Московский технический университет связи и информатики

На правах рукописи

Масленников Андрей Геннадьевич

РАЗРАБОТКА МЕТОДА ОБРАБОТКИ ТРАФИКА В ОЧЕРЕДЯХ МАРШРУТИЗАТОРОВ МУЛЬТИСЕРВИСНОЙ СЕТИ НА ОСНОВЕ НЕЧЁТКОЙ ЛОГИКИ

Специальность 05.12.13 — Системы, сети и устройства телекоммуникаций

Диссертация на соискание учёной степени
кандидата технических наук

Научный руководитель:
кандидат технических наук
Деарт В.Ю.

Москва – 2015

Оглавление

Стр.

Введение

Актуальность темы. Развитие технологий широкополосного доступа и рост популярности различных Интернет-услуг, таких как, Интернет-телевидение, OTT (Over–The–Top — передача видео через публичную сеть Интернет), видеоконференции, передача данных через файл-обменные сети P2P (peer–to–peer — одноранговые сети), доступ к социальным сетям, облачным хранилищам данных и облачным приложениям приводит к росту потребностей к доступной полосе пропускания и соответственно к постоянному наращиванию скоростей каналов передачи данных. Из–за всплесков трафика и несовершенства методов управления трафиком на сети возникают перегрузки, которые приводят к переполнению очередей маршрутизаторов и ухудшению качества обслуживания.

Отчёты о скорости подключения абонентов и наблюдение за количеством пользователей сети Интернет говорят о постоянном росте скоростей и увеличении аудитории Интернета. Так, компания «Akamai» опубликовала отчёт, согласно которому средняя скорость фиксированного подключения пользователей к сети Интернет на начало 2014 года в мире составила 3,9 Мбит/с и за год выросла на 24%. В России средняя скорость доступа к сети выросла до уровня 8.6 Мбит/с и за год рост составил 44%. Из года в год темпы роста скорости доступа продолжают расти. Согласно результатам ежегодного исследования Cisco Visual Networking Index (VNI) Global IP Traffic Forecast 2013–2018, к 2018 г. количество пользователей Интернета увеличится до 4 млрд (51% населения Земли) с 2,5 млрд в 2013 г., количество сетевых устройств вырастет до 21 млрд по сравнению с 12 млрд в 2013 г. Объём передаваемого мирового IP-трафика за последние 5 лет увеличился пятикратно, и ещё утроится в следующие 5 лет.

Таким образом, постоянный рост скоростей доступа и количества пользователей приведёт к неравномерности пропускных способностей каналов передачи данных и скачкам интенсивности трафика. Отсюда, неизбежно возникновение перегрузок на отдельных участках сети. Часто суммарная скорость подключения абонентов на узле доступа, при предоставлении услуг без гарантии качества, превосходит доступную скорость подключения к магистральной сети. Также возможно возникновение перегрузок при авариях на других участках

сети и перераспределении трафика или при наступлении событий вне сети, но связанных с внезапным всплеском объёма передаваемого трафика.

Изначально в маршрутизаторах пакеты данных, которые не могут быть переданы сразу, помещались в промежуточный буфер при переполнении которого, новые поступающие пакеты отбрасывались. Переполнение буфера происходит каждый раз при наступлении перегрузки в сети. Такой пассивный режим работы буфера называют Tail Drop (отбрасывание конца очереди). Для предотвращения перегрузок в очередях маршрутизатора используются активные методы управления на основе метода RED (Random Early Detection — раннее случайное обнаружение). Данный метод контролирует только усреднённое значение текущей длины очереди и изменяет вероятность сброса пакетов по линейному закону при нахождение усреднённой длины очереди между установленными верхним и нижним порогами.

Среди работ зарубежных учёных, исследующих проблемы обработки трафика в очередях и оценки параметров качества обслуживания, следует отметить работы следующих авторов: Jacobson V. [1, 2], Floyd S. [3], Hollot C., Misra V. [4], Towsley D. [5,6], Feng W. [7,8], Athuraliya S. [9], Kunniyurs S. [10,11], Fengyuan R. [12], Chrysostomou C. [13]. В России этой проблеме посвящены работы Гайдамаки Ю.В. [14, 15], Ефимушкина В.А. [16], Кучерявого Е.А. [17], Печинкина В.А. [18], Рослякова А.В. [19, 20], Самуйлова К.Е. [21], Степанова С.Н. [22–24], Цитовича И.И. [25, 26], Яновского Г.Г. [27–29] и др.

Метод отбрасывания конца очереди Tail Drop, а также метод случайного раннего обнаружения RED не всегда справляются с управлением трафиком со сложной динамикой, высокой пачечностью и нелинейностью изменения нагрузки, что приводит к возникновению перегрузок, явлению «глобальной синхронизации» TCP потоков, когда все TCP-источники при переполнении буфера теряют пакеты и одновременно снижают нагрузку, а затем опять одновременно её повышают. В результате чего, моменты перегрузки сменяются моментами простоя, что ведёт к неэффективному использованию ресурсов и снижению качества обслуживания. При передаче смешанного трафика TCP одновременно с UDP, синхронизация TCP–соединений и перегрузки также приводят к деградации сопутствующего UDP–трафика. Эти явления ухудшают такие параметры качества обслуживания трафика, как эффективная скорость передачи данных, процент потерянных пакетов, задержки и вариации задержек.

В системах автоматического управления процессами со сложной нелинейной динамикой, в робототехнике нашли широкое применение регуляторы на основе нечёткой логики FLC (Fuzzy Logic Controller). Такие регуляторы широко применяются когда, описание поведения системы с помощью точных математических методов представляется достаточно сложным, но доступно простое качественное описание поведения системы. Поэтому задача разработки регулятора на основе нечёткой логики FLC для применения в качестве активного метода управления очередями в маршрутизаторе является актуальной.

Результаты исследования соответствуют следующим пунктам паспорта научной специальности 05.12.13 — Системы, сети и устройства телекоммуникаций:

- пункту 4 — Исследование путей совершенствования управления информационными потоками;
- пункту 14 — Разработка методов исследования, моделирования и проектирования сетей, систем и устройств телекоммуникаций.

Объектом исследования является механизм обработки трафика в очереди маршрутизатора доступа для предотвращения перегрузки в мультисервисной сети.

Предметом исследования являются зависимости параметров качества обслуживания от механизма обработки трафика в очередях маршрутизаторов.

Цель работы и задачи исследования. Целью работы является разработка нового механизма обработки трафика в очередях маршрутизаторов на основе регулятора с нечёткой логикой FLC для предотвращения перегрузок, контролирования средней задержки в очереди и улучшения параметров качества обслуживания при передачи данных по пакетной сети на основе протоколов TCP/IP.

Для достижения поставленной цели решены следующие задачи:

1) проведено исследование существующих методов активного управления очередями маршрутизаторов для предотвращения перегрузок;
2) разработан нечёткий регулятор эффективно регулирующий заполнение очереди в зависимости от интенсивности нагрузки и заполненности буфера;
3) построена имитационная модель процесса обслуживания пакетов в очереди маршрутизатора и проведено сравнение параметров качества обслу-

живания трафика при использовании нечёткого регулятора с параметрами обслуживания, обеспечиваемыми другими методами управления в режиме перегрузки в сети;

4) построена математическая модель нечёткого регулятора трафика и рассчитаны вероятностно-временные характеристики модели;
5) проведено сравнение расчётных данных, полученных на математической модели, с данными имитационного моделирования, что позволило удостовериться в адекватности полученных результатов;
6) разработанный метод управления на базе регулятора с нечёткой логикой реализован в виде программного модуля для Linux–маршрутизатора, что позволило убедиться в его эффективности на практике.

Научная новизна:

1) Разработан новый метод обработки трафика в очередях маршрутизаторов на основе нечёткой логики с контролем уровня текущей длины очереди, контролем интенсивности нагрузки и аддитивным приращением вероятности сброса пакетов при сложной нелинейной динамике трафика, который позволяет удерживать длину очереди около заданного эталонного значения в режиме перегрузки маршрутизатора.
2) Разработана имитационная модель процесса обслуживания пакетов в очереди маршрутизатора с нечётким регулятором (FLC) в сетевом симуляторе NS–2. Имитационная модель позволила сравнить эффективность методов управления очередями и параметры качества в условиях перегрузки в мультисервисной сети.
3) Разработана математическая модель процесса обслуживания пакетов в очереди при использовании регулятора с нечёткой логикой на основе гистерезисного управления с порогами. Модель позволяет рассчитывать вероятностно–временные характеристики системы в зависимости от интенсивности поступающей нагрузки, путём численного решения системы уравнений равновесия (СУР), и тем самым оценивать среднюю задержку пакетов в очереди.

Научная и практическая значимость. Построенная математическая модель обработки трафика в очередях маршрутизатора с помощью нечёткой логики позволяет рассчитывать вероятностно–временные характеристики подобных систем, а имитационная модель оценивать влияние проектируемой си-

стемы на параметры качества обслуживания. Реализация разработанного метода обработки трафика в очередях маршрутизатора в виде программного модуля для Linux–маршрутизаторов открывает возможности внедрения нового метода в сетях передачи данных. Применение метода активного управления очередями FLC в пограничных маршрутизаторах на уровне доступа позволяет прогнозировать задержку в очереди, улучшить параметры качества обслуживания эластичного трафика в условиях перегрузки при недостаточной скорости канала передачи данных или при взрывном росте трафика, что подтверждается актом использования в учебном процессе (Приложение Б.1) и актом использования в тестировании (Приложение Б.2).

Методы исследования. Для решения поставленных задач используются методы теории вероятностей, математической статистики, теории массового обслуживания и имитационного моделирования.

Степень достоверности полученных результатов обеспечивается расчётами с использованием методов математической статистики и теории вероятности, сравнением аналитических результатов с данными, полученными при имитационном моделировании и измерениями, полученными на лабораторном фрагменте сети передачи данных с использованием Linux–маршрутизатора.

Апробация работы. Основные результаты работы докладывались и обсуждались на следующих 7 конференциях:

- 15-й международной конференции Conference of Open Innovations Association FRUCT (Finnish–Russian University of Cooperation in Telecommunications), (С.–Петербург), 2014г.;
- международной конференции DCCN–2013 «Распределённые компьютерные и коммуникационные сети: управление, вычисление, связь» (Институт проблем управления им. В. А. Трапезникова РАН, Москва);
- всероссийской конференции с международным участием «Информационно–телекоммуникационные технологии и математическое моделирование высокотехнологичных систем» (РУДН, Москва) в 2011, 2012 и 2013 годах;
- 6-й отраслевой научной конференциии «Технологии информационного общества» (МТУСИ, Москва), 2012 г.;

– 11-й международной конференции Conference of Open Innovations Association FRUCT (Finnish–Russian University of Cooperation in Telecommunications), (С.–Петербург), 2012г.;
– конференции «Телекоммуникационные и вычислительные системы», «Международный форум информатизации», МФИ–2011 (МТУСИ, Москва);
– XIX международной научно-технической конференции «Информационные Средства и Технологии», ИСТ–2011 (МЭИ, Москва).

Личный вклад. Все основные научные положения и выводы, составляющие содержание диссертации, разработаны соискателем самостоятельно.

Публикации. Основные результаты по теме диссертации опубликованы в 11 печатных изданиях [30–40], 2 из которых напечатаны в ведущих рецензируемых научных журналах и изданиях, внесённых в перечень утверждённый ВАК [30, 31].

Основные положения, выносимые на защиту:

1) Построена имитационная модель мультисервисной сети, позволяющая сравнить параметры качества различных методов обработки трафика в очередях маршрутизаторов, в момент перегрузки и при сложной нелинейной динамике трафика.
2) Разработанный метод обработки трафика в очереди на основе регулятора с нечёткой логикой (FLC) предотвращает перегрузку и обеспечивает стабилизацию длины очереди около заданного значения.
3) Построена математическая модель процесса обработки трафика в очереди в маршрутизаторе на базе регулятора с нечёткой логикой с помощью модели с гистерезисным управлением с порогами. Получены выражения для вероятностно-временных характеристик.
4) При использовании метода обслуживания очереди на основе регулятора с нечёткой логикой параметры качества передачи данных превосходят характеристики, получаемые при использовании традиционных методов Tail Drop и RED.

Объем и структура работы. Диссертация состоит из введения, четырёх глав, заключения, списка сокращений, списка литературы и двух приложений. Полный объем диссертации составляет 124 страницы машинописного текста,

содержащих 45 рисунка и 21 таблиц. Список использованных источников содержит 102 наименования.

Глава 1. Анализ методов активного управления очередями маршрутизатора

1.1 Современные технологии Интернет–услуг

Рост популярности технологий передачи видео, каких как видео по требованию VoD (Video–on–Demand), передача видео–программ через Интернет OTT (Over-the-Top), накладывает повышенные требования к характеристикам канала передачи данных, таким как гарантированная скорость передачи, потери пакетов, задержки передачи пакетов, а в случае интерактивных услуг видео–конференций и IP–телефонии [20, 41] (VoIP, Video–Chat) возникают повышенные требования к вариации задержек (джиттеру). В тоже время, в мультисервисной сети присутствует также трафик другого рода, например передача файлов P2P, просмотр Web и трафик создаваемый автоматическими устройствами M2M, IoT [19, 42], который менее требователен к скорости передачи данных и задержкам [43]. Различные приложения чувствительны к разным сетевым характеристикам. Так например, для голосового трафика доступная полоса пропускания является не критичной, а значение джиттера наоборот является очень важным [29]. Для передачи файлов, больше важна скорость передачи, чем задержки и джиттер [28].

Трафик сети Интернет в основном использует транспортные протоколы TCP и UDP, с преобладанием TCP протокола. Протокол TCP, в отличие от UDP протокола, предполагает установление соединения и подтверждение полученных пакетов. Такие сервисы как OTT, поиск и просмотр Web страниц, передача файлов HTTP/FTP используют в качестве транспорта TCP. Передача аудио и видео данных в IP–телефонии может быть с использованием как TCP, так и UDP протокола.

Для обеспечения необходимого качества обслуживания QoS (Quality of Service) [44] для каждого типа сетевого сервиса в мультисервисных маршрутизаторах используется ряд технологий управления трафиком [45].

1.2 Управление трафиком в маршрутизаторе

Доступ к сетевым услугам может быть организован по принципу «наилучшая попытка» (Best Effort), т.е. когда трафик обрабатывается с параметрами максимально возможными в данный момент времени, но без каких либо гарантий по обеспечению качества. Для обеспечения гарантированного качества необходимо управлять интенсивностью нагрузки в зависимости от предъявляемых требований.

Процесс управления интенсивностью нагрузки в маршрутизаторе включает в себя следующие действия [17]:

- мониторинг;
- маркировка;
- сглаживание;
- сброс.

Прежде всего маршрутизатор должен классифицировать приходящие пакеты, после чего становится возможным определение нагрузки каждого потока и соответствия текущих значения параметров заявленным. Далее происходит мониторинг нагрузки, и если значения параметров трафика превышают заданные в начальный момент, то маршрутизатор предпринимает действия по ограничению нагрузки (policing).

Оптимизация процесса передачи трафика ставит некоторые задачи на каждом этапе. Управление допуском нагрузки в сеть базируется не только на пропускных возможностях каналов, но и на ресурсах маршрутизаторов, или буферных пространствах. Большое количество мультиплексированных потоков, конкурирующих между собой, требуют реализации механизмов, позволяющих регулировать количество этих потоков в зависимости от требуемых параметров качества обслуживания и доступных сетевых ресурсов.

Современное развитие пакетных сетей передачи данных подняло технологию Ethernet на новый уровень позволяющий оказывать мультисервисные услуги с надёжностью и качеством сетей SDH, и механизмами приоритезации и разделения услуг доступными в недалёком прошлом лишь сетям ATM/FR [46, 47]. Технология Ethernet уже прочно заняла центральное место в сетях ведущих компаний провайдеров телекоммуникационных услуг благодаря совместимости

с различными средами передачи, низкой цене за Мегабит, простоте и гибкости. Бизнес–пользователи требуют оказания услуг с гарантированным уровнем обслуживания непосредственно в помещении офиса. Встаёт задача доставить трафик с гарантированным качеством до конечного абонента. Простые медиаконверторы работающие на первом уровне сетевой модели взаимодействия открытых систем OSI (Layer 1) [48, 49] не могут решить задачу обработки разнородного трафика с заданными приоритетами и контролем качества. Для этой задачи подходят интеллектуальные устройства демаркации (разделения ответственности) являющиеся, по сути, коммутаторами второго уровня с возможность обработки заголовков верхних уровней (Layer 2+) или маршрутизаторы уровня 3 (Layer 3) Рис. 1.1.

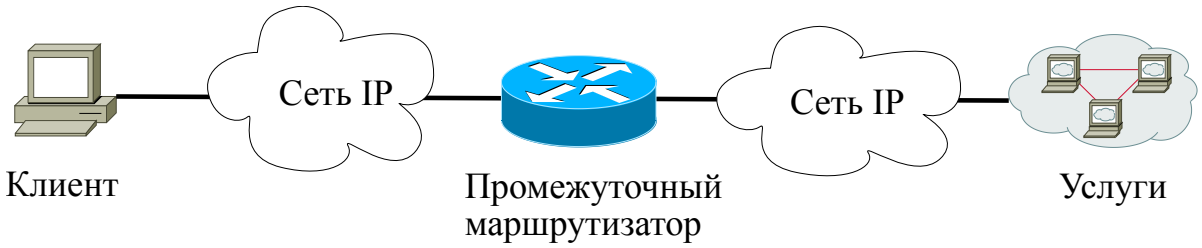


Рисунок 1.1 — Схема подключения промежуточного маршрутизатора

Новые устройства обеспечивают передачу нескольких сервисов через единый порт передачи данных. Каждая услуга (например, VoIP, важные данные или доступ в интернет) получит гарантированную полосу пропускания и соответствующий приоритет при обработке. Это также позволит освободить порты для других подключений и увеличит концентрацию сервисов на точке присутствия оператора.

1.2.1 Классификация нагрузки

Каждый сервис на входном порту классифицируется по некоторым признакам [50]:

- метка пользователя на втором уровне VLAN ID [51];
- приоритет пользователя на втором уровне 802.1p [52];
- код услуги заголовка третьего уровня DSCP/TOS [53];
- адрес отправителя или получателя;

– номер физического порта;

и назначается в отдельный поток. Каждому потоку присваивается класс обслуживания CoS и индивидуальный профиль полосы пропускания (Bandwidth profile). Например, поток с данными VoIP должен иметь небольшую, но гарантированную полосу пропускания CIR = const и обрабатываться с наивысшим приоритетом для предотвращения задержек в передаче пакетов. Поток, предназначенный для доступа в интернет, может совсем не иметь гарантированной полосы пропускания CIR = 0 (Best Effort data) и обрабатываться в последнюю очередь.

1.2.2 Управление нагрузкой

Для операторов появляется возможность увеличить прибыль от передачи данных за счёт перепродажи незадействованной (излишней) полосы пропускания. Вероятность одновременного занятия всей доступной полосы пропускания канала сразу несколькими абонентами или сервисами невелика. Имея канал, например, 100 Мбит/с можно четрыем абонентам продать гарантированную полосу по 25 Мбита/с и еще по 25 Мбита/с предложить использовать каждому абоненту дополнительно в случае отсутствия полного занятия канала. То есть физический канал 100 Мбит/с может быть продан как 200 Мбит/с. Данный способ подключения клиентов называют «переподпиской» (oversubscription). Параметры трафика пользователя измеряются на соответствие заявленным характеристикам (гарантированной скорости подключения), пакеты удовлетворяющие заявленному уровню обслуживания считаются конформными и должны быть гарантированно переданы в сеть. Дополнительный трафик пользователя, который передаётся сверх заявленного уровня, считается неконформным и может быть передан только в случае наличия свободных ресурсов, в противном случае, пакеты данных передающиеся сверх лимита будут сброшены на устройстве разграничения сети между поставщиком услуги доступа и абонентом.

Для контроля и измерения поступающего трафика используют алгоритм «корзина с жетонами» (Token Bucket) показанный на рис. 1.2 [17, 54]:

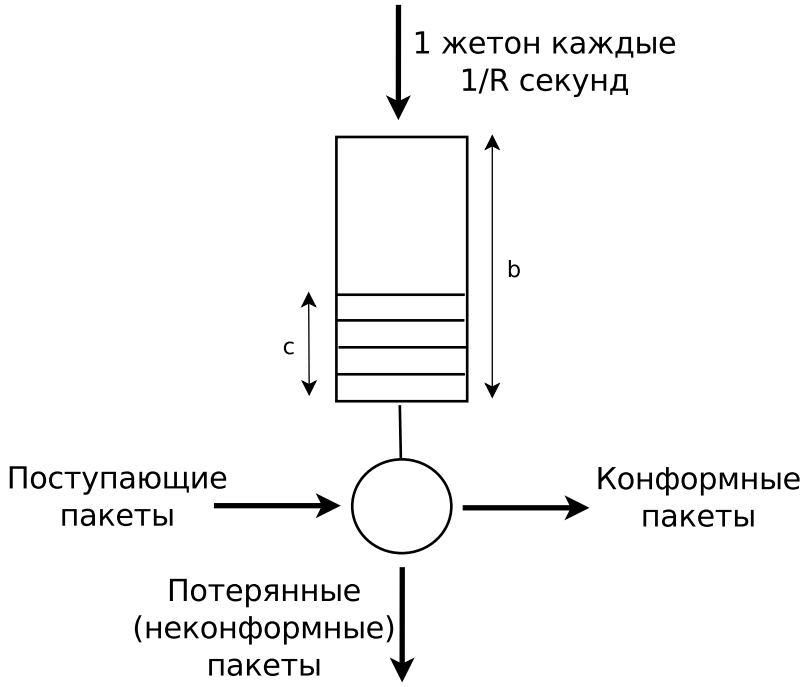


Рисунок 1.2 — Алгоритм Token Bucket

На рис. 1.2 R обозначает среднюю скорость обслуживания, b — размер корзины с жетонами, а C — количество жетонов в корзине в некоторый момент времени.

Жетоны автоматически генерируются системой с заданной интенсивностью R и помещаются в «корзину», если корзина переполнится, то жетоны теряются. При поступлении нового пакета, если в корзине имеются в наличии жетоны, то пакет передаётся дальше без задержек (конформные пакеты), а один жетон изымается из корзины. Если в корзине не оказывается жетона в момент поступления пакета, то этот пакет сбрасывается (неконформные пакеты) или помещается в очередь до генерации нового жетона в корзине. Такой алгоритм, при полной корзине с жетонами, даёт возможность передать пачку пакетов равной размеру корзины b с максимальной интенсивностью, а последующие пакеты будут передаваться с интенсивностью равной интенсивности генерации жетонов.

На практике, для обеспечения гарантированной скорости передачи данных, используется модифицированный алгоритм Token Bucket состоящий из двух корзин с жетонами, который называется «Двухскоростной трёхцветный маркировщик» (Two Rate Three Color Marker – TrTCM, RFC-2698) [55]. Каждому отдельному потоку передачи данных может быть обеспечена гарантированная скорость передачи (Committed Information Rate — CIR) и пиковая негарантиро-

ванная скорость (Peak Information Rate — PIR). Работа алгоритма TrTCM для варианта, когда пакеты не были предварительно маркированы, условно представлена на рис. 1.3.

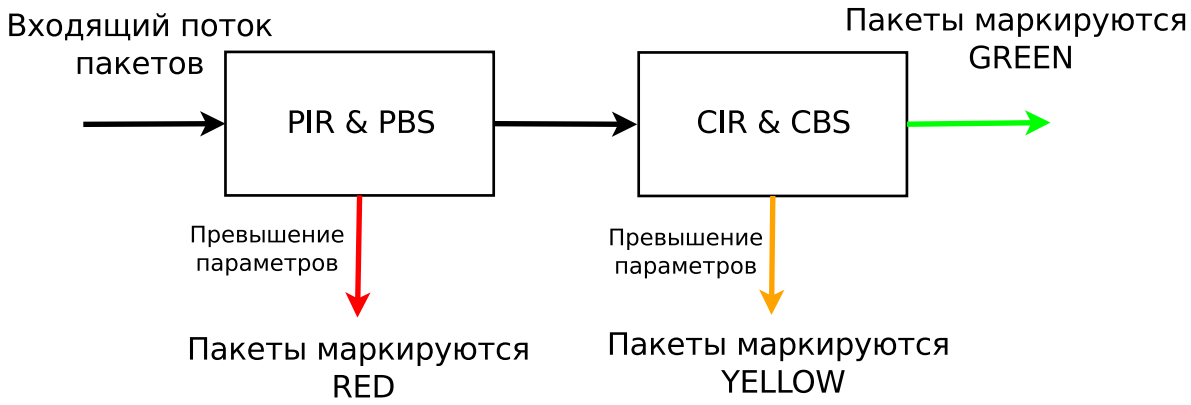


Рисунок 1.3 — Механизм трёхцветного маркировщика trTCM

Входящий поток немаркированных пакетов поступает в первый блок с корзиной с жетонами, где контролируются пиковая скорость PIR и размер пачки (Peak Burst Size — PBS). Пакеты которые превысили допустимые параметры и признаны неконформными маркируются как «красные» (RED), конформные пакеты поступают далее в блок со второй корзиной с жетонами. Во втором блоке уже контролируется гарантированная скорость CIR и гарантированный размер пачки (Committed Burst Size — CBS). Неконформные пакеты помечаются как «жёлтые» (YELLOW), а конформные соответственно как «зелёные» (GREEN). Таким образом, пакеты маркированные как RED, которые превысили пиковые параметры будут сброшены, а пакеты маркированные как GREEN будут всегда передаваться с гарантированной скоростью CIR и размером пачки CBS. При недостаточной загрузки общего канала передачи данных другими клиентами, могут быть также дополнительно переданы пакеты от данного клиента маркированные как YELLOW, но с параметрами не превышающими пиковых значений PIR/PBS.

1.2.3 Планирование обслуживания пакетов

Поток пакетов допущенный для передачи в сеть далее попадает в очередь, соответствующую приоритету данного потока. Причём, приоритет очереди мо-

жет устанавливаться строгим (Strict), то есть трафик из этой очереди передается сразу, и в это время остальные очереди с меньшим приоритетом ожидают окончания этой передачи. Для того чтобы передача в низших очередях не останавливалась, используют взвешенные типы очередей (WFQ). В этом случае трафик из очередей передается в зависимости от своего весового коэффициента. На последнем этапе обработки исходящий трафик соответственно маркируется для передачи по сети провайдера, например, добавляется метки SP-VLAN и биты приоритетов (P.bit).

Существует альянс организаций Metro Ethernet Forum (MEF) [56], который занимается разработкой технических спецификаций для предоставления сервисов в Ethernet сетях, а также проводит тестирование и сертификацию оборудования разных производителей. MEF определил следующие стандартные сервисы: Ethernet Private Line (EPL), Ethernet Virtual Private Line (EVPL), Ethernet Private LAN (EPLAN) и Ethernet Virtual Private LAN (EVPLAN). Сервис EPL обеспечивает подключение точка-точка между двумя офисами абонента через Ethernet сеть провайдера. EVPL – то же самое, но по схеме точка-многоточка с использованием в центре единственного порта Ethernet для подключения центрального офиса к филиалам (мультиплексирование сервисов). Сервис EPLAN позволяет подключить в единую сеть LAN несколько удаленных офисов, а EVPLAN тоже самое, но с использованием единственного порта для нескольких подключений.

Обмен служебной информацией между устройствами демаркации также стандартизирован и дает возможность управлять уровнем обслуживания из конца в конец соединения и гарантировать качество предоставляемых сервисов [57]. Поддерживается удаленная диагностика и установка проверочных петель, благодаря чему легко локализовать и идентифицировать проблемы в сети, и устранить неисправности до того, как сам абонент заметит их.

Существует несколько стандартных механизмов реализации процессов эксплуатации и техобслуживания OAM (Operation, Administration and Maintenance) Табл.1.1 [47]:

Устройства демаркации поддерживают мониторинг ключевых параметров качества сервисов (таких как процент потерянных пакетов, задержки, вариация задержек, производительность и доступность сети). При превышении одним из параметров заданного порога, генерируется соответствующее служебное

Таблица 1.1 — Механизмы OAM

Стандарт	Область мониторинга	Контроль производительности	Тип	Передача сигналов аварии
IEEE 802.3ah (Clause 57)	Один сегмент	Нет	Подключение	Нет
IEEE 802.1ag	Из конца в конец	Нет	Соединение	Нет
ITU-T Y.1731	Из конца в конец	Есть	Сервис	Есть

сообщение, а также периодически отправляются отчёты об актуальном состоянии качества обслуживания. С абонентом может быть заключен договор, в котором в зависимости от выбранного тарифного плана, прописываются уровни обслуживания (SLA) [16]. Новые пакетные сети передачи данных Ethernet возвращаются к прежним, опробованным технологиям, но уже на следующем новом витке развития. Сети Ethernet обеспечивают надёжность «пять девяток» как и сети SDH, а также приобрели черты сетей ATM/FR [27,46], обеспечивают передачу мультисервисного трафика с классификацией, приоритезацией, контролем полосы пропускания, механизмами резервирования и передачи сервисной информации OAM. При помощи интеллектуальных устройств демаркации стало возможным качественно передавать также и синхронный трафик TDM через пакетные сети.

1.2.4 Управление очередями

Функционирование алгоритма управления очередями оценивается как его способность эффективно контролировать трафик во время периодов перегрузки. Существует несколько алгоритмов управления этим процессом. Они подразделяются на два класса: пассивные и активные (учитывают состояние сети).

Наиболее простой метод — так называемый TailDrop, работающий с очередью с дисциплиной обслуживания FCFS (First Come First Served — пришедший первым обслуживается первым) [58]. Этот метод сбрасывает пакеты при переполнении очереди и помещает новые пакеты в очередь только при освобождении

в ней места. Из недостатков этого алгоритма можно отметить два. Во-первых, невозможность реализации принципа «справедливого распределения ресурсов». Во-вторых, невозможность заблаговременного определения момента перегрузки. Были предприняты попытки усовершенствовать механизм TailDrop (алгоритмы «вероятностный сброс пакета» и «сброс начала очереди»), чтобы решить проблему занятия очереди пакетами одного или нескольких потоков, однако осталась проблема заблаговременного обнаружения перегрузки очереди.

Структура маршрутизатора обеспечивающего классификацию, качество обслуживания и управление трафиком приведена на рис. 1.4 [17].

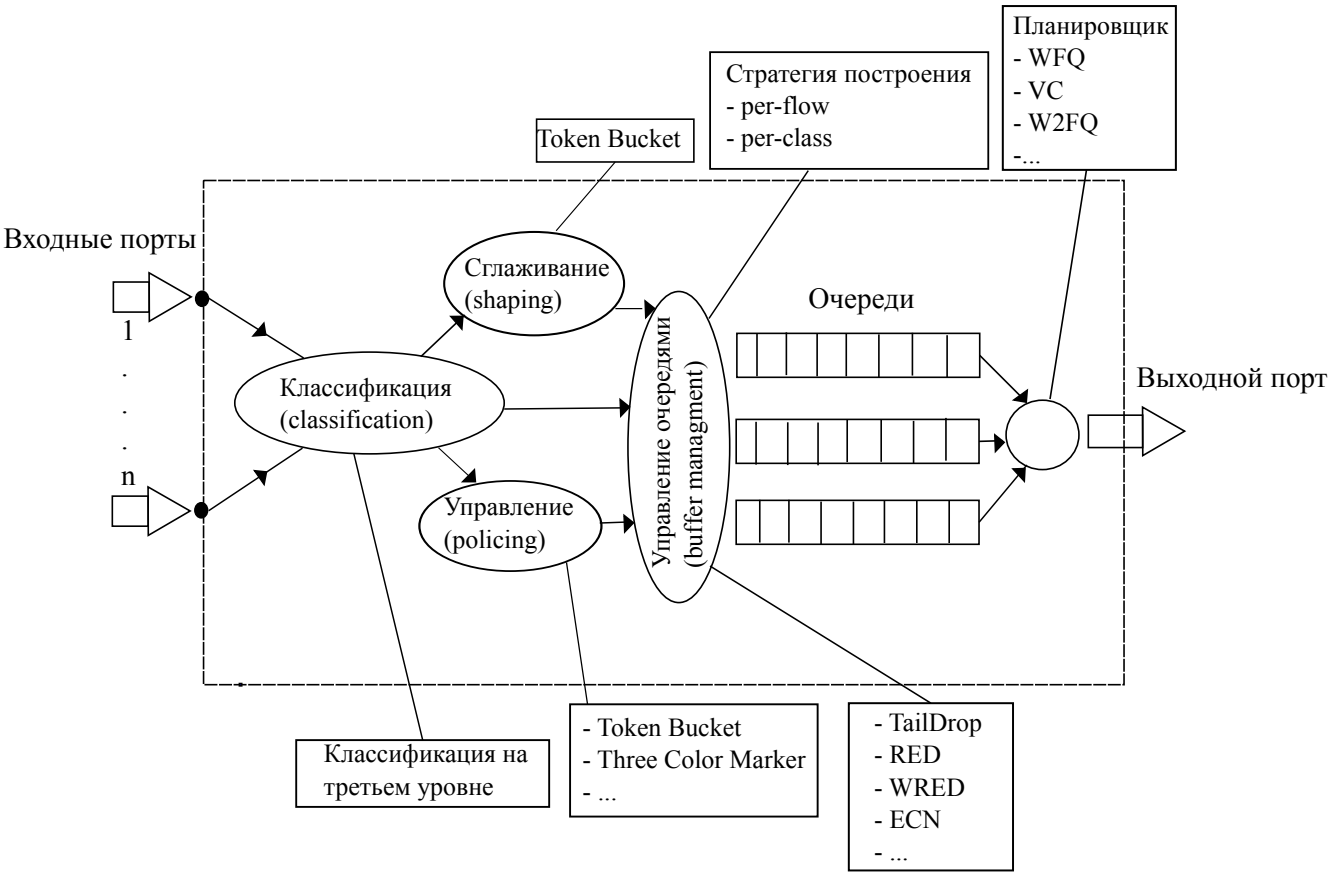


Рисунок 1.4 — Структура QoS-маршрутизатора

Пакеты поступающие на входящие порты маршрутизатора проходят процедуру классификации (Classification) (раздел 1.2.1), как правило на основе анализа заголовков IP-пакета по полю DSCP, либо на основе адреса отправители или получателя. Далее, в соответствии с присвоенным классом, пакеты могут проходить через процедуру сглаживания (Shaping) или управления нагрузкой (Policing) (раздел 1.2.2), которая определяет на основе установленных политик, сколько пакетов может быть допущено для передачи дальше для каждого класса. После применения политик управления, пакеты поступают в очереди,

которые могут быть определены на каждый поток или на каждый класс, и могут иметь иерархическую структуру в соответствии со структурой построения. Каждой очереди может быть присвоен свой метод управления (Buffer management), например TailDrop или RED (раздел 1.3), с помощью которого принимается решение о помещении каждого пакета в очередь или решение об его сбросе, либо маркировке. Методы управления очередью применяются также для упреждающего сброса или маркировки пакетов, чтобы заблаговременно определить момент всплеска нагрузки и не допустить переполнения очереди. Очереди могут иметь, например, строгий приоритет в соответствии с порядком которого планировщик (Scheduler) будет обслуживать пакеты из разных очередей, или планировщик может реализовывать обслуживание очередей в пропорции определяемой «относительным весом» очереди (раздел 1.2.3). Далее на выходном порту маршрутизатора пакеты из разных очередей могут заново перемаркироваться (например, устанавливаться новые коды DSCP в IP-заголовках) для информирования следующего маршрутизатора о существующих классах обслуживания.

1.2.5 Явное уведомление о перегрузке

Во всех методах активного управления перегрузкой предлагается использовать протокол явного уведомления о перегрузке ECN (Explicit Congestion Notification) описанный в документе RFC-3168 [59]. Использование этого протокола позволяет сократить процент потерянных TCP сегментов благодаря тому, что вместо сброса пакета из очереди для сигнализации передатчику о перегрузке, пакет специальным образом маркируется и передается дальше. Это также позволяет сократить время реакции передатчика на перегрузку и управлять передачей более эффективно. Протокол ECN задействует как сетевой уровень (протокол IP) для обработки маршрутизатором, так и транспортный уровень (протокол TCP) для сигнализации TCP-приёмником TCP-передатчику. На рисунках 1.5 и 1.6 приведены используемые протоколом ECN поля в заголовках TCP/IP.

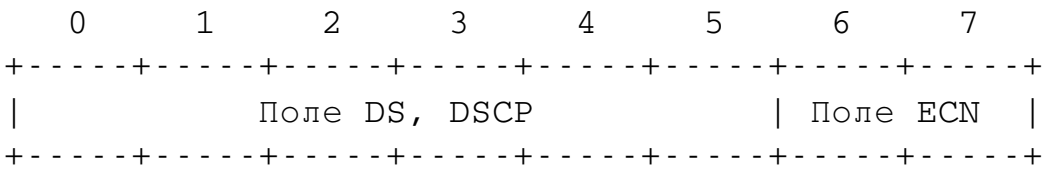


Рисунок 1.5 — Поля Differentiated Services и ECN в заголовке IP

Вверху рисунка пронумерованы положения битов в поле заголовка начиная с нуля. Поле ECN в IP заголовке занимает два бита с номерами 6 и 7, и может передавать значения двух маркеров:

– маркер ECT (ECN-capable transport): значения 01 или 10 — используется источником для информирования приемника о поддержке ECN;
– маркер CE (congestion experienced): значение 11 — используется промежуточным маршрутизатором для информирования приемника о перегрузке.

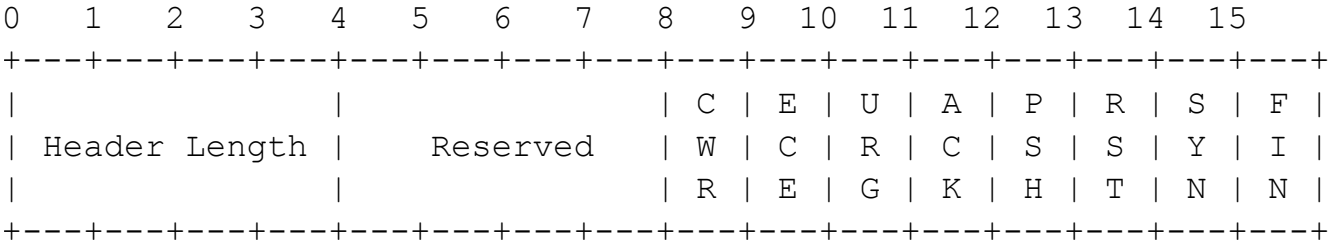


Рисунок 1.6 — Новое определение байтов 13 и 14 в заголовке TCP

В TCP заголовке пакетов протокол ECN использует биты 8 и 9 в качестве флагов для сигнализации между TCP приёмником и передатчиком:

– Congestion Window Reduced (CWR) — флаг подтверждения сокращения TCP–окна;
– ECN-Echo (ECE) — флаг уведомления о перегрузке.

Промежуточный маршрутизатор получив IP пакет с маркером ECT, может в случае перегрузки установить в пакете маркер CE и отправить его дальше, вместо сброса пакета. Приемник, получив IP пакет с маркером CE, уведомляет TCP–передатчик о перегрузке, установив в отправленном обратно TCP сегменте бит ECE, а TCP–приёмник, в свою очередь, получив такой сегмент, снижает размер TCP-окна в 2 раза, таким образом, уменьшая количество передаваемых сегментов. Сокращение окна TCP–передатчик подтверждает установкой бита CWR в TCP заголовке. Схема работы метода показана на рисунке 1.7.

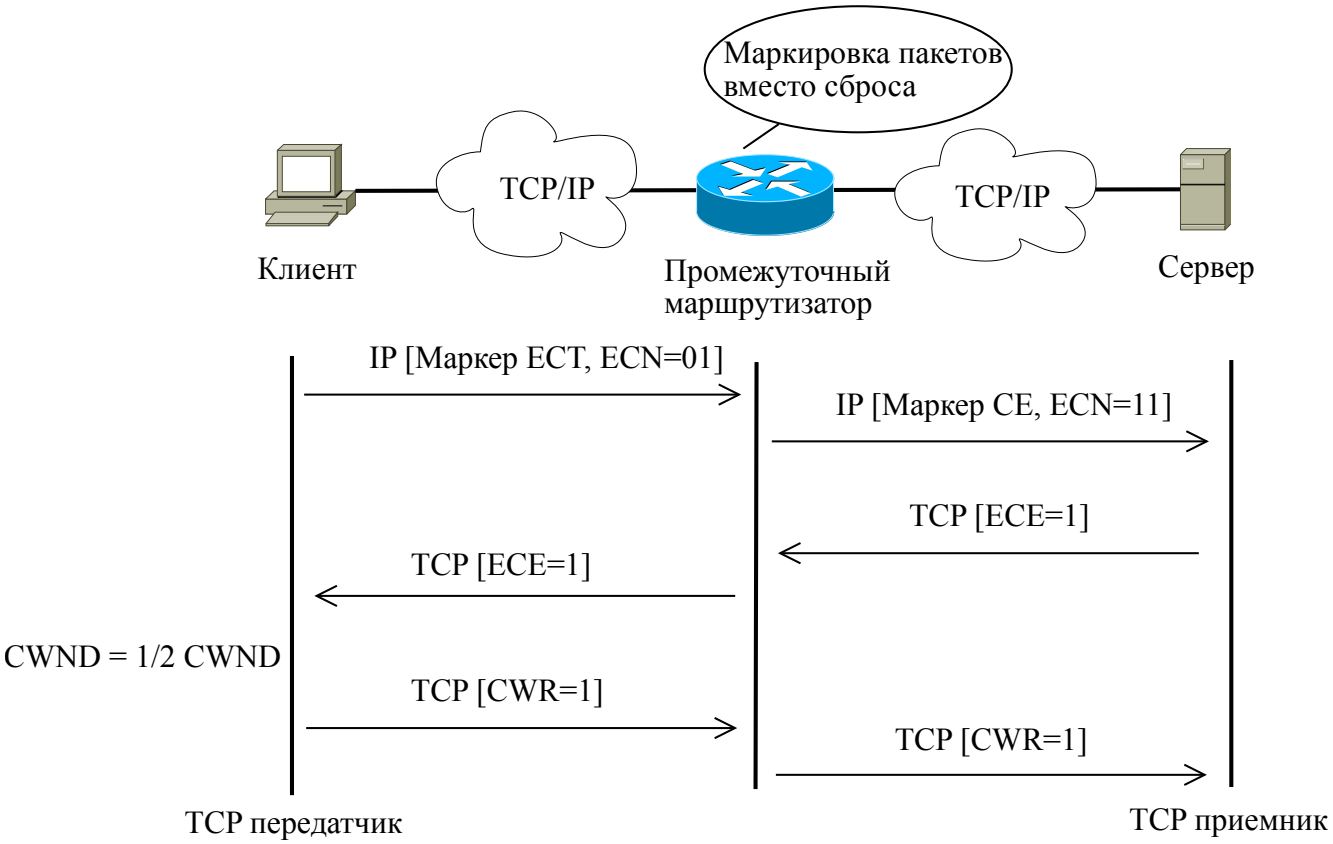


Рисунок 1.7 — Схема работы механизма ECN

1.3 Методы активного управления очередями

1.3.1 Проблемы пассивного управления очередями. Метод TailDrop

Использование метода **TailDrop** для управления TCP трафиком приводит к возникновению явления «глобальной синхронизации», когда при переполнении буфера маршрутизатора одновременно сбрасываются все приходящие пакеты, и все TCP передатчики уменьшают размер TCP–окна и потом синхронно его увеличивают, вызывая новую перегрузку. Это явление также приводит к увеличению сброса пакетов UDP трафика при его одновременном прохождении через общую очередь маршрутизатора [60]. Для противодействия этому явлению был разработан метод случайного раннего обнаружения перегрузок **RED**, сбрасывающий пакеты в вероятностью, линейно увеличивающейся с ростом усреднённой длины очереди. Механизм RED до сих пор является наиболее распространённым в маршрутизаторах, хотя многими исследователями указы-

вались его недостатки [61], так как метод отслеживает только усреднённую длину очереди, что допускает осцилляции мгновенной длины очереди, и изменяет вероятность сброса по линейному закону, что не позволяет эффективно регулировать нелинейную динамику трафика.

1.3.2 Метод RED и Adaptive RED

Метод **RED** [2] оценивает среднее значение очереди путём вычисления экспоненциально взвешенного скользящего среднего a по рекуррентной формуле:

$$a = (1 - w)a + w \cdot q, \tag{1.1}$$

где q — текущее значение очереди, а w — весовой коэффициента с рекомендованным значением 0,002.

Если очередь пуста, то делается оценка среднего значения очереди a в период отсутствия пакетов по следующей формуле:

$$a = a \cdot (1 - m)^m, \tag{1.2}$$

где m — предположительное количество пакетов поступивших в очередь. Без такого предположения оценка a будет неправильная. Если пакет поступает в момент когда очередь пуста, то m рассчитывается по формуле:

$$m = (t - t_0)/s, \tag{1.3}$$

где t — текущее время, t_0 — момент времени, когда очередь стала пустой, а s — средний размер пакета.

Вероятность маркировки/сброса пакетов линейно изменяется в интервале от 0 до max_p по формуле:

$$P_{drop} = \frac{max_p(a - min_{th})}{(max_{th} - min_{th})}, \tag{1.4}$$

где min_{th} — минимальный порог среднего значения очереди до которого не происходит сброса, max_{th} — максимальный порог, после которого сбрасываются все пакеты, а max_p — максимальное значение вероятности сброса. График вероятности сброса пакета приведён на рис. 1.8.

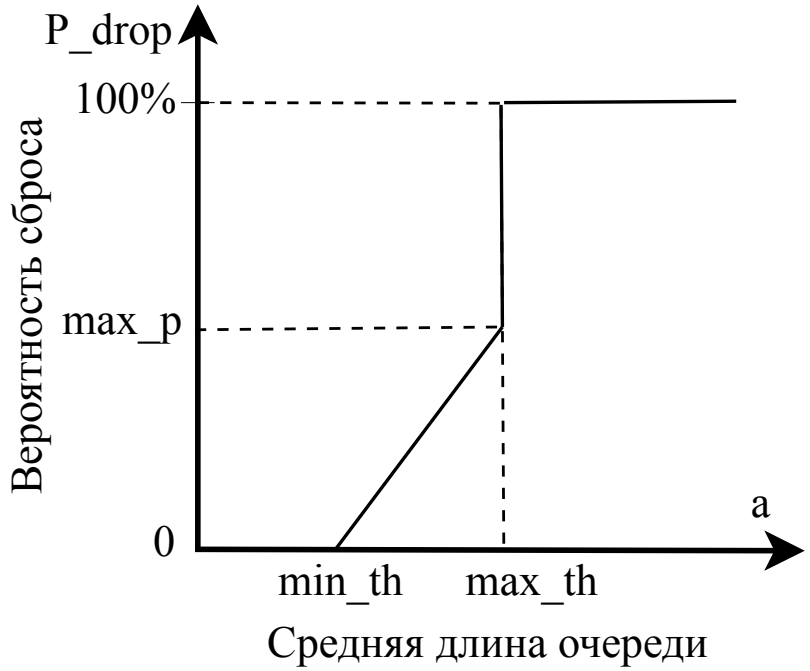


Рисунок 1.8 — Зависимость вероятности сброса от усреднённой длины очереди для метода RED

Реальная вероятность сброса пакетов P_a вычисляется на основе счётчика поступивших пакетов с момента последнего сброса:

$$P_a = \frac{P_{drop}}{(1 - n \cdot P_{drop})}, \tag{1.5}$$

где n — количество пакетов, пришедших в очередь с момента последнего сброса.

В современных маршрутизаторах применяется модификация WRED (Weighted RED) — взвешенный RED [62], которая расширяет применение алгоритма RED для обслуживания пакетов с разным приоритетом и для поддержки архитектуры дифференцированного обслуживания DiffServ [63]. В работах [64,65] было исследовано влияние параметров настройки RED, таких как весовой коэффициент w, используемый для расчёта усреднённой длины очереди, максимальная вероятность сброса max_p и установленные пороги на возникновение осцилляций длины очереди и параметры качества обслуживания. На основе проведённого имитационного моделирования в NS–2 и последующих

испытаний на сети передачи данных с применением коммерческих маршрутизаторов были выработаны рекомендации по выбору параметров RED, при которых не возникают осцилляции и получаются оптимальные параметры качества обслуживания. Тем не менее, выбор адекватных параметров RED является сложной задачей. Сложность подбора параметров и крайняя зависимость поведения алгоритма от установки этих фиксированных значений является одним из недостатков метода RED. Для адаптивной подстройки параметров был предложен механизм **Adaptive RED** [3, 7], способный динамически изменять max_p в зависимости от загрузки очереди и автоматически рассчитывающий значение max_{th} и весовой коэффициент усреднения w по формуле:

$$w = 1 - \mathrm{e}^{1/C}, \tag{1.6}$$

где C — скорость в канале в пакетах в секунду.

Для динамического регулирования параметра max_{th} вводится аддитивный коэффициент $\alpha = min(0.01, max_p/4)$ и мультипликативный коэффициент $\beta = 0.9$. Через каждый заданный интервал измерения среднее значение очереди a сравнивается с некоторым уровнем l:

$$l = [min_{th} + 0.4 \cdot (max_{th} - min_{th}), min_{th} + 0.6 \cdot (max_{th} - min_{th})], \tag{1.7}$$

и если выполняется условие:

$$(a > l) \quad and \quad (max_p \leqslant 0.5), \tag{1.8}$$

то max_p увеличивается по формуле:

$$max_p = max_p + \alpha. \tag{1.9}$$

В противном случае, если

$$(a < l) \quad and \quad (max_p \geqslant 0.01), \tag{1.10}$$

то max_p уменьшается по формуле

$$max_p = max_p \cdot \beta. \tag{1.11}$$

Таким образом параметры метода RED динамически адаптируются в зависимости от уровня интенсивности нагрузки.

1.3.3 Метод PI

Пропорционально-интегральный контроллер (PI) [5, 6] является классическим регулятором с обратной связью используемым в системах автоматического управления. Работа контроллера основывается на вычислении ошибки между текущей длиной очереди $q(t)$ и эталонной длинной q_{ref}:

$$e(t) = q(t) - q_{ref}. \tag{1.12}$$

Управляющий сигнал $u(t)$, изменяющий вероятность сброса вычисляется как значение пропорциональное текущей ошибки $e(t)$, интегралу ошибки (для устранения постоянной составляющей) и дифференциалу ошибки (для противодействия будущим отклонениям):

$$u(t) = K_P e(t) + K_I \int e(t)\,dt + K_D \frac{d}{dt} e(t), \tag{1.13}$$

где K_x — соответствующие коэффициенты пропорциональности.

На практике управления очередью используется упрощённая дискретная формула вычисления вероятности сброса без дифференциального слагаемого:

$$p(k) = (a - b)(q(k) - q_{ref}) + b(q(k) - q(k-1)) + p(k-1). \tag{1.14}$$

Рекомендованные коэффициенты пропорциональности a и b фиксированы и равны $1,822 \cdot 10^{-5}$ и $1,81 \cdot 10^{-5}$ соответственно. В каждом конкретном случае подобрать оптимальные коэффициенты пропорциональности для обеспечения необходимых параметров качества обслуживания очень сложно, поэтому в работе [66] для выбора коэффициентов пропорциональности был предложен метод машинного обучения с помощью генетического алгоритма. Для исследуемой системы из 100 одновременных TCP–соединений, передаваемых через канал 10 Мбит/с с задержкой 60 мс, были предварительно рассчитаны коэффициенты и

проведено имитационное моделирование в NS–2. Моделирование показало улучшение параметров качества обслуживание, но используемое машинное обучение требует предварительных расчётов.

1.3.4 Метод REM

В методе **случайного экспоненциального маркирования (REM)** [9] используется мера перегрузки p, называемая «ценой», и в момент времени kT вычисляемая по формуле:

$$p(kT) = max(0, p((k-1)T) + \gamma(\alpha(q(kT) - q_{ref}) + x(kT) - c)), \tag{1.15}$$

где c — скорость в канале (пакетов за интервал времени), $q(kT)$ — длина очереди в момент kT, q_{ref} — заданная длина очереди, $x(kT)$ — скорость поступления пакетов, α и γ — константы больше нуля, T — интервал времени измерений, k — номер интервала. Таким образом, при управлении учитывается, как текущая разница между текущей и заданной длиной очереди, так и текущая скорость поступления пакетов.

Вероятность сброса/маркировки пакетов рассчитывается по формуле:

$$prob(kT) = 1 - \varphi^{-p(kT)}, \tag{1.16}$$

где φ — константа больше единицы. Рекомендованные значения констант: $\alpha = 0.1$, $\gamma = 0.001$ и $\varphi = 1.001$

Авторами метода REM было проведено имитационное моделирование в NS–2 и сделано сравнение характеристик работы методов Tail Drop, RED и REM для системы с изменяющимся количеством TCP–соединений от 20 до 500, которые передавались через канал 64 Мбит/с с поддержкой механизма ECN и без. Метод REM показал способность удерживать длину очереди около заданного значения вне зависимости от количества TCP–источников, высокий коэффициент использования канала при малых потерях, а также стабильную работу при больших значениях задержки в канале.

1.3.5 Метод AVQ

Авторы метода адаптивной виртуальной очереди AVQ (Adaptive Virtual Queue) [11] предложили оригинальную идею организовать виртуальную очередь такой же длины, как и реальная очередь. При поступлении новых пакетов в реальную очередь пакеты также помещаются в виртуальную очередь, т е. пакеты не на самом деле помещаются в виртуальную очередь, а только отслеживаются. В отличие от реальной очереди, пакеты из виртуальной очереди обслуживаются виртуальным каналом со скоростью C_V меньшей, чем скорость реального канала C_R. Соответственно пакеты в виртуальной очереди будут накапливаться быстрее, и в момент времени, когда виртуальная очередь переполнится, принимается решение о превентивном сбросе или маркировке пакета в реальной очереди. Таким образом, можно избежать переполнения реальной очереди и изменяя скорость виртуального канала C_V регулировать коэффициент использования реального канала передачи данных. Изменять скорость виртуального канала предлагается по следующей формуле для производной по времени $\dot{C}_V$:

$$\dot{C}_V = \alpha(\gamma C_R - \lambda), \tag{1.17}$$

где λ — скорость поступления пакетов, γ — заданный коэффициент использования канала, α — сглаживающий параметр. То есть изменение виртуальной скорости канала пропорционально реальной скорости канала умноженной на желаемый коэффициент использования и обратно-пропорционально скорости поступления пакетов. Максимальное значение сглаживающего параметра α предлагается вычислять по формуле зависящей от задержки в канале, числа соединений и требуемого коэффициента использования канала. Рекомендованное значение $\alpha = 0.15$.

Авторами метода AVQ был проведён ряд экспериментов по имитационному моделированию в NS–2 для канала с пропускной способностью 10 Мбит/с с требуемым коэффициентом использования 0.98, с различными задержками в канале для метода AVQ и для методов RED, REM и PI. Метод AVQ показал способность удерживать малую среднюю длину очереди при высоком коэффициенте использования канала и малые потери, а также способность адаптироваться к изменениям параметров состояния сети.

1.3.6 Метод FLC

В управлении нелинейными системами со сложной динамикой себя хорошо зарекомендовали регуляторы на базе нечёткой логики **FLC** (Fuzzy Logic Controller — регулятор (контроллер) с нечёткой логикой) [67]. Описание теории нечёткой логики будет дано в главе 2.

Конструкция нечёткого регулятора для активного управления длиной очереди в маршрутизаторе была предложена в работе [12]. Авторами работы разработан нечёткий регулятор в двумя входами и одним выходом. Первым входным параметром является разница (ошибка) между текущей длиной очереди и заданным эталонным значением длины очереди, а в качестве второго параметра используется приращение ошибки длины очереди за последний интервал измерения. Таким образом, по приращению ошибки за фиксированный интервал времени можно судить о направлении и скорости изменения длины очереди. В отличие от метода RED, где расчёт вероятности сброса ведётся при каждом поступлении нового пакета на вход очереди, в методе FLC измерение и расчёт производится через некоторый интервал времени. При передаче данных через канал со скоростью 100 Мбит/с и при длине пакетов 1000 байт, каждые 10 мс (частота измерений 100 Гц) может приходить в очередь до 125 пакетов. С учётом того, что в маршрутизаторах длина очереди может достигать сотни пакетов, была выбрана частота измерений 160 Гц, как достаточная для оценки динамики изменения длины очереди. Входные значения приводились к нечётким значениям с помощью 7 функций принадлежности треугольной формы каждая, а выходная величина имела 11 треугольных функций принадлежности. Выходная величина являлась инкрементом вероятности сброса или маркировки пакетов и принимала значения в интервале $[-8.75 \cdot 10^{-5}, 8.75 \cdot 10^{-5}]$. То есть, за каждый интервал измерений 6.25 мс вероятность сброса максимально могла измениться на величину $8.75 \cdot 10^{-5}$. В NS–2 была создана имитационная модель сети с маршрутизатором реализующим управление длиной очереди с помощью метода FLC. На имитационной модели генерировался трафик FTP, HTTP и UDP при разных настройках параметров сети и поведение длины очереди сравнивалось с поведением при использовании метода PI в качестве метода управления. Метод FLC показал лучшую способность стабилизировать длину

очереди у заданного значения, чем метод PI. Также метод FLC при резком изменении интенсивности трафика быстрее проходит стадию переходного процесса и переходит в стационарный режим.

В работе [68] был предложен метод **Adaptive FLC (AFLC)** с сокращённым количеством функций принадлежности, но с использованием адаптивной подстройкой набора правил для нечёткого вывода. Метод был проверен с помощью имитационного моделирования в NS–2 и показал лучшее время регулирования и устойчивость средней длины очереди, чем метод PI.

В методе **Fuzzy Explicit Marking (FEM)** [13] в качестве входных параметров нечёткого регулятора использовались значение ошибки очереди (разница между текущим значением и эталонным) и значение предыдущей измеренной ошибки, которые масштабировались для получения значений на входе в интервале $[-1, 1]$. Выходной величиной была непосредственно вероятность сброса, в отличии от работы [12], где использовалось приращение вероятности сброса. Для входных и выходных величин использовалось по 7 треугольных функций принадлежности. Полученная с выхода нечёткого регулятора вероятность сброса не сразу применялась для управления очередью, адаптивно масштабировалась используя подход аналогичный в методе Adaptive RED [3]. Авторами метода FEM было проведено имитационное моделирование в NS–2 и выполнено сравнение с методами ARED, PI, REM и AVQ. Метод FEM показал наименьшую вариацию задержки пакетов при высоком коэффициенте использования канала передачи данных и при минимальных потерях пакетов.

Авторами работы [69] для расчёта функций принадлежности нечёткого регулятора в методе **Self–Organized Fuzzy Logic Congestion Detection (Self–Org FLCD)** предложено использовать алгоритм само–оптимизации на основе многоцелевого роя частиц (Multi–Objective Partcle Swarm Optimization, MOPSO). В методе Self-Org FLCD использовалась архитектура нечёткого регулятора с двумя входами и одним выходом. Первой входной величиной было отношение текущей длины очереди к максимальному размеру очереди. Второй входной величиной в отличии от работы [12] предложено использовать средневзвешенную скорость поступления пакетов в очередь на интервале измерения. Усреднение скорости поступления подстраивалось в учётом задержки распространения пакетов в канале, что однако требует зафиксировать эту величину при настройке. В качестве выходного значения использовалось приращение ве-

роятности сброса/маркировки пакетов. Дополнительно использовался активатор для резкого увеличения, в момент перегрузки, вероятности сброса пакетов для тех потоков, которые не реагировали на уведомления о перегрузке и не сокращали интенсивность нагрузки. Параметры функций принадлежности и максимальные значения приращения вероятности сброса были рассчитаны с помощью алгоритма MOPSO с целью выбора таких оптимальных значений, чтобы удовлетворить по возможности одновременно нескольким требованиям:

- максимальный коэффициент использования канала;
- минимальные потери пакетов;
- минимальная задержка пакетов;
- минимальный джиттер.

Имитационное моделирование проведённое в NS–2 показало улучшение параметров качества обслуживание по сравнению с методами ARED, FLC и FLCD. Однако, оптимизация параметров FLC требует дополнительных вычислительных ресурсов, которые в реальном маршрутизаторе ограничены и должны использоваться для основной задачи маршрутизации.

В работе [70] предложен метод **Fuzzy Random Exponential Marking (FREM)**, который также основан на оценке перегрузки p по формуле 1.15 аналогично методу REM, но для расчёта вероятности сброса используется нечёткий регулятор FLC. На первый вход регулятора подаётся текущее значение p, а на второй значение p измеренное в предыдущий интервал времени. Выходным значением является вероятность сброса пакетов. Таким образом, метод является комбинацией методов REM и FLC. Для входных и выходных переменных используются по 9 функций принадлежности треугольного вида. Авторами метода FREM проведено имитационное моделирование в NS–2 для одного вида трафика — длительные TCP–соединения (от 100 до 300 одновременных соединений), которые имитировались FTP приложением. Для оценки качества работы метода использовалась интегральная характеристика оценивающая время реакции метода на изменение интенсивности нагрузки и среднеквадратичную ошибку управления. Результаты имитационного моделирования показывают лучшее время стабилизации длины очереди при использовании метода FREM, чем у методов REM и PI, но и большие потери пакетов. В дальнейшем авторы планируют добавить PI–регулятор для динамической подстройки нечёткого регулятора для работы при различных сетевых топологиях и параметрах трафика.

Авторами метода **Fuzzy Logic Active Queue Managment (FLAQM)** [71] в качестве входных переменных для нечёткого регулятора используется специальный фактор интенсивности нагрузки z, который служит индикатором перегрузки, и приращение фактора Δz. Фактор z рассчитывается с учётом количества пришедших на вход очереди пакетов за интервал измерения, максимально возможной скорости передачи пакетов и разницы между текущей и заданной длиной очереди. В зависимости от положительного или отрицательного значения z, то есть растёт или падает мера перегрузки, то применяется один из двух нечётких регуляторов FLC. При росте нагрузки применяется регулятор с аддитивным увеличением вероятности сброса пакетов Additive Increase (AI) FLAQM, а при падении регулятор с мультипликативным сокращением вероятности сброса Multiplicative Decrease (MD) FLAQM. Имитационное моделирование с трафиком FTP имитирующим длительные по времени TCP–соединения и с Web–трафиком имитирующем короткие TCP–соединения показали превосходство метода FLAQM перед традиционными методами Tail Drop, RED и Adaptive RED. В дальнейших работах авторы предполагают использовать методы искусственного интеллекта, такие как нейронные сети и генетические алгоритмы для самонастройки параметров нечётких регуляторов FLAQM, что станет актуально с ростом вычислительной мощности маршрутизаторов в будущем.

1.4 Сравнение эффективности работы различных методов управления очередями на базе имитационной модели в NS–2

Для оценки эффективности работы различных методов управления в одинаковых сетевых условиях использовалось имитационное моделирование в программном комплексе NS–2 (Network Simulator–2) [72, 73]. Для имитации различных видов трафика сети Интернет, использовалась одновременная передача данных как с протоколом TCP, так и с UDP протоколом. В работе [74] приводятся данные измерений трафика на сети оператора связи предоставляющего услуги Интернет–доступа для домашних и корпоративных пользователей. Согласно этому исследованию на UDP–трафик приходятся 5,6% всего передавае-

мого объёма трафика, а на TCP–соединения 90% трафика. Среди TCP–трафика на долю протокола HTTP приходится 45% трафика по объёму.

В настоящем исследовании TCP–трафик, аналогично другим рассмотренным выше исследованиям, разделён на длительные соединения, которые имитируют сервисы передачи файлов или просмотр видео–программ, и короткие соединения, которые имитируют просмотр Web–страниц. Трафик UDP с небольшой постоянной скоростью имитирует IP–телефонное соединение или вещание аудио–потока Интернет–радио. Предположим, что весь трафик имеет один класс обслуживания CoS и передаётся через одну очередь на выходном интерфейсе маршрутизатора. Пограничный маршрутизатор подключён к каналу передачи данных с ограниченной полосой пропускания 50 Мбит/с и задержкой передачи пакетов 5 мс. С другой стороны, для создания перегрузки в канале, к маршрутизатору подключены источники трафика по каналам с пропускной способностью 100 Мбит/с каждый, но с разной задержкой в канале от 1 до 9 мс. Разная задержка использовалась, чтобы не создать перегрузку в начальный момент времени.

Длительные по времени TCP соединения в NS–2 создаются при помощи 100 одновременных FTP–сессий, а короткие с помощью HTTP–сессий запросов клиентов к веб–серверам с интенсивностью по 50 новых запросов в секунду. Дополнительно к TCP трафику одновременно передаются два потока UDP с постоянной скоростью (Constant Bit Rate, CBR) 128 кбит/с. Трафик TCP и UDP передавался пакетами по 1000 байт. Поддержка ECN включена для всех TCP соединений. На маршрутизаторе размер выходной очереди ограничен 500 пакетами. Схема сети для имитационного моделирования в NS–2 приведена на рисунке 1.9.

Кроме того, для имитации резких изменений интенсивности трафика, 50 FTP–передатчиков из 100 прекращали передачу через 40 секунд после начала передачи на 30 секунд, а потом снова возобновляли. Такой сценарий повторялся каждые 100 секунд.

На рис. 1.10 показан трафик FTP в канале с перегрузкой между двумя маршрутизаторами, а на рис. 1.11 трафик HTTP. На рис. 1.12 показано количество активных одновременных сессий к Web–серверу.

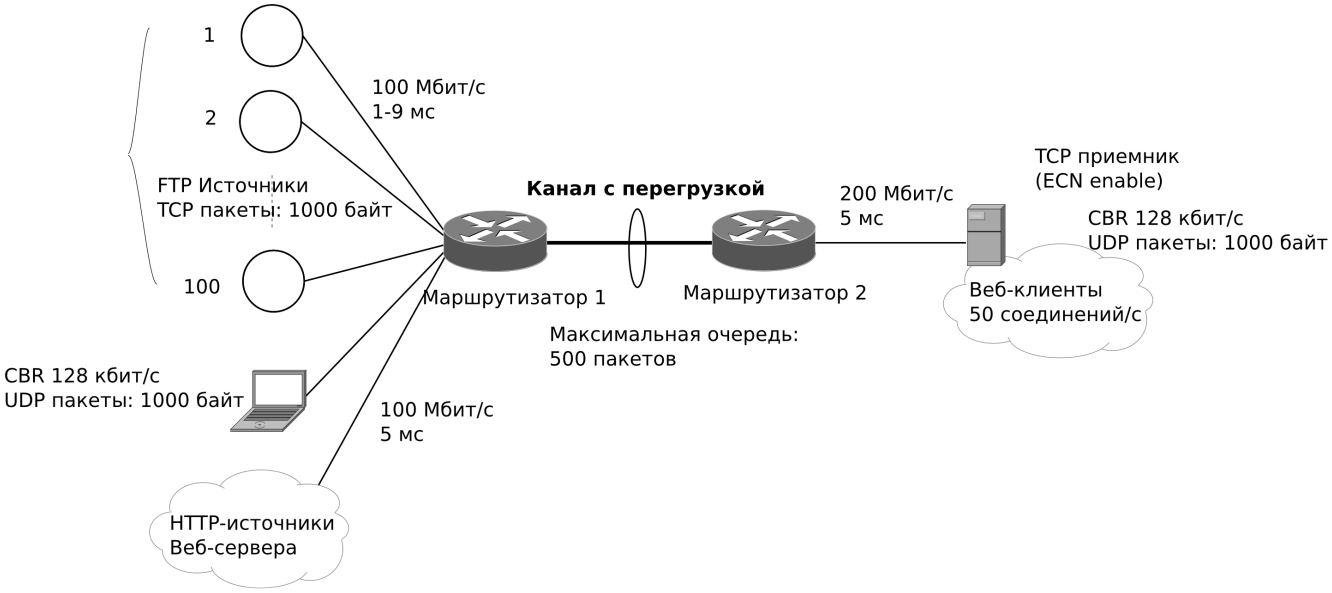


Рисунок 1.9 — Схема сети имитационного моделирования

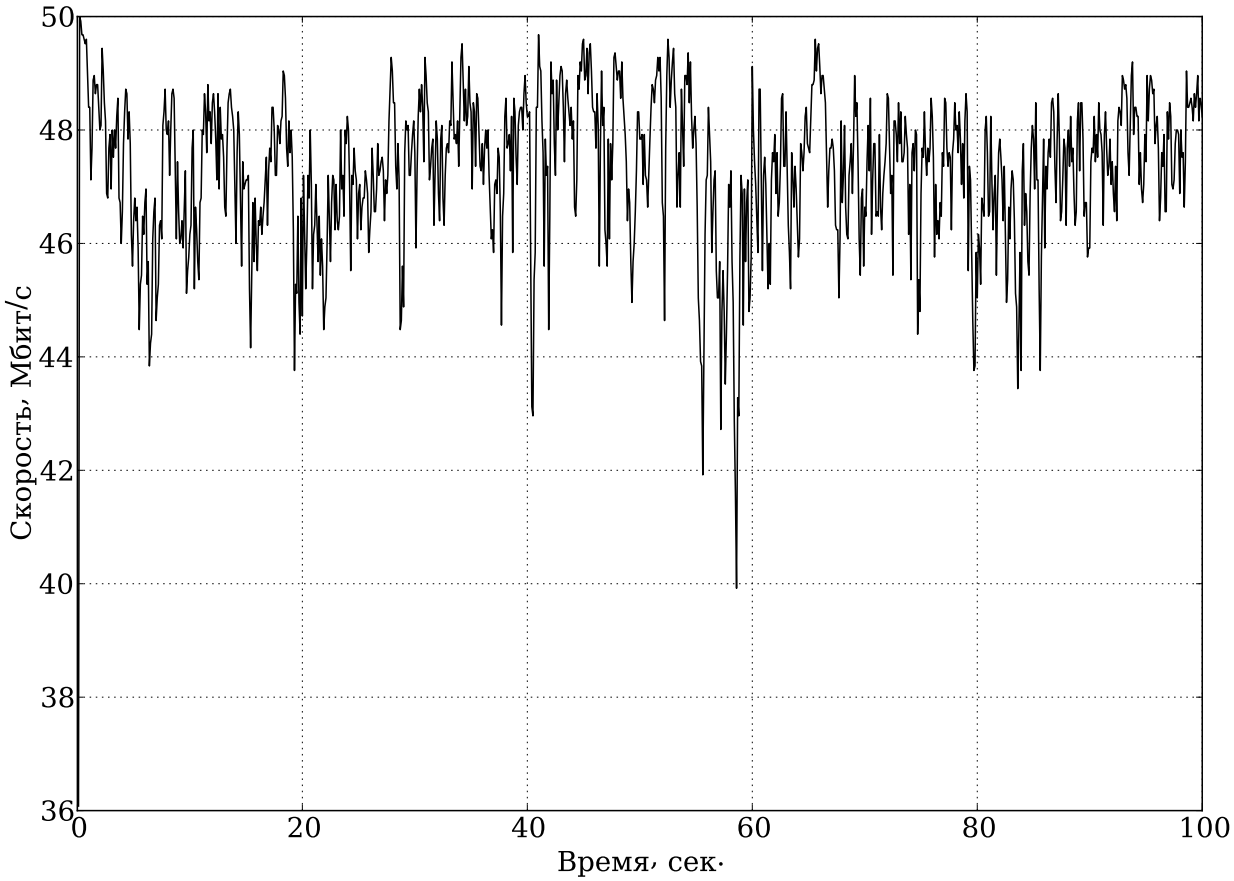


Рисунок 1.10 — FTP трафик

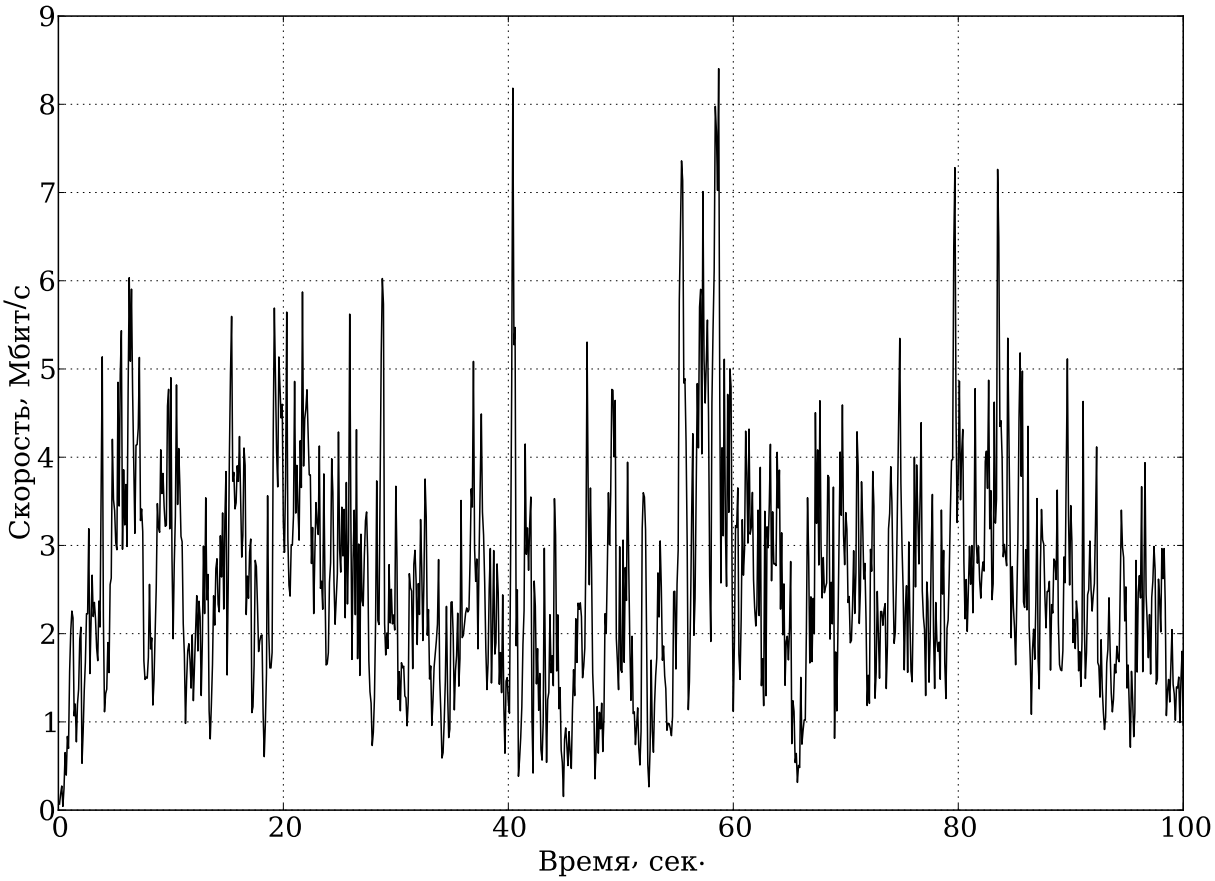


Рисунок 1.11 — HTTP трафик

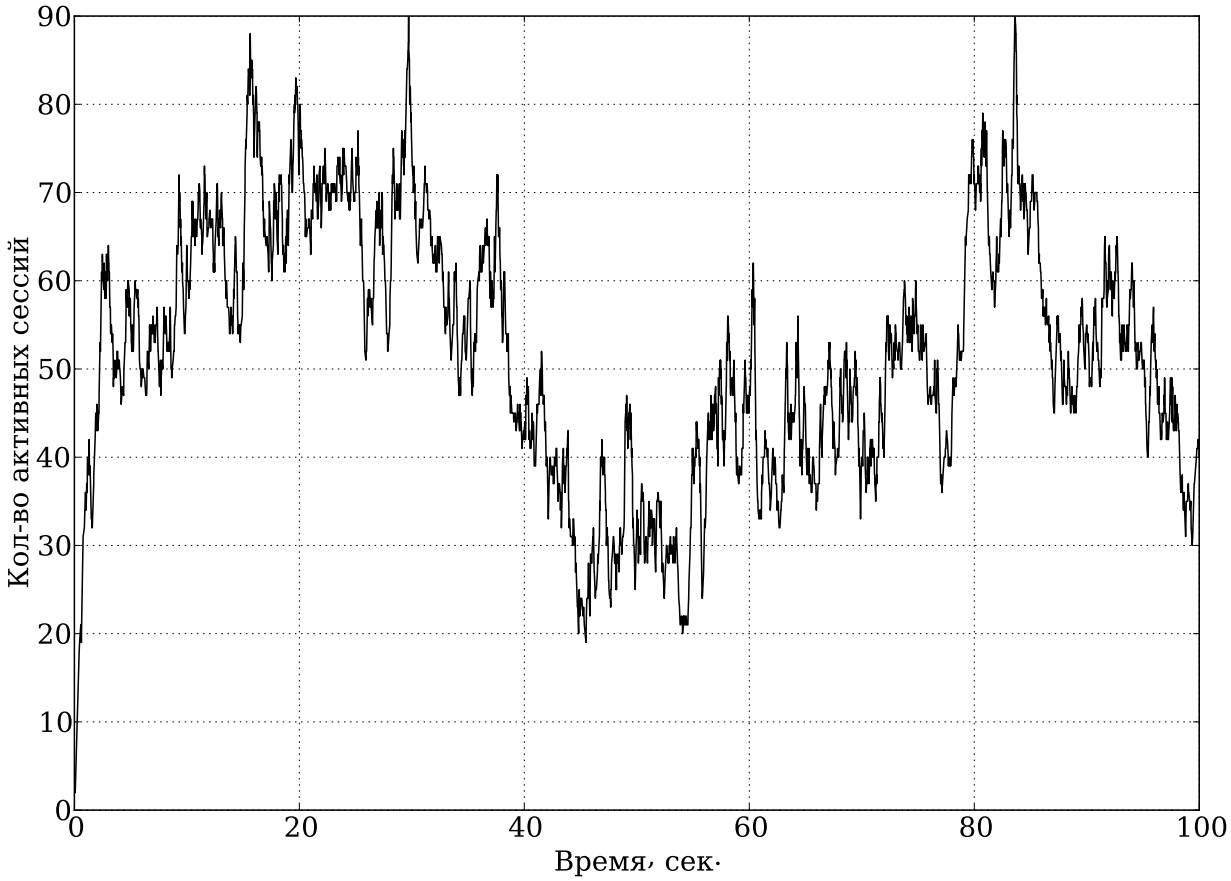


Рисунок 1.12 — HTTP сессии

С данной тестовой схемой использовалось 8 различных методов управления очередью маршрутизатора: ARED, AVQ, TailDrop, FEM, FLC2, PI, RED и REM. Для каждого метода моделирование длилось 100 секунд и потом повторялось ещё 6 раз для расчёта средних значений и доверительных интервалов. Сценарий для имитационного моделирования в NS–2 на языке программирования TCL ($flc2\_600.tcl$) приведён в репозитории [75]. Изменение длины очереди для различных методов показаны в приложении А.

Для настройки метода Adaptive RED использовались параметры приведённые в таблице 1.2.

Таблица 1.2 — Параметры метода Adaptive RED

Параметр	Значение
Минимальный порог min_{th}	200 пакетов
Максимальный порог max_{th}	400 пакетов
Максимальное значение вероятности сброса max_p	1/30
Аддитивный коэффициент α	0.01
Мультипликативный коэффициент β	0.9

График изменения длины очереди при использовании метода Adaptive RED приведён на рисунке в приложении А.1. На графике видно, что метод удерживает длину очереди в процессе эксперимента между установленными порогами, допуская значительные всплески в моменты скачкообразного изменения интенсивности трафика (в моменты времени 40 секунд и 70 секунд после начала передачи).

Изменения длины очереди для метода AVQ приведены в приложении А.2 при заданном коэффициенте использования канала $\gamma = 0.98$. Метод AVQ поддерживает длину очереди на минимальном уровне и не имеет механизмов удержания длины очереди на заданном уровне. В моменты резкого увеличения интенсивности трафика длина очереди возрастает.

Метод TailDrop также не имеет механизмов управления длиной очереди, и при его использовании очередь переполняется и держится на максимальном уровне, что приведёт к значительным потерям пакетов. График изменения длины очереди для метода RED приведён в приложении А.3.

В приложении А.4 приведён график изменения длины очереди для метода FEM с заданной длиной очереди $Q_{ref} = 300$ пакетов. Метод старается удержать

длину очереди около заданного значения, по при скачках интенсивности видны смещения длины очереди.

Для разработанного метода FLC2 изменение длины очереди приведено на рисунке в приложении А.5. При моделировании в NS–2 использовались параметры приведённые в таблице 1.3

Таблица 1.3 — Параметры метода FLC2

Параметр	Значение
Заданная длина очереди Q_{ref}	300 пакетов
Максимальное приращение вероятности сброса	$8 \cdot 10^{-5}$
Интервал измерений	6 мс

Метод FLC2 удерживает длину очереди у заданного значения, лишь в моменты скачков интенсивности трафика видны на графике пики длины очереди.

Изменения длины очереди для метода PI приведены в приложении А.6. Параметры метода приведены в таблице 1.4.

Таблица 1.4 — Параметры метода PI

Параметр	Значение
Заданная длина очереди Q_{ref}	300 пакетов
Коэффициент пропорциональности a	$1,822 \cdot 10^{-5}$
Коэффициент пропорциональности b	$1,81 \cdot 10^{-5}$

На графиках изменения длины очереди при использовании метода PI видно большое время реакции, по сравнению с другими методами, на скачок интенсивности трафика. После изменения интенсивности начинается длительный переходный процесс, которой постепенно приводит длину очереди к заданному значению.

В приложении А.7 приведён график изменения длины очереди для метода RED. При моделировании использовались параметры приведённые в таблице 1.5.

Метод RED, в зависимости от интенсивности трафика, устанавливает разный уровень длины очереди в пределах заданных порогов. При изменении интенсивности начинается переходный процесс выражающийся в возникновении значительных осцилляций длины очереди.

Таблица 1.5 — Параметры метода RED

Параметр	Значение
Минимальный порог min_{th}	200 пакетов
Максимальный порог max_{th}	400 пакетов
Максимальное значение вероятности сброса max_p	1/30
Весовой коэффициент w	0.002

Изменение длины очереди для метода REM приведено в приложении А.8. Параметры метода использованные для моделирования приведены в таблице 1.6.

Таблица 1.6 — Параметры метода REM

Параметр	Значение
Заданная длина очереди Q_{ref}	300 пакетов
Константа γ	0.001
Константа ϕ	1.001

График изменения длины очереди при использовании метода REM показывает значительные отклонения длины очереди в моменты изменения интенсивности трафика, и следующий затем продолжительный переходный процесс возвращающий длину очереди к заданному значению.

Характеристики системы передачи данных при имитационном моделировании перегрузки в сети со смешанным типом трафика приведены в таблице 1.7.

Таблица 1.7 — Характеристики системы передачи данных

Метод	Средняя длина очереди	СКО	Коэфф. использования	Потери пакетов, %
ARED	298	33	0,99996	0,13
AVQ	17	20	0,95747	0,01
TailDrop	468	31	0,99998	3,92
FEM	291	32	0,99996	0,36
FLC2	322	33	0,99996	0,34
PI	300	102	0,99996	0,22
RED	319	45	0,99996	0,11
REM	297	57	0,99996	0,05

Методы AVQ и TailDrop не имеют механизмов поддержания заданной длины очереди, поэтому при использовании метода TailDrop средняя длина очереди

находится у верхнего предела и при наступлении перегрузки переполняется, а метод AVQ, наоборот, держит размер очереди на минимальном уровне и показывает меньший коэффициент использования канала, чем другие методы.

Методы RED, REM и особенно метод PI допускают значительные отклонения длины очереди от среднего значения. Среднеквадратичное отклонение (СКО) длины очереди для этих методов является наибольшим. Значения средней длины очереди и среднеквадратичного отклонения с доверительными интервалами (достоверность 95%) для различных методов приведены на рис. 1.13.

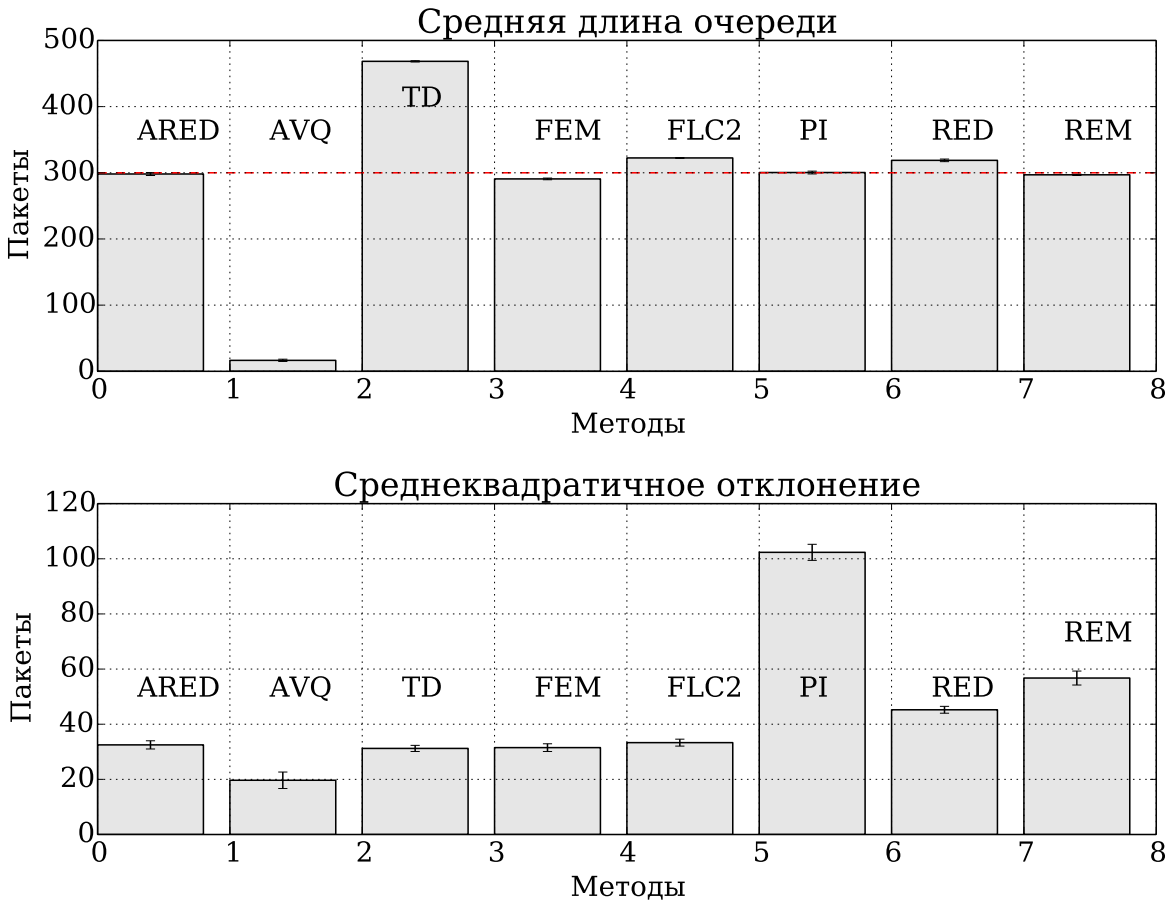


Рисунок 1.13 — Средняя длина очереди и среднеквадратичное отклонение при максимальном размере буфера 500 пакетов и эталонном значении 300 пакетов

Самые большие потери происходят при использование метода TailDrop (TD) из–за постоянного переполнения очереди. Наименьшие потери допускают методы AVQ и REM.

Рассчитанные параметры качества обслуживания при использовании различных приложений приведены в таблицах 1.8 и 1.9.

Таблица 1.8 — Параметры качества обслуживания FTP и UDP

	Трафик FTP		Трафик UDP		
Метод	Средняя скорость, Мбит/с	Потери пакетов, %	Потери пакетов, %	Средняя задержка, мс	Джиттер, мс
ARED	46,6	0,031	4,19	57	2,3
AVQ	44,5	0,013	0,02	13	1,44
TailDrop	46,6	3,875	3,28	78	3,17
FEM	46,6	0,003	14,43	55	3,1
FLC2	46,6	0,027	12,73	61	3
PI	46,6	0,126	3,97	58	2,54
RED	46,6	0,023	3,77	60	2,57
REM	46,6	0,051	0,02	57	2,47

Имитационное моделирование и рассчитанные параметры качества обслуживания показывают, что при использовании механизма TailDrop значительные потери из-за переполнения буфера приходятся на короткие HTTP соединения. При использовании активных методов управления TCP–соединения имеют незначительные потери. Сброс пакетов в основном приходится на пакеты протокола UDP, который не имеет механизмов контроля передачи.

Недостаточно интенсивный упреждающий сброс пакетов приводит к переполнению очереди и ещё большему одновременному сбросу всего трафика, и наоборот, чрезмерный упреждающий сброс — к опустошению очереди и к уменьшению коэффициента использования канала. Поэтому необходимо разработать нечёткий регулятор для автоматической подстройки вероятности сброса или маркировки пакетов, оптимальной для стабилизации длины очереди около эталонного значения, что также позволит уменьшить джиттер. Для оценки эффективности работы регулятора необходимо построить математическую модель и рассчитать вероятностно–временные характеристики. Полученные аналитические данные далее нужно сравнить с результатами имитационного моделирования и результатами испытаний на реальном оборудовании для проверки адекватности разработанного метода.

Таблица 1.9 — Параметры качества обслуживания HTTP

	Трафик HTTP			
Метод	Средняя скорость, Мбит/с	Потери пакетов, %	Среднее время сессии, мс	СКО времени сессии, мс
ARED	3,26	0,9998	428	1170
AVQ	3,26	0,0018	193	948
TailDrop	3,42	4,4611	490	1242
FEM	3,26	3,5622	849	2471
FLC2	3,26	3,1754	766	2022
PI	3,26	1,0619	430	1161
RED	3,26	0,8778	421	1111
REM	3,26	0,0369	302	911

Выводы

1. Проведённый анализ развития новых технологий OTT, VoD, P2P, M2M, IoT предопределяет в ближайшем будущем резкое увеличение нагрузки в IP–сетях с превалированием TCP трафика, что выдвигает на первый план методы борьбы с перегрузками.
2. Исследование методов управления нагрузкой наглядно показывает преимущество активных методов обработки трафика RED, ARED, PI, REM, AVQ, FEM, FLC. Вместе с тем, использование активного управления в условиях передачи разнородного трафика в IP–сетях провайдеров не всегда даёт требуемые результаты.
3. Применение методов обработки трафика на основе нечёткой логики показывают их высокую эффективность во многих областях. Необходимо провести разработку нового метода обработки трафика в очередях маршрутизаторов на основе нечёткой логики для предотвращения перегрузок и улучшения качества обслуживания в современных мультисервисных IP–сетях.

Глава 2. Разработка метода управления очередью на базе нечёткой логики

2.1 Нечёткая логика

Основные идеи теории нечётких множеств были изложены Лотфи Заде в середине 60–х годов 20 века [76]. Эта теория позволяет описывать нечёткие качественные параметры об окружающем мире, а главное оперировать этими данными, получать новую информацию и делать нечёткие выводы. Применение нечётких методов управления позволяет расширить сферу применения систем автоматизации за пределы классической теории [77]. Л. Заде говорил: «... излишнее стремление к точности стало оказывать действие, сводящее на нет теорию управления и теорию систем, так что оно приводит к тому, что исследования в этой области сосредотачиваются на тех и только тех проблемах, которые поддаются точному решению. В результате многие классы важных проблем, в которых данные, цели и ограничения являются слишком сложными или плохо определёнными для того, чтобы допустить точный математический анализ, оставались и остаются в стороне по той причине, что они не поддаются математической трактовке. ... мы должны отказаться от наших требований точности и допустить результаты, которые являются несколько размытыми или неопределёнными». Математические средства предложенные Заде для работы с нечёткими исходными условиями позволяют описывать процессы, в которых присутствует неопределённость, которая делает чрезвычайно сложным применение точных, количественных методов.

Нечёткие множества являются обобщением обычных математических множеств, когда происходит отказ от бинарного характера принадлежности некоторого элемента множеству, т.е. функция принадлежности принимает значение 1 («истина»), когда элемент принадлежит множеству и значение 0 («ложь»), когда не принадлежит [78]. Функция принадлежности нечёткого множества может принимать любые значения на отрезке $[0, 1]$.

В теории нечётких множеств, нечёткое множество характеризуется **функцией принадлежности**, а её значение μ_A — степенью принадлежности эле-

мента x нечёткому множеству A. Или более точно, нечётким множеством A называется совокупность пар:

$$A = \{\langle x, \mu_A(x)\rangle | x \in U\}, \tag{2.1}$$

где μ_A — функция принадлежности, т.е. $\mu_A : U \to [0, 1]$.

Нечёткая **лингвистическая переменная** отличается от обычной числовой переменной тем, что её значением является не число, а слово в естественном языке. Понятие лингвистической переменной даёт возможность приближённо описывать сложные явления, которые трудны для описания традиционными количественными понятиями.

В примере на рис. 2.1 показана некоторая лингвистическая переменная «Величина», которая может принимать нечёткие значения: «Малое», «Среднее» и «Большое», описанные треугольными функциями принадлежности.

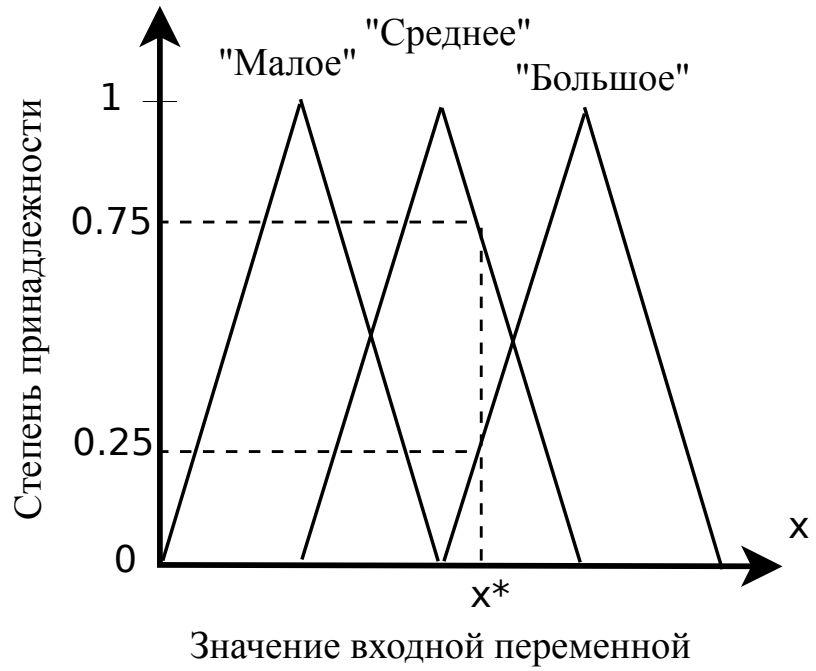


Рисунок 2.1 — Пример нечётких значений лингвистической переменной заданной треугольными функциями принадлежности

В данном примере, если входная переменная системы x приняла значение x^*, то степень принадлежности значению «Среднее» будет 0.75, а принадлежность значению «Большое» будет 0.25. Таким образом, после перехода к нечётким значениям «Среднее» и «Большое», далее возможно уже оперировать нечёткими понятиями, и на их основе составлять наборы правил и делать нечёткие выводы. Например, следующие правила сопоставляют входные переменные x_1 и

x_2 с нечёткими множествами условий A, а выходную переменную y с нечёткими множествами заключений B:

$$\begin{aligned} &\textit{Правило 1}: \ \textbf{ЕСЛИ}\ (x_1 = A_{11})\ \textbf{И}\ (x_2 = A_{12})\ \textbf{ТОГДА}\ (y = B_1), \\ &\textit{Правило 2}: \ \textbf{ЕСЛИ}\ (x_1 = A_{21})\ \textbf{И}\ (x_2 = A_{22})\ \textbf{ТОГДА}\ (y = B_2), \end{aligned} \tag{2.2}$$

В результате подстановки входящих значений в правила, вычисляются степени выполнения условий h отдельных правил, и формируются новые нечёткие множества путём отсечения множеств заключения по минимальному уровню h. Полученные таким образом из каждого правила заключения складываются в результирующее множество μ_Σ. Графическая интерпретация нечёткого вывода приведена на рис. 2.2 [12].

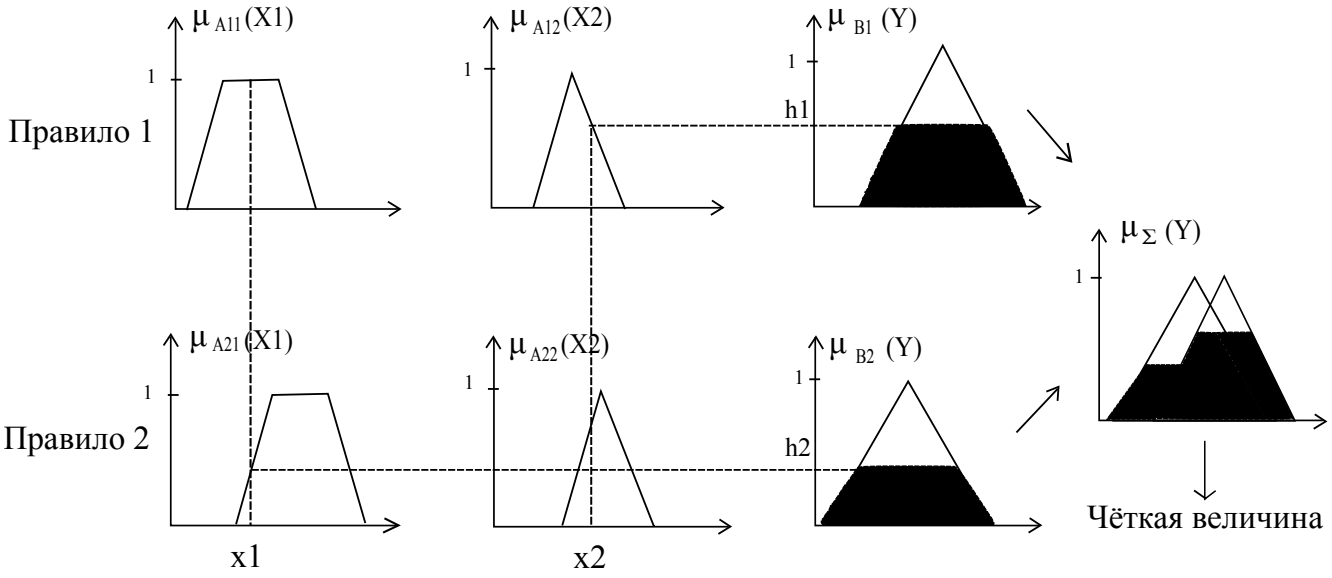


Рисунок 2.2 — Пример нечёткого вывода на основе двух правил

Результирующее множество μ_Σ может быть преобразовано в чёткую выходную величину путём нахождения интегрального среднего значения полученной фигуры. В итоге, с помощью нечётких набора правил или экспертных оценок, и оперируя нечёткими лингвистическими переменными, можно получить чёткий вывод от сложной системы с трудно описываемой традиционными методами процессами.

2.2 Примеры применения нечёткой логики

С середины 80–х годов прошлого века, первоначально в Японии, возник «бум нечёткости» — прикладное применение теории нечётких множеств, вызванный в результате появления множества устройств основанных на использовании нечёткой логики [67]. Нечёткая логика успешно применялась для задач управления поездами метрополитена, подъёмными кранами, лифтами и т.д. Основы методов нечёткого управления заложили такие исследователи, как Мамдани, Сугено [79], Такаги и другие. С начала 1990–х годов нечёткое управление находит ещё большее практическое применение в следующих областях:

- управление электронным кардиостимулятором;
- химические реакторы;
- водогрейные котлы и отопительные приборы;
- системы охлаждения и кондиционирования;
- насосные станции;
- обработка изображений;
- быстродействующие зарядные устройства;
- распознавание слов;
- управление биопроцессами;
- управление электродвигателями;
- системы управлением движения транспортом и др.

2.3 Описание метода нечёткого вывода

Один из широко распространённых методов нечёткого вывода — алгоритм Мамдани [80]. Математически он может быть представлен в виде следующих последовательных действий:

- нечёткость;
- нечёткий вывод;
- композиция;
- приведение к чёткости.

Пусть дана база правил для системы с двумя входами и одним выходом:

$$\begin{aligned} &R_1: \textbf{ЕСЛИ } (x_1 = A_{11}) \textbf{ И } (x_2 = A_{12}) \textbf{ ТОГДА } (y = B_1), \\ &... \\ &R_m: \textbf{ЕСЛИ } (x_1 = A_{m1}) \textbf{ И } (x_2 = A_{m2}) \textbf{ ТОГДА } (y = B_m), \end{aligned} \tag{2.3}$$

где $A_{11}, ..., A_{m2}$ — нечеткие множества условий, $B_1, ..., B_m$ — нечеткие множества заключений, x_1, x_2 — входы нечеткой системы, x_1^*, x_2^* — значения входов, y — выход нечеткой системы.

Пример правила: **ЕСЛИ** (Q_{error} = **«БОЛЬШОЕ»**) **И** (d_{error} = **«МАЛОЕ»**) **ТОГДА** (P_{drop} = **«СРЕДНЕЕ»**).

1) **Нечеткость** — вычисление степени истинности предпосылок для каждого правила: $\mu_{11}(x_1^*), \mu_{12}(x_2^*), \ldots, \mu_{m1}(x_1^*), \mu_{m2}(x_2^*)$, где μ_k — функции принадлежности соответствующих нечётких множеств.

   Функции принадлежности выбираются так чтобы выполнялось равенство:

$$\sum_{k=1}^{m} \mu_k(x_i) = 1 \tag{2.4}$$

2) **Нечеткий вывод** — вычисление степени h_k выполнения условий отдельных правил:

$$\begin{aligned} &h_1 = \mu_{11}(x_1^*) \wedge \mu_{12}(x_2^*), \\ &... \\ &h_m = \mu_{m1}(x_1^*) \wedge \mu_{m2}(x_2^*), \end{aligned} \tag{2.5}$$

   где через $\wedge$ обозначена операция логического минимума (MIN).

   Вычисление модифицированных функций принадлежности:

$$\begin{aligned} &\mu_{B1}^*(y) = h_1 \wedge \mu_{B1}(y), \\ &... \\ &\mu_{Bm}^*(y) = h_m \wedge \mu_{Bm}(y). \end{aligned} \tag{2.6}$$

3) **Композиция** — вычисление результирующей функции принадлежности:

$$\mu_\Sigma(y) = \mu_{B1}^*(y) \vee ... \vee \mu_{Bm}^*(y), \tag{2.7}$$

   где через $\vee$ обозначена логическая операция максимума (MAX).

4) **Приведение к четкости** — вычисление четкого значения y^* выхода системы:

$$y^* = \frac{\sum_{j=1}^{m} y_j \cdot \mu_\Sigma(y_j)}{\sum_{j=1}^{m} \mu_\Sigma(y_j)} \tag{2.8}$$

2.4 Регулятор на основе нечёткой логики FLC

Нечётким регулятором (контроллером) FLC принимаются решения об изменении текущего значения вероятности сброса/маркировки P поступившего пакета на основе значений двух входных переменных. Первая переменная q_{error} — разница (текущая ошибка) между текущим q и заданным значением длины очереди Q_{ref}, а в качестве второй переменной было предложено использовать интенсивность трафика r $(rate)$, т.е. отношение количества полученных пакетов к максимально возможному переданному количеству за интервал измерения.

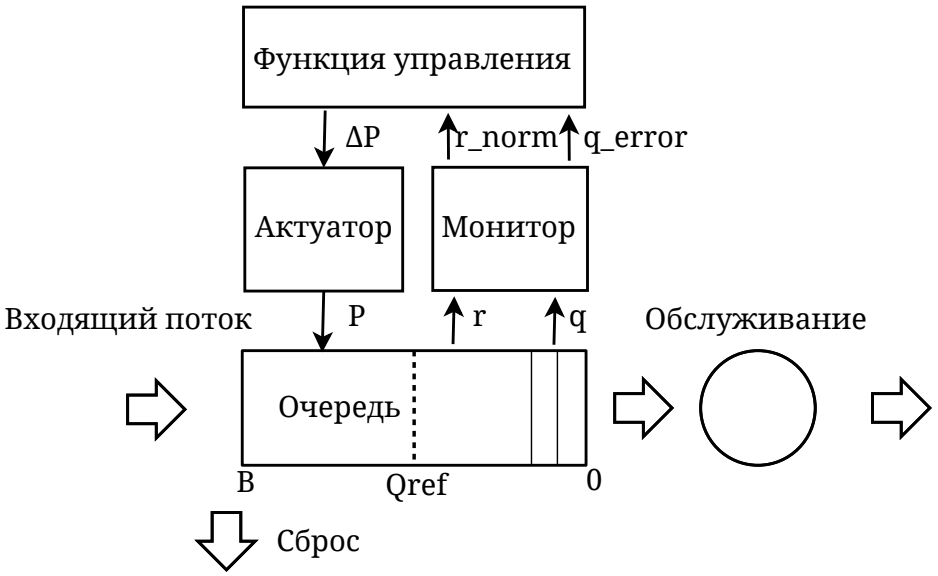


Рисунок 2.3 — Система передачи данных с активным управлением очередью

На вход модуля «Функция управления» модуль «Монитор» передаёт нормированное значение $q^i_{\text{errornorm}}$ отклонения длины очереди от эталонного значения и нормированное значение интенсивности поступления пакетов на интервале

Δt_i, которые вычисляются по формулам:

$$q^i_{\text{errornorm}} = \begin{cases} \dfrac{q^i - Q_{\text{ref}}}{B - Q_{\text{ref}}}, & q^i \geqslant Q_{\text{ref}}, \\ \dfrac{q^i - Q_{\text{ref}}}{Q_{\text{ref}}}, & q^i < Q_{\text{ref}}, \end{cases} \tag{2.9}$$

$$r^i_{\text{norm}} = \begin{cases} \dfrac{r^i - \mu}{\mu}, & \dfrac{r^i}{\mu} \leqslant 2, \\ 1, & \dfrac{r^i}{\mu} > 2, \end{cases} \tag{2.10}$$

где B — максимальный размер очереди, Q_{ref} — заданное эталонное значение длины очереди, q^i — значение длины очереди в момент времени i, r^i — значение интенсивности поступающего трафика в момент времени i и μ — интенсивность обслуживания.

Модуль «Функция управления» использует значение входных параметров $q^i_{\text{errornorm}}$ и r^i_{norm} для расчёта выходного параметра – приращения вероятности сброса пакета ΔP^i на следующем интервале наблюдения Δt^{i+1}, $i \geqslant 1$. Архитектура нечёткого регулятора реализующего функцию управления показана на рис. 2.4.

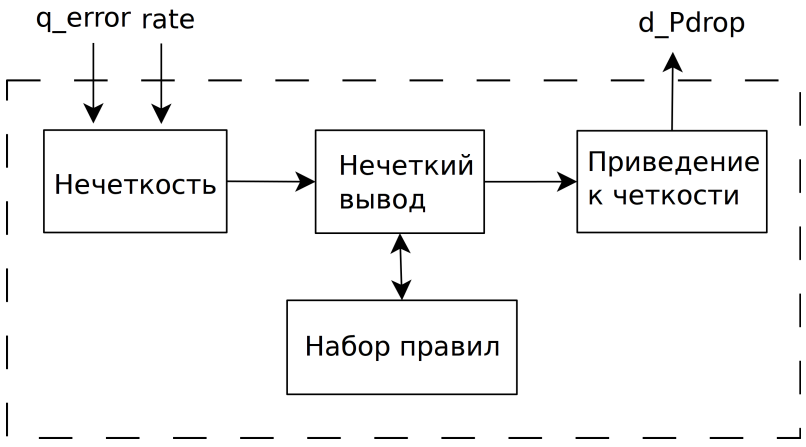


Рисунок 2.4 — Архитектура нечёткого регулятора

Нечёткий регулятор для входных переменных вычисляет значения функции принадлежности μ_x, т.е. степени принадлежности нечёткому значению. Функции принадлежности выбираются так, что сумма значений всех функции от входной переменной равна единице. Нечёткий вывод делается путём операций над множествами на основе соответствующего набора правил использующего экспертные оценки.

Согласно алгоритму Мамдани (раздел 2.3), нечёткий вывод делается на основе набора правил: вычисляется значение истинности для предпосылки каждого правила (операция minimum). Все нечёткие множества полученные из правил агрегируются вместе и формируется единственное нечёткое множество C (операция maximum). Приведение к чёткости выполняется обычно с помощью дискретного метода «центра тяжести» по формуле (2.11):

$$\Delta P_{drop} = \frac{\sum_{j=1}^{m} y_j \cdot \mu_C(y_j)}{\sum_{j=1}^{m} \mu_C(y_j)}, \tag{2.11}$$

где y_j — дискретное значение выходной переменной из m значений, а μ_C — функция принадлежности агрегированного множества C.

2.5 Выбор набора правил и функций принадлежности FLC

Форма и количество функций принадлежности для входных переменных, для перехода к нечётким значениям, а также набор правил, выбирался путём проведения ряда пробных экспериментов. Окончательный вариант выбирался тот, который обеспечивал наименьшее среднеквадратичное отклонение длины очереди от заданного значения.

Диапазон входных и выходных значений покрывался нечётным количеством нечётких величин, чтобы по центру диапазона со значением 0, присутствовала функция принадлежности отвечающая за стабильное состояние системы, когда длина очереди достигла заданного значения и интенсивность входящего трафика соответствует интенсивности обслуживания. Трех нечётких величин недостаточно для контроля за трафиком, а более 7-9 величин излишне и не улучшает управление.

В окончательном варианте диапазон значений каждой из входных переменных был разбит на 7 под–диапазонов, а выходной переменной разбит на 9 под–диапазонов, которые описывались пересекающимися на уровне значения 0.5, треугольными или трапециевидными функциями принадлежности. Треуголь-

ным и трапециевидным функциям было отдано предпочтение перед степенными или синусоидальными функциями, из-за скорости и простоты выполнения арифметических расчётов, с учётом будущего применения на аппаратной платформе IP–маршрутизаторов. Центральная функция принадлежности была выбрана в форме трапеции, чтобы создать область стабильности, где не производилось бы регулирование. Боковые функции трапециевидной формы переменных были продлены за границы диапазона для того, чтобы при нечётком выводе выходные значения могли достигать крайних значений [67].

Форму треугольных функций принадлежностей можно задать тремя значениями: крайнее левое значение соответствующее степени принадлежности 0, центральное значение — степень принадлежности 1 и правое значение — степень принадлежности 0. Трапециевидные функции принадлежности задаются четырьмя значениями: левое значение соответствующее степени принадлежности 0, два центральных значения — степень принадлежности 1 и крайнее правое значение. Задающие форму функций принадлежности значения (контрольные точки) для входных и выходных переменных приведены в таблицах 2.1 и 2.2 соответственно, где используются следующие условные обозначения для нечётких множеств:

- Z — zero (ноль);
- N — negative (отрицательный);
- P — positive (положительный);
- S — small (малый);
- B — big (большой);
- H — huge (огромный);
- T — tiny (крошечный).

Таблица 2.1 — Контрольные точки функций принадлежности входных переменных

Нечёткое множество	Форма функции	Значения
nh	трапеция	(-1.5,-1.0,-0.75,-0.5)
nb	треугольник	(-0.75,-0.5,-0.25)
ns	треугольник	(-0.5,-0.25,-0.05)
z	трапеция	(-0.25,-0.05,0.05,0.25)
ps	треугольник	(0.05,0.25,0.5)
pb	треугольник	(0.25,0.5,0.75)
ph	трапеция	(0.5,0.75,1.0,1.5)

Таблица 2.2 — Контрольные точки функций принадлежности выходной переменной

Нечёткое множество	Форма функции	Значения
nh	трапеция	(-1.5,-1.3,-0.8,-0.6)
nb	треугольник	(-0.8,-0.6,-0.4)
ns	треугольник	(-0.6,-0.4,-0.2)
nt	треугольник	(-0.4,-0.2,0.0)
z	треугольник	(-0.2,0.0,0.2)
t	треугольник	(0.0,0.2,0.4)
s	треугольник	(0.2,0.4,0.6)
b	треугольник	(0.4,0.6,0.8)
h	трапеция	(0.6,0.8,1.3,1.5)

Значения двух входных переменных, описанные с помощью 7 нечётких величин каждая, дают набор из 49 правил, отражающий все комбинации значений входных переменных. Набор правил использующийся для нечёткого вывода показан в табл. 2.3.

Таблица 2.3 — Набор правил

		Значения переменной rate						
		NH	NB	NS	Z	S	B	H
Значения переменной q_{error}	NH	NH	NH	NH	NH	NB	NB	NB
	NB	NH	NH	NB	NB	NS	NS	NS
	NS	NH	NB	NS	NS	NT	NT	NT
	Z	NB	NB	NT	Z	Z	T	T
	PS	NS	NS	NT	T	T	S	S
	PB	NT	NT	Z	S	S	B	B
	PH	Z	Z	T	B	B	H	H

В данной таблице отражены значения выходной нечёткой величины в зависимости от двух входных нечётких величин, т.е. если значение q_{error} равно «Zero» (ноль), а значение $rate$ равно «Big» (большое), то выходное значение равно «Tiny» (крошечное). Другими словами, если текущее значение длины очереди соответствует заданному значению, но текущая интенсивность поступающего трафика сильно превышает интенсивность обслуживания, то необходимо немного повышать вероятность сброса/маркировки пакетов.

На рис. 2.5 показана графическая интерпретация нечёткого вывода в программном обеспечении Xfuzzy [81].

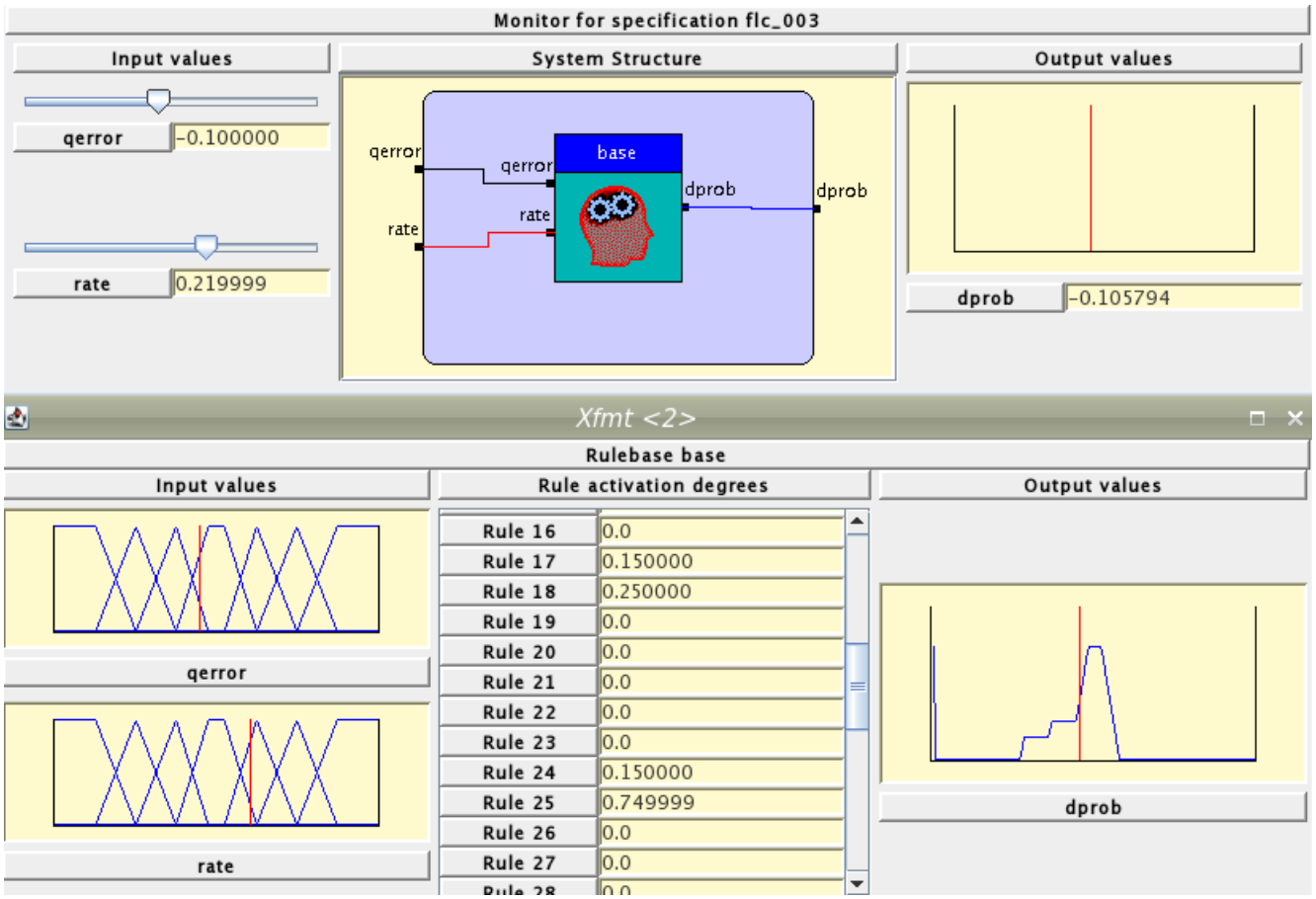


Рисунок 2.5 — Графическая интерпретация нечёткого вывода в программе Xfuzzy

Вид трёхмерной функции зависимости выходной величины приращения вероятности сброса от двух входных переменных $\Delta P_{drop}(q_{error}, rate)$ приведён на рис. 2.6, а исходный код нечёткого регулятора *flc_003.c* на языке программирования Си и заголовочный файл *flc_003.h* доступны в репозитории [75].

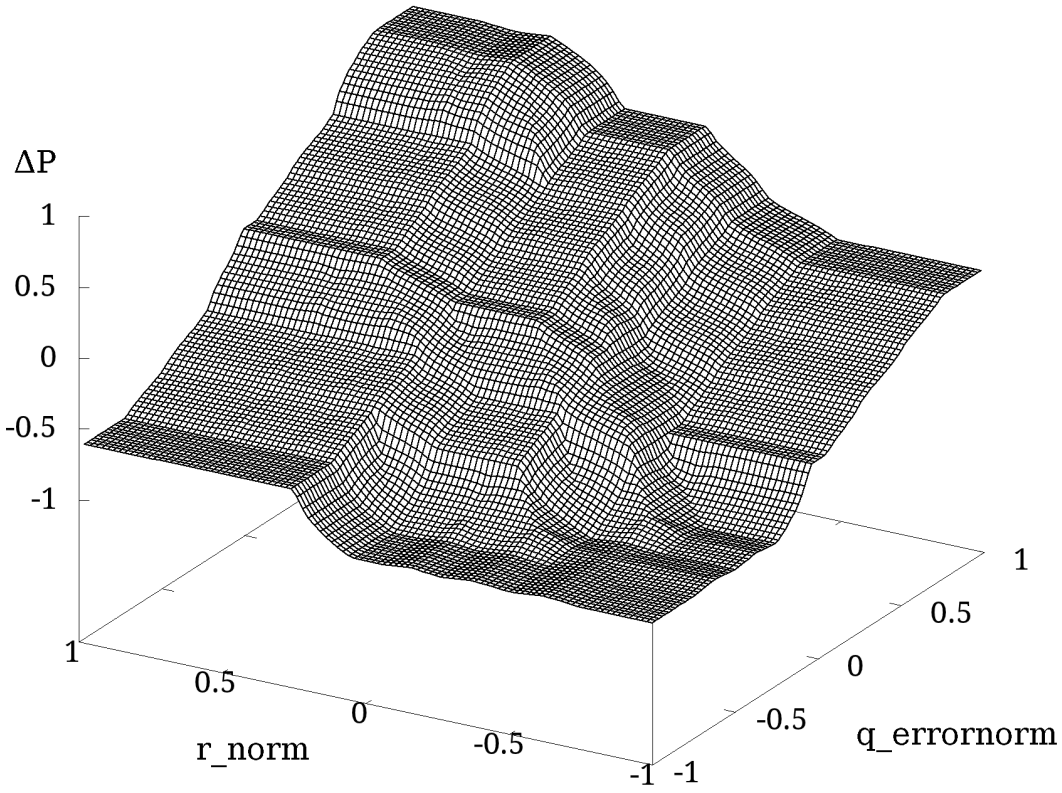


Рисунок 2.6 — Функция активного управления очередью

Рассчитанное с помощью функции управления значение приращения сброса пакета $\Delta P^i \in [-1, 1]$ используется модулем «Актуатор» для вычисления вероятности сброса пакета по формуле:

$$P^{i+1} = P^i + \Delta P^i \cdot P_{\max}, \quad i \geqslant 1, \tag{2.12}$$

где $0 < P_{\max} < 1$ максимальное значение приращения вероятности сброса на любом интервале наблюдения. Таким образом, на интервале Δt^i модуль «Актуатор» сбрасывает пакеты с вероятностью P^i при поступлении пакета в исходящий буфер маршрутизатора.

2.6 Имитационное моделирование FLC

Разработанный нечёткий регулятор был реализован в виде программного модуля для сетевого симулятора NS–2. В ходе ряда экспериментов имитационного моделирования, опытным путём, были оптимизированы формы функций принадлежностей и наборы правил нечёткого вывода. Модуль «Монитор» проводил измерения с интервалом $\Delta t = 6$ мс. Максимальное изменение вероятности сброса P_{max} за время Δt было принято $8 \cdot 10^{-5}$. При данных параметрах нечёткий регулятор давал наименьшее средне-квадратичное отклонение от заданной длины очереди. Параметры передаваемые в программный модуль и начальные значения переменных приведены в таблицах 2.4 и 2.5 соответственно.

Таблица 2.4 — Параметры передаваемые в программный модуль FLC

Название параметра	Имя переменной	Значение параметра
Эталонное значение длины очереди	$qref$	300 пакетов
Максимальное изменение вероятности за интервал измерений	d_scale	$8 \cdot 10^{-5}$
Длительность интервала измерений	$sampling$	0.006 секунд
Скорость передачи данных в канале	p_rate_0	50 Мбит/с
Максимальный размер очереди	q_lim	500 пакетов

Таблица 2.5 — Начальные значения переменных

Название параметра	Имя переменной	Значение параметра
Счётчик принятых байтов	$count\_bytes$	0
Счётчик принятых пакетов	$count\_p$	0
Время последнего измерения	$last\_call\_sampling$	0
Вероятность сброса/-маркировки пакета	$prob$	0

Упрощённая блок–схема работы алгоритма программного модуля FLC приведена на рис. 2.7. При поступлении нового пакета на вход в очередь, в блоке 1 вычисляются общее количество байт и количество пакетов полученных на текущем интервале измерений, фиксируется текущее время в переменной now и вычисляется время прошедшее с момента последнего измерения dt. В блоке 2 сравнивается dt с длительностью интервала $sampling$, и если время прошедшее с момента последнего измерения меньше установленного интервала $sampling$, то программа переходит к блоку 7, а если интервал измерений уже истёк, то выполнятся блоки 3, 4, 5 и 6 для расчёта новой вероятности сбора, которая будут действовать на следующем интервале. В блоке 3 вычисляются значения ошибки длины очереди q_error и текущая скорость входного потока данных p_rate необходимые для дальнейшего вычисления приращения вероятности сброса d_prob. Перед вычислением в блоке 5 приращения вероятности сброса, входные переменные нормируются в блоке 4. В блоке 5 вызывается функция *flc()* для расчёта приращения вероятности сброса (трёхмерный вид функции был приведён на рис. 2.6). Полученное приращение вероятности сброса в блоке 6 умножается на максимально возможное на интервале измерений приращение вероятности d_sacale и вычисляется новое значение $prob$. Для принятия решения о сбросе пакета либо помещения его в очередь, в блоке 7 генерируется случайное число u с равномерным распределением на интервале $[0, 1]$ и в блоке 8 это случайное сравнивается рассчитанной вероятностью сброса $prob$. Если вероятность $prob$ не попадает в интервал $[0, u]$, то пакет будет помещён в очередь в блоке 9, а если попадает, то будет сброшен (или маркирован в случае использования протокола ECN) в блоке 10. Таким образом происходит обработка при каждом поступлении нового пакета на вход очереди, но вероятность сброса рассчитывается и устанавливается только один раз на каждый интервал измерений $sampling$. Полный код программы *flc2.cpp* на языке программирования C++ доступен в репозитории [75].

Коэффициент использования канала во всех экспериментах был близок к единице. Графики изменения вероятности сброса и длины очереди за время моделирования (для эксперимента при скорости в канале 50 Мбит/с и задержке 5 мс) приведены на рисунке на рис. 2.8 и 2.9 соответственно:

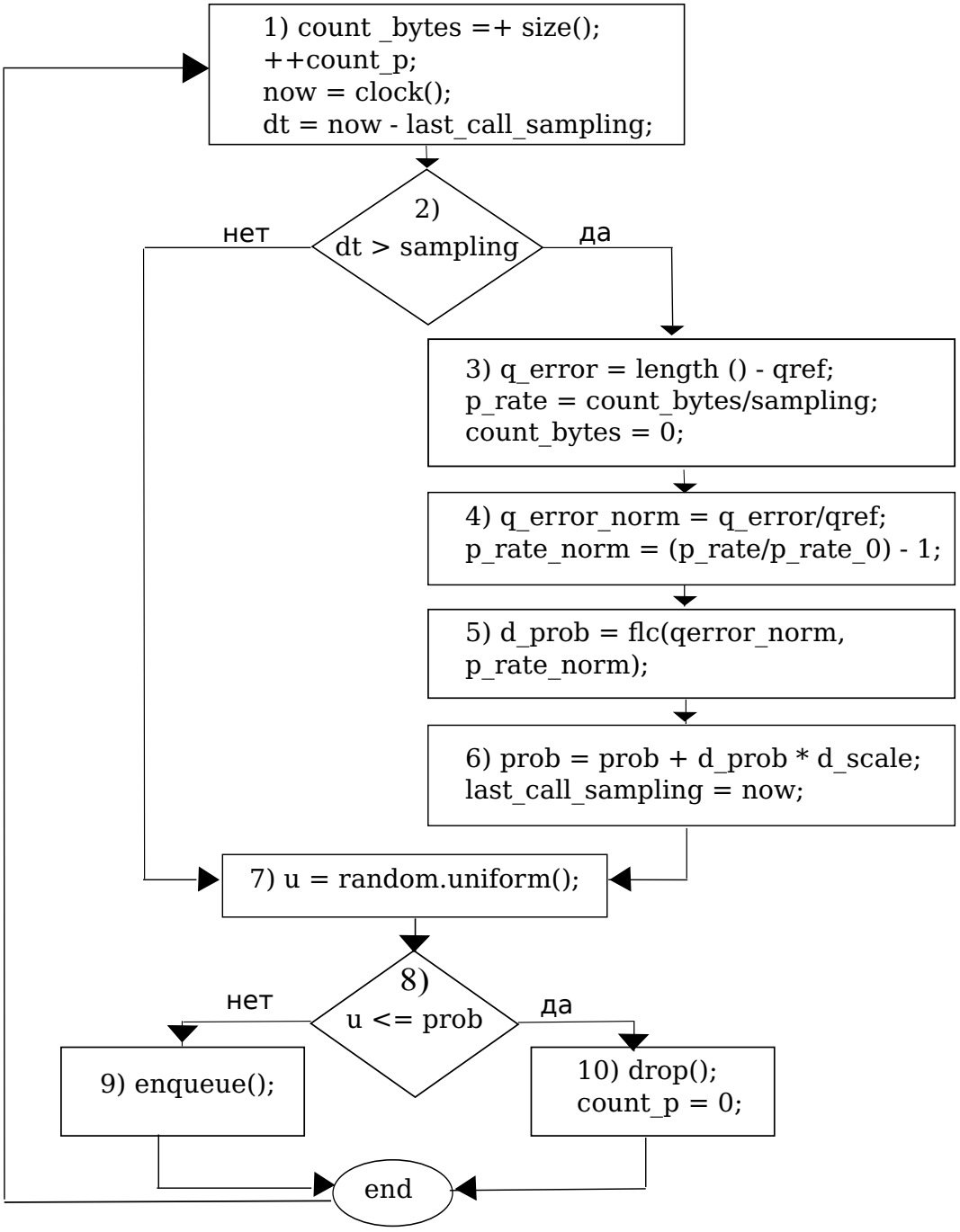


Рисунок 2.7 — Алгоритм программного модуля FLC

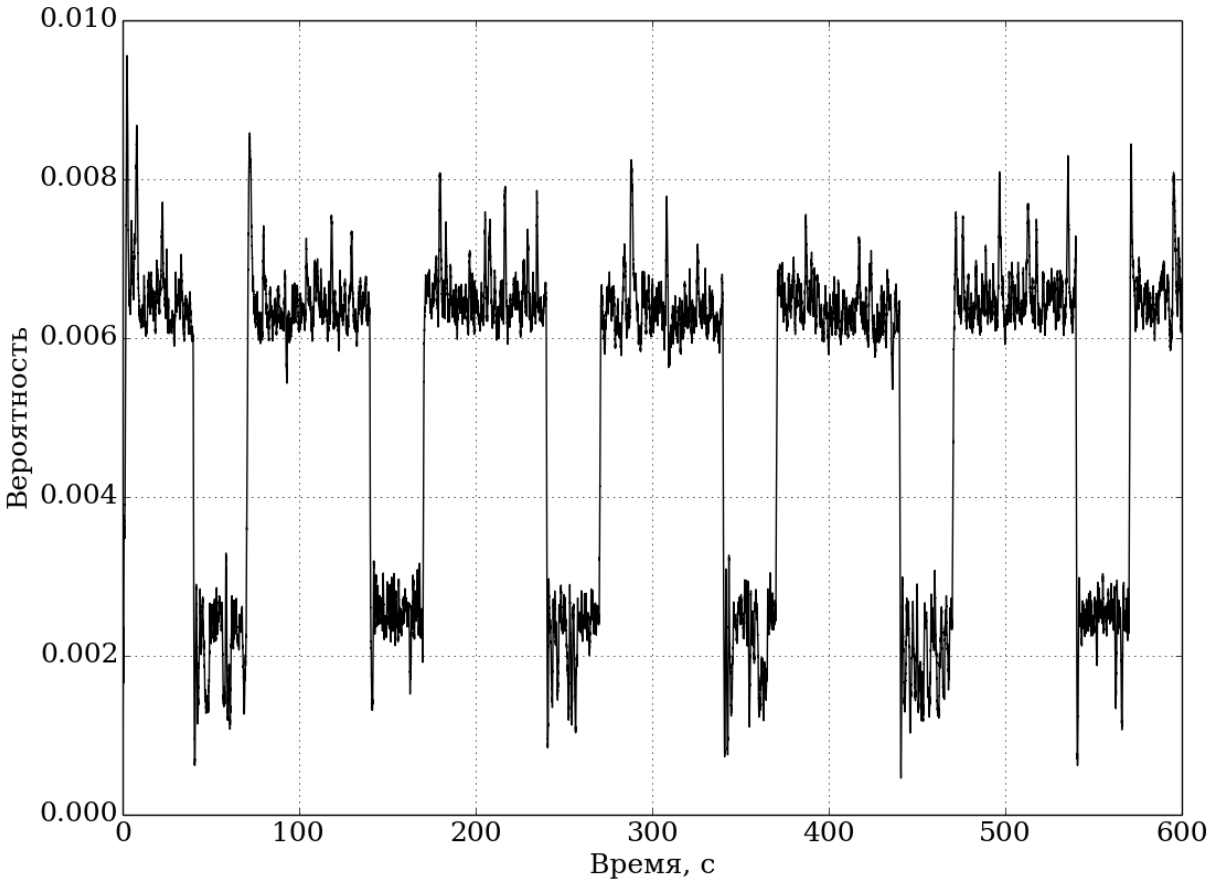


Рисунок 2.8 — Изменение вероятности сброса пакетов при управлении нечётким регулятором

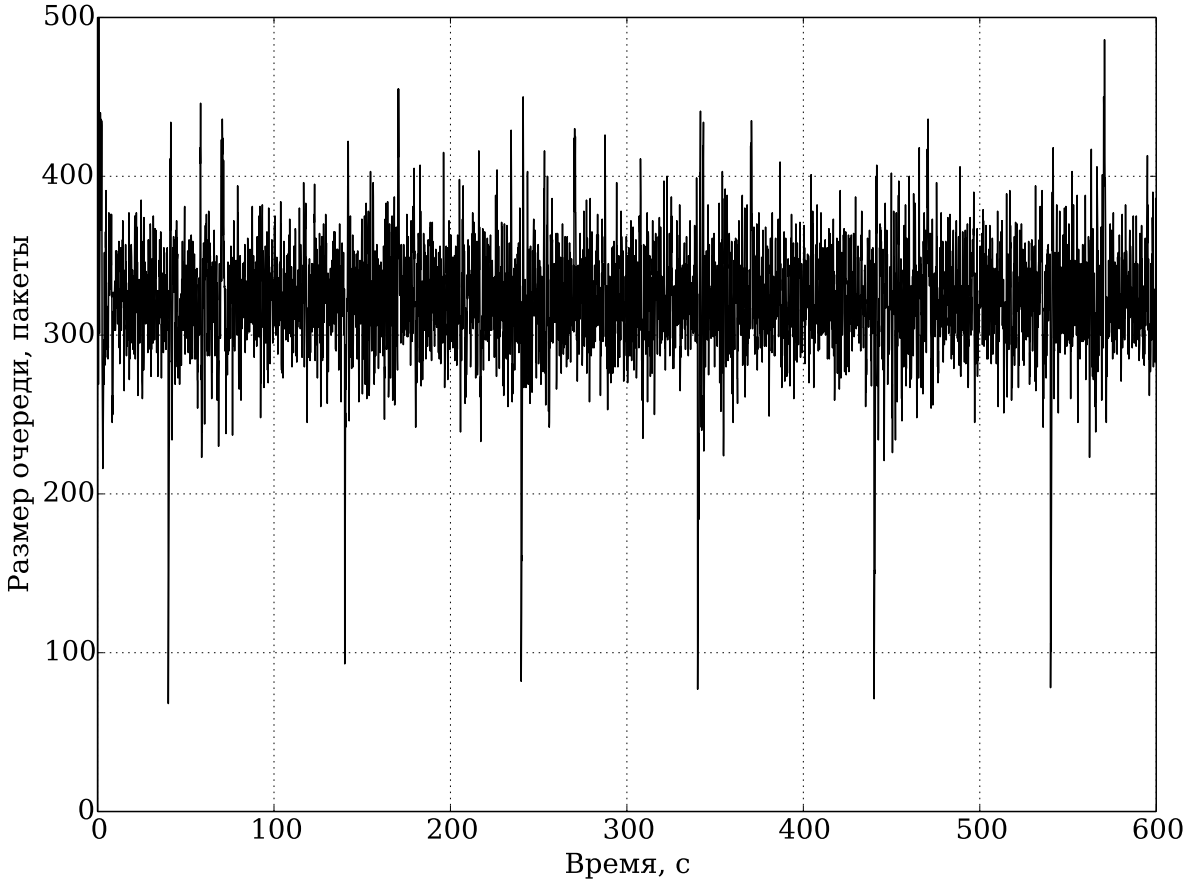


Рисунок 2.9 — Изменение длины очереди при управлении нечётким регулятором

На графиках видно как метод управления очередью в зависимости от интенсивности трафика автоматически подстраивает значения вероятности сброса пакетов для удержания текущей длины очереди около заданного значения — 300 пакетов. Средние значения длины очереди и среднеквадратичного отклонения с доверительным интервалом ±1 пакет (достоверность 95%), в данном эксперименте в пяти последовательных интервалах по 100 секунд, приведены в табл. 2.6.

Таблица 2.6 — Средние значения параметров

Характеристика	Значение
Среднее значение длины очереди	322 пакета
Среднеквадратичное отклонение	33 пакета

По стабильности среднего значения длины очереди можно сделать вывод о стационарности процесса приёма/передачи пакетов в очереди.

2.7 Оценка параметров качества

Для оценки параметров качества при работе механизма управления очередью в программном комплексе NS–2 было проведено моделирование фрагмента сети TCP/IP между двумя маршрутизаторами в момент перегрузки в канале аналогично схеме на рис. 1.9 в разделе 1.4 .

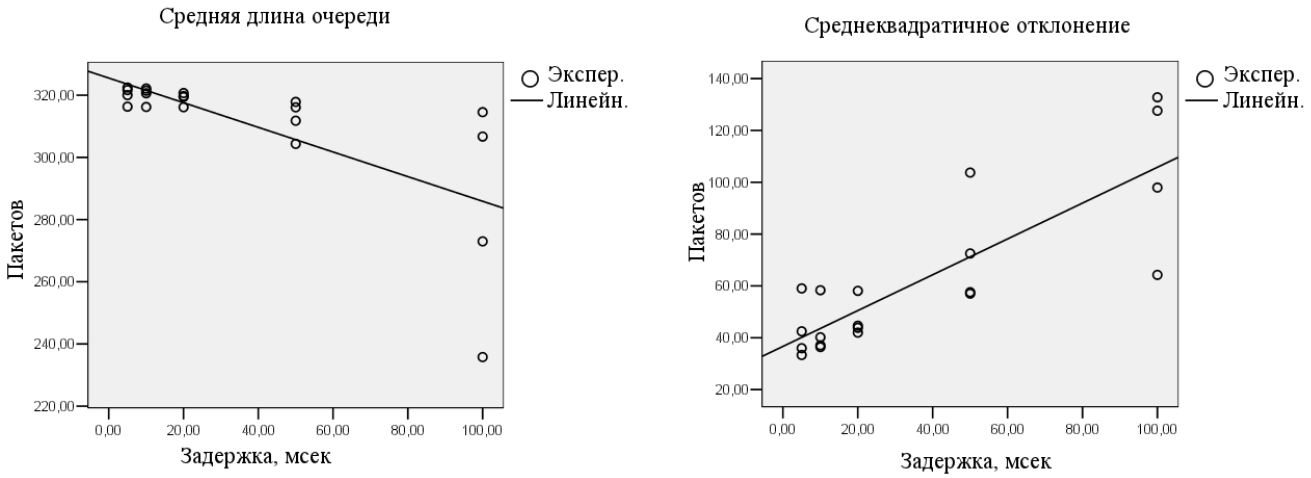
Скорость в канале для разных экспериментов принимала следующие значения: 10, 20, 35, 50 Мбит/с, а задержка в канале соответствовала 5, 10, 20, 50, 100 мс. Т.е. всего было проведено 20 экспериментов длительностью по 600 секунд модельного времени каждый. Через перегруженный канал передавался пакетами по 1000 байт мультисервисный трафик 3-х типов:

– продолжительные по времени TCP сессии (создавались 100 одновременными FTP приложениями);
– короткие по времени TCP сессии (создавались HTTP приложениями с интенсивностью 50 новых соединений в секунду);
– неуправляемый UDP трафик с постоянной скоростью 128 кбит/с.

Использовалась реализация NewReno [82] протокола TCP с включённой функцией ECN (Explicit Congestion Notification — явное уведомление о пере-

грузке, RFC –3168), для сигнализации маршрутизатором транспортному уровню о возможной перегрузке в канале. В случае UDP трафика пакеты случайным образом сбрасывались в момент перегрузки. FTP источники в канале до маршрутизатора имели различное время задержки, равномерно распределённое на интервале от 1 до 9 мс. В начальный момент времени все 100 FTP источников одновременно начинали передачу. Далее для моделирования динамики трафика использовался следующий сценарий, который повторялся 6 раз каждые 100 секунд: в момент времени 40 секунд — 50 FTP источников останавливают передачу, а в момент времени 70 секунд — снова возобновляют передачу.

Для определения зависимостей параметров работы предложенного метода обслуживания от параметров канала передачи данных был проведён регрессионный анализ. Найденные интересующие нас зависимости приведены на рис. 2.10 и 2.11.

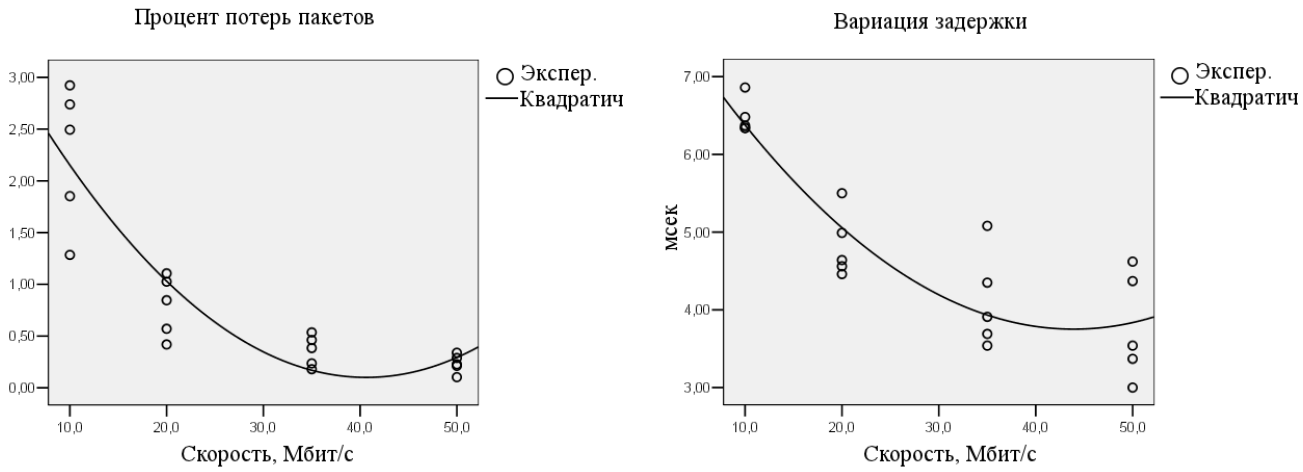


а) б)

Рисунок 2.10 — Зависимость средней длины очереди а) и среднеквадратичного отклонения б) от задержки в канале (мс)

На рис. 2.10а показана зависимость среднего значения длины очереди от задержки в канале. При увеличении задержки в канале среднее значение длины очереди уменьшается. При этом среднеквадратичное отклонение (СКО) от среднего значения растёт по линейному закону (рис. 2.10б). То есть при малых значениях задержки длина очереди стабильнее. Наименьшее значение СКО в экспериментах получено при задержке 5 мс. Данное поведение механизма управления очередью обусловлено тем, что при больших задержках TCP передатчику приходиться дольше дожидаться уведомления о перегрузке и инфор-

мация приходит с опозданием, что в свою очередь ухудшает качество управления очередью. Общие потери пакетов зависят от доступной скорости в канале (рис. 2.11а) и уменьшаются по квадратичному закону с ростом полосы пропускания. Действительно, при большей скорости в канале — меньше время обслуживания пакетов в очереди, соответственно меньше вероятность переполнения и меньше потерь пакетов. В основном потери пакетов приходятся на неуправляемый трафик UDP. Потери длительных TCP соединений минимальны и для FTP трафика в худшем случае не превышали 0,2% благодаря применению маркировки ECN. Значение вариации задержек UDP пакетов также уменьшается по квадратичному закону с ростом скорости в канале (рис. 2.11б), что также показывает увеличение качества работы метода управления при сокращении времени обслуживания, при одинаковом количестве TCP соединений.



а) б)

Рисунок 2.11 — Зависимости параметров качества от скорости в канале (Мбит/с): а) процента потерь пакетов P_{loss}; б) вариации задержек UDP пакетов (мс).

Имитационное моделирование работы разработанного метода обслуживания FLC в программном комплексе NS–2 в широком диапазоне параметров канала, и анализ результатов выявил следующие зависимости:

- стабильность длины очереди линейно возрастает с сокращением задержки в канале;
- процент потерянных пакетов и среднее значение вариации задержек пакетов уменьшаются с ростом полосы пропускания по квадратичному закону.

В дальнейшем, реализация разработанного метода обработки трафика в очередях на базе Linux–маршрутизатора с открытым исходным кодом, даст возможность проверить работу метода на реальной сети связи.

Выводы

1. Метод обработки трафика в очередях на основе нечёткой логики FLC может эффективно удерживать длину очереди около заданного значения в условиях сложного трафика с нелинейной динамикой.
2. Работа метода FLC в сочетании с явным уведомлением о перегрузке позволяет минимизировать потери длительных TCP соединений.
3. При использовании разработанного метода стабильность длины очереди линейно возрастает с сокращением задержки в канале.
4. При применении разработанного метода процент потерянных пакетов и среднее значение вариации задержек пакетов уменьшаются с ростом полосы пропускания по квадратичному закону.

Глава 3. Разработка математической модели процесса обслуживания пакетов в маршрутизаторе с управлением на базе нечёткой логики

3.1 Жидкостная модель

Для построения математической модели используем жидкостную модель [4] TCP–потока версии Reno [83], которая описывается с помощью системы дифференциальных уравнений (ДУ) (3.1):

$$\begin{cases} \frac{dW(t)}{dt} = \frac{1}{R} - \frac{W(t) \cdot W(t-R)}{2R} \cdot P_{drop}(t-R) \\ \frac{dQ(t)}{dt} = \frac{N}{R} \cdot W(t) \cdot (1 - P_{drop}(t)) - C, \end{cases} \tag{3.1}$$

где t — текущее время, $W(t)$ — размер TCP окна, $Q(t)$ — длина очереди, R — задержка «туда-обратно» (RTT), C — скорость в канале, N — количество TCP сессий, $P_{drop}(t)$ — вероятность сброса пакета. Первое уравнение описывает адаптивное поведение размера TCP–окна $W(t)$, когда при каждом получении пакета с подтверждением о доставке, размер окна аддитивно увеличивается со скоростью $1/R$ пакетов в секунду, и мультипликативно снижается, уменьшая вдвое размер окна при потере пакета. Второе уравнение описывает изменения длины очереди, которая опустошается со скоростью передачи пакетов из очереди C и наполняется со скорость отправки данных TCP–передатчиком $W(t)/R$ пакетов в секунду. В уравнении изменения значения длины очереди дополнительно учтён сброс пакетов с вероятностью P_{drop}, вследствие работы алгоритма активного управления очередью, аналогично работе [84]. Данная система решается численно методом Рунге-Кутта 4-го порядка для трёх различных методов управления длиной очереди: Tail Drop, RED и FLC. На каждой итерации численного метода в систему подставляется значение вероятности сброса пакета $P_{drop}(t)$. Для варианта Tail Drop вероятность сброса становиться равной единице только в момент переполнения очереди, а при использовании RED вероятность сброса начинает расти до момента перегрузки при росте средневзвешенной длины очереди и рассчитывается по формуле 1.4. Для расчёта вероятности сброса пакета при использовании метода FLC применялся тот же программный модуль, который был использован в главе 2.6 для имитационного моделирования

нечёткого регулятора в NS-2, скомпилированный в виде динамически подключаемой библиотеки. Исходный код программы *tcp_flc2.py* для расчёта дифференциальных уравнений доступен в репозитории [75]. Исходные параметры для решения системы дифференциальных уравнений приведены в таблице 3.1:

Таблица 3.1 — Исходные параметры для моделирования

Параметр	Значение
Максимальный размер очереди q_{max}	50 пакетов
Заданная длина очереди Q_{ref}	25 пакетов
Скорость передачи данных в канале	100 пакетов/с
Задержка RTT	10 мс

Результаты численного решения ДУ для одного TCP потока в сравнении с результатами имитационного моделирования в NS–2 при тех же исходных параметрах приведены для методов Tail Drop, RED и FLC на рисунках 3.1, 3.2 и 3.3 соответственно.

На графиках моделирования очереди работающей по принципу Tail Drop видно, как длина очереди достигает максимального значения в 50 пакетов, после чего происходит сброс новых пакетов, и TCP-передатчик вдвое снижает размер TCP окна [85–87], после чего количество пакетов в очереди сокращается. Дальнейшее увеличение TCP окна приводит к новому переполнению очереди и описанный сценарий снова повторяется. Графики полученные при решении ДУ имеют одинаковый характер с графиками полученными при имитационном моделировании за исключением начального периода. На графике полученном на модели в NS-2 в начальный момент времени, мы видим быстрый рост TCP окна, соответствующий фазе «slow start (медленный старт)» протокола TCP, когда рост окна перегрузки происходит по экспоненциальному закону. В фазу «congestion avoidance (предотвращение перегрузки)», с линейным ростом размера окна, протокол TCP переходит после снижения окна перегрузки вследствие потери пакетов, минуя фазу «медленный старт», что соответствует алгоритму «fast recovery (быстрое восстановление)» реализации Reno протокола TCP. В жидкостной модели потока TCP в дифференциальных уравнения фаза «медленный старт» не учитывается, и система сразу функционирует в фазе «предотвращение перегрузки», поэтому в начальный момент времени характер графиков полученных решением ДУ и на имитационной модели NS-2 отличается.

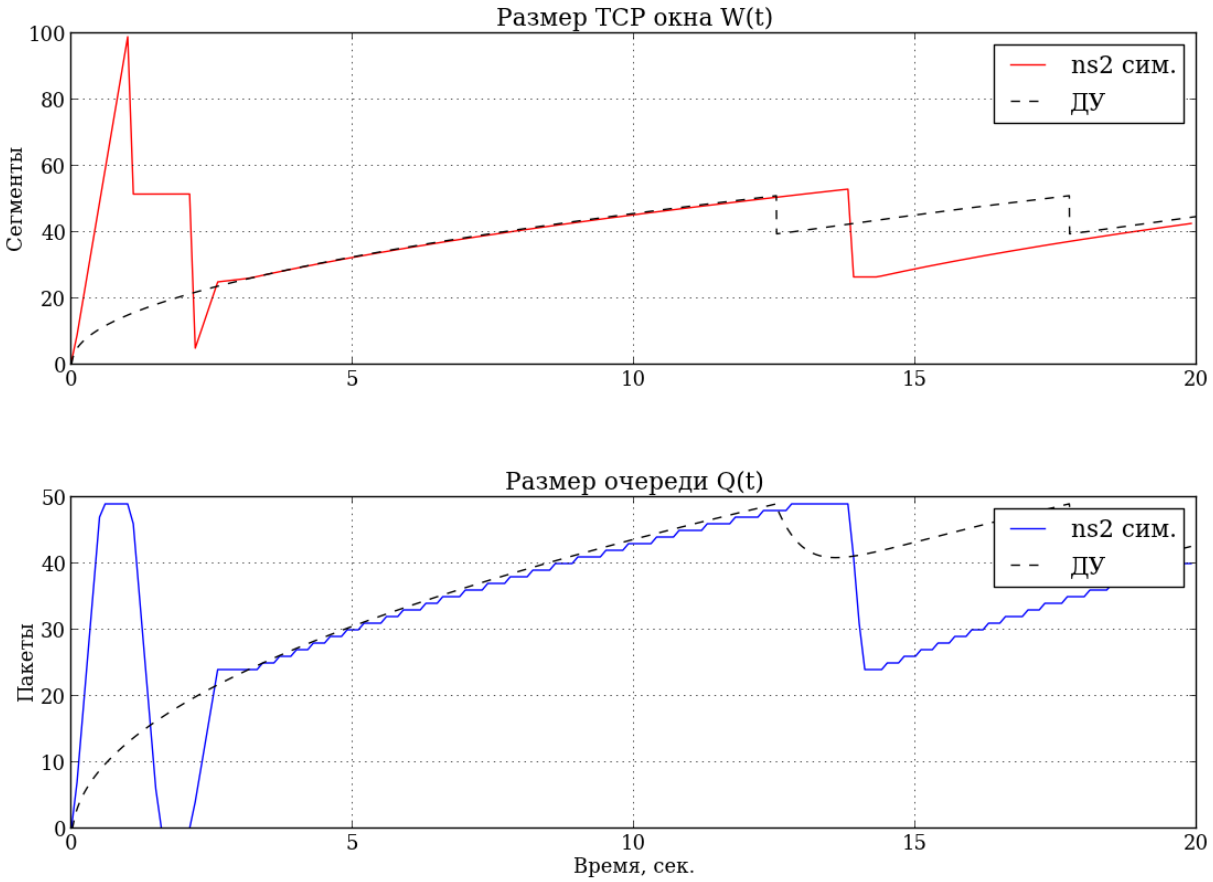


Рисунок 3.1 — Сравнение размеров TCP окна и размера очереди при моделировании в NS-2 (ns2 сим.) и при решении дифференциальных уравнений (ДУ) для метода Tail Drop

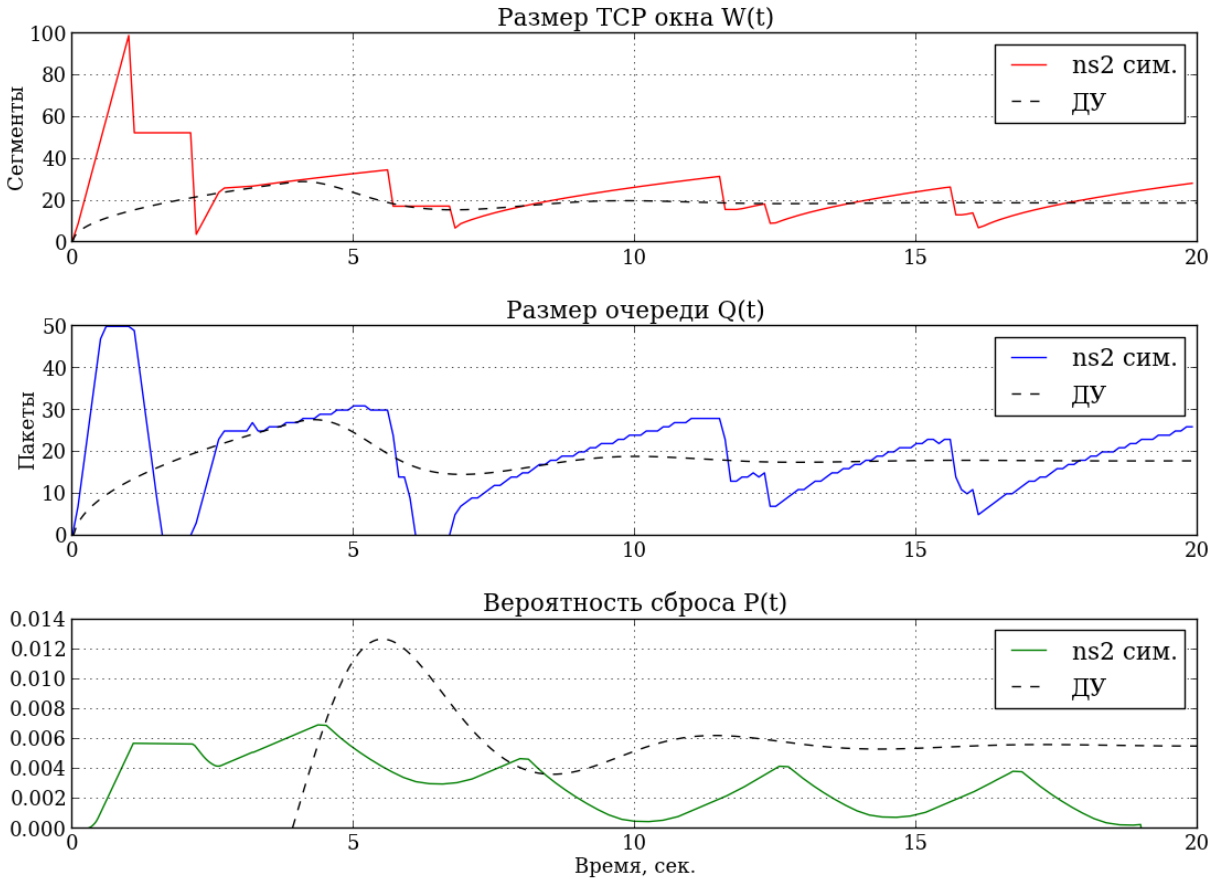


Рисунок 3.2 — Сравнение размеров TCP окна, размера очереди и вероятности сброса при моделировании в NS-2 (ns2 сим.) и при решении дифференциальных уравнений (ДУ) для метода RED

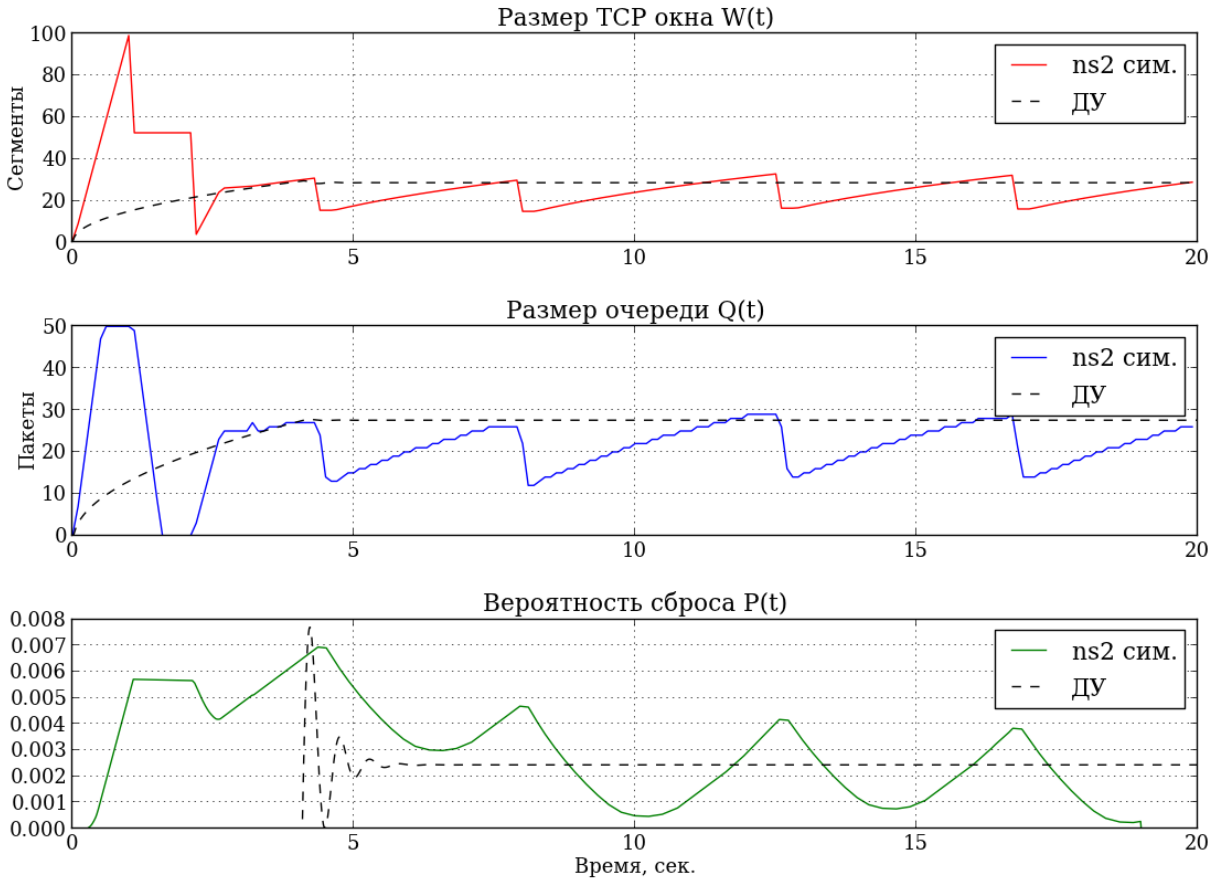


Рисунок 3.3 — Сравнение размеров TCP окна, размера очереди и вероятности сброса при моделировании в NS-2 (ns2 сим.) и при решении дифференциальных уравнений (ДУ) для метода FLC

При моделировании метода RED переполнения очереди не происходит, так как пакеты из очереди начинают сбрасываться до момента перегрузки, заставляя TCP передатчик снизить размер TCP–окна, предотвращая тем самым перегрузку. Вероятность сброса пакетов рассчитывалась по формуле 1.4 с параметрами приведёнными в таблице 3.2:

Таблица 3.2 — Исходные параметры для моделирования метода RED

Параметр	Значение
Максимальный порог max_{th}	50 пакетов
Минимальный порог min_{th}	16 пакетов
Вероятность max_p	0.1
Весовой коэффициент w	0.002

Таким образом, при достижении усреднённого значения длины очереди (по формуле 1.1) равного минимальному порогу 16 пакетов, то вероятность сброса начинает линейно расти от 0 до значения 0.1. Если средняя длина очереди достигнет верхнего порога 50 пакетов, то вероятность сброса скачком увеличится до 1.

На графиках моделирования нечёткого регулятора FLC видно, что при достижении длины очереди значения равного заданному $Q_{ref} = 25$, нечёткий регулятор автоматически увеличивает вероятность сброса и стабилизирует длину очереди около значения Q_{ref} не допуская перегрузки. При расчёте ДУ для метода FLC вероятность сброса после короткого переходного процесса автоматически приходит к стабильному значению около 0,0025. При использовании метода RED вероятность сброса растёт медленнее и переходный процесс длится дольше.

Сравнение полученных результатов решения дифференциальных уравнений с результатами полученными на имитационной модели даёт возможность говорить об адекватности построенной жидкостной модели TCP соединения при прохождении через маршрутизатор с активным управлением очередью.

3.2 Гистерезисная модель с пороговым управлением

3.2.1 Дискретизация параметров

Рассмотрим изображённую на рис. 3.4 систему массового обслуживания (СМО) [88–90]. На систему с буферным накопителем ёмкостью B, нижним порогом L, верхним порогом H и эталонным значением длины очереди Q_{ref} поступает поток заявок с распределением Пуассона с интенсивностью $\lambda(s, q, r)$, зависящей от состояния системы [24, 91, 92]. Заявки обслуживаются в порядке поступления по экспоненциальному закону с интенсивностью μ.

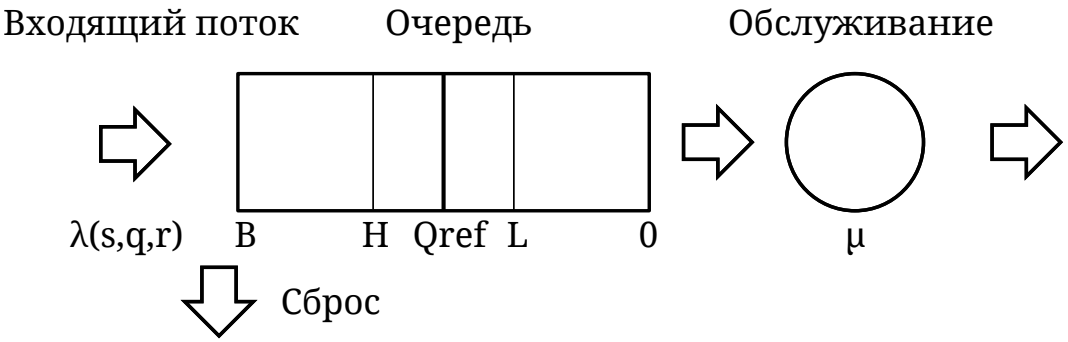


Рисунок 3.4 — СМО с активным пороговым управлением очередью

Построим математическую модель в виде СМО с гистерезисным управлением нагрузкой аналогично [14, 15]. Проведем дискретизацию параметров функции управления очередью, введя параметр $r \in \{0, 1, 2, 3, 4\}$, характеризующий уровень интенсивности нагрузки на систему, и параметр $s \in \{0, 1, 2, 3, 4\}$ статуса перегрузки, который определяет уровень загрузки системы, т.е. степень наполненности буфера. При этом состояниям с одинаковым уровнем интенсивности нагрузки r может соответствовать разный статус перегрузки s.

В табл. 3.3 показано соответствие значений параметра r диапазонам изменения интенсивности поступления пакетов r^i_{norm} полученного по формуле (2.10), а в табл. 3.4 — соответствие значения статуса перегрузки.

Для управления интенсивностью предложенной нагрузки в очереди системы вводятся два порога — нижний порог L и верхний порог H таким образом, что выполняется соотношение $L < Q_{\text{ref}} < H$. Пока общее число заявок в системе не превышает значение $H + 1$, система функционирует в нормальном режи-

Таблица 3.3 — Уровни интенсивности нагрузки

Уровень нагрузки	Значение параметра r	Диапазон параметра r^i_{norm}
Малая	0	[-1; -0,6)
Средняя	1	[-0,6; -0,2)
Норма	2	[-0,2; 0,2)
Высокая	3	[0,2; 0,6)
Перегрузка	4	[0,6; 1]

Таблица 3.4 — Уровни статуса перегрузки

Статус перегрузки	Значение параметра s
Малая нагрузка	0
Нормальная нагрузка	1
Начало перегрузки	2
Перегрузка	3
Сброс нагрузки	4

ме (малая и нормальная нагрузка). Если число заявок в системе превысило значение $H + 1$, система переходит в режим перегрузки (начало перегрузки и перегрузка). Система остается в режиме перегрузки до тех пор, пока число заявок в очереди q не достигнет значения $Q_{\text{ref}} - 1$ или B. При достижении длиной очереди значения $Q_{\text{ref}} - 1$ система возвращается в нормальный режим функционирования, а при достижении длиной очереди значения B система переходит в режим сброса нагрузки, в котором остается до тех пор, пока длина очереди превышает порог H, и возвращается в режим перегрузки, когда число заявок в очереди становится равным H.

Вид функции $\lambda(s, q, r)$ схематично изображен на рис. 3.5. Здесь сплошными линями показаны значения функции интенсивности потока и пунктирными стрелками направления переходов между множествами состояний системы.

Множество состояний СМО представим в виде

$$X = X_0 \cup X_1 \cup X_2 \cup X_3 \cup X_4, \tag{3.2}$$

где непересекающиеся множества X_r описывают состояния, соответствующие уровням r интенсивности нагрузки на СМО: X_0 — множество состояний малой нагрузки, X_1 — множество состояний средней нагрузки, X_2 — множество состояний нормальной нагрузки, X_3 — множество состояний высокой нагрузки и X_4 — множество состояний перегрузки.

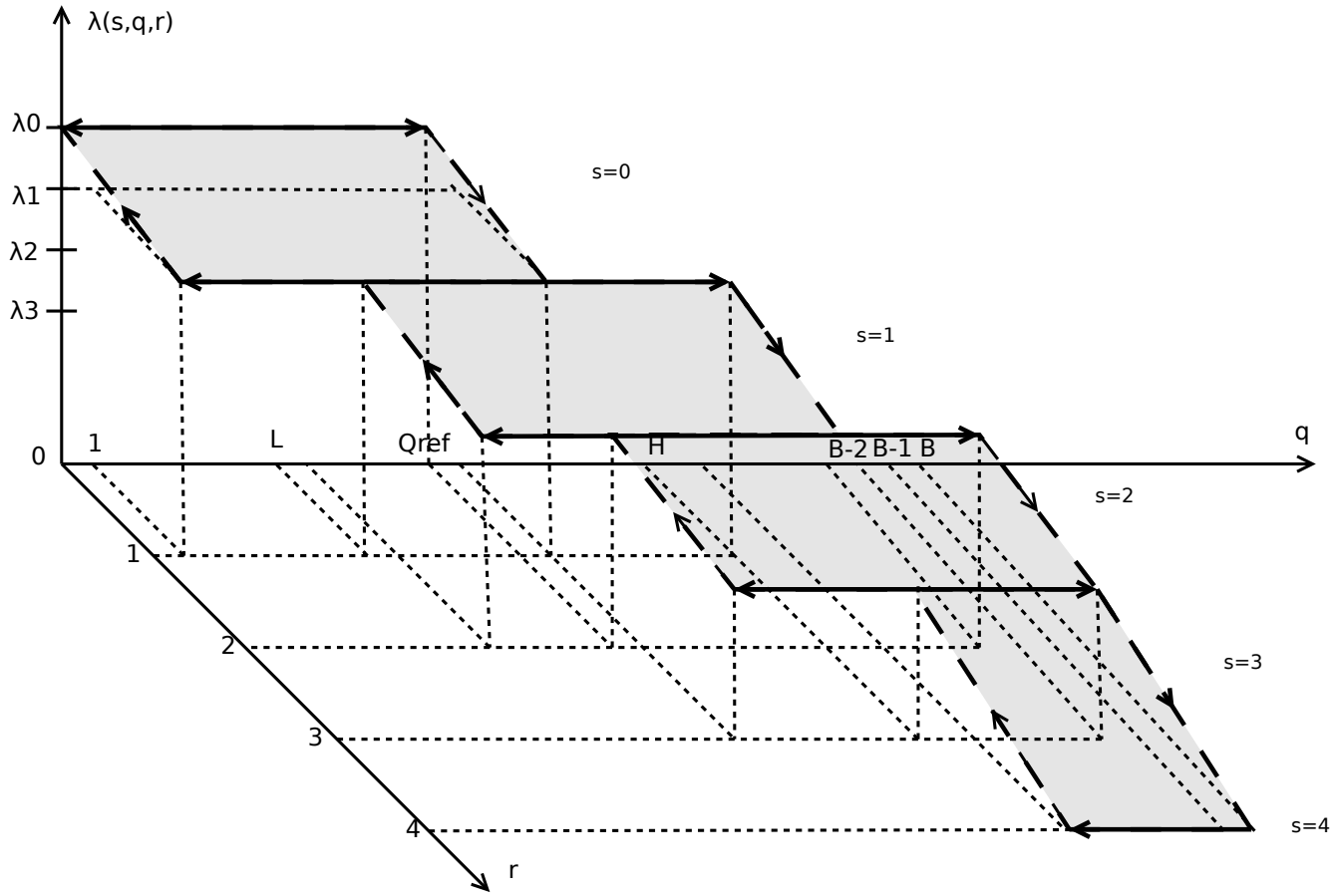


Рисунок 3.5 — Гистерезисное управление нагрузкой в СМО с активным управлением длиной очереди

Множества уровней r интенсивности нагрузки, в свою очередь, также могут быть представлены в виде объединения непересекающихся множеств по следующей формуле:

$$X_r = \begin{cases} X_{0,0}, \\ X_{s-1,r} \bigcup X_{s,r}, \quad s = r, \quad r = 1, 2, 3 \ , \\ X_{4,4,} \end{cases} \tag{3.3}$$

где множества $X_{s,r}$ имеют следующий вид:

$$
\begin{aligned}
X_{0,0} &= \{(s,q,r):\ s=0,\ r=0,\ 0 \leqslant q \leqslant Q_{\text{ref}}\}, \\
X_{0,1} &= \{(s,q,r):\ s=0,\ r=1,\ 1 \leqslant q \leqslant\ Q_{\text{ref}}+1\ \}, \\
X_{1,1} &= \{(s,q,r):\ s=1,\ r=1,\ L \leqslant q \leqslant H\}, \\
X_{1,2} &= \{(s,q,r):\ s=1,\ r=2,\ L+1 \leqslant q \leqslant H+1\}, \\
X_{2,2} &= \{(s,q,r):\ s=2,\ r=2,\ Q_{\text{ref}}\ \leqslant q \leqslant\ B-3\}, \\
X_{2,3} &= \{(s,q,r):\ s=2,\ r=3,\ Q_{\text{ref}}+1 \leqslant q \leqslant B-2\}, \\
X_{3,3} &= \{(s,q,r):\ s=3,\ r=3,\ H \leqslant q \leqslant\ B-1\}, \\
X_{4,4} &= \{(s,q,r):\ s=4,\ r=4,\ H+1 \leqslant q \leqslant B\}.
\end{aligned}
\tag{3.4}
$$

Обозначим λ_r, $r = 0,1,2,3,4$, интенсивность поступающего потока на СМО на r-м уровне интенсивности нагрузки, причем $\lambda_0 = \lambda$. Тогда интенсивность $\lambda(s,q,r)$ потока, поступающего на СМО, определяется формулой

$$
\lambda(s,q,r) = \begin{cases} \lambda_0, & (s,q,r) \in X_{0,0}, \\ (1-p_r)\lambda_{r-1}, & (s,q,r) \in X \backslash (X_{0,0} \cup X_{4,4}), \\ 0, & (s,q,r) \in X_{4,4}, \end{cases}
\tag{3.5}
$$

где p_r – вероятность сброса пакетов на r-м уровне.

3.2.2 Система уравнений равновесия

Функционирование построенной в предыдущем разделе СМО с активным управлением очередью описывается марковским процессом (МП) [93] $X(t)$ на множестве X. Фрагмент диаграммы интенсивностей переходов с одной петлёй гистерезиса при $s = 0$ приведён на рис. 3.6. Нетрудно убедиться, что полная диаграмма интенсивностей переходов марковского процесса $X(t)$ имеет вид, показанный на рис. 3.7.

Из диаграммы интенсивностей переходов может быть получена система уравнений равновесия (СУР) марковского процесса $X(t)$ в следующем виде:

$$
\begin{cases}
\lambda_0 p_{0,0,0} = \mu p_{0,1,0} + \mu p_{0,1,1}, \\
(\lambda_0 + \mu) p_{0,q,0} = \lambda_0 p_{0,q-1,0} + \mu p_{0,q+1,0}, \quad q = \overline{1, Q_{\text{ref}} - 1}, \\
(\lambda_0 + \mu) p_{0,Q_{\text{ref}},0} = \lambda_0 p_{0,Q_{\text{ref}}-1,0}, \\
(\lambda_1 + \mu) p_{0,1,1} = \mu p_{0,2,1}, \\
(\lambda_1 + \mu) p_{0,q,1} = \lambda_1 p_{0,q-1,1} + \mu p_{0,q+1,1}, \quad q = \overline{1, L-2},\ q = \overline{L, Q_{\text{ref}}}, \\
(\lambda_1 + \mu) p_{0,L-1,1} = \lambda_1 p_{0,L-2,1} + \mu p_{0,L,1} + \mu p_{1,L,1}, \\
(\lambda_1 + \mu) p_{0,Q_{\text{ref}}+1,1} = \lambda_1 p_{0,Q_{\text{ref}},0} + \lambda_1 p_{0,Q_{\text{ref}},1}, \\
(\lambda_1 + \mu) p_{1,L,1} = \mu p_{1,L+1,1} + \mu p_{1,L+1,2}, \\
(\lambda_1 + \mu) p_{1,q,1} = \lambda_1 p_{1,q-1,1} + \mu p_{1,q+1,1}, \quad q = \overline{L+1, Q_{\text{ref}} + 1},\ q = \overline{Q_{\text{ref}} + 3, H-1}, \\
(\lambda_1 + \mu) p_{1,Q_{\text{ref}}+2,1} = \lambda_1 p_{1,Q_{\text{ref}}+1,1} + \lambda_1 p_{0,Q_{\text{ref}}+1,1} + \mu p_{1,Q_{\text{ref}}+3,1}, \\
(\lambda_1 + \mu) p_{1,H,1} = \lambda_1 p_{1,H-1,1}, \\
(\lambda_2 + \mu) p_{1,L+1,2} = \mu p_{1,L+2,2}, \\
(\lambda_2 + \mu) p_{1,q,2} = \lambda_2 p_{1,q-1,2} + \mu p_{1,q+1,2}, \quad q = \overline{L+2, Q_{\text{ref}} - 2},\ q = \overline{Q_{\text{ref}}, H}, \\
(\lambda_2 + \mu) p_{1,Q_{\text{ref}}-1,2} = \lambda_1 p_{1,Q_{\text{ref}}-2,2} + \mu p_{1,Q_{\text{ref}},2} + \mu p_{2,Q_{\text{ref}},2}, \\
(\lambda_2 + \mu) p_{1,H+1,2} = \lambda_1 p_{1,H,2} + \lambda_1 p_{1,H,1}, \\
(\lambda_2 + \mu) p_{2,Q_{\text{ref}},2} = \mu p_{2,Q_{\text{ref}}+1,2} + \mu p_{2,Q_{\text{ref}}+1,3}, \\
(\lambda_2 + \mu) p_{2,q,2} = \lambda_2 p_{2,q-1,2} + \mu p_{2,q+1,2}, \quad q = \overline{Q_{\text{ref}} + 1, H+1},\ q = \overline{H+3, B-4}, \\
(\lambda_2 + \mu) p_{2,H+2,2} = \lambda_2 p_{2,H+1,2} + \lambda_2 p_{1,H+1,2} + \mu p_{2,H+3,2}, \\
(\lambda_2 + \mu) p_{2,B-3,2} = \lambda_2 p_{2,B-4,2}, \\
(\lambda_3 + \mu) p_{2,Q_{\text{ref}}+1,3} = \mu p_{2,Q_{\text{ref}}+2,3}, \\
(\lambda_3 + \mu) p_{2,q,3} = \lambda_2 p_{2,q-1,3} + \mu p_{2,q+1,3}, \quad q = \overline{Q_{\text{ref}} + 2, H-2},\ q = \overline{H, B-3}, \\
(\lambda_3 + \mu) p_{2,H-1,3} = \lambda_3 p_{2,H-2,3} + \mu p_{2,H,3} + \mu p_{3,H,3}, \\
(\lambda_3 + \mu) p_{2,B-2,3} = \lambda_3 p_{2,B-3,3} + \lambda_2 p_{2,B-3,2}, \\
(\lambda_3 + \mu) p_{3,H,3} = \mu p_{3,H+1,3} + \mu p_{4,H+1,4}, \\
(\lambda_3 + \mu) p_{3,q,3} = \lambda_3 p_{3,q-1,3} + \mu p_{3,q+1,3}, \quad q = \overline{H+1, B-2}, \\
(\lambda_3 + \mu) p_{3,B-1,3} = \lambda_3 p_{3,B-2,3} + \lambda_3 p_{2,B-2,3}, \\
\mu p_{4,B,4} = \lambda_3 p_{3,B-1,3}, \\
p_{4,q-1,4} = p_{4,q,4}, \quad q = \overline{H+1, B}.
\end{cases}
\tag{3.6}
$$

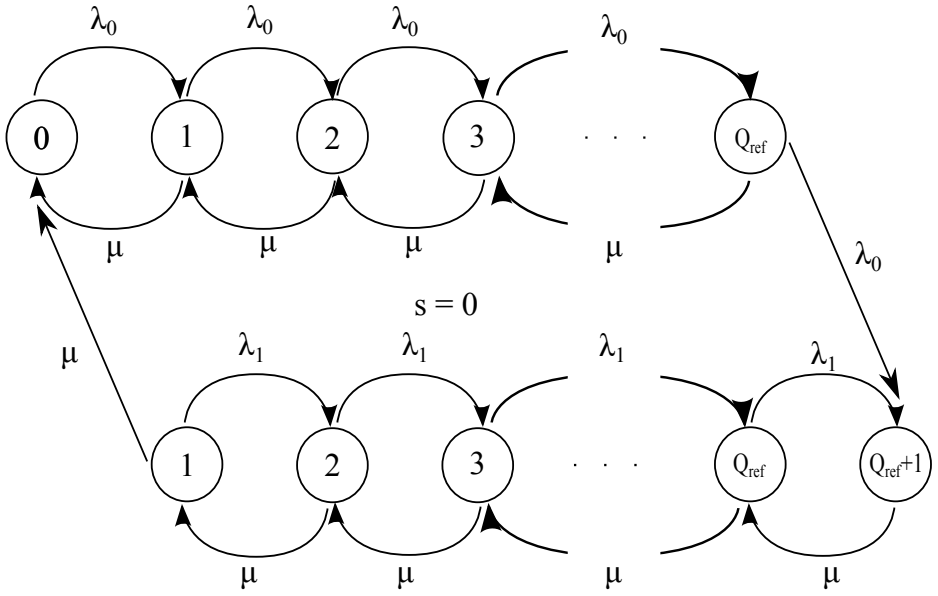


Рисунок 3.6 — Фрагмент диаграммы интенсивностей переходов марковского процесса

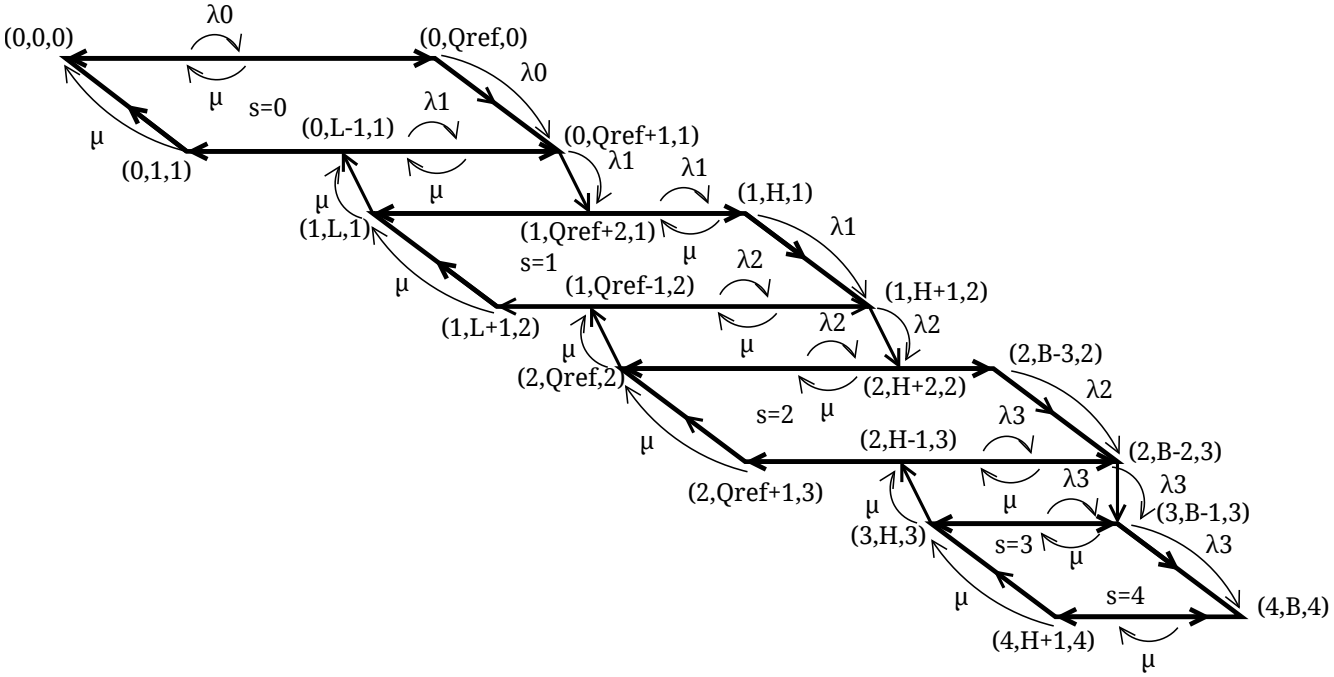


Рисунок 3.7 — Общий вид диаграммы интенсивностей переходов марковского процесса

В левой части уравнений мы выписываем вероятность для каждого состояния системы умноженную на интенсивность перехода из этого состояния, а в правой части соответственно вероятности соседних состояний умноженные на интенсивности перехода в данное состояние. Если в системе существует равновесное состояние, то решение этой системы уравнений даст нам значения стационарных вероятностей при известных интенсивностях переходов. Далее, зная значения стационарных вероятностей состояний, мы сможем найти такие вероятностные характеристики системы, как математическое ожидание длины очереди и дисперсию.

Чтобы сумма вероятностей всех состояний получалась равной 1, мы заменяем последнее уравление в системе условием нормировки:

$$\sum_{k=0}^{Q_{ref}} p_{0,k,0} + \sum_{k=1}^{Q_{ref}+1} p_{0,k,1} + \sum_{k=L}^{H} p_{1,k,1} + \sum_{k=L+1}^{H+1} p_{1,k,2} + \sum_{k=Q_{ref}}^{B-3} p_{2,k,2} + \\ + \sum_{k=Q_{ref}+1}^{B-2} p_{2,k,3} + \sum_{k=H}^{B-1} p_{3,k,3} + \sum_{k=H+1}^{B} p_{4,k,4} = 1, \quad (3.7)$$

Матрица интенсивностей переходов **A** состоит из 64 (8x8) подматриц и в соответствии с множествами $X_{0,0}, X_{0,1}, X_{1,1}, X_{1,2}, X_{2,2}, X_{2,3}, X_{3,3}, X_{4,4}$ имеет вид показанный в табл. 3.5:

Таблица 3.5 — Структура матрицы интенсивностей переходов **A**

	$X_{0,0}$	$X_{0,1}$	$X_{1,1}$	$X_{1,2}$	$X_{2,2}$	$X_{2,3}$	$X_{3,3}$	$X_{4,4}$
$X_{0,0}$	D	U	0	0	0	0	0	0
$X_{0,1}$	L	D	U	0	0	0	0	0
$X_{1,1}$	0	L	D	U	0	0	0	0
$X_{1,2}$	0	0	L	D	U	0	0	0
$X_{2,2}$	0	0	0	L	D	U	0	0
$X_{2,3}$	0	0	0	0	L	D	U	0
$X_{3,3}$	0	0	0	0	0	L	D	U
$X_{4,4}$	0	0	0	0	0	0	L	D

Символом **D** обозначены диагональные матрицы, символом **U** матрицы с одним значением интенсивности поступления пакетов, **L** — матрицы с одним значением интенсивности обслуживания пакетов и **0** — нулевые матрицы.

Система уравнений равновесия в матричном виде с учётом условия нормировки (3.7) принимает следующий вид:

$$\begin{cases} \vec{p} \cdot \mathbf{A} = 0 \\ \vec{p} \cdot 1 = 1 \end{cases}, \tag{3.8}$$

где $\vec{p}$ — вектор стационарных вероятностей, $\mathbf{A}$ — матрица интенсивностей переходов.

Для анализа вероятностных характеристик исследуемой СМО решим систему уравнений равновесия численно с помощью метода LU-разложения [94].

3.2.3 Численный анализ

Обозначим Y множество тех состояний СМО, в которых длина очереди находится в диапазоне $q \in [L, H]$, и представим его в виде

$$Y = Y_0 + Y_1 + Y_2, \tag{3.9}$$

где

$$Y_0 = \{(s,q,r) :\ s = 0,\ r\ = 0,\ L\ \leqslant\ q\ \leqslant\ Q_{\text{ref}}\,\} \cup \\ \cup\,\{(s,q,r) :\ s = 0,\ r\ = 1,\ L\ \leqslant q\ \leqslant\ Q_{\text{ref}} + 1\,\};$$

$$Y_1 = \{(s,q,r) :\ s = 1,\ r\ = 1,\ L\ \leqslant q\ \leqslant\ H\,\} \cup \\ \cup\,\{(s,q,r) :\ s = 1,\ r\ = 2,\ L + 1\ \leqslant q \leqslant H\,\};$$

$$Y_2 = \{(s,q,r) :\ s = 2,\ r\ = 2,\ Q_{\text{ref}}\ \leqslant\ q\ \leqslant\ H\} \cup \\ \cup\,\{(s,q,r) :\ s = 2,\ r\ = 3,\ Q_{\text{ref}}\ + 1\ \leqslant\ q\ \leqslant H\}\,.$$

Необходимо оптимизировать систему с целью достижения максимального значения вероятности $P(Y)$ отклонения длины очереди от эталонного значения в пределах $L \leqslant q \leqslant H$.

Для проведения численного анализа в качестве исходных данных выберем емкость буферного накопителя $B = 50$, значение эталонной длины очереди $Q_{\text{ref}} = 25$, значения порогов $L = 20$ и $H = 30$. Заметим, что при данных значениях мощность множества состояний СМО и, следовательно, размерность системы уравнений равновесия, равна 160. Для заполнения такой большой матрицы интенсивности переходов был написан отдельный программный код. Исходный код программы *sur_states.py* для заполнения матрицы и расчёта стационарных вероятностей, с использованием метода LU-разложения из программной библиотеки линейной алгебры, доступен в репозитории [75].

Подберём методом перебора значения интенсивностей предложенной нагрузки такими, чтобы вероятность $P(Y)$ достигла максимального значения. Для подбора интенсивностей перебирались значения удовлетворяющие условию:

$$\lambda_0 > \lambda_1 > \lambda_2 > \lambda_3 > \lambda_4, \tag{3.10}$$

где λ_n — интенсивность предложенной нагрузки в диапазоне $[0, 2]$. В рассматриваемом нами примере при значениях интенсивностей $\lambda_0 = 1.95$, $\lambda_1 = 1.2$, $\lambda_2 = 0.47$, $\lambda_3 = 0.43$, $\lambda_4 = 0$ и интенсивности обслуживания $\mu = 1$, эта вероятность достигает максимального значения $P(Y) = 0,68$. Стационарные вероятности пребывания системы в подмножествах множества состояний марковского процесса $X(t)$ показаны на рис. 3.8.

На рис. 3.9 показаны рассчитанные для данного численного примера зависимости средней длины очереди, среднеквадратического отклонения (СКО) длины очереди и вероятности $P(Y)$ отклонения длины очереди от эталонного значения в пределах $L \leqslant q \leqslant H$ в зависимости от нагрузки на систему ρ в диапазоне, включающем перегрузки системы $\rho \in [0, 2]$.

Из графиков на рис. 3.9 видно, что с увеличением нагрузки и переходе системы в режим перегрузки $\rho > 1$ средняя длина очереди стремится к заданному эталонному значению длины очереди $Q_{\text{ref}} = 25$, а вероятность отклонения длины очереди от эталонного значения в пределах порогов от $L = 20$ до $H = 30$ с началом перегрузки также достигает максимального значения $P(Y) = 0,68$.

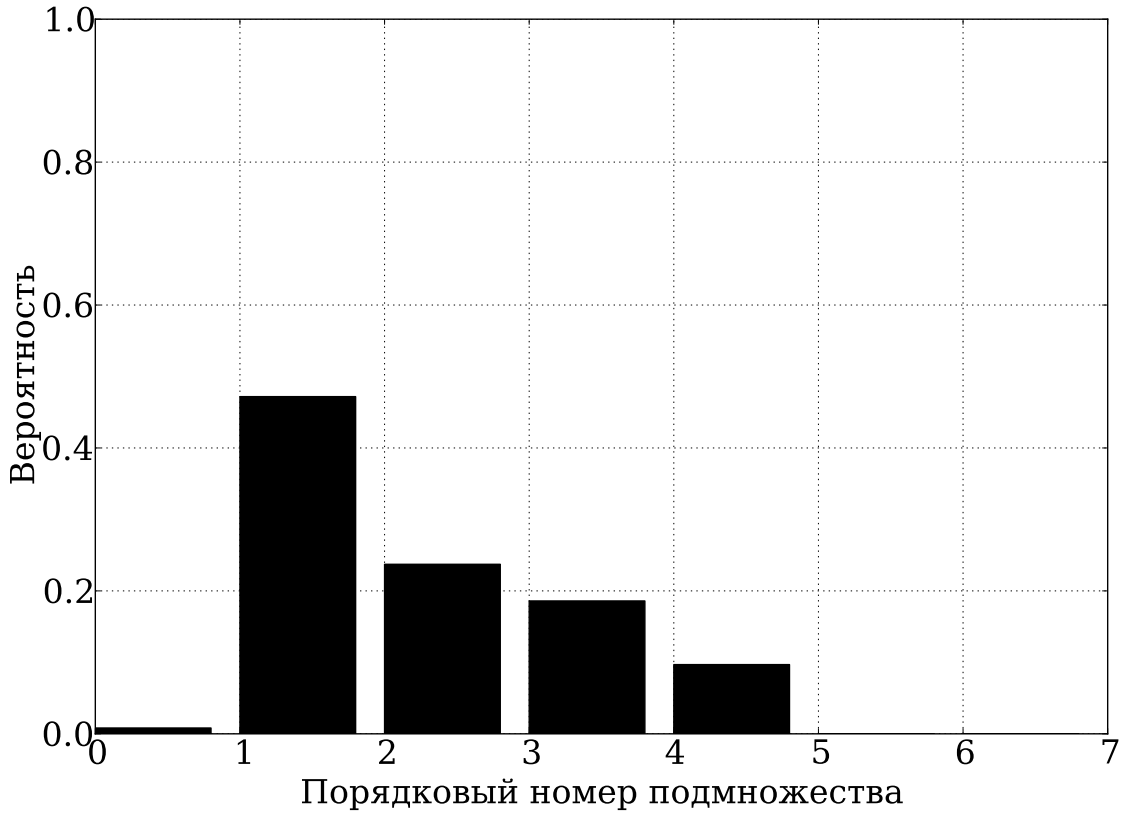


Рисунок 3.8 — Вероятности пребывания СМО в подмножествах $X_{0,0},\ X_{0,1}, X_{1,1}, X_{1,2}, X_{2,2}, X_{2,3}, X_{3,3}, X_{4,4}$

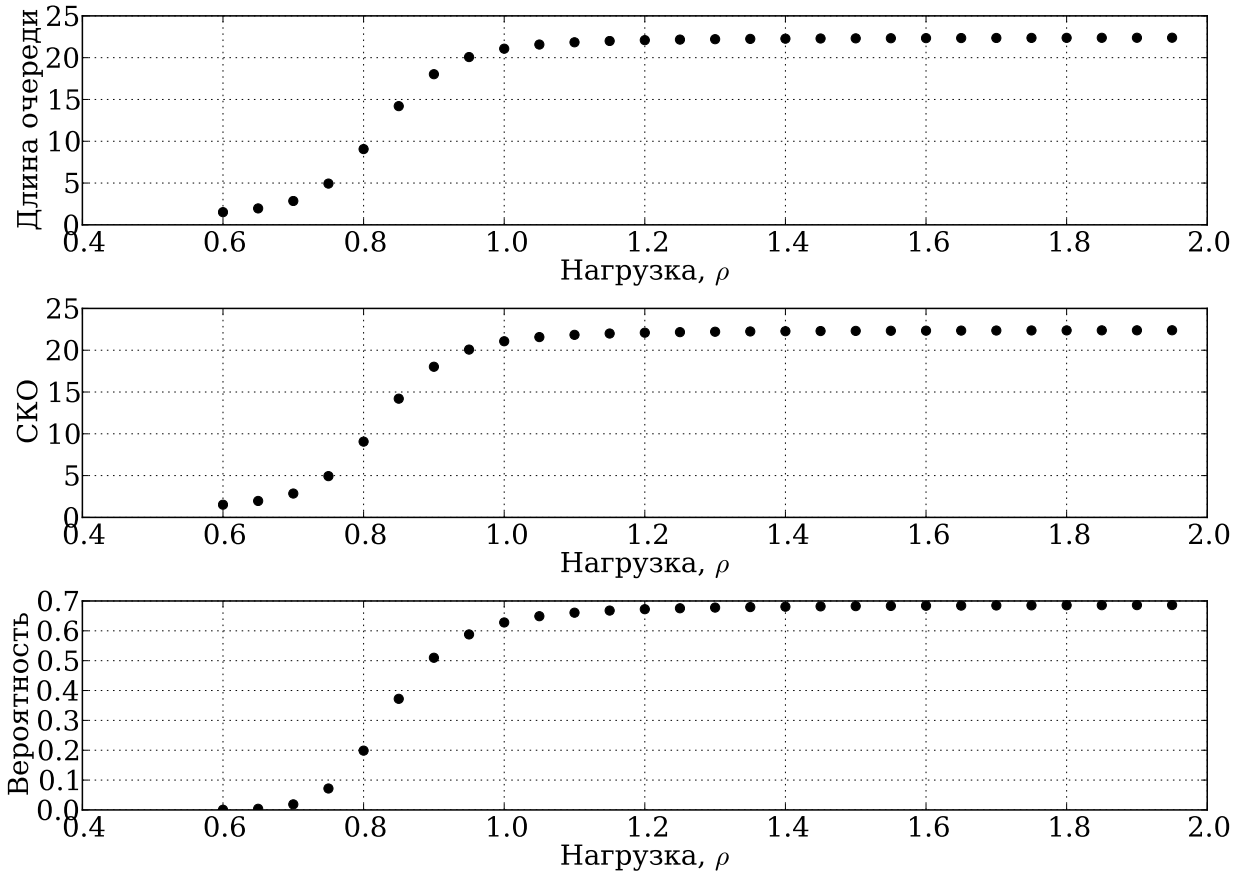


Рисунок 3.9 — Средняя длина очереди, среднеквадратическое отклонение длины очереди и вероятность отклонения длины очереди от эталонного значения

Сравнение рассчитанного среднего значения длины очереди при разной нагрузке с данными имитационного моделирования системы M/M/1/50 FLC при эталонном значении $Q_{ref} = 25$ приведено на рис. 3.10. Также для сравнения, на том же рисунке, приведёна нагрузочная кривая модели M/M/1/n Tail Drop рассчитанная по известной аналитической формуле. Стационарные вероятности нахождения числа пакетов в системе для модели M/M/1/n равны [95]:

$$p_k = \rho^k \frac{1-\rho}{1-\rho^{n+1}}, \tag{3.11}$$

где k — число пакетов в системе от 0 до n. В отличие от систем M/M/1, стационарное распределение числа пакетов в системе существует при любых конечных значени коэффициена нагрузки ρ. При $\rho = 1$ вычисление по формуле 3.11 можно выполнить, используя правило Лопиталя, то есть в пределе отношение функций будет равно пределу отношений производных этих функций.

Зная вероятности каждого состояния можно найти среднее число пакетов в системе равное математическому ожиданию [96]:

$$\overline{N} = \sum_{k=0}^{n} k p_k. \tag{3.12}$$

Из формул 3.11 и 3.12 среднее число пакетов в очереди может быть найдено как:

$$\overline{N} = \frac{\rho}{1-\rho} - \frac{(n+1)\rho^{n+1}}{1-\rho^{n+1}}. \tag{3.13}$$

Рост средней длины очереди на имитационной модели при небольшой нагрузке $\rho < 1$ повторяет график модели M/M/1/50 Tail Drop, а при переходе в режим перегрузки $\rho > 1$, также как и график математической модели, стремится к эталонному значению $Q_{ref} = 25$.

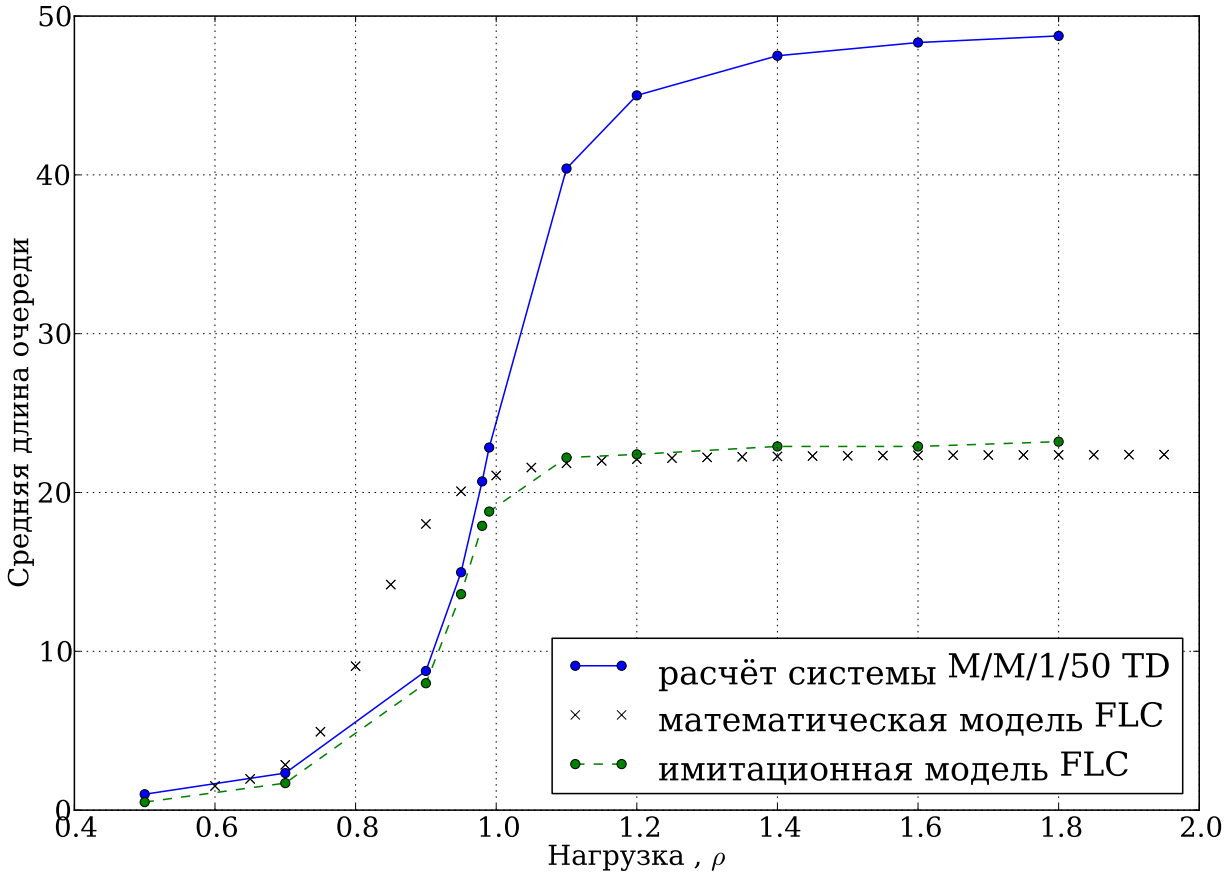


Рисунок 3.10 — Зависимость средней длины очереди от нагрузки

Выводы

1. Математическая модель процесса обслуживания пакетов в очереди маршрутизатора с применением метода на основе нечёткой логики (FLC), построенная на базе гистерезисной модели управления, адекватно описывает указанный процесс, что доказывается сравнением с результатами имитационного моделирования.
2. Анализ полученных результатов показал, что при малой загрузке маршрутизатора ($\rho < 1$) регулятор нагрузки с нечёткой логикой не влияет на процесс обслуживания пакетов в очереди маршрутизатора.
3. При перегрузке ($\rho > 1$) регулятор стремиться к установленному значению вне зависимости от уровня перегрузки в диапазоне ($1 < \rho < 2$).

Глава 4. Разработка и внедрение Linux-маршрутизатора с FLC

4.1 Управление трафиком в Linux-маршрутизаторе

Для внедрения разработанного метода управления очередями на основе нечёткой логики было выбрано ядро операционной системы Linux, так как оно свободно доступно в исходных кодах на языке программирования C на сайте проекта [97] и может быть скомпилировано для большого числа аппаратных архитектур. В ядре Linux уже присутствует поддержка множества сетевых технологий, таких как стек протоколов TCP/IP, технологии дифференцированного обслуживания DiffServ [98], включая различные методы управления трафиком и очередями, например метод управления очередями и предотвращения перегрузок RED, а также возможно добавление новых методов. В качестве транспортного протокола по умолчанию используется реализация TCP Cubic. Есть, также поддержка метода уведомления о перегрузке ECN, но по умолчанию она отключена. Таким образом, большинство сетевых технологий уже присутствует и нам необходимо только внедрить свой метод управления в виде модуля ядра. Следует отметить, что разработка модуля для ядра Linux накладывает ряд ограничений на использование математических методов. В ядре Linux запрещены математические операции с плавающей точкой, поэтому для некоторых операций приходится прибегать к методам целочисленной математики и использовать операции побитового сдвига целых чисел вместо умножения и деления вещественных чисел. Кроме, того модуль работает в общем адресном пространстве ядра как единая программа, и при компиляции модуля должны использоваться специфичные для данного ядра заголовочные файлы.

Непосредственно с аппаратным обеспечением маршрутизатора взаимодействует ядро Linux, и если полученный на сетевом интерфейсе IP-пакет адресован локальному IP-адресу, то ядро передаёт пакет в пользовательское окружение в обслуживающее приложение. Схема прохождения пакетов от сетевого интерфейса маршрутизатора через ядро Linux показана на рис. 4.1.

Для управления трафиком в ядре используется приложение *tc* (Traffic Control) [99] работающее в пользовательском окружении. С помощью пользо-

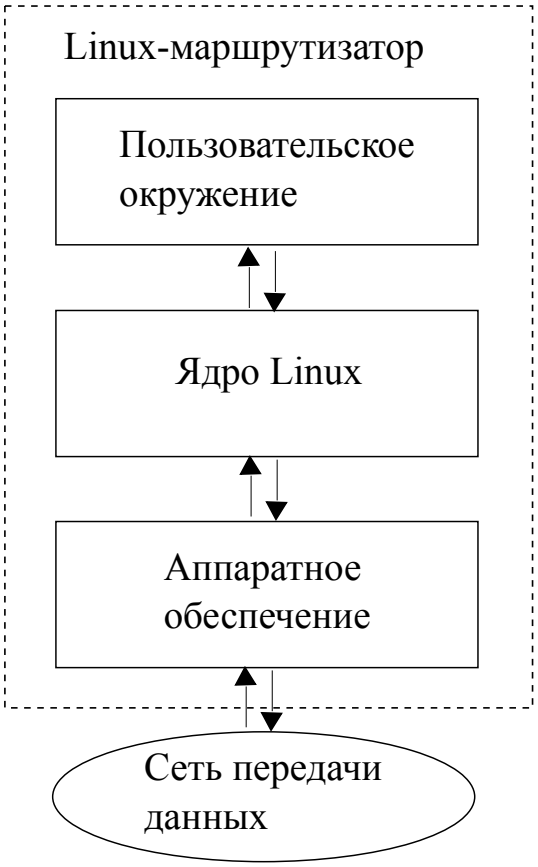


Рисунок 4.1 — Структура Linux-маршрутизатора

вательского приложения tc возможно создавать иерархические очереди, назначать дисциплины обслуживания, проводить классификацию трафика и применять политики обслуживания в зависимости от класса трафика. Для обработки пакетов поступающих на вход сетевого интерфейса в ядре организована входная дисциплина обслуживания обозначенная Ingress Qdics на рис. 4.2. Далее в зависимости от того, если IP-пакет адресован локальному приложению, то пакет поступает в IP-стек для распаковки данных для пользовательского приложения. Если получателем пакета является другой IP-адрес, то пакет пересылается на соответствующий выходной сетевой интерфейс (пересылка пакетов должна быть активирована в ядре). Исходящий трафик перед передачей в сеть может классифицироваться в блоке Egress Classifier, и в зависимости от присвоенного класса отправляться в отдельную очередь со своей дисциплиной обслуживания.

Для конфигурации разработанного модуля использовалась модифицированная утилита *tc*. В качестве основного метода контроля трафика применялся иерархический метод «Корзины с жетонами» (Hierarhial Tocken Bucket – HTB), принципы работы которого рассматривались в главе 1.2.2. Входящий трафик разделялся на два класса: класс 1:10 для трафика управления и класс 1:5 для

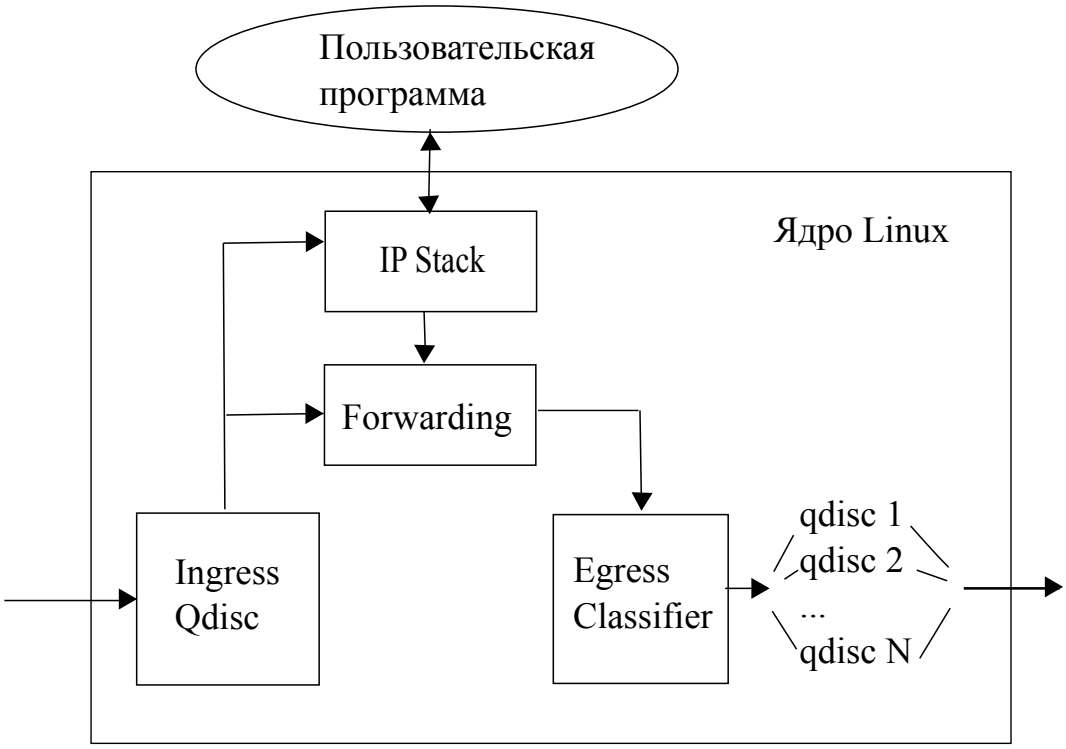


Рисунок 4.2 — Управление трафиком в Linux-маршрутизаторе

тестового трафика. Для класса тестового трафика установлено ограничение на максимальную скорость передачи данных 50 Мбит/с и далее этот трафик обрабатывается дисциплиной обслуживания FLC с максимальной длиной очереди 500 Кбайт и заданной длиной 250 Кбайт.

Параметры утилиты *tc* для создания описанной выше конфигурации приведены в листинге 4.1.

Листинг 4.1 — Создание дисциплин обслуживания с помощью утилиты tc

```
1 tc qdisc add dev eth0 root netem delay 10ms
2 tc qdisc replace dev eth0 root handle 1: htb default 10
3 tc class add dev eth0 parent 1: classid 1:1 htb rate 100Mbit
4 tc class add dev eth0 parent 1:1 classid 1:10 htb rate 50Mbit ceil 100Mbit prio 1
5 tc class add dev eth0 parent 1:1 classid 1:5 htb rate 50Mbit ceil 50Mbit prio 2
6 tc qdisc add dev eth0 parent 1:5 handle 5: flc limit 500K qref 250000 d_scale 12500 sampling 6
    p_rate0 50000 q_lim 500000 ecn
7 tc filter add dev eth0 parent 1:0 prio 1 protocol ip handle 5 fw flowid 1:5
```

Схема классификации трафика приведена на рис. 4.3.

В сроке 1 мы задаём задержку 10 мс. Строка 2 устанавливает основной метод управления HTB с классом по умолчанию 10, а строка 3 – максимальную общую скорости 100 Мбит/с. Строка 4 устанавливает параметры для класса 1:10 с гарантированной скоростью 50 Мбит/с и максимальной скоростью 100 Мбит/с для трафика управления. Строка 5 устанавливает параметры для класса 1:5 с гарантированной скоростью 50 Мбит/с и максимальной скоростью тоже

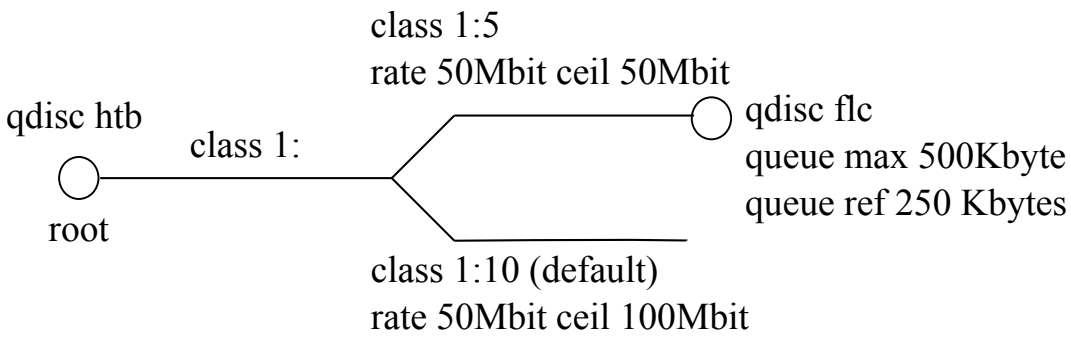


Рисунок 4.3 — Схема классификации исходящего трафика

50 Мбит/с для тестового трафика. Далее в строке 6 мы указываем, что трафик класса 1:5 нужно обрабатывать методом *flc* с соответствующими параметрами. Исходный код модуля *sch_flc_k3.5.c* для ядра Linux, реализующий разработанный метод FLC, доступен в репозитории [75]. Значения функции управления для дисциплины обслуживания с нечётким регулятором FLC рассчитывались предварительно на основе заданных параметров, и передавались в модуль ядра в виде таблицы целых чисел, для избежания вычислений с плавающей точкой внутри ядра Linux. В последней строке 7 мы задаём правило для фильтра, который будет назначать в класс 1:5 специально промаркированные пакеты тестового трафика. Предварительную маркировку пакетов мы выполняли с помощью пользовательского приложения *iptables*, являющегося стандартным приложением Linux.

С помощью цепочки фильтров *iptables* возможна обработка пакетов на каждом из этапов прохождения трафика в Linux. Схема цепочки фильтров iptables приведена на рис. 4.4.

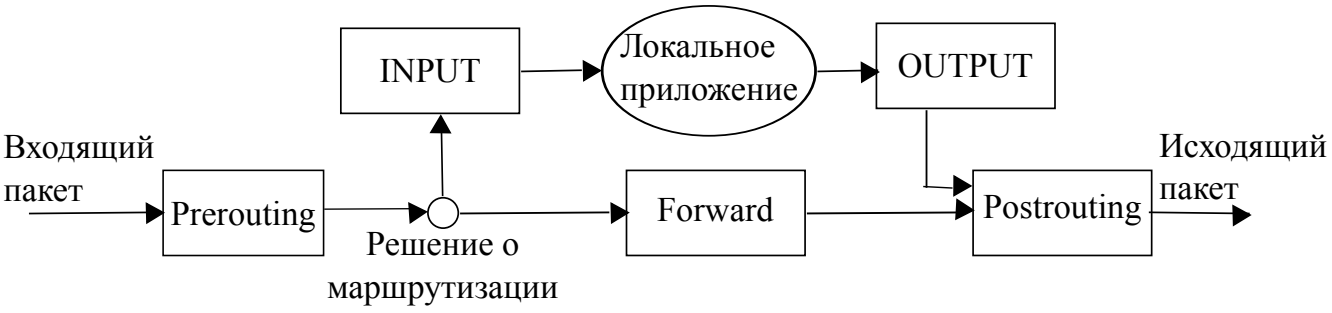


Рисунок 4.4 — Схема прохождения пакетом цепочки фильтров в Linux-маршрутизаторе

Входящие пакеты обрабатываются в блоке Prerouting, далее в зависимости от решения о маршрутизации пакет поступает в пользовательское окружение через блок Input или проходит через блок Forward к исходящему порту. Ес-

ли пакет отправляется из пользовательского окружения, то он проходит через блок Output. Перед отправкой в сеть все пакеты могут обрабатываться в блоке фильтров Postrouting.

Пример предварительной маркировки тестового трафика для его классификации с помощью утилиты *iptables* во время эксперимента, приведён в листинге 4.2.

Листинг 4.2 — Маркировка трафика для классификации

```
iptables -A OUTPUT -t mangle -p tcp --dport 5001 -j MARK --set-mark 5
```

Таким образом, пользовательскому TCP–трафику имеющему TCP–порт назначения 5001 в цепочке OUTPUT будет присвоена маркировка с индексом 5, и далее пакеты маркированные этим индексом будут классифицированы в соответствующий класс обслуживания 1:5, который обрабатывается в очереди с дисциплиной обслуживания FLC.

4.2 Испытания метода FLC для обработки трафика на виртуальной машине

Для эксперимента по обработке трафика в очереди маршрутизатора была запущена операционная система OpenWRT Linux [100] в виртуальном окружении Oracle VirtualBox [101]. Дисциплина обслуживания выделенной очереди для тестового трафика *flc* была сконфигурирована для проведения измерения длины очереди и интенсивности трафика каждые 6 мс и изменения, с помощью нечёткого регулятора, вероятности сброса/маркировки пакетов на величину не более $8 \cdot 10^{-5}$ за один интервал измерения. Такие же параметры использовались при моделировании работы нечёткого регулятора в NS–2 в раздле 2.6. Поддержка явного уведомления о перегрузки ECN была включена для TCP передатчиков. Схема эксперимента приведена на рис. 4.5.

В качестве генератора TCP и UDP трафика использовалась утилита *iperf* [102]. С помощью этой утилиты создавалось 100 потоков TCP с размером TCP сегмента 964 байта. В начальный момент времени все 100 TCP передатчиков устанавливали соединение и начинали отправку пакетов. Через 100 секунд после начала, 50 передатчиков прекращали передачу и пакеты передавали только

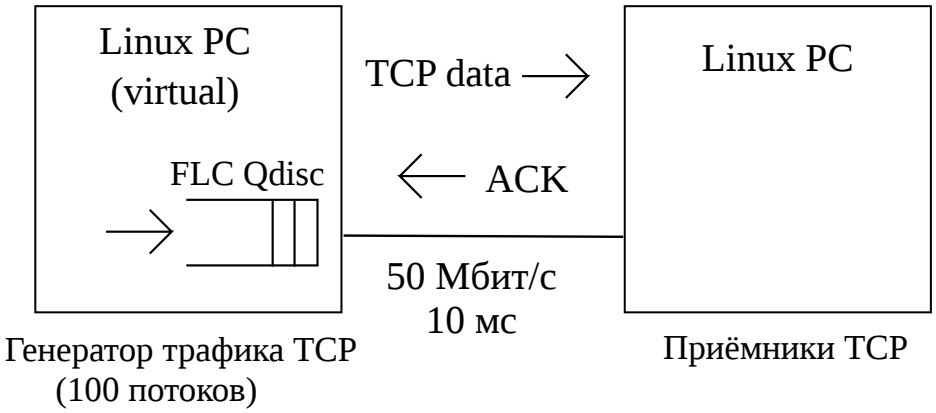


Рисунок 4.5 — Схема эксперимента

оставшиеся 50 TCP передатчиков. После 100 секундного прерывания возобновлялась передача всех 100 передатчиков. Таким образом имитировалась скачкообразное изменение интенсивности трафика. Общее время эксперимента составило 300 секунд. Дополнительно к TCP трафику передавался также UDP трафик со скоростью 128 кбит/с и с размером дейтаграммы 980 байт. Изменение длины очереди маршрутизатора за время эксперимента приведено на рис. 4.6.

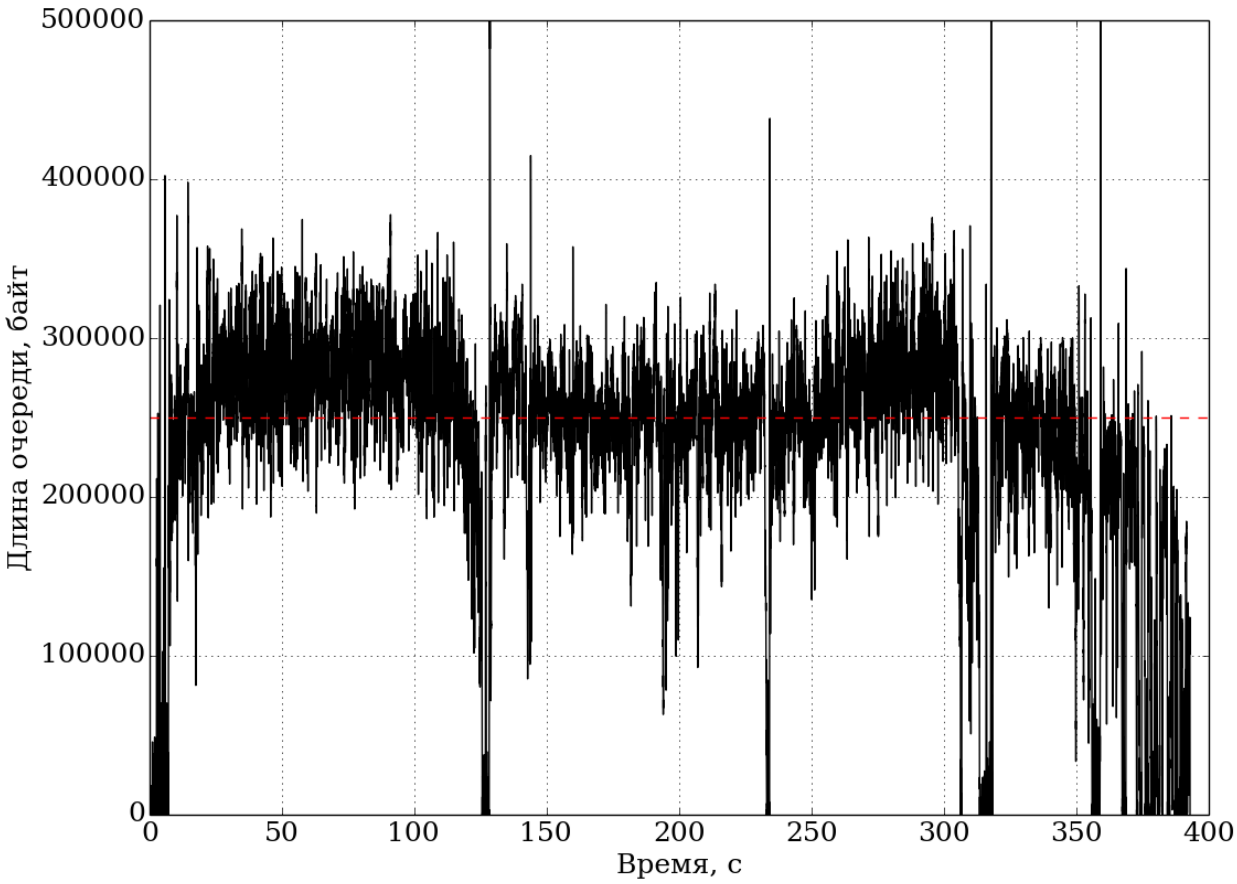


Рисунок 4.6 — Изменение длины очереди в Linux–маршрутизаторе при использовании метода FLC

Рис. 4.6 показывает, что разработанный метод управления очередью FLC может удерживать длину очереди около заданного значения 250 Кбайт, несмотря на скачкообразное изменение интенсивности трафика и перегрузку. На рисунке также видно, что все 100 TCP соединения не устанавливаются одновременно, и что требуется около 25 секунд для того чтобы начать передачу. Через 300 секунд эксперимента передача данных не заканчивается резко, и требуется ещё около 100 секунд, чтобы завершить все соединения. За время эксперимента через 100 TCP соединений было передано 2 Гбайт данных (или 2 млн пакетов) доступная полоса пропускания канала была полностью утилизирована. Сброс пакетов TCP из очереди из-за переполнения был незначительный (около сотни пакетов), благодаря применению маркировки пакетов и уведомлению о перегрузке с помощью технологии ECN. Сброс пакетов UDP составил 3% и джиттер 3 мс. Больший процент потери пакетов UDP по сравнению с TCP, объясняется тем, что применяемый метод FLC использовал маркировку пакетов TCP вместо их сброса, а пакеты UDP, не имеющие механизма маркировки сбрасывались.

Подобный сценарий тестирования был применён также для метода RED и TailDrop, для того чтобы выполнить сравнение параметров качества при использовании различных методов управления. Эксперимент с каждым методом был повторён 5 раз для каждого метода, для оценки доверительных интервалов измеренных характеристик. Для метода RED использовался минимальный порог длины очереди 125 Кбайт, максимальный порог 375 Кбайт и максимальная длина очереди 500 Кбайт. То есть, в соответствии с рекомендациями максимальный порог в 3 раза выше минимального, а средняя длина очереди соответствовала 250 Кбайт и равнялась заданной длине очереди для метода FLC. Вероятность сброса для метода RED была установлена 0,02. Поддержка ECN также включена. Для метода TailDrop размер очереди равнялся 500 Кбайт, технология ECN методом TailDrop не поддерживается. Средние значения потерь пакетов с доверительными интервалами для 5 экспериментов с тремя методами: FLC, TailDrop и RED приведены на рис. 4.7.

Благодаря применению ECN потери при использовании методов FLC и RED значительно ниже, чем для метода TailDrop. Метод TailDrop допускает значительные потери из–за переполнения очереди.

Средние величины значения джиттера с доверительными интервалами по уровню 95% для трафика UDP приведены на рис. 4.8.

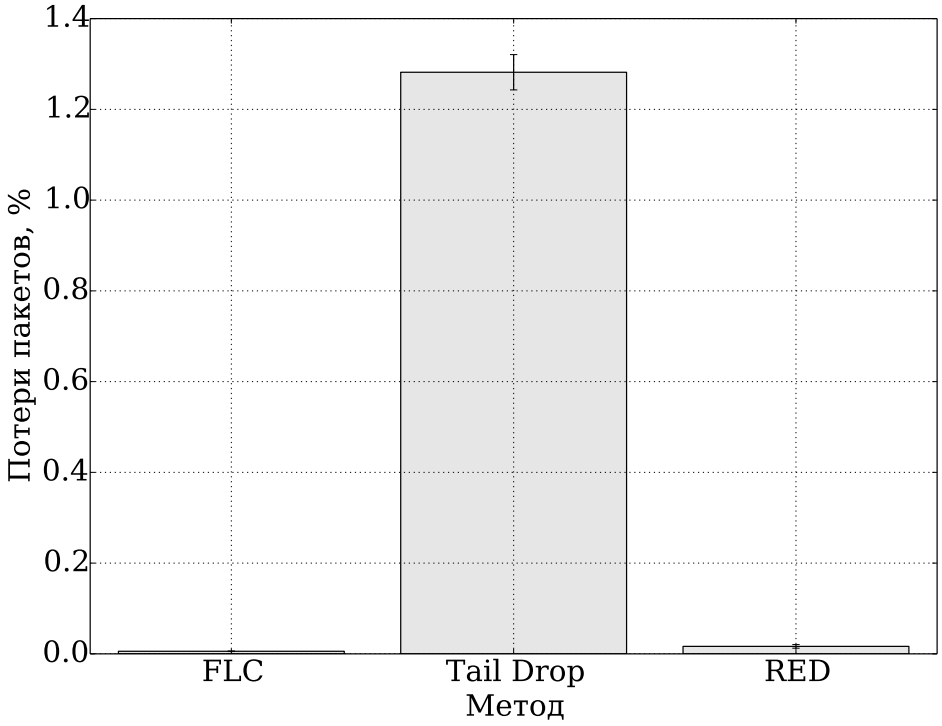


Рисунок 4.7 — Средние значения потерь пакетов

Наименьшее значение джиттера с меньшим доверительным интервалом было зафиксировано для метода FLC. Таким образом метод FLC показал лучшую способность по стабилизации длины очереди среди испытанных методов. Метод RED показал большую величину джиттера потому, что контролирует только усреднённую длину очереди в пределах установленных порогов и допускает колебания мгновенного значения длины очереди. Метод TailDrop показал нестабильное поведение длины очереди.

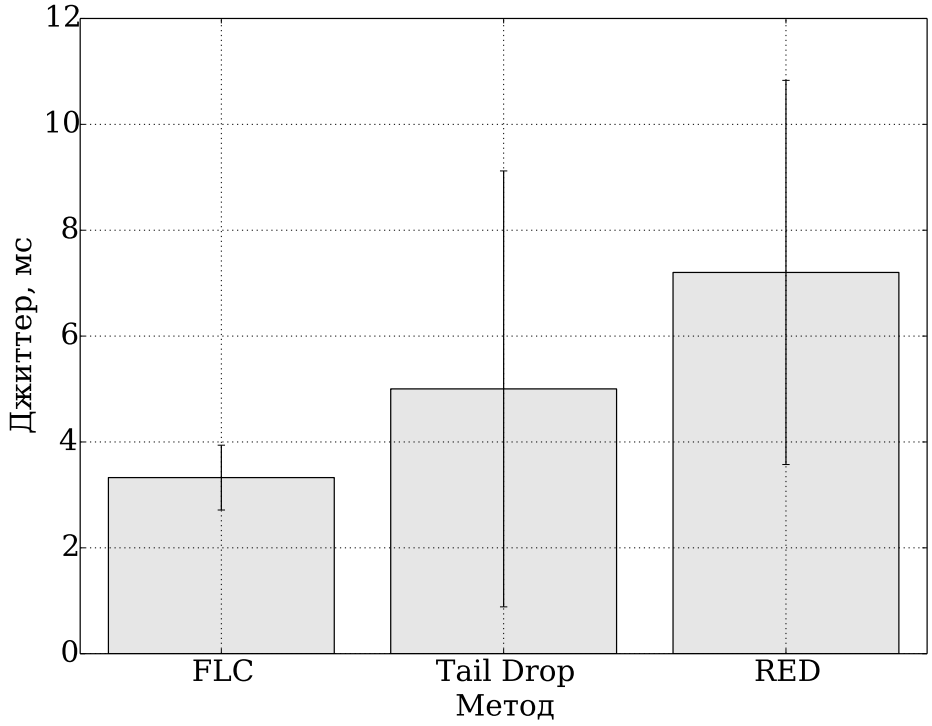


Рисунок 4.8 — Средние значения джиттера для UDP потока

4.3 Испытания метода FLC для обработки трафика в очереди маршрутизатора

Для внедрения метода обработки трафика в очередях на основе нечёткого регулятора, код разработанного программного модуля ядра был подготовлен для аппаратного маршрутизатора WL–500gP. Исходный код модуля *sch_flc_k2.4.c* доступен в репозитории [75]. Технические характеристики маршрутизатора приведены в табл. 4.1, а внешний вид показан на рис. 4.9.

Таблица 4.1 — Технические характеристики маршрутизатора

Характеристика	Значение
Архитектура	MIPS
Чипсет	Broadcom BCM5354
Частота процессора	240 МГц
Оперативная память	32 Мбайт

Рисунок 4.9 — Маршрутизатор WL–500gP под управлением ОС Linux

Маршрутизатор был подключён между двум тестовыми компьютерами выполняющими роль генератора и приёмника трафика, причём один из компьютеров был подключён по беспроводному каналу WLAN по стандарту IEEE

802.11g. Скорость в канале передачи данных была ограничена до 15 Мбит/с. Схема подключения маршрутизатора приведена на рис. 4.10.

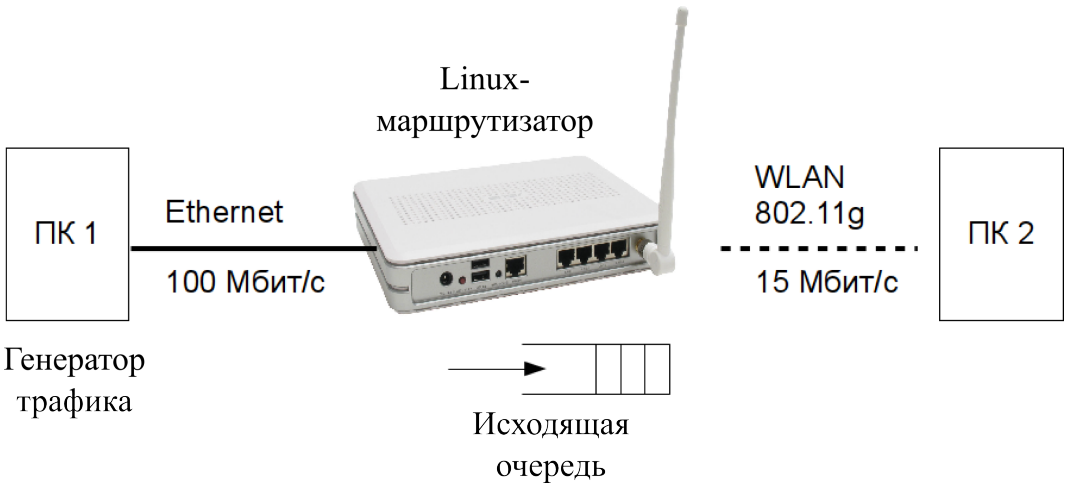


Рисунок 4.10 — Схема подключения маршрутизатора

Среднее значение двусторонней задержки RTT составило 2,5 мс. Через маршрутизатор передавался трафик со скачкообразно меняющейся интенсивностью. На всех устройствах была включена поддержка проколола ECN. Количество TCP соединений резко изменялось с 50 потоков до 25 и обратно до 25 соединений через интервалы по 100 секунд. Одновременно с TCP трафиком передавался тестовый поток UDP с постоянной скоростью 128 кбит/с. Размер пакетов для обоих типов трафика — 1000 байт. Максимальный размер выходной очереди маршрутизатора был установлен 500 Кбайт, а заданное значение Q_{ref} для метода FLC равнялось 250 Кбайт. Изменение длины очереди маршрутизатора во время эксперимента приведено на рис. 4.11.

На рис. 4.11 видны периоды по 100 секунд, когда количество TCP соединений вдвое уменьшалось, а затем снова увеличивалось. Среднее значение длины очереди за время эксперимента составило 308 Кбайт при среднеквадратичном отклонении 55 Кбайт. Общие потери пакетов — 0,027%. Значение джиттера для протока UDP составило 20 мс.

Маршрутизатор показал устойчивую работу в режиме перегрузки при 100% использовании канала передачи данных. Метод управления FLC совместно с использованием протокола ECN продемонстрировал способность удерживать длину очереди около заданного значения и не допускать переполнения буфера и потерь пакетов из–за переполнения.

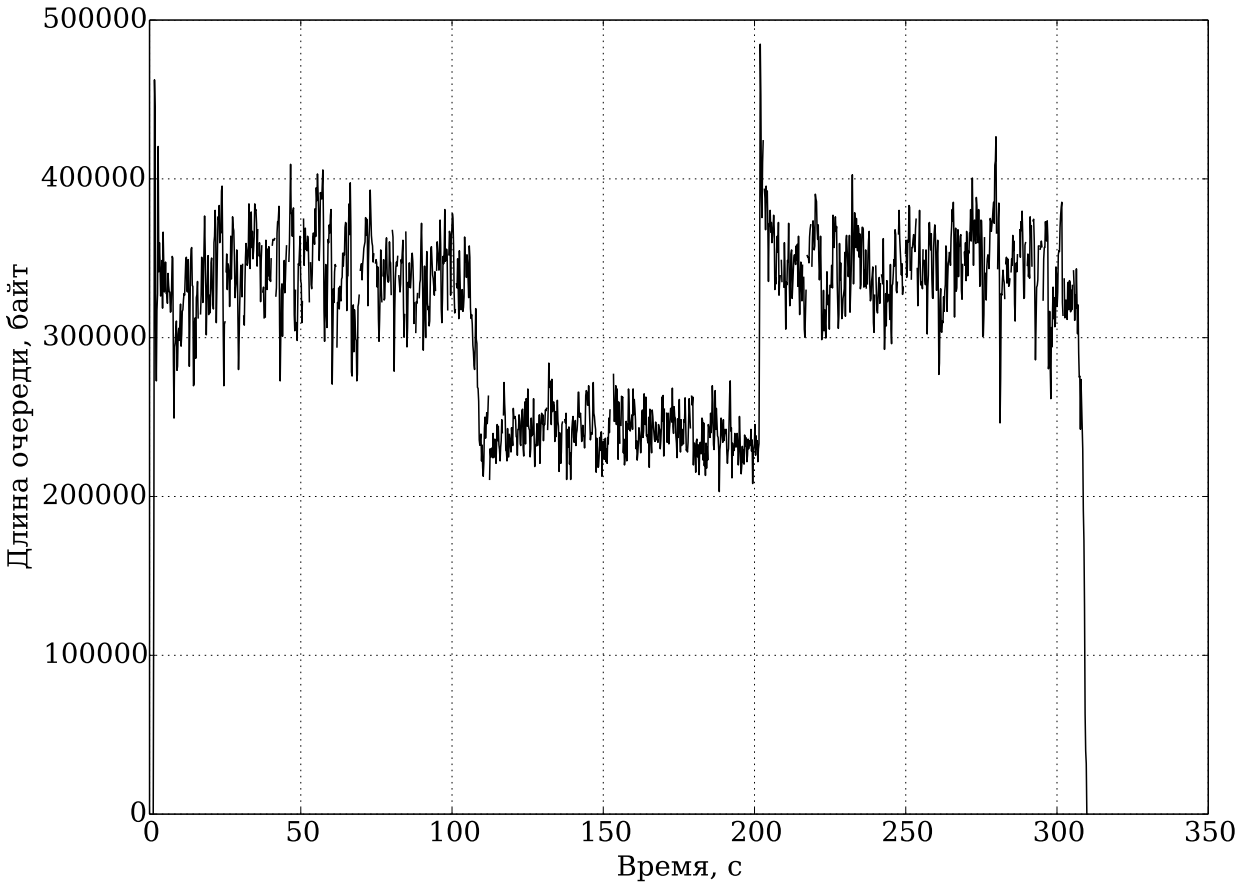


Рисунок 4.11 — Изменение длины очереди маршрутизатора

Выводы

1. Разработанный метод FLC для обработки трафика в очередях и реализованный на базе Linux–маршрутизатора показал способность удерживать длину очереди около заданного значения при работе в режиме перегрузки в лабораторной сети.
2. При использовании метода FLC в маршрутизаторе улучшаются такие параметры качества обслуживания, как процент потерянных пакетов и величина джиттера по сравнению с традиционными методами Tail Drop и RED.

Заключение

В работе были проанализированы современные методы обработки трафика в очередях, такие как RED, Adaptive RED, PI, REM, AVQ, FLC, FEM для использования в маршрутизаторах мультисервисных сетей. С помощью имитационного моделирования показана их сравнительная эффективность. Доказано, что метод управление длиной очереди на основе нечёткой логики дает лучшие результаты, чем перечисленные известные методы. Для этого были разработаны две аналитические модели процесса управления очередью: жидкостная модель TCP потока и модель с гистерезисным пороговым управлением. Разработанная имитационная модель обработки пакетов в маршрутизаторах на базе NS-2 также показала высокую эффективность разработанного метода. Особую практическую значимость придают результаты эксперимента на Linux–маршрутизаторе, в котором был реализован метод обработки трафика на основе нечёткой логики.

Основные результаты работы и выводы:

1. Построенная имитационная модель позволяет провести сравнительный анализ между различными методам обработки трафика в очереди маршрутизатора, такими как Tail Drop, RED, ARED, PI, REM, AVQ, FLC, при одинаковых условиях в режиме перегрузки и при сложной нелинейной динамике трафика в мультисервисной сети.
2. Результаты имитационного моделирования показали, что существующие методы обработки трафика в очередях допускают значительные колебания длины очереди и тем самым ухудшают показатели качества обслуживания, такие как средняя задержка передачи пакета в сети, джиттер, процент потерянных пакетов и коэффициент использования канала. Методы активного управления с использованием нечёткой логики более подходят для работы в условиях нелинейного, скачкообразного изменения интенсивности нагрузки.
3. Разработанный метод обработки трафика в очереди маршрутизатора на основе регулятора с нечёткой логикой (FLC) способен эффективно предотвращать перегрузку в мультисервисной сети и не допускает переполнения и опустошения очереди, а также обеспечивает стабилизацию

длины очереди около заданного эталонного значения со среднеквадратичным отклонением 10% от среднего значения при перегрузке в сети.

4. При обработке TCP–трафика с помощью разработанного контроллера FLC совместно с методом явного уведомления о перегрузке ECN, потери пакетов для длительных TCP–соединениях составили менее 0,03% в режиме перегрузки в сети.
5. Процесс обработки трафика в очереди в маршрутизаторе на базе регулятора с нечёткой логикой впервые описан с помощью математической модели процесса обслуживания с гистерезисным управлением с порогами. Составлена и решена система уравнений равновесия (СУР) этого процесса. Получены выражения для вероятностно–временных характеристик.
6. Численное решение СУР построенной модели с гистерезисным управлением позволяет оценить среднюю длину очереди и среднеквадратичное отклонение в широком диапазоне коэффициента нагрузки, включая режим перегрузки ($\rho > 1$). Построенные математические модели адекватны системе обслуживания очереди с регулятором на основе нечёткой логики, что подтверждается сравнением с данными имитационного моделирования.
7. Использование метода обслуживания очереди на основе FLC в Linux–маршрутизаторе улучшает параметры качества передачи данных, такие как процент потери пакетов и джиттер, и в режиме перегрузки превосходит характеристики, получаемые при использовании традиционных в маршрутизаторах методов Tail Drop и RED в исследуемом диапазоне характеристик канала передачи данных представляющих практический интерес.

Список сокращений и обозначений

ATM — Asynchronous Transfer Mode (Асинхронный режим передачи)

CIR — Committed Information Rate (Гарантированная скорость)

CoS — Class of Service (Класс обслуживания)

DSCP — Differentiated Service Code Point (Код дифференцированного обслуживания)

ECN — Explicit Congestion Notification (Явное уведомление о перегрузке)

FCFS — First Come First Served (Пришедший первым обслуживается первым)

FLC — Fuzzy Logic Controller (Регулятор с нечёткой логикой)

FR — Frame Relay (Ретрансляция кадров)

NS-2 — Network Simulator 2 (Сетевой симулятор 2)

OAM — Operation, Administration and Maintenance (Эксплуатация и техобслуживание)

OSI — Open Systems Interconnection (Взаимодействие Открытых Систем)

PIR — Peak Information Rate (Негарантированная скорость)

QoS — Quality of Service (Качество обслуживания)

RED — Random Early Detection (Случайное раннее обнаружение)

RTT — Round Trip Time (Время двусторонней задержки)

SLA — Service Level Agreement (Соглашение об уровне обслуживания)

TD — Tail Drop (Отбрасывание конца очереди)

TOS — Type of Service (Тип сервиса)

TrTCM — Two–Rate Three–Color Marker (Двухскоростной трёхцветный маркировщик)

VLAN — Virtual Local Area Network (Виртуальная ЛВС)

VoIP — Voice over IP (Голос через IP)

WFQ — Weighted Fair Queueing (Очереди со взвешенным обслуживанием)

МП — Марковский процесс

СКО — Среднеквадратичное отклонение

СМО — Система массового обслуживания

СУР — Система уравнений равновесия

Список литературы

1. Jacobson V. Congestion avoidance and control // In proceedings of ACM SIGCOMM. — Vol. 18. — 1988. — P. 314–329.

2. Floyd S., Jacobson V. Random Early Detection gateways for Congestion Avoidance // IEEE/ACM Transactions on Networking. — 1993. — Vol. 1, no. 4. — P. 397–413.

3. Adaptive RED: An Algorithm for Increasing the Robustness of RED's Active Queue Management : Rep. / ICSI ; Executor: S. Floyd, R. Gummadi, S. Shenker : 2001.

4. Misra V., Gong W., Towsley D. A Fluid-based Analysis of a Network of AQM Routers Supporting TCP Flows with an Application to RED // Proc. SIGCOMM 2000. — ACM, 2000. — P. 151–160.

5. On Designing Improved Controllers for AQM Routers Supporting TCP Flows / C. V. Hollot, V. Misra, D. Towsley, W. Gong // IN PROCEEDINGS OF IEEE INFOCOM. — 2000. — P. 1726–1734.

6. Analysis and Design of Controllers for AQM Routers Supporting TCP Flows / C. V. Hollot, V. Misra, D. Towsley, W. Gong // IEEE Transactions on Automatic Control. — 2002. — Vol. 47. — P. 945–959.

7. A Self-Configuring RED Gateway / Wu chang Feng, D. Kandlur, D. Saha, K. Shin // INFOCOM. — 1999. — P. 1320–1328.

8. BLUE: A New Class of Active Queue Management Algorithms : Rep. / University of Michigan ; Executor: Wu chang Feng, D.D. Kandlur, D.S., K.G. Shin : 1999.

9. REM: Active Queue Management / S. Athuraliya, Victor H. Li, Steven H. Low, Qinghe Yin // IEEE NETWORK. — 2000. — Vol. 15. — P. 48–53.

10. Kunniyur Srisankar S., Srikant R. Analysis and Design of an Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management // SIGCOMM'01. — 2001. — P. 123–134.

11. Kunniyur Srisankar S., Srikant R. An Adaptive Virtual Queue (AVQ) Algorithm for Active Queue Management // IEEE/ACM Transactions on Networking. — 2004. — Vol. 12. — P. 286–299.

12. Fengyuan R., Yong R., Xiuming S. Design of Fuzzy Logic Controller for Active Queue Management // Computer Communications. — 2002. — no. 25. — P. 874–883.

13. Chrysostomou C., Pitsillides A., Sekercioglu Y.A. Fuzzy Explicit Marking: a Unified Congestion Controller for Best-Effort and Diff-Serv Networks // Computer Networks. — 2009. — no. 53. — P. 650–667.

14. Абаев П.О., Гайдамака Ю.В., Самуйлов К.Е. Гистерезисное управление сигнальной нагрузкой в сети SIP-серверов // Вестник РУДН. Серия «Математика. Информатика. Физика». — 2011. — № 4. — С. 54–71.

15. Gaidamaka Yu.V. Model with Threshold Control for Analyzing a Server with an SIP Protocol in the Overload Mode // Automatic Control and Computer Sciences. — 2013. — Vol. 47, no. 4. — P. 211–218.

16. Ефимушкин В.А. Особенности взаимоувязанного функционирования систем управления качеством услуг и соглашений об уровне обслуживания // В сб.: Материалы конференции «Услуги электросвязи. Инновационные решения, тенденции и проблемы». — 2010. — С. 3–4.

17. Кучерявый Е.А. Управление трафиком и качество обслуживания в сети Интернет. — СПб. : Наука и Техника, 2004. — 336 с.

18. Бочаров П.П., Печинкин А.В. Теория массового обслуживания. — М. : РУДН, 1995. — 529 с.

19. Интернет вещей / А.В. Росляков, С.В. Ваняшин, А.Ю. Гребешков, М.Ю. Самсонов. — Самара: ПГУТИ : ООО«Издательство Ас Гард», 2014. — 340 с.

20. Росляков А.В., Самсонов М.Ю., Шибаева И.В. IP–телефония. — М. : Эко–Трендз, 2003. — 252 с.

21. Наумов В.А., Самуйлов К.Е., Яркина Н.В. Теория телетрафика мультисервисных сетей. — М. : РУДН, 2007. — 192 с.

22. Лагутин В.С., Степанов С.Н. Телетрафик мультисервисных сетей связи. — М. : Радио и связь, 2000. — 320 с.

23. Степанов С.Н., Столяр Н.Ф. Разработка алгоритмов оценки основных показателей качества обслуживания мультисервисных потоков сообщений на корпоративных сетях связи. — М. : Информсвязь, 2005. — 47–74 с.

24. Степанов С.Н. Основы телетрафика мультисервисных сетей. — М. : Эко-Трендз, 2010. — 392 с.

25. Цитович И. И., Чернушевич А. В. О влиянии гистерезиса управления трафиком на эффективность функционирования мультисервисной сети // Обозрение прикладной и промышленной математики. — 2010. — № 2. — С. 314–315.

26. Сегхайер А., Цитович И.И. Об интервальной модели для процесса рождения и гибели с гистерезисом // Информационные процессы. — 2012. — № 1. — С. 117–126.

27. Гольдштейн Б.С., Соколов Н.А., Яновский Г.Г. Сети связи: Учебник для ВУЗов. — СПб. : БХВ-Петербург, 2010. — 400 с.

28. Яновский Г.Г. Качество обслуживания в сетях IP // Вестник связи. — 2008. — № 1. — С. 65–74.

29. Яновский Г.Г. Оценка качества передачи речи в сетях IP // Вестник связи. — 2008. — № 2. — С. 68–70.

30. Гайдамака Ю.В., Масленников А.Г. Об одной системе массового обслуживания с активным управлением очередью // Вестник РУДН. Серия «Математика. Информатика. Физика». — 2013. — № 4. — С. 56–64.

31. Деарт В.Ю., Масленников А.Г. Исследование влияния параметров канала передачи данных на процедуры управления очередью // T–Comm — Телекоммуникации и Транспорт. — 2012. — № 7. — С. 77–81.

32. Deart V., Maslennikov A., Gaidamaka Y. A Hysteretic Model of Queuing System with Fuzzy Logic Active Queue Management // Proc. of 15th Conference of Open Innovations Association FRUCT. — St.–Petersburg : IEEE, 2014. — P. 32–38.

33. Maslennikov A. Mathematical model of queue system with fuzzy logic controller // Proc. of International Conference Distributed Computer and Communication Networks (DCCN): Control, Computation, Communications. — Moscow : Technosphera, 2013. — P. 338–344.

34. Деарт В.Ю., Масленников А.Г. Жидкостная модель управления очередью маршрутизатора с нечёткой логикой // Материалы Всероссийской конференции с международным участием «Информационно–телекоммуникационные технологии и математическое моделировании высокотехнологичных систем» (ИТТММ–2013). — М. : РУДН, 2013. — С. 89–91.

35. Deart V., Maslennikov V. Fuzzy logic queue discipline processing over bottleneck link // Proc. of 11th Conference of Open Innovations Association Finnish–Russian University of Cooperation in Telecommunications (FRUCT–11). — St.–Petersburg : State University of Aerospace Instrumentation (SUAI), 2012. — P. 40–46.

36. Деарт В.Ю., Масленников А.Г. Применение нечёткого регулятора для стабилизации длины очереди маршрутизатора // Тезисы докладов Всероссийской конференции с международным участием. — «Информационно–телекоммуникационные технологии и математическое моделирование высокотехнологичных систем» (ИТТММ–2012). — М. : РУДН, 2012. — С. 83–85.

37. Масленников А.Г. Методы активного управления очередями маршрутизаторов // Электронный журнал «Вычислительные сети. Теория и практика». — 2011. — № 2(19):4.2. — URL: http://network-journal.mpei.ac.ru.

38. Деарт В.Ю., Масленников А.Г. Разработка метода управления очередью с помощью регулятора на нечёткой логике // Труды конференции «Телекоммуникационные и вычислительные системы». — Международный форум информатизации (МФИ–2011). — М. : МАИ, 2011. — С. 21–22.

39. Деарт В.Ю., Масленников А.Г. Анализ методов активного управления очередями // Труды XIX Международной научно–технической конференции «Информационные средства и технологии». — Т. 1. — М. : Издательский дом МЭИ, 2011. — С. 259–266.

40. Деарт В.Ю., Масленников А.Г. Применение механизма с нечёткой логикой для управления очередями маршрутизаторов в сетях TCP/IP // Тезисы докладов Всероссийской конференции с международным участием. — «Информационно–телекоммуникационные технологии и математическое моделирование высокотехнологичных систем» (ИТТММ–2011). — М. : РУДН, 2011. — С. 102–105.

41. Гольдштейн Б.С., Пинчук А.В., Суховитский А.Л. IP-телефония. — М. : Радио и связь, 2006. — 336 с.

42. Гольдштейн Б.С., Кучерявый А.Е. Сети связи пост–NGN. — СПб. : БХВ-Петербург, 2013. — 160 с.

43. Деарт В.Ю. Мультисервисные сети связи. Транспортные сети и сети доступа. — М. : Инсвязьиздат, 2008. — 168 с.

44. Нетес В.А. Качество обслуживания на сетях связи. Обзор рекомендаций МСЭ–Т // Сети и системы связи. — 1999. — № 3. — С. 66–71.

45. Васильев А.Б., Тарасов Д.В., Андреев Д.В. Под общей редакцией проф. Кучерявого А.Е. Тестирование сетей связи следующего поколения. — М. : ФГУП ЦНИИС, 2008. — 144 с.

46. Денисова Т.Б., Лихтциндер Б.Я., Назаров А.Н. Мультисервисные АТМ–сети. — М. : Эко-Трендз, 2005. — 320 с.

47. Масленников А.Г. Доставка качественного Ethernet // «Вестник Связи». — 2009. — № 4. — С. 77–78.

48. Крухмалев В.В., Гордиенко В.Н., Моченов А.Д. и др. Основы построения телекоммуникационных систем и сетей. — М. : Горячая линия–Телеком, 2004. — 510 с.

49. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 3-е изд. — СПб. : Питер, 2009. — 958 с.

50. Филимонов А.Ю. Построение мультисервисных сетей Ethernet. — СПб. : БХВ-Петербург, 2007. — 592 с.

51. IEEE Std. 802.1Q-2011, Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks, 2011.

52. IEEE 802.1p: LAN Layer 2 QoS/CoS Protocol for Traffic Prioritization, 2012.

53. IETF: Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers, RFC–2474, 1998.

54. Кучерявый А.Е., Парамонов А.И., Кучерявый Е.А. Сети связи общего пользования. — М. : ФГУП ЦНИИС, 2008. — 296 с.

55. Heinanen J., Guerin R. — A Two Rate Three Color Marker, RFC–2698, 1999.

56. MEF 6.1 Ethernet Services Definitions — Phase 2, April, 2008.

57. Ефимушкин В.А., Углов И.В. Архитектура QoS для конвергентных сетей и особенности ее применения // T–Comm. — 2010. — № 7. — С. 162–163.

58. Телекоммуникационные системы и сети / В.В. Величко, Е.А. Субботин, В.П. Шувалов, А.Ф. Ярославцев. — М. : Горячая линия-Телеком, 2005. — Т. 3. — 592 с.

59. Ramakrishnan K., Floyd S., Black D. — The Addition of Explicit Congestion Notification (ECN) to IP, RFC–3168, 2001.

60. Sawashima H., Sunahara Y. H. H. Characteristics of UDP packet loss: effect of TCP traffic // Proc. INET '97. — Internet Society, 1997. — P. 6. — URL: http://www.isoc.org/inet97/proceedings/F3/F3_1.HTM.

61. Reasons not to deploy RED / M. May, J. Bolot, C. Diot, B. Lyles // Seventh International Workshop on Quality of service (IWQoS '99). — 1999. — P. 40–46.

62. Class-Based Weighted Fair Queueing and Weighted Random Early Detection. — URL: http://www.cisco.com/c/en/us/td/docs/ios/12_0s/feature/guide/fswfq26.html.

63. IETF: An Architecture for Differentiated Services, RFC–2475, 1998.

64. Гончаров А.А., Семенов Ю.А. Исследование влияния параметров алгоритма WRED на осциляции длин очередей в маршрутизаторе // Информационные процессы. — 2006. — Т. 6, № 2. — С. 153–195.

65. Гончаров А.А., Семенов Ю.А. Исследование влияния параметров алгоритма WRED на качество обслуживания при передаче данных по перегруженным каналам // Информационные процессы. — 2006. — Т. 6, № 4. — С. 364–374.

66. A GA–based PID active queue managment control design for TCP/IP networks / H.H Kuo, C.K. Chen, J.J. Yan, T.L. Liao // 2007 International Symposium on Nonlinear Dynamics. — 2008.

67. Пегат А. Нечёткое моделирование и управление / пер. с англ. — М. : Бином. Лаборатория знаний, 2009. — 798 с.

68. Jalili-Kharaajoo M. Adaptive Fuzzy Active Queue Managment and Congestion Avoidance in TCP/AQM Networks // FORTE 2004 Workshops. — 2004. — no. 3236. — P. 196–208.

69. Nyerda C.N., Dawoud D.S. Self-Organization in a Particle Swarm Optimized Fuzzy Logic Congestion Detection Mechanism for IP Networks // Scientia Iranica. — 2008. — Vol. 15, no. 6. — P. 589–604.

70. Tabash I.K., Mamun M.A.A., Negi A. A Fuzzy Based Network Congestion Control Using Active Queue Managment Techniques // Journal of Scientific Research. — 2010. — no. 2. — P. 273–284.

71. Disign Novel AQM Schemes By Using Artificial Intelligence Technologies / S. Ghasempour, M. Hedayati, S.H. Kamali, R. Shakerian // The Journal of Mathematics and Computer Science. — 2011. — Vol. 2, no. 3. — P. 436–447.

72. The Network Simulator 2, NS-2. — URL: http://www.isi.edu/nsnam/ns/.

73. Altman E., Jimenez T. — NS Simulator for beginners, Lecture notes, 2003.

74. Пилюгин А.В. Разработка метода оценки параметров качества обслуживания HTTP–трафика в мультисервисных сетях доступа // Диссертация на соискание ученой степени кандидата технических наук, МТУСИ. — 2010.

75. Хранилище исходных кодов программ на сайте Github. — URL: https://github.com/amasl2048/aspirant/.

76. Заде Л. Понятие лингвистической переменной и его применение к принятию приближенных решений / пер. с англ. — М. : Мир, 1976. — 165 с.

77. Дьяконов В.П., Круглов В.В. MATLAB 6.5 SPI/7/7 SPI/7 SP2 + Simulink 5/6. Инструменты искусственного интеллекта и биоинформатики. Библиотека профессионала. — М. : СОЛОН-ПРЕСС, 2006. — 456 с.

78. Яхъяева Г.Э. Нечёткие множества и нейронные сети: учебное пособие.- 2 изд., испр. — М. : Интернет-Университет Информационных Технологий; БИНОМ. Лаборатория знаний, 2010. — 316 с.

79. Тэрано Т, Асаи К., Сугено М. Прикладные нечеткие системы. — М. : Мир, 1993. — 336 с.

80. Mamdani E.H. Applications of fuzzy algorithms for control of simple dynamic plant // Proceedings of the IEEE 121. — 1974. — P. 1584–1588.

81. The fuzzy system development environment Xfuzzy. — URL: http://www.imse-cnm.csic.es/Xfuzzy/.

82. The NewReno Modification to TCP's Fast Recovery Algorithm, RFC-3782. — URL: http://tools.ietf.org/html/rfc3782.

83. Jacobson V. Modified TCP Congestion Avoidance Algorithm // end2end-interest mailing list, April 30. — 1990. — URL: ftp://ftp.isi.edu/end2end/end2end-interest-1990.mail.

84. Королькова А.В. Математическая модель процесса передачи трафика с регулируемой алгоритмом типа RED динамической интенсивностью потока // Автореферат диссертации на соискание ученой степени кандидата физико-математических наук, РУДН. — 2010.

85. IETF: Transmission Control Protocol, RFC–793, 1981.

86. IETF: Window and Acknowledgement Strategy in TCP, RFC–813, 1982.

87. IETF: TCP Congestion Control, RFC–2581, 1999.

88. Башарин Г.П., Бочаров П.П., Коган Я.А. Анализ очередей в вычислительных сетях. Теория и методы расчёта. — М. : Наука, 1989. — 336 с.

89. Башарин Г.П. Лекции по математической теории телетрафика. — М. : РУДН, 2004. — 190 с.

90. Ивницкий В.А. Теория сетей массового обслуживания. — М. : Физико–математическая литература, 2004. — 772 с.

91. Лившиц Б.С., Пшеничников А.П., Харкевич А.Д. Теория телетрафика. — М. : Связь, 1979. — 224 с.

92. Шелухин О.И., Тенякшеев А.М., Осин А.В. Моделирование информационных систем / Под. ред. О.И. Шелухина. Учебное пособие. — М. : Радиотехника, 2005. — 368 с.

93. Вентцель Е.С., Овчаров Л.А. Теория случайных процессов и ее инженерные приложения: Учеб. пособие для студентов втузов. - 3-е изд., перераб. и доп. — М. : Издательский центр «Академия», 2003. — 432 с.

94. Solve a linear matrix equation, or system of linear scalar equations. — URL: http://docs.scipy.org/doc/numpy/reference/generated/numpy.linalg.solve.html.

95. Вишневский В.М. Теоритические основы проектирования компьютерных сетей. — М. : Техносфера, 2003. — 512 с.

96. Крылов В.В., Самохвалова С.С. Теория телетрафика и ее приложения. — СПб. : БХВ-Петербург, 2005. — 288 с.

97. The Linux Kernel Archives. — URL: https://www.kernel.org.

98. Almesberger W., Salim J.H., Kuznetsov A. Differentiated Services on Linux // EPFL. — 1999. — URL: http://infoscience.epfl.ch/record/195.

99. Hubert B. — Linux Advanced Routing & Traffic Control HOWTO. — URL: http://lartc.org/howto/.

100. OpenWRT Linux. — URL: https://openwrt.org.

101. Oracle VM VirtualBox. — URL: https://www.virtualbox.org/.

102. Iperf, NLANR/DAST. — URL: http://iperf.sourceforge.net/.

Приложение А

Результаты имитационного моделирования

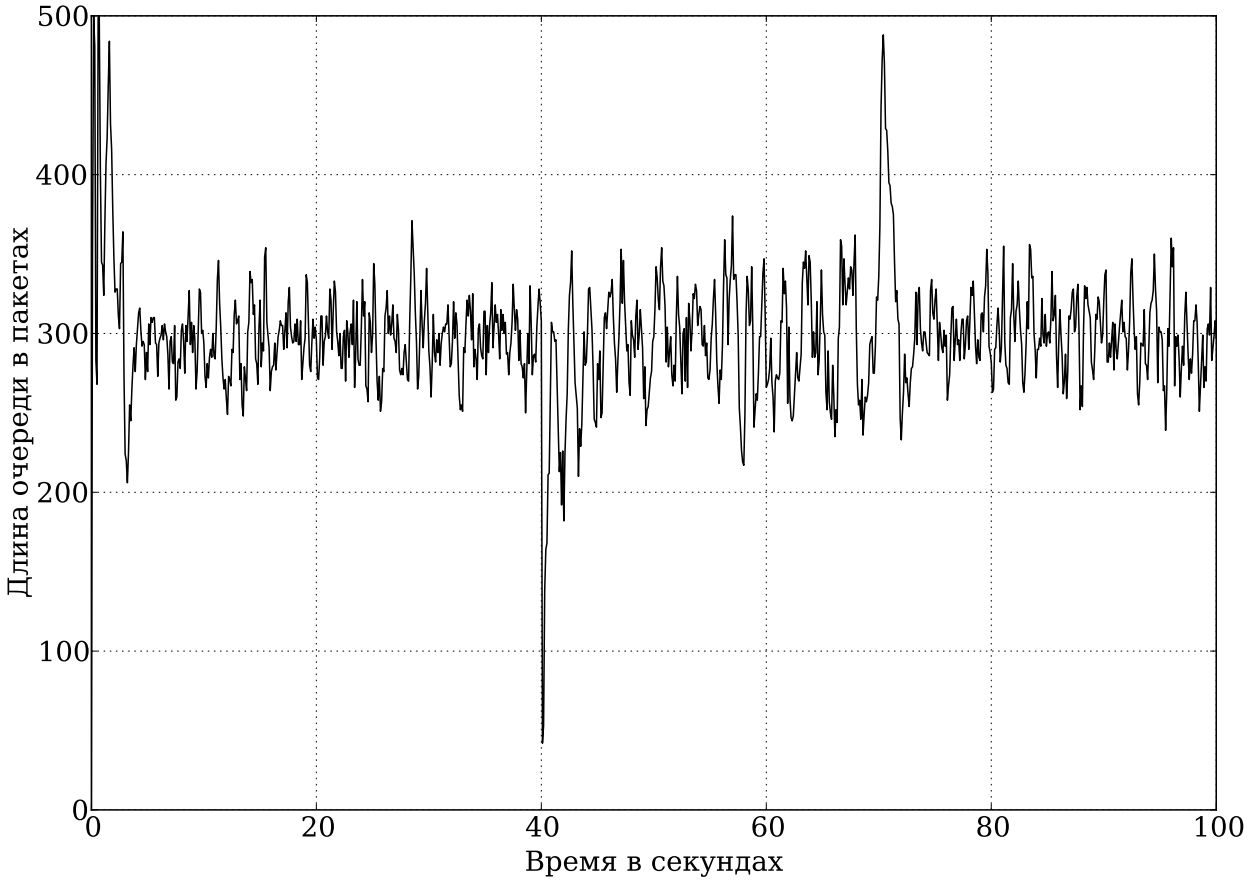


Рис. А.1 — Изменение длины очереди при использовании метода Adaptive RED

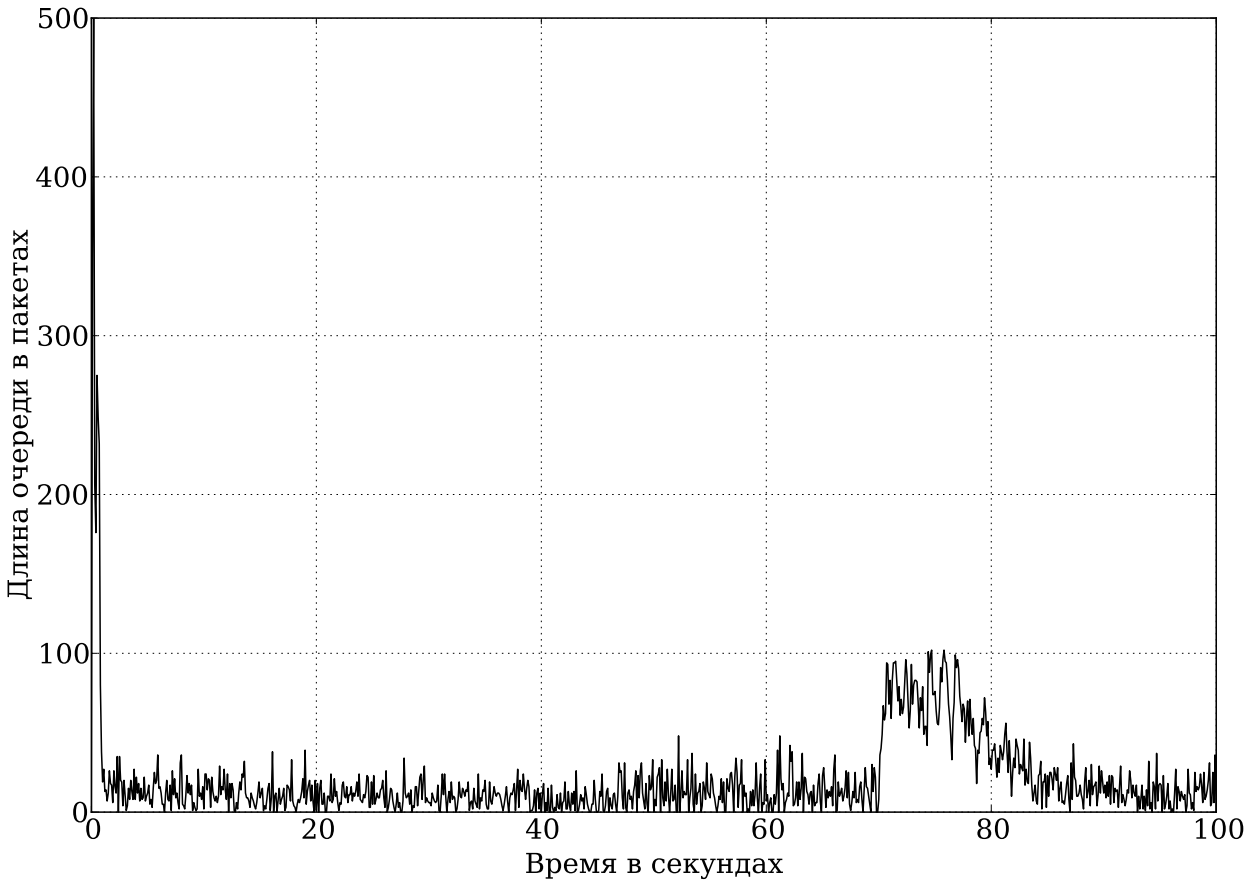


Рис. А.2 — Изменение длины очереди при использовании метода AVQ

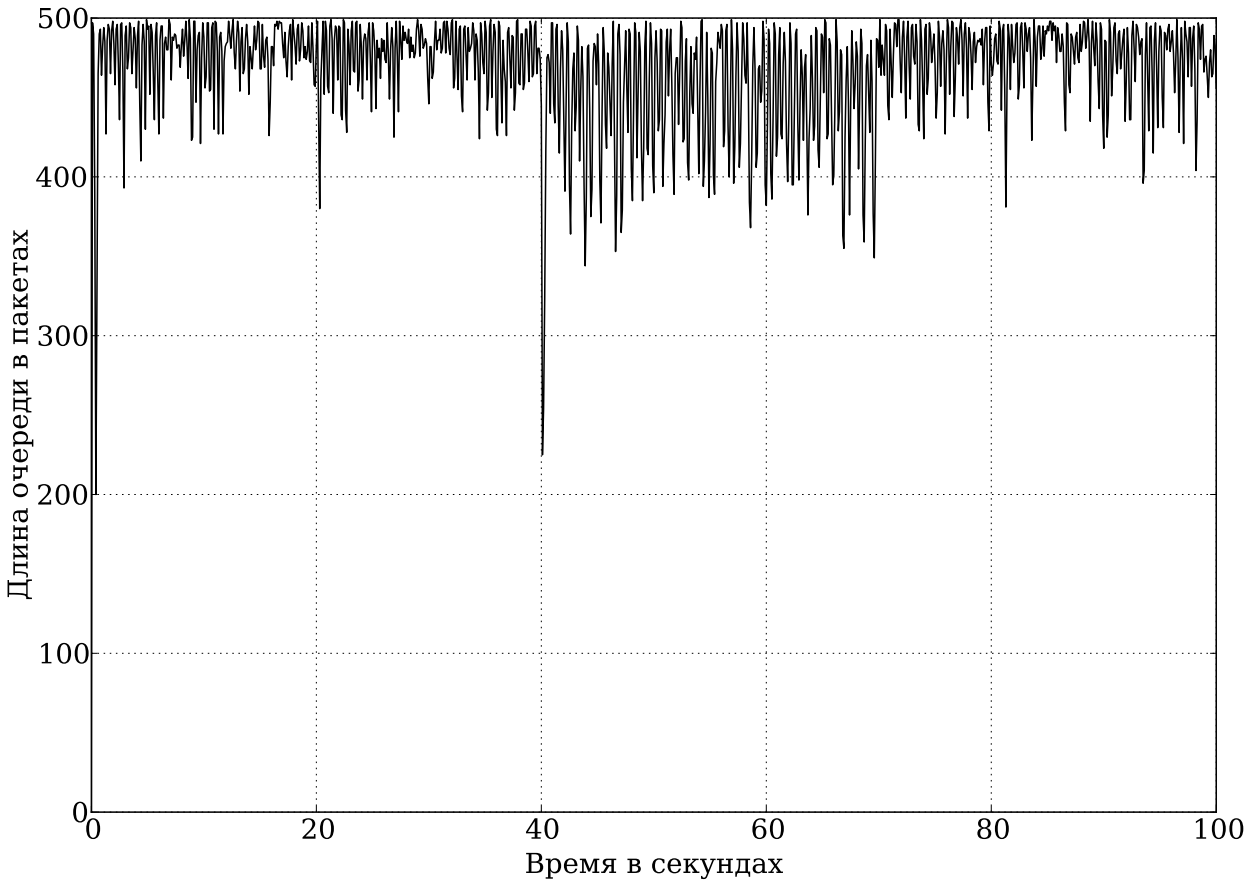


Рис. А.3 — Изменение длины очереди при использовании метода TailDrop

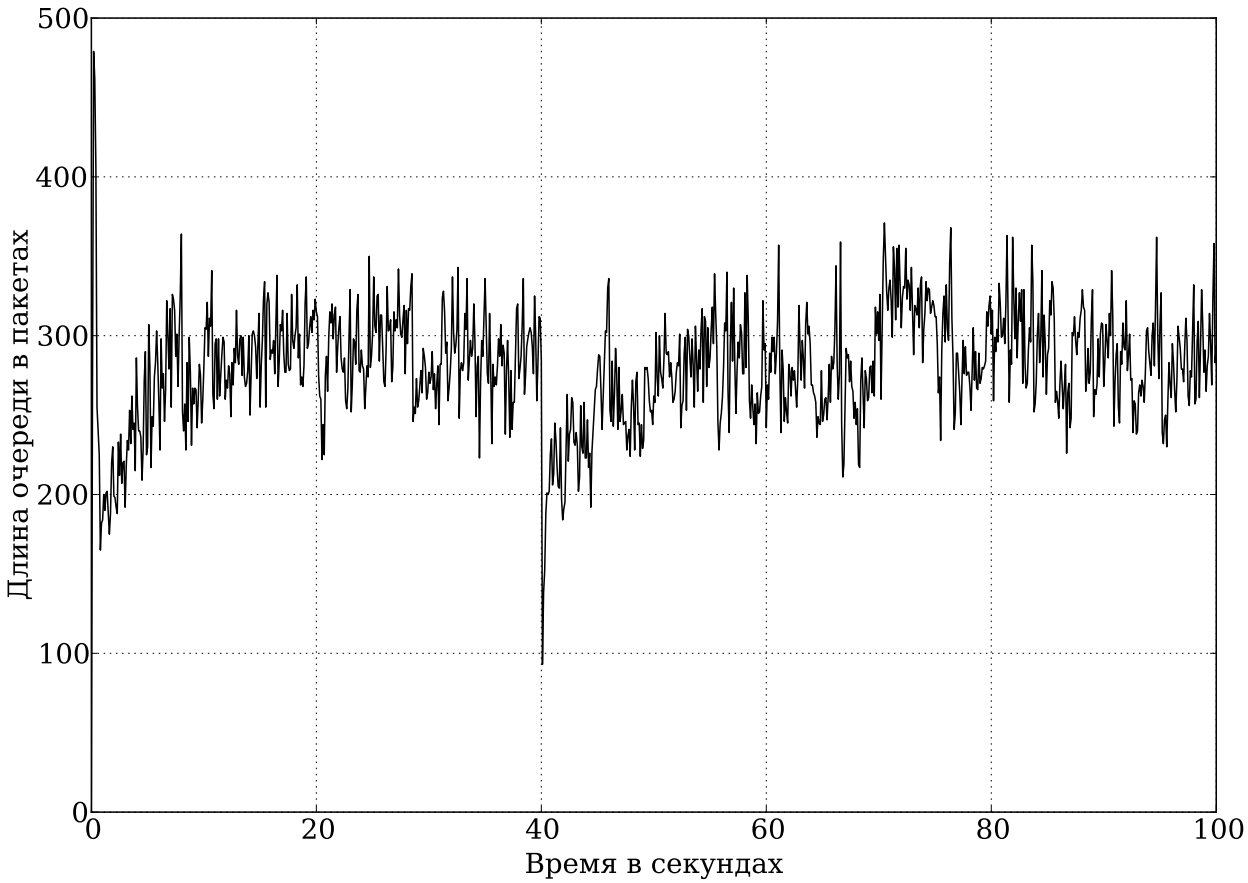


Рис. А.4 — Изменение длины очереди при использовании метода FEM

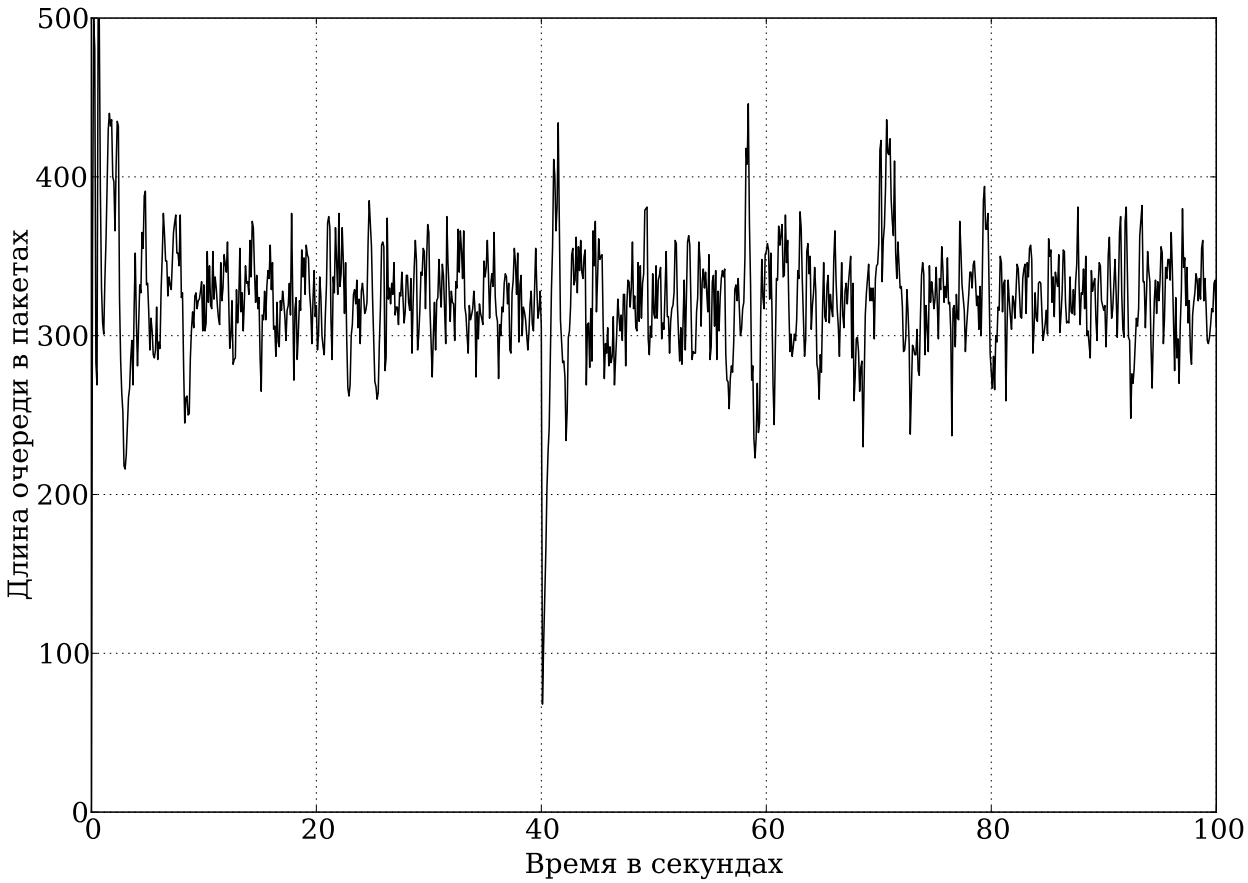


Рис. А.5 — Изменение длины очереди при использовании метода FLC2

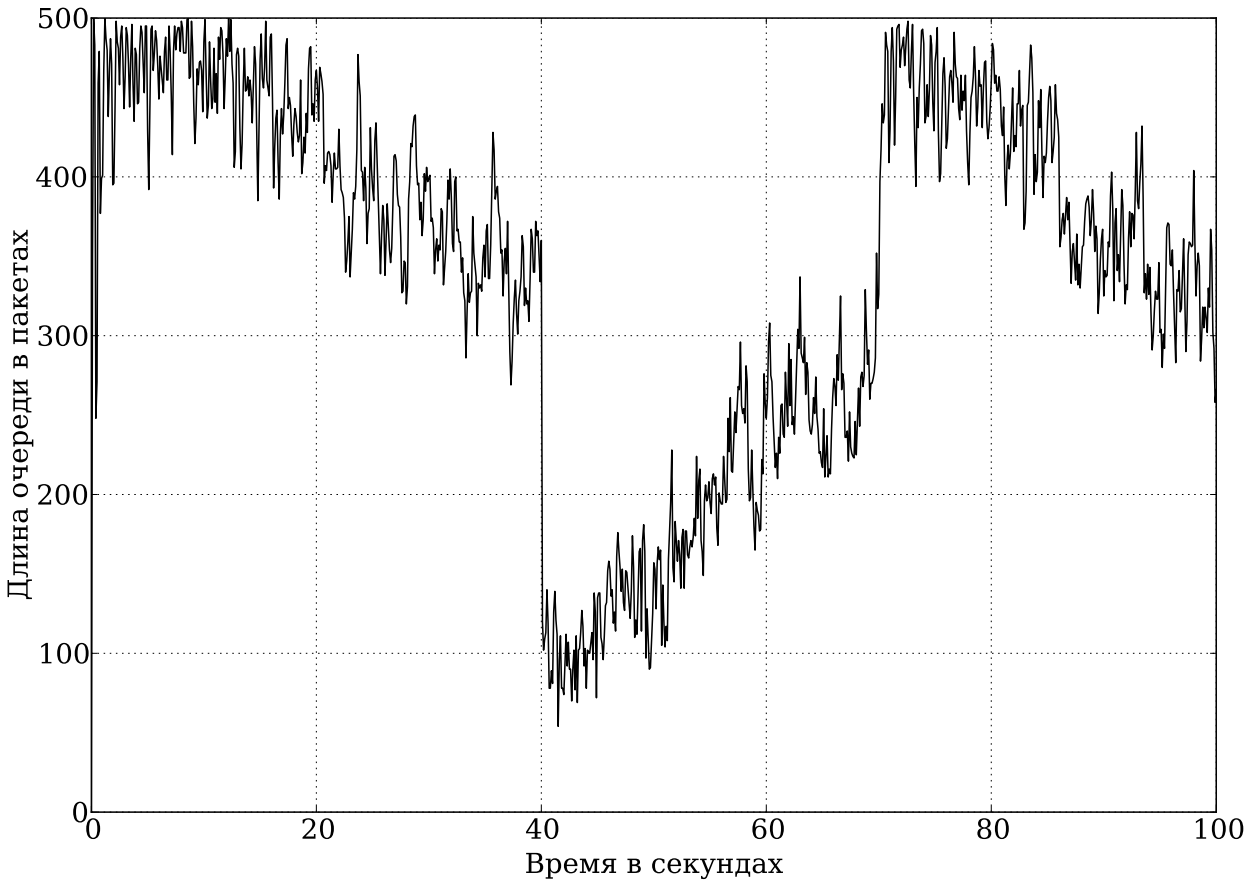


Рис. А.6 — Изменение длины очереди при использовании метода PI

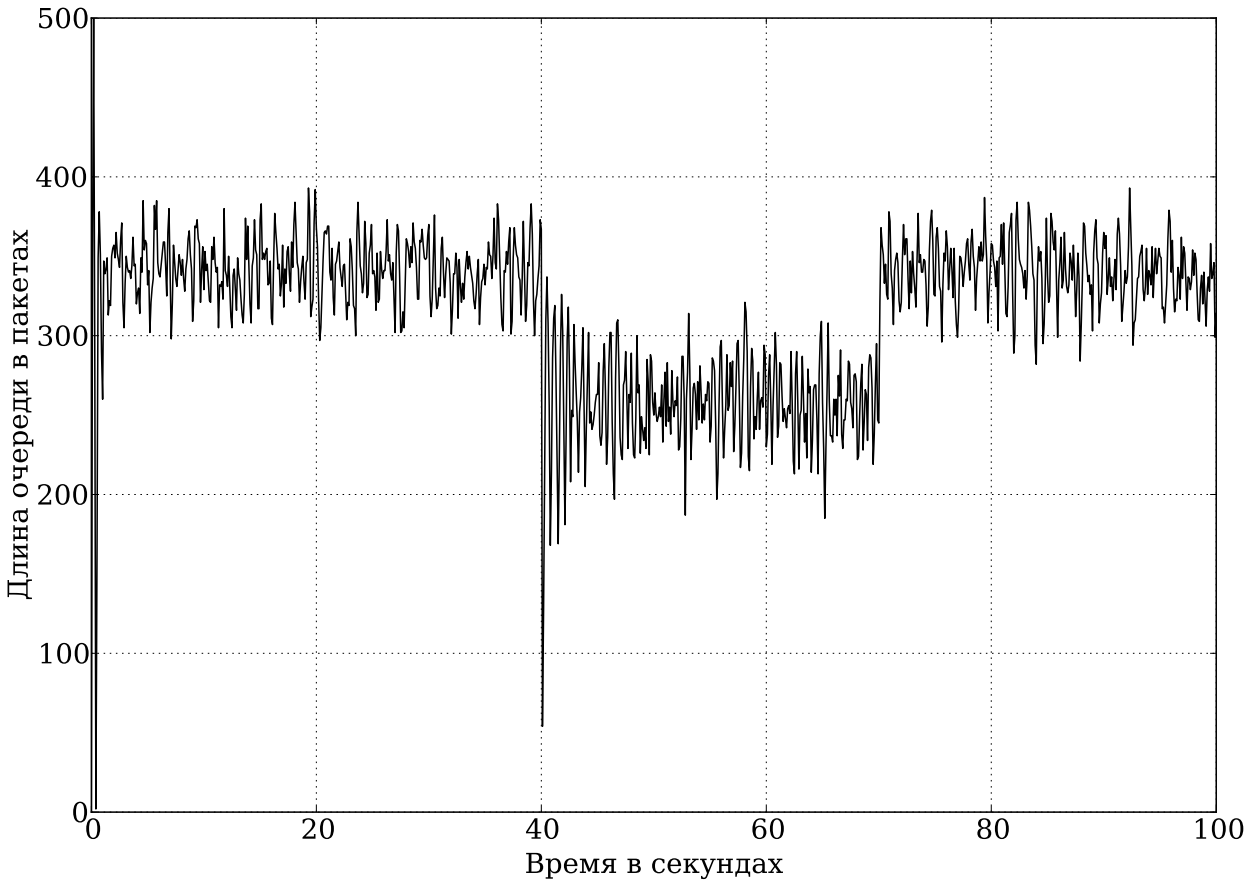


Рис. А.7 — Изменение длины очереди при использовании метода RED

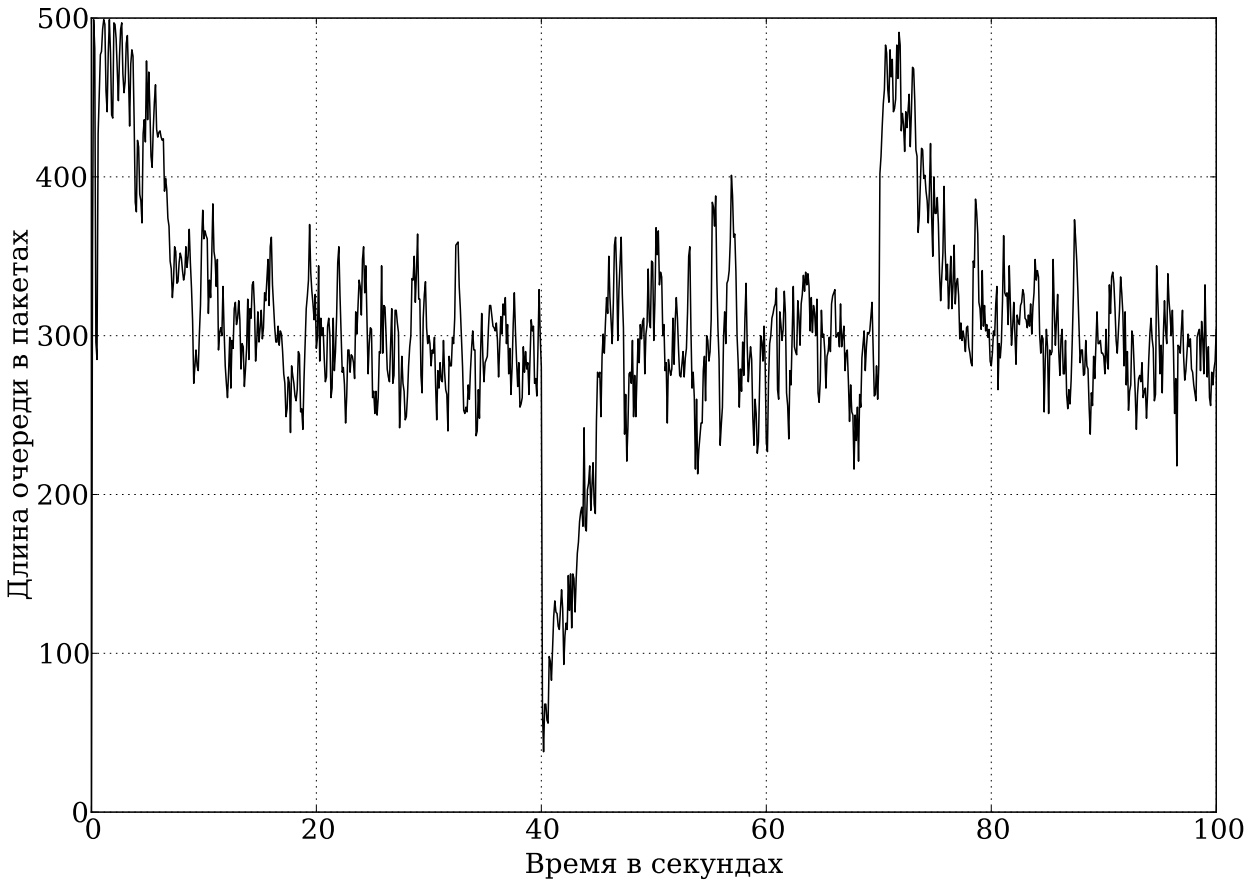


Рис. А.8 — Изменение длины очереди при использовании метода REM

Приложение Б

Акты использования результатов диссертационной работы

Б.1 Акт об использовании результатов в учебном процессе

УТВЕРЖДАЮ

Первый проректор-проректор

по учебной работе МТУСИ

_____ Титов Е.В.

_____ 2013

АКТ

об использовании результатов диссертационной работы Масленникова А.Г. на тему: «Разработка метода обработки трафика в очередях маршрутизаторов мультисервисной сети на основе нечеткой логики» в учебном процессе кафедры СС и СК МТУСИ

Комиссия в составе начальника Учебного управления Карпушиной Н.Д., заведующей Учебным отделом Патенченковой Е.К. и заведующего кафедрой Сети связи и системы коммутации (СС и СК) Пшеничникова А.П. составила настоящий акт о том, что подготовленная Масленниковым А.Г. лекция на тему «Методы активного управления очередями маршрутизаторов», составленная по материалам диссертационной работы, используется в учебном процессе кафедры СС и СК, а именно при чтении лекций по дисциплине «Перспективные сетевые телекоммуникационные технологии» для магистрантов, обучающихся по направлению подготовки магистров 210700 – Инфокоммуникационные технологии и системы связи.

Начальник Учебного управления **Карпушина Н.Д.** 31.10.2013

Заведующая Учебным отделом **Патенченкова Е.К.** 29.10.13

Заведующий кафедрой СС и СК **Пшеничников А.П.** 22.10.13

Б.2 Акт об использовании результатов в тестировании

DATATEL

УТВЕРЖДАЮ
Генеральный директор
ЗАО «ДАТАТЕЛ»
П.Б. Кузнецов

«17» декабря 2014 г.
М.П.

АКТ

об использовании результатов диссертационной работы А. Г. Масленникова на тему:
«Разработка метода обработки трафика в очередях маршрутизаторов мультисервисной сети на основе нечёткой логики»

Настоящим актом подтверждаем, что в ЗАО «ДАТАТЕЛ» в период с 15 по 17 декабря 2014 г. проводилось нагрузочное тестирование маршрутизатора ASUS WL-500gP V2 (s/n: BAIEG6001805) с установленной ОС Linux 2.4.37. Для управления трафиком и предотвращения перегрузок использовался модуль ядра Linux *sch_flc.o,* разработанный в диссертационной работе Масленникова Андрея Геннадьевича.

Данный маршрутизатор показал стабильную работу при 100% загрузке канала передачи данных. Тестирование проводилось с помощью программных генераторов трафика между двумя тестовыми компьютерами. Один из тестовых компьютеров был подключён по беспроводному каналу связи по стандарту IEEE 802.11g. В ходе эксперимента интенсивность трафика скачкообразно изменялась путём уменьшения количества TCP соединений с 50 до 25, и обратного увеличения до 50 соединений периодами по 100 с. На канале передачи данных, ограниченном скоростью 15 Мбит/с, и при средней задержке RTT=2,5 мс, средняя длина очереди в маршрутизаторе составила 308 Кбайт, при максимальном размере очереди 500 Кбайт, а среднеквадратичное отклонение длины очереди от среднего значения - 55 Кбайт. Общие потери пакетов - 0,027%. Значение джиттера для потока UDP, передающегося с постоянной скорость 128 кбит/с одновременно с TCP соединениями, составило 20 мс.

Планируем использовать данное решение в будущих проектах.

Технический эксперт, к.т.н.
Ю.Г. Писарев

Технический директор
О.В. Ребриков

Россия, 109240, Москва, ул. Верхняя Радищевская, д. 5, стр. 4
тел.: 7 (495) 915-3203, 7 (495) 915-0927, факс: 7 (495) 915-7950
e-mail: info@datatel.ru, http://www.datatel.ru