным компонентом алгоритма управления процессом буферизации. Без усреднения процесс буферизации был бы подвержен сильному влиянию случайных флуктуаций входного потока пакетов. Но именно усреднение является причиной возникновения осцилляций длины очереди. Ведь зависимость принятия решения об отбрасывании того или иного пакета определяется значением усредненной длины очереди, которое может существенно отличаться от текущего. Амплитуда вариации текущего значения длины очереди обычно существенно больше усредненного. Расчеты показывают, что при определенных параметрах текущая длина очереди может достигать в максимуме полного объема буфера, а в минимуме нуля (т.е. буфер уже пуст, а отбрасывание пакетов продолжается, см. рис. 4.13). Обе крайности нежелательны, так как приводят к неэффективности использования полосы канала, где работает данный буфер.

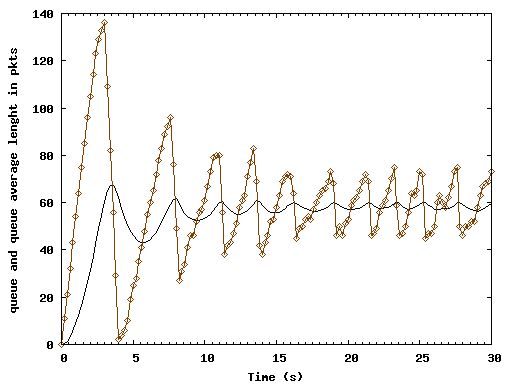


Рис. 4.13. Зависимость от времени и Q (Wq=0.002; pc=0.2; T1=25; T2=60; размер буфера = 800; время эксперимента 30 сек; перегрузка =1.4)

На рис. 4.13 ромбиками отмечена зависимость текущего значения длины очереди от времени. Отсюда видно, что усредненное значение длины очереди на начальном участке зависимости уступает текущей длине более чем в два раза. В расчетах входной поток  и выходной  задавались в битах в секунду. В области от 0 до Т1 рост длины очереди определяется произведением ()t. После достижения уровня Т1 скорость роста длины очереди замедляется, так как часть пакетов отбрасывается, зависимость становится квадратичной. Прекращение роста и начало спада Q происходит в момент, когда достигает уровня Т2.

Задачей данной работы было выявление области параметров управления очередью, при которых осцилляции длины очереди минимальны, а усреднение приемлемо.

Расчеты проводились с привлечением пакета программ моделирования NS-2 [2]. Значения Т1 и Т2 задавалось в пакетах. Отношение  определяет уровень перегрузки канала.

На рис. 4.14. показана зависимость усредненного значения длины очереди от времени и параметра PC. PC здесь варьировалось в интервале от 0,01 до 0,7.

Представленные на рисунке результаты показывают, что минимальные осцилляции происходят в области PC<0,3. Затухание осцилляций происходит за время менее 10сек.

При значениях PC> 0,6 осцилляции длины очереди не затухают даже спустя 30 сек после начала перегрузки канала. Производная **dA/dpc=10**, где А – максимальная амплитуда осцилляций очереди, остается постоянной в интервале 0,1< PC<0,7 (см. рис 4.14).

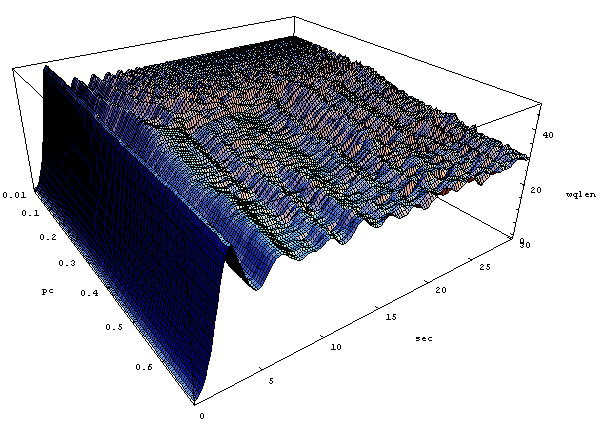


Рис. 4.14. Расчеты эволюции как функции PC выполнены при следующих значениях параметров:  = 1,4, qw = 0.002, T1=25, T2=40

Если в области малых PC осцилляции происходят вокруг равновесного значения ~Т2, то при PC > 0,4 этот уровень падает до 30, что связано с тем, что заметная доля пакетов отбрасывается еще до достижения уровня Т2.

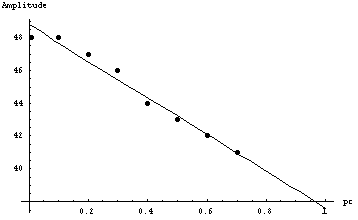


Рис. 4.15. Зависимость амплитуды осцилляции от PC

Если при малых значениях PC равновесное значение длины очереди равно Т2=40, то при больших значениях PC равновесное значение после затухания осцилляций приближается к 30.

На рис. 4.16 показана зависимость от времени и уровня перегрузки l/m в диапазоне от 1.1 до 2.0. Остальные параметры имели следующие значения: PC =0.5 и qw =0.002, T1=25, T2=40 (размер буфера В=180 пакетам). С ростом уровня перегрузки амплитуда осцилляций линейно падает, одновременно также линейно сокращается период осцилляций.

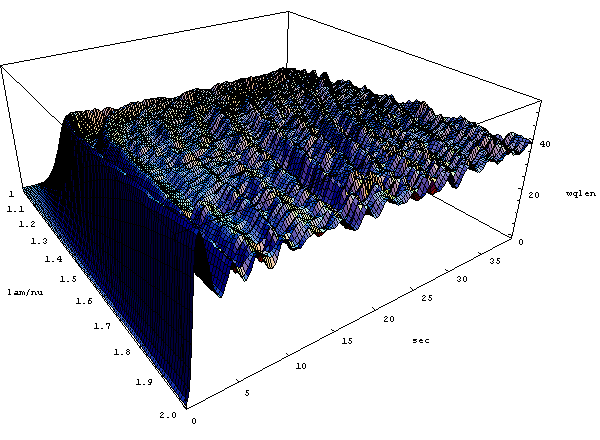


Рис. 4.16. Зависимость от времени и уровня перегрузки канала 

Из рисунка видно, что наименьший уровень осцилляций длины очереди имеет место для  в диапазоне 1,2-1,5. К сожалению, на практике этот параметр обычно не выбирается

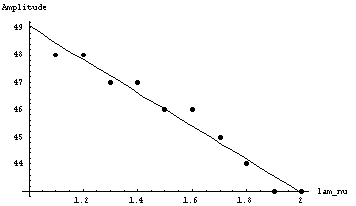


Рис. 4.17. Зависимость амплитуды осцилляции длины очереди от 

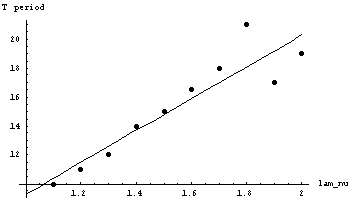


Рис. 4.18. Зависимость периода осцилляции длины очереди от 

На рис. 4.19 показана зависимость осцилляций длины очереди от фактора усреднения qW. Из рисунка видно, что приемлемые значения лежат в области >0,003. При меньших значениях qW осцилляции не затухают даже через 10 сек после начала перегрузки. Равновесное значение ~ T2.

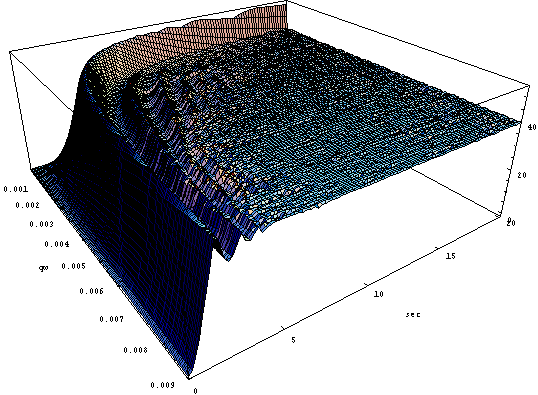


Рис. 4.19. Зависимость от фактора усреднения qW

Ниже на рис. 4.20. представлена зависимость от порога Т2. Фактор перегрузки постоянен =1.4; Т1=25=const; T2=(T1/10)\*index; index=[1:40]; Pc=0.1; B=900.

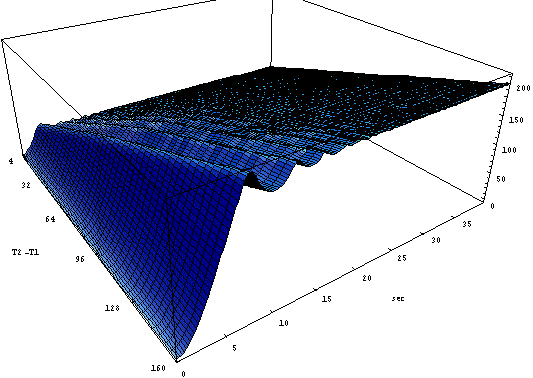


Рис. 4.20. Зависимость от разницы порогов Т2-Т1

Следует иметь в виду, что обычно осцилляции происходят вокруг значения Т2.

Оптимальный выбор параметров алгоритма WRED позволяет увеличить эффективность использования буферов маршрутизатора и, как следствие, поднять пропускную способность или улучшить уровень QoS.

Начинать надо с решения проблемы выявления перегрузок. Перегрузкой следует считать ситуацию, когда нагрузка в течение некоторого оговоренного времени превышает заданную величину. Параметрами, которые позволяют судить о наличии перегрузки могут служить:

* процент пакетов, отбрасываемых из-за отсутствия свободного буферного пространства.
* средняя длина очереди
* процент пакетов, пересылаемых повторно
* среднее время задержки пакета и некоторые другие величины

Когда перегрузка выявлена, нужно передать необходимую информацию из точки, где она обнаружена, туда, где можно что-то сделать для исправления ситуации.

Можно послать уведомление о перегрузке отправителю, загружая дополнительно и без того перегруженный участок сети. Альтернативой этому может быть применение специального поля в пакете, куда маршрутизатор может записать соответствующий код при перегрузке, и послать его соседям. Можно также ввести специальный процессор или маршрутизатор, который рассылает периодически запросы о состоянии элементов сети. При получении оповещения о перегрузки информационный поток может быть послан в обход.

При использовании обратной связи путем посылки сообщения-запроса понижения скорости передачи следует тщательно настраивать временные характеристики. В противном случае система либо попадает в незатухающий осциллятивный режим, либо корректирующее понижение потока будет осуществляться слишком поздно. Для корректного выбора режима обратной связи необходимо некоторое усреднение.

Преодоление перегрузки может быть осуществлено понижением нагрузки или добавлением ресурсов приемнику.

Положительный результат может быть достигнут изменением механизма подтверждения (например, уменьшением размера окна), вариацией значений таймаутов, вариацией политики повторной передачи пакетов. В некоторых случаях позитивный результат может быть получен изменением схемы буферизации. Иногда решить проблему может маршрутизатор, например, перераспределяющий трафик по нескольким направлениям.

Одной из причин перегрузки часто являются импульсные загрузки сегмента сети или сетевого устройства. По этой причине любые меры (напр., pipelining), которые могут выровнять поток пакетов, безусловно улучшат ситуацию (например, traffic shaping в сетях ATM). В TCP же с его окнами импульсные загрузки предопределены.

Необычайно важной проблемой при построении сетей является их устойчивость при возникновении перегрузок. В Интернет для этого используется специальная опция протокола ICMP, а во Frame Relay имеются меры для преодоления перегрузок непосредственно на нижних протокольных уровнях.

**Алгоритм leaky bucket ("дырявое ведро")**



Для систем без обратной связи решение проблемы выравнивания скорости передачи данных может быть решено с помощью алгоритма *leaky bucket*. Суть этого алгоритма заключается в том, что на пути потока устанавливается буфер выходной поток которого постоянен и согласован с возможностью приемника. Если буфер переполняется, пакеты теряются. Потеря пакетов вещь мало приятная, но это блокирует процессы, которые могут привести к коллапсу сегмента или всей сети. Там, где потеря пакетов нежелательна, можно применить более гибкий алгоритм.

**Алгоритм Token Bucket ("маркерное ведро")**

Алгоритм *token bucket* предполагает наличие в буферном устройстве (или программе) некоторого количества маркеров. При поступлении на вход буфера пакетов маркеры используются для их транспортировки на выход. Дальнейшая передача данных на выход зависит от генерации новых маркеров. Поступающие извне пакеты тем временем накапливаются в буфере. Таким образом, полной гарантии отсутствия потерь мы не имеем и здесь. Но алгоритм token bucket позволяет передавать на выход "плотные" группы пакетов ограниченной численности (по числу маркеров), снижая в некоторых случаях вероятность потери. Если буферное устройство "смонтировано" внутри ЭВМ-отправителя, потерь можно избежать вовсе, блокируя передачу при заполнении буфера. Как в одном так и в другом алгоритме мерой передаваемой информации может быть не пакет, а n-байт (где n некоторое оговоренное заранее число).

В системах, где управление трафиком осуществляется с использованием обратной связи, можно достичь большей эффективности. Одним из механизмов преодоления перегрузок является *управление разрешением (admission control)*. Суть метода заключается в том, что при регистрации перегрузки не формируется более никаких виртуальных соединений до тех пор, пока ситуация не улучшится. Альтернативным вариантом может служить решение, где формирование нового соединения разрешается, но при этом осуществляется маршрутизация так, чтобы обойти узлы, в которых выявлена перегрузка (смотри рис. 4.21 ).

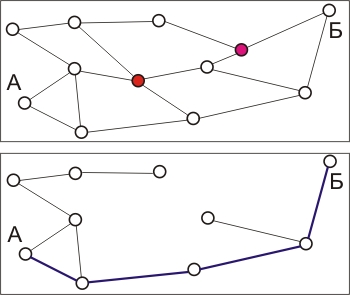


Рис. 4.21. Выбор маршрута нового виртуального канала при наличии перегрузки

На рис. 4.21 (верх) показан пример сети с двумя узлами, характеризующимися перегрузкой (помечены красным цветом). Предположим, что необходимо проложить виртуальный канал из узла А в узел Б. Из графа маршрутов удаляются перегруженные узлы, после чего осуществляется прокладка пути. В нижней части рисунка синим цветом показан новый виртуальный канал.

Еще более универсальным решение, пригодным для работы с установлением соединения и без, является посылка пакетов блокировки (choke packets). Маршрутизатор обычно контролирует загруженность всех своих внешних каналов **l**, которая может принимать значения от 0 до 1. Когда l достигает некоторого порогового значения, отправителю посылается пакет блокировки. При вычислении l следует использовать какую-либо методику усреднения, чтобы избежать слишком частых блокировок.

Когда отправитель получает пакет блокировки, он должен уменьшить трафик, посылаемый получателю на заданное число процентов. Так как на пути к месту назначения может быть много пакетов, это вызовет серию пакетов блокировки. Отправитель должен игнорировать пакеты блокировки в течение некоторого времени после получения первого такого пакета. По истечении этого периода отправитель прослушивает канал на протяжении аналогичного времени, ожидая получения новых пакетов блокировки. Если такой пакет приходит, канал все еще перегружен и отправитель снова должен понизить темп посылки пакетов. Если на протяжение периода прослушивания не приходит новых пакетов блокировки, отправитель может увеличить поток снова.

ЭВМ может понижать трафик, корректируя свои параметры, например, ширину окна или темп передачи на выходе устройства типа "дырявое ведро". Обычно первый блокирующий пакет уменьшает поток вдвое, следующий на 0,25 от первичного и т.д. Увеличение потока также производится аналогичными шагами. Существует большое число вариантов алгоритма управления потоком с использованием пакетов блокировки. Параметром, который контролируется и определяет условие отправки пакета блокировки, может служить длина очереди или заполненность буфера.

Ситуация перегрузки не всегда управляется однозначно. Например, при поступлении на вход пакетов от трех источников возможна ситуация, когда приемник посылает блокирующие пакеты всем отправителям, а откликнется сокращением потока только один. В результате этот узел, который "играет по правилам" (как это часто бывает и в жизни) оказывается в проигрыше. В 1987 году Нагле был предложен алгоритм *fair queueing* (честная очередь). В этом алгоритме маршрутизатор организует независимые очереди для пакетов, поступающих от разных источников. Когда выходной канал маршрутизатора оказывается свободным, он просматривает очереди циклически и отравляет очередной пакет. В результате при n очередях по завершении такого цикла просмотров-посылок оказываются посланы по одному пакету из каждой очереди. Такой алгоритм используется в некоторых ATM-переключателях. Следует заметить, что этот алгоритм дает некоторые преимущества тем узлам, которые посылают более длинные пакеты. Демерс (Demers) и др. в 1990 году предложил некоторое усовершенствование алгоритма. В данном варианте организуется циклический просмотр очередей не по-пакетно, а по-байтно. Система последовательно сканирует очереди и определяет положение концов пакетов. Первыми отправляются более короткие пакеты. Для иллюстрации предлагается рассмотреть рис. 4.22. (см. также [[39]](http://book.itep.ru/9/literat.htm" \l "point39))

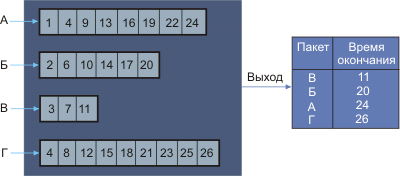


Рис. 4.22. Маршрутизатор с 4-мя входными каналами, в каждом из которых ждет очереди передачи по одному пакету. В правой части рисунка представлен порядок посылки этих пакетов.

Пакеты на рисунке имеют от трех до девяти октетов. Порядок пересылки октетов показан в левой части рисунка. В отсутствии поступления новых пакетов, кадры, записанные в буфер будут переданы в порядке, представленном в правой части рисунка. Особенностью этого алгоритма является равенство приоритета всех входных каналов.

При передаче данных на большие расстояния с большими значениями RTT эффективность использования метода блокирующих пакетов снижается. Пока блокирующий пакет дойдет через ряд промежуточных узлов до отправителя, на вход получателя поступит большое число пакетов, которые не только усугубят ситуацию перегрузки, но и могут вызвать потерю какой-то их доли, что, в свою очередь, может потребовать повторной пересылки следовавших за ними кадров. Для повышения эффективности часто применяется схема, при которой блокирующие пакеты воздействуют на все маршрутизаторы по пути своего следования. В этом случае снижения потока можно ожидать уже через время, равное RTT до узла, ближайшего к получателю пакетов. Такая схема требует того, чтобы все промежуточные узлы имели достаточно емкие буферы, в противном случае возможны потери.

В протоколе [TCP](http://book.itep.ru/4/44/tcp_443.htm)используется алгоритм управления трафиком, называемый "скользящее окно". Здесь размер окна, которое определяет число сегментов, посылаемых без получения подтверждения, варьируется в зависимости от наличия потерь пакетов. При большой вероятности потери система переходит в режим, когда очередной пакет не посылается до тех пор, пока не будет подтверждено получение предшествующего. При серьезных перегрузках, когда потери становятся значительными, нарушается механизм вычисления значений RTT и таймаутов, что может приводить к трудно предсказуемым последствиям. Следует обратить внимание, что в протоколе UDP какого-либо механизма управления трафиком не предусмотрено. По этой причине для некоторых мультимедийных задач, следует предусматривать другие, например, ICMP-способы подавления перегрузки. В приложениях типа NFS, где используется UDP, для подавления перегрузки используется упомянутый выше ICMP-механизм. К сожаления в ряде мультимедийных приложений потеря пакетов не контролируется (например IP-телефония). Там шлюз 100-10Мбит/с может восприниматься как канал с вероятностью потери пакета 90%.

Если другие способы испробованы, а перегрузка не исчезла, маршрутизатор начинает отбрасывать приходящие пакеты, которые уже не может обработать. Самое простое - это предоставить случаю выбор отбрасываемых пакетов. Но это не лучшая тактика. В случае пересылки мультимедийных данных предпочтение следует делать для последних полученных пакетов, а "старые" пакеты выбрасывать. При передаче файлов наоборот "старый" пакет имеет приоритет, ведь если его отбросить, придется повторно передавать не только его, но и все последующие пакеты. Некоторые методы передачи изображения требуют передачи время от времени всего кадра с последующей пересылкой только фрагментов, где произошли изменения. В таких условиях потеря пакета, составляющего базовый кадр, менее желательна. Сходные обстоятельства могут возникать и в других приложениях. Можно помечать пакеты, присваивая им определенные уровни приоритетов, что позволит осознанно принимать решение об отбрасывании того или иного пакета в условиях перегрузки. В перспективе проблема может быть решена на чисто коммерческой основе - компонента трафика, помеченная как высоко приоритетная, будет оплачиваться по более высокому тарифу. В некоторых сетях определенное количество пакетов объединяется в группы, образующие сообщение. Если одна ячейка такого сообщения выброшена, все сообщение будет повторно переслано (смотри, например, адаптационные уровни сетей ATM).

**Проблемы адресации**

Когда обсуждались варианты Интернет (разработчики, правда, еще не знали, что это будет Интернет), Ethernet, хотя и не долго, уже существовал. Почему Интернет нельзя было целиком выполнить на основе Ethernet?

Действительно, коммутатор локальной сети существенно дешевле маршрутизатора, Ethernet способен обеспечить уже сегодня пропускную способность 400 Гбит/c и т.д. Список кажущихся выгод можно было бы продолжить.

Проблемы становятся очевидны, если мы задумаемся над задачей маршрутизации (переадресации) пакетов. Каждому адресу в переключателе Ethernet соответствует одна запись в таблице переадресации. Таблица же IP-маршрутизации оперирует блоками адресов, используя префиксы и номера автономных систем. Именно это ограничивает объемы маршрутных таблиц в маршрутизаторах и минимизирует время переадресации. IP-адреса распределяются блоками не менее 256 (С-класс). Именно это позволяет легко с помощью масок разделить локальные и удаленные сетевые объекты. Почему нельзя использовать ту же технику для Ethernet?

Проблема в том, что адреса Ethernet присваиваются изготовителем оборудования, именно это гарантирует их уникальность и в редкой локальной сети можно найти адреса, отличающиеся на 1 в двоичном представлении. Разумеется, Ethernet-адреса можно изменить вручную, но при этом станет трудно, практически невозможно, обеспечить их уникальность. Либо нужно вводить порядок выдачи Ethernet-адресов блоками, как это делается для IP, либо разработать какой-то другой механизм обеспечения уникальности адресов.

Кроме того, когда переключатель обнаруживает незнакомый Ethernet-адрес места назначения, он переадресует такой пакет на все выходы, кроме того через который он пришел. Такой алгоритм исключает многовариантность маршрутов в сети (что допускается в современной IP-сети). Топология локальной сети разрешает только древовидные графы и не допускает петлевые структуры. При этом даже дубликация МАС-адресов во многих случаях не вызыват проблем, если эти адреса находятся в зоне ответственности разных коммутаторов.

Построить Интернет исключительно на переключателях Ethernet невозможно, так как таблица переадресации любого из переключателей в этом случае должна будет содержать слишком много записей, по числу объектов, подключенных к подсети, а время переадресации может стать слишком большим. В локальной сети, построенной исключительно на коммутаторах, число записей в таблице переадресации равно числу Ethernet-объектов в подсети, включая IP-шлюз подключения к Интернет. Построение таких сетей с большим числом машин без использования локального маршрутизатора становится не рациональным.

Именно это обстоятельство сделало актуальной проблему big data – объем информации, при котором время выборки начинает быстро стремиться к бесконечности. Это связано с тем, что в маршрутизаторах и базах данных используется выборка через посредство хэшей (адресация по содержимому).

Разработчики Интернет вынуждены были искать способы минимизации времени переадресации, а это возможно только при ограниченном объеме таблицы переадресации. Именно для этой цели в маршрутизаторах использовались сначала префиксы, а позднее номера автономных систем.

Число IP-адресов приближается к 10 миллиардам. В сочетании с ростом быстродействия каналов это накладывает жесткие требования на время выборки для таблицы переадресации (маршрутизации). На текущий момент все ресурсы уже исчерпаны, остается переходить на географический способ переадресации в рамках IPv6. Но это требует глобального применения этой технологии.