ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«ПОВОЛЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ И ИНФОРМАТИКИ»

Кафедра МСИБ

Киреева Н.В., Буранова М.А.

Изучение алгоритмов управления очередями в среде NS-2

Методические указания к выполнению лабораторной работы

по специальностям: 210700, 090106

Самара, 2014

1. Цель работы

Изучить основные механизмы обеспечения качества обслуживания (QoS). Изучить принципы функционирования следующих алгоритмов управления очередями: PQ, RR, WRR, WIRR. Получить навыки моделирования в среде NS-2.

2. Рекомендуемые источники

1. Галкин А.М., Кучерявый Е.А., Молчанов Д.А. NS-2
2. Кучерявый Е.А. Управление трафиком и качество обслуживания в сети Internet.
3. Кучерявый Е.А. NS-2 Как универсальное средство имитационного моделирования сетей связи.
4. Шринивас Вегешна Качество обслуживания в сетях IP.
5. Олифер В.Г., Олифер Н.А. Компьютерные сети.

1. Общие сведения об очередях

1.1 Образование очередей

Очереди в пакетных сетях возникают в связи с перегрузками. Перегрузкой на сети называется ситуация, когда количество пакетов, передаваемых одновременно по подсети (или ее части) превышает некое пороговое значение, вследствие чего производительность сети начинает снижаться. В результате перегрузок пакеты, ожидающие обслуживания, начинают скапливаться в буфере сетевого устройства, образуя очередь.

1.2 Параметры сетевых соединений

Как правило, основной причиной перегрузок в сетях является то, что каждое приложение предъявляет свои определенные требования к параметрам сетевых соединений. Чтобы более подробно понять причины образования очередей, необходимо рассмотреть эти параметры:

- Полоса пропускания.

Термин **полоса пропускания** (bandwidth) используется для описания номинальной пропускной способности среды передачи информации, протокола или соединения. Этот термин достаточно эффективно определяет “ширину канала”, требующуюся приложению для взаимодействия по сети. Как правило, каждое соединение, нуждающееся в гарантированном качество обслуживания, требует от сети резервирования минимальной полосы пропускания. К примеру, приложения, ориентированные на передачу оцифрованной речи, создают поток информации интенсивностью 64 Кбит/с. Эффективное использование таких приложений становится практически невозможным вследствие снижения полосы пропускания ниже 64 Кбит/с на каком-либо из участков соединения.

- Задержка и дрожание при передаче пакетов.

Задержка при передаче пакета (packet delay), или латентность (latency), на каждом переходе состоит из задержки сериализации, задержки распространения и задержки коммутации. Ниже приведены определения каждого из названных выше типов задержки.

Задержка сериализации (serialization delay). Время, которое требуется устройству на передачу пакета при заданной ширине полосы пропускания. Задержка сериализации зависит как от ширины полосы пропускания канала передачи информации, так и от размера передаваемого пакета. Например, передача пакета размером 64 байт при заданной полосе

пропускания 3 Мбит/с занимает всего лишь 171 нс. Обратите внимание, что задержка сериализации очень сильно зависит от полосы пропускания: передача того же самого пакета размером 64 байт при заданной полосе пропускания 19.2 Кбит/с занимает уже 26 мс. Довольно часто задержку сериализации называют еще задержкой передачи (transmission delay).

Задержка распространения (propagation delay). Время, которое требуется переданному биту информации для достижения принимающего устройства на другом конце канала. Эта величина довольно существенна, поскольку в наилучшем случае скорость передачи информации соизмерима со скоростью света. Обратите внимание, что задержка распространения зависит от расстояния и используемой среды передачи информации, а не от полосы пропускания. Для линий связи глобальных сетей задержка распространения измеряется в миллисекундах. Для трансконтинентальных сетей Соединенных Штатов характерна задержка распространения порядка 30 мс.

Задержка коммутации (switching delay). Время, которое требуется устройству, получившему пакет, для начала его передачи следующему устройству. Как правило, это значение меньше 10 нс.

Дрожание при передаче пакетов. Обычно каждый из пакетов, принадлежащий одному и тому же потоку трафика, передается с различным значением задержки. Задержка при передаче пакетов меняется в зависимости от состояния промежуточных сетей. В том случае, если сеть не испытывает перегрузки, пакеты не ставятся в очередь в маршрутизаторах, а общее время задержки при передаче пакета состоит из суммы за сериализации и задержки распространения на каждом промежуточном переходе.

В этом случае можно говорить о минимально возможной задержке при передаче пакетов через заданную сеть. Следует отметить, что задержка сериализации становится незначительной по сравнению с задержкой распространения при передаче пакета по каналу с большой пропускной способностью. Если же сеть перегружена, задержки при организации очередей в маршрутизаторах начинают влиять на общую задержку при передаче пакетов и приводят к возникновению разницы в задержке при передаче различных пакетов одного и того же потока. Колебания задержки при передаче пакетов получило название дрожания при передаче пакетов (packet jitter). Дрожание пакетов имеет большую важность, поскольку именно оно определяет максимальную задержку при приеме пакетов в конечном пункте назначения. Принимающая сторона, в зависимости от типа используемого приложения, может попытаться компенсировать дрожание пакетов за счет организации приемного буфера для хранения принятых пакетов на время, меньшее или равное верхней границе дрожания.

К этой категории относятся приложения, ориентированные на передачу/прием непрерывных потоков данных, например, Internet-телефония или приложения, обеспечивающие проведение видеоконференций.

- Потеря пакетов.

Уровень **потери пакетов** (packet loss) определяет количество пакетов, отбрасываемых сетью во время передачи. Основными причинами потери пакетов являются перегрузки сети и повреждение пакетов во время передачи по линии связи. Чаще всего отбрасывание пакетов происходит в местах перегрузки, где число поступающих пакетов намного превышает верхнюю границу размера выходной очереди. Кроме того, отбрасывание пакетов может быть вызвано недостаточным размером входного буфера.

Как правило, уровень потери пакетов выражается как доля отброшенных пакетов за определенный интервал времени. Некоторые приложения не способны нормально функционировать или же функционируют крайне неэффективно в случае потери пакетов. Подобные приложения требуют от сети гарантии надежной доставки всех пакетов. Как правило, хорошо спроектированные сети характеризуются очень низким значением потери пакетов. Потеря пакетов также несвойственна приложениям, для которых были заранее зарезервированы требуемые этими приложениями ресурсы. Отбрасывание пакетов, к сожалению, является неизбежным явлением при негарантированной доставке трафика, хотя и в этом случае оно обуславливается крайней необходимостью. Следует отметить, что отброшенные пакеты указывают на неэффективное использование ресурсов сети, часть которых была потрачена на доставку пакетов в точку, где они были потеряны.

1.4 Классификация трафика

Как уже было сказано выше, каждое приложение предъявляет свои требования к определенным характеристикам сетевых соединений, теперь, когда нам известны эти характеристики, мы можем рассмотреть классификацию приложений на основе этих характеристик:

Относительная предсказуемость скорости передачи данных;

В отношении предсказуемости скорости передачи данных приложения делятся на два больших класса: приложения с потоковым трафиком и приложения с пульсирующим трафиком.

Приложения с потоковым трафиком (stream) порождают равномерный поток данных, который поступает в сеть с **постоянной**

битовой скоростью (Constant Bit Rate, CBR). В случае коммутации пакетов трафик таких приложений представляет собой последовательность пакетов одинакового размера (равного B бит), следующих друг за другом через один и тот же интервал времени T (рис. 1).

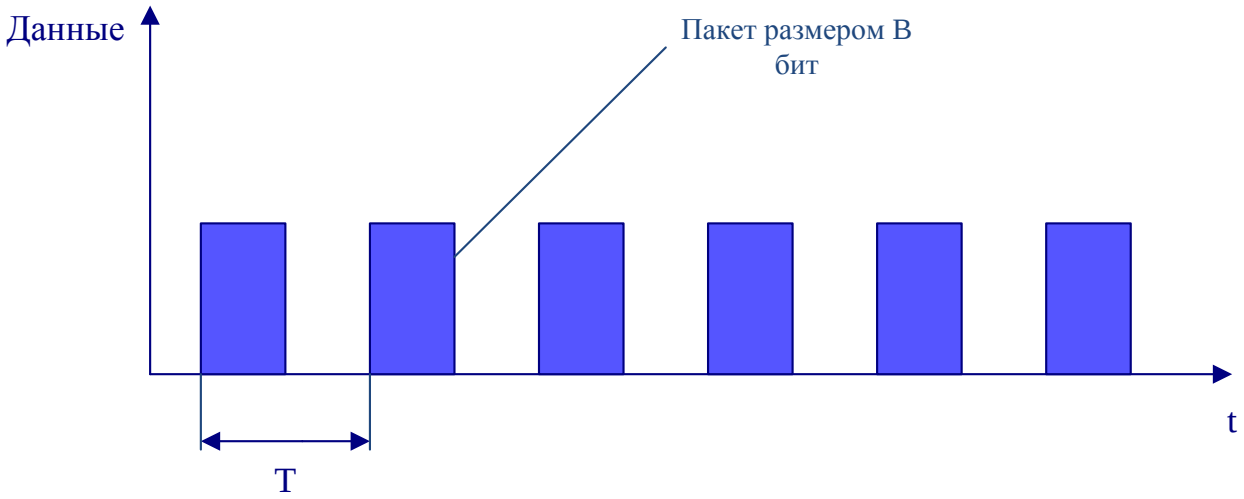

рис.1.1 Потоковый трафик

Постоянная битовая скорость потокового трафика (CBR) может быть вычислена путем усреднения на одном периоде:

бит/с.

В общем случае, постоянная битовая скорость потокового трафика меньше номинальной максимальной битовой скорости протокола, с помощью которого передаются данные, так как между пакетами существуют паузы. Так, например, для протокола Ethernet максимальная скорость будет составлять 9,76 Мбит/сек (для кадров максимальной длины), что меньше номинальной скорости этого протокола, равной 10 Мбит/сек.

Приложения с пульсирующим трафиком (burst) отличаются высокой степенью непредсказуемости, в этих приложениях периоды молчания сменяются пульсацией, в течение которой пакеты «плотно» следуют друг за другом. В результате трафик характеризуется **переменной битовой скоростью** (Variable Bit Rate, VBR), что изображено на рисунке 2.3. Так, при работе приложений файлового сервиса интенсивность трафика, генерируемого приложением, может падать до нуля, когда файлы не передаются, и повышаться до максимально доступной, ограниченной только возможностями сети, когда файловый сервер передает файл.

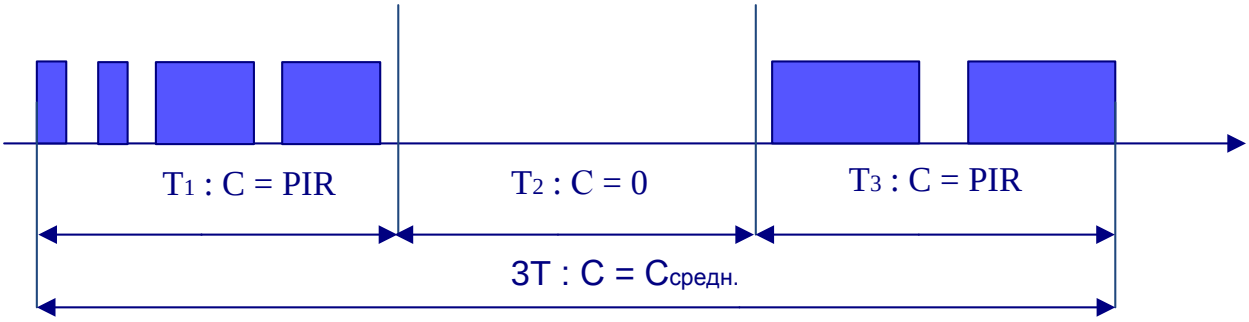

Рис. 1.2 Пульсирующий трафик

На рисунке показано три периода изменений T1, T2 и T3. Для упрощения расчетов принято, что пиковые скорости на первом и третьем периодах равны между собой и равны PIR, а все три периода имеют одинаковую длительность T. Учитывая это, можно вычислить величину пульсации B, которая равна количеству битов, переданных на периоде пульсации:

Таким образом величина пульсации для периодов T1 и T3 равна B, а на периоде T2 – нулю.

Для приведенного примера можно подсчитать коэффициент пульсации. (Равный отношению пиковой скорости на каком-либо небольшом периоде времени к средней скорости трафика, измеренной на длительном периоде времени.) Так как пиковая скорость на периоде T1 (или T3) равна B/T, а средняя скорость на суммарном периоде T1+ T2+ T3 составляет 2B/3T, коэффициент пульсации равен 3/2.

Практически любой трафик, даже трафик потоковых приложений, имеет ненулевой коэффициент пульсации. Просто значения коэффициентов пульсации у потокового и пульсирующего трафика значительно различаются. Так, у приложений с пульсирующим трафиком он обычно находится в пределах от 2 до 100, а у потоковых приложений он близок к 1. В локальных сетях этот коэффициент обычно выше, чем в глобальных, так как на магистралях глобальных сетей трафик представляет собой сумму трафиков многих источников, что по закону больших чисел приводит к сглаживанию результирующего трафика.

Чувствительность трафика к задержкам пакетов;

Еще один критерий классификации приложений по типу трафика – их чувствительность к задержкам пакетов и их вариациям. Далее приведены основные типы приложений в порядке повышения их чувствительности к задержкам пакетов.

- Асинхронные приложения. Практически не имеют ограничений на время задержки (т.н. эластичный трафик). Примером такого приложения может служить электронная почта.
- Интерактивные приложения. Задержки могут быть замечены пользователями, но они не сказываются на функциональности приложений негативно. Пример – текстовый редактор, работающий с удаленным файлом.
- Изохронные приложения. Такие приложения имеют определенный порог чувствительности к вариациям задержек, при превышении которого функциональность приложения резко снижается. Примером может выступать передача голоса, когда при превышении порога вариации задержек в 100-150 мс резко снижается качество воспроизводимого голоса.
- Сверхчувствительные к задержкам приложения. У этого вида приложений задержка доставки данных сводит у нулю их функциональность. Пример – приложения, осуществляющие управление техническим объектом в реальном времени. При запаздывании управляющего сигнала на объекте может произойти авария.

Вообще говоря, интерактивность приложений всегда повышает их чувствительность к задержкам. Например, широковещательная рассылка аудиоинформации может выдерживать значительные задержки передачи пакетов (оставаясь чувствительной к их вариациям), а интерактивный телефонный или видеоразговор их не терпит, что хорошо заметно при трансляции через спутник. Длительные паузы в разговоре вводят собеседников в заблуждение, зачастую они теряют терпение и начинают очередную фразу одновременно.

Наряду с приведенной выше классификацией, тонко дифференцирующей чувствительность приложений к задержкам и их вариациям, существует и более грубое деление приложений по этому же признаку на два класса: асинхронные и синхронные. К асинхронным относятся те приложения, которые нечувствительны к задержкам передачи данных в очень широком диапазоне, вплоть до нескольких секунд, а все остальные приложения, на функциональность которых задержки влияют существенно, относят к синхронным приложениям. Интерактивные приложения могут быть отнесены как к асинхронным (текстовый редактор), так и к синхронным (видеоконференция).

Чувствительность трафика к потерям и искажениям пакетов.

И, наконец, последним критериям классификации приложений является их чувствительность к потерям пакетов. Здесь обычно делят приложения на две группы.

Приложения, чувствительные к потере данных. Практически все приложения, передающие алфавитно-цифровые данные (к которым относятся текстовые документы, коды программ, числовые массивы и т.д.), обладают высокой чувствительностью к потере отдельных, даже небольших фрагментов данных. Такие потери часто ведут к полному обесцениванию остальной успешно принятой информации. Например отсутствие хотя бы одного байта в коде программы делает ее совершенно неработоспособной. Все традиционные сетевые приложения (файловый сервис, сервис баз данных, электронная почта и т.д.) относятся к этому типу приложений.

Приложения, устойчивые к потере данных. К этому типу относятся многие приложения, передающие трафик с информацией об инерционных физических процессах. Устойчивость к потерям объясняется тем, что небольшое количество отсутствующих данных можно определить на основе принятых. Так, при потере одного пакета, несущего несколько последовательных замеров голоса, отсутствующие замеры при воспроизведении голоса могут быть заменены аппроксимацией на основе соседних значений. К такому типу относится большая часть приложений, работающих с мультимедийным трафиком (аудио- и видеоприложения). Однако устойчивость к потерям имеет свои пределы, поэтому процент потерянных пакетов не может быть большим (например, не более 1%). Можно так же отметить, что не любой мультимедийный трафик столь устойчив к потерям данных, так, компрессированный голос и видеоизображение очень чувствительны к потерям, поэтому относятся к первому типу приложений.

1.4 Классы приложений

Стоит отметить, что между значениями трех перечисленных выше характеристик качества обслуживания (относительная предсказуемость скорости передачи данных; чувствительность трафика к задержкам пакетов; чувствительность трафика к потерям и искажениям пакетов) нет строгой взаимосвязи. Другими словами, приложение с равномерным потоком может быть, как асинхронным, так и синхронным, а, например, синхронное приложение может быть, как чувствительным к потерям пакетов, так и не чувствительным к ним. Тем не менее практика показывает, что из всего многообразия возможных сочетаний значений трех этих характеристик есть несколько таких сочетаний, которые

охватывают большую часть приложений, используемых на сегодняшний день.

Например, следующее сочетание характеристик приложения «порождаемый трафик – равномерный поток, приложение изохронное, устойчивое к потерям» соответствует таким популярным приложениям, как IP-телефония, поддержка видеоконференций, аудиовещание через Интернет. Устойчивых сочетаний характеристик, описывающих определенный класс приложений, существует не так уж и много. Так, при стандартизации технологии ATM, которая изначально разрабатывалась для поддержания различных типов трафика, были определены 4 класса трафика (и соответствующих приложений): A, B, C и D. Для каждого из этих классов рекомендуется использовать собственный набор характеристик качества обслуживания. Кроме того, для всех приложений, не включенных ни в один из перечисленных классов, был определен класс X, в котором сочетание характеристик приложения может быть произвольным. Классификация ATM приведена ниже (табл. 1.1)

Класс трафика	Характеристики
A	Постоянная битовая скорость, чувствительность к задержкам, передача с установлением соединения (например, голосовой трафик, трафик телевизионного изображения). Параметры QoS: пиковая скорость передачи данных, задержка, джиттер
B	Переменная битовая скорость, чувствительность к задержкам, передача с установлением соединения (например, компрессированный голос, компрессированное видеоизображение). Параметры QoS: пиковая скорость передачи данных, пульсация, средняя скорость передачи данных, задержка, джиттер
C	Переменная битовая скорость, эластичность, передача с установлением соединения (например, трафик компьютерных сетей, в которых конечные узлы работают по протоколам с установлением соединений – frame relay, X.25, TCP). Параметры QoS: пиковая скорость передачи данных, пульсация, средняя скорость передачи данных.
D	Переменная битовая скорость, эластичность, передача без установления соединения (например, трафик компьютерных сетей, в которых конечные узлы работают по протоколам без установления соединений – IP/UDP, Ethernet). Параметры QoS не определены.
X	Тип трафика и его параметры определяются пользователем

Таблица 1.1 Классы приложений

1.4 Модель M/M/1

Наиболее часто для описания очереди в сетях с коммутацией пакетов используется модель из теории очередей, получившая название M/M/1.

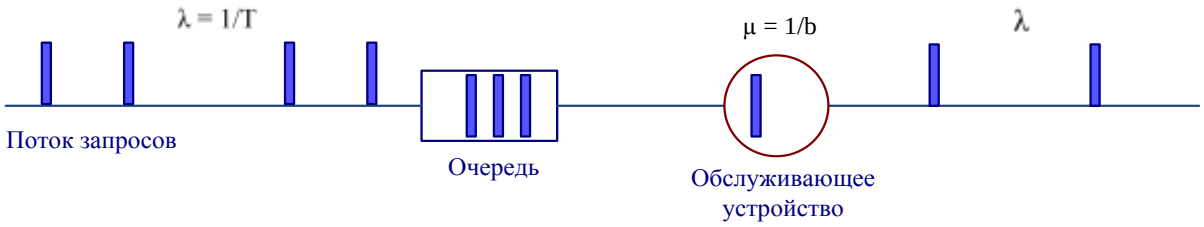

Рисунок 1.3 Модель M/M/1

В названии модель M/M/1: первая буква M обозначает тип распределения интервалов поступления заявок (пуассоновское распределение), вторая – тип распределения значений времени обслуживания (так же пуассоновское), а единица означает, что моделируется только одно обслуживающее устройство.

Основными элементами модели являются:

- Входной поток абстрактных заявок на обслуживание
- Буфер
- Обслуживающее устройство
- Выходной поток обслуженных заявок

Заявки поступают на вход буфера в случайные моменты времени. Если в момент поступления заявки буфер пуст и обслуживающее устройство свободно, то заявка сразу же передается в это устройство для обслуживания. Обслуживание также длится случайное время. Если в момент поступления заявки буфер пуст, но обслуживающее устройство занято обслуживанием заявки, поступившей ранее, то заявка ожидает его завершения в буфере. Как только обслуживающее устройство завершает обслуживание очередной заявки, она передается на выход, а прибор выбирает из буфера следующую заявку (если буфер не пуст). Выходящие из обслуживающего устройства заявки формируют выходной поток. Буфер считается бесконечным, то есть заявки никогда не теряются из-за того, что исчерпана его емкость.

Если прибывшая заявка застает буфер не пустым, то она становится в очередь и ожидает обслуживания. Заявки выбираются из очереди в порядке поступления, то есть соблюдается дисциплина обслуживания **первым пришел – первым обслужен** (First-In, First-Out, FIFO).

Теория очередей позволяет оценить для этой модели среднюю длину очереди и среднее время ожидания заявки в зависимости от характеристик входного потока и времени обслуживания.

Будем считать, что среднее время между поступлениями заявок известно и равно T. Это значит, что интенсивность поступления заявок, которая традиционно обозначается в теории очередей символом λ, равна

, заявок в секунду

Будем также считать, что среднее время обслуживания заявки равно b. Это значит, что обслуживающий прибор способен продвигать заявки на выход с интенсивностью . Опять же для получения аналитического результата принято считать, что время обслуживания – случайная величина с пуассоновской плотностью распределения. Принятие таких предположений позволяет нам выразить среднее время ожидания заявки в очереди, которое мы обозначим через w:

Здесь через p обозначено отношение λ/μ.

Параметр p называют коэффициентом использования (utilization) обслуживающего прибора. Для любого периода времени этот показатель равен отношению времени занятости обслуживающего прибора к величине этого периода.

Зависимость среднего времени ожидания w от p иллюстрирует рисунок 1.4. Как видно из поведения кривой, параметр p играет ключевую роль в образовании очереди. Если значение p близится к нулю, то среднее время ожидания также очень близко к нулю. Это означает, что заявки почти никогда не ожидают обслуживания в буфере, который оказывается пустым в момент их прихода, а сразу поступают на обслуживающее устройство. И наоборот, если p стремится к 1, то время ожидания растет очень быстро и нелинейно (и в пределе равно бесконечности). Такое поведение очереди интуитивно понятно, ведь p – это отношение средней интенсивности входного потока к средней интенсивности его обслуживания. Чем ближе средние значения интервалов между пакетами среднему времени обслуживания, тем сложнее обслуживающему устройству справляться с нагрузкой.

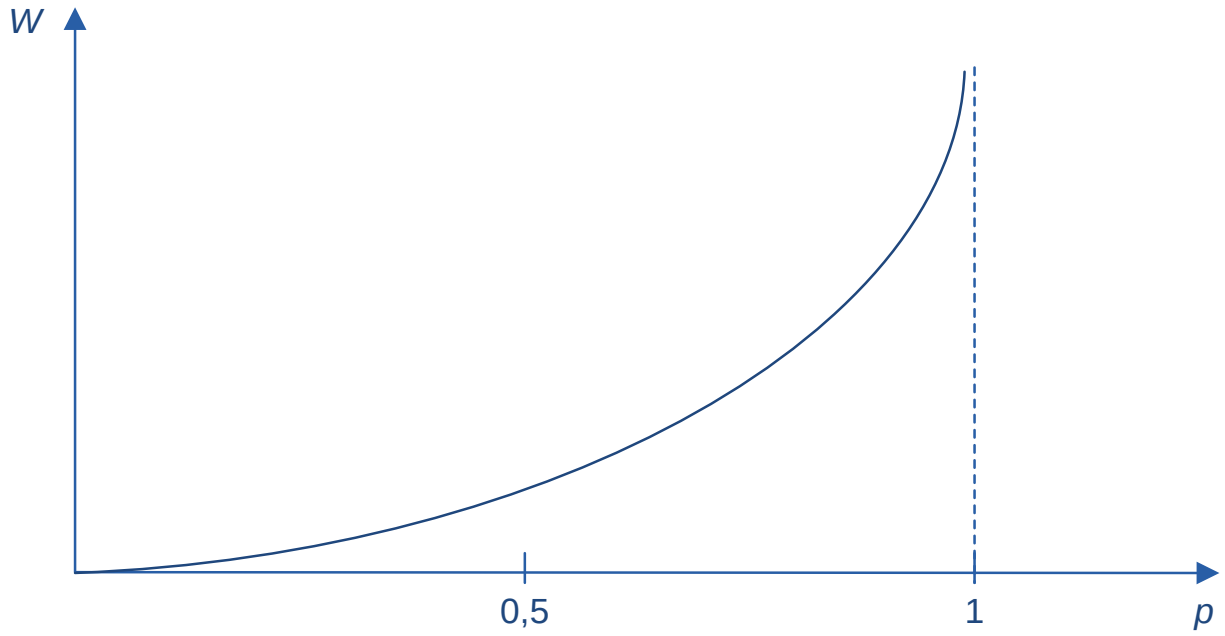

Рисунок 3.3 Зависимость среднего времени ожидания заявки от коэффициента использования ресурса

С помощью модели M/M/1 можно приближенно моделировать сеть с коммутацией пакетов (рис 1.5). Так, входной поток пакетов, поступающих на вход интерфейса коммутатора (обобщим этим термином все устройства коммутации пакетов), представлен в модели потоком заявок, а буфер модели соответствует буферу интерфейса коммутатора. Среднее время обслуживания заявки соответствует среднему времени продвижения пакета процессором коммутатора из входного буфера в выходной канал.

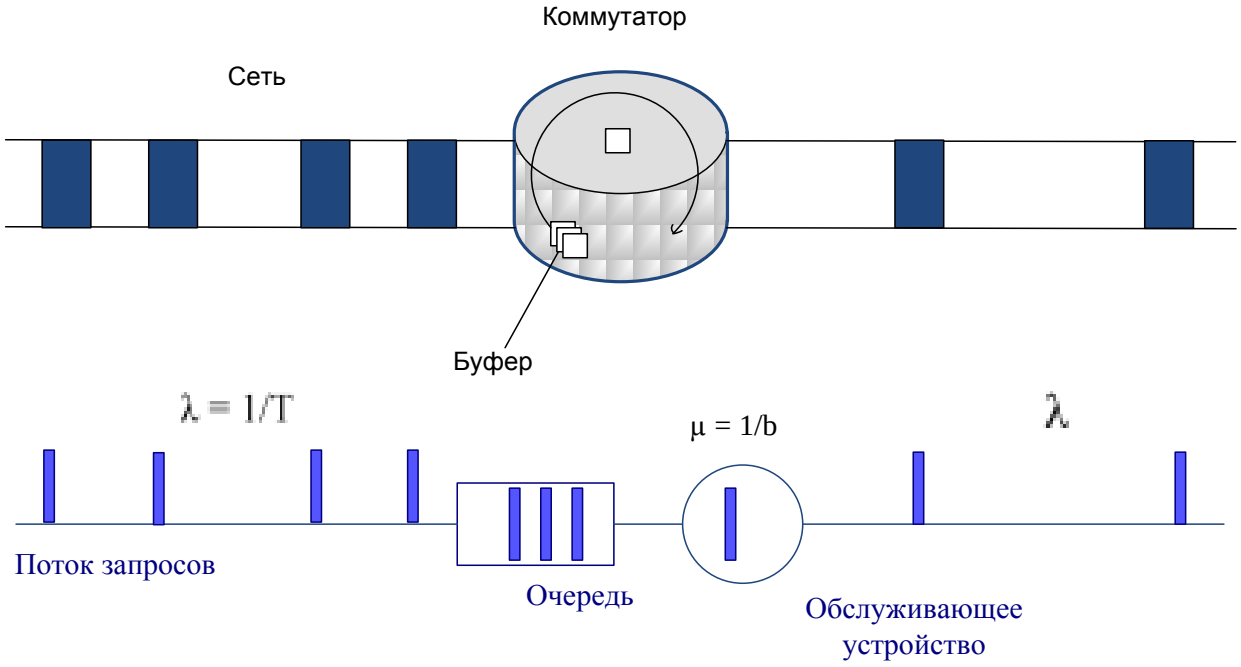

Рис. 1.5 Выходной интерфейс как разделяемый ресурс коммутатора

Понятно, что приведенная модель очень упрощенно описывает процессы, происходящие в коммутаторе. Тем не менее она очень полезна для понимания основных факторов, влияющих на величину очереди.

1.5 Влияния пульсации трафика на возникновение очередей

Важным параметром, влияющим на образование очередей, является вариация интервалов входного потока пакетов, то есть пульсация входного трафика. В ранее рассмотренной модели теории очередей мы предполагали, что входной поток описывается пуассоновским распределением, имеющим довольно большое стандартное отклонение вариации (средняя вариация равна T при среднем значении интервала T, а коэффициент вариации равен 1). Но что будет, если вариация интервалов входного потока будет меньше, или если входной поток окажется сверхпульсирующим? К сожалению, модели теории очередейне дают для этих случаев простых аналитических зависимостей. Поэтому для получения результатов приходится применять методы имитационного моделирования сетей или производить измерения в реальной сети. На рисунке 3.5 показано семейство зависимостей w от p, полученных для разных значений коэффициента вариации CV входного потока. Из рисунка видно, что чем меньше пульсирует входной поток (CV приближается к нулю), тем меньше проявляется эффект лавинообразного образования очереди при приближении загрузки ресурса к 1. И наоборот, чем больше CV, тем раньше (при меньших значениях p) начинает этот эффект проявляться.

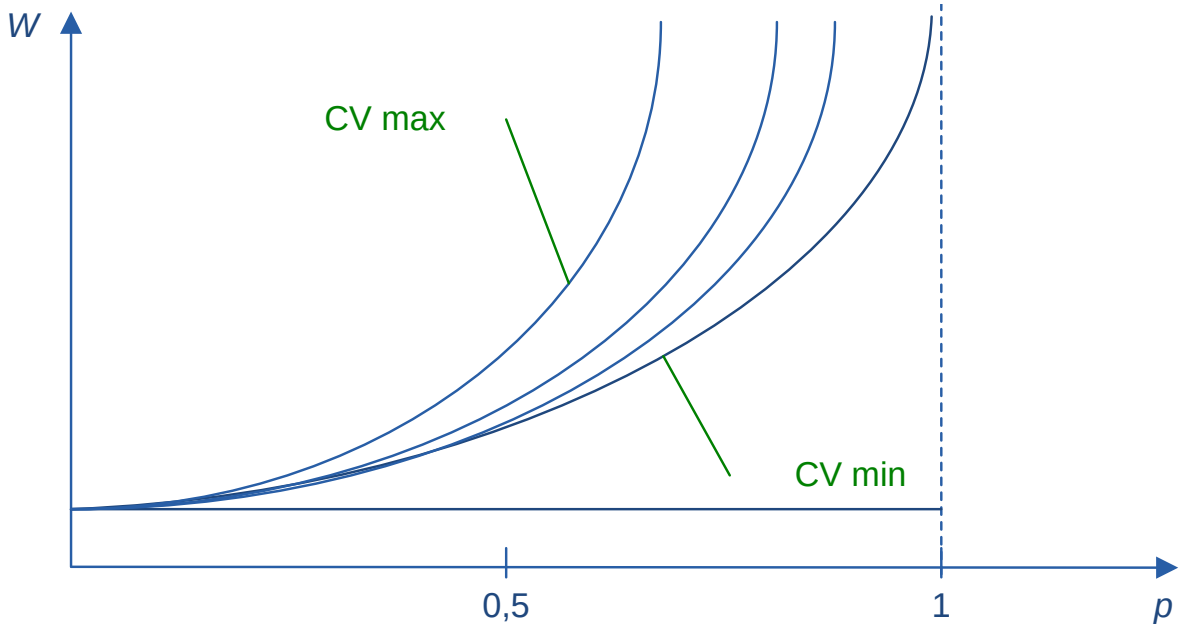

Рис. 1.6 Влияние степени пульсации на задержки

Из поведения графиков на рисунке можно сделать два вывода: во-первых, для оценки значений задержек в очередях на коммутаторах сети недостаточно информации о коэффициенте загрузки p, необходимо также знать параметры пульсации трафика; во-вторых, для снижения уровня задержек необходимо снижать значение p и уменьшать пульсацию трафика.

2. Механизмы управления очередями

Механизмы управления очередями нужны для работы в периоды временных перегрузок, когда сетевое устройство не справляется с передачей пакетов на выходной интерфейс в том темпе, в котором они поступают. Если причиной перегрузки является недостаточная производительность процессорного блока сетевого устройства, то необработанные пакеты временно накапливаются во входной очереди соответствующего входного интерфейса. Очередей к входному интерфейсу может быть несколько, если дифференцировать запросы на обслуживание по нескольким классам. В том же случае, когда причина перегрузки заключается в ограниченной пропускной способности выходного интерфейса, пакеты временно сохраняются в выходной очереди (или очередях) этого интерфейса.

2.1 Очередь FIFO

В очереди FIFO в случае перегрузки все пакеты перемещаются в одну общую очередь и выбираются из нее в том порядке, в котором поступили. Во всех устройствах с коммутацией пакетов алгоритм FIFO используется по умолчанию, так что такая очередь зачастую называется очередью «по умолчанию». Достоинствами этого подхода является простота реализации и отсутствие потребности в конфигурировании. Однако этому методу также присущ серьезный недостаток, а именно невозможность дифференцированной обработки пакетов различных потоков. Все пакеты стоят в общей очереди на равных основаниях. Вместе оказываются и пакеты, чувствительные к задержкам, например, голосового трафика, и пакеты нечувствительного к задержкам, но очень интенсивного трафика резервного копирования, длительные пульсации которого могут надолго задержать голосовой пакет. Структура очереди FIFO представлена на рисунке 2.1.

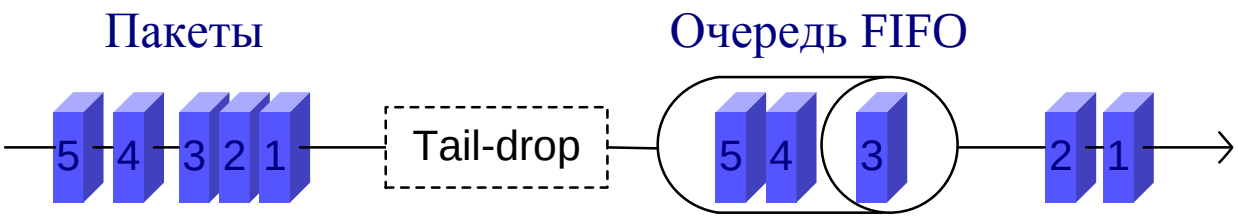

Рис. 2.1 Очередь FIFO

Традиционный механизм обслуживания очередей FIFO предусматривает отбрасывание всех входящих пакетов после достижения максимального значения длины очереди. Подобный способ управления очередью получил название “отбрасывание хвоста” (taildrop) и характеризуется тем, что сигнал о перегрузке поступает лишь в момент фактического переполнения очереди. К сожалению, механизм FIFO не предусматривает проведения каких-либо активных действий по предотвращению перегрузки или по уменьшению размера очереди с целью снижения времени задержки.

2.2 Приоритетное обслуживание

Очереди с приоритетным обслуживанием (Priority Queue, PQ) очень популярны во многих областях вычислительной техники, в частности в операционных системах, когда одним приложениям нужно отдать предпочтение перед другими при обработке их в мультипрограммной смеси. Применяются эти очереди и для преимущественной по сравнению с другими обработки одного класса трафика.

Механизм приоритетного обслуживания основан на разделении всего сетевого трафика на небольшое количество классов и последующего назначения каждому классу некоторого числового признака – приоритета.

Классификация трафика представляет собой отдельную задачу. Пакеты могут разбиваться на отдельные классы на основании различных признаков, таких как адрес источника, адрес назначения и любых других комбинаций признаков, содержащихся в заголовках пакетов.

Приоритеты могут быть назначены не только коммутатором или маршрутизатором, но и приложением на узле-отправителе. Также необходимо учитывать, что если в сети отсутствует централизованная политика назначения приоритетов, каждое устройство в сети может не согласиться с приоритетом, назначенным данному пакету в другой точке сети. В этом случае значение приоритета будет переписано в соответствии с локальной политикой, принятой непосредственно на данном устройстве.

В сетевом устройстве, поддерживающем приоритетное обслуживание, имеется несколько очередей (буферов) – по одной для

каждого приоритетного класса. Пакет, поступивший в период перегрузки, помещается в очередь соответствующую его приоритетному классу. На рисунке 2.2 приведен пример использования четырех приоритетных очередей с высоким, средним, нормальным и низким приоритетами, каждый из которых окрашен в свой цвет (красный, синий, зеленый и серый соответственно). До тех пор, пока из более приоритетной очереди не будут выбраны все имеющиеся в ней пакеты, устройство не переходит к обработке следующей, менее приоритетной очереди. Поэтому пакеты с низким приоритетом обрабатываются только тогда, когда пустеют все вышестоящие очереди: с высоким, средним и нормальным приоритетами.

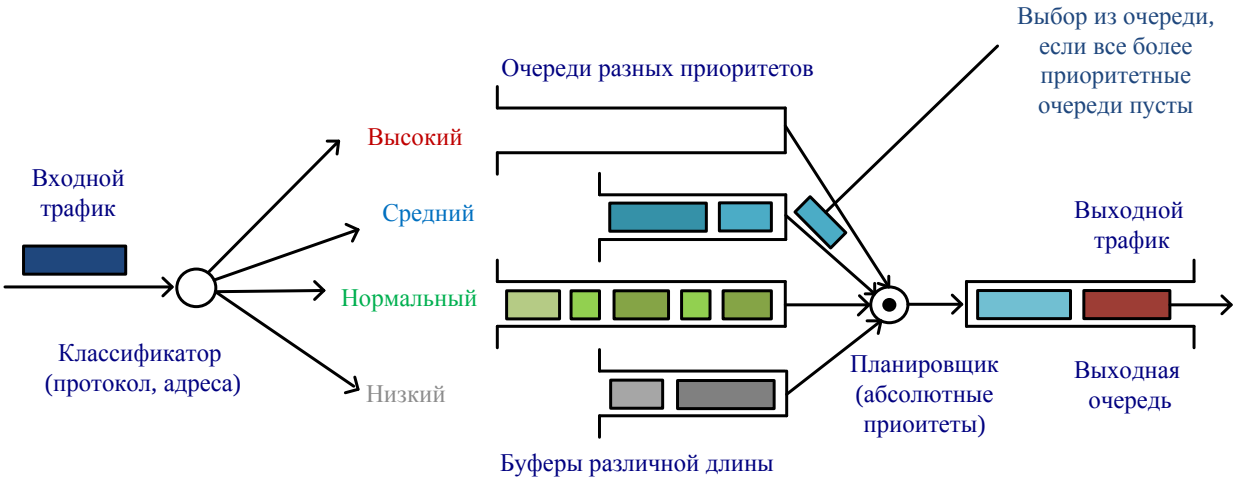

Рис. 2.1 Приоритетное обслуживание

Размер буфера сетевого устройства определяет максимальную длину очереди ожидающих обслуживания пакетов, если пакет поступает при заполненном буфере, то он просто отбрасывается. Обычно по умолчанию всем приоритетным очередям отводятся одинаковые буферы, но многие устройства разрешают администратору назначать каждой очереди буфер индивидуального размера. Размер буфера определяется в индивидуальном случае таким образом, чтобы его хватило с некоторым запасом для хранения очереди среднестатистической длины. Однако установить подобное значение довольно тяжело, так как оно изменяется в зависимости от нагрузки сети, поэтому требуется постоянное и длительное наблюдение за работой сети. В общем случае, чем выше значимость трафика для предприятия, чем больше его интенсивность и пульсации, тем больший размер буфера требуется этому трафику. В примере, изображенном на рисунке 3.8, для трафика высшего и нормального приоритетов выбраны большие размеры буферов, а для остальных двух классов меньшие. Мотивы принятого решения для наивысшего приоритета очевидны, а трафик нормального приоритета имеет, очевидно, высокую интенсивность и значительный коэффициент

пульсаций. Приоритетное обслуживание очередей обеспечивает высокое качество обслуживания для пакетов из самой приоритетной очереди. Если средняя интенсивность их поступления в устройство не превосходит пропускной способности выходного интерфейса (и производительности внутренних продвигающих блоков самого устройства), то пакеты высшего приоритета всегда получают ту пропускную способность, которая им нужна. Уровень задержек высокоприоритетных пакетов также минимален. Однако он не нулевой и зависит в основном от характеристик потока этих пакетов – чем выше пульсации потока и его интенсивность, тем вероятнее возникновения очереди, образованной пакетами данного высокоприоритетного потока. Трафик всех остальных приоритетных классов почти прозрачен для пакетов высшего приоритета. Слово «почти» относится к ситуации, когда высокоприоритетный пакет вынужден ждать завершения обслуживания низкоприоритетного пакета, если время его прибытия совпало с временем начала продвижения низкоприоритетного пакета на выходной интерфейс. Что же касается остальных приоритетных классов, то качество их обслуживания будет ниже, чем у самых высокоприоритетных пакетов, причем уровень их снижения может быть очень существенным. Если коэффициент нагрузки выходного интерфейса, определяемый только трафиком высшего приоритетного класса приближается в какой-то момент к единице, то трафик остальных классов просто замораживается на это время. Поэтому приоритетное обслуживание обычно применяется для чувствительного к задержкам трафика, имеющего небольшую интенсивность. При таких условиях обслуживание этого класса не слишком ущемляет обслуживание остального трафика. Например, голосовой трафик чувствителен к задержкам, но его интенсивность как правило редко превышает 8-16 Кбит/с, так что при назначении ему наивысшего приоритета ущерб остальным классам оказывается на очень значительным.

2.3 Алгоритм Round Robin

Алгоритм кругового обслуживания Round Robin используется в том случае, если канал сильно загружен и необходимо предоставить приложениям одинаковую возможность по передаче данных.

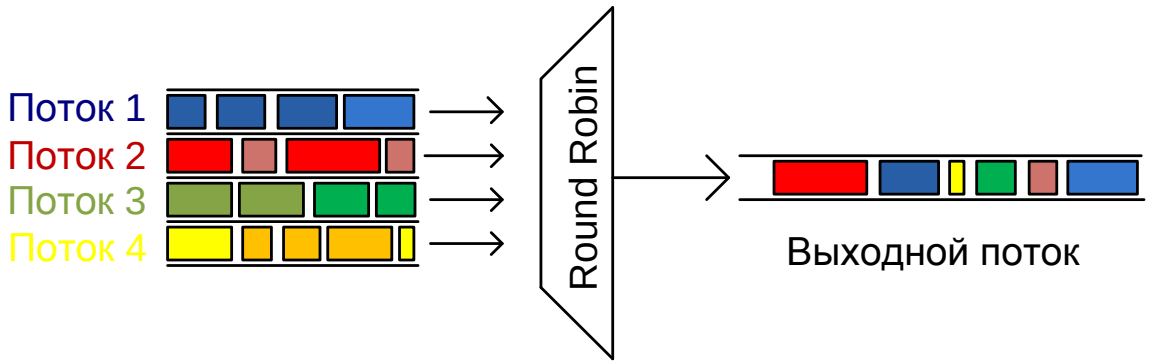

Рисунок 2.2 Круговое обслуживание

Из примера, изображенного на рисунке 2.2 видно, что при наличии нескольких входных потоков алгоритм RR выбирает из каждого по одному пакету и формирует выходную последовательность. После того, как выборка пакетов из каждого потока произведена, она повторяется снова. Таким образом обеспечивается равное обслуживание каждого потока. Недостатком является то, что одно приложение, открыв много потоков, может заглушить остальные подключения.

2.4 Взвешенное обслуживание

Механизм взвешенного обслуживания очередей (Weighted Queuing, WQ) разработан для того, чтобы можно было предоставить всем классам трафика определенный минимум пропускной способности. Под весом данного класса понимается процент, предоставляемой классу трафика пропускной способности от полной пропускной способности выходного интерфейса. При взвешенном облуживании, так же, как и при приоритетном, весь трафик делится на несколько классов, и для каждого класса определяется отдельная очередь пакетов. Но с каждой очередью связывается не приоритет, а процент пропускной способности ресурса, гарантируемый данному классу трафика при перегрузках этого ресурса. Для входного потока таким ресурсом является процессор, а для выходного (после выполнения коммутации) – выходной интерфейс. Пример такого типа обслуживания приведен на рисунке 2.3. На данном рисунке представлено устройство для пяти классов трафика, поддерживающее пять очередей к выходному интерфейсу коммутатора. Этим очередям при перегрузках выделяется соответственно 10, 10, 30, 20 и 30% пропускной способности выходного интерфейса.

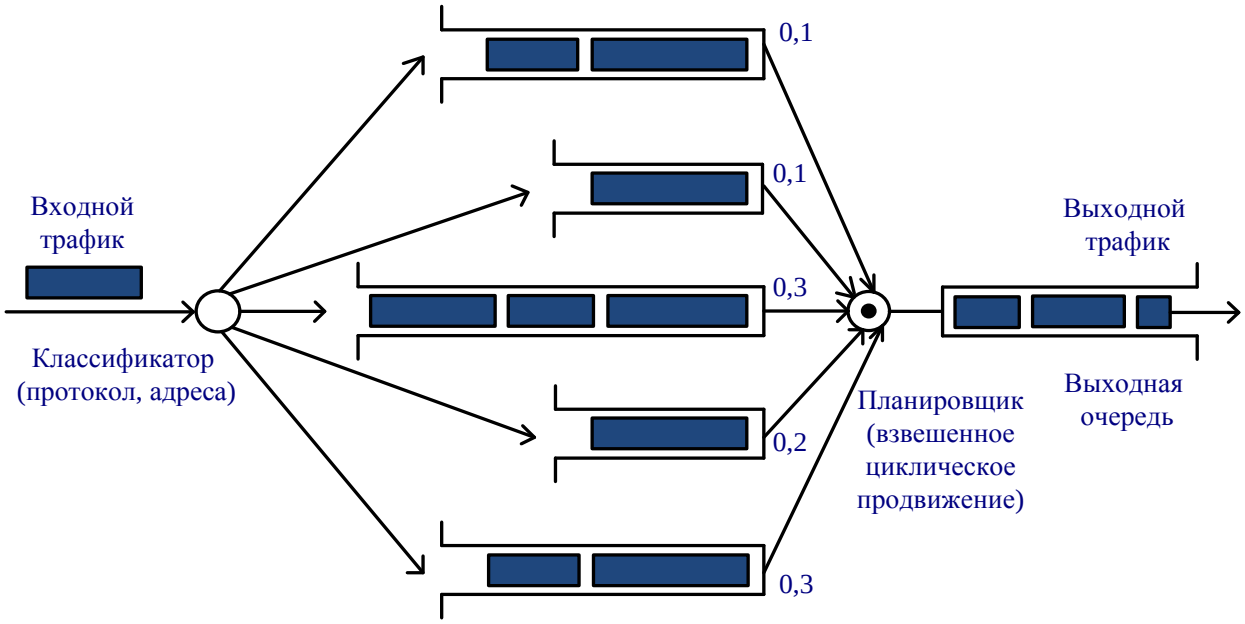

Рис. 2.3 Взвешенное обслуживание

Достигается подобный результат за счет того, что очереди обслуживаются последовательно и циклически, и в каждом цикле обслуживания из каждой очереди выбирается число байт, соответствующее весу данной очереди. Так если цикл просмотра очередей в примере, изображенном на рисунке 3.9, равен одной секунде, а скорость выходного интерфейса составляет 100 Мбит/с, то при перегрузке в каждом цикле первой очереди уделяется 10% времени, то есть 100 мс и выбирается 10 Мбит данных, из второй тоже 10 Мбит, из третьей – 30 Мбит, из четвертой – 20 Мбит, из пятой – 30 Мбит.

Так как данные выбираются из очереди пакетами, а не битами, то реальное распределение пропускной способности между классами трафика всегда немного отличается от планируемого. Так, в предыдущем примере вместо 10% первый класс трафика мог бы получать при перегрузке 9 или 12%. Чем больше время цикла, тем точнее соблюдаются требуемые пропорции между классами трафика, так как из каждой очереди выбирается большее число пакетов, и влияние размера каждого пакета усредняется. В то же время длительный цикл приводит к большим задержкам передачи пакетов. Так, при выбранном нами для примера цикле в одну секунду задержка может составить одну секунду и больше – ведь арбитр возвращается к каждой очереди не чаще, чем раз в секунду, кроме того, в очереди может находиться более одного пакета. Поэтому при выборе времени цикла нужно обеспечить баланс между точностью соблюдения пропорций пропускной способности и стремлением к снижению задержки.

Для рассмотренного нами примера время цикла в 1000 мкс является примером такого баланса. С одной стороны, оно обеспечивает обслуживание очереди каждого класса каждые 1000 мкс, а значит более низкий уровень задержек, а с другой, это времени достаточно, чтобы выбрать из каждой очереди в среднем по несколько пакетов (первой очереди в этом примере будет отводиться 100 мкс, что достаточно для передачи в выходной канал одного пакета Fast Ethernet или десяти пакетов Gigabit Ethernet).

На уровень задержек и вариации задержек пакетов для некоторого класса трафика при взвешенном обслуживании в значительной степени влияет относительный коэффициент использования. В этом случае коэффициент подсчитывается как отношение интенсивности входного трафика к пропускной способности, выделенной этому классу в соответствии с его весом. Например, если мы выделили первой очереди 10% от общей пропускной способности выходного интерфейса, то есть 10 Мбит/с, а средняя интенсивность потока, который попадает в эту очередь, равна 3 Мбит/с, то коэффициент использования для этого потока составляет 3/10 = 0,3. Зависимость на рисунке 3.3 показывает, что задержки при таком значении коэффициента использования будут незначительными. Если бы интенсивность входного потока этой очереди была 9 Мбит/с, то очереди были бы значительными, а при превышении предела в 10 Мбит/с часть пакетов потока постоянно бы отбрасывалась из-за переполнения очереди.

Качественное поведение очереди и, соответственно, задержек здесь выглядит примерно так же, как и в случае очереди FIFO – чем меньше коэффициент загрузки, тем меньше средняя длина очереди и тем меньше задержки. Как и для приоритетного обслуживания, при взвешенном обслуживании администратор может назначать разным классам очередей буферы разных размеров. Уменьшение размеров буферов для очередей ведет к росту числа потерь пакетов при перегрузках, но зато снижает время ожидания для тех пакетов, которые не были отброшены и попали в очередь.

2.5 Справедливое взвешенное обслуживание

Взвешенное справедливое обслуживание (Weighted Fair Queuing, WFQ) — это комбинированный механизм обслуживания очередей, сочетающий приоритетное обслуживание со взвешенным. Производители сетевого оборудования предлагают многочисленные собственные реализации WFQ, отличающиеся способом назначения весов и поддержкой различных режимов работы, поэтому в каждом конкретном случае необходимо внимательно изучить все детали поддерживаемого WFQ. Наиболее распространенная схема предусматривает существование одной особой очереди, которая обслуживается по приоритетной схеме — всегда

в первую очередь и до тех пор, пока все заявки из нее не будут исполнены. Эта очередь предназначена для системных сообщений, сообщений управления сетью и, возможно, пакетов наиболее критических и требовательных приложений. Во всяком случае, предполагается, что её трафик имеет невысокую интенсивность, поэтому значительная часть пропускной способности выходного интерфейса остается другим классам трафика.

Остальные очереди устройство просматривает последовательно, в соответствии с алгоритмом взвешенного обслуживания (см. Рисунок 3.10). Администратор может задать вес для каждого класса трафика аналогично тому, как это делается в случае взвешенного обслуживания. Вариант работы по умолчанию предусматривает для всех остальных классов трафика равные доли пропускной способности выходного интерфейса (за вычетом оставшейся от приоритетного трафика).

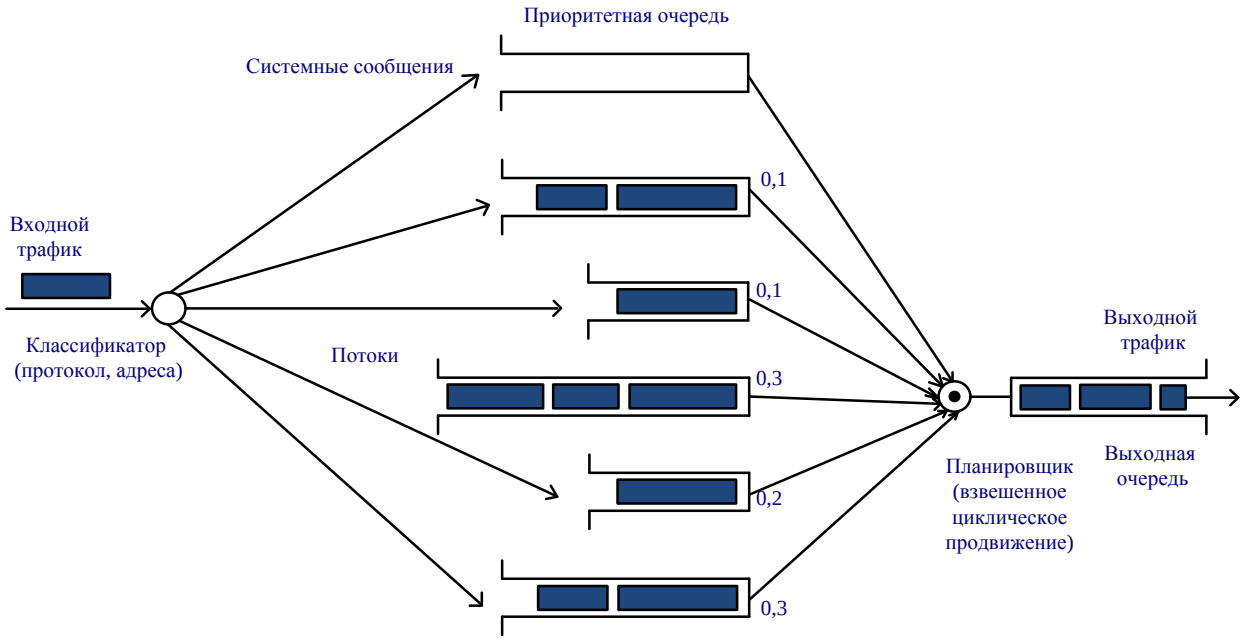

Рисунок 2.4 Справедливое взвешенное обслуживание

3. Схема моделируемой сети

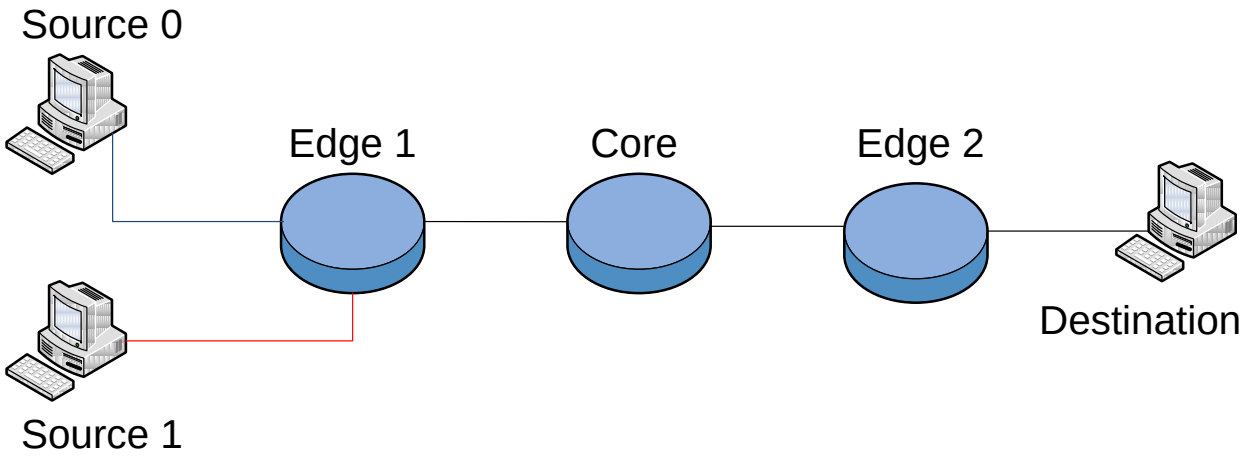

Рисунок 3.1 Схема исследуемой сети

Разберем приведенную схему более подробно. Она состоит из шести функциональных блоков: двух компьютеров-источников Source 1 и Source 2, одного компьютера-получателя Destination, двух пограничных маршрутизаторов Edge 1 и Edge 2, и центрального маршрутизатора Core.

Топология задается следующим образом:

set s1 [$ns node] //Задание первого узла-источника

set s2 [$ns node] //Задание второго узла источника

set e1 [$ns node] //Задание первого пограничного маршрутизатора

set core [$ns node] //Задание центрального маршрутизатора

set e2 [$ns node] //Задание второго пограничного маршрутизатора

set dest [$ns node] //Задание узла-получателя

Далее необходимо задать связь между узлами сети:

$ns duplex-link $s1 $e1 10Mb 5ms DropTail

$ns duplex-link $s2 $e1 10Mb 5ms DropTail

$ns simplex-link $e1 $core 10Mb 5ms dsRED/edge

$ns simplex-link $core $e1 10Mb 5ms dsRED/core

$ns simplex-link $core $e2 5Mb 5ms dsRED/core

$ns simplex-link $e2 $core 5Mb 5ms dsRED/edge

$ns duplex-link $e2 $dest 10Mb 5ms DropTail

Таким образом мы задаем дуплексный канал между узлами s1 и e1, s2 и e1, а также между узлами dest и e2, с использованием алгоритма DropTail

на этих участках. Между маршрутизаторами e1 и core, а также между e2 и core мы устанавливаем 2 симплексных канала с использованием алгоритма RED.

Далее необходимо задать ориентацию для визуализатора NAM, чтобы он не изменил топологию нашей сети:

```
$ns duplex-link-op $s1 $e1 orient down-right

$ns duplex-link-op $s2 $e1 orient up-right

$ns duplex-link-op $e1 $core orient right

$ns duplex-link-op $core $e2 orient right

$ns duplex-link-op $e2 $dest orient right
```

Данные строки показывают, что сеть располагается в одну линию.

В заданной сети трафик будет идти от первого источника к получателю и от второго источника к получателю, исходя из этого задаются буферы виртуальных очередей:

```
set qE1C [[$ns link $e1 $core] queue]

set qE2C [[$ns link $e2 $core] queue]

set qCE1 [[$ns link $core $e1] queue]

set qCE2 [[$ns link $core $e2] queue]
```

Далее зададим параметры алгоритма RED для маршрутизаторов:

```
$qE1C meanPktSize $packetSize

$qE1C set numQueues_ 1

$qE1C setNumPrec 2

$qE1C addPolicyEntry [$s1 id] [$dest id] TokenBucket 20 $cir0 $cbs0

$qE1C addPolicyEntry [$s2 id] [$dest id] TokenBucket 10 $cir1 $cbs1

$qE1C addPolicerEntry TokenBucket 10 11

$qE1C addPolicerEntry TokenBucket 20 21

$qE1C addPHBEntry 10 0 0

$qE1C addPHBEntry 11 0 1

$qE1C addPHBEntry 20 0 0

$qE1C addPHBEntry 21 0 1
```

$qE1C configQ 0 0 20 40 0.02

$qE1C configQ 0 1 10 20 0.10

В последних двух строчках задаются параметры виртуальных очередей: $qE1C configQ 0 0 20 40 0.02 – Здесь мы указываем что если количество пакетов в очереди будет от 20 до 40 то вероятность отбрасывания будет равна 0.02. Соответственно в строке $qE1C configQ 0 1 10 20 0.10

Говорится о том, что если число пакетов в очереди будет от 10 до 20 то вероятность отбрасывания пакета будет равна 0.1.

Необходимо настроить траффик в нашей сети. Для этого создадим 2 UDP-агента для каждого источника, а после этого прикрепим к ним CBR траффик и экспоненциальный трафик.

set udp0 [new Agent/UDP] //Создание агента UDP

$ns attach-agent $s1 $udp0 //Прикрепление агента UDP к узлу s1

Создание источника CBR-трафика и присоединение его к агенту udp0:

set cbr0 [new Application/Traffic/CBR]

$cbr0 attach-agent $udp0

$cbr0 set packet_size_ $packetSize

$udp0 set packetSize_ $packetSize

$cbr0 set rate_ $rate0

$udp0 set fid_ 1

set null0 [new Agent/Null]

$ns attach-agent $dest $null0 //Присоединение трафика к агенту

$ns connect $udp0 $null0

Второй агент создается аналогичным образом:

set udp1 [new Agent/UDP]

$ns attach-agent $s2 $udp1

Далее создается источник экспоненциального трафика:

set ex2 [new Application/Traffic/Exponential]

$ex2 attach-agent $udp1

$ex2 set packetSize_ $packetSize

$udp1 set packetSize_ $packetSize

$ex2 set burst_time_ 500ms

$ex2 set idle_time_ 500ms

$ex2 set rate_ $rate1

$udp1 set fid_ 2

set null1 [new Agent/Null]

$ns attach-agent $dest $null1 //Присоединение трафика к агенту

$ns connect $udp1 $null1

Таким образом мы настроили трафик в сети. Имеется два потока: CBR и экспоненциальный, первый идет от источника s1 к получателю dest, второй от источника s2 к получателю dest. Оба потока обрабатываются с помощью протокола UDP.

4. Задание

1. Запустить файл Lab_PRI.tcl в среде ns-2 со схемой, аналогичной п.3, задать источники трафика соответственно своему варианту.

2. Реализовать визуализацию полученной схемы в NAM.

3. Получить графики изменения задержки и джиттера во времени с помощью программы Trace graph.

4. Повторить пункты 1-3 с файлами Lab_RR.tcl, Lab_WRR.tcl, Lab WIRR.tcl.

5. Оформить отчет.

Варианты выполнения лабораторной работы

Для Приоритетного обслуживания

1	2	3	4	5	6	7	8	9
CBR	Exp	Exp	CBR	CBR	Парето	Парето	Парето	Exp
Exp	CBR	Exp	CBR	Парето	Exp	Парето	CBR	Парето

Для алгоритма RR

1	2	3	4	5	6	7	8	9

Exp	Парето	Парето	Парето	CBR	CBR	Exp	Exp	CBR
Парето	CBR	Парето	Exp	Парето	CBR	Exp	CBR	Exp

Для алгоритма WRR

1	2	3	4	5	6	7	8	9
Exp	CBR	CBR	Парето	Парето	Exp	Exp	CBR	Парето
CBR	Exp	CBR	Парето	CBR	Парето	Exp	Парето	CBR

Для алгоритма WiRR

1	2	3	4	5	6	7	8	9
Парето	CBR	CBR	Exp	Парето	Exp	Парето	CBR	Exp
Exp	Парето	CBR	Парето	CBR	Exp	Парето	Exp	CBR

- CBR (Constant Bit Rate) – Источник траффика с постоянной скоростью
- Exp – Источник траффика с распределением по экспоненциальному закону
- Парето – Источник траффика с распределением по закону Парето

5. Содержание отчета

1. Цель работы

2. Рисунок исследуемой схемы

3. Часть кода, содержащая в себе задание источников трафика

4. Графики задержки и джиттера для каждого алгоритма обслуживания, полученные в программе Trace graph.

5. Ответы на контрольные вопросы.

6. Вывод по проделанной работе.

6. Контрольные вопросы

1. Параметры сетевых соединений.

2. Классификация трафика по относительной предсказуемости скорости передачи данных.

3. Классификация трафика по чувствительности к задержкам пакетов.

4. Классификация трафика по чувствительности к потерям и искажениям пакетов.

5. Классы приложений.

6. Модель M/M/1.

7. Очередь FIFO. Алгоритм Round Robin.

8. Приоритетное обслуживание очередей.

9. Взвешенное обслуживание очередей. Взвешенное справедливое обслуживание очередей.

10. Поясните реализацию топологии исследуемой сети в ns2.

11. Поясните прикреплению агентов с различными типами трафика к определенным узлам сети.