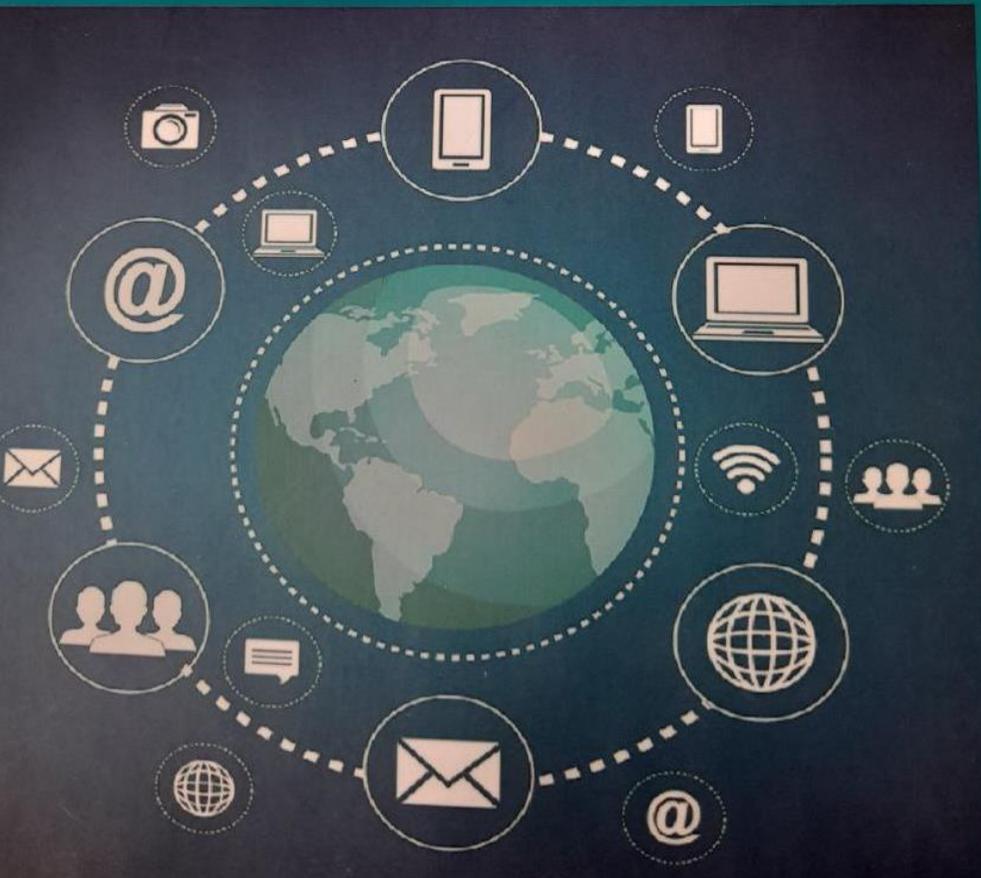


АМИРСАИДОВ У.Б.

ОПТИМИЗАЦИЯ СЕТЕЙ



МИНИСТЕРСТВО ВЫСШЕГО ОБРАЗОВАНИЯ, НАУКИ И
ИННОВАЦИЙ РЕСПУБЛИКИ УЗБЕКИСТАН

МИНИСТЕРСТВО ЦИФРОВЫХ ТЕХНОЛОГИЙ
РЕСПУБЛИКИ УЗБЕКИСТАН

ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ ИМЕНИ МУХАММАДА АЛ-ХОРАЗМИЙ

АМИРСАИДОВ У.Б.

ОПТИМИЗАЦИЯ СЕТЕЙ

Учебное пособие

Ташкент
"METODIST NASHRIYOTI"
2024

УДК: 004.7(075.8)
ББК: 32.973я7
А 620

Амирсаидов У.Б.
Оптимизация сетей./ Учебное пособие./ – Ташкент
“METHODIST NASHRIYOTI” 2024. - 168 с.

В учебном пособии приведены основы и задачи моделирования сетей телекоммуникации, рассмотрены типы моделирования, значение и роль теории систем массового обслуживания в телекоммуникации, проведен анализ типов и моделей систем массового обслуживания, освещены процессы моделирования систем и сетей передачи данных, рассмотрены вопросы оптимизации систем передачи данных, методы оптимизации процессов маршрутизации и оптимального построения виртуальных сетей.

Учебное пособие предназначено для магистрантов специальности 70611001- Телекоммуникационный инжиниринг (Системы передачи информации), а также для специалистов отрасли связи и информатизации.

Рецензенты:

- Курбовов Ж.Ф.** - Заведующий кафедрой «Автоматика и телемеханика» Ташкентского государственного транспортного университета, д.т.н., доцент.
- Парсиев С.С.** - Заведующий кафедрой «Аппаратное и программное обеспечение систем управления в телекоммуникации» Ташкентского университета информационных технологий имени Мухаммада ал-Хоразмий, д.т.н, доцент.

Публикуется на основе приказа № 439 от 25 апреля 2023 года Ташкентского университета информационных технологий имени Мухаммада ал-Хоразмий.

ISBN 978-9910-03-160-1

© Амирсаидов У.Б., 2024.
© “METHODIST NASHRIYOTI”, 2024.

ВВЕДЕНИЕ

Широкое распространение сетей телекоммуникации различного назначения ставит задачу совершенствования теоретической, технологической и методологической базы для их проектирования и анализа функционирования. Научно обоснованное решение такой задачи приводит к необходимости создания имитационных и аналитических моделей, методов и алгоритмов для выявления оптимальных способов организации телекоммуникационных систем. Эффективность функционирования сетей телекоммуникации определяется возможностью обеспечения технико-экономических показателей при минимальных затратах на их строительство и эксплуатацию. Еще на этапе проектирования важно понимать, какие глобальные задачи планируется решать с помощью сети. В первую очередь это зависит от рода деятельности предприятия. Например, операторы связи стремятся занять лидирующее положение на рынке услуг, поэтому для них важны показатели прибыли. Отсюда, как правило, следует, что в каждый момент времени сеть должна обладать наилучшими показателями производительности и максимальным функционалом, чтобы быть готовой для внедрения новых услуг и подключения новых клиентов.

В основе практически всех методов оптимизации лежат два основных принципа рассмотрения функциональных и структурных особенностей сетей телекоммуникации. Первоначально необходимо определиться с описанием сети телекоммуникации как системы массового обслуживания, и далее составить модель функционирования сети телекоммуникации в заданных условиях и с соблюдением существующих нормативно-правовых актов. Также необходимо согласовать сетевую архитектуру с протоколами и интерфейсами, участие которых предполагается при взаимодействии различных телекоммуникационных систем.

Базовой задачей оптимизации в области телекоммуникации является обобщенная задача поиска оптимальной сетевой структуры. Эта задача включает ряд подзадач, например, задачу поиска оптимального (кратчайшего) пути, задача поиска оптимальных значений пропускной способности и многие другие.

Решение задач оптимизации сетей телекоммуникации основывается на методологию системного анализа. Систе.

анализ — научный метод познания, представляющий собой последовательность действий по установлению структурных связей между переменными или элементами исследуемой системы. Ясно, что установление таких связей невозможно без знания характеристик элементов системы. Поэтому системный анализ подразумевает и изучение этих элементов с последующим объединением результатов исследования в единую систему знаний, которая позволяет решать ту или иную задачу управления с выработкой решения.

Процедура принятия решений является воплощением системного анализа, его основным содержанием и включает в себя формулировку проблемной ситуации, определение цели, определение критериев достижения цели, поиск оптимального решения, построение моделей для обоснования решений, поиск оптимального решения, реализацию решения. Системный анализ вполне приемлем для решения задач отрасли телекоммуникаций и информатизации. С его использованием можно успешно решать проблемы, связанные с обеспечением роста пропускной способности каналов и эффективности маршрутизации потоков, оптимизацией структуры сетей телекоммуникации, эффективного использования ресурсов сетей и информационных систем.

1. ОСНОВЫ МОДЕЛИРОВАНИЯ СЕТЕЙ ТЕЛЕКОММУНИКАЦИИ

1.1. Основные задачи моделирования

Модель — способ замещения реального объекта, используемый для его изучения [1,2]. Модель вместо исходного объекта используется в случаях, когда эксперимент опасен, дорог, происходит в неудобном масштабе пространства и времени (долговременен, слишком кратковременен, протяжен...), невозможен, неповторим, ненагляден и т. д. Проиллюстрируем это:

«эксперимент опасен» — при деятельности в агрессивной среде вместо человека лучше использовать его макет; примером может служить луноход;

«дорог» — прежде чем использовать идею в реальной экономике страны, лучше опробовать её на математической или имитационной модели экономики, просчитав на ней все «за» и «против» и получив представление о возможных последствиях;

«долговременен» — изучить коррозию — процесс, происходящий десятилетия, — выгоднее и быстрее на модели;

«кратковременен» — изучать детали протекания процесса обработки металлов взрывом лучше на модели, поскольку такой процесс скоротечен во времени;

«протяжен в пространстве» — для изучения космогонических процессов удобны математические модели, поскольку реальные полёты к звёздам (пока) невозможны;

«микроскопичен» — для изучения взаимодействия атомов удобно воспользоваться их моделью;

«невозможен» — часто человек имеет дело с ситуацией, когда объекта нет, он ещё только проектируется. При проектировании важно не только представить себе будущий объект, но и испытать его виртуальный аналог до того, как дефекты проектирования проявятся в оригинале. Важно: моделирование теснейшим образом связано с проектированием. Обычно сначала проектируют систему, потом её испытывают, потом снова корректируют проект и снова испытывают, и так до тех пор, пока проект не станет удовлетворять предъявляемым к нему требованиям. Процесс «проектирование-моделирование» цикличен. При этом цикл имеет вид спирали — с

Абстрактная модель воспроизводит систему с точки зрения её внутреннего устройства, копирует её более точно. У неё больше возможностей, шире класс решаемых задач.

Активные модели взаимодействуют с пользователем; могут не только, как пассивные, выдавать ответы на вопросы пользователя, когда тот об этом попросит, но и сами активируют диалог, меняют его линию, имеют собственные цели. Все это происходит за счёт того, что активные модели могут самоизменяться.

Статические модели описывают явления без развития. Динамические модели прослеживают поведение систем, поэтому используют в своей записи, например, дифференциальные уравнения, производные от времени.

Дискретные и непрерывные модели. Дискретные модели изменяют состояние переменных скачком, потому что не имеют детального описания связи причин и следствий, часть процесса скрыта от исследователя. Непрерывные модели более точны, содержат в себе информацию о деталях перехода.

Детерминированные и стохастические модели. Если следствие точно определено причиной, то модель представляет процесс детерминировано. Если из-за неизученности деталей не удастся описать точно связь причин и следствий, а возможно только описание в целом, статистически (что часто и бывает для сложных систем), то модель строится с использованием понятия вероятности.

Распределённые, структурные, сосредоточенные модели. Если параметр, описывающий свойство объекта, в любых его точках имеет одинаковое значение (хотя может меняться во времени!), то это система с сосредоточенными параметрами. Если параметр принимает разные значения в разных точках объекта, то говорят, что он распределён, а модель, описывающая объект, — распределённая. Иногда модель копирует структуру объекта, но параметры объекта сосредоточены, тогда модель — структурная.

Функциональные и объектные модели. Если описание идёт с точки зрения поведения, то модель построена по функциональному признаку. Если описание каждого объекта отделено от описания другого объекта, если описываются свойства объекта, из которых вытекает его поведение, то модель является объектно-ориентированной.

Каждый подход имеет свои достоинства и недостатки. Разные математические аппараты имеют разные возможности (мощность)

для решения задач, разные потребности в вычислительных ресурсах. Один и тот же объект может быть описан различными способами. Инженер должен грамотно применять то или иное представление, исходя из текущих условий и стоящей перед ним проблемы.

Приведённая выше классификация является идеальной. Модели сложных систем обычно имеют комплексный вид, используют в своём составе сразу несколько представлений. Если удастся свести модель к одному типу, для которого уже сформулирована алгебра, то исследование модели, решение задач на ней существенно упрощается, становится типовым. Для этого модель должна быть различными способами (упрощением, переобозначением и другими) приведена к каноническому виду, то есть к виду, для которого уже сформулирована алгебра, её методы. В зависимости от используемого типа модели (алгебраические, дифференциальные, графы и т. д.) на разных этапах её исследования используются различные математические аппараты.

Конечно, моделирование, как уже было сказано, в соединении с проектированием — это технология решения проблем, задач. Но у каждой технологии все-таки есть граница, за которой она менее эффективна. Такая граница есть и здесь. Первые этапы моделирования решают менее формализованные задачи, а последующие — все более формальные. Соответственно, методы первых этапов менее формализованы, а последующих — более формальные, мощные. Это означает, что самые трудные и ответственные этапы для моделировщика — первые. Здесь от него требуется больше интуитивных решений. И ошибка на более ранних этапах больше сказывается на дальнейших решениях, возвращаться и переделывать приходится гораздо больше, чем на последних этапах. Поэтому удачные решения на первых этапах вызывают пристальный интерес системотехников, наука моделирования проявляет к ним повышенное внимание. Поскольку формальные методы легко автоматизируются, то последние этапы схемы поддержаны программными продуктами и легко доступны конечным пользователям, но наибольший интерес сегодня представляют программные продукты, поддерживающие первые этапы — системы, помогающие формализовать задачи. А также системы, обеспечивающие сквозное проектирование, доведённое до моделирования и конечной реализации (автоматическое порождение кода по описанию проекта).

1.3. Концептуальная модель сетей телекоммуникации

В рекомендации ITU-T серии Y (Y.2001 и Y.2011) предложена модель инфокоммуникационной системы, которая включает четыре основных компонента (рис. 1.1) [5]:

- сеть в помещении пользователя (Customer Premises Network), которая может состоять как из одного терминала (Terminal Equipment, TE), так и из нескольких;
- сеть доступа (Access Network), которая обеспечивает подключение оборудования, находящегося в помещении пользователя, к транспортной сети;
- транспортная (базовая) сеть (Core Network), состоящая из совокупности коммутационных узлов и станций, обеспечивающих коммутацию и прозрачную передачу информации пользователя, а также обеспечивающую выход к средствам поддержки иных инфокоммуникационных услуг;
- средства поддержки услуг, обеспечивающие предоставление инфокоммуникационных услуг, управление услугами и приложениями.

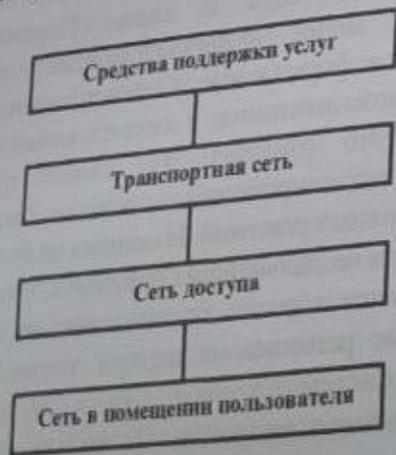


Рис. 1.1. Модель инфокоммуникационной сети

Сквозное качество обслуживания пользователей определяется рядом факторов, среди которых качество сетевого соединения, специфические для конкретного приложения действия, аспекты

восприятия самого пользователя. В соответствии с перечисленным можно выделить три уровня качества обслуживания [6-7]:

- качество работы сети (Network Performance, NP), соответствующее на уровне сети;
- качество инфокоммуникационных услуг (обслуживания) (Quality of Service, QoS) на уровне приложения;
- воспринимаемое качество инфокоммуникационной услуги (Quality of Experience, QoE) на уровне пользователя.

Сквозная оценка качества обслуживания пользователей и точки измерения параметров качества приведены соответственно на рис. 1.2 и рис. 1.3.

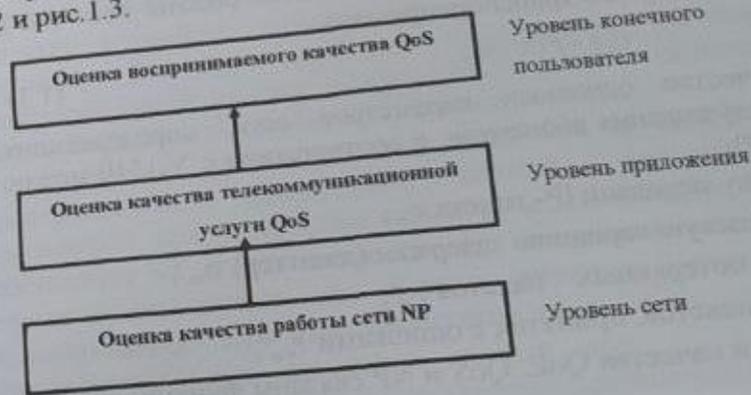


Рис. 1.2. Оценка качества обслуживания

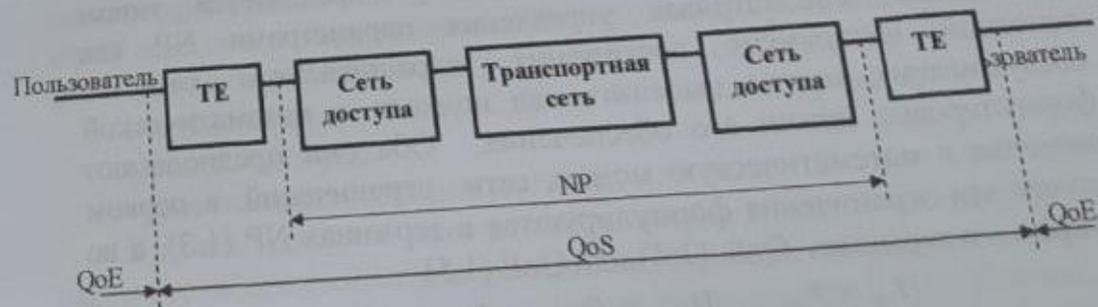


Рис. 1.3. Точки оценки качества обслуживания

Оценка QoE представляет собой количественную выраженную степень удовлетворенности конечного пользователя качеством

полученной инфокоммуникационной услуги. На достижение хороших значений именно этих оценок направлены все усилия операторов связи. Это интегральная оценка качества, выражаемая в технической терминологии, а в некоторых баллах. Рекомендациями ITU-T определены несколько таких оценок, среди которых в качестве основных следует выделить [7]:

- рейтинг качества R (Quality Rating), $0 \leq R \leq 100$;
- среднюю экспертную оценку MOS (Mean Opinion Score), $1 \leq MOS \leq 5$;

Показатели QoS являются функцией качества работы терминального оборудования (TE) и качества работы сети (NP), включая сети доступа и транспортную сеть

$$QoS = F(TE, NP) \quad (1.1)$$

В качестве основных параметров сети, определяющих качество обслуживания абонентов, в соответствии с Y.1540 можно выделить [6, 7]:

- задержку передачи IP- пакета T_{NP} ;
- межконцевую вариацию задержки (джиттер) D_{NP} ;
- процент потерянных пакетов P_{NP} ;
- процент пакетов, принятых с ошибками P_{ENP} ;

Показатели качества QoE, QoS и NP связаны функциональной зависимостью

$$QoE = F(QoS, NP) \quad (1.2)$$

Функциональная зависимость (1.2) определяется типом приложения. Рассматривая управление параметрами NP как основной инструмент достижения запрашиваемого качества обслуживания, можно выделить два подхода к математической формулировке задачи его обеспечения. Оба они предполагают введение в математическую модель сети ограничений, в первом случае эти ограничения формулируются в терминах NP (1.3), а во втором – в терминах QoS (1.4) или QoE (1.5)

$$T_{NP} \leq T_{NP\text{треб}}; D_{NP} \leq D_{NP\text{треб}}; P_{NP} \leq P_{NP\text{треб}}, \quad (1.3)$$

$$T_{QoS} \leq T_{QoS\text{треб}}; D_{QoS} \leq D_{QoS\text{треб}}; P_{QoS} \leq P_{QoS\text{треб}}, \quad (1.4)$$

$$MOS \geq MOS_{\text{треб}} \text{ или } R \geq R_{\text{треб}}. \quad (1.5)$$

В общем виде задача гарантированного качества обслуживания может быть сформулирована как оптимизационная, связанная с поиском экстремума некоторого, как правило, стоимостного функционала (например, максимизация прибыли оператора связи) при наличии ограничений (1.3) или (1.4), (1.5). Для решения данной оптимизационной задачи необходимо разработать аналитическую или имитационную модель функционирования сети.

Современные сети строятся на основе технологии коммутации пакетов. Очереди являются неотъемлемым атрибутом сетей с коммутацией пакетов. Принцип работы таких сетей подразумевает наличие буфера у каждого входного и выходного интерфейсов сетевых устройств (коммутаторов, маршрутизаторов, мостов, шлюзов). Поэтому наиболее подходящим математическим аппаратом исследования современных сетей является теория очередей (теория массового обслуживания) [8,9].

Процесс моделирования сетей, как и любой сложной, пространственно разнесенной системы является сложной процедурой. Для упрощения процесса моделирования вес процесс разбивается на отдельные этапы. На каждом из этапов ставится определенная задача из общего перечня решаемых задач моделирования. При этом результаты i -го этапа моделирования используются как исходные данные для $i+1$ -го этапа. Использование такого подхода (метода декомпозиции) позволяет упростить процесс моделирования.

В соответствии с методом декомпозиции и теории массового обслуживания обобщенную математическую модель сети NGN можно представить как взаимосвязанную совокупность моделей сетей массового обслуживания (SeMO), моделирующие процессы функционирования компонентов (сегментов) сети (рис.4).

Модель сети доступа (МСД) состоит из двух уровней:

- уровень доступа;
- уровень агрегирования доступа.

Уровень доступа обеспечивает подключение пользователей (частных абонентов, предприятий, мобильных пользователей) к сети. В зависимости от используемой технологии (xDSL, Ethernet, PON, Cable, Wi-Fi, WiMax) выбирается оборудование: DSLAM, коммутаторы Ethernet, точки доступа Wi-Fi, базовые станции WiMax. На уровне агрегирования происходит сбор и доставка трафика, поступающего с уровня доступа, к устройствам

распределения магистральной сети. Уровень агрегирования состоит из сети маршрутизаторов, развернутой в масштабах города или области с применением технологий Ethernet, IP, WDM.

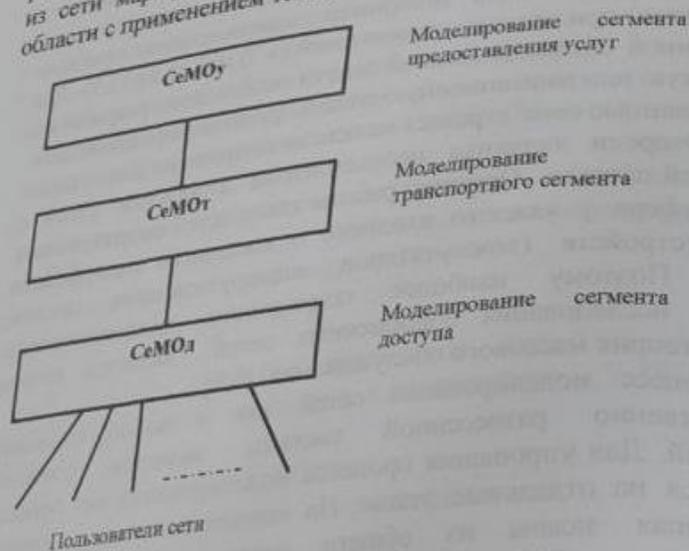


Рис. 1.4. Комплексная модель сети

Модель уровня доступа сети

Сети с полностью оптоволоконным доступом (All-fiber) становятся реальностью, и в настоящее время быстро увеличивается доля различных вариантов сетевого доступа, полностью реализуемого на основе оптоволокна. Конечной целью является построение сети с полностью оптоволоконным доступом (FTTH/GPON), однако при этом принимается во внимание, что еще не исчерпан потенциал многих существующих подключений на последней мили. Разработаны различные сценарии гибридного доступа FTTh по оптоволоконным и медным линиям связи. Узкое определение "FTTh" охватывает два различных типа гибридного доступа: "оптоволоконно до распределительного шкафа" (Fiber-to-the-Curb: FTTC) и "оптоволоконно до здания" (FTTB).

Разработаны нижеперечисленные типовые решения по гибриднему доступу "оптоволоконно-медь":

- FTTC с VDSL2;

- FTTC с ADSL2+;
- FTTB с VDSL2;
- FTTB с ADSL2+;
- FTTB с Ethernet (ETTH).

Техническое решение FTTC с VDSL2 обеспечивает скорость передачи данных до 100 Мбит/с (на расстоянии до 1 км) для отдельного подключения конечного пользователя. Такая пропускная способность позволяет предложить полную услугу Triple Play в этом варианте доступа с использованием существующей проводки на последней миле. При этом обычные аналоговые подключения могут быть преобразованы в подключения, основанные на IP. Абоненты подключаются к узлу доступа (MSAN), который размещается в находящемся неподалеку уличном шкафу.

Техническое решение FTTB с VDSL2 обеспечивает подключение абонентов в многоквартирных домах с применением узла доступа MSAN с платами VDSL2. Устанавливаемый в здании узел доступа размещается в компактном защитном контейнере. Высокоскоростное подключение VDSL2 обеспечивает скорость передачи до 100 Мбит/с.

Техническое решение FTTB с Ethernet (ETTH) обеспечивает подключение абонентов в многоквартирных домах с использованием коммутаторов Ethernet и устанавливаемого в здании узла доступа MSAN, функционирующего в качестве агрегирующего коммутатора (узла коммутаторов). По мере увеличения количества пользователей, узлы коммутаторов могут устанавливаться на каждом этаже или в каждом подъезде.

Сети в помещении пользователей (Customer Premises Network) радикально различаются по требованиям к обслуживанию (от традиционной телефонной связи до функциональных возможностей Triple-Play Services). Продукты в семействе домашних шлюзов (шлюзы xDSL, шлюзы Ethernet, оконечные сетевые устройства FTTH) реализуют механизмы обеспечения безопасности и качества обслуживания.

Разрабатываемая модель мультисервисной сети доступа (МСД) должна обеспечивать подключение всех используемых в настоящее время и в обозримой перспективе мультимедийных терминалов (домашних шлюзов и серверов), а также выход в базовую сеть, основанные на различных технологиях коммутации. Основная задача – создание эффективных аппаратно-программных средств,

базе которых может быть реализован узел доступа. Этот узел должен поддерживать несколько интерфейсов, различные протоколы сигнализации, обеспечивать экономичное введение услуг и отвечать всем требованиям качества обслуживания для мультисервисного трафика.

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ГЛАВЕ 1

1. Какие задачи решаются при моделировании систем?
2. В чем разница между алгоритмом и моделью?
3. Этапы моделирования систем.
4. Какова схема моделирования?
5. Объясните типы моделей.
6. Рекомендации по оценке качества обслуживания.
7. Объясните функцию магистральной сети.
8. Объясните структуру магистральной сети.
9. Каким образом сеть доступа представлена как система массового обслуживания?
10. Каким образом магистральная сеть представлена как система массового обслуживания?
11. Что лежит в основе аналитического моделирования?
12. Что лежит в основе имитационного моделирования?
13. Охарактеризуйте показатели качества в телекоммуникациях?
14. Какие показатели качества обслуживания вы знаете?
15. Какие требования предъявляются к методам моделирования систем?

2. МОДЕЛИ СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ

2.1. Структура, параметры и характеристики систем массового обслуживания

Системой массового обслуживания (СМО) называется система, процесс функционирования которой является, по сути, процессом обслуживания, который состоит в предоставлении той или иной услуги, определяемой из функционального назначения системы.

Для формализации любой СМО необходимо описать:

- процесс поступления заявок в систему;
- процесс обслуживания заявок в системе;
- дисциплину обслуживания.

Модели потоков заявок

Поступающие на вход системы массового обслуживания требования (заявки, запросы) образуют поток дискретных событий, полностью определяемый множеством моментов времени их поступления $\varepsilon = \{t_n\}$. Для детерминированного потока значения t_n задаются таблицей или формулой. На практике этот поток случайный и значения моментов поступления запросов есть значения случайной величины, задаваемой функциями распределения вероятности t_n , либо интервала между поступлениями $\Delta t: t = t_n - t_{n-1}$.

В зависимости от вида функции распределения вероятности потоки требований наделяют соответствующими названиями. В общем случае случайные потоки можно классифицировать по наличию или отсутствию трех основных свойств: стационарности, последствия и ординарности [8].

Стационарность - независимость вероятностных характеристик от времени. Так вероятность поступления определенного числа требований в интервал времени длиной t для стационарных потоков не зависит от выбора начала его измерения.

Последствие - вероятность поступления требований в интервале (t_1, t_2) зависит от событий, произошедших до момента t_1 .

Ординарность - вероятность поступления двух и более требований за бесконечно малый интервал времени Δt есть величина бесконечно малая более высокого порядка, чем Δt .
К основным характеристикам случайных потоков относят интенсивность потока.

Пуассоновский (простейший) поток запросов

Стационарный ординарный поток без последствия называют простейшим. Он задается набором вероятностей $P_i(t)$ поступления i требований в промежутке длиной t .
Можно показать, что при этих предположениях формула для $P_i(t)$ дается формулой Пуассона (Poisson)[8]:

$$P_i(t) = \frac{(\lambda t)^i}{i!} e^{-\lambda t} \quad (2.1)$$

Если рассмотреть закон распределения вероятностей промежутка между поступлением соседних требований τ , то можно показать, что

$$P(\tau \leq t) = 1 - P_0(t) = 1 - e^{-\lambda t} \quad (2.2)$$

Дифференцируя, получаем плотность распределения вероятностей:

$$p(t) = \lambda e^{-\lambda t} \quad (2.3)$$

Случайная величина с такой плотностью вероятностей называется экспоненциально - распределенной (с показательным распределением). Математическое ожидание экспоненциально распределенной случайной величины равно:

$$M(\tau) = \bar{\tau} = \int_0^{\infty} \tau p(\tau) d\tau = \int_0^{\infty} \tau \lambda e^{-\lambda \tau} d\tau = 1/\lambda, \quad (2.4)$$

а дисперсия и среднее квадратическое отклонение соответственно будут равны:

$$D(\tau) = \int_0^{\infty} \tau^2 p(\tau) d\tau - \bar{\tau}^2 = \int_0^{\infty} \tau^2 \lambda e^{-\lambda \tau} d\tau - 1/\lambda^2 = 1/\lambda^2, \quad (2.5)$$

$$\sigma_{\tau} = \sqrt{D(\tau)} = 1/\lambda. \quad (2.7)$$

Коэффициент вариации равен:

$$v_{\tau} = \frac{\sigma_{\tau}}{M(\tau)} = 1.$$

Процесс обслуживания

По аналогии с процессами поступления заявок в систему для описания процессов обслуживания необходимо задать функцию распределения $B_k(t)$ длительности обслуживания для каждой k -й заявки ($k = 1, 2, 3, \dots$), которая в общем случае является случайной величиной. При этом под длительностью обслуживания τ_k понимается промежуток времени, в течение которого заявка находится в обслуживающем приборе. Далее будем считать, что все заявки создают статистически однородную нагрузку, т.е. длительности обслуживания всех заявок распределены по одному и тому же закону:

$$B_k(t) = B(t) = \Pr\{\tau_k \leq t\}, \quad k = 1, 2, 3, \dots \quad (2.9)$$

Важной характеристикой процесса обслуживания является интенсивность обслуживания μ , характеризующая среднее число заявок, обслуживаемых системой в единицу времени.

Величина b , обратная интенсивности μ ($b = 1/\mu$), определяет среднее время обслуживания одной заявки. Как и в случае интервалов поступления, если функция распределения $B(t)$ неизвестно, то для многих приложений (теоретических и практических) оказывается достаточным определить интенсивность обслуживания μ (или среднее время обслуживания b) и коэффициент вариации v_{τ} длительности обслуживания. Если длительность обслуживания распределена по экспоненциальному закону, то достаточно задать интенсивность обслуживания μ (или среднее время обслуживания b). Следует отметить, что, в отличие от интервалов поступления заявок, отказ от экспоненциального характера распределения длительности их обслуживания не столь усложняет задачу аналитического исследования СМО, и многие содержательные результаты получены при произвольном характере распределения времени обслуживания.

Дисциплина обслуживания

Дисциплиной обслуживания (ДО) называется правило, по которому выбираются на обслуживание заявки из очереди [9]. Различают следующие ДО:

- 1) обслуживание в порядке поступления или дисциплина FIFO (FirstInput, FirstOutput — первым пришел, первым ушел);
- 2) обслуживание в обратном порядке или дисциплина LIFO (LastInput, FirstOutput — последним пришел, первым ушел);
- 3) обслуживание в случайном порядке, когда заявка на обслуживание выбирается случайно среди ожидающих заявок.

В дальнейшем в качестве ДО будем рассматривать ДО FIFO. Таким образом, для описания СМО необходимо задать:

- 1) функцию распределения $A(t)$ интервалов поступления (общий случай) или интенсивность поступления λ (или средний интервал $a=1/\lambda$) и коэффициент вариации v_a интервалов поступления;
- 2) функцию распределения $B(t)$ длительности обслуживания (общий случай) или интенсивность обслуживания μ (или среднее время обслуживания $b=1/\mu$) и коэффициент вариации v_b времени обслуживания;

3) дисциплина обслуживания (ДО FIFO).

Следует отметить, что на практике СМО описывается, как правило, путем определения совокупности параметров $\{\lambda, v_a\}$ и $\{\mu, v_b\}$, считая, что ДО по умолчанию является дисциплина FIFO. Более того, если интервалы поступления или длительности обслуживания распределены по экспоненциальному закону, то нет необходимости задавать и соответствующий коэффициент вариации, т.к. в таком случае он равен 1.

Характеристики СМО

1) Загрузка системы — это отношение интенсивности поступления λ к интенсивности обслуживания μ и обозначается через ρ :

$$\rho = \lambda/\mu = \lambda b = b/a, \quad (2.10)$$

где $a=1/\lambda$ и $b=1/\mu$ — средние значения интервалов поступления и длительности обслуживания соответственно.

Значение загрузки определяет условие существования в системе стационарного режима. Необходимым и достаточным условием существования в стохастической СМО стационарного режима является условие, когда $\rho < 1$ или $\lambda < \mu$. Выполнение этого условия означает, что система в среднем справляется с поступающей нагрузкой. Если $\rho \geq 1$, то система работает в режиме перегрузок.

2) Время ожидания — это, как правило, случайное время, которое заявка проводит в очереди в состоянии ожидания. Среднее значение этого времени, которое представляет наибольший интерес, обозначается через ω .

3) Время пребывания — это случайный промежуток времени от момента поступления заявки в систему до момента окончания ее обслуживания. Для среднего значения u и времени пребывания справедливо равенство:

$$u = \omega + b. \quad (2.11)$$

3) Среднее число заявок в очереди или средняя длина очереди

$$l = \lambda \omega. \quad (2.12)$$

5) Среднее число заявок m , находящихся в системе, складывается из средних значений числа заявок, находящихся в очереди (l) и в приборе (ρ):

$$m = l + \rho = \lambda \omega + \lambda b = \lambda(\omega + b) = \lambda u. \quad (2.13)$$

Формулы $\rho = \lambda b$, $l = \lambda \omega$ и $m = \lambda u$ называются формулами Литтла соответственно для прибора, очереди и системы в целом.

Зная среднее число заявок в системе (m) и в очереди (l), соответствующие временные характеристики можно определить по формуле Литтла:

$$u = m/\lambda \quad \text{и} \quad \omega = l/\lambda = u - b. \quad (2.14)$$

Полученные соотношения взаимосвязи между характеристиками функционирования системы справедливы при любых законах распределений интервалов поступления длительности обслуживания заявок и таким образом носят фундаментальный (универсальный) характер. Единствен

требование — это требование, чтобы система была без отказов, т.е. емкость накопителя была не ограничена.

Обозначения СМО (символика Кендалла)

Для компактного описания систем массового обслуживания часто используются обозначения, предложенные Д. Кендаллом, в виде [8, 9]:

$A/B/N/L$, где A и B — задают законы распределений соответственно интервалов времени между моментами поступления заявок в систему и длительности обслуживания заявок в приборе; N — число обслуживающих приборов в системе ($N = 1, 2, \dots$); L — число мест в накопителе, которое может принимать значения $0, 1, 2, \dots$ (отсутствие L означает, что накопитель имеет неограниченную емкость).

Для задания законов распределений A и B используются следующие обозначения:

G (General) — произвольное распределение общего вида;

M (Markovian) — экспоненциальное (показательное) распределение;

D (Deterministik) — детерминированное распределение;

U (Uniform) — равномерное распределение;

Ek (Erlangian) — распределение Эрланга k -го порядка (с k последовательными одинаковыми экспоненциальными фазами);

hk (hipoexponential) — гипоэкспоненциальное распределение k -го порядка (с k последовательными разными экспоненциальными фазами);

Hr (Hyperexponential) — гиперэкспоненциальное распределение порядка r (с r параллельными экспоненциальными фазами);

g (gamma) — гамма-распределение;

P (Pareto) — распределение Парето и т.д.

Примеры:

$M/M/1$ — одноканальная СМО с накопителем неограниченной емкости, в которую поступает однородный поток заявок с экспоненциальным распределением интервалов времени между последовательными заявками (простейший поток) и экспоненциальной длительностью обслуживания заявок в приборе.

$M/G/3/10$ — трёхканальная СМО с накопителем ограниченной емкости, равной 10, в которую поступает однородный поток заявок с экспоненциальными заявками (простейший поток) и длительностью обслуживания заявок, распределённой по закону общего вида.

$D/E_2/7/0$ — семиканальная СМО без накопителя (ёмкость накопителя равна 0), в которую поступает однородный поток заявок с детерминированными интервалами времени между последовательными заявками (детерминированный поток) и длительностью обслуживания заявок в приборе, распределённой по закону Эрланга 2-го порядка.

Для обозначения более сложных СМО дополнительно могут использоваться обозначения, описывающие неоднородный поток заявок и приоритеты между заявками разных классов.

Характеристики сетей массового обслуживания (СМО)

При исследовании сети необходимо задавать топологическую структуру сети, так как она определяет возможные переходы заявок между узлами. Требуется также описать маршруты отдельных заявок и вероятностные модели потоков заявок между узлами сети [11, 12].

Пусть сеть содержит n узлов. В каждый узел i поступает пуассоновский поток заявок с интенсивностью γ_i . Покидая i -узел, заявка с вероятностью p_{ij} поступает в j узел.

Обозначим λ_i полную интенсивность потока, поступающего в i -й узел, можно показать, что должно выполняться условие баланса:

$$\lambda_i = \gamma_i + \sum_{j=1}^n \lambda_j p_{ji}, \quad i = 1, 2, \dots, n. \quad (2.15)$$

Вероятность, того, что заявка после обслуживания в i -том узле вообще покинет сеть будет равна $1 - \sum_{j=1}^n p_{ij}$.

Выполнение условия эргодичности марковской модели каждого узла будет обеспечено, если $\lambda_i < \mu_i$.

Джексоном было доказано далеко не тривиальное утверждение, что каждый узел в сети ведет себя так, как если бы он был независимой СМО типа $M/M/m$ с входящим пуассоновским потоком λ_i [9]. В общем случае полный входящий поток не является пуассоновским. Состояние сети с n узлами описывается вектором

компонентами которого являются число заявок в каждом из узлов сети (k_1, k_2, \dots, k_n) .

Джексоу удалось доказать, что стационарная вероятность этого состояния разлагается в произведение безусловных распределений:

$$p(k_1, k_2, \dots, k_n) = p_1(k_1) p_2(k_2) \dots p_n(k_n). \quad (2.16)$$

Исследование сети с помощью изложенной здесь методики приводит к громоздким системам уравнений для определения интенсивностей входных потоков. Исследованиями было показано, что для получения многих результатов можно, вместо условий глобального равновесия для сети в целом применять условия локального равновесия для отдельных подсистем, что позволяет сильно упростить задачу.

Для описания линейных разомкнутых и замкнутых однородных экспоненциальных сетей массового обслуживания используется следующая совокупность параметров [8,9]:

- число узлов в сети: n ;
- число обслуживающих приборов в узлах сети: K_1, \dots, K_n ;
- матрица вероятностей передач: $P = [p_{ij} | i, j = 0, 1, \dots, n]$, где p_{ij} — вероятность передачи заявки из узла i в узел j ;
- интенсивность λ_0 источника заявок, поступающих в разомкнутую СеМО (РСеМО), или число заявок M , циркулирующих в замкнутой СеМО (ЗСеМО);
- средние длительности обслуживания заявок в узлах сети: b_1, \dots, b_n .

Для линейных СеМО элементы матрицы вероятностей передач должны удовлетворять условию:

$$\sum_{j=0}^n p_{ij} = 1 \quad (j = 0, n). \quad (2.17)$$

Это условие отражает тот факт, что любая заявка, покинувшая некоторый узел, обязательно (с вероятностью 1) перейдет в какой-то узел, включая тот же самый или нулевой. Переход заявки в нулевой узел означает, что заявка покинула сеть.

На основе узловых характеристик рассчитываются сетевые характеристики СеМО:

- суммарная загрузка всех узлов СеМО, характеризующая среднее число параллельно работающих узлов сети;

$$R = \sum_{i=1}^n \rho_i, \quad (2.18)$$

где ρ_i — загрузка узла i .

- среднее число заявок, находящихся в очередях всех узлов сети и ожидающих обслуживания:

$$L = \sum_{i=1}^n l_i, \quad (2.19)$$

где l_i — средняя длина очереди заявок в узле i ;

- среднее число заявок, находящихся в сети:

$$M = \sum_{i=1}^n m_i, \quad (2.20)$$

где m_i — среднее число заявок в узле i .

- среднее время ожидания заявок в сети:

$$W = \sum_{i=1}^n \alpha_i w_i, \quad (2.21)$$

где w_i — среднее время ожидания заявок в узле i ; α_i — коэффициент передачи для узла i , показывающий среднее число попаданий заявки в узел i за время её нахождения в сети.

- среднее время пребывания заявок в сети:

$$U = \sum_{i=1}^n \alpha_i u_i. \quad (2.22)$$

2.2. Модели одноканальных систем массового обслуживания

Предположим, что задана СМО общего вида (типа $G/G/1$), для которой определены параметры нагрузки, а, именно, интенсивность λ и коэффициент вариации (v_a) интервалов поступления, интенсивность обслуживания μ и коэффициент вариации (v) длительности обслуживания:

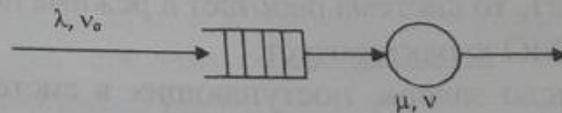


Рис.2.1. Структура СМО.

Основными характеристиками, определяющими качество функционирования такой СМО, являются:

- 1) вероятности состояний системы;
- 2) загрузка или коэффициент использования системы;

- 3) время ожидания заявок в системе;
- 4) время пребывания в системе;
- 5) число заявок в очереди системы или длина очереди;
- 6) число заявок в системе.

Следует отметить, что все перечисленные характеристики имеют смысл только в том случае, когда система функционирует в установленном режиме (без перегрузок), что и предполагается далее. Кроме того, последние четыре характеристики являются случайными величинами ("3" и "4" — непрерывные, "5" и "6" — дискретные) и полный анализ этих характеристик предполагает определение соответствующих функций распределения. Однако в большинстве практических приложений достаточно анализировать данные характеристики на уровне их средних значений, что и делается далее.

Остановимся на перечисленных характеристиках более подробно.

1) Вероятности состояний системы — это наиболее полная характеристика системы в том смысле, что, зная вероятности состояний, можно определить все остальные характеристики. При этом под состоянием СМО понимается число заявок, находящихся в системе. Вероятность состояния системы, когда в ней находится k заявок, обозначим далее через P_k , $k=0, 1, 2, \dots$

2) Загрузка системы — это отношение интенсивности поступления λ к интенсивности обслуживания μ и обозначается через ρ (2.10)

Значение загрузки определяет условие существования в системе стационарного режима. Необходимым и достаточным условием существования в стохастической СМО стационарного режима является условие, когда $\rho < 1$ или $\lambda < \mu$. Выполнение этого условия означает, что система в среднем справляется с поступающей нагрузкой. Если $\rho \geq 1$, то система работает в режиме перегрузок.

Загрузка ρ СМО характеризует:

- а) среднее число заявок, поступающих в систему за среднее время обслуживания одной заявки;
- б) долю времени, в течение которого прибор занят обслуживанием;
- в) вероятность того, что прибор занят обслуживанием заявок;

г) среднее число заявок, находящихся в обслуживающем приборе.

Перечисленные утверждения составляют физический смысл загрузки.

Справедливость утверждения "а" следует из определения загрузки $\rho = \lambda b$: если λ — среднее число заявок, поступающих в единицу времени, то за время b в систему поступят в среднем λb заявок.

Справедливость утверждения "б" можно показать следующими простыми рассуждениями. Рассмотрим достаточно длинный интервал t времени функционирования системы. Для простоты предположим, что в начале и в конце этого интервала система была свободна. Очевидно, что за время t в систему в среднем поступят λt заявок. Каждая из этих заявок в среднем обслуживается за время b . Тогда суммарное время обслуживания всех заявок равно $\lambda t b$. Отсюда доля времени, в течение которого прибор занят обслуживанием заявок, равна $\lambda t b / t = \lambda b = \rho$, что и следовало показать.

Утверждение "в" напрямую следует из утверждения "б", ибо рассмотренная ранее доля времени и есть вероятность занятости прибора. Тогда вероятность простоя системы равна $1 - \rho$.

Справедливость утверждения "г", в свою очередь, следует из утверждения "в": в приборе может находиться 1 заявка с вероятностью ρ и 0 заявок с вероятностью $1 - \rho$. Тогда среднее число заявок в приборе равно $1 \cdot \rho + 0 \cdot (1 - \rho) = \rho$.

3) Время ожидания — это, как правило, случайное время, которое заявка проводит в очереди в состоянии ожидания. Среднее значение этого времени, которое представляет наибольший интерес, обозначается через ω .

4) Время пребывания — это случайный промежуток времени от момента поступления заявки в систему до момента окончания ее обслуживания. Для среднего значения u времени пребывания справедливо равенство (2.11)

5) Среднее число заявок в очереди или средняя длина очереди (2.12).

Формулы $\rho = \lambda b$, $l = \lambda \omega$ и $m = \lambda u$ называются формулами Литтла соответственно для прибора, очереди и системы в целом.

$$\begin{aligned}
 -\lambda p_0 + \lambda p_1 &= 0 \\
 \lambda p_0 - (\lambda + \mu) p_1 + \mu p_2 &= 0 \\
 \lambda p_{i-1} - (\lambda + \mu) p_i + \mu p_{i+1} &= 0
 \end{aligned}
 \tag{2.28}$$

Решая систему алгебраических уравнений получаем:

$$\begin{aligned}
 p_1 &= \frac{\lambda}{\mu} p_0 = \varphi p_0 \\
 p_2 &= \varphi^2 p_0 \\
 &\dots \\
 p_n &= \varphi^n p_0
 \end{aligned}
 \tag{2.29}$$

Суммируя полученные вероятности находим с использованием формулы суммы геометрической прогрессии и с учетом равенства

$$\sum_{n=0}^{\infty} p_n = 1:$$

$$p_0 + p_0 \varphi + p_0 \varphi^2 + \dots + p_0 \varphi^n + \dots = p_0 (1 + \varphi + \varphi^2 + \dots + \varphi^n + \dots) = \frac{p_0}{1 - \varphi} = 1,
 \tag{2.30}$$

откуда:

$$p_0 = 1 - \varphi$$

$$p_n = \varphi^n (1 - \varphi)$$

В установившемся состоянии находим $T_{сист}$ и $N_{сист}$:

$$\begin{aligned}
 N_{сист} &= \sum_{n=0}^{\infty} n p_n = p_1 + 2p_2 + 3p_3 + \dots + n p_n + \dots = p_0 \varphi + 2p_0 \varphi^2 + \dots + n p_0 \varphi^n + \dots = \\
 &= p_0 \varphi (1 + 2\varphi + 3\varphi^2 + \dots + n\varphi^{n-1} + \dots),
 \end{aligned}$$

поскольку:

$$(\varphi + \varphi^2 + \varphi^3 + \dots + \varphi^n + \dots) = [\varphi(1 + \varphi + \varphi^2 + \dots + \varphi^{n-1} + \dots)] \approx \left(\frac{\varphi}{1 - \varphi} \right) = \frac{1}{(1 - \varphi)^2},$$

окончательно получаем:

$$N_{сист} = \frac{p_0 \varphi}{(1 - \varphi)^2} = \frac{\varphi}{1 - \varphi}
 \tag{2.31}$$

Можно получить среднее количество заявок в очереди как часть $N_{сист}$.

$$N_{оч} = \frac{\lambda}{\mu} N_{сист} = \frac{\varphi^2}{(1 - \varphi)},
 \tag{2.32}$$

$$T_{оч} = \frac{N_{оч}}{\lambda}
 \tag{2.33}$$

СМО типа М/М/1

Как было описано при классификации систем, это система с пуассоновским входным потоком заявок, экспоненциальным законом распределения времени обслуживания и одним сервером.

На рис. 2.2 приведена простейшая схема такой системы. Она содержит буфер, который может хранить очередь бесконечной длины, состояние которой может быть отождествлено с числом заявок, содержащихся в очереди в каждый момент времени.

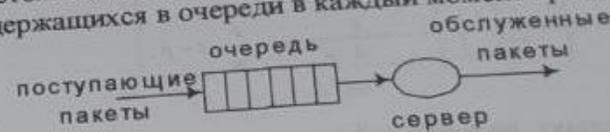


Рис. 2.2. СМО типа М/М/1.

Поскольку входной процесс ординарный, то в каждый момент времени к очереди может добавиться только одна заявка, поскольку сервер один, то в каждый момент времени может быть обслужена, то есть уйти из очереди только одна заявка. Таким образом, рассматриваемая СМО относится к процессу класса «гибели-размножения». Для анализа необходимо конкретизировать параметры системы. Распределение вероятностей входного потока и времени обслуживания позволяет полагать интенсивности вероятностей в модели постоянными:

$$\lambda_k = \lambda, k = 0, 1, 2, \dots$$

$$\mu_k = \mu = 1/\tau, k = 1, 2, 3, \dots$$

Здесь τ – среднее время обслуживания в сервере.

На рис. 2.3. приведена диаграмма интенсивностей переходов для рассматриваемой системы.

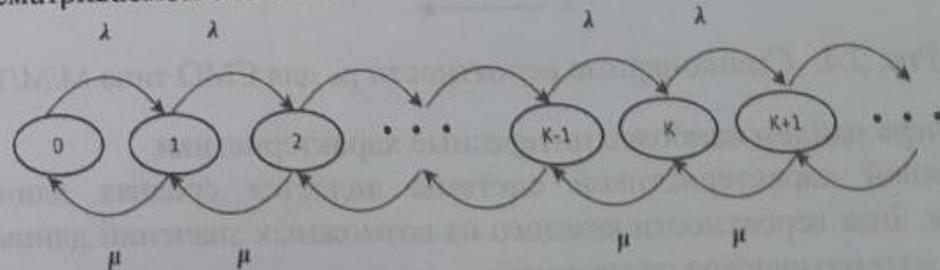


Рис. 2.3. Диаграмма интенсивности переходов для СМО типа М/М/1

Полученное ранее общее решение позволяет сразу записать вероятность того, что в стационарном состоянии в очереди будет находиться k заявок

$$p_k = p_0 \prod_{i=0}^{k-1} \frac{\lambda}{\mu} = p_0 \left(\frac{\lambda}{\mu}\right)^k, k \geq 1. \quad (2.34)$$

Найдем начальное значение вероятности, учитывая сходимость соответствующего ряда

$$p_0 = 1 \left[1 + \sum_{k=1}^{\infty} \left(\frac{\lambda}{\mu}\right)^k \right]^{-1} = \frac{1}{1 + \frac{\lambda/\mu}{1 - \lambda/\mu}} = 1 - \frac{\lambda}{\mu} = 1 - \rho. \quad (2.35)$$

Окончательно получаем формулу для вероятности длины очереди

$$p_k = (1 - \rho) \rho^k, k = 0, 1, 2, 3, \dots \quad (2.36)$$

На рис. 2.4 приведен график вероятностей того, что в очереди находится k заявок в установившемся режиме.

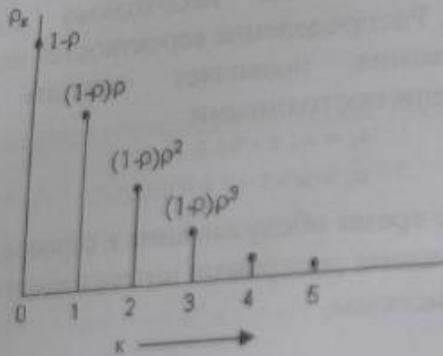


Рис. 2.4. Стационарные вероятности p_k для СМО типа М/М/1.

Теперь найдем наиболее интересные характеристики.

Важной характеристикой системы является средняя длина очереди. Зная вероятности каждого из возможных значений длины, найдем математическое ожидание:

$$\bar{N} = \sum_{k=0}^{\infty} k p_k = (1 - \rho) \sum_{k=1}^{\infty} k \rho^k = \frac{\rho}{1 - \rho}. \quad (2.37)$$

График средней длины очереди заявок в системе в зависимости от значения коэффициента использования или нагрузки показан на рис. 2.5.

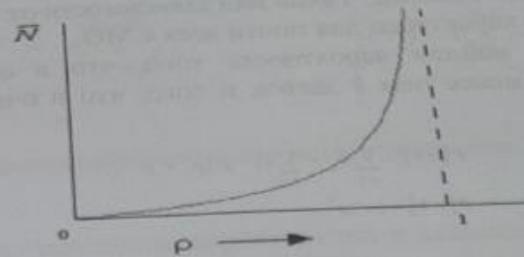


Рис. 2.5. Среднее число требований в системе типа М/М/1.

Найдем теперь дисперсию длины очереди:

$$\sigma_N^2 = \sum_{k=0}^{\infty} (k - \bar{N})^2 p_k = \frac{\rho}{(1 - \rho)^2}. \quad (2.38)$$

Для нахождения среднего значения времени пребывания в очереди воспользуемся формулой Литтла:

$$\tau = \frac{\bar{N}}{\lambda} = \left(\frac{\rho}{1 - \rho}\right) \left(\frac{1}{\lambda}\right) = \frac{1/\mu}{1 - \rho}. \quad (2.39)$$

На рис. 2.6 приведен график зависимости среднего времени пребывания в очереди в зависимости от коэффициента использования (нагрузки).

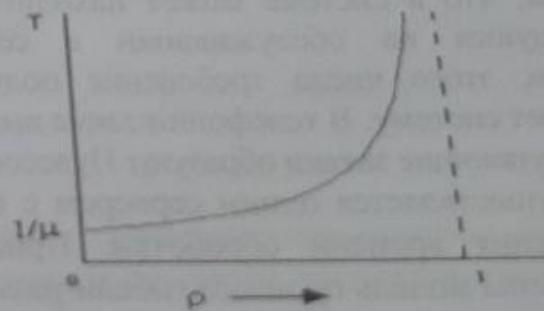


Рис. 2.6. Среднее время пребывания требования в системе типа М/М/1

Рассматривая полученные результаты, нетрудно видеть, что при увеличении коэффициента использования, как длина очереди, так и время пребывания в ней неограниченно возрастают при приближении ρ к единице. Такой вид зависимости от коэффициента использования характерен для почти всех СМО.

Наконец найдем вероятность того, что в очереди будет находиться не менее чем k заявок и того, что в очереди менее k заявок.

$$P[\geq k] = \sum_{i=k}^{\infty} p_i = \sum_{i=k}^{\infty} (1-\rho)\rho^i = \rho^k, \\ P[\leq k] = 1 - \rho^k. \quad (2.40)$$

Итак, в ходе анализа простейшей системы M/M/1 нам удалось в аналитическом виде найти все практически интересные характеристики QoS системы.

СМО с конечным накопителем типа M/M/1/N

Рассмотрим СМО, для которой фиксировано максимальное число ожидающих заявок (рис.2.7).

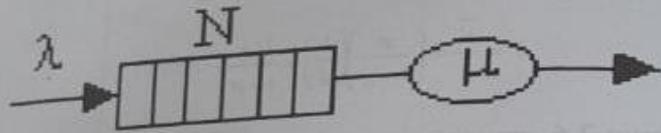


Рис.2.7. Структура СМО с конечным накопителем

Предположим, что в системе может находиться N заявок, включая находящуюся на обслуживании в сервере. Любое поступившее сверх этого числа требование получает отказ и немедленно покидает систему. В телефонии такие вызовы называют потерянными. Поступающие заявки образуют Пуассоновский поток, а обслуживание осуществляется одним сервером с показательным законом распределения времени обработки. Приспособим для описания такой системы модель процесса гибели-размножения [8]:

$$\lambda_k = \begin{cases} \lambda, & k < N \\ 0, & k \geq N \end{cases}, \mu_k = \mu, \quad k = 1, 2, \dots, N. \quad (2.41)$$

Эта система эргодична и диаграмма интенсивностей переходов может быть изображена так как на рис.2.8.

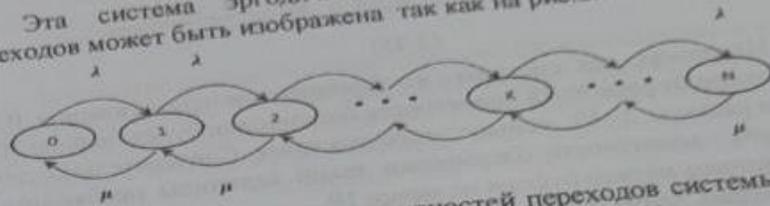


Рис.2.8. Диаграмма интенсивностей переходов системы типа M/M/1/N.

Найдем распределение вероятностей в стационарном режиме непосредственно из общей формулы [8,9]

$$p_k = p_0 \prod_{i=0}^{k-1} \frac{\lambda}{\mu}, \quad k \leq N, \\ p_k = p_0 \left(\frac{\lambda}{\mu}\right)^k, \quad k \leq N, \\ p_k = 0, \quad k \geq N. \quad (2.42)$$

Найдем теперь начальную вероятность, следуя общей формуле:

$$p_0 = \left[1 + \sum_{k=1}^N \left(\frac{\lambda}{\mu}\right)^k \right]^{-1} = \left[1 + \frac{(\lambda/\mu)(1 - (\lambda/\mu)^N)}{1 - (\lambda/\mu)} \right]^{-1} = \frac{1 - \lambda/\mu}{1 - (\lambda/\mu)^{N+1}}. \quad (2.43)$$

Таким образом, окончательная формула для стационарных вероятностей будет:

$$p_k = \begin{cases} \frac{1 - \rho}{1 - \rho^{N+1}} \rho^k, & 0 \leq k \leq N, \\ 0, & k < 0; k > N. \end{cases} \quad (2.44)$$

Проанализируем характеристики качества обслуживания (QoS) для такой системы. Важнейшей характеристикой будет являться вероятность блокировки – потери заявки. Очевидно, что это произойдет с вероятностью переполнения буфера, поэтому для расчета вероятности блокировки можно использовать формулу:

$$P_n = P_n(k=N) = \frac{(1-\rho)\rho^N}{1-\rho^{N+1}} \quad (2.45)$$

Например, для системы с коэффициентом использования 0.5 при размере буфера $N=18$ вероятность блокировки будет больше 10^{-6} , а при размере $N=19$, меньше этого значения. Следовательно, для получения вероятности блокировки такой величины необходимо предусмотреть размер буфера не менее 19.

Средняя длина очереди в буфере может быть найдена как:

$$\bar{L} = \sum_{k=0}^N k P_k = \frac{(1-\rho)}{1-\rho^{N+1}} \sum_{k=0}^N k \rho^k = \frac{\rho(1-\rho)(1+2\rho+3\rho^2+4\rho^3+\dots+N\rho^{N-1})}{1-\rho^{N+1}} \quad (2.46)$$

Соответственно задержка может быть найдена на основе формулы Литтла

$$\bar{T} = \frac{1}{\lambda} \bar{L} \quad (2.47)$$

Определим пропускную способность системы как число заявок, обслуживаемых системой в одну секунду. Очевидно, что при вероятности блокировки P_B пропускная способность может быть найдена как чистая интенсивность поступлений, то есть:

$$\gamma = \lambda(1 - P_B) \quad (2.48)$$

С точки зрения выхода системы пропускная способность может быть определена иначе. Если система всегда была бы непуста, то ее производительность равнялась бы величине обратной среднему времени обслуживания, то есть μ . Однако, поскольку часть времени система может простаивать, вероятность того, что в ней нет ни одной заявки отлична от нуля, реальная производительность может быть выражена как:

$$\gamma = \mu(1 - p_0) \quad (2.49)$$

Подставив выражения для вероятности простоя сервера для системы с бесконечным размером буфера, получим:

$$\gamma = \mu(1 - (1 - \rho)) = \lambda \Rightarrow P_B = 0 \quad (2.50)$$

Для системы с конечным буфером получаем:

$$\gamma = \mu \left(1 - \frac{1-\rho}{1-\rho^{N+1}}\right) \Rightarrow P_B = \frac{(1-\rho)\rho^N}{1-\rho^{N+1}} = (1-\rho)\rho^N, \rho^N \ll 1 \quad (2.51)$$

В качестве реального примера рассмотрим концентратор сети с коммутацией пакетов, который обрабатывает пакеты со средней длиной 1200 бит. При скорости передачи в канале 2400 бит/с средняя пропускная способность его составит $\mu=2$ пакета/с. Если полный входной поток имеет интенсивность $\lambda=1$ пакет/с, то $\rho=0.5$ и можно рассчитать, что при размере буфера $N=9$ пакетов в среднем по 1200 бит, вероятность блокировки составит 0.001. Для того, чтобы получить вероятность блокировки 0.000 001 нужно предусмотреть буфер длиной не менее 19 пакетов по 1200 бит, т.е. около 2850 байт.

2.3. Модели многоканальных систем массового обслуживания

Рассмотрим сначала простой случай системы, содержащей два сервера, любой из которых доступен для поступающих на вход заявок (рис.2.9). Системы с несколькими серверами такого типа называют полноступными. Очевидно, что по сравнению с односерверной системой производительность будет выше. Сразу отметим, что интерес будет представлять сравнение с односерверной системой интенсивность обслуживания в которой в среднем вдвое выше, то есть мы ответим на вопрос что эффективнее удвоение скорости обработки или распараллеливание обработки.

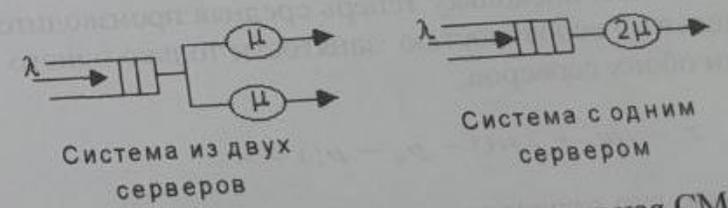


Рис.2.8. Многоканальная и одноканальная СМО.

Система M/M/2 может быть представлена как процесс размножения-гибели с параметрами:

$$\lambda_n = \lambda, \mu_1 = \mu, \mu_n = 2\mu, \forall n \geq 2.$$

Найдем сначала распределение вероятностей в стационарном режиме:

$$p_s = p_0 \prod_{i=0}^{s-1} \frac{\lambda_i}{\mu_{i+1}} = p_0 \left(\frac{\lambda}{\mu} \right) \left(\frac{\lambda}{2\mu} \right)^{s-1} = p_0 \frac{1}{2^{s-1}} \rho^s = p_0 2 \rho^s, \quad \rho_s = \frac{\lambda}{2\mu} \quad (2.52)$$

Находя из условий нормировки вероятность простоя (нулевого состояния), находим

$$p_0 = \frac{1 - \rho_2}{1 + \rho_2},$$

$$p_k = \frac{2(1 - \rho_2)}{(1 + \rho_2)} \rho_2^k \quad (2.53)$$

Найдем теперь основные характеристики качества обслуживания.

Средняя длина очереди составит:

$$\bar{L}_2 = \sum_{k=0}^{\infty} k p_k = \frac{2\rho_2}{(1 - \rho_2^2)} \leq \bar{L}_1 = \frac{\rho}{1 - \rho} \quad (2.54)$$

Теперь найдем среднее время задержки в системе по формуле Литтла:

$$\bar{T}_2 = \frac{\bar{L}_2}{\lambda} = \frac{1}{\mu(1 - \rho_2^2)} \quad (2.55)$$

Таким образом, в системе с двумя серверами время задержки сокращается. Нетрудно убедиться, что производительность системы М/М/2 также выше, поскольку теперь средняя производительность будет определяться вероятностью занятости только одного сервера и незанятости обоих серверов:

$$\gamma = \mu p_1 + 2\mu(1 - p_0 - p_1) = \lambda \quad (2.56)$$

Получилось, что производительность системы без блокировки также как и для системы с одним сервером совпадает с входной нагрузкой, тогда как максимальная производительность могла равняться 2μ .

Найдем теперь для сравнения характеристики качества обслуживания для односерверной системы с вдвое большей пропускной способностью сервера μ . Воспользуемся формулами для системы М/М/1.

На рис.2.9 представлены нормированные графики среднего времени задержки в системе с одним и с двумя серверами работающими с той же производительности и с одним сервером, работающим с вдвое большей скоростью. Как видно из сравнения, увеличение скорости работы сервера оказывается более эффективным, чем введение параллельного сервера той же производительности.

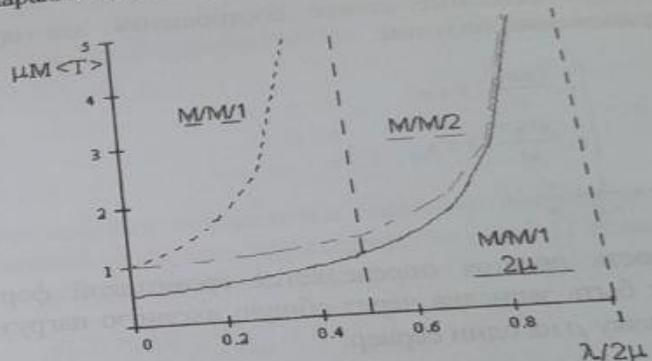


Рис.2.9. Нормированные графики среднего времени задержки в системе с одним и с двумя серверами

Рассмотрим теперь общий случай СМО с m серверами. Диаграмма интенсивностей переходов для такой системы представлена на рис.2.10.

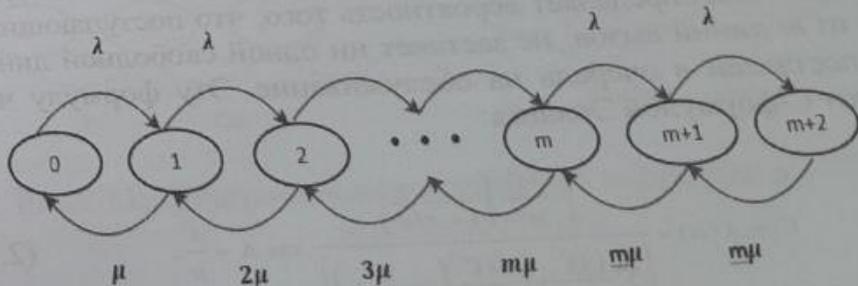


Рис.2.10. Диаграмма интенсивностей переходов для СМО М/М/м.

Интенсивности переходов могут быть определены следующим образом:

$$\lambda_n = \lambda, n = 0, 1, 2, 3, \dots$$

$$\mu_n = \min[n\mu, m\mu] = \begin{cases} n\mu, & 0 \leq n \leq m \\ m\mu, & m \leq n \end{cases}$$

Используя основные общие соотношения для процессов гибели-размножения, получим:

$$p_k = \begin{cases} p_0 \frac{(m\rho)^k}{k!}, & k \leq m \\ p_0 \frac{\rho^k m^n}{m!}, & k \geq m \end{cases}$$

$$\rho = \frac{\lambda}{m\mu} = \frac{A}{m} < 1. \quad (2.57)$$

Вероятность простоя определяется громоздкой формулой, которая может быть записана через общую входную нагрузку A и удельную нагрузку ρ на один сервер:

$$p_0 = \left[\sum_{k=0}^{m-1} \frac{(A)^k}{k!} + \frac{(m\rho)^m}{m!} \frac{1}{1-\rho} \right]^{-1}. \quad (2.58)$$

Полученные здесь соотношения позволяют рассчитать все характеристики QoS, мы приведем только одну формулу, позволяющую найти вероятность того, что поступающая в систему заявка окажется в очереди. Эту формулу широко используют в телефонии: она определяет вероятность того, что поступающий на пучок из m линий вызов, не застанет ни одной свободной линии и будет поставлен в очередь на обслуживание. Эту формулу часто называют С-формулой Эрланга:

$$C(m, \lambda/\mu) = \frac{\left(\frac{(A)^m}{m!} \right) \left(\frac{1}{1-A/m} \right)}{\left[\sum_{k=0}^{m-1} \frac{(A)^k}{k!} + \left(\frac{(A)^m}{m!} \right) \left(\frac{1}{1-A/m} \right) \right]}, \text{ где } A = \frac{\lambda}{\mu}. \quad (2.59)$$

Модель СМО, описываемая С - формулой Эрланга называется также Lost Calls Delayed (LCD).

Система обслуживания с m серверами и с иными потерями: M/M/m:Loss

Предметом рассмотрения теперь будет система без образования очереди для заявок, поступивших в моменты, когда все m серверы были заняты. Такие заявки будут просто теряться. В телефонии это типичный случай коммутирования на конечном коммутационном поле. Опишем такую систему подходящим процессом типа гибели-размножения. Его параметры могут быть определены так

$$\lambda_n = \begin{cases} \lambda, & n < m \\ 0, & n \geq m \end{cases}$$

$$\mu_n = n\mu, n = 1, 2, 3, \dots, m.$$

Такая система оказывается также эргодичной и диаграмма интенсивностей переходов, приведенная на рис.2.11 позволяет найти распределение вероятностей:

$$p_k = p_0 \prod_{i=0}^{k-1} \frac{\lambda}{(i+1)\mu} = p_0 \left(\frac{\lambda}{\mu} \right)^k \frac{1}{k!}, k \leq m, \quad (2.60)$$

$$p_0 = \left[\sum_{k=0}^m \left(\frac{\lambda}{\mu} \right)^k \frac{1}{k!} \right]^{-1}.$$

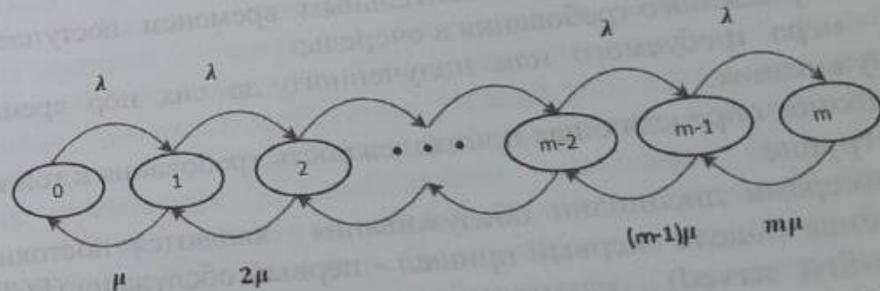


Рис.2.11. Диаграмма интенсивностей переходов для СМО типа M/M/m:Loss.

Основной характеристикой QoS для этой системы является средняя доля времени, когда все серверы оказываются занятым. В этом случае говорят о том, что в системе наступила блокировка. Вероятность такой блокировки определяется по формуле, носящей название формулы Эрланга.

в телефонии название B - формулы Эрланга или формулой потерь Эрланга:

$$P_n = E_n(m, A) = p_n = \frac{A^n}{\sum_{k=0}^n \frac{A^k}{k!}}, \quad A = \frac{\lambda}{\mu} \quad (2.61)$$

Эта формула играет столь большую роль в телефонии, что ее значения табулированы и существует масса таблиц, обратного расчета, то есть определения нагрузки, при которой обеспечивается заданная вероятность блокировки для заданного числа серверов. Такая таблица важна при расчетах многих сетей и систем массового обслуживания. Модель СМО, описываемая B - формулой Эрланга называется также Lost Calls Cleared (LCC).

2.4. Модели систем массового обслуживания с приоритетами

Дисциплина обслуживания – это способ определения того, какое требование в очереди должно обслуживаться следующим. Решение может основываться на одной из приведенных ниже характеристик или на их совокупности:

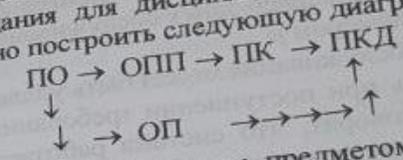
- мера, определяемая относительным временем поступления рассматриваемого требования в очередь;
- мера требуемого или полученного до сих пор времени обслуживания;
- функция, определяющая принадлежность требования к той или иной группе.

Примерами дисциплин обслуживания являются постоянно используемая модель «первый пришел - первый обслужен» (FCFS-first come-first served), называемая в русскоязычной литературе «дисциплина обслуживания в порядке поступления»-ОПП. Приведем здесь список некоторых типичных дисциплин обслуживания.

- ОПП- обслуживание в порядке поступления (FCFS);
- ООП – обслуживание в обратном порядке, т.е. последнее поступившее требование обслуживается первым (LCFS);
- ПК – первоочередное обслуживание требований с кратчайшей длительностью обслуживания (SPT/SJE);

- ПКД – первоочередное обслуживание требований с кратчайшей длительностью дообслуживания (SRPT);
- ПКС – первоочередное обслуживание требований с кратчайшей средней длительностью обслуживания (SEPT);
- ПКСД – первоочередное обслуживание требований с кратчайшей средней длительностью дообслуживания (SERPT);
- ПКОВ – первоочередное обслуживание требований с кратчайшим обязательным временем (SIPT).

Если сравнивать эти дисциплины по среднему времени ожидания попарно, и обозначать тот факт, что среднее время ожидания для дисциплины D_1 больше или равно среднему времени ожидания для дисциплины D_2 следующим образом: $D_1 \rightarrow D_2$, то можно построить следующую диаграмму



Итак, основным предметом анализа различных дисциплин обслуживания будем считать расчет среднего времени ожидания требования в очереди или среднего времени пребывания в системе.

Предположим, что требования принадлежат одному из P различных приоритетных классов, обозначаемых индексом $p=1,2,3,\dots,P$. Каждому требованию, находящемуся в системе в момент времени t ставится в соответствие значение некоторой приоритетной функции $q_p(t)$. Чем больше значение этой функции, тем выше приоритет требования. Всякий раз, когда принимается решение для выбора требования на обслуживание, выбор делается в пользу требования с наибольшим значением приоритетной функции. В простейшем случае в качестве приоритетной функции выбирается просто значение p . В этом случае приоритет требования тем больше, чем больший номер класса принадлежности оно имеет. Рассмотрим достаточно общую модель, основанную на системе M/G/1. Предположим, что требования из приоритетного класса p образуют пуассоновский поток с интенсивностью λ_p требований в секунду. Время обслуживания каждого требования из этого класса выбирается независимо в соответствии с распределением плотности с вероятностью $b_p(x)$ со средним значением

$$\bar{x}_p = \int_0^{\infty} x b_p(x) dx \quad (2.62)$$

Введем следующие определения

$$\begin{aligned} \lambda &= \sum_{p=1}^P \lambda_p, \\ \bar{x} &= \sum_{p=1}^P \frac{\lambda_p}{\lambda} \bar{x}_p, \\ \rho_p &= \lambda_p \bar{x}_p, \\ \rho &= \lambda \bar{x} = \sum_{p=1}^P \rho_p. \end{aligned} \quad (2.63)$$

Здесь ρ интерпретируется как доля времени, в течение которого сервер занят ($\rho < 1$), а каждый из парциальных коэффициентов ρ_p — как доля времени, в течение которого сервер занят обслуживанием заявок из приоритетного класса с номером p .

Если требование в процессе обслуживания может быть удалено из сервера и возвращено в очередь при поступлении требования с более высоким приоритетом, то говорят, что система работает с абсолютным приоритетом, если обслуживание любого требования, находящегося в сервере не может быть прервано, то говорят что СМО работает с относительным приоритетом.

Основная модель расчета среднего времени ожидания

Будем использовать далее следующие обозначения для среднего значения времени ожидания в очереди требований из приоритетного класса p — W_p , и среднего времени пребывания в системе для требований этого класса — T_p :

$$T_p = W_p + \bar{x}_p. \quad (2.64)$$

Основное внимание будем уделять системам с относительным приоритетом. Рассмотрим процесс с момента поступления некоторого требования из приоритетного класса p . Будем далее называть это требование меченым. Первая составляющая времени ожидания для меченого требования связана с требованием, которое оно застает в сервере. Эта составляющая равна остаточному времени обслуживания другого требования. Обозначим теперь и будем использовать это обозначение и далее, среднюю задержку меченого требования, связанную с наличием другого требования на обслуживании W_0 . Зная распределение времени между соседними поступлениями входных требований для

каждого приоритетного класса, можно всегда вычислить эту величину. В нашем предположении пуассоновского закона для потока заявок каждого класса можно записать

$$W_0 = \sum_{i=1}^P \rho_i \frac{\bar{x}_i^2}{2x_i} = \sum_{i=1}^P \frac{\lambda_i \bar{x}_i^3}{2}. \quad (2.65)$$

Вторая составляющая времени ожидания для меченого требования определяется тем, что перед меченым требованием обслуживаются другие требования, которые меченое требование застало в очереди. Обозначим далее число требований из класса i , которое застало в очереди меченое требование (из класса p) и которые обслуживаются перед ним N_{ip} . Среднее значение этого числа будет определять величину среднего значения этой составляющей задержки

$$\sum_{i=1}^P \bar{x}_i N_{ip}. \quad (2.66)$$

Третья составляющая задержки связана с требованиями, поступившими после того как пришло меченое требование, однако получившими обслуживание раньше его. Число таких требований обозначим M_{ip} . Среднее значение этой составляющей задержки находится аналогично и составляет

$$\sum_{i=1}^P \bar{x}_i M_{ip}. \quad (2.67)$$

Складывая все три составляющие, получаем, что среднее время ожидания в очереди для меченого требования определяется формулой

$$(*) W_p = W_0 + \sum_{i=1}^P \bar{x}_i (\bar{N}_{ip} + \bar{M}_{ip}), \quad p = 1, 2, \dots, P. \quad (2.68)$$

Очевидно, что независимо от дисциплины обслуживания число требований, N_{ip} и M_{ip} в системе не может быть произвольным, поэтому существует некоторый набор соотношений, связывающий между собой задержки для каждого из приоритетного класса. Важность этих соотношений для СМО позволяет называть их законами сохранения. Основой законов сохранения для задержек является тот факт, что незаконченная работа в любой СМО в течение любого интервала времени занятости не зависит от порядка обслуживания, если система является консервативной (требования

не исчезают внутри системы и сервер не простаивает при пустой очереди).

Распределение времени ожидания существенно зависит от порядка обслуживания, но если дисциплина обслуживания выбирается от требований независимо от времени их обслуживания (или любой меры, зависящей от времени обслуживания), то распределение числа относительно порядка обслуживания в системе инвариантно.

Для СМО типа M/G/1 можно показать, что для любой дисциплины обслуживания должно выполняться следующее важное равенство:

$$\sum_{p=1}^r \rho_p W_p = \begin{cases} \frac{\rho W_0}{1-\rho}, & \rho < 1, \\ \infty, & \rho \geq 1. \end{cases} \quad (2.69)$$

Это равенство означает, что взвешенная сумма времен ожидания никогда не изменяется, независимо от того, насколько сложна или искусно подобрана дисциплина обслуживания. Если удастся сократить задержку для одних требований, то она немедленно возрастет для других.

Для более общей системы с произвольным распределением времени поступления требований G/G/1 закон сохранения может быть записан в виде:

$$\sum_{p=1}^r \rho_p W_p = \bar{U} - W_0. \quad (2.70)$$

Общий смысл этого соотношения таков: взвешенная сумма времен задержки остается постоянной. Просто в правой части стоит разность средней незавершенной работы и остаточного времени обслуживания. Если предположить пуассоновский характер входного потока, то выражение для незавершенной работы можно записать в виде:

$$\bar{U} = \frac{W_0}{1-\rho}. \quad (2.71)$$

Подставляя его в предыдущее выражение, сразу получается приведенный ранее закон сохранения для СМО типа M/G/1.

Рассмотрим теперь расчет среднего времени ожидания для СМО с обслуживанием в порядке приоритета, задаваемого приоритетной функцией $q_p(i) = p$.

На рис. 2.12 приведена схема функционирования СМО с такой дисциплиной обслуживания: поступающее требование ставится в очередь слева от требования с равным или большим приоритетом.

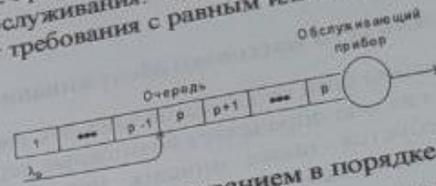


Рис. 2.12. СМО с обслуживанием в порядке приоритета.

Все требования более высокого, чем у меченого приоритета будут обслужены раньше. Из формулы Литтла число требований класса i находящихся в очереди, будет равно:

$$\bar{N}_p = \lambda_i W_i, \quad i = p, p+1, p+2, \dots, r. \quad (2.72)$$

Требования более высокоприоритетных классов, поступившие в систему после меченого требования, пока оно находится в очереди, также будут обслужены перед ним. Так как меченое требование будет находиться в очереди в среднем W_p секунд, то число таких требований будет равно:

$$\bar{M}_p = \lambda_i W_p. \quad (2.73)$$

Непосредственно из формулы (2.72) получаем:

$$W_p = W_0 + \sum_{i=p}^r \bar{x}_i \lambda_i W_i + \sum_{i=p+1}^r \bar{x}_i \lambda_i W_p, \quad (2.74)$$

$$W_p = \frac{W_0 + \sum_{i=p}^r \rho_i W_i}{1 - \sum_{i=p+1}^r \rho_i}$$

Эта система уравнений может быть решена рекуррентно, начиная с W_1, W_2 и т.д.

$$(**) W_p = \frac{W_0}{(1-\sigma_p)(1-\sigma_p)} \quad (2.75)$$

$$\sigma_p = \sum_{i=p}^r \rho_i$$

Полученная формула позволяет рассчитывать характеристики качества обслуживания для всех классов приоритетных классов.

2.5. Модель сети массового обслуживания

При исследовании сети необходимо задавать топологическую структуру сети, так как она определяет возможные переходы заявок между узлами. Требуется также описать маршруты отдельных заявок и вероятностные модели потоков заявок между узлами сети.

Пусть сеть содержит n узлов. В каждый узел сети поступает пуассоновский поток заявок с интенсивностью λ_i . Из i -узла, заявка с вероятностью p_{ij} поступает в j узел.

Характеристики сети массового обслуживания рассчитываются по формулам 2.15-2.22.

Пусть заданы следующие исходные данные:

Таблица 2.1
Топология сети и матрица вероятностей передачи пакетов

Узлы $i \setminus j$	1	2	3	4	5	6	7
1	0	1/4	1/4	0	0	0	0
2	0	0	1/4	1/2	0	0	0
3	1/3	0	0	1/3	0	0	0
4	0	1/4	1/3	0	0	0	0
5	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0
7	0	0	0	0	0	0	$p_{7,7} = 0$

p_{ij} - вероятность передачи пакета из узла i в узел j .

Таблица 2.2
Вектор интенсивностей внешних входящих потоков в узлах

Интенсивность внешнего входящего потока пакетов (пакет/сек.)	Узлы						
	1	2	3	4	5	6	7
λ	0,12	0,22	0,14	0,12	0	0	0

Таблица 2.3
Типы математических моделей узлов сети

Математические модели	Узлы						
	1	2	3	4	5	6	7
	M/U/1	M/M/1	M/E ₂ /1	M/D/1	-	-	-

Распределение времени обслуживания пакетов: M - экспоненциальное; D - детерминированное; E₂ - Эрланга 2-го порядка; U - равномерное.

Исходя из исходных данных видно (табл.2.1), что в сети имеются 4 активных узла.

На основании данных матрицы вероятностей (табл.2.1) построим граф топологии сети (рис.2.13). На рис.1 внесем величины интенсивностей внешних входящих потоков в узлах (табл.2.2).

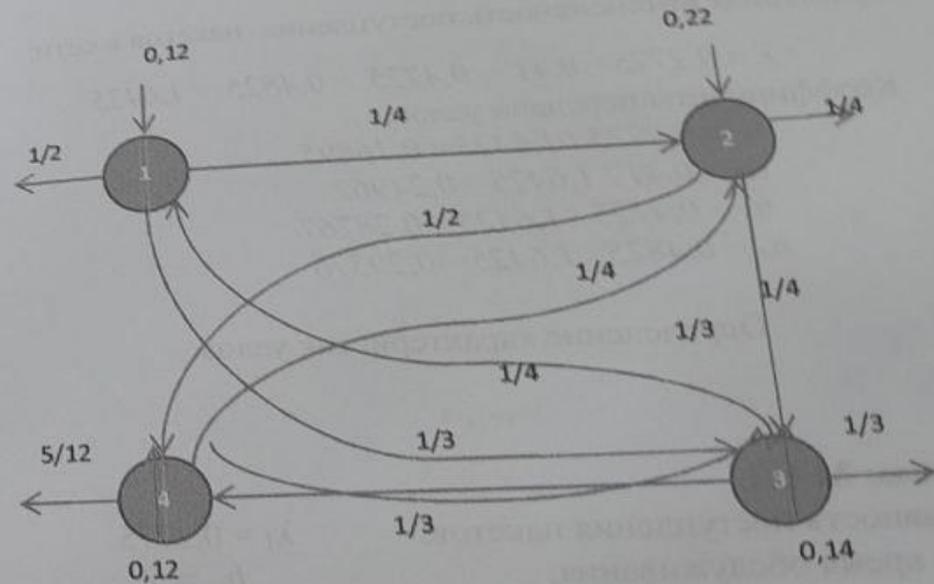


Рис.2.13. Топология сети.

На основании графа топологии сети составляем уравнения баланса интенсивностей:

$$\begin{aligned} \lambda_1 &= 0.12 + \frac{1}{3}\lambda_3 \\ \lambda_2 &= 0.22 + \frac{1}{4}\lambda_1 + \frac{1}{4}\lambda_4 \\ \lambda_3 &= 0.14 + \frac{1}{4}\lambda_1 + \frac{1}{4}\lambda_2 + \frac{1}{3}\lambda_4 \\ \lambda_4 &= 0.12 + \frac{1}{2}\lambda_2 + \frac{1}{3}\lambda_3 \end{aligned} \quad (2.84)$$

Решая уравнения баланса интенсивностей методом Гаусса находим интенсивности поступления пакетов в узлах сети:

$\lambda_1 = 0,2775$	$\lambda_2 = 0,41$	$\lambda_3 = 0,4725$	$\lambda_4 = 0,4825$
----------------------	--------------------	----------------------	----------------------

Формула для определения коэффициента передачи узлов:

$$\alpha_i = \frac{\lambda_i}{\sum_{i=0}^n \lambda_i} \quad (2.85)$$

Суммарная интенсивность поступления пакетов в сети:

$$\lambda = 0,2775 + 0,41 + 0,4725 + 0,4825 = 1,6425$$

Коэффициенты передачи узлов:

$$\begin{aligned} \alpha_1 &= 0,2775 / 1,6425 = 0,16895 \\ \alpha_2 &= 0,41 / 1,6425 = 0,24962 \\ \alpha_3 &= 0,4725 / 1,6425 = 0,28767 \\ \alpha_4 &= 0,4825 / 1,6425 = 0,29376 \end{aligned}$$

Определение характеристик узлов

Узел 1

Модель узла: M/U/1

Интенсивность поступления пакетов: $\lambda_1 = 0,2775$

Среднее время обслуживания: $b_1 = 2$

Коэффициент вариации узла: $v = (b-a) / (\sqrt{3}(a+b))$

Возьмем значения $b=3$ и $a=1$: $v = (3-1) / (\sqrt{3}(3+1)) = 0,29$

1. Загрузка узла: $\rho_1 = \lambda_1 b_1 = 0,2775 * 2 = 0,56$
2. Коэффициент простоя узла: $\eta = 1 - \rho = 1 - 0,56 = 0,44$
3. Время ожидания: $w = \lambda b^2 (1 + v^2) / 2(1 - \rho) = 1,35$
4. Среднее время пребывания

- в системе:
5. Средняя длина очереди заявок: $l = \lambda w = 0,277 * 1,35 = 0,37$
 6. Среднее число заявок в системе: $m = \lambda u = 0,277 * 3,35 = 0,93$

Узел 2

- Модель узла: M/M/1
- $\lambda_2 = 0,41$ $b_2 = 1,5$
1. Определение загрузки узла: $\rho_2 = \lambda_2 b_2 = 0,41 * 1,5 = 0,615$
 2. Коэффициент простоя узла: $\eta = 1 - \rho = 1 - 0,615 = 0,385$
 3. Время ожидания: $w = \rho b / (1 - \rho) = 0,615 * 1,5 / (1 - 0,615) = 2,4$
 4. Среднее время пребывания в системе: $u = w + b = 2,4 + 1,5 = 3,9$
 5. Средняя длина очереди заявок: $l = \lambda w = 0,41 * 2,45 = 0,98$
 6. Среднее число заявок в системе: $m = \lambda u = 0,41 * 3,9 = 1,6$

Узел 3

- Модель узла: M/E₂/1
- $\lambda_3 = 0,4725$ $b_3 = 1,4$ $k = 2$
- Коэффициент вариации узла: $v = 1/\sqrt{k} = 1/\sqrt{2} = 0,707$
1. Определение загрузки узла: $\rho_3 = \lambda_3 b_3 = 0,4725 * 1,4 = 0,662$
 2. Коэффициент простоя узла: $\eta = 1 - \rho = 1 - 0,662 = 0,34$
 3. Время ожидания: $w = \lambda b^2 (1 + v^2) / 2(1 - \rho) = 2,05$
 4. Среднее время пребывания в системе: $u = w + b = 2,05 + 1,4 = 3,45$
 5. Средняя длина очереди заявок: $l = \lambda w = 0,4725 * 2,05 = 0,97$
 6. Среднее число заявок в системе: $m = \lambda u = 0,4725 * 3,45 = 1,63$

Узел 4

- Модель узла: M/D/1
- $\lambda_4 = 0,4825$ $b_4 = 1$
1. Определение загрузки узла: $\rho_4 = \lambda_4 b_4 = 0,4825 * 1 = 0,4825$
 2. Коэффициент простоя узла: $\eta = 1 - \rho = 1 - 0,4825 = 0,5175$
 3. Время ожидания: $w = \lambda b^2 / 2(1 - \rho) = 0,4825 * 1^2 / 2(1 - 0,4825) = 0,47$
 4. Среднее время пребывания в системе: $u = w + b = 0,47 + 1 = 1,47$

5. Средняя длина очереди заявок: $l = \lambda w = 0,4825 * 0,47 = 0,225$
 6. Среднее число заявок в системе: $m = \lambda u = 0,4825 * 1,47 = 0,707$
 Характеристики узлов сведены в таблицу 2.4.
 Определение характеристик сети.

1. Суммарная нагрузка сети: $R = 0,56 + 0,615 + 0,66 + 0,4825 = 2,314$
 2. Среднее число заявок, находящихся в очереди: $L = 0,37 + 0,98 + 0,97 + 0,22 = 2,55$
 3. Среднее число заявок, находящихся в сети: $M = 0,93 + 1,6 + 1,63 + 0,707 = 4,87$
 4. Среднее время ожидания: $W = 0,23 + 0,6 + 0,57 + 0,14 = 1,536$
 5. Среднее время пребывания заявок в сети: $U = 0,566 + 0,97 + 0,99 + 0,43 = 2,97$

Сводная таблица характеристик узлов

Узлы		1	2	3	4
Модель узла		M/U/1	M/M/1	M/E ₂ /1	M/D/1
Рассчитываемая величина					
		3	4	5	6
Интенсивность поступления заявок	λ	0,2775	0,41	0,4725	0,4825
Время между соседними заявками	$a = 1/\lambda$	3,60360	2,4390 2	2,11640	2,0725 4
Средняя длительность обслуживания заявок в узлах	b	2,0	1,5	1,4	1
Коэффициент передачи узлов	$\alpha = \lambda / \sum \lambda$	0,16895	0,2496 2	0,28767	0,2937 6
Загрузка узла	$\rho = \lambda b$	0,555	0,615	0,6615	0,4825
Интенсивность обслуживания	$\mu = 1/b$	1,8018	1,6260 2	1,51172	2,0725 4
Коэффициент простоя	$\eta = 1 - \rho$	0,445	0,385	0,3385	0,5175
1	2	3	4	5	6

Время ожидания	w	$w = \lambda b^2(1 + v^2)/2(1 - \rho)$	$w = \rho b / (1 - \rho)$	$w = \lambda b^2(1 + v^2)/2(1 - \rho)$	$w = \lambda b^2 / 2(1 - \rho)$
				1,35112	2,396
Среднее время пребывания заявок в системе	$u = w + b$	3,351	3,896	3,452	1,466
Средняя длина очереди заявок	$l = \lambda w$	0,37494	0,9824 0	0,96953	0,2249 3
Среднее число заявок в системе	$m = \lambda u$	0,9299	1,5974	1,6310	0,7074

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ГЛАВЕ 2

1. Какими свойствами должен обладать нормальный поток?
2. Виды систем массового обслуживания (СМО).
3. Объясните значение параметров входного потока.
4. Как определяется скорость обслуживания?
5. Как образом отображается топология сети?
6. Как составляется уравнение балансировки нагрузки?
7. Какие процедуры обслуживания вы знаете?
8. Объясните одноканальную модель СМО с неограниченной памятью.
9. Объясните одноканальную модель СМО с ограниченной памятью.
10. Как определяется нагрузка системы?
11. Объясните многоканальные модели СМО.
12. Характеристики многоканальной СМО.
13. Алгоритм обслуживания потока с относительным приоритетом.
14. Алгоритм обслуживания потока с абсолютным приоритетом.
15. Характеристики СМО с относительным и абсолютным приоритетами.

3. МАТЕМАТИЧЕСКИЕ ОСНОВЫ ОПТИМИЗАЦИИ СЕТЕЙ ТЕЛЕКОММУНИКАЦИИ

3.1. Методы оптимизации

Выбор метода решения задачи оптимизации - один из важнейших этапов оптимизации. Можно выделить группы методов [10-12]:

- аналитические методы;
- методы математического программирования.

Группа аналитических методов оптимизации объединяет аналитический поиск экстремума функции, метод множителей Лагранжа, вариационные методы и принцип максимума. Аналитический поиск экстремума функций, заданных без ограничений на независимые переменные, применяется к задачам, у которых оптимизируемая функция имеет аналитическое выражение, дифференцируемое во всем диапазоне исследования, а число переменных невелико. Это один из наиболее простых методов.

Группа методов математического программирования включает линейное программирование, нелинейное и динамическое программирование.

Линейное программирование - метод для решения задач оптимизации с линейными выражениями для критерия оптимальности и линейными ограничениями на область изменения переменных. Подобные задачи решаются итерационными способами. Одним из способов реализации линейного программирования является симплекс-алгоритм, который почти всегда, за исключением некоторых случаев, может найти оптимальное решение. В основе этого алгоритма лежит полный перебор возможных вариантов решения задачи.

Задачей линейного программирования называется задача, в которой минимизируемым или максимизируемым критерием является линейная функция, причем на переменные налагаются ограничения, которые также представляются линейными функциями. Комбинация скаляров или векторов, обычно обозначаемых X_i , называется линейно, если она может быть записана в виде:

$$c_1X_1 + c_2X_2 + \dots + c_nX_n \quad (3.1)$$

где все коэффициенты c - константы. Например, функция $4x_1 + 3x_2 + 5x_3 + 2$ линейно относительно переменных x_1, x_2, x_3 , тогда как функция $2x_1^2 + x_1x_2 + 3e^{x_3}$ не линейна относительно тех же переменных. Величины X_1, \dots, X_n называются линейно зависимыми, если для некоторого набора c_i в предположении, что не все c_i равны нулю, имеет следующее равенство:

$$C_1x_1 + C_2x_2 + \dots + C_nx_n = \sum_{i=1}^n C_i x_i = 0 \quad (3.2)$$

С другой стороны, если $\sum_{i=1}^n C_i X_i = 0$ только тогда, когда все коэффициенты c_i равны нулю, то величины X_1, \dots, X_n называются линейно независимыми.

Хотя задача линейного программирования может быть сформулирована по-разному, запишем её в следующей форме:

Минимизировать: $y = \sum_{i=1}^n C_i x_i \Rightarrow \min \quad (3.3)$

при ограничениях $\sum_{i=1}^n a_{ij} x_i - b_j \geq 0, \quad j = 1, 2, \dots, m: \quad (3.4)$

$$x_i \geq 0 \quad (3.5)$$

где a, b и c - константы, а x_i - искомые переменные. Уравнения (3.3)-(3.5) можно компактно записать в матричной форме. Пусть X и c - вектор-столбцы размерности $(n \times 1)$ и b - вектор-столбец размерности $(m \times 1)$:

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix}, a = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1m} \\ a_{21} & a_{22} & \dots & a_{2m} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nm} \end{pmatrix}, b = \begin{pmatrix} b_1 \\ b_2 \\ \dots \\ b_m \end{pmatrix}, c = \begin{pmatrix} C_1 \\ C_2 \\ \dots \\ C_n \end{pmatrix}$$

Тогда в матричных обозначениях уравнения (3.3)-(3.5) запишутся следующим образом: (3.6)

$$\text{минимизировать } y = f(X) = C^T X = \min$$

при ограничениях $a^T X \geq b \quad (3.7)$

$$X \geq 0 \quad (3.8)$$

где верхний индекс T означает транспонирование.

Общим для методов нелинейного программирования является то, что их используют при решении задач с нелинейными критериями оптимальности. Все методы нелинейного программирования – это численные методы поискового типа. Суть их – в определении набора независимых переменных, дающих наибольшее приращение оптимизируемой функции.

Пусть непрерывная функция $f(X)$ представляет собой целевую функцию; $h_1(X), \dots, h_m(X)$ задают ограничения в виде равенств, а $g_{m+1}(X) \dots g_p(X)$ – ограничения в виде неравенств, где $X = \{x_1, \dots, x_n\}$ – вектор-столбец компонентов x_1, \dots, x_n в n -мерном евклидовом пространстве E^n .

Формально задача нелинейного программирования может быть сформулирована следующим образом:

$$\text{минимизировать } f(x) \Rightarrow \min, \quad X \in E^n, \quad (3.9)$$

при m линейных и/или нелинейных ограничениях в виде равенств

$$h_j(x) = 0, \quad j = 1, 2, \dots, m \quad (3.10)$$

и $(p-m)$ линейных и(или) нелинейных ограничениях в виде неравенств

$$g_j(x) \geq 0, \quad j = m + 1, m + 2, \dots, p; \quad (3.11)$$

Динамическое программирование – эффективный метод решения задач оптимизации многостадийных процессов. Метод предполагает разбивку анализируемого процесса на стадии (во времени и пространстве). Рассмотрение задачи начинается с последней стадии процесса, и оптимальный режим определяется поэтапно.

Эвристический метод направлен на сокращение перебора. Решения, получаемые данным методом, не являются наилучшими, а относятся лишь к множеству допустимых.

Каждый метод имеет свои особенности, которые определяются их принципом работы и реализацией, и отличаются друг от друга как сложностью и граничными условиями, так и отклонением от оптимального значения. Точное решение проблемы оптимизации можно получить с помощью линейного программирования, однако сложность вычислений при линейном программировании быстро

возрастает с увеличением числа узлов в сети и для больших сетей является критической, что приводит к необходимости использования эвристических методов.

3.2. Функции MATLAB для решения задач оптимизации

Решение этих задач в MATLAB достигается с помощью пакета оптимизации (Optimization Toolbox). Этот пакет представляет собой библиотеку функций, расширяющих возможности системы MATLAB по численным вычислениям и предназначенную для решения задач оптимизации и систем нелинейных уравнений; поддерживает основные методы оптимизации функций [10]:

- Безусловная оптимизация нелинейных функций.
- Метод наименьших квадратов.
- Решение нелинейных уравнений.
- Линейное программирование.
- Квадратичное программирование.
- Условная минимизация нелинейных функций.
- Методы минимакса.
- Многокритериальная оптимизация.

Различные типы таких задач вместе с применяемыми для их решения функциями пакета Optimization Toolbox даны в табл. 3.1.

В таблице 3.1 использованы такие обозначения:

- a – скалярный аргумент; x, y – в общем случае векторные аргументы;
- $f(a), f(x)$ – скалярные функции; $F(x), c(x), seq(x), K(x,w)$ – векторные функции;
- A, Aeq, C, H – матрицы;
- b, beq, d, f, w – векторы;
- xL, xU – соответственно, нижняя и верхняя границы области изменения аргумента.

Таблица 3.1

Типы задач, решаемых средствами Optimization Toolbox

Тип задачи	Математическая запись	Функция MATLAB
Скалярная(одномерная) минимизация	$\min f(a) \quad a_1 < a < a_2$ x	fminunc,
Безусловная минимизация (без ограничений)	$\min f(x)$ x	fminunc, fminsearch

Линейное программирование	$\min f^T x$ при условиях $A^*x < b, Aeq^*x = beq,$ $x_L < x < x_U$	linprog
Квадратичное программирование	$\min \frac{1}{2} x^T H x + f^T x$ x при $A^*x < b, Aeq^*x = beq,$ $x_L < x < x_U$	quadprog
Минимизация при наличии ограничений	$\min f(x)$ при условиях $A^*x < b, Aeq^*x = beq,$ $x_L < x < x_U$	fmincon

Формулировка проблемы в задачах минимизации чаще всего выглядит следующим образом:

$$\min f(x), \quad (3.12)$$

где x – множество возможных значений целевая функция $f(x)$, на которые могут быть наложены ограничения. В большинстве случаев целевая функция является скалярной, а её аргумент – скаляр или вектор. Однако для некоторых задач мультиобъектной минимизации, поиска решений уравнений или минимизации сумм квадратов применяется векторная или матричная целевая функция $F(x)$

Большинство инструментов пакета Optimization Toolbox рассчитаны на решение задачи минимизации целевой функции. Если же стоит задача максимизации некоторого критерия, то разумным является использование дополнительной целевой функции

$$g(x) = -f(x), \quad (3.13)$$

которую требуется минимизировать.

Для решения задач линейного программирования в канонической форме в пакете MATLAB имеется функция Linprog. Эту функцию можно использовать с различным набором аргументов для решения задач с различными видами ограничений.

Канонической формой считается нахождение минимума целевой функции при ограничениях типа «меньше или равно».

Поэтому:

– при решении задачи нахождения максимума целевой функции необходимо задавать ее коэффициенты с обратными знаками;

– при наличии ограничений типа «больше или равно» неравенство необходимо умножить на -1 .
Рассмотрим разновидности задач, для которых используется функция linprog с различным набором аргументов.

1 Найти минимум функции $f(x)$ при ограничениях $a^*x \leq b$:
 $x = \text{linprog}(f, a, b)$,
где a – матрица, соответствующая коэффициентам левых частей ограничений типа «меньше или равно»;
 b – вектор-столбец правых частей ограничений типа «меньше или равно».

2 Найти минимум функции $f(x)$ при ограничениях $a^*x \leq b$ при дополнительных ограничениях типа равенства $aeq^*x = beq$:
 $x = \text{linprog}(f, a, b, aeq, beq)$,
где a – матрица, соответствующая коэффициентам левых частей ограничений типа «меньше или равно»;
 b – вектор-столбец правых частей ограничений типа «меньше или равно»;
 aeq – матрица, соответствующая коэффициентам левых частей ограничений типа «равно»;

beq – вектор-столбец правых частей ограничений типа «равно».

3 Найти минимум функции $f(x)$ при ограничениях $a^*x \leq b$, изменения переменных x $lb \leq x \leq ub$:
 $x = \text{linprog}(f, a, b, aeq, beq, lb, ub)$,
где a – матрица, соответствующая коэффициентам левых частей ограничений типа «меньше или равно»;
 b – вектор-столбец правых частей ограничений типа «меньше или равно»;

aeq – матрица, соответствующая коэффициентам левых частей ограничений типа «равно»;
 beq – вектор-столбец правых частей ограничений типа «равно»;
 lb – ограничение «снизу» на x ;
 ub – ограничение «сверху» на x .

Примечание: Если x_i не ограничен снизу, $lb(i) = -Inf$; если не ограничен сверху, $ub(i) = Inf$.

Если необходимо получить значение целевой функции в точке оптимума, используйте следующий формат вызова:

$$[x, fval] = \text{linprog}(f, a, b)$$

Пример:

$$F(x) = x_1 + 5x_2 - 2x_3 \rightarrow \max$$
$$x_1 - 2x_2 + x_3 \geq 20$$
$$x_1 + 5x_2 + 4x_3 \leq 42$$
$$2x_1 + x_2 \geq 30$$
$$x_1 = 0, x_2 \geq 0, x_3 \geq 0$$

Решение

- 1 Зададим коэффициенты целевой функции в виде вектор-столбца:
 $f = [-1; -5; 2]$
- 2 Зададим матрицу коэффициентов ограничений:
 $a = [-1 \ 2 \ -1; 1 \ 5 \ 4; -2 \ -1 \ 0]$
- 3 Зададим правые части ограничений (вектор-столбец):
 $b = [-20; 42; -30]$
- 4 Зададим ограничения снизу на переменные x :
 $lb = [0; 0; 0]$
- 5 Вызовем функцию:
 $[x, fval] = \text{linprog}(f, a, b, [], [], lb)$

Функция `fmincon` (нелинейное программирование)

`fmincon` – функция поиска минимума скалярной функции многих переменных при наличии ограничений вида:

$$\begin{aligned} c(x) < 0, \text{seq}(x) = 0, \\ A x < b, \text{Aeq} x = \text{beq}, \\ lb < x < ub \end{aligned} \quad (3.14)$$

`fmincon` находит минимум для скалярной функции нескольких переменных с ограничениями начиная с начального приближения. В общем случае, эта задача относится к нелинейной оптимизации с ограничениями или к нелинейному программированию.

- $x = \text{fmincon}(\text{fun}, x_0, A, b)$ начинает с точки x_0 и находит минимум от x для функции представленной как `fun` при условии выполнения линейных неравенств $A*x \leq b$. x_0 может быть скаляром, вектором или матрицей.
- $x = \text{fmincon}(\text{fun}, x_0, A, b, \text{Aeq}, \text{beq})$ минимизирует `fun` при условии выполнения линейных равенств $\text{Aeq}*x = \text{beq}$, а так же $A*x \leq b$. Устанавливается $A=[]$ и $b=[]$ в случае отсутствия неравенств.

$x = \text{fmincon}(\text{fun}, x_0, A, b, \text{Aeq}, \text{beq}, lb, ub)$ определяет набор нижних и верхних ограничений на конструируемые переменные x так, что решение всегда находится в диапазоне $lb \leq x \leq ub$. Устанавливается $\text{Aeq}=[]$ and $\text{beq}=[]$ в случае отсутствия равенств.

• $x = \text{fmincon}(\text{fun}, x_0, A, b, \text{Aeq}, \text{beq}, lb, ub, \text{nonlcon})$ подчиняет минимизацию определенных в `nonlcon` `fmincon` нелинейных неравенств $c(x)$ или равенств $\text{seq}(x)$ такому оптимуму, что $c(x) \leq 0$ и $\text{seq}(x) = 0$. Устанавливается $lb=[]$ и/или $ub=[]$ в случае отсутствия ограничений.

Пример 1. Пусть требуется найти минимум функции $f(x) = -x_1 x_2 x_3$ при начальном $x = [10; 10; 10]$ и при наличии ограничения $0 \leq x_1 + 2x_2 + 2x_3 \leq 72$

Решение. Вначале создадим m-файл, определяющий целевую функцию:

```
function f=myfun(x)
f=-x(1)*x(2)*x(3);
```

Затем запишем ограничения в виде неравенств:

$$\begin{aligned} -x_1 - 2x_2 - 2x_3 &\leq 0 \\ x_1 + 2x_2 + 2x_3 &\leq 72 \end{aligned}$$

или в матричной форме:

$$Ax \leq b, \quad \text{где } A = \begin{bmatrix} -1 & -2 & -2 \\ 1 & 2 & 2 \end{bmatrix}, b = \begin{bmatrix} 0 \\ 72 \end{bmatrix}$$

Теперь нахождение решения представляется следующим образом:

```
>> A = [-1 -2 -2; 1 2 2]
>> b = [0; 72]
>> x0 = [10; 10; 10]; % Стартовое значение
>> [x, fval] = fmincon('myfun', x0, A, b)
x = 24000
12.0000
12.0000
Fval = -3.4560e+003
```

3.3. Задачи оптимизации сетей телекоммуникации

Для проектирования сетей телекоммуникации можно выделить три основные задачи оптимизации [1, 11, 12].

1. **Оптимизация топологии сети.** Даны узлы сети в некоторых географических точках и прогнозируемые потребности (спрос) на передачу данных. Необходимо решить, какие узлы связать каналами связи. Выбор топологии нужно осуществлять в определенном классе сетей: звездообразных, древовидных, иерархических, т.е. необходимо зафиксировать тип структуры.

2. **Определение оптимальных пропускных способностей каналов.**

Даны множества возможных технологий для сетей передачи данных. Необходимо определить для каждого канала связи его тип и пропускную способность.

3. **Определение оптимального маршрута передачи.** Из заданного множества технологий и возможных пропускных способностей необходимо выбрать те маршруты передачи, которые бы удовлетворяли требованиям заданного качества обслуживания.

При организации сетей одной из основных задач является распределение потоков информации по кратчайшим путям. Под такими путями понимают пути передачи информации, кратчайшие по времени передачи или протяженности, или пути с минимальными помехами, числом задействованных узлов, стоимостью и т.п. Таким образом, оптимизация путей может проводиться по различным критериям и выбранные пути должны обеспечивать при заданных требованиях наиболее эффективное использование линий и узлов связи. В связи с этим имеет смысл проведение сравнительного анализа существующих на сегодняшний день алгоритмов маршрутизации с целью выработки закономерностей эволюции топологии и методики управления потоками.

Было выявлено, что использование адаптивной маршрутизации может уменьшить среднее время нахождения пакета в сети, позволяет минимизировать затраты на его доставку получателю в сетях с разнородным трафиком, увеличить общую надежность сети за счет возможности автоматического выбора альтернативного маршрута на основании данных о топологии сети. Однако, данные преимущества достигаются увеличением нагрузки на

вычислительные центры узлов коммутации, поэтому использование адаптивной маршрутизации ограничено размерами автономной системы (домена). Очевидно, что при этом возникает проблема разработки математического обеспечения оптимизации процедур выбора маршрута с целью снижения нагрузки на вычислительные центры.

На этапе выбора оптимального маршрута для отправки пакетов следующему узлу сеть будем рассматривать как взвешенный ориентированный граф. Вершины графа представляют маршрутизаторы, а дуги, соединяющие эти вершины, - физические линии между вершинами. Каждой линии связи соответствует некоторое интегральное значение, представленное посредством «стоимости» пересылки пакета по ней, которое может зависеть как от физической длины линии, временных затрат при передаче данных, так и от финансовых издержек транспортировки пакета. В настоящее время известен ряд алгебраических методов, позволяющих в определенной степени описать этот процесс с тем, чтобы получать результаты в форме, удобной для последующих исследований. А для этого необходимо сделать усилия по классификационному мониторингу известных методов с целью оценки их прикладной значимости и ограничений их использования.

Методы динамического программирования

Задачам отыскания кратчайших путей передачи информации присущи три основных свойства динамического программирования: многоходовой выбор, аддитивность и независимость оптимального пути от предыстории. В самом деле, исследование процесса передачи сообщений от узла к узлу в сети распадается на отдельные этапы (многоходовой выбор); длина пути, состоящего из нескольких ветвей, равна сумме длин этих ветвей (свойство аддитивности), и, наконец, кратчайший путь передачи сообщений из какого-либо узла не зависит от того, как сообщение попало в этот узел, а только от расположения этого узла в сети (свойство независимости от предыстории). На практике нашли применение: метод Флойда-Уоршелла, метод Беллмана-Форда, а также менее известный матричный метод.

Метод Флойда-Уоршелла, основанный на понятиях базисной линии связи и тернарной операции, применяется для

нахождения кратчайших расстояний между всеми вершинами взвешенного ориентированного графа. Достоинства данного метода – простота реализующего алгоритма и возможность получения маршрутной информации сразу для всех узлов сети, что делает его применением целесообразным при централизованных структурах управления информационными потоками. Говоря о сложности данного метода при программной реализации, стоит отметить, что три вложенных цикла содержат операцию, выполняемую за константное время, то есть алгоритм имеет кубическую сложность. В настоящее время существуют способы ускорения систем, использующих данный метод, позволяющих снизить сложность с до n^2 , где n – количество узлов сети. Речь идет об использовании битовых масок (в модели с промежуточным хранением результатов вычислений в RAM) и специализированных наборов микропроцессорных команд, как SSE (Streaming SIMD Extensions), в которых преимущество в производительности достигается в случае проведения одной и той же последовательности операций над разными

данными. Метод Беллмана-Форда [9] применяется для поиска кратчайшего пути во взвешенном графе. Основным достоинством является возможность расчета пути в графе, в котором есть ребра с отрицательным весом. Если при выполнении последней итерации длина кратчайшего пути до какой-либо вершины строго уменьшилась, то в графе есть отрицательный цикл. Можно отслеживать изменения в графе и, как только они закончатся, дальнейшие итерации будут бессмысленны. Данный метод используется в протоколе маршрутизации RIP (Routing Information Protocol) и его модификациях, причем некоторые его модифицированные версии задействованы в небольших сетях (не более 15 рабочих станций), в которых не требуется больших вычислительных мощностей для расчета путей, а также - для сетей с почти неизменной структурой.

Графовые алгоритмы в топологии сети/особое место в математическом обеспечении детерминированных процедур выбора трафика передачи информационных потоков занимают методы Флойда и Дейкстры, базирующиеся на принципе оптимальности.

Метод Дейкстры позволяет находить кратчайшие пути от любой из вершин графа до всех остальных. Метод работает только

для графов без ребер отрицательного веса, хотя в настоящее время существуют обобщенные методы для устранения данного недостатка (метод Дейкстры с потенциалами). Суть метода Дейкстры состоит в поэтапном наращивании дерева кратчайших маршрутов от исходного узла. При этом необходимо, чтобы после добавления на каждом этапе линии связи и узла вновь образуемый кратчайший маршрут был минимально возможным по всем конечным узлам, еще не вошедшим в дерево. В процессе построения дерева кратчайших маршрутов вычисляются векторы весов маршрутов и корректируются векторы начальных компонент кратчайших маршрутов. Сложность метода Дейкстры зависит от способа нахождения вершины v , а также от способа хранения множества непосещенных вершин и способа обновления меток. В графе G , n и m соответственно количество вершин и ребер для поиска вершины с минимальной длиной пути до вершины v , просматривается все множество n . Метод Дейкстры широко применяется в сетевом программировании и технологиях, например, его используют в протоколе OSPF (Open Shortest Path First) для устранения кольцевых маршрутов. Использование модифицированного алгоритма Дейкстры, как эффективного инструмента для распределения входных информационных потоков в магистральных IP-сетях с протоколом OSPF, позволяет улучшить робастность сети к информационным перегрузкам [14]. При этом возможно в качестве критерия распределения информационных потоков использовать остаточную пропускную способность канала. Необходимо отнести к положительным качествам протокола относительную простоту реализации.

Метод Джонсона позволяет найти кратчайшие пути между всеми парами вершин взвешенного ориентированного графа. Данный метод работает, если в графе содержатся ребра с положительным или отрицательным весом, но отсутствуют циклы с отрицательным весом. В методе Джонсона используется метод Беллмана-Форда и метод Дейкстры, реализованные в виде подпрограмм. Ребра хранятся в виде списков смежных вершин. Если в методе Дейкстры неубывающая очередь с приоритетами реализована в виде фибоначиевого множества, то время работы метода Джонсона равно . При более простой реализации неубывающей очереди с приоритетами время работы становится равным , но для разреженных графов эта величина в

асимптотическом пределе ведет себя с точки зрения нечеткого критерия логики лучше, чем время работы алгоритма Флойда-Уоршелла.

На каждый вход узла сети поступает поток нагрузки, характеризующийся своими параметрами. Под j -м каналом связи будем понимать j -ый вход рассматриваемого узла связи.

Вероятность своевременной доставки сообщения в j -м канале связи равна:

$$Q_j(\mu_j) = \frac{\mu_{эj} - \lambda_j}{\mu_{эj} - \lambda_j + \theta_{эj}} \quad (3.15)$$

Среднее время доставки сообщения в j -м канале связи равно:

$$T_j(\mu_j) = \frac{\mu_{эj} T_{пj} K_{пj} + 1}{\mu_{эj} - \lambda_j} \quad (3.16)$$

Все характеристики относятся к j -му каналу связи и означает:

λ_j - интенсивность потока данных;

μ_j - интенсивность обслуживания потока данных;

$\mu_{эj} = \mu_j K_{эj}$ - эквивалентная пропускная способность канала связи,

$K_{гj} = \frac{T_{иj}}{T_{иj} + T_{пj}}$ - коэффициент готовности,

$T_{иj}(T_{пj})$ - среднее время исправной работы,

v_j - интенсивность старения информации,

$v_{эj} = v_j + \mu_{эj} f_j$ - эквивалентная интенсивность старения,

$f_j = K_{пj} l(K_{гj} + \frac{1}{v_j T_{пj}})$ - коэффициент ненадежности канала связи.

Задача 1. Оптимизационная задача расчета каналов сети минимальной стоимости, удовлетворяющего требованиям для обоих типов нагрузки, формулируется следующим образом:

$$S \rightarrow \min, Q \geq Q_{зад}, T \geq T_{зад} \quad (3.17)$$

где S - стоимость системы, Q - средняя вероятность своевременной доставки пакетов,

T - среднесетевое значение математического ожидания времени задержки пакетов,

$Q_{зад}, T_{зад}$ - заданные значения Q и T , причем $Q_{зад} \in [Q_{мин}, Q_{макс}]$, $T_{зад} \in [T_{мин}, T_{макс}]$.

Границы указанных диапазонов вычисляются по характеристикам каналов сети. Важно отметить, что из-за неабсолютной надежности каналов $Q_{макс} < 1$, а $T_{мин} > 0$.

Задача 2. $S \rightarrow \min, T \leq T_{зад} \quad (3.18)$

при $Q_{зад} > Q_2$ данная задача аналогичным образом становится эквивалентной задаче 3.

Задача 3. $S \rightarrow \min, Q \geq Q_{зад} \quad (3.19)$

Если же $Q_{зад} \in (Q_1, Q_2)$, то необходимо решать задачу 1 при наличии двух ограничений в виде равенств, что составляет содержание задачи 4.

Задача 4. $S \rightarrow \min, Q = Q_{зад}, T = T_{зад} \quad (3.20)$

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ГЛАВЕ 3

1. Объясните важность линейного программирования в задачах оптимизации систем.
2. Объясните алгоритм симплекс-метода оптимизации.
3. Этапы решения задач линейного программирования.
4. Приведите примеры решения задач на основе линейного программирования.
5. Важность линейного программирования в телекоммуникациях.
6. Как формулируется транспортная задача оптимизации?
7. Приведите примеры решения транспортных задач.
8. Особенности решения транспортной задачи в телекоммуникации.
9. Методы решения задач нелинейного программирования.
10. Какие функции для оптимизации имеются в среде MATLAB?
11. Приведите примеры оптимизационных задач в сети телекоммуникации.
12. С помощью какой функции MATLAB можно решить задачи линейного программирования?

13. С помощью какой функции MATLAB можно решить задачи нелинейного программирования?
14. С помощью какой функции MATLAB можно решить задачи целочисленного линейного программирования?
15. Какую функцию MATLAB можно использовать для решения задач линейного программирования смешанного типа?

4. ОПТИМИЗАЦИЯ СИСТЕМ И СЕТЕЙ ТЕЛЕКОММУНИКАЦИИ

4.1. Модель и расчет характеристик систем передачи данных

В устройствах доступа и агрегации реализуются функции канального и физического уровней OSI [11-14]. В существующих сетях доступа, построенных в соответствии с эталонной моделью взаимодействия открытых систем OSI (Open System Interconnection), длина блока (кадра, пакета) данных протоколов канального уровня (HDLC, LAPB, LAPM, PPP, LLC) составляет порядка сотни байтов [15]. Поэтому, достоверность данных на канальном уровне проверяется с помощью алгоритмов обратной связи, а на физическом уровне только на основе помехоустойчивого кодирования. При этом кадр данных делится на несколько частей короткой длины (слов, символов) и осуществляется помехоустойчивое кодирование этих слов.

В соответствии с OSI, блок данных (пакет) сетевого уровня передается канальному уровню сети (рисунок 4.1). На канальном уровне формируется блок данных (кадр) канального уровня со следующими полями: флаг-начало, адресное поле, управляющее поле, поле данных, поле контрольной суммы и флаг-окончание. Дополнение остатка от деления полинома кадра данных на стандартный образующий полином является контрольной суммой. В поле данных кадра вставляется пакет данных сетевого уровня.

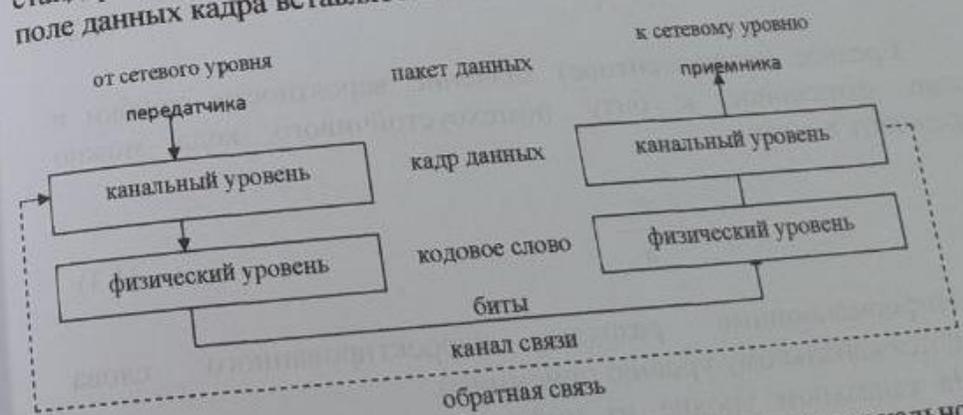


Рис. 4.1. Процесс передачи данных в физическом и канальном уровнях сети

Сформированный кадр данных канального уровня передается к физическому уровню сети. На физическом уровне кадр данных длиной n_k bit делится на слова длиной k bit. При этом получается n_k/k слов, где $n_k = \lfloor n_k/k \rfloor$.

Далее, осуществляется помехоустойчивое кодирование слов. В соответствии с выбранным методом кодирования определяются проверочные разряды длиной r bit. Тогда длина каждого слова физического уровня равна $n_s = k+r$.

Кодовые слова передаются по каналу связи со скоростью C bit/s. Пусть канал связи является биномиальным с вероятностью ошибки единичного элемента p . Тогда вероятность возникновения i -кратных ошибок в слове определяется по формуле:

$$P_{ij} = C_{n_s}^i p^i (1-p)^{n_s-i} \quad (4.1)$$

Кодовые слова по каналу связи поступают на физический уровень принимающей стороны. Здесь производится декодирование и исправление ошибок в слове. Ошибка с кратностью (весом) t_n может быть исправлена при $d_0 \geq 2t_n + 1$, где d_0 - минимальное кодовое расстояние Хэмминга.

Если при декодировании слова исправляются все ошибки кратностью t_n , то вероятность ошибочного приема слова равна:

$$P_{of} = \sum_{i=t_n+1}^{n_s} P_{ij} \quad (4.2)$$

Среднее (эквивалентное) значение вероятности ошибки в слове, отнесенное к биту помехоустойчивого кода можно определить в виде:

$$p_s = \frac{P_{of}}{n_s} \quad (4.3)$$

Информационные разряды корреktированного слова передаются канальному уровню приемника.

На канальном уровне из информационных разрядов слов формируется кадр данных. Вероятность возникновения i -кратных ошибок в кадре данных определяется в виде:

$$P_{in} = C_{n_s}^i p_s^i (1-p_s)^{n_s-i} \quad (4.4)$$

Вероятность правильного приема кадра данных равна:

$$P_{on} = (1-p_s)^{n_s} \quad (4.5)$$

Вероятность необнаруженной ошибки в кадре данных определяется по приближенной формуле [13]:

$$P_{on} \approx \frac{1}{2^{g_c}} \sum_{i=d_0}^{n_s} C_{n_s}^i p_s^i (1-p_s)^{n_s-i} \quad (4.6)$$

где g_c - количество разрядов CRC. В стандартных протоколах $g_c=16$ или $g_c=32$.

Вероятность обнаружения ошибки в кадре данных определяется в виде:

$$P_{oo} = 1 - (P_{on} + P_{in}) \quad (4.7)$$

Таким образом, после проверки достоверности кадра данных методом сопоставления контрольных сумм с вероятностью P_{in} обнаруживается наличие ошибки и передается кадр с отрицательной квитанцией длиной n_{kv} в передающую сторону. В противном случае $(P_{on} + P_{in})$ передается кадр с положительной квитанцией. Передающая сторона при приеме кадра с отрицательной квитанцией повторяет предыдущий переданный кадр данных, а при приеме кадра с положительной квитанцией начинает передачу следующего кадра данных. Из-за малой вероятности ошибочного приема квитанции можно считать, что обратный канал идеальный. Данный алгоритм используется во многих стандартных протоколах и называется методом передачи данных с решающей обратной связью с ожиданием или методом автоматического повтора запроса (ARQ, Automatic Repeat Quest) [11].

Вероятность (распределения) количества повторов передачи кадра данных определяется в виде:

$$P(k_s) = (1 - P_{oo}) P_{oo}^{k_s-1}, \quad k_s = \overline{1, N_s}, \quad (4.8)$$

где: N_s - максимальное количество повторов.

Вероятность остаточного искажения кадра данных после повторных передач определяется в виде [13]:

$$P_{ост} = \sum_{k=1}^{N_p} P_{ост} P_{ост}^{k-1} = \frac{P_{ост} (1 - P_{ост}^{N_p+1})}{1 - P_{ост}} \quad (4.9)$$

Процедура повтора увеличивает общее время обслуживания (передачи) кадров данных. Удобным математическим аппаратом для определения дискретного времени (интервала) обслуживания кадра данных является z-преобразование.

Z-преобразование дискретного времени обслуживания кадра данных с учетом повторов определяется в виде:

$$f_o(z) = \sum_{k=1}^{N_p} P(k_n) [f_k(z) f_{ks}(z)]^{k_n} \quad (4.10)$$

где $f_k(z)$ и $f_{ks}(z)$ - z-преобразования дискретного времени передачи кадра данных и кадра с квитанцией соответственно, и определяются в виде:

$$f_k(z) = z^{-m_k n_f} \quad (4.11)$$

$$f_{ks}(z) = z^{-m_{ks} m_f} \quad (4.12)$$

где $m_{ks} = \left\lfloor \frac{n_{ks}}{n_f} \right\rfloor$.

Подставив (4.8) в (4.10), находим:

$$f_o(z) = \frac{(1 - P_{ост}) f_k(z) f_{ks}(z) [1 - (P_{ост} f_k(z) f_{ks}(z))^{N_p+1}]}{1 - P_{ост} f_k(z) f_{ks}(z)} \quad (4.13)$$

Дискретное среднее время обслуживания кадра данных определяется в виде:

$$\bar{n}_o = - \frac{df_o(z)}{dz} \Big|_{z=1} \quad (4.14)$$

Подставив (4.13) в (4.14) и выполнив необходимые операции с учетом (2.32) и (2.33) при $N_p \rightarrow \infty$, получаем:

$$\bar{n}_o = \frac{n_f (m_k + m_{ks})}{1 - P_{ост}} \quad (4.15)$$

Тогда среднее время обслуживания кадра данных равно:

$$\bar{t}_{обс} = \frac{\bar{n}_o}{C} \quad (4.16)$$

Скорость передачи в канале связи C определяется технологией доступа. Исходными данными для расчета вероятностно-временных характеристик являются: длина кадра данных 2000 bit, длина кадра с квитанцией 40 битов и скорость передачи 64 Kbit/s, при различных длинах кодовых слов.

На рисунке 4.2 представлен график зависимости среднего времени обслуживания кадров данных от вероятности ошибок битов в канале связи при различных длинах кодовых слов, образованных с помощью циклического кода, исправляющего однократные ошибки.

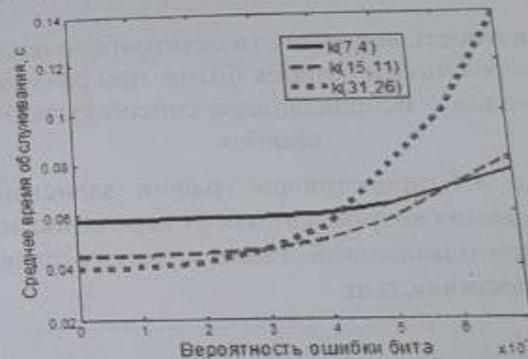


Рис. 4.2. Зависимость среднего времени обслуживания кадров данных при различных длинах циклических кодов с исправляющей способностью однократных ошибок

Сравнительный анализ показывает, что по времени обслуживания кадров данных код (31,26) имеет преимущество при $p < 3 \cdot 10^{-3}$, код (15,11) при $3 \cdot 10^{-3} \leq p \leq 6 \cdot 10^{-3}$, код (7,4) при $p > 6 \cdot 10^{-3}$.

На рисунке 4.3 приведен график зависимости вероятности остаточного искажения кадра данных после повторных передач от вероятности ошибок битов в канале связи при различных длинах циклических кодов с исправляющей способностью однократных ошибок.

Из рисунка 4.3 видно, что при одинаковой исправляющей способности код с наименьшей длиной имеет наименьшую вероятность остаточного искажения. При $p < 10^{-4}$ рассматриваемые

коды обеспечивают почти одинаковую вероятность искажения кадра данных, но код наибольшей длины (31,26) имеет наименьшее время обслуживания.

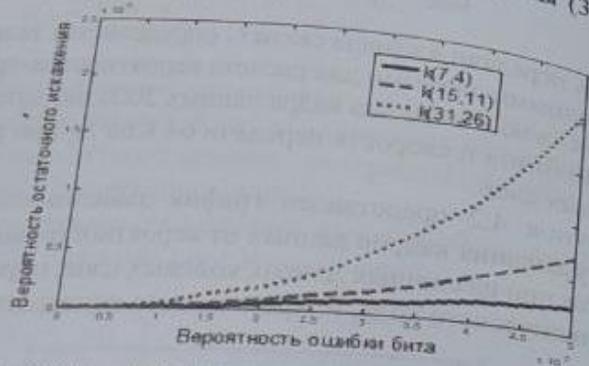


Рис. 4.3. Зависимость вероятности остаточного искажения кадра данных от вероятности ошибок битов при различных длинах циклических кодов с исправляющей способностью однократных ошибок

На рисунке 4.4 представлен график зависимости среднего времени обслуживания кадров данных от вероятности ошибки битов в канале связи при одинаковой длине кодового слова с различной исправляющей способностью.

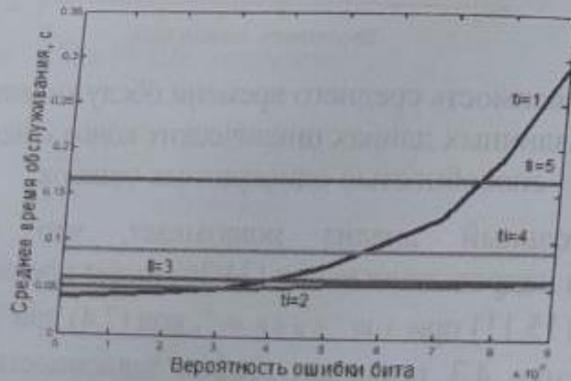


Рис. 4.4. Зависимость среднего времени обслуживания кадров данных от вероятности ошибки битов в канале связи при одинаковой длине кодового слова с различной исправляющей способностью

При $p < 3 \cdot 10^{-3}$ наименьшее среднее время обслуживания кадров имеет код с наименьшей исправляющей способностью (исправляющий однократные ошибки), а при $p \geq 3 \cdot 10^{-3}$ код, исправляющий двукратные ошибки. Коды, исправляющие более чем двукратные ошибки, имеют более высокое среднее время обслуживания кадров. При этом среднее время обслуживания кадров данных почти независимо от вероятности ошибок битов в областях $p < 10^{-3}$.

На рисунке 4.5 представлен график зависимости вероятности остаточного искажения кадров данных от вероятности ошибки битов при различных исправляющих способностях кода.

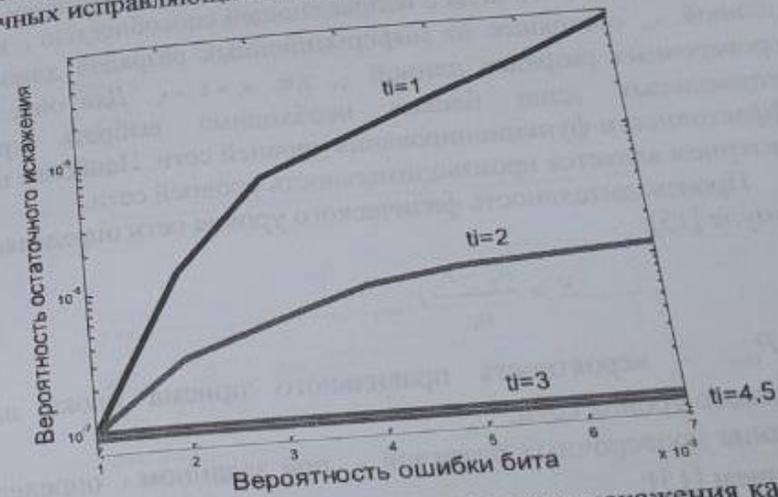


Рис. 4.5. Зависимость вероятности остаточного искажения кадров данных от вероятности ошибки битов при различных исправляющих способностях кода

Из рисунка 4.5 видно, что чем больше исправляющая способность кода, тем меньше вероятность остаточного искажения. При $p < 4 \cdot 10^{-3}$ коды, исправляющие более чем двукратные ошибки, обеспечивают почти одинаковые вероятности остаточного искажения кадра данных.

4.2. Оптимизация систем передачи данных

Задача определения оптимальных длин блоков данных решена в многочисленных работах, например в [11,15]. В этих работах

решается без учета особенностей функционирования систем передачи данных в соответствии с эталонной моделью OSI каждый вышестоящий уровень пользуется услугами нижестоящего канального уровня СПД. В работе [11] оптимальная длина кадра данных физического уровня СПД n в [15] без учета длины кодового слова определена без учета длины кадра данных.

В данном подразделе рассматриваются задачи оптимизации длин блоков физического и канального уровней сети [16]. Блоком данных физического уровня сети является кодовое слово помехоустойчивого кода с исправляющей способностью t и общей длиной n , состоящее из информационных разрядов длиной k и проверочных разрядов длиной r , т.е. $n = k + r$. Для определения оптимальных длин блоков необходимо выбрать критерии эффективности функционирования уровней сети. Наиболее полным критерием является производительность уровней сети.

Производительность физического уровня сети определяется по формуле [15]:

$$E = \frac{n_1 - r_1}{n_1} P_{1mn}, \quad (4.17)$$

где P_{1mn} - вероятность правильного приема блока данных физического уровня сети. Длина проверочных разрядов r_1 при заданном t определяется выражением [15]:

$$r_1 = \text{ceil} \left[\log_2 \left(1 + \sum_{i=1}^t C_{n_1}^i \right) \right]. \quad (4.18)$$

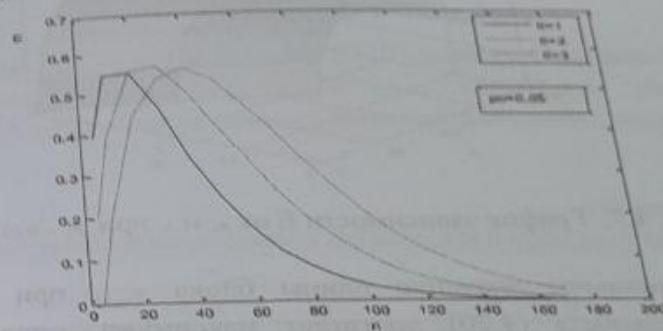
Пусть канал связи является биномиальным с вероятностью ошибок единичного элемента p_0 . Тогда P_{1mn} определяется выражением:

$$P_{1mn} = \sum_{i=0}^t C_{n_1}^i p_0^i (1 - p_0)^{n_1 - i}. \quad (4.19)$$

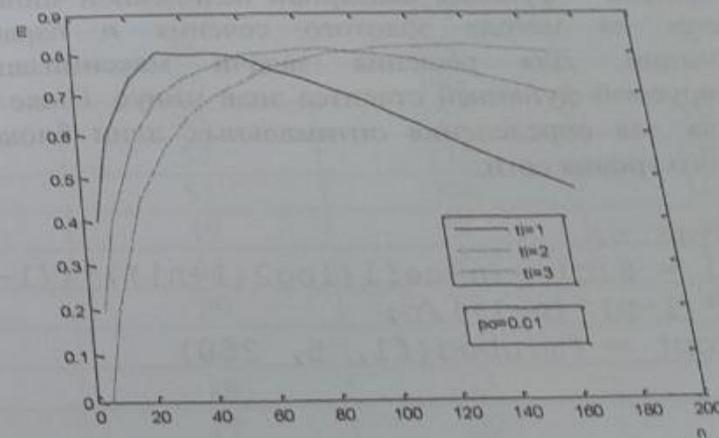
Производительность физического уровня сети (4.17) с учетом (4.18) и (4.19) определяется в виде:

$$E = \frac{n_1 - \text{ceil} \left[\log_2 \left(1 + \sum_{i=1}^t C_{n_1}^i \right) \right]}{n_1} \sum_{i=0}^t C_{n_1}^i p_0^i (1 - p_0)^{n_1 - i} \quad (4.20)$$

На рисунке 4.6 представлены графики зависимостей производительности физического уровня сети от длины блока данных при различных t и p_0 .



a)



b)

Рис. 4.6. Зависимости производительности физического уровня сети от длины блока данных при $p_0 = 0,05$ (a) и $p_0 = 0,01$ (b)

Из рисунка 4.6 видно, что производительность E имеет максимальное значение при определенных значениях n , t и p_0 .

более наглядно показано на графике поверхности (рисунок 4.7), созданной командой MATLAB surf(x,y,E) [10].

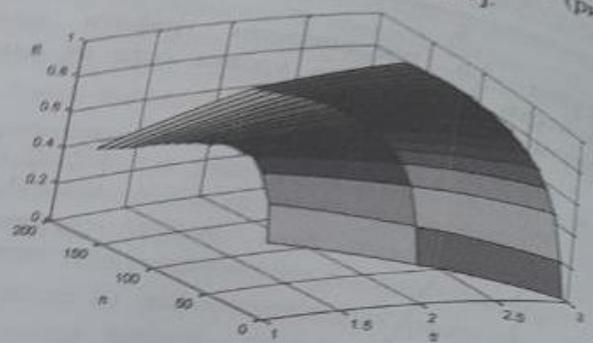


Рис. 4.7. График зависимости E от n, и l, при p₀ = 0,01

Оптимальное значение длины блока n_{opt}, при котором производительность (4.20) достигает максимума, определим с помощью функции fminbnd пакета Optimization Toolbox MATLAB [10]. fminbnd – функция скалярной нелинейной минимизации и базируется на методе золотого сечения и параболической интерполяции. Для решения задачи максимизации перед оптимизируемой функцией ставится знак минус. Ниже приведены выражения для определения оптимальных длин блоков данных физического уровня сети.

```
>> syms n;
>> f1 = @(n) -(n-ceil(log2(1+n)))*((1-p)^n+n*p*(1-p)^(n-1))/n;
>> nlopt = fminbnd(f1, 5, 200)
```

```
>> syms n;
>> f2 = @(n) -(n-ceil(log2(1+n*(1+n)/2)))*((1-p)^n+n*p*(1-p)^(n-1)+n*(n-1)/2*(p^2)*(1-p)^(n-2))/n;
>> nlopt = fminbnd(f2, 10, 200)
```

```
>> syms n;
>> f3 = @(n) -(n-ceil(log2(1+n*(1+n)/2+(n-2)*(n-1)*n/6)))*((1-p)^n+n*p*(1-p)^(n-1)+(n*(n-1)/2)*(p^2)*(1-p)^(n-2)+(n-2)*(n-1)*n/6*(p^3)*(1-p)^(n-3))/n;
>> nlopt = fminbnd(f3, 20, 200)
```

Функции f1 (5), f2 (6) и f3 (7) записаны согласно (2.41) соответственно для l₁=1, l₁=2 и l₁=3. Результаты расчетов оптимальных длин блоков данных физического уровня сведены в таблицу 4.1

Таблица – 4.1

Оптимальные длины блоков физического уровня сети

p ₀	n _{opt}		
	l ₁ = 1	l ₁ = 2	l ₁ = 3
0,001	216	361	465
0,002	127	255	385
0,003	103	217	347
0,004	86	174	278
0,005	75	149	232
0,006	63	127	204
0,007	57	116	180
0,008	53	106	164
0,009	49	98	146
0,01	46	89	138
0,02	29	55	82
0,03	23	41	62
0,04	19	35	46
0,05	17	29	36

Графики зависимости оптимальных длин блоков данных n_{opt} от вероятности ошибки единичного элемента p, при различных l₁ приведены на рисунке 4.8.

Таким образом, оптимальная длина блока физического уровня уменьшается с ухудшением качества канала связи. При задан

качестве канала связи оптимальную длину блока данных можно увеличить путем повышения исправляющей способности помехоустойчивого кода.

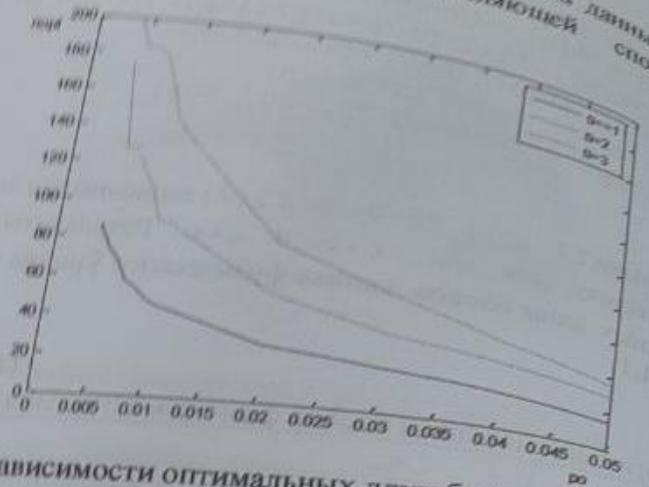


Рис. 4.8. Зависимости оптимальных длин блоков данных физического уровня от вероятности ошибок в канале связи при различных исправляющих способностях помехоустойчивого кода

Полученные результаты верны только для одноуровневых систем. При этом метод помехоустойчивого кодирования реализуется как самостоятельная система. Ниже рассматривается многоуровневая система, где метод помехоустойчивого кодирования оказывает услугу вышестоящему уровню.

Определение оптимальной длины блока данных физического уровня с учетом длины блока данных канального уровня сети

На канальном уровне формируется кадр данных длиной \$n_2\$ и передается физическому уровню сети. Пусть на физическом уровне реализован помехоустойчивый код \$(n_1, k_1)\$. Тогда на физическом уровне кадр данных длиной \$n_2\$ bit делится на слова длиной \$k_1\$ bit. При этом получается \$m_1\$ слов, где \$m_1 = \lfloor n_2 / k_1 \rfloor\$.

После передачи и приема \$m_1\$ слов на канальном уровне приемника собирается кадр данных. Вероятность правильного приема кадра равна:

$$P_{2,ns} = (P_{1,ns})^{m_1}, \quad (4.21)$$

где: \$P_{1,ns}\$ определяется выражением (4.19).
С вероятностью \$Q_{1,ns} = 1 - P_{1,ns}\$ кадр данных принимается с ошибкой и начинается процедура повторной передачи кадров данных согласно протоколам канального уровня.

Среднее количество повторов при очень низкой вероятности обнаружения ошибок в кадре данных можно определить выражением:

$$k_n = \frac{1}{P_{2,ns}} \quad (4.22)$$

Процедура повторов увеличивает время обслуживания кадров данных и уменьшает производительность канального уровня сети \$E_1\$. Поэтому производительность \$E_1\$ можно определить по формуле:

$$E_1 = \frac{n_2}{k_n(n_2 + m_1 r_1)} P_{2,ns} \quad (4.23)$$

Выражение (4.23) с учетом (4.21) и (4.22) записывается в виде:

$$E_1 = \frac{n_2}{n_2 + m_1 r_1} (P_{1,ns})^{m_1 + 1} \quad (4.24)$$

На рисунке 4.9 представлены графики зависимости производительности канального уровня от длины блоков данных физического уровня при различных длинах блоков данных канального уровня сети.

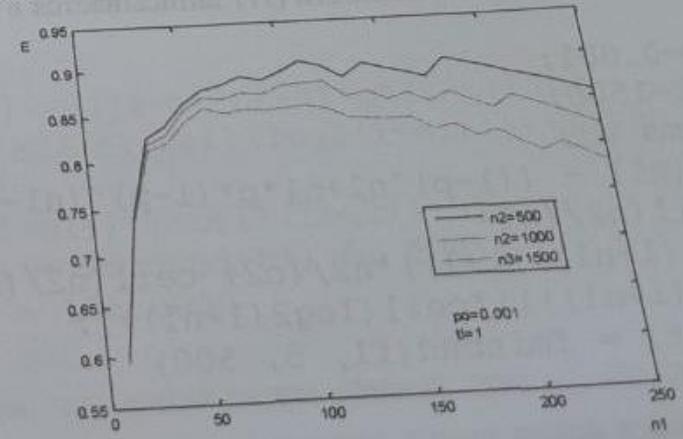


Рис. 4.9. Зависимость \$E_1\$ от \$n_1\$ при \$p_0 = 0.001, r_1 = 1\$ и различных \$n_2\$

На рисунке 4.10 показан график зависимости оптимальной длины блока данных от длины кадра данных.

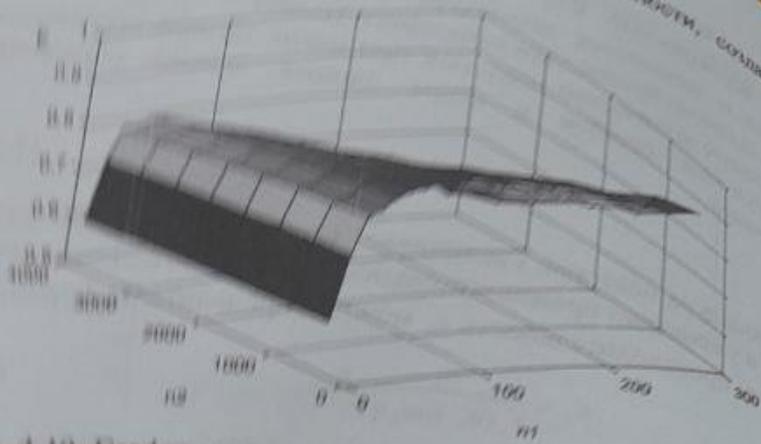


Рис. 4.10. График зависимости n_{opt} от n и p , при $p_s = 0,001$, $i_1 = 1$

Из рисунков 4.9 и 4.10 видно, что при определенных значениях n и p производительность f достигает максимального значения. Оптимальное значение n , при заданном n_2 и i_1 , определим с помощью функции `fminbnd`. Для случая, когда на физическом уровне используется помехоустойчивый код с исправляющей способностью $i_1 = 1$, функция производительности (f) записывается в виде:

```
>>p=0,001;
>>n2=1500;
>>syms n1;
f1=@(n1) - ((1-p)^n1+n1*p*(1-p)^(n1-
2))^(ceil(n2/(n1-
ceil(log2(1+n1))))+1))*n2/(n2+
ceil(log2(1+n1)))*ceil(log2(1+n1));
>>nlopt1 = fminbnd(f1, 5, 500)
```

Результаты расчетов оптимальных длин блоков данных физического уровня сети сведены в таблицу 4.2.

Таблица - 4.2

Оптимальные длины блоков данных физического уровня с учетом длин блоков данных канального уровня сети

p	n_{opt}			
	$n_2 = 500$	$n_2 = 1000$	$n_2 = 1500$	$n_2 = 2000$
0,001	133	118	87	86
0,003	62	48	28	26
0,005	26	23	18	16
0,007	23	19	14	13
0,009	20	14	13	12

Анализ таблицы 4.2 показывает, что с увеличением длины блока данных канального уровня и ухудшением качества канала связи, оптимальные длины блоков данных физического уровня уменьшаются.

Из таблиц 4.1 и 4.2 видно, что полученные оптимальные длины блоков данных помехоустойчивого кода для одноуровневых и многоуровневых систем различны.

Допустим, результаты таблицы 4.1 используются для многоуровневых систем. Например, при $p_s = 0,005$ оптимальная длина блока данных помехоустойчивого кода при $i_1 = 1$ равна $n_{opt} = 75$. Определим длину кадра данных, при которой достигается максимальная производительность канального уровня

```
p=0.005;
n1=75;
syms n2;
f1=@(n2) - ((1-p)^n1+n1*p*(1-p)^(n1-
1))^(ceil(n2/(n1-
ceil(log2(1+n1))))+1))*n2/(n2+
ceil(n2/(n1-
ceil(log2(1+n1))))*ceil(log2(1+n1)));
n2opt1 = fminbnd(f1, 5, 500)
n2opt1 = 136.0000
```

Получаем, что длина кадра данных при $n_{opt} = 75$ должна быть равна 136 bit. Однако, минимальная длина кадра в соответствии протоколами канального уровня должна быть 512 бит [14]. Поэтому результаты таблицы 4.1 нельзя использовать в многоуровневых системах.

Двухпараметрическая оптимизация систем передачи данных

В транзитных узлах связи (мостах и коммутаторах) не имеется возможность изменения длины кадра данных. Такая возможность имеется в конечных узлах обработки данных (ООД). Поэтому в ООД можно комплексно определить оптимальные значения n_{1opt} и n_{2opt} . Функция `fminsearch` позволяет найти минимум функции нескольких переменных без ограничений, то есть решение задачи безусловной оптимизации с использованием симплексного метода [10].

Пусть имеется вектор переменных $n=[n1,n2]$, где $n1$ и $n2$ являются переменными для длин блоков данных соответственно физического и канального уровней сети. Ниже на основе функции `fminsearch` приведены выражения для определения оптимальных значений $n1$ и $n2$ при которых производительность канального уровня сети достигает максимального значения

```
>> p=0.005;
>> syms n;
>> f=@(n) -((1-p)^n(1)+n(1)*p*(1-p)^(n(1)-
1))^(ceil(n(2)/(n(1)-
ceil(log2(1+n(1)))))+1))*n(2)/(n(2)+
ceil(n(2)/(n(1)-
ceil(log2(1+n(1))))))*ceil(log2(1+n(1)));
>> n0=[100,1000];
>> n = fminsearch(f,n0)
n =
15.0000 700.5149
```

Результаты расчетов оптимальных длин блоков данных при сведены в таблицу 4.3. Из таблицы 4.3 видно, что оптимальные значения n_{1opt} и n_{2opt} уменьшаются с ухудшением качества канала связи.

Таблица - 4.3

Оптимальные длины блоков данных физического и канального уровней сети

p_i	n_{1opt}	n_{2opt}
0.0008	99	1570
0.0009	82	1286
0.001	74	1196
0.003	25	906
0.005	15	700

4.3. Модель и расчет характеристик маршрутизатора

В соответствии с концептуальной моделью телекоммуникации система массового обслуживания (СМО), моделирует различные узлы (маршрутизаторы) сети. Для того, чтобы определить характеристики каждой системы массового обслуживания (узла сети).

Допустим, сеть состоит из k СМО. Основной характеристикой СМО $_i$ является средняя длина очереди заявок (кадров, пакетов) Q_i , где i - номер СМО. Если известна Q_i , то по формуле Литтла определяется среднее время ожидания пакетов в очереди СМО $_i$:

$$\bar{t}_{i,ож} = \frac{Q_i}{\lambda_i}, \quad (4.25)$$

где $\lambda_i = 1/\bar{t}_{i,ин}$ - интенсивность поступления пакетов в СМО $_i$, $\bar{t}_{i,ин}$ - средний интервал времени между пакетами на входе СМО $_i$. Среднее время задержки пакетов в СМО $_i$:

$$\bar{t}_{i,заб} = \bar{t}_{i,ож} + \bar{t}_{i,обс}, \quad (4.26)$$

где $\bar{t}_{i,обс}$ - среднее время обслуживания пакетов (кадров) в СМО $_i$. Загрузка СМО $_i$ определяется по формуле:

$$\rho_i = \frac{\lambda_i}{\mu_i}, \quad (4.27)$$

где: $\mu_i = 1/I_{обс}$ - интенсивность обслуживания пакетов в СМО,
 $\rho_i < 1$.

Задержка пакетов в сети определяется как сумма задержек в СМО

$$\bar{t}_{обс} = \sum_{i=1}^k \bar{t}_{i,обс} \quad (4.28)$$

Для расчета характеристик сети необходимо знать законы распределения интервалов поступления и времени обслуживания пакетов.

Показатели времени обслуживания пакетов определяются в соответствии с протоколами, реализуемых в узлах сети.

В многочисленных работах принято предположение о пуассоновском характере входящего потока [7,8,17]. Анализ реального трафика данных в современных мультисервисных пакетных сетях связи показал некорректность использования пуассоновских моделей для расчета показателей QoS. Многочисленными исследованиями было доказано, что реальный трафик является фрактальным (самоподобным) случайным процессом [18]. Закон распределения интервалов времени между заявками (пакетами) в этих потоках является произвольным (G - general, общий).

Фрактальный трафик задается распределением Вейбулла [20]:

$$w(x) = \alpha \beta x^{\beta-1} e^{-\alpha x^\beta} \quad (4.29)$$

где α и β - некоторые параметры, влияющие на форму данного распределения.

Математическое ожидание $M[x]$ и дисперсия $D[x]$ распределения Вейбулла определяются следующим образом:

$$M[x] = \alpha \cdot \Gamma\left(\frac{\beta+2}{\beta}\right), \quad (4.30)$$

$$D[x] = \alpha^2 \left[\Gamma\left(\frac{\beta+2}{\beta}\right) - \Gamma^2\left(\frac{\beta+2}{\beta}\right) \right], \quad (4.31)$$

где $\Gamma(\cdot)$ - гамма функция.

Параметр распределения Вейбулла можно выразить через параметр Херста H :

$$\beta = 2 - 2H \quad (4.32)$$

В качестве примера в таблице 4.4 приведены значения фрактальности для различных типов трафика [18]. Учет самоподобных (фрактальных) свойств трафика позволит более точно описать его в моделях, что, в свою очередь, обеспечит возможность получения показателей качества обслуживания, соотносимых с реально наблюдаемыми.

Таблица 4.4

Тип трафика	Фрактальность трафика (H)
Ethernet	0.9
HTTP	0.75-0.92
Видео	0.6-0.9
Аудио	0.6-0.9
P2P	0.6-0.9

При исследовании СМО типа G/G/m (m - количество каналов) применялись различные методы и получены многочисленные приближенные результаты [17]. Однако, до настоящего времени не существует достаточно простых и точных, непосредственно применяемых на практике, формул для расчета показателей QoS.

В данном подразделе приведен метод расчета средней задержки пакетов в СМО с фрактальным трафиком [19, 20]. Для расчета средней задержки пакетов вначале необходимо определить среднюю длину очереди. В настоящее время для расчета средней длины очереди в СМО с фрактальным трафиком используются две формулы.

Первая формула [18], разработанная для системы G/G/m:

$$Q = P(\rho, m) \frac{\rho}{1-\rho} \cdot \frac{C_n^2 + C_{обс}^2}{2}, \quad (4.33)$$

где $P(\rho, m)$ - вероятность того, что пакет, придя в систему, застаёт все каналы занятыми; C_n^2 - квадратичный коэффициент вариации интервалов между пакетами; $C_{обс}^2$ - квадратичный коэффициент вариации времени обслуживания пакета; и m - количество каналов (обслуживающих устройств).

Вторая формула [18]:

$$Q = c \cdot \frac{\rho^{1/(2(1-H))}}{(1-\rho)^{H/(1-H)}} \quad (4.34)$$

где H - показатель самоподобности (Херста) трафика, c - постоянная (при экспоненциальном времени обслуживания $c = \rho/2$); при фиксированном времени обслуживания $c = \rho$.

При экспоненциальном входном потоке и экспоненциальном времени обслуживания и $m=1$ выражения (4.33) и (4.34) приводятся к хорошо известной формуле для системы M/M/1:

$$Q = \frac{\rho^2}{1-\rho} \quad (4.35)$$

Результаты расчетов показывают, что формулы (4.33) и (4.34) при одинаковых исходных данных дают разные значения средней длины очереди. Эта разница увеличивается с повышением загрузки и коэффициента Херста.

В [20] рассмотрены вопросы расчета характеристик СМО с фрактальным трафиком. Для этого проведено имитационное моделирование рассматриваемой системы в среде GPSS World [17].

Значения параметров входящего потока, определенные путем имитационного моделирования, сведены в таблицу 4.5. В таблицу также сведены параметры входящего потока, которые рассчитаны с помощью аналитических формул.

Таблица 4.5
Параметры фрактального трафика

ρ	H	Математическое ожидание		Дисперсия		Среднее квадратичное отклонение		Коэффициент вариации	
		имитация	аналит.	имитация	аналит.	имитация	аналит.	имитация	аналит.
1	0,5	1	1	1	1	1	1	1	1
8	0,6	1,133	1,13	2,039	2,039	1,428	1,42	1,26	1,26
5	0,7	1,505	1,5046	6,985	6,997	2,643	2,64	1,756	1,75
1	0,8	3,322	3,32	107,85	108,95	10,385	10,43	3,126	3,14
	0,9	121,24	120	2890000	3614400	1700	1901	14,02	15,83

Сравнительный анализ теоретических и экспериментальных данных, приведенных в таблице 4.5, показывает достоверность генерации самоподобного трафика на основе распределения Вейбулла.

Для расчета вероятности ожидания заявки в системе G/G/1 предложена формула:

$$P_{ож} = \frac{Q}{\rho \cdot \bar{t}_{ож}} \quad (4.36)$$

Для определения $P_{ож}$ должны быть известны длина очереди Q и среднее время ожидания $\bar{t}_{ож}$. Однако, эти характеристики, в свою очередь определяются через $P_{ож}$.

В [20] предлагается следующая формула для расчета средней длины очереди:

$$Q = \frac{\rho \cdot P_{ож}}{1 - P_{ож}} \quad (4.37)$$

Для систем M/G/1 $P_{ож} = \rho$. Вероятность ожидания заявок в системе G/G/1 можно определить как соотношение задержанных заявок n_z и общего количества поступивших заявок $n_{общ}$:

$$P_{ож} = \frac{n_z}{n_{общ}} \quad (4.38)$$

На основе результатов моделирования и аппроксимации, эмпирическая формула для расчета вероятности ожидания заявок имеет вид:

$$P_{ож} = a \cdot \rho^2 + b \cdot \rho + c, \quad (4.39)$$

где: a , b и c - коэффициенты аппроксимации, приведенные в таблице 4.6.

Таблица 4.6
Коэффициенты аппроксимации

H	a	b	c
	0	1	0
0,5	0	0,94	0,096
0,6	0	1,4	0,15
0,7	-0,98	1,2	0,41
0,8	-0,68	0,39	0,66
0,9	-0,28		

Полученные результаты показывают [20], что средняя длина очереди, рассчитанная на основе предложенной формулы (pred for), на 99 % совпадает с экспериментальными (imod) результатами.

4.4. Модель и расчет характеристик сети

Процесс передачи пакета данных от сетевого уровня передатчика (источника сообщения) до сетевого уровня приемника (получателя сообщения) показан на рисунке 4.11 [19].

Пакет данных от транспортного уровня передатчика поступает к сетевому уровню передатчика. На сетевом уровне передатчика определяется маршрут передачи пакета данных в соответствии с протоколом сетевого уровня.

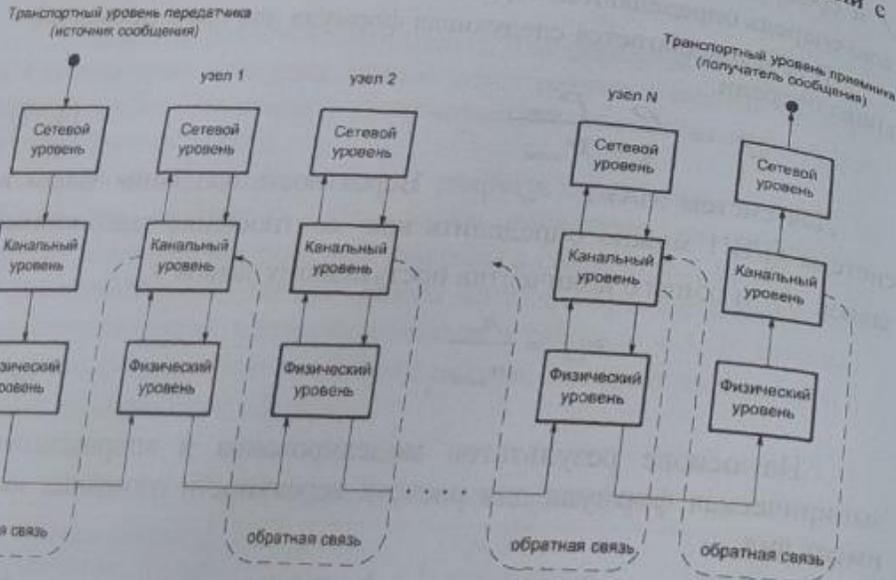


Рис. 4.11. Комплексная модель физического, канального и сетевого уровней магистральной сети

Далее пакет данных передается канальному уровню передатчика. На канальном уровне формируется кадр данных. В поле данных кадра вставляется пакет данных сетевого уровня. Основной задачей протокола канального уровня передатчика является надежная (достоверная) передача кадра данных до канального уровня первого узла связи (маршрутизатора,

коммутатора) в маршруте передачи. Процесс передачи кадра данных от канального уровня передатчика через физический уровень канального уровня канала связи, физический уровень первого узла до вероятности остаточного искажения и среднее количество повторов кадров данных на канальном уровне с учетом характеристик канала связи и методов помехоустойчивого кодирования, реализованного на физическом уровне.

Процесс обработки и передачи пакетов на сетевых уровнях узлов связи показан на рисунке 4.12.

При приеме пакета данных на сетевом уровне первого узла сети могут возникнуть следующие события:

- правильный прием пакета данных (ПП);
- обнаружение ошибки при обработке пакета данных (НО);
- обнаружение ошибки в пакете данных (ОО).

Вероятность ошибки каждого бита принятого пакета данных длиной n_c bit на сетевом уровне определяется по формуле:

$$P_c = \frac{P_{ош}}{n_c}, \quad (4.40)$$

где $P_{ош}$ - вероятность остаточного искажения кадра данных на канальном уровне сети.

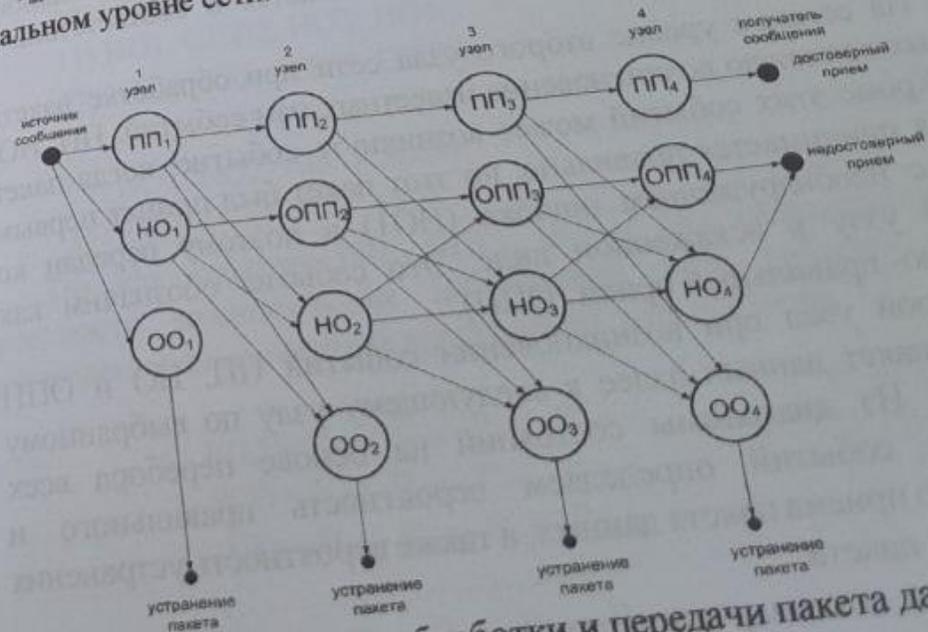


Рис. 4.26 - Процесс обработки и передачи пакета данных на сетевых уровнях узлов связи

Вероятность правильного приема пакета данных равна:

$$P_{\text{пр}} = (1 - p_e)^n.$$

Вероятность необнаруженной ошибки в пакете данных может быть оценена соотношением:

$$P_{\text{пр}} < P_e \cdot n / 2' \quad \text{при} \quad n \cdot p_e < 1, \quad (4.41)$$

где n - число контрольных разрядов в пакете данных.

Вероятность обнаружения ошибки в пакете определяется в виде:

$$P_{\text{об}} = 1 - (P_{\text{пр}} + P_{\text{оо}}). \quad (4.42)$$

Вероятности возникновения событий ПП, НО и ОО в первом узле обозначим соответственно $P_{\text{пп1}}$, $P_{\text{но1}}$ и $P_{\text{оо1}}$. При обнаружении ошибки ($P_{\text{оо1}}$) пакет данных устраняется (уничтожается) из сети. Следовательно, вероятность устранения (потери) пакета данных в первом узле связи $P_{\text{уе1}}$ равна вероятности обнаружения ошибки, т.е. $P_{\text{уе1}} = P_{\text{оо1}}$.

В случае возникновения событий ПП и НО пакет данных передается к следующему узлу в соответствии с выбранным маршрутом передачи. Пакет данных с сетевого уровня первого узла передается каналному уровню данного узла. Процесс передачи кадра данных от каналного уровня первого узла до каналного уровня второго узла связи также описывается в соответствии с рисунком 4.1.

На сетевом уровне второго узла сети при обработке пакета данных возможно возникновение известных нам событий: ПП, НО, ОО. Кроме этих событий может возникнуть событие, когда пакет данных принимается правильно, но этот пакет был принят первым узлом с необнаружением ошибки (НО1) и поэтому передан ко второму узлу в искаженном виде. Это событие обозначим как ошибочно-правильный прием (ОПП).

Второй узел при возникновении событий ПП, НО и ОПП передает пакет данных далее к следующему узлу по выбранному маршруту. Из диаграммы состояний на основе перебора всех возможных событий определяем вероятность правильного и искаженного приема пакета данных, а также вероятность устранения (недоставки) пакета.

Очевидно, что получатель принимает пакет данных правильно только в том случае, когда все узлы и получатель принимают пакет

данных правильно (ПП). Следовательно, вероятность правильного приема пакета данных получателем $P_{\text{ппп}}$ равна:

$$P_{\text{ппп}} = P_{\text{пр1}} \cdot P_{\text{пр2}} \cdot P_{\text{пр3}} \cdot \dots \cdot P_{\text{прN}} = \prod_{i=1}^N P_{\text{прi}}, \quad (4.44)$$

где N - общее количество каналов «точка-точка» в маршруте передачи пакета данных, т.е. канал связи между передатчиком и последним узлом, каналы связи между узлами, канал связи между транзитными узлами и получателем пакета данных (количество транзитных узлов равно $N-1$).

Искаженный прием пакетов в соответствии с диаграммой состояний возможен при следующих комбинациях событий:

- 1) ПП1, ПП2, ПП3, НО4;
- 2) ПП1, ПП2, НО3, НО4;
- 3) ПП1, ПП2, НО3, НО4;
- 4) ПП1, НО2, ОПП3, ОПП4;
- 5) ПП1, НО2, ОПП3, НО4;
- 6) ПП1, НО2, НО3, ОПП4;
- 7) ПП1, НО2, НО3, НО4;
- 8) НО1, ОПП3, ОПП4;
- 9) НО1, ОПП2, ОПП3, НО4;
- 10) НО1, ОПП2, НО3, ОПП4;
- 11) НО1, ОПП2, НО3, НО4;
- 12) НО1, НО2, ОПП3, ОПП4;
- 13) НО1, НО2, ОПП3, НО4;
- 14) НО1, НО2, НО3, ОПП4;
- 15) НО1, НО2, НО3, НО4.

Так как события ПП, ОПП, НО и ОО независимые, то вероятность возникновения комбинации этих событий равна произведению вероятностей наступления этих событий:

$$\begin{aligned} P(1) &= P_{\text{пп1}} \cdot P_{\text{пп2}} \cdot P_{\text{пп3}} \cdot P_{\text{но4}} \\ P(2) &= P_{\text{пп1}} \cdot P_{\text{пп2}} \cdot P_{\text{но3}} \cdot P_{\text{опп4}} \\ P(3) &= P_{\text{пп1}} \cdot P_{\text{пп2}} \cdot P_{\text{но3}} \cdot P_{\text{но4}} \\ &\dots \dots \dots \\ P(15) &= P_{\text{но1}} \cdot P_{\text{но2}} \cdot P_{\text{но3}} \cdot P_{\text{но4}} \end{aligned} \quad (4.45)$$

Вероятность искаженного приема пакета данных равна сумме вероятностей наступления этих комбинаций событий:

$$P_{\text{из}} = \sum_{i=1}^{13} P(i) \quad (4.46)$$

Выражение (4.46) с учетом (4.45) можно записать:

$$P_{\text{из}} = P_{\text{оо}} \cdot \sum_{k=1}^{N-1} P_{\text{пт}}^{k-1} \cdot (P_{\text{пт}} + P_{\text{но}})^{N-k} \quad (4.47)$$

Устранение пакета данных из сети возможно при следующих комбинациях событий:

- 1) ОО1;
- 2) ПП1, ОО2;
- 3) НО1, ОО2;
- 4) ПП1, ПП2, ОО3;
- 5) НО1, ОПП2, ОО3;
- 6) НО1, НО2, ОО3;
- 7) ПП1, ПП2, ПП3, ОО4;
- 8) НО1, ОПП2, ОПП3, ОО4;
- 9) НО1, ОПП2, НО3, ОО4;
- 10) НО1, НО2, ОПП3, ОО4;
- 11) НО1, НО2, НО3, ОО4;
- 12) ПП1, НО2, ОО3;
- 13) ПП1, ПП2, НО3, ОО4.

При количестве каналов «точка-точка» N вероятность устранения пакета равна:

$$P_{\text{yc}} = P_{\text{оо1}} + \sum_{k=2}^N P_{\text{оок}} \cdot \prod_{i=1}^{k-1} (P_{\text{пт}} + P_{\text{но}}) \quad (4.48)$$

Если вероятности наступления событий ПП, НО и ОО в каждом узле одинаковы, то выражение (4.48) можно записать в виде:

$$P_{\text{yc}} = P_{\text{оо}} \cdot \sum_{k=1}^N (P_{\text{пт}} + P_{\text{но}})^{k-1} \quad (4.49)$$

На рисунке 4.13 приведены графики зависимости вероятностей устранения, правильного и ошибочного приема пакета данных от количества транзитных узлов при $n_s = 4000$, $P_s = 10^{-4}$.

Из рисунка 4.13 видно, что с увеличением транзитных узлов связи в маршруте передачи вероятность правильного приема пакетов уменьшается, а вероятность устранения пакета увеличивается. Вероятность искаженного приема пакетов получателем с увеличением транзитных узлов также увеличивается.

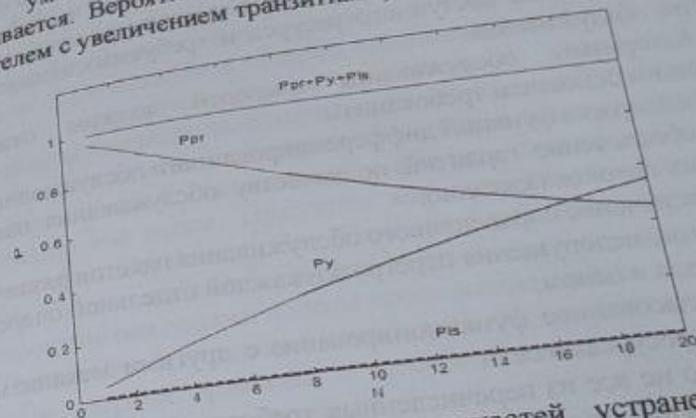


Рис. 4.13. Зависимость вероятностей устранения (P_y), правильного приема (P_{pr}) и искаженного приема (P_{is}) пакета данных от количества транзитных узлов

Из рисунка 4.27 также видно, что сумма вероятностей устранения, правильного и искаженного приема пакетов данных равна 1. Это подтверждает достоверность полученных результатов.

4.5. Оптимизация процессов обслуживания пакетов

Базовыми принципами иерархической системы являются принципы согласованности и координации, принимаемых на всех уровнях управления. В узлах сети телекоммуникации эти принципы заключаются, что все сетевые элементы управления буферным и каналным ресурсом должны быть согласованными между собой с целью достижения заданных показателей качества обслуживания (QoS) [21]. Характерной чертой существующих решений по управлению буферным (очередями) и каналным (пропускной способностью канала) ресурсом является статистическая стратегия

их распределения. Порядок распределения буферного пространства и канальной емкости по-прежнему преимущественно устанавливается административно, т.е. вне реального времени, хотя динамика изменения состояния отдельного сетевого узла выше чем динамика загрузки сети телекоммуникации. В связи с этим актуальной представляется задача придания динамического характера решениям по управлению канальным и буферным ресурсом в зависимости от характеристик поступающего на узел трафика, величины доступного ресурса и требуемых показателей качества обслуживания.

Алгоритмы обслуживания очередей должны отвечать следующим основным требованиям:

- поддержка функций дифференцированного обслуживания;
- обеспечение гарантий по качеству обслуживания пакетов различных потоков (классов);
- обеспечение справедливого обслуживания пакетов различных приоритетов, недопущения перегрузки каждой отдельной очереди и сетевого узла в целом;
- согласованное функционирование с другими механизмами управления ресурсами сети.

Однако не все из перечисленных требований в должной мере учтены в существующих алгоритмах обслуживания очередей, которые можно разделить на несколько классов [21]:

- без приоритетного обслуживания (FIFO – First In – First Out);
- приоритетного обслуживания (PQ – Priority Queuing, CQ – Custom Queuing);
- взвешенного обслуживания (FQ- Fair Queuing, WFQ – Weighted Fair Queuing);
- гибридные (LLQ – Low Latency Queuing, CBWFQ – Class Based Weighted Fair Queuing);

В алгоритме FIFO все поступившие пакеты ставятся в одну очередь и обслуживаются в порядке поступления. Поэтому алгоритм FIFO не обеспечивает дифференцированного качества обслуживания трафика.

Алгоритм приоритетного обслуживания трафика предусматривает разделение всего сетевого трафика на небольшое количество классов с назначением каждому классу некоторого числового признака — приоритета. Классификация пакетов может

производиться разными способами. Например, в пакете IP для этой цели предусмотрено трехразрядное подполе IP Precedence в поле Type Of Service (TOS). В узле сети имеется несколько очередей в соответствии с количеством классов. В начале обслуживания очередь с пакетами наибольшего приоритета. После обслуживания всех пакетов данной очереди, обслуживание получает очередь, приоритет у которой ниже и т.д. Из принципа работы приоритетного обслуживания очевидно, что постоянное наличие высокоприоритетного трафика в очереди приведет к существенным задержкам обслуживания и потерям низкоприоритетных трафиков. С целью исключения данного недостатка приоритетного обслуживания разработаны алгоритмы взвешенного обслуживания.

В алгоритмах взвешенного обслуживания очереди выделяется сетевой (канальный) ресурс, пропорциональный назначенным им весам. При этом все очереди гарантированно обслуживаются. Например в OpenFlow – коммутаторе по умолчанию имеются восемь очередей на физический порт с минимально гарантированными долями полосы пропускания канала.

Выделение постоянного канального ресурса при отсутствии пакетов в очереди приводит к неэффективному использованию пропускной способности канала.

В гибридном алгоритме обслуживания LLQ для трафика, чувствительного к задержкам, выделяется одна очередь, для обслуживания которой резервируется определенная пропускная способность канала. Остальные очереди обслуживаются в соответствии с алгоритмом взвешенного обслуживания. Поэтому гибридные алгоритмы также имеют недостаток алгоритмов взвешенного обслуживания.

Во всех вышерассмотренных алгоритмах решения задачи распределения пропускной способности между очередями имеет статистический характер, т.е. решается администратором путем настройки. Например, в алгоритме CQ распределение пропускной способности канала определяется значением задаваемого вручную счетчиком байт (*byte count*) для каждой очереди. В алгоритмах CBWFQ и LLQ порядок распределения пропускной способности задается также вручную командами *bandwidth* и *priority*.

Недостатки алгоритмов статистического обслуживания очередей определили актуальность разработки потоковых моделей (*flow-based model*) обслуживания очередей, в рамках которых

учитывается интенсивность трафика наряду с другими важными параметрами.

В потоковых моделях рассматривается следующая задача распределения трафика [22-26]. Имеются M классов трафиков, различаемых типом приоритета и N обслуживаемых очередей. Интенсивность трафика i -го класса равна d_i , ($i = \overline{1, M}$). Часть пропускной способности исходящего канала, закрепленная за j -й очередь равна c_j , ($j = \overline{1, N}$). Необходимо выполнить следующие условия:

$$\sum_{j=1}^N c_j \leq c, \quad (4.50)$$

$$\sum_{i=1}^M d_i \leq \sum_{j=1}^N c_j. \quad (4.51)$$

Динамический характер распределения пакетов трафика в очереди узла сети осуществляется за счет введения переменной x_{ij} , под которой подразумевается часть i -го трафика, который будет направлен для обслуживания в j -ю очередь. Для предотвращения перегрузки очередей вводятся условия:

$$\sum_{i=1}^M \sum_{j=1}^N x_{ij} < c_j, \quad (4.52)$$

$$\sum_{i=1}^M d_i x_{ij} < c_j. \quad (4.53)$$

Так как поток одного класса может быть обслужен только в рамках одной очереди, то искомая переменная x_{ij} принимает только два значения: 0 или 1, $x_{ij} = \{0, 1\}$.

Задача определения значений переменных x_{ij} формулируется в виде оптимизационной задачи с различными целевыми функциями.

В [22] минимизируется следующая целевая функция:

$$F(x) = \sum_{i=1}^M \sum_{j=1}^N f_{ij} x_{ij}, \quad (4.54)$$

где f_{ij} — характеризует относительную стоимость использования пакетами i -го трафика ресурсов j -й очереди.

В [23] в качестве искомой переменной выбирается вектор:

$$\vec{x} = \left[\frac{x_{ij}}{c_j} \right], (i = \overline{1, M}; j = \overline{1, N}). \quad (4.55)$$

Минимизируется следующая целевая функция:

$$F(x) = \vec{s} \vec{x}, \quad (4.56)$$

где координаты вектора весовых коэффициентов:

$$\vec{s} = \left[\frac{s_{ij}}{s_j} \right], \quad (4.57)$$

характеризуют условную стоимость (s_{ij}) использования пакетами i -го трафика ресурсов j -й очереди, а также стоимость (s_j) выделения j -й очереди того или иного объема пропускной способности исходящего канала передачи данных.

В [24] с целью сбалансированной загрузки буферных ресурсов дополнительно вводится следующее условие:

$$f(p_j) Q_j \leq \alpha, (j = \overline{1, N}), \quad (4.58)$$

где α — верхний динамически управляемый предел загруженности очередей узла сети.

Решается задача минимизации следующей целевой функции:

$$\min \alpha, \quad (4.59)$$

где управляемыми переменными являются x_{ij}, c_j и α .

В [25] используется целевая функция минимизации суммы средних длин очередей:

$$F = \sum_{j=1}^N f(p_j) Q_j, \quad (4.60)$$

где $f(p_j)$ — некоторая функция от характеристик пакетов j -ой очереди с приоритетом p_j .

Значение функции $f(p_i)$ должно быть тем больше, чем выше приоритет. При этом поток с более высоким приоритетом обслуживается лучше, чем поток с низким приоритетом.

В [26] условия (4.52) и (4.53) дополняются условиями предотвращения перегрузки очередей по их длине:

$$Q_i \leq Q_i^{\max},$$

где Q_i^{\max} – максимальная емкость очереди.

Решается оптимизационная задача, связанная с минимизацией целевой функции вида:

$$F(x) = \sum_{i=1}^M \sum_{j=1}^K f_{ij} x_{ij} + \alpha.$$

Целевая функция (4.66) включает в себя функции (4.58) и (4.63).

Оптимизационные задачи (4.54) и (4.56) относятся к задачам линейного программирования, а задачи (4.59), (4.60) и (4.62) – нелинейного программирования. В задачах (4.54) и (4.56) результаты оптимизации существенно зависят от метрик f_v и \bar{z} . Чем меньше метрика f_v и \bar{z} , тем больше данная очередь загружается. Минимизация целевых функций (4.59), (4.60) и (4.62) обеспечивает сбалансированную загрузку очередей узла сети.

Общими недостатками рассмотренных потоковых методов обслуживания очередей являются:

- объем буферных ресурсов задается и не распределяется между очередями;

- на практике каждому трафику выделяется своя очередь и нет необходимости распределения трафика между очередями.

Для устранения вышеуказанных недостатков предложен метод динамического обслуживания очередей. Каждому трафику i – го приоритета выделяется j – я очередь, при этом $i = \overline{1, M}, j = \overline{1, N}$. Так как $M = N$, далее будем использовать N . Общий буферный ресурс с объемом L распределяется между очередями в зависимости от интенсивностей трафиков. Определяется суммарная интенсивность трафиков

$$D = \sum_{i=1}^N d_i \quad (4.63)$$

и доля i – го трафика в общем потоке

$$\gamma_i = \frac{d_i}{D}, i = \overline{1, N}. \quad (4.64)$$

Объем буфера, выделяемого для i – й очереди, прямо пропорционален доле i – го трафика

$$l_i = \gamma_i L, i = \overline{1, N}. \quad (4.65)$$

Управляемой переменной служит x_i , которая характеризует долю пропускной способности канала выделяемого для обслуживания i – й очереди. При этом необходимо выполнить условия предотвращения перегрузки очередей:

$$d_i = x_i c, i = \overline{1, N},$$

$$\sum_{i=1}^N x_i = 1, 0 \leq x_i \leq 1. \quad (4.67)$$

Вводится коэффициент важности трафика i – приоритета - v_i ($v_i \geq 0$).

Минимизируется сумма длин всех очередей с учетом коэффициентов важности трафика:

$$\min \sum_{i=1}^N v_i Q_i. \quad (4.68)$$

Если каждая очередь представляет собой экспоненциальную систему массового обслуживания с ограниченной очередью, то вероятность потери пакетов из-за переполнения буфера определяется выражением:

$$p_i = \frac{1 - \rho_i}{1 - \rho_i^{l_i+1}} \rho_i^{l_i+1}, \quad (4.69)$$

где ρ_i – загрузка i – й очереди ($i = \overline{1, N}$):

$$\rho_i = \frac{d_i}{x_i c}. \quad (4.70)$$

Средняя длина очереди определяется по формуле:

$$Q_i = \frac{\rho_i^2}{(1 - \rho_i)^2} p_{0i} \{ 1 - \rho_i^2 [l_i (1 - \rho_i) + 1] \}, \quad (4.71)$$

где p_{0i} – вероятность отсутствия пакетов в i – й очереди.

$$p_{0i} = \frac{1 - \rho_i}{1 - \rho_i^{L+1}}$$

Среднее время ожидания (W_i) и задержка (T_i) пакетов: (4.72)

$$W_i = \frac{Q_i}{d_i}, i = \overline{1, N}$$

$$T_i = W_i + 1/x_i c, i = \overline{1, N} \quad (4.73)$$

На рисунке 4.14 приведен график зависимости средней задержки пакетов от интенсивности трафика каждого приоритета при коэффициентах важности трафика $\nu = [18, 14, 11, 8, 5, 4, 2, 1]$, $N = 8$, и $L = 80$ (4.74)

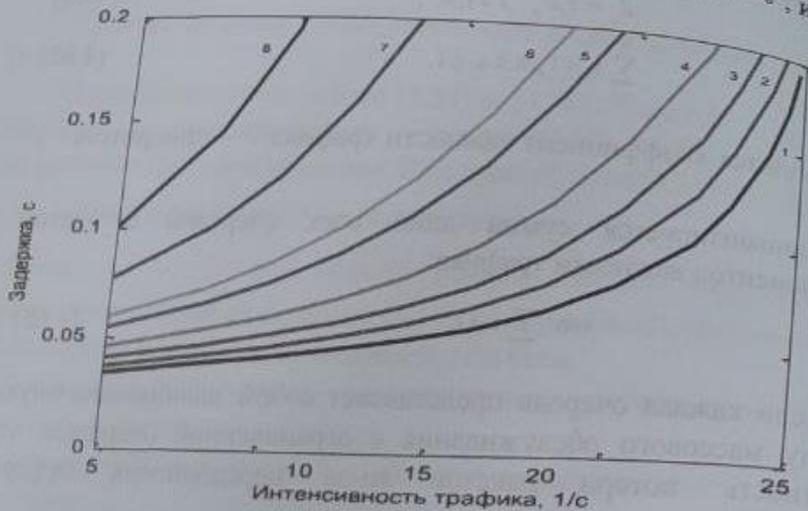


Рис. 4.14. Зависимость задержки пакетов от интенсивности трафика каждого приоритета

Наименьшую задержку имеют пакеты трафика высокого приоритета. С уменьшением приоритета увеличивается средняя задержка пакета.

На рисунке 4.15 приведен график зависимости вероятности потери пакетов от интенсивности трафика каждого приоритета. Из рисунка 4.15 следует, трафик с более высоким приоритетом по вероятности потерь также имеет преимущество по сравнению с низкими приоритетами.



Рис. 4.15. График зависимости вероятности потери пакетов от интенсивности поступления трафика каждого приоритета

Как указано выше, алгоритмы обслуживания очередей должны обеспечить гарантированное обслуживание очередей с низкими требованиями. Далее с целью проверки выполнения данного алгоритма обслуживания проводится сравнительный анализ предложенного приоритетного обслуживания с алгоритмом относительного обслуживания.

Для определения характеристик обслуживания трафика с относительным приоритетом будем использовать формулы, приведенные в работе [27].

Вероятность потери пакетов i -го приоритета:

$$p_i = P(N) \frac{1 - \rho_i}{1 - \rho_i^{L+1}} \rho_i^i, \quad (4.75)$$

где

$$\rho_i = \sum_{j=1}^i d_j / \mu_j, \mu_j = c_j, j = \overline{1, N}, \quad (4.76)$$

$$P(N) = \frac{\rho_N - \rho_N^{L+1}}{1 - \rho_N^{L+1}}. \quad (4.77)$$

Среднее время ожидания обслуживания пакетов i -го приоритета:

$$W_i = \begin{cases} W_i^*, i = 1; \\ \frac{\wedge_i}{d_i} (W_i^* - \frac{\wedge_{i-1}}{\wedge_i} W_{i-1}^*), i = \overline{2, N}, \end{cases} \quad (4.78)$$

где:

$$\lambda_i = \sum_{j=1}^L d_j,$$

$$W_i^* = P(N) \frac{1 - (L+1)\rho_i^L + L\rho_i^{L+1}}{\mu_i(1-\rho_i)(1-\rho_i^{L+1})}.$$

Среднее время задержки пакетов i -го приоритета:

$$T_i = W_i^* + 1/\mu_i.$$

Для сравнительного анализа предложенного метода обслуживания и приоритетного обслуживания приведен график зависимости среднего времени задержки пакетов от интенсивности трафика первого приоритета при заданных интенсивностях других видов трафика d_j ($d_j = 5, j = 2, N$).

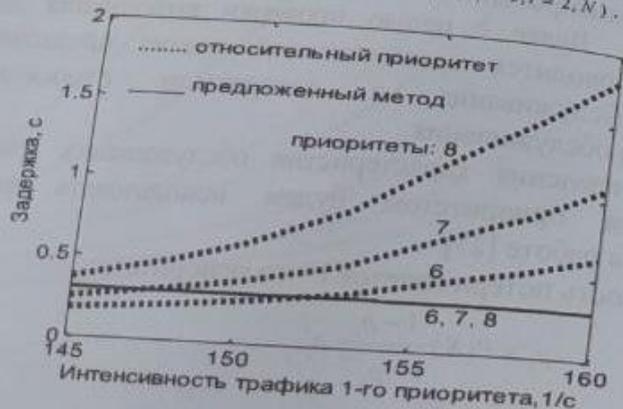


Рис.4.16. График зависимости средней задержки пакетов от интенсивности трафика первого приоритета при заданных интенсивностях других видов трафика

Из рисунка 4.16 следует, что при алгоритме относительного приоритетного обслуживания при высоких интенсивностях трафика высокого приоритета, средняя задержка пакетов низких приоритетов резко возрастает, а в предложенном алгоритме обслуживания средняя задержка пакетов низкого приоритета почти не изменяется. При этом, метод динамического обслуживания уменьшает среднюю задержку низкоприоритетных пакетов до 3-х раз по сравнению с алгоритмом относительного приоритетного обслуживания.

4.6. Оптимизация процессов маршрутизации пакетов

На сегодняшний день многие ученые в области телекоммуникаций говорят о недостатках традиционных TCP/IP-сетей, которые сводятся к главному – невозможности гибко управлять телекоммуникационной сетью (ТКС), что приводит к ухудшению показателей качества обслуживания (Quality of Service, QoS). Большая часть этих недостатков минимизируется при реализации идей, заложенных в концепцию программно-конфигурируемых сетей (Software Defined Networking, SDN), которая сегодня активно развивается и, по мнению своих разработчиков, должна эффективно дополнить и модернизировать многие существующие сетевые технологии. Главная идея SDN состоит в отделении управляющего уровня (control plane) от уровня передачи данных (forwarding data plane), что предполагает передачу ряда основных управляющих функций от операционных систем (ОС) узлов (маршрутизаторов и коммутаторов) ТКС к сетевой ОС. При этом сетевая ОС с помощью определенных протоколов, основываясь на работе специальных облачных серверов, решает задачи управления трафиком, в том числе задачи маршрутизации, что повышает уровень централизации управления в сети. В связи с этим развитие и внедрение SDN-решений требует усовершенствования протоколов маршрутизации, а также их адаптацию под новые условия.

Анализ известных протоколов маршрутизации в ТКС показал, что они основаны преимущественно на графовых моделях, в рамках которых задача поиска кратчайшего пути решается при помощи алгоритмов Дейкстры, Беллмана-Форда и др. Однако существующие на сегодняшний день протоколы не укладываются в полной мере в рамки концепции QoS-маршрутизации, поскольку в них заложена идея поиска кратчайшего пути в одной, пусть даже композитной, метрике без учета достигаемых при этом значений других метрик. Так, например, несмотря на то, что в рамках EIGRP при выборе маршрута может учитываться множество показателей качества обслуживания (задержка, загрузка, надежность, пропускная способность), использование комбинированной метрики (1) вовсе не гарантирует наилучших значений, например, задержки, вдоль найденного маршрута. Более того, задержка, используемая в EIGRP, является административно назначаемым при настройке

интерфейса параметром и в общем случае может не соответствовать той величине задержки, которая на самом деле имеет место в данном интерфейсе. Другой проблемой, связанной с применением существующих протоколов маршрутизации, является несбалансированное использование сетевых ресурсов. В основе всех перечисленных протоколов маршрутизации положены алгоритмы нахождения кратчайшего пути в графе (Дijkstra или алгоритм Форда), результатом работы которых является один, кратчайший выбранной метрике, путь. Поскольку ни в одном из упомянутых выше протоколов в метрику не входит упомянутая (не задействованная) пропускная способность трактов передачи (не для заданной пары адресатов в качестве кратчайшего будет выбираться один и тот же путь, что в конечном итоге приводит к его перегрузке даже при наличии свободных обходных маршрутов. Исключение составляет алгоритм Constrained Shortest Path First (CSPF), применяемый в сетях MPLS TE для расчета пути коммутации меток (Label Switching Path, LSP) и учитывающий при этом текущую загруженность трактов передачи и атрибуты нового LSP. Целесообразным решением в данной ситуации является multipутевая маршрутизация с балансировкой нагрузки не только между путями равной стоимости, что заложено во всех протоколах маршрутизации, но и между путями неравной стоимости, что присутствует только в протоколах IGRP, EIGRP и внутримоменной версии BGP – iBGP (interior BGP). Как правило, балансировка между маршрутами с различной стоимостью в рамках данных протоколов требует от администраторов сети дополнительных настроек и зачастую ими не используется. Вычисление маршрута в рассмотренных выше протоколах маршрутизации как внутренних, так и внешних (межсетевых или междоменных), реализуется распределенно и в момент поступления трафика в сеть, т.е. по требованию.

При переходе к QoS-маршрутизации заслуживают внимания следующие маршрутные концепции, во-первых, концепция централизованного вычисления путей в рамках маршрутизации от источника (Source routing) – концепция сервера маршрутов (Route Server, RS), во-вторых, концепция предвычисления путей (Precomputed routing, PR). При решении задач внутренней (в пределах одного домена или сети одного провайдера) маршрутизации функции сервера маршрутов в зависимости от

сетевой архитектуры возлагаются на различные устройства: элемент вычисления путей PCE (Path Computation Element) в домене MPLS; брокер пропускной способности BB (Bandwidth Broker) в DiffServ-домене IP-сети или иное специализированное устройство, например, сервер маршрутизации Routing and Traffic Engineering Server (RATES) для сетей MPLS или управляющая маршрутами BGP платформа Routing Control Platform (RCP). Наличие единого центра управления в сети (домене) позволяет не только решить задачу управления в сети (домене) позволяет не только решить текущее состояние, но и обеспечивает решение задач управления доступом и резервирования ресурсов, что немаловажно для обеспечения гарантированного QoS в целом. Масштабируемость в условиях централизованного принятия решения сервером RS обеспечивается за счет предвычисления пути коммутации меток LSP. Предварительное установление пути коммутации меток LSP Много рациональных идей было заложено в протокол маршрутизации PNNI (Private Network-to-Network Interface), используемый в прежде достаточно перспективной технологии ATM (Asynchronous Transfer Mode). Данный протокол реализует стратегию маршрутизации «от источника», поддерживая композитную метрику по ключевым показателям качества обслуживания, к которым, прежде всего, относятся доступная скорость передачи ячеек (Available Cell Rate, AvCR); максимальная задержка передачи ячеек (Maximum Cell Transfer Delay, MaxCTD); процент потерь (Cell Loss Ratio, CLR) – среднее количество потерянных во время передачи ячеек; разброс задержки (Cell Delay Variation, CDV); максимальная скорость передачи (Maximum Cell Rate, MaxCR). Однако, наряду с традиционными достоинствами с точки зрения поддержки функциональности качества обслуживания, данный маршрутизирующий протокол является вдобавок и сигнальным протоколом, отвечающим за установление соединения и обеспечение QoS-гарантий на основе резервирования ресурсов. Интеграция в рамках одного протокола функций маршрутизации и сигнализации заметно повышает его возможности с точки зрения обеспечения качества обслуживания. Напомним, что в IP-сетях задачи резервирования ресурсов несколько отделены от протоколов маршрутизации и возложены на специальный протокол RSVP (Resource ReSerVation Protocol). В области внешней (межсетевой или междоменной) маршрутизации можно выделить два направлени

развития. Первое связано с использованием протокола пограничного шлюза BGP и его QoS-расширения. Как известно, BGP является протоколом маршрутизации дистанционно-векторного типа, в котором решение о направлении продвижения пакета принимается каждым узлом в отдельности (подход hop-by-hop). Отсутствие на маршрутизаторе информации о текущем состоянии всей сети не позволяет в условиях применения BGP получить в конечном итоге маршрут, оптимальный с точки зрения всей сети. Другой подход к межсетевой маршрутизации заключается в использовании маршрутизации от источника []. Хотя маршрутизация от источника имеет свои сложности, связанные, например, с необходимостью сбора подробной информации о топологии сети, именно она потенциально способна обеспечить расчет такого маршрута, вдоль которого QoS-требования гарантировано выполняются. Масштабируемость в этом случае обеспечивается за счет иерархического представления топологической информации. Таким образом, хотя маршрутизация потенциально является одним из наиболее действенных механизмов в плане предоставления качества обслуживания, на практике в рамках существующих протоколов маршрутизации подобные функции остаются нереализованными в полной мере. В связи с переходом к сетям NGN вопросы совершенствования алгоритмов и протоколов являются особо актуальными.

В целом на основании выявленных недостатков существующих протоколов и с учетом перспективных концепций можно сформулировать требования к протоколу маршрутизации в современных сетях. Во-первых, протокол маршрутизации должен отвечать концепции QoS-маршрутизации, т.е. оперировать не только номинальными, но и доступными сетевыми ресурсами, информация о которых должна обеспечиваться путем их постоянного мониторинга, а также обеспечивать маршрут не просто минимальной стоимости, а такой, вдоль которого гарантированно выполняются требования приложений к качеству их обслуживания как по показателям сетевой производительности, так и (в идеале) по показателям воспринимаемого качества обслуживания – QoE (Quality of experience). Во-вторых, протокол маршрутизации должен обеспечивать сбалансированное использование сетевых ресурсов (Load-Balancing Routing), что требует перехода от однопутевых к многопутевым стратегиям. В третьих, задача маршрутизации

должна решаться согласованно (в комплексе) с задачами управления доступом и резервирования ресурсов. Все перечисленное в совокупности нацеливает на реализацию в рамках перспективных протоколов маршрутизации концепции сервера маршрутов в сочетании с концепцией предвычисления путей и иерархическим представлением топологической информации как средств обеспечения масштабируемости, что в целом требует пересмотра теоретических решений, закладываемых в их основу, и ориентирует на поиск новых, более конструктивных методов управления трафиком в современных мультисервисных ТКС.

Анализ комбинаторных решений, представляющих собой задачу QoS-маршрутизации. Представление прикладной задачи как комбинаторной задачи и решение ее путем направленного перебора является одним из распространенных подходов в процессе анализа и синтеза ТКС. Так, как уже было отмечено, основу существующих протоколов маршрутизации составляют различные алгоритмы кратчайшего пути в графе. Прежде всего, это алгоритмы Дейкстры и Беллмана-Форда, которые обеспечивают нахождение дерева кратчайших в выбранной метрике путей от узла источника ко всем остальным узлам, и вычислительная сложность которых приемлема для реализации в реальном масштабе времени. С появлением концепции NGN и смещением акцентов при оценке алгоритмов управления трафиком на их возможности по поддержке функций QoS графокомбинаторные модели и методы были существенно пересмотрены. В целом описанные в литературе подходы могут быть разделены на два класса: модели и методы QoS-маршрутизации и модели и методы k-путевой маршрутизации, которые, за исключением единичных предложений, почти не пересекаются. Концепция QoS-маршрутизации требует определения такого пути (путей), между заданной парой узлов-адресатов, вдоль которого будут выполняться требования одновременно по нескольким QoS-показателям (метрикам). В рамках подобной маршрутизации требования к QoS-показателям вдоль пути выступают в качестве ограничений на этапе его поиска, задача которого в общем случае может быть сформулирована двояко: как задача поиска пути с ограничениями (Multi-Constrained Path, MCP) или как задача поиска оптимального пути с ограничениями (Multi-Constrained Optimal Path, MCOP). И в том и в другом случае использование одновременно двух и более метрик независимо от их типа (аддитивных,

мультипликативных) ведет к тому, что задача нахождения пути становится NP-полной и может быть решена путем полного перебора. В этой связи в литературе предложено множество эвристических алгоритмов, обладающих полиномиальной сложностью задач MCR и MCOR можно указать три основных подхода к которым соответствует одному QoS показателю, каждая из которых соответствует одному QoS показателю, к одной композитной (комбинированной), представляющей собой некоторую функцию отдельных метрик. 2. Поочередное использование метрик, т.е. поиск всех путей, удовлетворяющих требованиям в первой метрике, затем поиск среди найденных, уже с использованием второй метрики и т.д. 3. Замена всего множества метрик одной, той, которая лежит в основе каждой из отдельных метрик и определяет их значения.

Наличие единственной метрики позволяет применить известные алгоритмы поиска кратчайшего пути и их всевозможные расширения. В целом задача конструирования метрик как одиночных, так и комбинированных представляет собой отдельный предмет исследований. Примером второго подхода является алгоритм решения задачи поиска пути, удовлетворяющего требованиям по пропускной способности и по задержке. Алгоритм предусматривает два этапа: на первом этапе из графа удаляются все ветви, пропускные способности которых не отвечают выдвинутым требованиям, а на втором этапе в графе, полученном после удаления части ветвей, при помощи алгоритма Дейкстры ищется путь с минимальной задержкой. Другим примером является алгоритм Delay-Cost-Constrained Routing (DCCR), в котором на первом этапе в исходном графе генерируется множество путей, удовлетворяющих требованиям по задержке, а на втором этапе среди этого множества, начиная с пути с минимальной задержкой, выбирается один, удовлетворяющий еще и требованию по стоимости. К этой же группе относятся часто встречающиеся в литературе алгоритмы Widest Shortest Path и Shortest Widest Path, где первый обеспечивает поиск пути с наибольшей пропускной способностью среди множества путей с минимальной стоимостью, а второй – поиск пути с наименьшей стоимостью среди множества путей с максимальной пропускной способностью. Третий подход для случая

использования механизма обслуживания очередей WFQ такие метрики как скорость передачи, межконцевая задержка, джиттер и используемый объем буферной памяти не являются независимыми. Все они представляют собой функции от зарезервированной пропускной способности, выбранного пути и характеристик трафика, а потому исходная задача нахождения пути с множеством ограничений на основании установленной взаимосвязи может быть решена при помощи модифицированного алгоритма Беллмана-Форда. Задача MCR может быть решена за полиномиальное время, если $(r-1)$ из r метрик будут представлены целями, ограниченными сверху числами, для чего предлагается сначала аппроксимировать метрики, являющихся на самом деле действительными числами, а затем применить расширенные алгоритмы Дейкстры и Беллмана-Форда (Extended Dijkstra shortest path, EDSP, Extended Bellman-Ford, EBF). Кроме того графовые модели по своей сути ориентированы на однопродуктовые двухполосные сети и их расширение на случай множества продуктов и пар адресатов (многопродуктовые многополосные сети) приводит к сложности предоставления услуг QoS вычислительного характера. В рамках подобных моделей остается нерешенным один из важнейших для предоставления услуг вопрос распределения ресурсов вдоль найденного пути, что является необходимым условием для достижения согласованного решения задач маршрутизации, резервирования ресурсов и управления доступом. Успешное их решение невозможно без учета наряду со структурными характеристиками сети параметров информационного трафика. Таким образом, несмотря на использование моделей и методов кратчайшего пути в рамках существующих протоколов управления трафиком, данный подход в свете перехода к мультисервисным сетям NGN с предоставлением широкого спектра услуг гарантированного качества не является перспективным и может рассматриваться как промежуточный этап на пути перехода к более сложным, но более адекватным потоковым моделям ТКС.

Потоковые модели маршрутизации

На сегодняшний день наиболее перспективными являются потоковые модели маршрутизации, в большинстве из ко-

используется критерий, базирующийся на минимизации максимального использования каналов ТКС. В рамках моделей маршрутизации по коэффициенту максимальной загрузки каналов ТКС коэффициент балансировки с ростом загрузки каналов растет линейно, это гарантирует отсутствие колебаний в численных значениях основных показателей качества обслуживания. Однако в результате исследования подобных моделей выявлено условие, при которых балансировка нагрузки по критерию максимального использования каналов связи не всегда (не для всех структур и типов каналов связи) позволяет максимально улучшить значения показателей качества обслуживания. Поэтому предлагается изменить критерий при решении задачи маршрутизации с целью повышения значений показателей качества обслуживания. Как известно, рост интенсивности случайного и нестационарного по своей природе сетевого трафика вызывает образование очередей на узлах ТКС. Именно в очереди пакеты испытывают максимальные задержки, а при их переполнении возникают потери пакетов. В этой связи при разработке модели маршрутизации с балансировкой нагрузки в качестве критерия предлагается использовать минимум длин очередей на маршрутизаторах ТКС.

Основные задачи оптимизации в потоковой маршрутизации

Анализ потоковых решений задач QoS-маршрутизации в рамках потоковых моделей задача управления трафиком, в т.ч. маршрутизации, зачастую формулируется в виде задачи математического программирования: линейного, нелинейного, целочисленного, смешанного с использованием терминологии распределения потока на графах. Обязательными компонентами такой постановки задачи являются целевая функция, которая в зависимости от вкладываемого физического смысла подлежит минимизации или максимизации; условия сохранения потока; условия, связанные с ограниченностью сетевых ресурсов, прежде всего пропускной способности трактов передачи (условия отсутствия перегрузки) и условия неотрицательности потока. В зависимости от уровня рассмотрения потока и физического смысла, вкладываемого в используемые переменные, каждое из перечисленных условий конкретизируется.

С целью недопущения перегрузки маршрутизаторов и сети в целом необходимо обеспечить выполнения условий сохранения потока [28-32]:

$$\begin{cases} \sum_{j:(j,i) \in E} x_{ij}^k - \sum_{j:(j,i) \in E} x_{ji}^k = 1, k \in K, i = s_k \\ \sum_{j:(j,i) \in E} x_{ij}^k - \sum_{j:(j,i) \in E} x_{ji}^k = 0, k \in K, i \neq s_k, t_k \\ \sum_{j:(i,j) \in E} x_{ij}^k - \sum_{j:(i,j) \in E} x_{ji}^k = -1, k \in K, i = t_k. \end{cases} \quad (4.82)$$

Кроме этого, важно добиться выполнения условия предотвращения перегрузки каналов сети:

$$\sum_{k \in K} d_k x_{i,j}^k \leq c_{i,j}. \quad (4.83)$$

Чтобы в сетях с полудуплексными и дуплексными каналами передачи не возникали закливания пакетов, необходимо обеспечить выполнения условий

$$x_{ij}^k \cdot x_{ji}^k = 0, (i,j) \in E, k \in K. \quad (4.84)$$

При однопутевой маршрутизации на управляющие переменные $x_{i,j}^k$ накладываются ограничения вида:

$$x_{i,j}^k \in \{0, 1\}, \quad (4.85)$$

а при многопутевой маршрутизации:

$$0 \leq x_{i,j}^k \leq 1. \quad (4.86)$$

Эффективность методов маршрутизации зависит от выбранного критерия оптимизации. Основные модели маршрутизации с различными критериями оптимизации.

1. Минимизация целевой (стоимостной) функции, представленную линейной формой:

$$\min \sum_{(i,j)} f_{i,j} x_{i,j}, \quad (4.87)$$

где каждому каналу передачи присваивается метрика, используем в протоколе IGRP:

$$f_{i,j} = 10^7 / c_{i,j}. \quad (4.88)$$

2. Минимизация максимального порога использования пропускной способности канала передачи:

$$\min \alpha,$$

при соблюдении условий

$$\sum_{k \in K} d_k x_{i,j}^k \leq \alpha c_{i,j}, \quad 0 \leq \alpha \leq 1. \quad (4.89)$$

3. Минимизация средней задержки потока в сети:

$$\min \left(\frac{1}{d_k} \sum_{(i,j) \in E} \frac{d_k x_{i,j}^k}{c_{i,j} - d_k x_{i,j}^k} \right). \quad (4.90)$$

4. Минимизация суммарной длины очереди в маршрутизаторах сети:

$$\min \sum_{i \in V} \sum_{j \in V} Q_{ij}, \quad j \neq i. \quad (4.91)$$

где Q_{ij} - длина очереди в i -м маршрутизаторе на исходящем интерфейсе к j -му маршрутизатору.

5. Минимизация максимальной длины очереди в маршрутизаторах сети:

$$\min (\max \sum_{i \in V} \sum_{j \in V} Q_{ij}), \quad j \neq i. \quad (4.92)$$

Маршрутизаторы сети представлены как системы массового обслуживания M/M/1. При этом средняя длина очереди и средняя задержка определяются следующими выражениями:

$$Q_{ij} = \frac{\rho_{ij}^2}{1 - \rho_{ij}}, \quad (4.93)$$

$$\bar{t}_{ij} = \frac{1}{c_{ij} - \rho_{ij}}, \quad (4.94)$$

где ρ_{ij} - интенсивность потока, передаваемого по каналу $(i, j) \in E$; $\rho_{ij} = \phi_{ij} / c_{ij}$ - загрузка канала $(i, j) \in E$.

Для оценки эффективности рассмотренных методов маршрутизации будем использовать среднюю многопутевую задержку. Задержка $t(m_l)$ вдоль пути из множества путей $M = \{m_1, m_2, \dots, m_L\}$, где L - количество путей между маршрутизатором-отправителем и маршрутизатором-получателем, определяется по формуле:

$$t(m_l) = \sum_{(i,j) \in m_l} \bar{t}_{ij}, \quad (4.95)$$

а межконцевая многопутевая задержка:

$$T(M) = \max\{t(m_l)\}, \quad m_l \in M. \quad (4.98)$$

На рисунке 4.17 приведены графики зависимости задержки потока от интенсивности поступающего в сеть трафика при однопутевой и многопутевой маршрутизации соответственно.



Рис. 4.17. Сравнение методов маршрутизации

Однопутевая маршрутизация с пятью различными критериями оптимизации обеспечивает почти одинаковую задержку потока. Многопутевая маршрутизация с минимизацией стоимости (4.87) и средней задержки (4.92) потока имеет наибольшую многопутевую задержку, а с минимизацией максимального порога загрузки каналов (4.89), суммарной длины очереди (4.93) и максимальной длины очереди (4.94) до высокой загрузки сети обеспечивают одинаковую многопутевую задержку, а в областях высокой загрузки наиболее эффективной является маршрутизация с минимизацией максимальной длины очереди.

4.7. Оптимальное встраивание виртуальных сетей

Основной задачей при осуществлении виртуализации является выделение ресурсов физической сети для той или иной виртуальной сети. Встраивание виртуальной сети (Virtual Network Embedding) происходит через динамическое сопоставление виртуальных ресурсов с физическим оборудованием, за счет чего получают преимущество использования существующего оборудования. При этом в сетях будущего (Future Networks) в це

обеспечения гарантированных услуг для конечного пользователя, необходимо оптимальное динамическое распределение ресурсов [33,34], позволяющее самоконфигурироваться и самоорганизацию сетей в отношении различных целевых задач, к примеру, качество обслуживания (QoS), экономическая эффективность, живучесть, энергоэффективность, безопасность сетей.

Истраивание виртуальной сети связано с распределением виртуальных ресурсов как в узлах, так и в каналах. Другими словами, могут быть рассмотрены две подзадачи: отображение виртуального узла (*Virtual Node Mapping VNoM*), когда виртуальные узлы должны быть распределены в физических узлах, и отображение виртуальных каналов (*Virtual Link Mapping VLiM*), когда виртуальные каналы, соединяющие виртуальные узлы, должны быть сопоставлены с путями, соединяющими соответствующие узлы в физической сети.

Применение механизмов виртуализации к сетевым ресурсам приводит к необходимости решения вопроса о том, как виртуализированные ресурсы должны быть реализованы ресурсами физической сети. Важно отметить, что физические ресурсы сами по себе могут быть виртуальными – это так называемая вложенная виртуализация (*Nested virtualization*). В таком случае, только самый низкий уровень должен состоять из физических ресурсов.

Аппаратная абстракция, предоставляемая решением виртуализации, обеспечивает общую платформу, позволяющую любому физическому ресурсу размещать виртуальные ресурсы того же типа. Как правило, физический ресурс разбит на несколько узлов для виртуальных ресурсов. Например, виртуальный узел может, в принципе, быть размещен на любом доступном узле физической сети. Кроме того, один и тот же физический узел может размещать несколько виртуальных узлов. Таким образом, отображение виртуальных узлов на физические узлы описывается как отношение $1:1$ (строгое разделение физических ресурсов).

В некоторых случаях физические ресурсы могут быть объединены с целью создания новых виртуальных ресурсов: это относится к виртуальному каналу, который охватывает несколько узлов в физической сети. В этом случае виртуальный канал между двумя виртуальными узлами v и w отображается в физический тракт,

который соединяет физические узлы v и w . Каждый физический канал может быть частью нескольких виртуальных каналов. Таким образом, отображение виртуальных каналов в физическом тракте описывается отношением $n:m$ (как для случая разделения, так для комбинирования физических ресурсов).

Как правило, существуют некоторые ограничения, которые следует учитывать во время отображения. Очевидно, что потенциальные физические ресурсы для отображения должны иметь возможность поддерживать требования производительности виртуальных ресурсов. Например, виртуальный канал 1000 Мбит/с не может быть отображен на тракте, содержащем физический канал на 100 Мбит/с. Аналогично, процессорная мощность (или запрашиваемая виртуальным узлом, должна быть меньше (или равна) мощности процессора, фактически обеспечиваемого физическим узлом. Если требуется избыточность, возможно, придется зарезервировать еще больше физических ресурсов.

В связи с тем, что физические ресурсы следует экономить, отображение должно быть оптимизировано. Эта проблема оптимального отображения виртуальных ресурсов на физические ресурсы широко известна как проблема встраивания (внедрения) виртуальной сети. Обычно она моделируется путем описания запроса виртуальной сети (*Virtual Network Request VNR*) по отношению к требованиям узлов и каналов, в то время, как физическая сеть описывается узловыми и канальными ресурсами. Для того, чтобы осуществить встраивание, необходимо, чтобы требования и ресурсы были сопоставлены: это означает, что виртуальные ресурсы сначала сопоставляются с ресурсами-кандидатами физической сети.

Только если все виртуальные ресурсы имеют возможность отображения, сеть *встраивается*, и физические ресурсы фактически расходуются. Если запросы VNR поступают по одному, может потребоваться реконфигурация, возвращая предыдущее встраивание и вычисляя новое отображение.

В целом, основная задача встраивания VNE состоит в нахождении оптимальных функций f_i и g_i для узлового (VNoM) канального (VLiM) решения в отношении конкретной задачи. В этом цели, которые преследуются подходами к решению V, можно описать нижеприведенными факторами.

1). Обеспечить QoS-совместимое встраивание: Запросы в виртуальной сети устанавливаются и обслуживаются VNO в соответствии с набором ограничений качества обслуживания, определенных поставщиком услуг. Существует несколько ситуаций, когда эти требования проявляются в запросе. Например, виртуальная сеть, которая обеспечивает услуги VoIP, должна рассчитывать на среднюю пропускную способность, низкие задержки и высокие требования к процессору. Другим примером может быть виртуальная сеть, предлагающая сервисы P2P, которые должны обеспечивать требования к средней пропускной способности, отсутствие соответствующих ограничений по задержкам и требования к средней производительности процессора.

2). Максимизировать экономическую прибыль InP: с точки зрения InP, естественной целью алгоритма встраивания в реальном масштабе времени является максимизация экономического дохода при встраивании при запросе VNR (долгосрочный средний доход). Эта цель прямо пропорциональна максимизации числа встроженных VNR (коэффициент встраивания). Чтобы достичь этой цели, подходы VNE должны пытаться минимизировать ресурсы, потраченные SN для отображения VNR, также известного как стоимость внедрения.

3). Обеспечить живучесть VNE: Устойчивость с точки зрения VNE может быть обеспечена путем интеграции резервных ресурсов в физической сети. Резервные узлы/каналы могут быть настроены либо для всех, либо только для некоторых определенных первичных узлов или каналов, которые могут выйти из строя.

В любой момент времени должна быть гарантирована согласованность сетевой топологии, особенно в отношении ресурсов, которые были определены как устойчивые к отказу. Восстановление должно быть прозрачным для пользователя. То есть, он не должен замечать, что сеть переключилась на резервные ресурсы (даже при использовании приложений, чувствительных ко времени).

Сами резервные ресурсы могут быть либо выделенными, либо разделяемыми. Выделенный ресурс означает, что для каждой виртуальной сети может быть настроена полная резервная сеть и резервные ресурсы полностью назначены виртуальным сетям и независимы друг от друга. Однако это неэффективное использование ресурса, поскольку для каждого встраиваемого

виртуального ресурса необходим выделенный элемент физической сети. В некоторых случаях это также может быть приемлемым для совместного или повторного использования резервных ресурсов. Как правило, более высокая степень повторного использования резервного ресурса приводит к снижению надежности, и наоборот.

Классификация методов встраивания виртуальных сетей
В зависимости от того, где и как осуществляется встраивание, различают статический (т.е. с неизменными в виртуальной и физической инфраструктуре) подход к VNE. Все подходы к VNE, предложенные в литературе могут быть классифицированы в зависимости от того, являются ли они статическими или динамическими, централизованными или распределенными, без или с избыточностью. Ниже рассмотрены эти подходы.

Статическое и динамическое встраивание VNE
В большинстве ситуаций в реальном мире VNE должна решаться как проблема реального времени, т.е. запросы VNR не будут известны заранее. Другими словами, они приходят в систему динамически и могут оставаться в сети в течение произвольного времени. Для работы в реальных условиях алгоритм VNE должен обрабатывать VNR по мере их поступления, нежели назначать однократно набор VNR (автономный VNE). Хотя в принципе все подходы могут работать в режиме реального времени (онлайн), статические подходы VNE не рассматривают возможность переназначения одного или нескольких VNR для повышения эффективности встраивания в SN.

Централизованное и распределенное встраивание
Проблема VNE может быть решена централизованным, либо распределенным способом. Оба подхода принципиально разные, каждый из них имеет свои преимущества и недостатки.
Централизованное: при централизованном подходе имеется один объект, который отвечает за выполнение встраивания. Это может быть выделенная машина для вычисления оптимальных решений проблемы. Преимущество этого подхода заключается том, что объект отображения на каждом шаге встраивания знает общую ситуацию в сети (т.е. обладает глобальным знанием). Это способствует более оптимальному встраиванию.

Распределенное: в отличие от централизованного решения, распределенный подход использует несколько объектов для вычисления вложения. Может использоваться для организации того, как отображение объектов для участвующих объектов, либо распределенно организовано полностью адаптированным способом. Преимущество такого подхода заключается в его лучшей масштабируемости. Поскольку нагрузка распределяется между несколькими узлами, каждый отдельный узел будет лучше справляться с вложениями. Однако, при этом увеличиваются служебные данные синхронизации. В частности, каждому узлу требуется доступ к информации о глобальном состоянии сети. Чем больше информация доступно узлу, тем лучше будут результаты.

Оптимизация виртуальных сетей: обзор критериев и моделей

Критерии необходимы для количественной оценки качества успешного встраивания виртуальной сервисной сети в общую физическую сеть. Они используются для сравнения различных алгоритмов встраивания VNE. В качестве критериев эффективности используются следующие показатели:

- показатели качества обслуживания;
- экономические показатели;
- показатели устойчивости;

Основными показателями качества обслуживания являются задержка и джиттер пакета. Задержка показывает время передачи пакета от одного узла к другому. Алгоритм встраивания может минимизировать задержку пакета между виртуальными узлами. Джиттер показывает дисперсию времени задержки пакетов. В настоящее время не существует алгоритм вложения с целью минимизации джиттера.

В дополнение к типичным показателям QoS, необходимо включить среднюю длину пути виртуального канала между виртуальными узлами. Чем длиннее путь, тем больше физических ресурсов (физических узлов и каналов) необходимо зарезервировать для вложения виртуального канала. Поскольку каждый физический узел является частью виртуального канала, то задержка передачи пакетов между виртуальными узлами увеличивается.

Основными экономическими показателями являются стоимость и доход. Стоимость в контексте алгоритма встраивания

относится к сумме стоимости физических ресурсов физической сети, которые использовались для внедрения виртуальных сетей. Стоимость узлов и полосы пропускания каналов физической сети суммируются, и тем выше затраты на внедрение виртуальной сети, тем больше доход от виртуальных сетей, предоставляющих различные сервисы.

Показателями устойчивости являются количество резервных виртуальных узлов и путей передачи пакетов, время реконфигурации топологии после сбоя, возможность миграции виртуальных узлов и путей с целью недопущения перегрузок физических узлов и каналов передачи данных. В сетевой виртуализации появляется возможность применения индивидуальной политики безопасности для каждой виртуальной сервисной сети. Все эти механизмы направлены на обеспечение устойчивого и безопасного функционирования виртуальных сетей.

Концептуальная модель виртуальных сервисных сетей

Модель физической инфраструктуры инфокоммуникационной сети будем задавать графом [35]

$$GP = (PN, PL), \quad (4.99)$$

где PN- множество физических устройства (Physical Node) и PL – множество физических каналов передачи данных (Physical Link). Множество физических устройств состоит из подмножеств физических серверов (PS-Physical Server), физических маршрутизаторов (PR-Physical Router) и физических устройств хранения данных (PSG-Physical Storage):

$$PN = (PS, PR, PSG). \quad (4.100)$$

Основными характеристиками (H) физических ресурсов графа GP являются

$$H_{GP} = (V_{ps}, C_{pr}, M_{psg}, C_{pl}), \quad (4.101)$$

где V_{ps} - производительность физического сервера $ps \in PS$, C_{pr} - пропускная способность физического маршрутизатора $pr \in PR$, M_{psg} - емкость физического устройства хранения данных $psg \in PSG$, C_{pl} - пропускная способность физического канала передачи данных $pl \in PL$.

Модель виртуальной наложенной сервисной сети задается графом GV :

$$GV = (VN, VL), \quad (4.102)$$

$$VN = (VS, VR, VSG),$$

где VS, VR, VSG и VL - соответственно виртуальный сервер, виртуальный маршрутизатор, виртуальное устройство хранения данных и виртуальный канал передачи данных.

Набор характеристик виртуальных ресурсов совпадает с набором характеристик соответствующего ему физического ресурса $H_{GV} = (V_{vs}, C_{vr}, M_{vsg}, C_{vl})$, $vs \in VS, vr \in VR, vsg \in VSG, vl \in VL$.

Для построения виртуальной наложенной сервисной сети в физической инфраструктуре инфокоммуникационной сети необходимо выполнить отображение $GV \rightarrow GP = \{VS \rightarrow PS, VR \rightarrow PR, VSG \rightarrow PSG, VL \rightarrow (PR, PL)\}$.

При отображении $GV \rightarrow GP$ необходимо соблюдать следующие условия недопустимости перегрузки физических ресурсов:

$$\sum_{vs \in VS} V_{vs} \leq V_{ps}, \quad ps \in PS, \quad (4.105)$$

$$\sum_{vr \in VR} C_{vr} \leq C_{pr}, \quad pr \in PR, \quad (4.106)$$

$$\sum_{vsg \in VSG} M_{vsg} \leq M_{psg}, \quad psg \in PSG, \quad (4.107)$$

$$\sum_{vl \in VL} C_{vl} \leq C_{pl}, \quad pl \in PL. \quad (4.108)$$

а также условия соблюдения требований на показатели качества обслуживания ($QoS_{треб}$):

$$T_{Qos} \leq T_{Qos_{треб}}, D_{Qos} \leq D_{Qos_{треб}}, P_{Qos} \leq P_{Qos_{треб}}, \quad (4.109)$$

где T_{Qos} - задержка передачи пакетов, D_{Qos} - вариация задержки (джиттер) пакетов и P_{Qos} - вероятность потери и ошибки пакетов. После отображения $GV \rightarrow GP$ значения характеристик физических ресурсов уменьшаются и формируется остаточный граф (residual graph) GPR с переопределенными значениями характеристик

$$H_{GPR} = H_{GP} - H_{GV}.$$

Следующая виртуальная SON отображается в физической сети ($GV \rightarrow GPR$) с соблюдением условий (4.105-4.109).

Формализация задачи оптимального построения виртуальных сервисных сетей

Учитывая, что на сегодняшний день в инфраструктуре телекоммуникаций большинства операторов используется решение, основанное на применении технологии MPLS (англ. Multiprotocol Label Switching - мультипротокольная коммутация по меткам), в данном разделе рассматривается базовая сеть на основе IP/MPLS [35].

В соответствии с технологией IP/MPLS множество маршрутизаторов физической сети PR можно разделить на два подмножества:

$$PR = (PR^{pe} \cup PR^p), \quad (4.11)$$

$$PR^{pe} = \{pr_i^{pe}, i = \overline{1, n_{pe}}\},$$

$$PR^p = \{pr_j^p, j = \overline{1, n_p}\},$$

где PR^{pe} - подмножество граничных маршрутизаторов и подмножество маршрутизаторов ядра физической сети.

К граничным маршрутизаторам PR^{pe} источники и получатели информации. Источники информации могут быть пользователи сети, сервера и устройства хранения информации. Трафик между маршрутизаторами PR^{pe} задается следующей матрицей:

$$F_{pk} = |f_{ij}^{pe}|, i = \overline{1, n_{pe}}, j = \overline{1, n_{pe}},$$

$$f_{ij}^{pe} = 0 \text{ при } i = j. \quad (4.112)$$

где f_{ij}^{pe} - интенсивность потока, передаваемого от i -го граничного маршрутизатора k -ому граничному маршрутизатору.

Зададим связность граничных маршрутизаторов с маршрутизаторами ядра и связность маршрутизаторов ядра между собой в соответствии со следующими матрицами:

$$A_{pe}^p = |a_{ij}^{pe,p}|, i = \overline{1, n_{pe}}, j = \overline{1, n_p}, \quad (4.113)$$

если pr_i^{pe} соединен с pr_j^p , то $a_{ij}^{pe,p} = 1$, иначе $a_{ij}^{pe,p} = 0$,

$$A_p^p = |a_{ij}^{p,p}|, i = \overline{1, n_p}, j = \overline{1, n_p}, \quad (4.114)$$

если pr_i^p соединен с pr_j^p , то $a_{ij}^{p,p} = 1$, иначе $a_{ij}^{p,p} = 0$,

$$a_{ij}^{p,p} = 0 \text{ при } i = j.$$

Пропускные способности физических каналов передачи данных между граничными маршрутизаторами и маршрутизаторами ядра, а также между маршрутизаторами ядра задаются следующими матрицами:

$$C_{pe}^p = |c_{ij}^{pe,p}|, i = \overline{1, n_{pe}}, j = \overline{1, n_p}, \text{ если } a_{ij}^{pe,p} = 0, \text{ то } c_{ij}^{pe,p} = 0, \quad (4.115)$$

$$C_p^p = |c_{ij}^{p,p}|, i = \overline{1, n_p}, j = \overline{1, n_p}, \text{ если } a_{ij}^{p,p} = 0, \text{ то } c_{ij}^{p,p} = 0, \quad (4.116)$$

$$f_{ij}^{pe,p} \leq c_{ij}^{pe,p}, f_{ij}^{p,p} \leq c_{ij}^{p,p},$$

где $f_{ij}^{pe,p}$ - интенсивность потока, передаваемого между pr_i^{pe} и pr_j^p ;

$f_{ij}^{p,p}$ - интенсивность потока, передаваемого между pr_i^p и pr_j^p .

Матрицы связности физических серверов (PS) и устройств хранения данных (PSG) с граничными маршрутизаторами (PR^{pe}) задаются следующими матрицами:

$$A_{ps}^{pe} = |a_{ij}^{ps,pe}|, i = \overline{1, n_{ps}}, j = \overline{1, n_{pe}}, \quad (4.117)$$

если ps_i соединен с pr_j^{pe} , то $a_{ij}^{ps,pe} = 1$, иначе $a_{ij}^{ps,pe} = 0$,

если psg_i соединен с pr_j^{pe} , то $a_{ij}^{psg,pe} = 1$, иначе $a_{ij}^{psg,pe} = 0$,
 Пропускные способности физических каналов передачи данных, соединяющие PS и PSG с PR^{pe} :

$$C_{ps}^{pe} = |c_{ij}^{ps,pe}|, i = \overline{1, n_{ps}}, j = \overline{1, n_{pe}}, \text{ если } a_{ij}^{ps,pe} = 0, \text{ то } c_{ij}^{ps,pe} = 0, \quad (4.118)$$

$$C_{psg}^{pe} = |c_{ij}^{psg,pe}|, i = \overline{1, n_{psg}}, j = \overline{1, n_{pe}}, \text{ если } a_{ij}^{psg,pe} = 0, \text{ то } c_{ij}^{psg,pe} = 0, \quad (4.119)$$

$$f_{ij}^{ps,pe} \leq c_{ij}^{ps,pe}, f_{ij}^{psg,pe} \leq c_{ij}^{psg,pe},$$

где $f_{ij}^{ps,pe}$ - интенсивность потока, передаваемого между ps_i и pr_j^{pe} ;
 $f_{ij}^{psg,pe}$ - интенсивность потока, передаваемого между psg_i и pr_j^{pe} .

В соответствии с вышеизложенной методикой, формализация конфигурации виртуальной сети задается следующим образом:

- виртуальные маршрутизаторы:

$$VR = (VR^{pe} \cup VR^p), \quad (4.121)$$

$$VR^{pe} = \{vr_i^{pe}, i = \overline{1, n_{vpe}}\},$$

$$VR^p = \{vr_j^p, j = \overline{1, n_{vp}}\},$$

- трафик между виртуальными граничными маршрутизаторами:

$$F_{VPE} = |f_{ij}^{vpe}|, i = \overline{1, n_{vpe}}, j = \overline{1, n_{vpe}},$$

$$f_{ij}^{vpe} = 0 \text{ при } i = j;$$

- матрицы связности виртуальных маршрутизаторов:

$$A_{vpe}^{vp} = |a_{ij}^{vpe,vp}|, i = \overline{1, n_{vpe}}, j = \overline{1, n_{vp}}, \quad (4.123)$$

если vr_i^{vpe} соединен с vr_j^{vp} , то $a_{ij}^{vpe,vp} = 1$, иначе $a_{ij}^{vpe,vp} = 0$,
 $a_{ij}^{vpe,vp} = 0$ при $i = j$; (4.124)

$$A_{vp}^p = |a_{ij}^{vp,vp}|, i = \overline{1, n_{vp}}, j = \overline{1, n_{vp}},$$

если vr_i^p соединен с vr_j^p , то $a_{ij}^{vp,vp} = 1$, иначе $a_{ij}^{vp,vp} = 0$,
 $a_{ij}^{vp,vp} = 0$ при $i = j$;

$$C_{vp}^{vp} = |c_{ij}^{vp,vp}|, i = \overline{1, n_{vp}}, j = \overline{1, n_{vp}}, \text{ если } a_{ij}^{vp,vp} = 0, \text{ то } c_{ij}^{vp,vp} = 0; \quad (4.125)$$

$$f_{ij}^{vpe,vp} \leq c_{ij}^{vpe,vp}, f_{ij}^{vp,vp} \leq c_{ij}^{vp,vp},$$

- матрицы связности виртуальных серверов и устройств хранения данных с виртуальными граничными маршрутизаторами:

$$A_{vs}^{vpe} = \{a_{ij}^{vs,vpe}\}, i = \overline{1, n_{vs}}, j = \overline{1, n_{vpe}},$$

если vs_i соединен с vr_j^{vpe} , то $a_{ij}^{vs,vpe} = 1$, иначе $a_{ij}^{vs,vpe} = 0$, (4.126)

$$A_{vsg}^{vpe} = \{a_{ij}^{vsg,vpe}\}, i = \overline{1, n_{vsg}}, j = \overline{1, n_{vpe}},$$

если vsg_i соединен с vr_j^{vpe} , то $a_{ij}^{vsg,vpe} = 1$, иначе $a_{ij}^{vsg,vpe} = 0$, (4.127)

- матрицы пропускных способностей виртуальных каналов передачи, соединяющие VS и VSG с VR^{vpe}:

$$C_{vs}^{vpe} = \{c_{ij}^{vs,vpe}\}, i = \overline{1, n_{vs}}, j = \overline{1, n_{vpe}},$$

если $a_{ij}^{vs,vpe} = 0$, то $c_{ij}^{vs,vpe} = 0$, (4.128)

$$C_{vsg}^{vpe} = \{c_{ij}^{vsg,vpe}\}, i = \overline{1, n_{vsg}}, j = \overline{1, n_{vpe}},$$

если $a_{ij}^{vsg,vpe} = 0$, то $c_{ij}^{vsg,vpe} = 0$, (4.129)

$$f_{ij}^{vs,vpe} \leq c_{ij}^{vs,vpe}, f_{ij}^{vsg,vpe} \leq c_{ij}^{vsg,vpe}$$

Требуется отобразить (встроить) виртуальные сети в физической сети таким образом, чтобы стоимость отображения была бы наименьшей.

Результаты отображения фиксируются в соответствующих матрицах с переменными элементами x , значения которых необходимо определить на основе используемого алгоритма отображения.

Матрица отображения виртуальных серверов в физических серверах:

$$X_{vs}^{ps} = \{x_{ij}^{vs,ps}\}, i = \overline{1, n_{vs}}, j = \overline{1, n_{ps}},$$

если $vs_i \rightarrow ps_j$, то $x_{ij}^{vs,ps} = 1$, иначе $x_{ij}^{vs,ps} = 0$. (4.130)

Матрица отображения виртуальных устройств хранения данных на физических устройствах хранения данных:

$$X_{vsg}^{psg} = \{x_{ij}^{vsg,psg}\}, i = \overline{1, n_{vsg}}, j = \overline{1, n_{psg}},$$

если $vsg_i \rightarrow psg_j$, то $x_{ij}^{vsg,psg} = 1$, иначе $x_{ij}^{vsg,psg} = 0$. (4.131)

Матрица отображения виртуальных граничных маршрутизаторов в физических граничных маршрутизаторах:

$$X_{vpe}^{pe} = \{x_{ij}^{vpe,pe}\}, i = \overline{1, n_{vpe}}, j = \overline{1, n_{pe}},$$

если $vr_i^{vpe} \rightarrow pr_j^{pe}$, то $x_{ij}^{vpe,pe} = 1$, иначе $x_{ij}^{vpe,pe} = 0$. (4.132)

Матрица отображения виртуальных маршрутизаторов в физических маршрутизаторах ядра сети:

$$X_{vp}^p = \{x_{ij}^{vp,p}\}, i = \overline{1, n_{vp}}, j = \overline{1, n_p},$$

если $vr_i^{vp} \rightarrow pr_j^p$, то $x_{ij}^{vp,p} = 1$, иначе $x_{ij}^{vp,p} = 0$. (4.133)

Матрицы отображения виртуальных каналов в физических каналах передачи данных: (4.134)

$$X_{vt}^{pl} = \{x_{ij}^{vt,pl}\}, i = \overline{1, n_{vt}}, j = \overline{1, n_{pl}},$$

если $vt_i \rightarrow pl_j$, то $x_{ij}^{vt,pl} = 1$, иначе $x_{ij}^{vt,pl} = 0$.

Виртуальный канал, состоящий из составных физических каналов, также отображается в физических маршрутизаторах. Поэтому необходимо составить матрицы отображения виртуальных каналов в граничных маршрутизаторах и маршрутизаторах ядра:

$$X_{vt}^{pe} = \{x_{ij}^{vt,pe}\}, i = \overline{1, n_{vt}}, j = \overline{1, n_{pe}},$$

если $vt_i \rightarrow pr_j^{pe}$, то $x_{ij}^{vt,pe} = 1$, иначе $x_{ij}^{vt,pe} = 0$, (4.135)

$$X_{vt}^p = \{x_{ij}^{vt,p}\}, i = \overline{1, n_{vt}}, j = \overline{1, n_p},$$

если $vt_i \rightarrow pr_j^p$, то $x_{ij}^{vt,p} = 1$, иначе $x_{ij}^{vt,p} = 0$. (4.137)

Стоимости отображения виртуальных ресурсов рассчитываются как выделяемые величины характеристик физических ресурсов виртуальных серверов:

$$S_{vs} = \sum_{i=1}^{n_{vs}} \sum_{j=1}^{n_{ps}} x_{ij}^{vs,ps} S_{ij}^{vs,ps} V_{vs_i},$$

где $S_{ij}^{vs,ps}$ - стоимость единицы производительности j -го физического сервера при отображении в нем i -го виртуального сервера с производительностью V_{vs_i} . (4.138)

Стоимость отображения виртуальных устройств хранения данных:

$$S_{vsg} = \sum_{i=1}^{n_{vsg}} \sum_{j=1}^{n_{psg}} x_{ij}^{vsg,psg} S_{ij}^{vsg,psg} M_{vsg_i},$$

где $S_{ij}^{vsg,psg}$ - стоимость единицы ёмкости j -го физического устройства хранения данных при отображении в нем i -го виртуального устройства хранения данных с ёмкостью M_{vsg_i} . (4.139)

Стоимость маршрутизаторов; отображения виртуальных граничных

$$S_{vpe} = \sum_{i=1}^{n_{vpe}} \sum_{j=1}^{n_p} x_{ij}^{vpe,pe} s_{ij}^{vpe,pe} C_{vpe_i}, \quad (4.140)$$

где $s_{ij}^{vpe,pe}$ - стоимость единицы пропускной способности j -го физического граничного маршрутизатора при отображении в нем i -го виртуального граничного маршрутизатора с пропускной способностью C_{vpe_i} .

Стоимость отображения виртуальных маршрутизаторов ядра:

$$S_{vp} = \sum_{i=1}^{n_{vp}} \sum_{j=1}^{n_p} x_{ij}^{vp,p} s_{ij}^{vp,p} C_{vp_i}, \quad (4.141)$$

где $s_{ij}^{vp,p}$ - стоимость единицы пропускной способности j -го физического маршрутизатора ядра при отображении в нем i -го виртуального маршрутизатора ядра с пропускной способностью C_{vp_i} .

Стоимость отображения виртуальных каналов передачи данных:

$$S_{vl} = \sum_{i=1}^{n_{vl}} \left(\sum_{j=1}^{n_{pl}} x_{ij}^{vl,pl} s_{ij}^{vl,pl} + \sum_{j=1}^{n_{pe}} x_{ij}^{vl,pe} s_{ij}^{vl,pe} + \sum_{j=1}^{n_p} x_{ij}^{vl,p} s_{ij}^{vl,p} \right) C_{vl_i}, \quad (4.142)$$

где $s_{ij}^{vl,pl}$, $s_{ij}^{vl,pe}$ и $s_{ij}^{vl,p}$ - соответственно стоимости единиц пропускной способности j -го физического канала передачи данных, пропускной способности граничного маршрутизатора и маршрутизатора ядра при отображении в них i -го виртуального канала передачи данных с пропускной способностью C_{vl_i} .

Стоимость отображения виртуальной сети равна сумме стоимостей отображений всех её виртуальных ресурсов:

$$S_{VN} = S_{vs} + S_{vsg} + S_{vpe} + S_{vp} + S_{vl}. \quad (4.143)$$

Доход от реализации виртуальной сети определяется по формуле:

$$D_{VN} = \sum_{i=1}^{n_{vpe}} \sum_{j=1}^{n_{pe}} d_{ij} f_{ij}^{vpe}, \quad (4.144)$$

где d_{ij} - доход на единицу полосы пропускания при обслуживании трафика от i -го виртуального граничного маршрутизатора к j -ому виртуальному граничному маршрутизатору; f_{ij}^{vpe} - объем трафика от i -го виртуальному граничному маршрутизатору.

В физической сети могут быть отображены несколько виртуальных сетей. Количество отображаемых виртуальных сетей является переменной величиной k ($k = 1, 2, \dots, N_{VN}$), зависимой от эффективности методов (алгоритмов) отображения. Чем больше эффективность сетей отображаются в физической сети, тем доход (стоимость) метод отображения и больше (меньше).

Таким образом, необходимо:

$$\max: \sum_{k=1}^{N_{VN}} (D_{VN}(k) - S_{VN}(k)), \quad (4.145)$$

при следующих ограничениях:

$$\sum_{k=1}^{N_{VN}} \sum_{i=1}^{n_{vs}} x_{ij}^{vs,ps}(k) V_{vs_i}(k) \leq V_{ps_j}, \quad j = \overline{1, n_{ps}}, \quad (4.146)$$

$$\sum_{k=1}^{N_{VN}} \sum_{i=1}^{n_{vsg}} x_{ij}^{vsg,psg}(k) V_{vsg_i}(k) \leq V_{psg_j}, \quad j = \overline{1, n_{psg}}, \quad (4.147)$$

$$\sum_{k=1}^{N_{VN}} \left(\sum_{i=1}^{n_{vpe}} x_{ij}^{vpe,pe}(k) C_{vpe_i}(k) + \sum_{i=1}^{n_{vp}} x_{ij}^{vp,pe}(k) C_{ij}^{vp,pe}(k) \right) \leq C_{pe_j}, \quad j = \overline{1, n_{pe}}, \quad (4.148)$$

$$\sum_{k=1}^{N_{VN}} \left(\sum_{i=1}^{n_{vp}} x_{ij}^{vp,p}(k) C_{vp_i}(k) + \sum_{i=1}^{n_p} x_{ij}^{vl,p}(k) C_{ij}^{vl,p}(k) \right) \leq C_{p_j}, \quad j = \overline{1, n_p}, \quad (4.149)$$

$$\sum_{k=1}^{N_{VN}} \sum_{i=1}^{n_{vl}} x_{ij}^{vl,pl}(k) C_{ij}^{vl,pl}(k) \leq C_{pl_j}, \quad f_{ij}^{vl,pl}(k) \leq C_{ij}^{vl,pl}(k), \quad j = \overline{1, n_{pl}}, \quad (4.150)$$

$$T_{QoS}(k) \leq T_{QoS_{mpe}}(k); D_{QoS}(k) \leq D_{QoS_{mpe}}(k); P_{QoS}(k) \leq P_{QoS_{mpe}}(k), \quad k = \overline{1, N_{VN}}. \quad (4.151)$$

Ограничения (4.146-4.151) означают, что сумма характеристик виртуальных ресурсов (серверов, устройств хранения данных граничных маршрутизаторов, маршрутизаторов ядра и канал передачи данных) виртуальных сетей не должна превышать

характеристик физических ресурсов физической сети. Ограничения (4.151) означают, что для всех виртуальных сетей необходимо соблюдать требования на показатели качества обслуживания.

Метод встраивания виртуальных сетей с перебором вариантов
 (NP-Non-deterministic polynomial-timehardness). Задача встраивания виртуальных сетей является NP-трудной комбинаторной задачей с полиномиальной оценкой числа итераций). Для больших размеров виртуальных и физических сетей время нахождения оптимального решения становится очень большим.

Метод оптимального встраивания виртуальных сетей с учетом связности маршрутизаторов

Данный метод учитывает связность маршрутизаторов [35]. Значение связности маршрутизатора равно количеству маршрутизаторов, подключенных к этому маршрутизатору. Основное правило встраивания данного метода заключается в том, что виртуальный маршрутизатор с наибольшей связностью встраивается в физический маршрутизатор с наибольшей связностью. Далее виртуальные маршрутизаторы, подключенные к виртуальному маршрутизатору с наибольшей связностью, встраиваются в физические маршрутизаторы, подключенные к физическому маршрутизатору с наибольшей связностью. Это правило встраивания нацелено на организацию виртуального канала, состоящего только из одного физического канала.

Оценка эффективности методов встраивания виртуальных сетей

С целью оценки эффективности методов встраивания виртуальных сетей проведены вычислительные эксперименты [35]. Исходными данными для вычислительных экспериментов являются:

- топология и параметры элементов физической сети;
- топологии и параметры элементов виртуальных сетей;
- алгоритм встраивания виртуальных сетей.

Выходными данными для вычислительных экспериментов являются:

- топология виртуальной сети, встроенной в физическую сеть;
- количество встроенных виртуальных сетей;
- объем физических сетевых ресурсов, используемых при встраивании виртуальных сетей.

Эффективность методов встраивания виртуальных сетей будем оценивать на основе следующих критериев:

- коэффициент встраивания виртуальных сетей;
- коэффициент эффективного использования ресурсов физической сети.

Коэффициент эффективного использования ресурсов физической сети определяется по формуле:

$$k_{cv} = \frac{N_{vc}}{N_{max\ vc}} \quad (4.152)$$

где $N_{max\ vc}$ - максимально возможное количество встраиваемых виртуальных сетей в заданную физическую сеть; N_{vc} - количество встроенных виртуальных сетей при реализованном алгоритме встраивания.

Коэффициент эффективного использования ресурсов физической сети определяется по формуле:

$$k_{ef} = \frac{V_{min\ fc}}{V_{fc}} \quad (4.153)$$

где $V_{min\ fc}$ - минимальный объем физических сетевых ресурсов для встраивания N_{mvc} виртуальных сетей; V_{fc} - объем физических сетевых ресурсов, используемых для встраивания виртуальных сетей при реализованном алгоритме встраивания.

Значения N_{mvc} и $V_{min\ fc}$ определяются с помощью алгоритма встраивания виртуальных сетей на основе перебора. Значения k_{cv} и k_{ef} , полученные на основе вычислительных экспериментов при различных исходных данных, приведены в таблице 4.6 и на рисунке 4.18.

Таблица 4.6

Результаты вычислительных экспериментов			
№	Метод встраивания виртуальной сети	k_{cv}	k_{ef}
1	Метод встраивания с сортировкой пропускных способностей маршрутизаторов по возрастанию	0.31	0.65
2	Метод встраивания с сортировкой пропускных способностей маршрутизаторов по убыванию	0.52	0.71
3	Метод встраивания с учетом связности маршрутизаторов	0.86	0.94

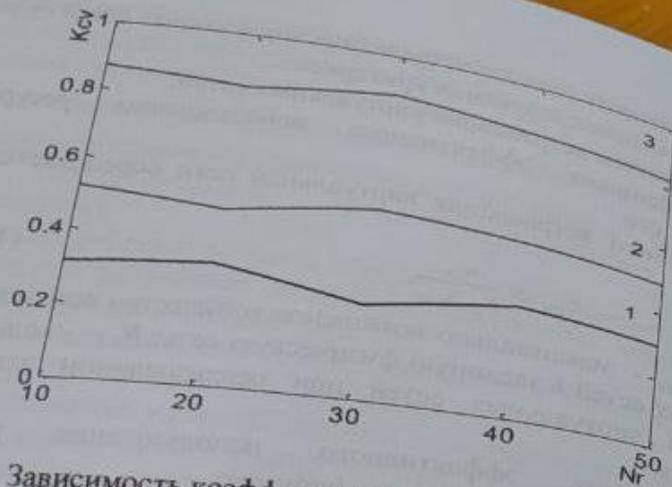


Рис.4.18. Зависимость коэффициента встраивания виртуальных сетей от количества маршрутизаторов (Nr) при различных методах

Из таблицы 4.6 и рисунка 4.18 следует, что метод встраивания с учетом связности маршрутизаторов (3) обеспечивает наибольшие коэффициенты встраивания виртуальных сетей и эффективного использования физических сетевых ресурсов по сравнению с другими методами (1 и 2). Как видно, в среднем 86 % виртуальных сетей встраиваются.

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ГЛАВЕ 4

1. Какие протоколы передачи данных вы знаете?
2. Объясните модель передачи данных.
3. Объясните параметры и характеристики системы передачи данных.
4. Перечислите задачи оптимизации системы передачи данных.
5. Объясните процессы передачи и приема пакетов.
6. Каковы причины ошибок пакетов?
7. Каковы причины потери пакетов?
8. Объясните процесс расчета вероятности ошибки приема пакетов.
9. Объясните процесс расчета вероятности потери пакетов.
10. Какие методы маршрутизации вы знаете?
11. Каковы недостатки традиционных протоколов маршрутизации?
12. Объясните основы QoS-маршрутизации.

13. Важность балансировки сетевых устройств.
14. Критерии оптимизации маршрутизации.
15. Эффективность методов маршрутизации.
16. Цель построения виртуальных сетей.
17. Методы встраивания виртуальных сетей?
18. Какие задачи решаются при встраивании виртуальных сетей в физическую сеть?
19. Алгоритмы оптимизацией виртуальных сетей?
20. Какие показатели входят в критерии оптимизации виртуальных сетей?

5. СИМУЛЯЦИЯ И ОПТИМИЗАЦИЯ СЕТЕЙ С ПОМОЩЬЮ СПЕЦИАЛЬНЫХ ПРОГРАММНЫХ СРЕДСТВ

5.1. Симуляция систем в среде GPSS World

Исходная программа на языке GPSS (GPSS/PC, GPSS WORLD), как и программа на любом языке программирования, представляет собой последовательность операторов [34]. Операторы GPSS записываются и вводятся в ПК в следующем формате:

Все операторы исходной программы должны начинаться с номера строки - целого положительного числа от 1 до 9999999. После ввода операторов они располагаются в исходной программе в соответствии с нумерацией строк.

В GPSS WORLD номер строки можно не писать для лучшей читаемости текста.

В поле операции записывается ключевое слово (название оператора), указывающее конкретную функцию, выполняемую данным оператором.

Каждый оператор GPSS относится к одному из четырех типов: операторы-блоки, операторы определения объектов, управляющие операторы и операторы-команды.

Блоки, связанные с транзактами

С транзактами связаны блоки создания, уничтожения, задержки транзактов, изменения их атрибутов и создания копий транзактов.

Для создания транзактов, входящих в модель, служит блок GENERATE (генерировать), имеющий следующий формат:

имя GENERATE A,B,C,D,E.

В поле A задается среднее значение интервала времени между моментами поступления в модель двух последовательных транзактов. Если этот интервал постоянен, то поле B не используется. Если же интервал поступления является случайной величиной, то в поле B указывается модификатор среднего значения, который может быть задан в виде модификатора-интервала или модификатора-функции.

Например, блок GENERATE 100,40 создает транзакты через случайные интервалы времени, равномерно распределенные на отрезке [60;140].

В поле C задается момент поступления в модель первого транзакта. Если это поле пусто или равно 0, то момент появления первого транзакта определяется операндами A и B.

Поле D задает общее число транзактов, которое должно быть создано блоком GENERATE. Если это поле пусто, то блок генерирует неограниченное число транзактов до завершения моделирования.

В поле E задается приоритет, присваиваемый генерируемым транзактам. Число уровней приоритетов неограничено, причем самый низкий приоритет - нулевой. Если поле E пусто, то генерируемые транзакты имеют нулевой приоритет.

Для присваивания параметрам начальных значений или изменения этих значений служит блок ASSIGN (присваивать), имеющий следующий формат:

имя ASSIGN A,B,C.

В поле A указывается номер или имя параметра, в который заносится значение операнда B. Если в поле A после имени (номера) параметра стоит знак + или -, то значение операнда B добавляется или вычитается из текущего содержимого параметра. В поле C может быть указано имя или номер функции-модификатора, действующей аналогично функции-модификатору в поле B блока GENERATE.

Например, блок

ASSIGN 5,0

записывает в параметр с номером 5 значение 0, а блок

ASSIGN COUNT+,1

добавляет 1 к текущему значению параметра с именем COUNT.

Для удаления транзактов из модели служит блок TERMINATE (завершить), имеющий следующий формат:

имя TERMINATE A.

Значение поля A указывает, на сколько единиц уменьшает содержимое так называемого счетчика завершений при вхо-

транзакта в данный блок TERMINATE. Если поле A не определено, то оно считается равным 0, и транзакты, проходящие через такой блок, не уменьшают содержимого счетчика завершений.

Начальное значение счетчика завершений устанавливается управляющим оператором START (начать), предназначенным для запуска прогона модели. Поле A этого оператора содержит начальное значение счетчика завершений. Прогон модели заканчивается, когда содержимое счетчика завершений обращается в 0. Таким образом, в модели должен быть хотя бы один блок TERMINATE с непустым полем A, иначе процесс моделирования никогда не завершится.

Участок блок-схемы модели, связанный с парой блоков GENERATE-TERMINATE, называется сегментом. Простые модели могут состоять из одного сегмента, в сложных моделях может быть несколько сегментов.

Например, простейший сегмент модели, состоящий всего из двух блоков GENERATE и TERMINATE и приведенный на рис. 5.1, в совокупности с управляющим оператором START моделирует процесс создания случайного потока транзактов, поступающих в модель со средним интервалом в 100 единиц модельного времени, и уничтожения этих транзактов. Начальное значение счетчика завершений равно 1000. Каждый транзакт, проходящий через блок TERMINATE, вычитает из счетчика единицу, и таким образом моделирование завершится, когда тысячный по счету транзакт войдет в блок TERMINATE. При этом точное значение таймера в момент завершения прогона непредсказуемо. Следовательно, в приведенном примере продолжительность прогона устанавливается не по модельному времени, а по количеству транзактов, прошедших через модель.

```
GENERATE 100,40
TERMINATE 1
START 1000
```

Рис. 5.1. Простой сегмент модели.

Если необходимо управлять продолжительностью прогона по модельному времени, то в модели используется специальный сегмент, называемый сегментом таймера (рис. 5.2).

```
GENERATE 100,40
TERMINATE
GENERATE 100000
TERMINATE 1
START 1
```

Рис. 5.2. Сегмент таймера.

Например, в модели из двух сегментов, приведенной на рис. 5.2, первый (основной) сегмент выполняет те же функции, что и в предыдущем примере. Заметим, однако, что поле A блока TERMINATE в первом сегменте пусто, т.е. уничтожаемые транзакты не уменьшают содержимого счетчика завершений. Во втором сегменте блок GENERATE создаст первый транзакт в момент модельного времени, равный 100000. Но этот транзакт окажется и последним в данном сегменте, так как, войдя в блок TERMINATE, он обратит в 0 содержимое счетчика завершений, установленное оператором START равным 1. Таким образом, в этой модели гарантируется завершение прогона в определенный момент модельного времени, а точное количество транзактов, прошедших через модель, непредсказуемо.

В приведенных примерах транзакты, входящие в модель через блок GENERATE, в тот же момент модельного времени уничтожались в блоке TERMINATE. В моделях систем массового обслуживания заявки обслуживаются приборами (каналами) СМО в течение некоторого промежутка времени прежде, чем покинуть СМО. Для моделирования такого обслуживания, т.е. для задержки транзактов на определенный отрезок модельного времени, служит блок ADVANCE (задержать), имеющий следующий формат:

имя ADVANCE A,B

Операнды в полях A и B имеют тот же смысл, что и в соответствующих полях блока GENERATE. Следует отметить, что транзакты, входящие в блок ADVANCE, переводятся из списка текущих событий в список будущих событий, а по истечении вычисленного времени задержки возвращаются назад, в список текущих событий, и их продвижение по блок-схеме продолжается. Если вычисленное время задержки равно 0, то транзакт в тот же

момент модельного времени переходит в следующий блок, оставаясь в списке текущих событий.

Например, в сегменте, приведенном на рис.5.3, транзакты, поступающие в модель из блока GENERATE через случайные интервалы времени, имеющие равномерное распределение на отрезке [60;140], попадают в блок ADVANCE. Здесь определяется случайное время задержки транзакта, имеющее равномерное распределение на отрезке [30;130], и транзакт переводится в список будущих событий. По истечении времени задержки транзакт возвращается в список текущих событий и входит в блок TERMINATE, где уничтожается. Заметим, что в списке будущих событий, а значит и в блоке ADVANCE может одновременно находиться произвольное количество транзактов.

```
GENERATE 100,40
ADVANCE 80,50
TERMINATE 1
```

Рис.5.3. Сегмент с оператором ADVANCE.

Блоки, связанные с аппаратными объектами

Все примеры моделей, рассматривавшиеся выше, пока еще не являются моделями систем массового обслуживания, так как в них не учтена основная особенность СМО: конкуренция заявок на использование некоторых ограниченных ресурсов системы. Все транзакты, входящие в эти модели через блок GENERATE, немедленно получают возможность "обслуживания" в блоке ADVANCE, который никогда не "отказывает" транзактам во входе, сколько бы транзактов в нем не находилось.

Для моделирования ограниченных ресурсов СМО в модели должны присутствовать аппаратные объекты: одноканальные или многоканальные устройства. Одноканальные устройства создаются в текущей модели при использовании блоков SEIZE (занять) и RELEASE (освободить), имеющих следующий формат:

```
имя SEIZE A
имя RELEASE A
```

В поле A указывается номер или имя устройства. Если транзакт входит в блок SEIZE, то устройство, указанное в поле A, становится занятым и остаётся в этом состоянии до тех пор, пока

этот же транзакт не пройдет соответствующий блок RELEASE, освобождая устройство. Если устройство, указанное в поле A блока SEIZE, уже занято каким-либо транзактом, то никакой другой транзакт не может войти в этот блок и остаётся в предыдущем блоке. Транзакты, задержанные (заблокированные) перед блоком SEIZE, остаются в списке текущих событий и при освобождении устройства обрабатываются с учетом приоритетов и очередности поступления. Каждое устройство имеет следующие СЧА: F - состояние устройства (0 - свободно, 1 - занято); FR - коэффициент использования в долях 1000; FC - число занятых устройства; FT - целая часть среднего времени занятия устройства.

Блоки для сбора статистических данных

Модели СМО разрабатываются обычно для исследования различных характеристик, связанных с ожиданием заявок в очереди: длины очереди, времени ожидания и т.п., а в приведенных примерах очередь транзактов образуется в списке текущих событий и недоступна исследователю. Для регистрации статистической информации о процессе ожидания транзактов в модели должны присутствовать статистические объекты: очереди или таблицы.

Объекты типа очередь создаются в модели путем использования блоков - регистраторов очередей: QUEUE (стать в очередь) и DEPART (уйти из очереди), имеющих следующий формат:

```
имя QUEUE A,B
имя DEPART A,B.
```

В поле A указывается номер или имя очереди, а в поле B - число единиц, на которое текущая длина очереди увеличивается при входе транзакта в блок QUEUE или уменьшается при входе транзакта в блок DEPART. Обычно поле B пусто, и в этом случае его значение по умолчанию принимается равным 1.

Для сбора статистики о транзактах, заблокированных перед каким-либо блоком модели, блоки QUEUE и DEPART помещаются перед и после этого блока соответственно. При прохождении транзактов через блоки QUEUE и DEPART соответствующим образом изменяются следующие СЧА очередей: Q - текущая длина очереди; QM - максимальная длина очереди; QA - целая часть средней длины очереди; QC - общее число транзактов, вошедших в

очередь; QZ - число транзактов, прошедших через очередь без ожидания (число "нулевых" входов); QT - целая часть среднего времени ожидания с учетом "нулевых" входов; QX - целая часть среднего времени ожидания без учета "нулевых" входов.

Блоки, изменяющие маршруты транзактов

В приведенных выше примерах транзакты, выходящие из любого блока, всегда поступали в следующий блок. В более сложных моделях возникает необходимость направления транзактов к другим блокам в зависимости от некоторых условий. Эту возможность обеспечивают блоки изменения маршрутов транзактов.

Блок TRANSFER (передать) служит для передачи входящих в него транзактов в блоки, отличные от следующего. Блок имеет девять режимов работы, из которых рассмотрим здесь лишь три наиболее часто используемых. В этих трех режимах блок имеет следующий формат:

имя TRANSFER A,B,C.

Смысл операндов в полях A, B и C зависит от режима работы блока.

В режиме безусловной передачи поля A и C пусты, а в поле B указывается имя блока, к которому безусловным образом направляется транзакт, вошедший в блок TRANSFER. Например:

TRANSFER ,FINAL.

В режиме статистической передачи операнд A определяет вероятность, с которой транзакт направляется в блок, указанный в поле C. С вероятностью -A транзакт направляется в блок, указанный в поле B (в следующий, если поле B пусто). Вероятность в поле A может быть задана непосредственно десятичной дробью, начинающейся с точки. Например, блок

TRANSFER .75,THIS,THAT

с вероятностью 0,75 направляет транзакты в блок с именем THAT, а с вероятностью 0,25 - в блок с именем THIS.

Если же поле A начинается не с десятичной точки и не содержит одного из ключевых слов - признаков других режимов работы блока, то его значение рассматривается как количество тысячных долей в вероятности передачи. Например, предыдущий блок TRANSFER можно записать также в следующем виде:

TRANSFER 750,THIS,THAT.
В режиме логической передачи в поле A записывается ключевое слово BOTH (оба). Транзакт, поступающий в блок TRANSFER, сначала пытается войти в блок, указанный в поле B (или в следующий блок, если поле B пусто), а если это не удается, т.е. блок B отказывается от транзакта во входе, то транзакт задерживается в поле C. Если и эта попытка неудачна, то транзакт задерживается в блоке TRANSFER до изменения условий в модели, делающего возможным вход в один из блоков B или C, причем при одновременном возникшей возможности предпочтение отдается блоку B. Например:

TRANSFER BOTH,MET1,MET2.

Блок TEST (проверить) служит для задержки или изменения маршрутов транзактов в зависимости от соотношения двух СЧА. Он имеет следующий формат:

имя TEST X A,B,C.

Вспомогательный операнд X содержит условие проверки соотношения между СЧА и может принимать следующие значения: L (меньше); LE (меньше или равно); E (равно); GE (больше или равно); G (больше). Поле A содержит первый, а поле B - второй из сравниваемых СЧА. Если проверяемое условие A X B выполняется, то блок TEST пропускает транзакт в следующий блок. Если же это условие не выполняется, то транзакт переходит к блоку, указанному в поле C, а если оно пусто, то задерживается перед блоком TEST.

Например, блок

TEST LE P\$TIME,C1

не впускает транзакты, у которых значение параметра с именем TIME больше текущего модельного времени. Блок

TEST L Q\$LINE,5,OUT

направляет транзакты в блок с именем OUT, если текущая длина очереди LINE больше либо равна 5.

Для задержки или изменения маршрута транзактов в зависимости от состояния аппаратных объектов модели служит блок GATE (впустить), имеющий следующий формат:

имя GATE X A,B

Вспомогательный операнд X содержит код состояния проверяемого аппаратного объекта, а в поле A указывается имя или номер этого объекта. Если проверяемый объект находится

заданном состоянии, то блок GATE пропускает транзакт к следующему блоку. Если же заданное в блоке условие не выполняется, то транзакт переходит к блоку, указанному в поле В, а если это поле пусто, то задерживается перед блоком GATE.

Операнд X может принимать следующие значения: U (устройство занято); NU (устройство свободно); I (устройство захвачено); NI (устройство не захвачено); SE (MKY не пусто); SF (MKY заполнено); SNF (MKY не заполнено); LS (ЛП включен), LR (ЛП выключен).

Например, блок

```
GATE SNE BUF3
```

отказывает во входе транзактам, поступающим в моменты, когда в MKY с именем BUF3 все каналы обслуживания свободны. Блок

```
GATE LR 4,BLOK2
```

направляет транзакты в блок с именем BLOK2, если в момент их поступления ЛП с номером 4 включен.

Блоки, работающие с памятью

Для хранения в памяти отдельных числовых значений и массивов таких значений используются сохраняемые величины и матрицы сохраняемых величин.

Сохраняемые величины могут использоваться в модели для хранения исходных данных, которые надо изменять при различных прогонах модели, промежуточных значений и результатов моделирования. В начале моделирования все сохраняемые величины устанавливаются равными 0. Для установки отличных от 0 начальных значений сохраняемых величин используется оператор INITIAL, имеющий следующий формат:

```
INITIAL X$ имя, значение;
```

```
INITIAL Xj, значение.
```

Здесь имя и j - соответственно имя и номер сохраняемой величины, а значение - присваиваемое ей начальное значение (константа).

Для изменения сохраняемых величин в процессе моделирования служит блок SAVEVALUE (сохранить величину), имеющий следующий формат:

```
имя SAVEVALUE A,B.
```

В поле А указывается номер или имя сохраняемой величины, в которую записывается значение операнда В. Если в поле А после имени (номера) сохраняемой величины стоит знак + или -, то значение операнда В добавляется или вычитается из текущего содержимого сохраняемой величины. Например:

```
SAVEVALUE 5,Q$LINE  
SAVEVALUE NREF+1.
```

Управляющие операторы GPSS

Для управления прогоном модели используются управляющие операторы GPSS. С одним из них - оператором START - мы уже сталкивались при рассмотрении блока TERMINATE. Оператор START (начать) имеет следующий формат:

```
START A,B,C,D.
```

Поле А содержит константу, задающую начальное значение счетчика завершений. В поле В может быть записано ключевое слово NP - признак подавления формирования стандартного отчета по завершении моделирования. Если поле В пусто, то по окончании прогона модели формируется отчет со стандартной статистической информацией о всех объектах модели. Поле С не используется и сохранено для совместимости со старыми версиями GPSS. Поле D может содержать 1 для включения в отчет списков текущих и будущих событий. Если поле D пусто, то выдача в отчет содержимого этих списков не производится.

Оператор SIMULATE (моделировать) устанавливает предел реального времени, отводимого на прогон модели. Если прогон не завершится до истечения этого времени, то он будет прерван принудительно с выдачей накопленной статистики в отчет.

Оператор SIMULATE имеет единственный операнд А, содержащий предельное время моделирования в минутах, задаваемое константой. Оператор размещается перед оператором START, начинающим лимитированный прогон.

Оператор RESET (сбросить) сбрасывает всю статистическую информацию, накопленную в процессе прогона модели. При этом состояние аппаратных, динамических и запоминающих объектов, а также генераторов случайных чисел сохраняется, и моделирование может быть возобновлено с повторным сбором статистики. Оператор не имеет операндов.

Получение и интерпретация стандартного отчета

По завершении прогона модели раздается звуковой сигнал, и в строке состояния появляются сообщения

Writing REPORT.GPS Simulation Complete Reporting Complete
сигнализирующие о том, что моделирование закончено и в данный момент производится создание отчета о прогоне модели. Затем система переходит в состояние ожидания дальнейших команд.

Отчет, создаваемый по завершении моделирования, записывается в файл со стандартным именем REPORT.GPS. Это имя может быть изменено командой REPORT (создать отчет), имеющий следующий формат:

REPORT A,B.

В поле A указывается спецификация файла, в который должен быть выведен отчет. Если поле B содержит ключевое слово NOW, то отчет создается немедленно после ввода команды.

Необходимо иметь в виду, что отчет, создаваемый автоматически по завершении прогона модели или командой REPORT, является неформатированным, т.е. непригодным для непосредственного просмотра. Для форматирования и создания стандартного отчета GPSS необходимо завершить сеанс и выполнить программу форматирования отчета. Выход из интегрированной среды (завершение сеанса) производится путем ввода управляющего оператора END (закончить). При этом производится выход в MS DOS или в программу-оболочку Norton Commander.

Для форматирования отчета необходимо загрузить модуль форматирования GPSSREPT.EXE. После его загрузки на экране появляется "заставка" с названием модуля, двумя окнами в нижней части экрана и сообщениями-подсказками. В левом окне выведено имя файла, в котором находится неформатированный отчет (по умолчанию - это файл REPORT.GPS). В правом окне выведено обозначение устройства, куда должен быть выведен форматированный отчет (по умолчанию - это экран дисплея SCRN). Форматированный отчет может быть также выведен на печать или на диск. Для этого в правое окно надо ввести обозначение PRN или имя файла на диске соответственно. Для переключения окон используется клавиша Enter. Для создания отчета на выбранном

устройстве следует нажать клавишу Пробел, для выхода из программы клавишу Esc.
Если содержимое окон по умолчанию оставлено без изменения, то после нажатия клавиши Пробел на экране появляется отчет о последнем прогоне модели, выполненном перед завершением сеанса работы с модулем GPSSPC.EXE. Отчет содержит следующую информацию:

- общие сведения о модели и ее прогоне, включающие модельное время начала (START_TIME) и конца (END_TIME) прогона, количество блоков в модели (BLOCKS), количество устройств (FACILITIES), объем памяти, оставшейся свободной при прогоне (STORAGES), количество объектов модели, включающие для модели (FREE_MEMORY);
- сведения об именах объектов модели, включающие для каждого имени идентификатор (NAME), присвоенное ему числовое значение (VALUE) и тип имени: 0, если числовое имя является именем пользователя с помощью оператора EQU; 1, если числовое значение имени присвоено системой; 2, если - сведения о блоках модели, включающие для каждого блока номер строки исходной программы (LINE), номер или имя блока (LOC), название блока (BLOCK_TYPE), количество транзактов, прошедших через блок (ENTRY_COUNT), текущее количество транзактов в блоке в момент завершения моделирования (CURRENT_COUNT), количество транзактов, заблокированных перед блоком в момент завершения моделирования (RETRY0);
- сведения об устройствах модели, включающие для каждого устройства его имя или номер (FACILITY), количество занятий устройства (ENTRIES), коэффициент использования (UTIL), среднее время на одно занятие (AVE_TIME) и ряд других данных;
- сведения о многоканальных устройствах модели, включающие для каждого MKU его имя или номер (STORAGE), емкость (CAP), количество свободных каналов в момент завершения моделирования (REMAIN), наименьшее (MIN) и наибольшее (MAX) количество занятых каналов в процессе моделирования, количество занятий MKU (ENTRIES), среднее количество занятых каналов (AVE.C), коэффициент использования (UTIL) и ряд других данных;

сведения об очередях модели, включающие для каждой очереди ее имя или номер (QUEUE), максимальную длину очереди в процессе моделирования (MAX), текущую длину очереди в момент завершения моделирования (CONT), общее количество транзактов, вошедших в очередь в процессе моделирования (ENTRIES), и количество "нулевых" входов в очередь (ENTRIES(0)), среднюю длину очереди (AVE.CONT), среднее время ожидания в очереди с учетом всех транзактов (AVE.TIME) и без учета "нулевых" входов (AVE.(-0));

сведения о статистических таблицах модели, включающие для каждой таблицы ее имя или номер (TABLE), среднее значение (MEAN) и среднеквадратическое отклонение (STD.DEV) табулируемой величины, границы частотных интервалов (RANGE), частоты (FREQUENCY) и накопленные частоты в процентах (CUM.%) попадания наблюдений в эти интервалы.

Разумеется, сведения об объектах того или иного типа появляются в отчете только в том случае, если в модели присутствует хотя бы один объект данного типа.

Модель СМО с неограниченной очередью

```
Time1 VARIABLE 10
Time2 VARIABLE V$Time1-5
GENERATE (exponential(1,0,V$Time1))
QUEUE Buff
SEIZE Proc
DEPART Buff
ADVANCE (exponential(2,0,V$Time2))
RELEASE Proc
TERMINATE 1
START 10000
```

Рис.5.4. Модель СМО с неограниченной очередью

START TIME	END TIME	BLOCKS	FACILITIES	STORAGE
0.000	101150.393	7	1	0

LABEL	LOC	BLOCK TYPE	ENTRY COUNT	CURRENT COUNT	RETRY	FACILITY							
						UTIL.	AVE. TIME	AVAIL.	OWNER	PEND	INTER	RETRY	DELAY
1	1	GENERATE	10000	0	0	0.500	5.057	1	0	0	0	0	0
2	1	QUEUE	10000	0	0								
3	1	SEIZE	10000	0	0								
4	1	DEPART	10000	0	0								
5	1	ADVANCE	10000	0	0								
6	1	RELEASE	10000	0	0								
7	1	TERMINATE	10000	0	0								

Рис. 5.5. Отчет по результатам моделирования.

Модель СМО с ограниченной очередью

```
Time1 VARIABLE 10
Time2 VARIABLE V$Time1-5
QLen VARIABLE 5
GENERATE (exponential(1,0,V$Time1))
TEST L Q$Buff,VSQLen,out
QUEUE Buff
SEIZE Proc
DEPART Buff
ADVANCE (exponential(2,0,V$Time2))
RELEASE Proc
out TERMINATE 1
START 10000
```

Рис. 5.6. Модель СМО с ограниченной очередью

5.2. Симуляция систем в среде AnyLogic

AnyLogic — программное обеспечение для имитационного моделирования, разработанное российской компанией TheAnyLogicCompany (бывшая «Экс ДжейТекнолоджис», англ. XJ Technologies). Инструмент обладает современным графическим интерфейсом и позволяет использовать язык Java для разработки моделей [36-39].

AnyLogic включает в себя графический язык моделирования, а также позволяет пользователю расширять созданные модели с помощью языка Java. Интеграция компилятора Java в AnyLogic предоставляет более широкие возможности при создании моделей, а также создание Java апплетов, которые могут быть открыты любым браузером. Эти апплеты позволяют легко размещать модели AnyLogic на веб-сайтах.

В дополнение к Java-апплетам, AnyLogicProfessional поддерживает создание Java-приложений, в этом случае пользователь может запустить модель без инсталляции AnyLogic.

Графическая среда моделирования AnyLogic включает в себя следующие элементы:

Stock&FlowDiagrams (диаграмма потоков и накопителей) применяется при разработке моделей, используя метод системной динамики.

Statecharts (карты состояний) в основном используется в агентных моделях для определения поведения агентов. Но также часто используется в дискретно-событийном моделировании, например, для симуляции машинных сбоев.

Actioncharts (блок-схемы) используется для построения алгоритмов. Применяется в дискретно-событийном моделировании (маршрутизация звонков) и агентном моделировании (для логики решений агента).

Processflowcharts (процессные диаграммы) основная конструкция, используемая для определения процессов в дискретно-событийном моделировании.

Среда моделирования также включает в себя: низкоуровневые конструкции моделирования (переменные, уравнения, параметры, события и т.п.), формы представления (линии, квадраты, овалы и т.п.), элементы анализа (базы данных, гистограммы, графики), стандартные картинки и формы экспериментов.

Среда моделирования AnyLogic поддерживает проектирование, разработку, документирование модели, выполнение компьютерных экспериментов с моделью, включая различные виды анализа — от анализа чувствительности до оптимизации параметров модели относительно некоторого критерия.

AnyLogic включает в себя набор следующих стандартных библиотек:

ProcessModelingLibrary разработана для поддержки дискретно-событийного моделирования в таких областях как Производство, Цепи поставок, Логистика и Здравоохранение. Используя ProcessModelingLibrary, вы можете смоделировать системы реального мира с точки зрения заявок (англ. entity) (сделок, клиентов, продуктов, транспортных средств, и т. д.), процессов (последовательности операций, очередей, задержек), и ресурсов. Процессы определены в форме блочной диаграммы. ProcessModelingLibrary используется в AnyLogic 7 вместе с библиотекой EnterpriseLibrary, использовавшейся в тех же целях в AnyLogic 6.

PedestrianLibrary создана для моделирования пешеходных потоков в «физической» окружающей среде. Это позволяет Вам создавать модели с большим количеством пешеходного трафика (как станции метро, проверки безопасности, улицы и т.д.). Модели поддерживают учёт статистики плотности движения в различных областях. Это гарантирует приемлемую работу пунктов обслуживания с ограничениями по загруженности, оценивает длину простаивания в определённых областях, и обнаруживает потенциальные проблемы с внутренней геометрией — такие как эффект добавления слишком большого числа препятствий — и другими явлениями. В моделях, созданных с помощью PedestrianLibrary, пешеходы движутся непрерывно, реагируя на различные виды препятствий (стены, различные виды областей) так же, как и обычные пешеходы. Пешеходы моделируются как взаимодействующие агенты со сложным поведением. Для быстрого описания потоков пешеходов PedestrianLibrary обеспечивает высокоуровневый интерфейс в виде блочной диаграммы.

RailYardLibrary поддерживает моделирование, имитацию и визуализацию операций сортировочной станции любой сложности и масштаба. Модели сортировочной станции могут использовать

комбинированные методы моделирования (дискретно-событийное и агентное моделирование), связанные с действиями при транспортировке: погрузками и разгрузками, распределением ресурсов, обслуживанием, различными бизнес-процессами.

Графическая среда моделирования AnyLogic поддерживает проектирование, разработку, документирование модели, выполнение компьютерных экспериментов с моделью, различные виды анализа.

Однако разработка реально полезных моделей всегда требует использования в той или иной степени программного кода. Необходимость программирования при разработке моделей является одной из основных проблем, затрудняющих освоение имитационного моделирования. Это требование является свойством всех инструментов моделирования: в любом из них для разработки серьезной модели требуется использование встроенного в этот симулятор языка программирования – либо специализированного языка, либо одного из универсальных языков программирования, с которым интегрирован симулятор.

В AnyLogic базовым языком, совмещенным со средой разработки моделей, является Java, один из наиболее мощных и в то же время простых языков программирования. Включение программного кода на Java в модель AnyLogic является органичным и естественным, тексты языка интегрируются в модель чрезвычайно просто и элегантно. Если разработчик глубоко владеет приемами программирования, то он может с успехом использовать при построении модели всю мощь языка. Однако для построения даже сложных моделей на AnyLogic от разработчика требуются лишь самые общие знания о программировании.

AnyLogic – это инструмент визуальной разработки моделей и визуального представления результатов моделирования. Использование визуализации при имитационном моделировании систем трудно переоценить. Наибольший эффект – вплоть до эффекта присутствия – дает анимированное представление поведения системы и ее частей в виде некоторой формы виртуальной реальности.

Для создания имитационной модели сети передачи запускаем программу AnyLogic. Далее нажав на кнопку *Файл*, выбираем пункт *Модель* в меню *Создать*. В появившемся окошке вводим имя модели, ее местоположение. Нажав на кнопку *Далее* выбираем

способ создания модели: с нуля, либо используя шаблон. Выбираем *Использовать шаблон модели* → *Дискретно-событийное моделирование*.

5.3. Проведение оптимизационных экспериментов в среде AnyLogic

Если Вам нужно изучить поведение модели при каких-то заданных условиях или улучшить производительность модели, найдя значения параметров, при которых достигается наилучший результат работы модели, то Вы можете воспользоваться возможностью оптимизации модели AnyLogic. Оптимизация модели AnyLogic заключается в последовательном выполнении нескольких прогонов модели с различными значениями параметров и нахождении оптимальных для данной задачи значений параметров.

В AnyLogic встроен оптимизатор OptQuest – лучший из предлагаемых сегодня оптимизаторов. Оптимизатор OptQuest автоматически находит лучшие значения параметров и учитывает заданные ограничения. AnyLogic предоставляет удобный графический интерфейс для конфигурирования и отслеживания хода оптимизации.

Оптимизация состоит из нескольких последовательных прогонов модели с различными значениями параметров. Комбинируя эвристики, нейронные сети и математическую оптимизацию, OptQuest позволяет находить значения параметров модели, соответствующие максимуму или минимуму целевой функции, как в условиях неопределенности, так и при наличии ограничений.

OptQuest является торговой маркой компании OptTekSystems, Inc. Более подробную информацию об исполняющем модуле OptQuest Вы можете найти на сайте OptTek.

Чтобы оптимизировать модель

- Создайте оптимизационный эксперимент.
- Задайте целевой функционал (функцию, которую нужно минимизировать или максимизировать).
- Задайте оптимизационные параметры (параметры, значения которых будут меняться).
- Задайте ограничения, которые будут наложены на значения параметров и переменных. (опционально).

- Задайте условия остановки прогона.
- Задайте условия остановки оптимизации.
- Запустите оптимизационный эксперимент.

Процесс оптимизации представляет собой итеративный процесс, который состоит в том, что:

Оптимизатор OptQuest выбирает допустимые значения оптимизационных параметров и запускает модель с этими значениями.

Завершив «прогон» модели, OptQuest вычисляет значение целевой функции на момент завершения.

Оптимизатор анализирует полученное значение, изменяет значения оптимизационных параметров в соответствии с алгоритмом оптимизации и процесс повторяется заново.

После того, как Вы зададите все настройки оптимизационного эксперимента, Вы можете запустить оптимизацию Вашей модели.

Процесс оптимизации, моделируемой Вами системы включает в себя следующие шаги:

Вы запускаете оптимизацию.

Запустив оптимизацию, Вы сможете изучать состояние системы и следить за тем, как оптимизатор решает оптимизационную задачу, предлагая более оптимальные значения целевой функции и оптимизационных параметров.

После того, как оптимизатор закончит оптимизацию модели, Вам будут выданы полученные оптимальное значение целевой функции и те значения параметров, при которых это значение было достигнуто. Если Вы планируете в дальнейшем использовать полученные оптимальные значения в Вашей имитационной модели, то Вы можете скопировать эти значения в другой эксперимент (обычно это простой эксперимент) этой модели, щелкнув по кнопке копировать в окне презентации эксперимента.

Чтобы запустить оптимизационный эксперимент

В панели Проект, щелкните правой кнопкой мыши по эксперименту, который Вы хотите запустить, и выберите Запуск из контекстного меню.

Либо выберите Модель|Запуск из главного меню или щелкните по кнопке со стрелкой справа от кнопки панели инструментов Запуск и выберите эксперимент, который Вы хотите запустить, из выпадающего списка.

При запуске эксперимента AnyLogic автоматически производит построение запускаемой модели. Поэтому в случае обнаружения ошибки Вам будет показано сообщение об ошибке (ошибках), а более подробная информация будет выведена в панель Консоль.

Чтобы запустить последний запущенный эксперимент, щелкните по кнопке панели инструментов Запуск или нажмите F5.

Запустив эксперимент, Вы увидите отдельное окно - окно презентации. В нем будет отображен интерфейс, созданный Вами для запущенного оптимизационного эксперимента.

Постановка задачи №1:

$$\min \sum_{i=1}^n V t_{3i},$$

$$\sum_{i=1}^n V_i \leq V_{\Sigma},$$

где $V t_i$ - средняя задержка запроса в i -ом устройстве обслуживания delay (УО); V_i - производительность i -го УО и V_{Σ} - общая производительность всех УО.

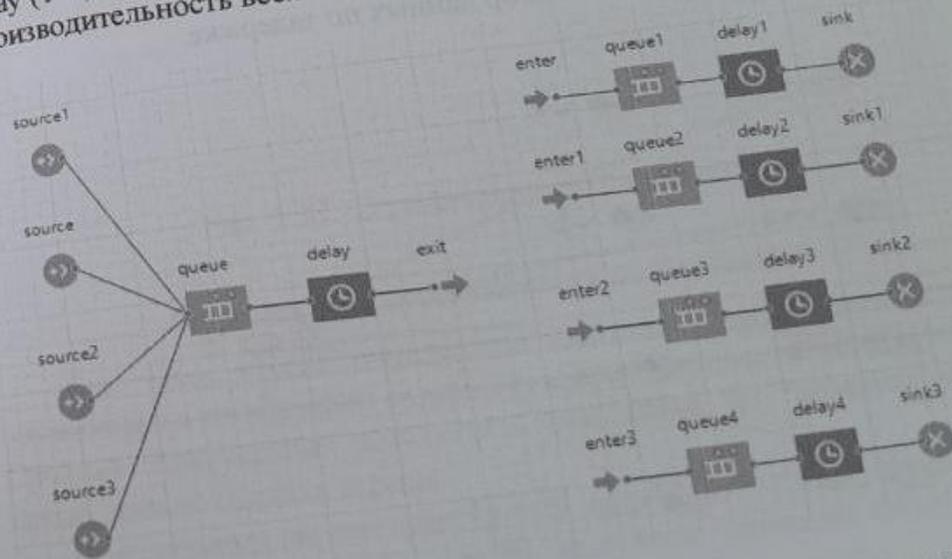


Рис. 5.7. Схема модели оптимизации по времени задержки

На рисунке 5.7 представлена модель оптимизации где в качестве параметров оптимизации выступают временные характеристики обработки заявок. Устройства обслуживания должны быть задействованы таким образом, чтобы минимизировать количество простаивающих заявок в системе.

Для этих целей дополнительно вводятся данные гистограмм, которые занимают сбором временных данных, как показано на рисунке 5.8.

Имя: Отображать имя Исключить На верхнем уровне

Тип: Класс заявки:

Пакет:

Действие при входе^D

```
t_zad1.add(entity.vixod_1-entity.vxod_1);
Vt1=t_zad1.mean();
```

Рисунок 5.8. Сбор данных по задержке

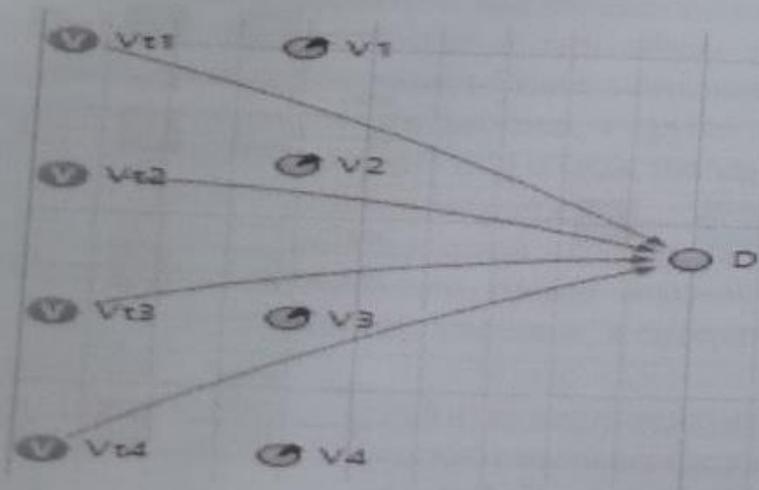


Рис. 5.9. Параметры и переменные модели оптимизации

Имя: Отображать имя

Массив Внешняя Константа

$D = Vt1 + Vt2 + Vt3 + Vt4$

Рис. 5.10. Целевая функция оптимизации по минимуму времени задержки обработки заявок

Параметры	Тип	Значение		Шаг	Начальное
		Мин.	Макс.		
V1	дискретный	1	100	1	
V2	дискретный	1	100	1	
V3	дискретный	1	100	1	
V4	дискретный	1	100	1	

Рис. 5.11. Значения параметров оптимизации

Ограничения, накладываемые на параметры (проверяются перед запуском):

Вкл.	Выражение	Тип	Граница
<input checked="" type="checkbox"/>	$V1 + V2 + V3 + V4$	=	200.0

Рис. 5.12. Ограничения оптимизации

Определение эффективности оптимизации

Для того, чтобы определить насколько эффективен метод оптимизации по времени задержки, следует провести эксперимент. Сравнив полученные результаты симуляции до оптимизации и после, можно сделать вывод.

При входных параметрах: $\lambda_1 = 10, \lambda_2 = 40, \lambda_3 = 20, \lambda_4 = 50, \lambda_k = 200$ и $V_{вч1} = V_{вч2} = V_{вч3} = V_{вч4} = 50$, получаем значения следующий результат (рис. 5.13).

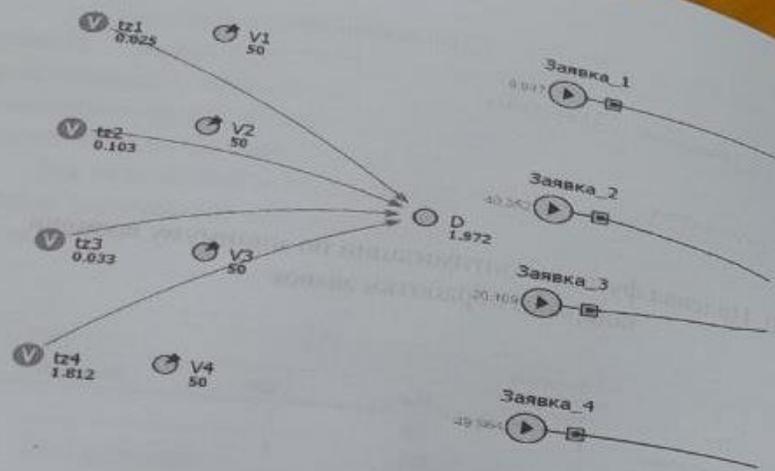


Рис. 5.13. Результат симуляции без оптимизации

Далее следует оптимизирующий эксперимент. Результаты эксперимента представлены на рисунке 5.14.

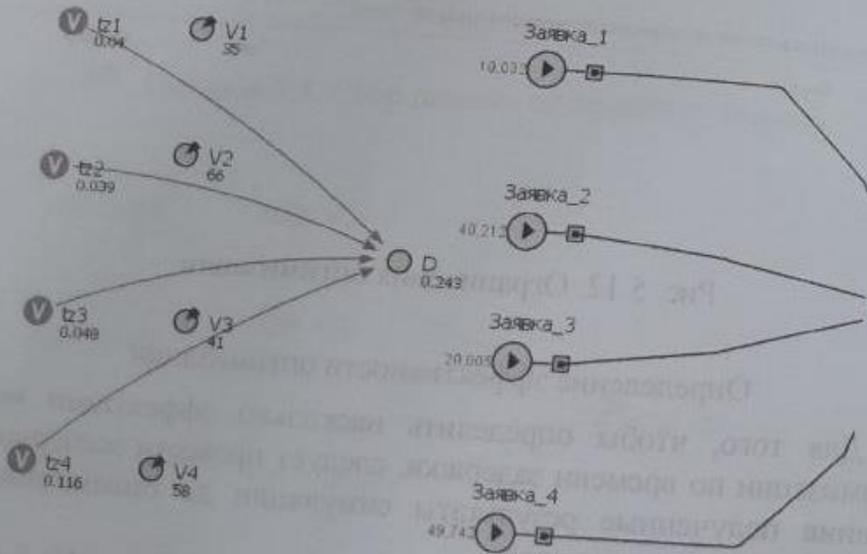


Рисунок 5.14. Результаты оптимизации

Постановка задачи №2:

$$\min \sum_{i=1}^n Q_i,$$

$$\sum_{i=1}^n V_i \leq V_{\Sigma},$$

где Q_i — длина очереди i-го УО;

V_{Σ} — общая производительность всех УО;

V_i — производительность i-го УО.

Ниже представлены свойства заявок.

Имя: Заявка_1

Тип: Source<T extends Entity>

Пакет: om.x.anylogic.libranes.enterprise

Заявки прибывают согласно

Время между прибытиями: Интенсивности Времени между прибытиями

Количество заявок, прибывающих за один раз: exponential (10)

Ограниченное количество прибытий: 1

Новая заявка: new MyClass()

Действие при выходе: entity.ad_1 = 1;

Класс заявки: MyClass

Отображать имя Исключить На верхнем уровне На 1

Рис.5.15. Свойства первого узла заявки

Говоря о queue, имеется в виду буфер вычислительного узла и его буфер представляет из себя вместимость вычислительного узла и его способность к построению очередей из заявок в системе (рис. 5.16).

Имя: queue1

Тип: Queue<T extends Entity>

Пакет: om.x.anylogic.libranes.enterprise

Класс заявки: MyClass

Максимальная вместимость:

Действие при входе:

Действие при подходе к выходу:

Действие при выходе:

```
entity.vход_1 = time();
Q1=queue1.size();
t_oj1d1.add(time()-entity.vход_1);
```

Рис.5.16. Свойство queue1

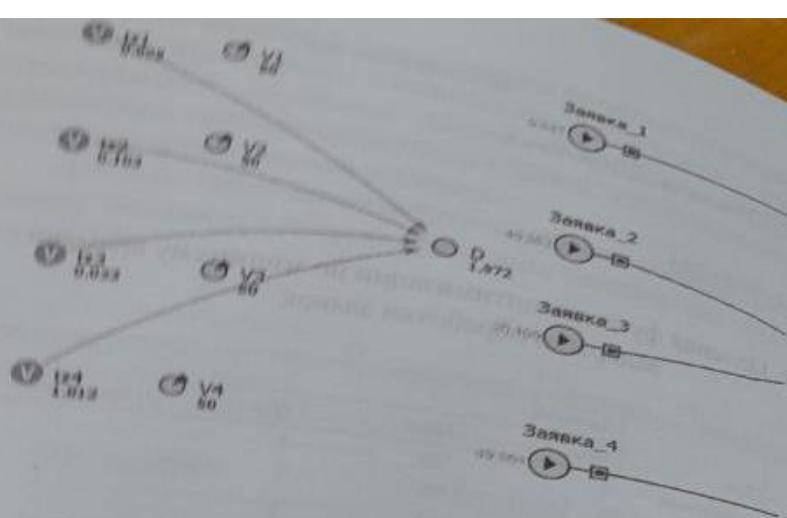


Рис. 5.13. Результат симуляции без оптимизации

Далее следует оптимизирующий эксперимент. Результаты эксперимента представлены на рисунке 5.14.

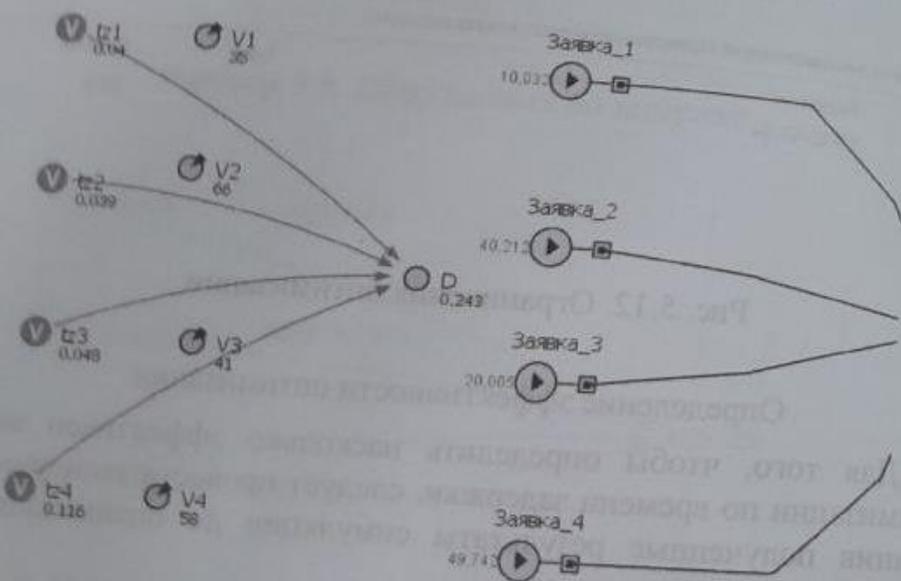


Рисунок 5.14. Результаты оптимизации

Постановка задачи №2:

$$\min \sum_{i=1}^n Q_i,$$

$$\sum_{i=1}^n V_i \leq V_{\Sigma},$$

где Q_i — длина очереди i -го УО; V_i — производительность i -го УО; V_{Σ} — общая производительность всех УО.
Ниже представлены свойства заявок.

Имя: Заявка_1 Отображать имя Исключить На верхнем уровне На 1

Тип: Source<T extends Entity>

Пакет: om.x.anylogic.librales.enterprise

Заявки прибывают согласно

Время между прибытиями¹

Количество заявок, прибывающих за один раз²

Ограниченное количество прибытий

Новая заявка³

Действие при выходе⁴

Интенсивности Время между прибытиями

exponential(10)

1

new MyClass()

entity.ad_1 = 1;

Рис.5.15. Свойства первого узла заявки

Говоря о queue, имеется в виду буфер вычислительного узла. Буфер представляет из себя вместимость вычислительного узла и его способность к построению очередей из заявок в системе (рис. 5.16).

Имя: queue1 Отображать имя Исключить На верхнем уровне

Тип: Queue<T extends Entity>

Пакет: om.x.anylogic.librales.enterprise

Максимальная вместимость

Действие при входе¹

Действие при подходе к выходу²

Действие при выходе³

entity.vход_1 = time();

Q1=queue1.size();

τ_oj1d1.add(time()-entity.vход_1);

Рис.5.16. Свойство queue1

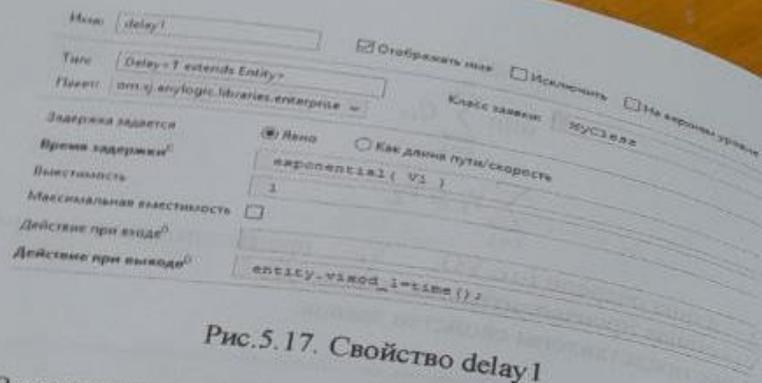


Рис.5.17. Свойство delay1

В результате оптимизации к минимуму целевой функции D , где $f(D) = (Q_1 + Q_2 + Q_3 + Q_4)$, с ограничением оптимизации $(V_1 + V_2 + V_3 + V_4) \leq 200$;

где $Q_1 - Q_4$, параметры целевой функции, которые динамически изменяются для достижения соответствия с ограничением оптимизации и $V_1 - V_4$, динамические переменные, которые получают от длины очереди каждого из узлов, и они вписываются, как вычислительные ресурсы каждого из узлов. Параметры и переменные показаны на рисунке 5.18.

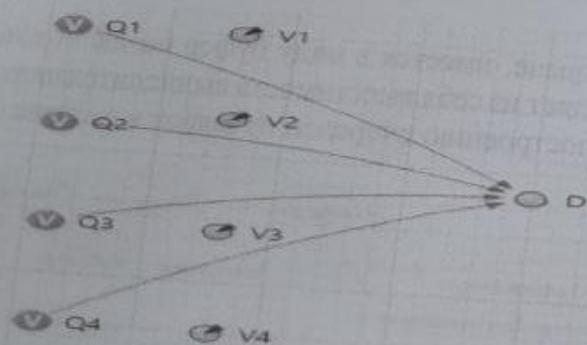


Рис.5.18. Параметры и переменные модели оптимизации

Тем самым получается добиться оптимизации заданной функции, путем изменения длины очереди. На рисунке 5.19 можно

увидеть результаты оптимизации по длине очереди, при разных исходных значениях интенсивности поступления заявок.

Dist : Optimization

Optimization Summary

Iteration	Current	Best
Objective	100	28
Parameters	3,088	1,146
V1	71	98
V2	52	13
V3	49	48
V4	23	42

Copy the best solution to the clipboard.

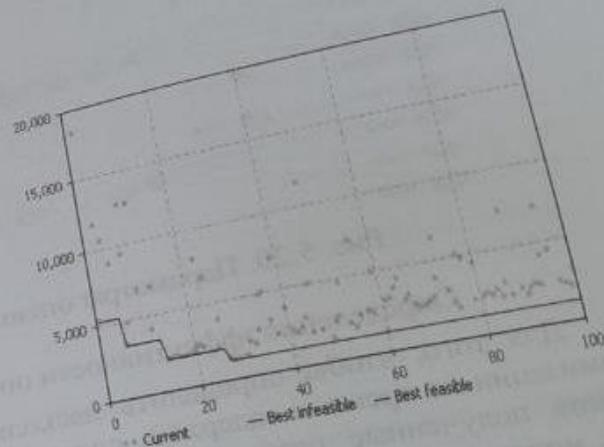


Рис.5.19. Оптимизационный эксперимент по длине очереди

В рис.5.19 iteration означает количество прогонов. Колонка current означает данные за текущий прогон, а колонка best означает лучший результат.

Постановка задачи №3:

$$\min \sum_{i=1}^n k_1 \frac{P_{пот i}}{N} + k_2 Q_i + k_3 v_{t_{3i}}$$

$$\sum_{i=1}^n V_i \leq V_{\Sigma},$$

где коэффициенты важности показателей $k_1 + k_2 + k_3 = 1$, Q_i - длина очереди i-го УО, $v_{t_{3i}}$ - время задержки обработки заявки в i-ом УО, $P_{пот i}$ - вероятность потерь запроса в i-ом УО, V_i - производительность i-го УО и V_{Σ} - общая производительность всех УО.

Параметры оптимизации приведены на рис.5.20.

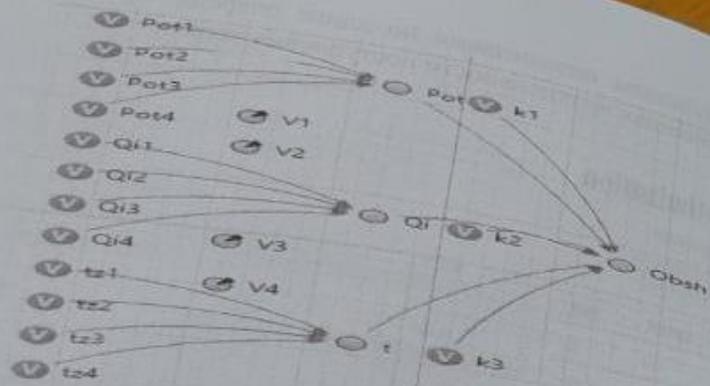


Рис. 5.20. Параметры оптимизации

Определение эффективности оптимизации

Для того, чтобы определить насколько эффективен метод оптимизации по времени задержки, следует провести эксперимент. Сравнив полученные результаты симуляции до оптимизации и после, можно сделать вывод.

При входных параметрах: $\lambda_1 = 100$, $\lambda_2 = 20$, $\lambda_3 = 130$, $\lambda_4 = 50$, $\lambda_k = 120$ и $V_{вч1} = V_{вч2} = V_{вч3} = V_{вч4} = 50$, получаем значения следующий результат (рис. 5.21).

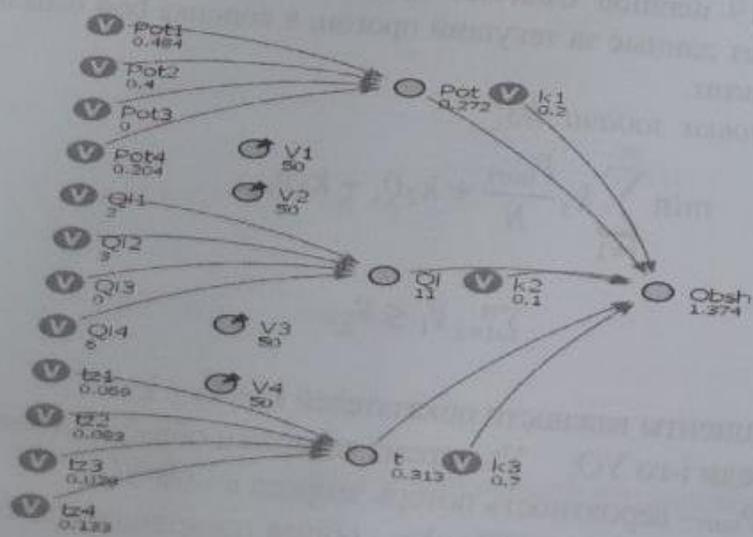


Рис. 5.21. Результат симуляции без оптимизации

Далее следует оптимизирующий эксперимент. Результаты эксперимента представлены на рисунке 5.22.

Dis1 : Optimization

Optimization Experiment

	Current	Best
Iterations	100	32
Objective	0.333	0.693

Parameters	Value
V1	50
V2	34
V3	52
V4	36

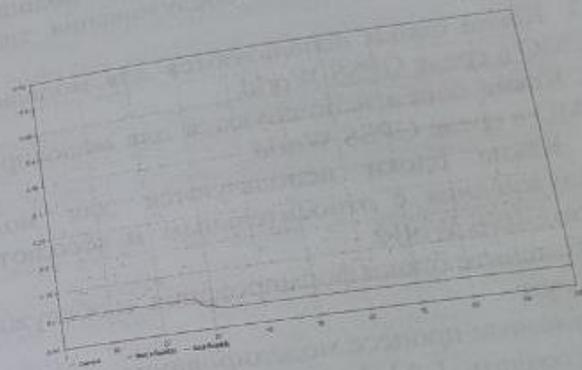


Рис. 5.22. Результат оптимизации

После того, как в симуляцию подставим значения оптимизации, получаем следующий результат (рис. 5.23).

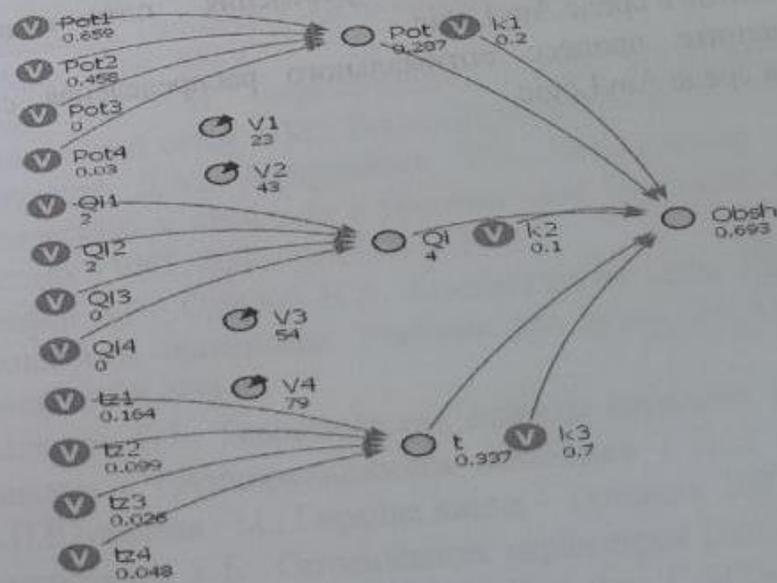


Рис. 5.23. Оптимальные значения параметров

КОНТРОЛЬНЫЕ ВОПРОСЫ ПО ГЛАВЕ 5.

1. Объясните процесс создания транзакций в среде GPSS World.
2. Объясните процесс обслуживания транзакций в среде GPSS World.
3. Какие блоки используются для моделирования одноканальной СМО в среде GPSS World.
4. Какие блоки используются для моделирования многоканальной СМО в среде GPSS World.
5. Какие блоки используются для моделирования процесса обслуживания с относительным и абсолютным приоритетами в среде GPSS World.
6. Объясните блоки формирования и обслуживания потоков в среде AnyLogic.
7. Объясните процесс моделирования сети в среде AnyLogic.
8. Как создать LAVA - классы в среде AnyLogic?
9. Как реализовать условные переходы в среде AnyLogic?
10. Как отобразить гистограмму случайных чисел в среде AnyLogic?
11. Как настроить параметры моделирования в среде AnyLogic?
12. Как организовано приоритетное обслуживание в среде AnyLogic?
13. Объясните процедуры проведения оптимизационных экспериментов в среде AnyLogic.
14. Объясните процесс оптимального распределения сетевых ресурсов в среде AnyLogic.

СПИСОК ЛИТЕРАТУРЫ

1. Димарский Я.С. Методы оптимизации сетей связи. – СПбГУТ, 2005. – 124 с.
2. Wehrle K., Gunes M. Modeling and Tools for Network Simulation. - Springer-Verlag Berlin Heidelberg, 2010. - p.537.
3. Guizani M., Rayes A. Network Modeling and Simulation. - John Wiley & Sons, Ltd, 2010. - p.273.
4. Michel C. Jeruchim Simulation of Communication Systems. - New York, Kluwer Academic Publishers, 2002.
5. Рекомендация ITU-T Y.2011 «Принципы и эталонная модель NGN»
6. Кох Р., Г.Г. Яновский Г. Г. Эволюция и конвергенция в электросвязи. М.: Радио и связь, 2001
7. Евсеева О.Ю. Обеспечение гарантированного качества обслуживания в сетях NGN с использованием оценок конечных пользователей. – Радиотехника, 2008, вып.155
8. Клейнрок Л. Теория массового обслуживания», перевод с английского – М.: Машиностроение, 1979.
9. Клейнрок Л. Вычислительные сети с очередями, перевод с английского – М.: Мир, 1979.
10. Дьяконов В.П. MATLAB 6.5 SP1/7 + Simulink 5/6 в математике и моделировании – М.: СОЛОН-Пресс, 2005. – 576 с.
11. Бертсекас Д., Галлагер Р. Сети передачи данных: Пер. с англ. - М.: Мир, 1989. - 544 с.
12. Вишневский В.М. Теоретические основы проектирования компьютерных сетей. – М.: Техносфера, 2003. - 512 с.
13. Абдуллаев Д.А., Амирсаидов У.Б. Комплексная модель физического и канального уровней сети передачи данных. - Вестник ТУИТ, 2007, №4, с.19 -23.
14. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: Учебник для вузов. 3-э изд. - СПб. Питер, 2006. -958 с.
15. Мелентев О.Г. Теоретические аспекты передачи данных по каналам с группирующимися ошибками / Под ред. проф. В.П.Шувалова. - М.: Горячая линия – Телеком, 2007. – 232 с.
16. Амирсаидов У.Б. Оптимизация параметров систем передачи данных средствами МАТЛАБ. «Вестник ТУИТ». Ташкент - 2010 г. №2. - С.50-54.

17. Алиев Т.И. Основы моделирования дискретных систем. - СПб: СПбГУ ИТМО, 2009. -363с.
18. Шелухин О.И., Тенякшев А.М., Осин А.В. Фрактальные процессы в телекоммуникациях. - М.: Радиотехника, 2003. - 479 с.
19. Амирсаидов У.Б. Оценка достоверности и потери пакетов в сетях передачи данных.- «Вестник ТУИТ». Ташкент – 2009 г. №2. - С.73-77
20. Амирсаидов У.Б. Модели оценки качества обслуживания в сетях телекоммуникаций. Монография. -Ташкент: "Алокачи", 2020. - 171 с.
21. Яновский, Г.Г. Качество обслуживания в сетях IP. / Г.Г. Яновский. // Вестник связи. №1, 2008. - С. 65-74.
22. Лемешко А.В., Ватти М., Симоненко А.В. Управление очередями на узлах активной сети. Радиотехника: Всеукр. межвед. науч.-техн. сб. 2007. Вып. 151. С. 92-97. <http://openarchive.nure.ua/handle/document/2847>
23. Лемешко А.В., Симоненко А.В. Математическая модель динамического управления канальными и буферными ресурсами на узлах телекоммуникационной сети. Радиотехника: Всеукр. межвед. науч.-техн. сб. 2009. Вып. 156. С. 36-41. <http://openarchive.nure.ua/handle/document/2608>
24. Али С. Али, Симоненко А.В. Потокковая модель динамической балансировки очередей в MPLS- сети с поддержкой traffic engineering queues. Электронное научно специализированное издание – журнал «Проблемы телекоммуникаций», № 1(1), 2010. С.60-67.
25. Семеняка М.В. Двухуровневый метод иерархически-координационного обслуживания очередей на узлах телекоммуникационной сети. Научно-технический вестник информационных технологий, механики и оптики. 2014, №4 (92). С.98-105.
26. Симоненко А.В., Андрушко Д.В. Математическая модель управления очередями на маршрутизаторах телекоммуникационной сети на основе оптимального агрегирования потоков и распределения пакетов по очередям. Электронное научно специализированное издание – журнал «Проблемы телекоммуникаций», № 1(16), 2015. С.94-102. (<http://pt.journal.kh.ua>).

27. Назаров А.Н., Сычев К.И. Модели и методы исследования процессов функционирования узлов коммутации сетей связи следующего поколения при произвольных распределениях поступления и обслуживания заявок различных классов качества. Т-Сомм, №7, 2012. С.135-140.
28. Маршрутизация и обслуживания заявок различных классов на узлах телекоммуникационной сети / Стерин В.Л., Вавенко Т.В., Эферов Д.М. // Вестник НТУ «ХПИ». Серия: Новы рішення в сучасних технологях. - Х: НТУ «ХПИ», - 2013. -№1 (977). -С.45-49
29. Вавенко Т.В. Потокковая модель адаптивной маршрутизации с балансировкой нагрузки для программно-конфигурируемых сетей. Первая Международная научно-практическая конференция «Проблемы инфокоммуникаций. Наука и технологии». Харьков, Украина 9-11 октября 2013 г. - С. 161-164.
30. Вавенко Т.В., Стерин В.Л., Симоненко А.В. Потокковая модель маршрутизации с балансировкой нагрузки по длине очереди в программно-конфигурируемых сетях. Научно-технический вестник информационных технологий, механики и оптики. 2013, №4(86). С.38-45.
31. Лемешко А. В., Вавенко Т. В. Анализ решений задач однопутевой и многопутевой маршрутизации многопоточкового трафика в телекоммуникационных сетях // Системы обробки інформації. - Вип. 8(98). - 2011. - С. 224-228.
32. Лемешко А.В., Вавенко Т.В. Усовершенствование потокковой модели многопутевой маршрутизации на основе балансировки нагрузки [Электронный ресурс] // Проблемы телекоммуникаций. - 2012. - № 1 (6). - С. 12 - 29.
33. Рекомендация МСЭ-Т У.3001 (2011 г.). Будущие сети: целевые установки и цели проектирования. Женева 2012, 26 с.
34. Рекомендация МСЭ-Т У.3011 (2012 г.), Структура виртуализации сети для будущих сетей. Женева, 2013 г., 28 с.
35. Амирсаидов У.Б., Усманова Н.Б. Моделирование и реализация виртуальных наложенных сервисных сетей. Монография. - Ташкент: "Алокачи". 2019. - 180 с.
36. Боев В.Д. Исследование адекватности GPSS WORLQ и ANYLOGIC при моделировании дискретно-событийных

процессов. - Санкт-Петербург: Военная Академия Связи, 2011. - 404 с.

37. Амирсaidов У.Б. Моделирование и симуляция сетей передачи данных: Методические указания к курсовому проектированию. - Ташкент, ТУИТ, 2015. - 41 с.
38. Амирсaidов У.Б. Моделирование и симуляция сетей передачи данных: Методические указания к практическим занятиям. - Ташкент, ТУИТ, 2016. - 58 с.
39. Amirsaidov U.B., Sadatdiyov K.E. Aloqa tizimlarini modellashtirish va simulyatsiyalash: Amaliy mashg'ulotlarni bajarish bo'yicha uslubiy ko'rsatmalar. - Tomkent, TATU, 2018. - 90 b.

ОГЛАВЛЕНИЕ

1.	ВВЕДЕНИЕ	3
	ОСНОВЫ МОДЕЛИРОВАНИЯ СЕТЕЙ	5
	ТЕЛЕКОММУНИКАЦИИ	5
1.1.	Основные задачи моделирования	7
1.2.	Типы и этапы моделирования	10
1.3.	Концептуальная модель сетей телекоммуникации	17
2.	МОДЕЛИ СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ	17
2.1.	Структура, параметры и характеристики систем массового обслуживания	17
2.2.	Модели одноканальных систем массового обслуживания	25
2.3.	Модели многоканальных систем массового обслуживания	37
2.4.	Модели систем массового обслуживания с приоритетами	42
2.5.	Модель сети массового обслуживания с приоритетами	48
3.	МАТЕМАТИЧЕСКИЕ ОСНОВЫ ОПТИМИЗАЦИИ СЕТЕЙ	54
3.1.	Методы оптимизации	54
3.2.	Функции Matlab optimization toolbox	57
3.3.	Задачи оптимизации сетей телекоммуникации	62
4.	ОПТИМИЗАЦИЯ СИСТЕМ И СЕТЕЙ	69
	ТЕЛЕКОММУНИКАЦИИ	69
4.1.	Модель и расчет характеристик систем передачи данных	75
4.2.	Оптимизация систем передачи данных	85
4.3.	Модель и расчет характеристик маршрутизатора	90
4.4.	Модель и расчет характеристик сети	95
4.5.	Оптимизация процессов обслуживания пакетов	105
4.6.	Оптимизация процессов маршрутизации пакетов	115
4.7.	Оптимальное встраивание виртуальных сетей	134
5.	СИМУЛЯЦИЯ И ОПТИМИЗАЦИЯ СЕТЕЙ С ПОМОЩЬЮ СПЕЦИАЛЬНЫХ ПРОГРАММНЫХ СРЕДСТВ	134
5.1.	Симуляция систем в среде GPSS World	148
5.2.	Симуляция систем в среде AnyLogic	15
5.3.	Проведение оптимизационных экспериментов в среде Anylogic	16
	СПИСОК ЛИТЕРАТУРЫ	

АМИРСАИДОВ У.Б.

ОПТИМИЗАЦИЯ СЕТЕЙ

Учебное пособие

Toshkent - "METODIST NASHRIYOTI" - 2024

Muharrir: Bakirov Nurmuhammad

Texnik muharrir: Tashatov Farrux

Musahhih: Muhammadiyeva Sevinch

Dizayner: Ochilova Zarnigor

Bosishga 10.05.2024. da ruxsat etildi.

Bichimi 60x90. "Times New Roman" garniturasida.

Ofset bosma usulida bosildi.

Shartli bosma tabog'i 11. Nashr bosma tabog'i 10,5.

Adadi 300 nusxa.

"METODIST NASHRIYOTI" MCHJ matbaa bo'limida chop etildi.

Manzil: Toshkent shahri, Shota Rustaveli 2-vagon tor ko'chasi, 1-uy.



+99893 552-11-21

Nashriyot roziligisiz chop etish ta'qiqlanadi.

ISBN 978-9910-03-160-1



9 789910 031601