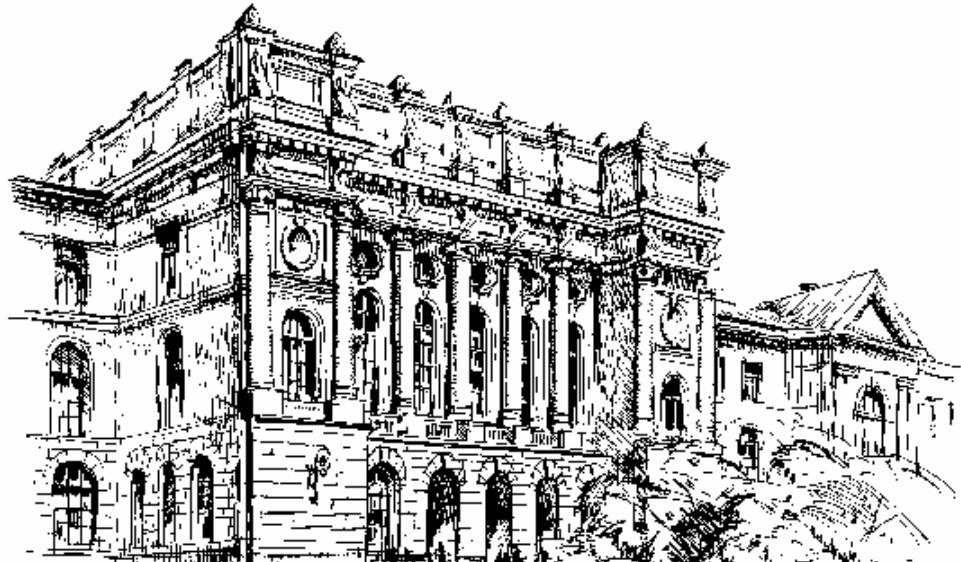


**Д.Н. Колесников, Е.Н. Бендерская, В.Н. Козлов,
А.В. Лупин, В.И. Пахомова, А.Г. Сиднев, В.Н. Цыган**

**Системный анализ
и
принятие решений**

Учебное пособие



**Санкт-Петербург
2008**

Федеральное агентство по образованию

Санкт-Петербургский государственный
политехнический университет

Д.Н. Колесников, Е.Н. Бендерская, В.Н. Козлов,
А.В. Лупин, В.И. Пахомова, А.Г. Сиднев, В.Н. Цыган

**Системный анализ
и
принятие решений**

Учебное пособие

Под редакцией
доктора технических наук Д.Н. Колесникова

*Рекомендовано Учебно-методическим объединением
по университетскому политехническому образованию в качестве
учебного пособия для студентов высших учебных заведений,
обучающихся по направлениям подготовки и специальностям
техники и технологии*

Санкт-Петербург
2008

УДК 303.732
ББК 22.18я73
С 409

Системный анализ и принятие решений. Учебное пособие.

/ Под ред. д.т.н. Д.Н. Колесникова. – СПб.: Изд-во Политехн. ун-та, 2008. – 471 с.: ил.

ISBN

Рецензенты:

доктор физ.-мат. наук, профессор РГГМУ А.В. Белоцерковский;
кандидат технических наук, доцент СПбГПУ С.В. Хлопин.

Авторы:

Е.Н. Бендерская, В.Н. Козлов, Д.Н. Колесников, А.В. Лупин,
В.И. Пахомова, А.Г. Сиднев, В.Н. Цыган

Пособие соответствует государственному образовательному стандарту направлений подготовки и специальностей в области техники и технологии и содержанию учебной программы дисциплины «Системный анализ и принятие решений» направления 220100 «Системный анализ и управление», а также дисциплин «Методы оптимизации» и «Теория принятия решений» направления 2301100 «Информатика и вычислительная техника».

В пособии с позиций системного анализа рассматриваются основные подходы к построению и анализу оптимационных моделей принятия решений и представлению систем в форме моделей массового обслуживания. Описаны эффективные методы оптимизации в задачах непрерывного и дискретного программирования. Приводятся примеры задач оптимизации с использованием известного пакета MATLAB. Значительная часть материала отводится изложению принципов построения формальных моделей систем с использованием теории массового обслуживания, а также аналитических и имитационных методов анализа их характеристик. Представленный материал иллюстрируется подробными примерами.

Предназначено для студентов высших учебных заведений, обучающихся по направлениям «Системный анализ и управление», «Информатика и вычислительная техника». Пособие также может быть рекомендовано студентам, обучающимся по направлению «Автоматизация и управление».

Печатается по решению редакционно-издательского совета Санкт-Петербургского государственного политехнического университета.

© Е.Н. Бендерская, В.Н. Козлов, Д.Н. Колесников, А.В. Лупин,
В.И. Пахомова, А.Г. Сиднев, В.Н. Цыган, текст, 2008

© Санкт-Петербургский политехнический государственный
университет, 2008

ISBN 978-5-7422-2098-5

ОГЛАВЛЕНИЕ

Посвящение	9
Введение	11
1. Основные понятия и определения.....	13
1.1. Принципы системного анализа. Системный анализ и исследование операций	13
1.2. Терминология операционного исследования	14
1.3. Принципы принятия решений в задачах исследования операций. Классификация задач	16
1.4. Неопределенность целей. Выбор решения по многим критериям	17
1.4.1. Выделение главного критерия	17
1.4.2. Методы формирования свёртки критериев	18
1.4.3. Введение метрики в пространстве целевых функций	19
1.4.4. Метод максиминной свёртки	20
1.4.5. Метод последовательных уступок	20
1.4.6. Компромиссы Парето.	21
1.5. Экспертные методы принятия решений	24
1.5.1. Проведение экспертного опроса	26
1.5.2. Обработка результатов опроса	29
1.5.3. Формирование окончательного решения	33
1.6. Принятие решения в условиях неопределённости и риска	34
1.6.1. Принятие решений в условиях неопределённости	34
1.6.2. Принятие решений в условиях риска	38
1.7. Принятие решения в условиях конфликта	39
1.8. Примеры построения операционных моделей.	45
Задачи.....	53
2. Задачи линейного программирования	61
2.1. Задача линейного программирования и приведение ее к канонической форме	61
2.2. Содержание алгоритма симплекс-метода	62
2.3. Поиск допустимого базиса	64
2.4. Табличная форма алгоритма симплекс-метода	66
2.5. Понятие двойственности в линейном программировании. Двойственный симплекс метод	72
2.6. Примеры решения задач линейного программирования	80

3. Задачи нелинейного программирования при отсутствии ограничений	89
3.1. Основные положения	89
3.2. Классификация методов безусловной оптимизации	91
3.3. Методы выбора направлений	92
3.4. Методы выбора длины шага	99
3.5. Сравнительный анализ методов	102
3.6. Пример решения задачи безусловной оптимизации	104
4. Задачи нелинейного программирования при наличии ограничений	113
4.1. Основные положения	113
4.2. Классификация методов условной оптимизации	119
4.3. Использование необходимых условий оптимальности	121
4.4. Метод проекции градиента.....	136
4.5. Метод возможных направлений Зойтендейка	145
4.6. Сведение задач условной оптимизации к задачам безусловной оптимизации. Метод штрафных и барьерных функций ...	151
4.7. Сравнительный анализ методов условной оптимизации	158
5. Задачи дискретного программирования	161
5.1. Содержание метода ветвей и границ	161
5.2. Метод ветвей и границ в задачах "о коммивояжере" и "о назначении"	164
5.3. Метод ветвей и границ в задачах распределения ресурсов....	168
Задачи	175
6. Задачи динамического программирования	181
6.1. Содержание метода динамического программирования	181
6.2. Задачи оптимизации функций на графах	182
6.3. Динамическое программирование в задачах поиска неисправностей	185
6.4. Динамическое программирование в задачах распределения ресурсов.....	188
Задачи.....	191
7. Решение задач оптимизации с использованием библиотеки OPTIM системы MATLAB 5	200
7.1. Состав и назначение функций библиотеки OPTIM	201
7.2. Демонстрационные примеры	202

7.2.1. Задачи безусловной оптимизации (<i>unconstrained optimization</i>). Функция <i>fminu</i> и вектор параметров оптимизации <i>options</i>	202
7.2.2. Задачи условной оптимизации с ограничениями типа \geq , \leq без ограничений, на пределы изменения (<i>constrained problem</i>). Функция <i>constr</i>	207
7.2.3. Задачи условной оптимизации с ограничениями на область допустимых значений X (<i>bounded example</i>)	208
7.2.4. Задачи условной оптимизации с ограничениями типа " $=$ " (<i>equality constrained example</i>)	210
7.2.5. Использование вектора параметров оптимизации <i>options</i> : изменение установок по умолчанию (<i>changing the default settings</i>).....	211
7.3. Описание основных функций для решения задач оптимизации	214
7.4. Пример создания подсистемы моделей для решения задач оптимизации	228
7.4.1. Решение задач линейного и квадратичного программирования (программа «Lqr»)	229
7.4.2. Решение задач безусловной оптимизации программа « <i>Unconmin</i> »)	232
7.4.3. Решение задач условной оптимизации программа « <i>Conmin</i> »)	238
8. Теория массового обслуживания, основные понятия и определения	242
Основные обозначения.....	242
8.1. Классификация систем массового обслуживания (СМО) ..	243
8.2. Потоки событий и их свойства	245
8.2.1. Классификация потоков	245
8.2.2. Простейший поток и его свойства	247
8.2.3. Потоки Пальма. Рекуррентные потоки.....	250
8.3. Общие результаты теории СМО	252
8.4. Примеры формализации задач в терминах СМО	253
Задачи	255
9. Марковские системы массового обслуживания	258
9.1. Марковские случайные процессы и их классификация	258
9.2. Дискретные марковские последовательности	259
9.2.1. Эргодические однородные марковские цепи	260
9.2.2. Неэргодические цепи с поглощающими состояниями ...	261

9.3. Общие уравнения непрерывного марковского процесса	265
9.4. Процесс гибели и размножения	275
9.5. Метод частичного укрупнения моделей при использовании марковских процессов	278
9.6. Основные соотношения для простейших систем массового обслуживания	281
10. Немарковские системы массового обслуживания	291
10.1. Классификация методов исследования немарковских СМО...	291
10.1.1. Метод фаз.....	291
10.1.2. Метод введения избыточной переменной	292
10.1.3. Метод вложенных марковских цепей	293
10.1.4. Использование полумарковских процессов	293
10.1.5. Интегральный метод	296
10.2. Характеристики простейших систем массового обслуживания типа M/D/1, M/E _K /1, M/G/1, G/M/1	298
10.3. Приближенные оценки характеристик СМО	303
11. Системы массового обслуживания с неоднородным потоком ..	306
11.1. Одноканальные СМО с бесприоритетным обслуживанием ...	306
11.2. Одноканальные СМО с приоритетами	307
11.2.1. Относительный приоритет	307
11.2.2. Абсолютный приоритет	308
11.3. Одноканальные СМО со смешанными приоритетами	312
Задачи	320
12. Стохастические сетевые модели СМО	333
12.1. Предпосылки и цели применения аппарата сетей СМО	333
12.2. Однородные экспоненциальные сети СМО	335
12.2.1. Разомкнутые сети СМО. Анализ их характеристик.....	335
12.2.2. Замкнутые сети СМО. Анализ их характеристик	344
12.2.3. Определение основных показателей качества обслуживания в узлах замкнутой сети СМО	350
12.2.4. Расчёт нормирующей константы однородной замкнутой сети СМО	353
12.2.5. Расчётные соотношения, используемые для определения характеристик узлов однородной замкнутой сети СМО	357
12.2.6. Метод анализа средних значений характеристик узлов однородной замкнутой сети СМО	359

12.3. Неоднородные сети СМО	361
12.3.1. Общие сведения о неоднородных сетях СМО	361
12.3.2. Разомкнутые неоднородные сети СМО	362
12.3.3. Замкнутые неоднородные сети СМО	367
Задачи	390
13. Имитационное моделирование	406
13.1. Понятие об имитационном моделировании	406
13.2. Событийный подход к построению моделирующих алгоритмов	410
13.3. Общие принципы имитации случайных факторов	415
13.3.1. Моделирование дискретных случайных величин	418
13.3.2. Моделирование непрерывных скалярных случайных величин	420
13.3.3. Моделирование случайных векторов по заданным многомерным распределениям	428
13.3.4. Моделирование марковских процессов	431
13.3.5. Моделирование потоков	435
13.4. Построение оценок и доверительных интервалов характеристик случайных величин	435
13.4.1. Оценки характеристик случайных величин	436
13.4.2. Вычисление доверительных интервалов	440
13.5. Сравнительный анализ систем имитационного моделирования	444
Библиографический список.....	456
Приложения	460
Приложение 1. Учебные задания по линейному программированию	461
Приложение 2. Учебные задания по нелинейному программированию	463
Приложение 3. Построение матриц A_i, B_i, C_i для регулярного Марковского графа состояний типа «решетка».....	466



Дмитрий Николаевич Колесников
(1938 – 2005)

Посвящается памяти Дмитрия Николаевича Колесникова

В основе предлагаемой вниманию читателей книги лежит курс лекций «Системный анализ и принятие решений», который на протяжении ряда лет читал профессор Санкт-Петербургского государственного политехнического университета Дмитрий Николаевич Колесников на факультете технической кибернетики. Эта книга, задуманная, структурно организованная и в значительной части написанная им, была завершена учениками и коллегами уже после его кончины, и посвящается памяти этого замечательного человека.

Говорят, что обстоятельства формируют характер человека. Это справедливо лишь отчасти — некоторые склоняются под ударами судьбы, а есть люди, которые наперекор трудностям сами строят свою собственную жизнь. Именно таким человеком был Дмитрий Николаевич.

Родился он 20 апреля 1938 года в большой дружной семье главного инженера текстильной фабрики. Отец, мать, тетя, бабушки, дедушки — все жили вместе. Отличительную особенность этой семьи — интеллигентность — Дмитрий Николаевич в полной мере впитал еще ребенком.

Всю блокаду от самого ее начала и до снятия Дмитрий Николаевич пережил в Ленинграде. На его глазах умерли все ближайшие родственники. С блокадных времен Дмитрий Николаевич фактически воспитывался тетей.

Учился Дмитрий Николаевич легко и блестяще — в 1955 году с золотой медалью он окончил школу, а в 1961 с красным дипломом ленинградский военно-механический институт. Затем работа в НИИ.

1963 год — поступление в аспирантуру на кафедру автоматики и телемеханики ленинградского политехнического института (ЛПИ).

1966 год — успешная защита кандидатской диссертации.

Молодой ученый сразу находит приложение своего таланта в новейших космических разработках. Вместе со своим руководителем, профессором Евгением Ивановичем Юрьевичем, он стоит у истоков создания центрального научно-исследовательского института робототехники (ЦНИИ РТК). Участие Дмитрия Николаевича в таких ответственных работах, как исследование надежности фотонных систем мягкой посадки, систем автоматическойстыковки и высотомеров космических аппаратов, сформировали стиль и методы его научной деятельности.

В 1970 году уже со стажем практических разработок Дмитрий Николаевич получает должность доцента кафедры автоматики и телемеханики ЛПИ и скоро становится в один ряд с лучшими преподавателями кафедры. Благодаря своему научному авторитету молодой ученый организует со-

трудничество кафедры с такими ведущими организациями ВПК страны, как НПО «Ленинец» и «ОКБ Антонова», возглавляя направление, связанное с исследованием, разработкой и оценкой надежности бортовых радионавигационных комплексов самолетов.

1993 г. — защита докторской диссертацию на тему "Основы теории проектирования систем функционального диагностирования пилотажно-навигационного оборудования самолетов на основе структурной и функциональной избыточности".

С 1994 г. — профессор, действительный член Всесоюзной ассоциации "Диагностика и отказоустойчивость в технике".

У Дмитрия Николаевича много учеников. Все студенты кафедры осваивали его дисциплину «Системный анализ и принятие решений», концептуально формирующую мышление будущих инженеров. Важное место в учебном плане кафедры занимает основанный им курс «Функциональная диагностика систем управления». Многие нынешние доценты и профессора, сделавшие успешную карьеру, являются его воспитанниками — им подготовлено 7 докторантов.

Его интересы были широки и разнообразны. Он хорошо знал и любил искусство: живопись, музыку. Классическую музыку хорошо понимал. Любил петь. Часто посещал музеи, выставки. Имел спортивные разряды по шахматам и волейболу. Много путешествовал с рюкзаком по стране, бывал за рубежом. Дмитрий Николаевич был для нас величайшим авторитетом во многих областях, и поэтому существовал некоторый «вектор дистанции» от нас к нему, однако встречный вектор был нулевым.

У него было много верных друзей со школьной скамьи и институтских времен, среди коллег ученых и преподавателей. Все они высоко ценили его удивительную искренность и исключительную порядочность. Он был примерным семьянином и воспитал сына, который также окончил ЛПИ и защитил кандидатскую диссертацию.

19 января 2005 года Дмитрий Николаевич скоропостижно скончался — не выдержало сердце, которое, принимая на себя любые жизненные невзгоды и трудности, давало ему удивительную способность всегда сохранять спокойствие и доброжелательность. Остались незавершенными научные проекты, аспиранты лишились своего наставника, кафедра — руководителя цикла важнейших дисциплин.

Прошло уже более трех лет со дня безвременной кончины Дмитрия Николаевича — казалось бы, время лечит — но чувство утраты не покидает всех, кто его знал. Авторы книги надеются, что её издание послужит доброй памяти нашего друга, коллеги и учителя.

ВВЕДЕНИЕ

Проблема принятия решений имеет в жизни человека особое значение, так как любой вид деятельности — это, в конечном счете, цепочка принятия решений. В обыденной жизни выбор одной из альтернатив действия не требует привлечения научных методов — люди обходятся опытом, традиционными навыками, интуицией. При создании же сложных технических систем, анализе экономических проблем, управлении экологическими системами, принятии решений в чрезвычайных ситуациях и др. у человека, принимающего решение, уже нет уверенности в том, что выбор его правильный и эффективный. В таких случаях возникает необходимость в научных методах принятия решений. На современном этапе развития эти методы сложились в отдельную дисциплину — теорию принятия решений, опирающуюся на широкое использование ЭВМ, а система моделей и подходов превратилась в сложную развитую систему, которая получила название "системного анализа".

Сегодня системный анализ — это обширная синтетическая дисциплина, включающая в себя целый ряд разделов, носящих характер самостоятельных научных дисциплин.

Главное внимание в книге удалено той части вопросов, где формальные методы и подходы являются основными в общей проблеме принятия решений. В соответствии с этим и определено содержание книги, материал которой поделен на две части.

Первая часть, посвященная методике построения и анализа оптимизационных моделей, состоит из семи разделов.

Первый раздел дает основные определения и понятия системного анализа применительно к методике построения оптимизационных моделей.

В разделах 2 – 6 рассматриваются способы разрешения построенных моделей в зависимости от их класса (линейные, нелинейные, дискретные и др.).

Раздел 7 посвящен вопросам использования системы MATLAB для решения оптимизационных задач.

В приложениях приводятся учебные задания по линейному и нелинейному программированию, ориентированные на использование предложенных методов.

Во второй части книги всестороннему рассмотрению подвергаются модели и методы теории массового обслуживания. Уделяется внимание изложению основ аппарата теории массового обслуживания и наиболее распространенных способов его применения.

С этой целью даются формулировки основных задач, осуществляется вывод соответствующих систем дифференциальных или алгебраических

уравнений, получение главных формул в результате решения этих уравнений, а также приводятся примеры, иллюстрирующие использование теории массового обслуживания для решения прикладных задач.

Вторая часть книги, состоящая из перечисленных ниже шести разделов, отражает стремление авторов проиллюстрировать принципы формализации и конструктивного математического описания процессов поведения систем различного назначения в терминах и понятиях массового обслуживания.

Восьмой раздел посвящен основным понятиям и определениям, а также классификации систем массового обслуживания.

В девятом разделе рассмотрено применение случайных марковских процессов для анализа показателей качества систем массового обслуживания.

Десятый раздел посвящен немарковским системам массового обслуживания и сравнительному анализу методов их исследования.

В одиннадцатом разделе рассматриваются системы массового обслуживания с неоднородным потоком, при этом исследуется их поведение, как при бесприоритетном обслуживании, так и при наличии относительных, абсолютных и смешанных приоритетов.

В двенадцатом разделе представлены основные результаты практического применения теории сетей массового обслуживания в решении прикладных задач.

Тринадцатый раздел посвящен имитационному подходу к моделированию систем массового обслуживания. В нем также приводится сравнительный анализ известных систем имитационного моделирования.

Существенное внимание уделяется практическим вопросам оптимизации и моделирования систем. Первая часть книги содержит много примеров и задач, направленных на развитие умения и навыков в формализации и анализе оптимизационных моделей на основе современных математических подходов и методов. Вторая часть также сопровождается значительным числом заданий и подробно рассмотренных примеров по построению и практическому использованию моделей на основе теории массового обслуживания.

Для полного освоения представленного материала читателю желательно располагать определенными знаниями в области математического анализа, линейной и матричной алгебры, теории вероятностей и математической статистики, объем которых не выходит за рамки стандартных вузовских курсов.

1. ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ

1.1. ПРИНЦИПЫ СИСТЕМНОГО АНАЛИЗА. СИСТЕМНЫЙ АНАЛИЗ И ИССЛЕДОВАНИЕ ОПЕРАЦИЙ

Дисциплина, именуемая «системный анализ», родилась в силу необходимости вести исследования междисциплинарного характера. Поэтому можно принять за основу следующее определение:

системный анализ представляет собой научное направление, связанное с исследованием слабоструктурированных сложных проблем междисциплинарного характера.

Он включает в себя и создание сложных технических систем, проектирование больших технологических, энергетических, гидротехнических, информационных комплексов, анализ экономических и экологических ситуаций, и многие другие направления инженерной, научной и хозяйственной деятельности. Решение этих задач требует объединения усилий специалистов разных научных профилей, унификации и согласования информации, получаемой в результате исследований конкретного характера.

Системный анализ является необходимой методической базой для проведения системных исследований. Предметом системных исследований могут быть различного рода объекты, процессы, явления, а также связанные с ними проблемы. В рамках системных исследований решаются два основных класса задач.

1. Задачи анализа объекта исследования.
2. Задачи синтеза наилучшего в определенном смысле варианта решения проблемы.

Системный анализ как научное направление базируется на четырех основополагающих принципах: системности, иерархичности, интегральности и формализма [12, 14].

Принцип системности требует всестороннего, многоаспектного, целостного, целеориентированного подхода к исследуемому предмету, исходя из возможного набора сфер его существенного проявления и назначения.

Принцип иерархичности требует многоуровневого рассмотрения предмета исследований.

Принцип интегральности требует выявления, в первую очередь, общих свойств предмета исследований, дающих о нем целостное представление. Этот принцип базируется на фундаментальном свойстве систем, которое может быть сформулировано следующим образом: свойства системы *не есть* простая сумма свойств, составляющих ее частей.

В силу этого нельзя получить адекватного представления о системе, изучая по отдельности ее компоненты.

Принцип формализма требует построения формальных моделей исследуемого объекта, позволяющего получить конструктивные результаты и сделать необходимые позитивные выводы. Однако этот принцип не исключает возможности использования неформальных методов и процедур.

Результатом системных исследований является, как правило, выбор вполне определенной альтернативы (решения). Таким образом, системный анализ — это дисциплина, занимающаяся проблемами принятия решений в условиях, когда выбор альтернативы требует анализа сложной информации различной физической природы.

Поэтому истоки системного анализа, его методических концепций лежат в тех дисциплинах, которые занимаются проблемами принятия решений. Одной из основных дисциплин такого рода является исследование операций. Термин "операция" достаточно общий, он означает любое целенаправленное действие.

Исследование операций нельзя считать дисциплиной чисто математической, хотя она широко использует математические методы и породила целый ряд направлений прикладной математики.

Главным же содержанием дисциплины являются сложные проблемы принятия решений, при изучении которых неформальные методы, представления здравого смысла и способы описания — математическая формализация, — играют не меньшую роль, чем формальный математический аппарат.

1.2. ТЕРМИНОЛОГИЯ ОПЕРАЦИОННОГО ИССЛЕДОВАНИЯ

Введем основные понятия и определения, необходимые для операционного исследования [11, 13].

Операция — совокупность взаимно согласованных действий, направленных на достижение определенной цели.

Примеры операций: выпуск новой продукции в заданные сроки, повышение производительности вычислительной системы при обработке данных в реальном масштабе времени и др.

Оперирующая сторона — отдельные лица и коллективы, объединенные организационным руководством и активно стремящиеся к достижению поставленной цели.

Активные средства проведения операции — совокупность материальных, энергетических и других ресурсов, а также организационных воз-

можностей, используемых оперирующей стороной для обеспечения успешного хода операций и достижения цели.

Стратегии оперирующей стороны — допустимые способы использования активных средств. "Допустимые" следует понимать как не выходящие за пределы технических, организационных и физических возможностей (ограничений).

Факторы операции — объективные условия и обстоятельства, определяющие ее особенности и влияющие на исход операции. Различают обычно факторы операции: определенные (контролируемые, управляемые) и неопределенные, имеющие вероятностную природу, обычно неуправляемые.

Эффективность операции — отражает соответствие между результатом предпринимаемых действий и целью операции. Для количественной оценки выбирается скалярный или векторный показатель эффективности.

Любое операционное исследование включает следующие этапы.

1. Построение математической модели операции — формальных соотношений, устанавливающих связь принятой системы показателей эффективности $f(x, z)$ с действующими факторами операции и стратегиями оперирующей стороны:

$$f(x, z), \quad x \in X, \quad z \in Z,$$

где x — решение, управление, связанное с выбором стратегии оперирующей стороны из допустимого множества X , z — неконтролируемые, неуправляемые факторы операций, принадлежащие множеству Z .

2. Формулировка оптимизационной задачи выбора решений $x \in X$, т.е. выбора стратегии, наилучшей в некотором смысле с точки зрения принятой системы показателей эффективности $f(x, z)$.

3. Решение возникающей оптимизационной задачи. Данный этап исследования операции можно отнести собственно к математике.

С точки зрения математики — это обычные задачи математического программирования, которые в зависимости от вида ограничений и области допустимых решений распадаются на определенные классы: линейное программирование, нелинейное программирование, целочисленное программирование, дискретное, динамическое и др. Для решения каждой из указанных задач существуют разнообразные, хорошо изученные алгоритмы оптимизации [5, 6, 7, 9]. При этом следует отметить, что теория решения экстремальных задач возникла и развивалась именно благодаря потребностям исследования операций. Заметим также, что ряд математических задач оптимизации, к которым сводятся исследуемые операции, не содержат неопределенностей. Но задачи, не содержащие неопределенностей, являются скорее исключением, чем правилом.

В исследовании операций принято различать три типа неопределенностей.

1. Неопределенность целей.

2. Неопределенность наших знаний об окружающей природе, внешней среде.

3. Неопределенность действий реального противника или партнера.

Рассмотрим, что необходимо исследователю операций для анализа соответствующих задач средствами математики.

1.3. ПРИНЦИПЫ ПРИНЯТИЯ РЕШЕНИЙ В ЗАДАЧАХ ИССЛЕДОВАНИЯ ОПЕРАЦИЙ. КЛАССИФИКАЦИЯ ЗАДАЧ

Рассмотрим существенные элементы процесса принятия решений.

1. Необходимость определения одной или нескольких целей, которые должны быть достигнуты в результате проведения операции.

2. Наличие лица, принимающего решение (ЛПР), ответственного за последствия этих решений.

3. Возможные влияния на исход решения совокупности неуправляемых факторов, объединяемых понятием «внешняя среда».

4. Существование альтернативных решений, определяющих различные варианты достижения целей.

5. Определение и оценка последствий возможных решений.

6. Следование правилам выбора решений (решающие правила), определяющим наиболее предпочтительное решение.

В соответствии с выделенными элементами различают следующие задачи процесса принятия решений.

1. По количеству целей: одноцелевые, многоцелевые (многокритериальные).

2. По виду ЛПР: индивидуальный выбор (решение формирует и принимает отдельный субъект), групповой выбор (в качестве ЛПР участвует группа экспертов или консультантов по конкретной проблеме).

3. По наличию информации о внешней среде (ВС):

– полная определённость (известны все параметры и характеристики и их взаимосвязи и поэтому возможно построение формальных математических моделей и разрешение их на основе методов оптимизации);

– полная неопределенность (отсутствует полная информация о факторах и характеристиках ВС); для таких задач характерна малая достоверность, сложные взаимосвязи между факторами; основную роль в принятии решения играют неформальные методы);

– вероятностная неопределенность — наличие риска (известны статистические данные о факторах, влияющих на процесс принятия решений).

Решение таких задач формируется на базе теории статистических решений).

Далее рассмотрим основные особенности процесса принятия решений для выделенного класса задач.

1.4. НЕОПРЕДЕЛЁННОСТЬ ЦЕЛЕЙ. ВЫБОР РЕШЕНИЯ ПО МНОГИМ КРИТЕРИЯМ

Назначение цели и выбор показателя эффективности операции, т. е. формализация цели всегда является трудной проблемой. Достаточно просто решается вопрос, когда удаётся цель операции отобразить одним скалярным показателем эффективности.

Если критериев много:

$$f_1(x) \rightarrow \max, f_2(x) \rightarrow \max, \dots, f_i(x) \rightarrow \max, \dots, f_r(x) \rightarrow \max, \quad (1.1)$$

а ресурс для их достижения находится в одних руках, то необходимо сформировать единую цель.

В соответствии с соотношением (1.1) считаем, что в дальнейшем все поставленные цели согласованы с таким выбором критерия

$$f_i(x), i = \overline{1, r}, \text{ что его необходимо максимизировать при условии } x \in X.$$

Одновременно также примем, что данная постановка задачи относится к условиям полной определённости относительно внешней среды, т. е. множество $Z = \emptyset$.

Указанная многоцелевая задача принятия решений типична при разработках любой сложной управляемой системы, при экономическом обосновании крупного технического проекта, решении сложных экономических и экологических проблем.

Остановимся на некоторых наиболее употребительных способах преодоления неопределенности целей и рассмотрим далее случай, когда перед исследователем стоит задача выбора решения (вектора x), обеспечивающего максимальное значение функциям $f_1(x), f_2(x), \dots, f_i(x), \dots, f_r(x)$ одновременно.

1.4.1. Выделение главного критерия

Предположим, что введена некоторая система показателей $f_i^*(x)$, относительно которых $f_i(x)$ должен удовлетворять ограничениям:

$$f_i(x) \geq f_i^*(x), \quad (i = \overline{1, r}). \quad (1.2)$$

Среди показателей $f_i(x)$ выделим некоторый основной $f_k(x)$, тогда переходим к однокритериальной задаче:

$$\max_{x \in X} f_k(x), \quad (1.3)$$

при ограничениях (1.2), где $i \neq k$.

Такая схема редукции многокритериальной задачи к однокритериальной является, вероятно, самой простой и наиболее употребительной в практике. При этом следует заметить следующее [23]:

- переход от исходной задачи (1.1) к задаче (1.3) не есть переход от одной эквивалентной задачи к другой;
- не всегда ясен алгоритм определения границ $f_i^*(x)$, ($i = \overline{1, r}$);
- возможно наличие нескольких «главных» критериев, находящихся в противоречии друг с другом.

1.4.2. Методы формирования свёртки критериев

В случае использования аддитивных преобразований свёртка выглядит следующим образом:

$$F(x) = \sum_{i=1}^r b_i f_i(x), \quad (1.4)$$

где $b_i(x)$ — коэффициент важности соответствующего критерия $f_i(x)$, $0 < b_i < 1$.

Аддитивное преобразование для построения единого критерия очень часто используется, если объединение различных частных критериев проводится на экономической основе.

В случае использования мультиплективной свёртки обобщённый критерий формируется следующим образом:

$$F(x) = \prod_{i=1}^r f_i(x)^{\lambda_i}, \quad (1.5)$$

где λ_i — некоторые вещественные числа.

Использование преобразования (1.5) характерно для тех задач принятия решений, где частными критериями являются некоторые вероятностные характеристики или где возможно использование в качестве $F(x)$ некоторых удельных характеристик.

Например, обобщённый критерий технико-экономической эффективности ЭВМ, предложенный в [26], построен следующим образом:

$$F(T) = \frac{f_1(T)}{T \cdot V(T)}, \quad (1.6)$$

где $V(T) = f_2(T)/T$ и отражает эффективное быстродействие ЭВМ, причём является критерием оценки технической эффективности ЭВМ и зависит от её важнейших технических характеристик (системы операций, номинального быстродействия, ёмкости памяти, надёжности, скорости работы устройств ввода-вывода); $f_1(T)$ — суммарные затраты на изготовление, амортизацию и эксплуатацию ЭВМ за время T ; $f_2(T)$ — объём работы, выполненной ЭВМ за время её живучести, в пересчёте на число операций типового набора; T — время живучести ЭВМ.

Если критерии $f_i(x)$, ($i = \overline{1, r}$) измеряются в различных шкалах, то необходимо привести их к единой шкале. Для этого обычно формируют следующий критерий:

$$\min_{x \in X} F(x) = \min_{x \in X} \sum_{i=1}^r b_i \frac{f_i - f_i(x)}{|f_i|}, \quad (1.7)$$

где $f_i = \max_x f_i(x_i)$.

К недостаткам данного метода можно отнести существование возможных неоднозначных компенсаций значений критериев.

1.4.3. Введение метрики в пространстве целевых функций

Предположим, что решена система однокритериальных задач:

$$\max f_i(x) = f_i, \quad (i = \overline{1, r}). \quad (1.8)$$

Совокупность скалярных величин f_i определяет в пространстве критериев некоторую точку, которую назовём точкой «абсолютного максимума». Если найденные векторы x^* в каждой из задач (1.8) различны, то не существует такого выбора, который позволил бы достичь этой точки. Точка $(f_1, f_2, \dots, f_i, \dots, f_r)$ является недостижимой в пространстве критериев.

Введём метрику в виде евклидова расстояния от точки

$[f_1(x), f_2(x), \dots, f_r(x)]$ до точки (f_1, f_2, \dots, f_r)

в пространстве критериев:

$$R = \sqrt{\sum_{i=1}^r [f_i(x) - f_i]^2}. \quad (1.9)$$

В качестве нового скалярного критерия можно принять функцию (1.9). Её минимизация даёт исследователю определённую полезную информацию и показывает предельные возможности достижения абсолютного максимума.

1.4.4. Метод максиминной свёртки

Задаётся некоторая система нормативов: $(f_1^*, f_2^*, \dots, f_r^*)$. Это значит, что параметры будущего проекта должны быть таковы, чтобы максимизировать функции $f_i(x) \geq f_i^*$, $i = \overline{1, r}$.

В таких случаях интегральный критерий удобно представить в виде:

$$F(x) = \min_i \frac{f_i(x)}{f_i^*} \quad (1.10)$$

и искать вектор x^* , который обеспечивает максимальное значение $F(x)$. Если значения f_i^* жестко не заданы, то они могут быть определены в результате экспертного опроса.

1.4.5. Метод последовательных уступок

Все критерии $(f_1, f_2, \dots, f_i, \dots, f_r)$ расположим в порядке убывания важности. В основу построения процедуры упорядоченности может быть положен метод экспертных оценок. Каждый из критериев необходимо максимизировать.

Процедура построения компромиссного решения сводится к следующему. Сначала ищется решение, обращающее в максимум главный критерий f_1 . Затем назначается, исходя из практических соображений и заданной точности, некоторая уступка Δf_1 , которую согласны мы допустить, чтобы обратить в максимум второй критерий f_2 . Накладываем на критерий f_1 такое ограничение, чтобы он был не меньше $f_1^* - \Delta f_1$, где f_1^* — максимально возможное значение f_1 , и при этом ограничении ищем решение, образующее максимум f_2 .

Далее снова назначается уступка в критерии f_2 , ценой которой можно максимизировать f_3 и т. д. до последнего критерия f_r .

Такой способ компромиссного решения хорош тем, что здесь сразу видно, ценой какой уступки в одном критерии приобретается выигрыш в другом.

1.4.6. Компромиссы Парето

Все рассмотренные выше способы разрешения многокритериальной проблемы выбора были основаны на различных операциях свёртывания скалярных критериев к интегральным. К решению многокритериальных задач можно подойти с других позиций — попытаться сократить множество исходных вариантов, т. е. исключить из неформального анализа те вари-

анты решений, которые заведомо плохи. Один из подобных путей был предложен итальянским экономистом Парето в 1904 г.

Для раскрытия содержания этого подхода воспользуемся теоретико-множественной интерпретацией, для чего введём следующие операции и понятия [24].

Определение 1. Отношением R на множестве элементов Ω называется подмножество R множества $\Omega \times \Omega$, т. е. $R \in \Omega \times \Omega$.

Содержательный смысл этого определения состоит в том, что задание подмножества R на множестве $\Omega \times \Omega$ определяет, какие пары находятся в отношении R . Это подчёркивается следующим соглашением об обозначениях:

если пара $\langle x, y \rangle$ входит в R , т. е. $\langle x, y \rangle \in R$, то пишут $x R y$, что читается: “ x находится в отношении R с y ”.

Множество Ω называется областью задания отношения и в тех случаях, где существенна область задания отношения, используется пара обозначений $\langle R, \Omega \rangle$.

Пример 1.1

Пусть Ω_1 — множество студентов группы, Ω_2 — множество студентов факультета, Ω_3 — множество студентов всего института. Естественно определяются три разных отношения: $\langle R_1, \Omega_1 \rangle$, $\langle R_2, \Omega_2 \rangle$, $\langle R_3, \Omega_3 \rangle$, где R_i — множество таких пар $\langle x, y \rangle$, что « x » знаком с « y », но при $i = 1$ областью задания отношения $\langle R_1, \Omega_1 \rangle$ является множество студентов одной группы; при $i = 2$ — факультета, при $i = 3$ — института.

Способы задания отношений

Существуют три основных способа задания отношений: матрицей, графом, сечениями.

Матричный способ. Общее правило задания матрицы отношения формулируется так:

$$A = \{a_{ij}\}, (i, j = \overline{1, n}),$$

где $a_{ij}(R) = 1$, если выполнено $x_i R x_j$, $a_{ij}(R) = 0$, если не выполнено $x_i R x_j$.

Задание графом. Элементы конечного множества Ω — вершины графа. От элемента x_i к элементу x_j проводится дуга тогда и только тогда, когда выполнено $x_i R x_j$ (при $i = j$ дуга $(x_i x_j)$ превращается в петлю при вершине x_j).

Задание сечениями. Этот способ менее распространён, чем предыдущие, однако он пригоден и для задания отношений на бесконечных множествах.

Определение 2. Верхним сечением называется множество элементов $y \in \Omega$ таких, что $\langle y, x \rangle \in R$;

$$R^+(x) = \{y \in \Omega \mid \langle y, x \rangle \in R\}. \quad (1.11)$$

Аналогично определяется нижнее сечение:

$$R^-(x) = \{y \in \Omega \mid \langle x, y \rangle \in R\}. \quad (1.12)$$

Таким образом, множество $R^-(x)$ — это множество всех элементов $y \in \Omega$, с которыми фиксированный элемент $x \in \Omega$ находится в отношении R . Множество $R^+(x)$ — это множество всех элементов $y \in \Omega$, которые находятся в отношении R с фиксированным элементом $x \in \Omega$ (рис. 1.1).

Определение 3. Отношение \bar{R} называется дополнением отношения R , если оно выполняется для тех и только тех пар, для которых не выполняется отношение R . Очевидно, что $\bar{R} = ((\Omega \times \Omega) \setminus R)$.

Определение 4. Отношение Парето (P) определяется следующим образом:

$$\begin{aligned} (\forall x, y \in \Omega)[x P y] \Leftrightarrow & \{(\forall j=1, n)[x_j \geq y_j] \wedge \\ & \wedge (\exists j_0 \in [1, \dots, n])[x_{j_0} > y_{j_0}]\}, \end{aligned} \quad (1.13)$$

где множество Ω — n -мерное пространство R^n , $y \in R^n$.

Определение 5. Множеством Парето на $\Omega \subseteq R^n$ называется множество:

$$\Omega^P = \{x \in \Omega \mid \forall y \in \Omega [y \bar{P} x]\}. \quad (1.14)$$

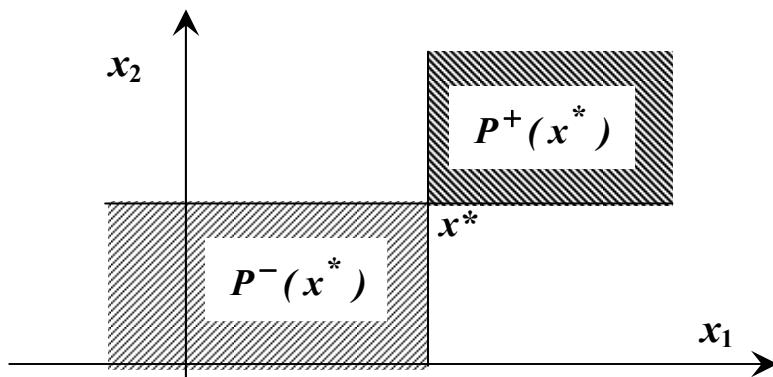


Рис. 1.1

Из определения множества следует, что Ω^P содержит те и только те элементы x^* для которых $P_\Omega^+(x^*) = \emptyset$.

Пример 1.2

Пусть $\Omega = \{x^{(1)}, x^{(2)}, \dots, x^{(6)}\}$;

$$\begin{bmatrix} x_1^{(1)} = 2 \\ x_2^{(1)} = 5 \end{bmatrix}; \begin{bmatrix} x_1^{(2)} = 3 \\ x_2^{(2)} = 3 \end{bmatrix}; \begin{bmatrix} x_1^{(3)} = 1 \\ x_2^{(3)} = 4 \end{bmatrix}; \begin{bmatrix} x_1^{(4)} = 1 \\ x_2^{(4)} = 3 \end{bmatrix}; \begin{bmatrix} x_1^{(5)} = 4 \\ x_2^{(5)} = 3 \end{bmatrix}; \begin{bmatrix} x_1^{(6)} = 5 \\ x_2^{(6)} = 4 \end{bmatrix}.$$

Множества Ω и Ω^P изображены на рис. 1.2.

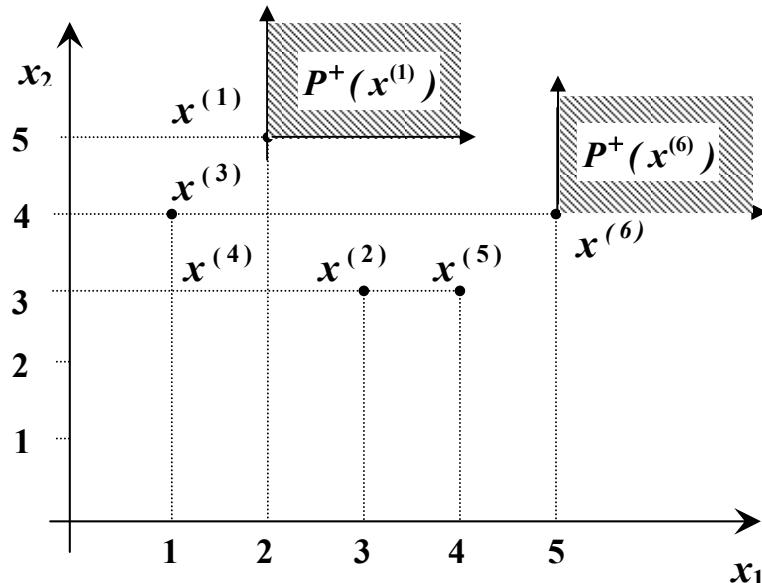


Рис. 1.2.

Легко видеть, что

$$P_\Omega^+(x^{(1)}) = \emptyset, \quad P_\Omega^+(x^{(2)}) = \{x^{(5)}, x^{(6)}\};$$

$$P_\Omega^+(x^{(3)}) = \{x^{(1)}, x^{(6)}\};$$

$$P_\Omega^+(x^{(4)}) = \{x^{(1)}, x^{(2)}, x^{(3)}, x^{(5)}, x^{(6)}\};$$

$$P_\Omega^+(x^{(5)}) = \{x^{(6)}\}; \quad P_\Omega^+(x^{(6)}) = \emptyset.$$

Таким образом, множество Парето Ω^P включает только точки $x^{(1)}$ и $x^{(6)}$. Переходя к многокритериальному пространству R^r , где $f^T = [f_1, f_2, \dots, f_i, \dots, f_r]$ и множеству решений $x \in X$, где

$x^T = [x_1, x_2, \dots, x_j, \dots, x_n]$ отношение Парето представляет собой в соответствии с определением 4 следующее отношение доминирования:

$$\begin{aligned} (\forall x, y \in \Omega)[x P y] \Leftrightarrow & \{(\forall i=1, r)[f_i(x) \geq f_i(y)] \wedge \\ & \wedge (\exists i_0 \in [1, 2, \dots, r])[f_{i_0}(x) > f_{i_0}(y)]\}. \end{aligned} \quad (1.15)$$

Определение 6. Если для некоторой точки $x^* \in Y$ не существует более предпочтительной по Парето точки, т. е. такой точки x , что $\langle x, x^* \rangle \in P$, тогда x^* называется эффективным или Парето-оптимальным решением многокритериальной задачи:

$$f_i(x) \Rightarrow \max, i = \overline{1, r}, x \in X.$$

Множество, включающее в себя все эффективные элементы x множества Ω^P , называется множеством Парето и обозначается $P(x)$.

Отображение множества $P(x)$ в пространстве критериев R^r обозначается $P(f)$ и называется множеством эффективных оценок. Смысл введённого понятия состоит в том, что оптимальный исход следует искать среди элементов множества недоминируемых элементов $P(x)$ (принцип Парето). В противном случае всегда найдётся точка $y \in X$, оказывающаяся более предпочтительной с учётом всех частных целевых функций $f_i(y), i=\overline{1, r}$.

Заметим, что принцип Парето не выделяет единственного решения, он только сужает множество альтернатив, окончательный выбор остаётся за ЛПР на основе дополнительной информации о его предпочтениях.

1.5. ЭКСПЕРТНЫЕ МЕТОДЫ ПРИНЯТИЯ РЕШЕНИЙ

В практике принятия решений часто возникают такие ситуации, при которых частично или полностью неизвестна или трудно доступна информация для описания ситуации, или которые невозможно формировать с достаточной степенью точности. В этом случае такие проблемы обычно решаются с помощью привлекаемой группы экспертов, анализирующих и оценивающих имеющуюся ситуацию и генерирующих некоторое множество альтернатив ее решения.

Суть метода принятия решений с привлечением экспертов состоит в том, чтобы получить экспертные оценки выставляемые индивидуально каждым экспертом и сформулировать обобщённое мнение группы в целом о наилучшем объекте (решении).

Для большей определённости анализа экспертного подхода в целом рассмотрим далее задачу определения полезности соответствующего критерия в заданном критериальном пространстве R^r :

$$f^T = [f_1, f_2, \dots, f_i, \dots, f_r]$$

при выборе решения $x \in X$.

Сама процедура проведения экспертного опроса проводится в несколько этапов:

- формирование экспертной группы;
- проведение опроса;
- обработка и анализ результатов опроса;
- формирование окончательного решения.

Дадим краткую характеристику указанным этапам.

При формировании экспертной группы необходимо учитывать такие факторы, как компетентность экспертов, их независимость, деловые качества и совпадение целей экспертизы для каждого из них.

Количество экспертов в группе должно составлять от 5 до 15 человек. Если членов экспертной группы больше указанного числа, это ухудшает процедуру принятия решения из-за увеличения времени оценки и анализа, сложности процессов согласования мнений экспертов, снижения качества статистической обработки информации.

Проведение опроса предваряется заданием процедуры оценивания, указанием типа шкалы, по которой необходимо оценивать объекты и определением основных оцениваемых параметров объекта. Процедура оценивания обычно проводится в форме опроса, интервью, анкетирования, либо дискуссии. Далее осуществляется обработка данных, полученных на этапе оценивания. Обработка может быть как количественной, так и качественной. При этом оцениваются как эксперты, так и сама проблемная ситуация. На основе личных оценок каждого эксперта вычисляются групповые оценки, которые дополнительно проверяются на достоверность, причём они считаются таковыми, если индивидуальные оценки согласованы между собой. Экспертный опрос завершается формированием окончательного решения с использованием тех или иных известных стратегий принятия решений.

Будем считать, что этап формирования экспертной группы успешно завершен и рассмотрим более подробно содержание оставшихся трех.

1.5.1. Проведение экспертного опроса

Проведение опроса целесообразно представить в виде процедуры сравнения. В настоящее время достаточно хорошо проработаны и используются на практике следующие процедуры сравнения:

- ранжирование критериев;
- парное сравнение критериев;
- непосредственная оценка критериев;
- последовательное сравнение критериев [22].

Ранжирование критериев предполагает, что они упорядочены в соответствии с относительной ценностью, так что на первом месте находится самый главный критерий.

Далее проводится нумерация критериев полученного ряда, причём неразличимые критерии, которые оказались на одной позиции, нумеруются в произвольном порядке.

Ранг r_i критерия определяется его номером, если на его месте в ряду отсутствуют какие-либо другие. Если на одном месте находятся несколько неразличимых критериев, то ранг каждого из них равен среднему арифметическому их новых номеров.

Переход от рангов к коэффициентам c_i относительной ценности критерия производится в соответствии со следующим соотношением:

$$c_i = 1 - \frac{r_i - 1}{r}, \quad i = \overline{1, r}. \quad (1.16)$$

Пример 1.3

Пусть для ЛПР имеем следующий ряд упорядоченных критериев f_1, f_2, \dots, f_8 :

$$f_3; \left| \begin{matrix} f_5 \\ f_6 \end{matrix} \right|; \left| \begin{matrix} f_2 \\ f_4 \end{matrix} \right|; f_8; f_7 : f_1. \quad (1.17)$$

Ранги критериев, вычисленные в соответствии с указанной процедурой приводятся в табл. 1.1.

Таблица 1.1

i	1	2	3	4	5	6	7	8
r_i	8,0	4,5	1,0	4,5	2,5	2,5	7,0	6,0

Коэффициенты относительной ценности критериев, вычисленные в соответствии с соотношением (1.16) сведены в табл. 1.2.

Таблица 1.2

i	1	2	3	4	5	6	7	8
c_i	0,125	0,433	1,000	0,433	0,812	0,812	0,250	0,375

Метод парного сравнения заключается в установлении предпочтений при сравнении двух критериев. Матрица предпочтений A составляется следующим образом:

$$a_{ij} = \begin{cases} 1, & \text{если } f_i \succ f_j \text{ или } f_i \sim f_j; \\ 0, & \text{если } f_i \prec f_j. \end{cases}$$

Значение «1» при сравнении двух критериев выставляется лишь в том случае, если i -й критерий более предпочтителен, чем j -й или эквивалентен ему. Матрица A результатов обработки ряда (1.17) по методу парных сравнений представлена в табл. 1.3.

Метод непосредственной оценки основывается на присвоении экспертом каждому критерию некоторой вычислённой формально или эвристической оценки, отражающей его количественную характеристику, т. е. абсолютное или относительное значение интегральной характеристики критерия на некотором числовом отрезке. В табл. 1.4. приводятся оценки критериев f_1, f_2, \dots, f_8 , присвоенные ЛПР для отрезка $[0,1]$.

Таблица 1.3

	1	2	3	4	5	6	7	8
1	1	0	0	0	0	0	0	0
2	1	1	0	1	0	0	1	1
3	1	1	1	1	1	1	1	1
4	1	1	0	1	0	0	1	1
5	1	1	0	1	1	1	1	1
6	1	1	0	1	1	1	1	1
7	1	0	0	0	0	0	1	0
8	1	0	0	0	0	0	1	1

Таблица 1.4

i	1	2	3	4	5	6	7	8
c _i	0,1	0,4	1,0	0,4	0,6	0,6	0,3	0,3

В основу этого метода положена менее жёсткая гипотеза об убывающей (но не обязательно линейной) зависимости между рангом и относительной ценностью критерия. Вначале ЛПР производит упорядочение всех критериев, после этого эвристическим путём даёт численную оценку относительной ценности каждого критерия по сравнению с самым глав-

ным, которому присвоено значение, равное единице. Всем неразличимым критериям присваивается одинаковое значение c_i .

В результате каждому критерию в упорядоченном ряду вместо рангов присваиваются числа c_i , совокупность которых должна образовать невозрастающую последовательность.

При использовании метода непосредственных оценок возникает возможность более дифференцированного подхода к оценке важности отдельных критериев, но при этом понижается достоверность полученной информации.

Метод последовательного сравнения предназначен для повышения достоверности информации, полученной от экспертов методом непосредственных оценок. Он позволяет каждому эксперту провести самоконтроль суждения на основе ранжирования критериев, числовой оценки их полезности и сравнения $r - 2$ пар специально подобранных абстрактных объектов. Последняя процедура, отображающая сущность метода последовательного сравнения основана на следующей гипотезе [22]. Если полезность i -го критерия есть c_i , ($i = \overline{1, r}$), то полезность всего объекта, «в полной мере» обладающего всеми этими критериями есть $\sum_{i=1}^r c_i$. В процессе коррекции оценок эксперт должен ответить на последовательность вопросов (для $i = 1, 2, \dots, r - 2$).

В зависимости от ответа на i -й вопрос составляется одно из трёх соотношений:

$$c_{ij} R \sum_{k=i+1}^r c_{kj}, \text{ где } R \in [>, <, =].$$

В результате будут получены $(r - 2)$ условия:

$$c_{1j} R \sum_{k=2}^r c_{kj}, c_{2j} R \sum_{k=3}^r c_{kj}, \dots, c_{r-2,j} R (c_{r-1,j} + c_{rj}),$$

где c_{ij} — числовая оценка полезности i -го критерия, выставленная j -м экспертом ($i = \overline{1, r}$).

Далее производится последовательная проверка каждого из этих условий, начиная с последнего, на соответствие ранее выбранным оценкам c_{ij} и их ранжировке.

При выявлении противоречий в i -м условии эксперт должен либо изменить знак отношения R , либо откорректировать значение величины c_{ij} . В последнем случае он обязан убедиться в том, что не оказалась нару-

шенной первоначальная ранжировка критериев. При нарушении ее необходимо либо изменить порядок следования критериев, либо откорректировать значение c_{ij} . Следует отметить, что в этом случае психологические ограничения не дают использовать метод последовательных сравнений, когда число рассматриваемых критериев превышает семь.

1.5.2. Обработка результатов опроса

Основная ее цель сводится к решению следующих задач:

- определение достоверности результатов проведённого экспертного опроса;
- построение результирующей оценки по оценкам, даваемым экспертами.

Далее, в первую очередь, речь пойдёт о статистических методах, основанных на предположении, что отклонение оценок экспертов от истинных происходит в силу случайных причин и задача состоит в том, чтобы восстановить истинное значение с наименьшей погрешностью.

Содержание процедуры обработки зависит от того, как проводится опрос экспертов.

Для количественной оценки достоверности результатов ранжирования используется коэффициент конкордации W , характеризующий степень согласованности мнений экспертов [24]. В случае строгого ранжирования (отсутствуют критерии, находящиеся на одной позиции) значение W определяется следующим образом:

$$W = \frac{12 \sum_{i=1}^r \left[\sum_{j=1}^m r_{ij} - m(r+1)/2 \right]^2}{m^2 (r^3 - r)}, \quad (1.18)$$

где r — число критериев, m — число экспертов, r_{ij} — ранг, выставленный j -м экспертом i -му критерию.

В случае нестрогого ранжирования:

$$W = \frac{12 \sum_{i=1}^r \left[\sum_{j=1}^m r_{ij} - \frac{1}{2} m(r+1) \right]^2}{m^2 (r^3 - r) - m \sum_{j=1}^m k_j \sum_{l=1}^{k_j} (t_{lj}^3 - t_{lj})}, \quad (1.19)$$

где k_j — число групп равных рангов, введённых j -м экспертом; t_{lj} — количество дробных рангов в l -ой группе, введённых j -м экспертом.

Значение W находится в пределах $0 \leq W \leq 1$, причём, случай $W=0$ означает полную противоположность, а $W=1$ — полное совпадение ранжировок.

Практически, достоверность экспертного опроса считается хорошей, если

$$0,6 \leq W \leq 0,8.$$

Для характеристики степени согласованности мнений экспертов при использовании метода непосредственной оценки критериев служит дисперсия оценок, вычисляемая по следующей формуле:

$$\sigma^2 = \frac{\sum_{j=1}^m (c_j - c)^2 \alpha_j}{\sum_{j=1}^m \alpha_j}, \quad (1.20)$$

где $c = \frac{\sum_{j=1}^m c_j \alpha_j}{\sum_{j=1}^m \alpha_j}$, α_j ($j = 1, m$) — веса экспертов, (при отсутствии информации о компетенции экспертов можно считать $\alpha_j = 1$, $j = 1, m$), c_j — оценка j -го эксперта.

При решении многокритериальной задачи соответствующие оценки дисперсии (1.20) могут быть получены по каждому из критериев.

Используя распределение Стьюдента [24], можно определить статистическую значимость результатов и построить соответствующие доверительные интервалы.

Если выбранные критерии согласованности не выполнены, то необходимы повторные экспертизы либо с расширением числа участников, либо с представлением экспертам дополнительной информации. Кроме того, в этом случае должны быть приняты специальные организационные меры, чтобы обеспечить достаточную достоверность результатов.

Соответствующие правила и процедуры, конечно, во многом зависят от содержания конкретных экспертиз, но, тем не менее, может быть высказан целый ряд общих соображений, которые достаточно хорошо сформулированы в методе «Дельфы» [24]. Данный метод широко использовался в США для решения разнообразных политических, экономических, социальных и военных проблем.

Метод получил название по имени древнегреческого города Дельфы, где по преданию находился известный дельфийский оракул.

Метод Дельфы содержит серию рекомендаций.

1. Экспертами могут быть только признанные специалисты в соответствующей области.

2. Так как каждый эксперт может ошибаться, то только мнение достаточно большого числа экспертов может удовлетворительно охарактеризовать исследуемый вопрос.

3. Вопросы, которые ставятся экспертам, должны быть относительно просты и поставлены настолько чётко, чтобы исключить неоднозначность толкования.

4. Исключаются открытые дискуссии, но допускается процедура получения и анализа экспертами дополнительной информации.

5. Лица, организующие экспертизы, должны проявлять большой такт, чтобы в максимальной степени исключить влияние организаторов опроса на мнение экспертов.

Перейдем ко второй задаче обработки результатов опроса — построению результирующей оценки критериев по оценкам отдельных экспертов. Пусть в результате выбранной процедуры опроса построена матрица

$$c = \{c_{ij}\}_r^m$$

где r — число критериев, m — число экспертов, c_{ij} — мнение j -го эксперта об относительной ценности i -го критерия.

Процедура построения результирующей оценки зависит от процедуры проведения опроса.

При ранжировании критериев для построения средней оценки в соответствии с [24] вводится понятие расстояния между ранжировками $r^{(k)}$ и $r^{(l)}$, проведёнными k -м и l -м экспертами:

$$d(r^{(k)}, r^{(l)}) = \frac{1}{2} \sum_{i=1}^r |r_i^{(k)} - r_i^{(l)}|. \quad (1.21)$$

При наличии m ранжировок $r^{(1)}, r^{(2)}, \dots, r^{(j)}, \dots, r^{(m)}$, проведённых m экспертами, результирующей считается та ранжировка, сумма расстояний от которой до исходных есть величина наименьшая из всех возможных. Такая ранжировка называется *медианой Кемени*.

Медиана Кемени \tilde{r} находится из задачи минимизации по всем возможным ранжировкам из множества R (таких ранжировок $r!$) суммы расстояний от r до $r^{(1)}, r^{(2)}, \dots, r^{(j)}, \dots, r^{(m)}$, т. е. определяется формулой:

$$\sum_{j=1}^m d(\tilde{r}, r^{(j)}) = \min_{r \in R} \sum_{j=1}^m d(r, r^{(j)}), \quad (1.22)$$

где m — число экспертов, $r^{(j)}$ — ранжировка на множестве критериев, проведённая j -м экспертом.

Наряду с принципом Кемени, для выбора результирующего ранжирования часто используется принцип большинства. Он состоит в следующем: на первое место в результирующую ранжировку попадает критерий, который большинство экспертов поставили на первое место, на второе — критерий, который поставили на второе и т. д.

Однако, как показано в [24], данный принцип в отличие от принципа Кемени не всегда приводит к выбору наилучшего решения.

Метод непосредственной оценки критериев определяет иную процедуру построения результирующей оценки критериев. Сначала предложенные экспертами оценки c_{ij} ($i = \overline{1, r}$, $j = \overline{1, m}$) масштабируются:

$$b_{ij} = c_{ij} / \sum_{j=1}^r c_{ij}; \quad \sum_{j=1}^r b_{ij} = 1, \quad (i = \overline{1, r}, j = \overline{1, m}). \quad (1.23)$$

Дальнейшее построение результирующих оценок полезности критериев после проведения экспертного опроса может быть осуществлено различными способами, которые поясняются ниже.

1. Способ арифметического усреднения:

$$b_i = \frac{1}{m} \sum_{j=1}^m b_{ij}. \quad (1.24)$$

2. Способ средне-взвешенного усреднения:

$$b_i = \frac{\sum_{j=1}^m \alpha_j b_{ij}}{\sum_{j=1}^m \alpha_j}, \quad (1.25)$$

где α_j учитывает компетентность j -го эксперта.

3. Способ порядкового усреднения [24]:

$$b_i = med(b_{11}, b_{12}, \dots, b_{ij}, \dots, b_{im}), \quad (1.26)$$

где использование медианы (*med*) позволяет исключить влияние на окончательные результаты опроса аномальных, резко отличающихся от других мнений экспертов.

1.5.3. Формирование окончательного решения

Данный этап предполагает формирование интегрального критерия $F(x)$ для решения задачи (1.1). Построение такого критерия может быть проведено как с использованием формальных подходов, рассмотренных в разделе 1.5, так и результатов теории полезности. Теория полезности раз-32

работана Дж. фон Нейманом и О. Моргенштерном. Её математическая основа — система аксиом, в которых утверждается, что существует некоторая мера ценности (полезности), позволяющая упорядочить результаты решений. Эта мера называется функцией полезности или полезностью [28].

Методика определения полезности возможных результатов разработана и представлена в [28]. В соответствии с ней построение интегрального критерия может быть произведено на аддитивной основе:

$$F(x) = \sum_{i=1}^r b_i f_i^{omn}(x), \quad (1.27)$$

где b_i — полезность критерия определяется в соответствии с соотношениями (1.24), (1.25), (1.26); $f_i^{omn}(x)$ — безразмерное значение i -го критерия для варианта $x \in X$, определяемое как:

$$f_i^{omn}(x) = \frac{f_i(x) - f_{ih}}{f_{ie} - f_{ih}}, \quad i = \overline{1, r}, \quad (1.28)$$

где f_{ih} , f_{ie} — нижнее и верхнее значения критерия $f_i(x)$ при $x \in X$.

Линейное преобразование (1.28) может быть заменено на нелинейное с использованием экспоненциальных и логарифмических функций, если при выборе решения необходимо выделить какой-то поддиапазон в интервале $[f_{ih}, f_{ie}]$ или в случаях, когда $f_{ih} = \infty$ или $f_{ie} = \infty$.

Подводя итог рассмотрению метода экспертных оценок, следует сказать несколько слов о его применении. При использовании метода экспертных оценок предлагается разделить задачи на два класса [24]:

1. Задачи, которые достаточно хорошо изучены, обеспечены информацией и для которых групповое мнение экспертов близко к истинному;
2. Задачи, которые изучены недостаточно; экспертов нельзя рассматривать как хороших «измерителей» и необходимо осторожно подходить к обработке результатов экспертного опроса. В этом случае мнение одного эксперта более осведомлённого о малоизученной проблеме может оказаться наиболее значимым, а при формальной обработке в соответствии с соотношениями (1.25), (1.26) оно будет утрачено. В связи с этим к задачам второго класса в основном следует применять качественную обработку результатов.

Поэтому при использовании метода экспертных оценок особое внимание следует обращать:

- на формирование экспертной группы;
- на методы обработки результатов опроса, особо выделяя и учитывая редкие и противоречивые мнения.

Следует учитывать, что получаемые усреднённые оценки отражают некоторую общественную точку зрения, зависящую от уровня научно-технических знаний о предмете исследования, которая может изменяться по мере развития наших представлений о нём.

1.6. ПРИНЯТИЕ РЕШЕНИЙ В УСЛОВИЯХ НЕОПРЕДЕЛЁННОСТИ И РИСКА

1.6.1. Принятие решений в условиях неопределенности

Задача принятия решений в условиях неопределенности фактически называется «игрой с природой».

Природа выступает в роли внешней среды. ЛПР должен сделать некоторый выбор из множества допустимых альтернатив, принятие которых зависит от того, какое конкретное состояние природы будет наблюдаться.

При этом предполагается, что множество состояний природы не известно ЛПР.

Множество неопределённых факторов можно разделить на *объективные и субъективные*.

Субъективная неопределенность характеризует индивидуальные психофизические параметры ЛПР.

Основные причины её возникновения:

- неполнота знаний ЛПР о закономерностях и условиях развития ситуации или самой системы;
- неверная мотивация при выборе альтернатив;
- отсутствие информации о наличии в рассматриваемом процессе или системе активных элементов, обладающих возможностями и правами по выбору управления и др.

Причинами возникновения *объективной неопределенности* служат:

- неполнота информации об операции;
- неопределенность цели функционирования;
- наличие случайных факторов;
- неопределенность процедуры принятия решений из-за громоздкости многоуровневой системы управления операцией, либо наличия возможности образования коалиций с противоположными интересами в структуре управления.

В дальнейшем обозначим любую неопределенность векторным параметром $z \in Z$, где Z — множество неопределённостей, а вектор z учитывает любые неконтролируемые, неуправляемые для ЛПР факторы операции.

Эффективность операции определяется скалярным критерием $f(x, z)$, где $x \in X$, $z \in Z$, причём необходимо найти $\max f(x, z)$.

К наиболее часто используемым методам выбора оптимальной стратегии в условиях неопределённости можно отнести [27]:

- 1) принцип максимума (критерий Вальда),
- 2) принцип оптимизма,
- 3) принцип Гурвица,
- 4) принцип Сэвиджа,
- 5) принцип Лапласа.

Для иллюстрации указанных выше принципов, считаем, что X дискретно, т. е. $x = \{x_1, x_2, \dots, x_i, \dots, x_n\}$, Z также дискретно, т. е. $z = \{z_1, z_2, \dots, z_j, z_m\}$, а исходная информация для принятия решения представлена таблицей $\{f(x_i, z_j)\}$.

Критерии выбора оптимальной стратегии в соответствии с указанными выше принципами имеют следующий вид:

1. Критерий Вальда (гарантированного результата):

$$\max_{x_i} \min_{z_j} f(x_i, z_j). \quad (1.29)$$

2. Критерий оптимизма:

$$\max_{x_i} \max_{z_j} f(x_i, z_j). \quad (1.30)$$

3. Критерий Гурвица (комбинированный критерий):

$$\max_{x_i} [\alpha \min_{z_j} f(x_i, z_j) + (1 - \alpha) \max_{z_j} f(x_i, z_j)], \quad 0 < \alpha < 1. \quad (1.31)$$

4. Критерий Сэвиджа:

$$\min_{x_i} \max_{z_j} f_c(x_i, z_j), \quad (1.32)$$

где $f_c(x_i, z_j) = -f(x_i, z_j) + \max_{x_i} f(x_i, z_j).$

(1.33)

5. Критерий Лапласа:

$$\max_{x_i} \frac{1}{m} \sum_{j=1}^m f(x_i, z_j). \quad (1.34)$$

Рассмотрим применение указанных критериев в условиях неопределённости на простом примере.

Пример 1.4

Для исходных данных (таб. 1.5) оптимальные решения:

- для критерия Вальда — решение x_3 ;
- для критерия оптимизма — решение x_1 ;
- для критерия Гурвица — решение x_3 , $\alpha = 0,7$;
- для критерия Сэвиджа — решение x_1 ;
- для критерия Лапласа — решение x_1 .

Таблица 1.5

Реше- ния x_i	Среда z		
	z_1	z_2	z_3
x_1	2	7	9
x_2	5	3	4
x_3	4	8	5

Пример 1.5 [25]

Тренерский коллектив лыжной команды обсуждает вариант мази для лыж. Тренеры знают, что во время лыжной гонки погода (природа) может находиться в трёх состояниях $\beta_1, \beta_2, \beta_3$. В их распоряжении имеется четыре типа лыжной мази $\alpha_1, \alpha_2, \alpha_3, \alpha_4$. Матрица A вероятностей удачного прохождения трассы гонки спортсменами известна:

$$\begin{matrix} & \overbrace{\beta_1 \quad \beta_2 \quad \beta_3} \\ \left\{ \begin{matrix} \alpha_1 & \begin{pmatrix} 0,20 & 0,30 & 0,15 \end{pmatrix} \\ \alpha_2 & \begin{pmatrix} 0,75 & 0,20 & 0,35 \end{pmatrix} \\ \alpha_3 & \begin{pmatrix} 0,25 & 0,80 & 0,25 \end{pmatrix} \\ \alpha_4 & \begin{pmatrix} 0,85 & 0,05 & 0,45 \end{pmatrix} \end{matrix} \right. & = A. \end{matrix}$$

Элемент $\langle \alpha_3, \beta_2 \rangle$ означает, что если будет использоваться мазь α_3 , и природа будет находиться в состоянии, то с вероятностью 0,8 эта мазь позволит спортсмену пройти дистанцию в полную силу. Вероятности состояний погоды тренерам были неизвестны (недоверие прогнозам погоды) и поэтому на тренерском совете им пришлось решать вопрос о том, какую мазью следует мазать лыжи.

Тренер-пессимист руководствовался критерием Вальда. Критерий ориентирован на худшие погодные условия и носит явно пессимистический характер. Стратегия, предложенная пессимистом была α_3 .

Тренер-оптимист как всегда ориентировался на лучшее и предложил четвёртый вариант смазки α_4 , так как выигрыш, соответствующий благоприятной погоде, наибольший.

Старший *тренер-рационалист*, выслушав свою команду, предложил использовать критерий пессимизма-оптимизма Гурвица при $\alpha = 0,6$ и в соответствии этим выбрать стратегию α_3 . С точки зрения здравого смысла такое решение достаточно оправдано, так как нельзя впадать при принятии стратегических решений ни в крайний оптимизм, ни в крайний пессимизм.

Представитель тренерского совета (возмутитель спокойствия) предложил:

1) исключить стратегию α_1 из рассмотрения, так как она заведомо хуже стратегии α_3 ,

2) вместо матрицы успехов A ориентироваться на матрицу рисков $R = (r_{ij})$, где r_{ij} — отклонение выигрыша a_{ij} от максимально возможного, получаемого при условии заблаговременной информации о том, что природа находится в состоянии β_j . Матрица рисков R выглядит следующим образом:

$$\begin{array}{c} \overbrace{\beta_1 \quad \beta_2 \quad \beta_3} \\ \left\{ \begin{array}{l} \alpha_1 \left(\begin{array}{ccc} 0,10 & 0,60 & 0,10 \end{array} \right) \\ \alpha_2 \left(\begin{array}{ccc} 0,60 & 0,00 & 0,20 \end{array} \right) \\ \alpha_3 \left(\begin{array}{ccc} 0,00 & 0,75 & 0,00 \end{array} \right) \end{array} \right. \end{array}$$

В соответствии с этим следует выбирать такую стратегию, при которой величина риска окажется наименьшей в самой неблагополучной ситуации, т.е. необходимо выбрать стратегию α_2 или α_3 характеризующие наименьший риск при любом состоянии природы.

Старший тренер, подводя итог дискуссии, заметил, что последнее предложение немногим лучше предложения «пессимиста»; различие лишь в том, что по предложенному критерию наихудшим следует считать не минимальный выигрыш, а максимальный риск. Это хорошо известный критерий Сэвиджа. Критерий этот столь же пессимистичен, как и «мак-

сминный» критерий Вальда. Поэтому, придерживаясь золотой середины, целесообразно остановиться на стратегии α_3 .

Выбор критерия из рассмотренных является наиболее сложным и ответственным этапом в исследовании операций. При этом не существует каких-либо общих рекомендаций и советов. Выбор критерия должен быть в максимальной степени согласован с конкретной спецификой задачи, а также с поставленными целями. Например, если минимальный риск недопустим, то следует применять критерий Вальда.

1.6.2. Принятие решений в условиях риска

В отличие от условий полной неопределённости на множестве Z задана вероятностная метрика. Если множество Z дискретно, то каждому z_j соответствует вероятность

$$P_j, \quad (j = \overline{1, m}), \quad \sum_{j=1}^m P_j = 1.$$

В этих условиях наиболее распространённым принципом принятия решений является принцип Байеса:

$$\max_{x_i} \sum_{j=1}^m P(z_j) f(x_i, z_j). \quad (1.35)$$

При равновероятном распределении:

$$P(z_j) = \frac{1}{m}, \quad (j = \overline{1, m})$$

решения, выбранные в соответствии с принципом Байеса (1.35) и Лапласа (1.34) совпадают.

Если множество Z непрерывно, то на множестве задаётся дифференциальный или интегральный закон распределения вектора $z \in Z$. В этом случае возможны различные постановки задач выбора оптимального решения:

$$\max_{x \in X} \overline{f(x, z)}, \quad (1.36)$$

$$\overline{\max_{x \in X} f(x, z)}, \quad (1.37)$$

$$\max_{x \in X} f(x, \bar{z}), \quad (1.38)$$

где \bar{y} — математическое ожидание вектора y .

Рассмотренные задачи, строго говоря, не эквивалентны. Выбор любой из них отражает предположение о том, что определение решения в со-

ответствие с выбранной постановкой обеспечивает требуемую эффективность операции. Постановка (1.36) соответствует использованию принципа Байеса в непрерывном случае.

1.7. ПРИНЯТИЕ РЕШЕНИЯ В УСЛОВИЯХ КОНФЛИКТА

Для обоснования решений в условиях конфликта разработаны специальные математические методы, которые рассматриваются в теории игр. Её возникновение относится к 1944г., когда вышла в свет монография Неймана и Моргенштерна «Теория игр и экономическое поведение». В настоящее время теория игр превратилась в самостоятельное математическое направление, имеющее много практических приложений.

В отличие от раздела 1.6, когда параметром z управляла «природа», в условиях конфликта параметр z отражает действия «разумного» противника, преследующего собственные цели. Столкновение противоположных интересов участников приводит к возникновению конфликтных ситуаций. Необходимость анализировать такие ситуации привела к возникновению теории игр, задачей которой является выработка рекомендаций по рациональному образу действий участников конфликта.

В дальнейшем рассматриваются только стратегические игры, в отличие от комбинаторных и азартных игр, когда источник неопределённости состоит в отсутствии информации о действиях противника, о его стратегии.

Рассмотрим модель игры более подробно. Если сталкиваются интересы двух противников, игра называется *парной*, если более двух — *множественной*.

Так как наибольшее практическое значение имеют парные игры, то рассмотрим только их. Итак, участников игры двое — ***A*** и ***B***. Задано множество стратегий X , игроков ***A*** и ***B***. Соответственно:

$$x = (x_1, x_2, \dots, x_i, \dots, x_m),$$

$$z = (z_1, z_2, \dots, z_i, \dots, z_m),$$

Простейший вид стратегической игры — игра двух лиц с нулевой суммой (сумма выигрышей равна нулю). Выбор стратегии каждым из игроков производится при полном незнании выбора другого игрока. В результате выигрыш $\varphi_1(x_i, z_j)$ и $\varphi_2(x_i, z_j)$ каждого из игроков удовлетворяет соотношению:

$$\varphi_1(x_i, z_j) + \varphi_2(x_i, z_j) = 0, \quad (1.39)$$

откуда, если

$$\varphi_1(x_i, z_j) = f(x_i, z_j),$$

то

$$\Phi_2(x_i, z_j) = -f(x_i, z_j).$$

Цель игрока A — максимизировать функцию $f(x_i, z_j)$, в свою очередь, цель игрока B — минимизировать эту же функцию.

Пусть $f(x_i, z_j) = a_{ij}$, $i = \overline{1, n}$, $j = \overline{1, m}$.

Составим матрицу $A = \{a_{ij}\}$. Матрица A называется платёжной, её строки соответствуют стратегиям x_i , столбцы — стратегиям z_j , элемент a_{ij} — выигрыш игрока A , если он выбрал стратегию x_i , а игрок B — стратегию z_j .

Игрок A , не зная информации о выборе стратегии игроком B , должен сделать такой выбор, чтобы максимизировать свой минимальный выигрыш:

$$\alpha = \max_i \min_j a_{ij}, \quad (1.40)$$

где α — гарантированный выигрыш игрока A — называется *нижней ценой игры*. Стратегия x_{i0} , обеспечивающая получение α , называется *максиминной*.

Соответственно игрок B , выбирая стратегию, исходит из следующего принципа: при выборе некоторой стратегии z_j его проигрыш не превысит максимального из значений элементов j -го столбца матрицы, т. е. меньше или равен $\max_i a_{ij}$. Естественно, что игрок B выберет такое значение j , при котором его максимальный проигрыш β минимизируется:

$$\beta = \min_j \max_i a_{ij}. \quad (1.41)$$

Величина β называется *верхней ценой игры*, а соответствующая выигрышу β стратегия z_{j0} *минимаксной*.

Фактический выигрыш игрока A при разумных действиях партнёров ограничен нижней и верхней ценой игры. Если же эти выражения равны

$$\max_i \min_j a_{ij} = \min_j \max_i a_{ij} = v, \quad (1.42)$$

то выигрыш игрока A — *вполне определённое число*, и игра называется *вполне определённой*, а выигрыш (1.42) называется решением игры и равен элементу матрицы a_{i0j0} .

Вполне определённые игры называются играми *с седловой точкой*, а элемент a_{i0j0} называется *седловой точкой*.

Седловой точке соответствует оптимальные стратегии игроков, их совокупность является решением игры, которое обладает следующим свойством: если один из игроков придерживается своей оптимальной стратегии, то для другого отклонение от его оптимальной стратегии не может быть выгодно.

Пример 1.6

Игры заданы платёжными матрицами:

$$A_1 = \begin{pmatrix} 4 & 3 & 4 & 2 \\ 3 & 4 & 6 & 5 \\ 2 & 5 & 1 & 3 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 1 & 0 & 3 & 5 \\ 3 & 2 & 4 & 3 \\ 0 & 1 & -1 & 4 \end{pmatrix}.$$

Для матрицы A_1 : $\alpha_1 = 3$, $\beta_1 = 4$.

Для матрицы A_2 : $\alpha_2 = 2$, $\beta_2 = 2$.

Таким образом, цена второй игры $v = 2$, решение игры состоит в выборе игроком A стратегии x_2 , а игроком B — стратегии y_2 . Легко видеть, что отклонение одним из игроков от оптимальной стратегии приводит к уменьшению выигрыша для игрока A и к увеличению проигрыша для игрока B .

Итак, если игровая матрица содержит *седловую* точку, то решение игры известно. Возникает вопрос нахождения решения для игр, матрицы которых не содержат седловой точки, в этих играх $\alpha < \beta$.

Естественным для каждого игрока является вопрос увеличения выигрыша (уменьшения проигрыша). Поиски такого решения состоят в том, что игроки применяют не одну, а несколько стратегий, причём выбор стратегий осуществляется случайным образом. Случайный выбор игроком своих стратегий называется *смешанной стратегией*.

В данной игре стратегии игрока A задаются набором вероятностей:

$$P^T = (p_1, p_2, \dots, p_i, \dots, p_n),$$

а стратегии игрока B задаются набором вероятностей :

$$Q^T = (q_1, q_2, \dots, q_j, \dots, q_m),$$

причём:

$$\sum_i p_i = 1, \quad \sum_j q_j = 1.$$

Стратегии игроков A и B , для которых вероятности p_i и q_j отличны от нуля, называются *активными*.

Согласно основной теореме игр [28], каждая конечная игра имеет, по крайней мере, одно решение, которым может быть и смешанная стратегия. При использовании смешанной стратегии выигрыш игрока A — случайная величина с математическим ожиданием:

$$\sum_{i=1}^n \sum_{j=1}^m a_{ij} p_i q_j,$$

или в векторной записи $\mathbf{P} \mathbf{A} \mathbf{Q}^T$. Применение оптимальной стратегии позволяет получить выигрыш, равный цене игры v :

$$\alpha \leq v \leq \beta. \quad (1.42)$$

Для оптимальных стратегий имеет место соотношение:

$$\max_P \min_Q \mathbf{P} \mathbf{A} \mathbf{Q}^T = \min_Q \max_P \mathbf{P} \mathbf{A} \mathbf{Q}^T. \quad (1.43)$$

Применение игроком A стратегии оптимальной стратегии \mathbf{P}^* должно обеспечить ему при любых действиях игрока B математическое ожидание выигрыша не меньше цены игры v . Поэтому выполняются соотношения:

$$\sum_{i=1}^n p_i^* a_{ij} \geq v, \quad j = 1, 2, \dots, m. \quad (1.44)$$

Аналогично для игрока B оптимальная стратегия \mathbf{Q}^* должна обеспечить ему при любых действиях игрока A проигрыш, не превышающий цену игры v , т. е. справедливо соотношение:

$$\sum_{j=1}^m q_j^* a_{ij} \leq v, \quad i = 1, 2, \dots, n. \quad (1.45)$$

Соотношения (1.44) и (1.45) могут быть использованы при построении формальных математических моделей для определения решения игры.

Рассмотрим, как выглядит модель выбора для определения стратегии при использовании соотношения (1.44). Введём новые переменные $p_i/v = t_i$, $i = \overline{1, n}$. Тогда (1.44) перепишется следующим образом:

$$\sum_{i=1}^n t_i a_{ij} \geq 1, \quad j = \overline{1, m}, \quad (1.46)$$

а из условия $\sum_{i=1}^n p_i = 1$ следует, что

$$\sum_{i=1}^n t_i = \frac{1}{v}. \quad (1.47)$$

Так как решение игры должно максимизировать значение v , то для выбора оптимальной смешанной стратегии игрока A необходимо найти:

$$\min \sum_{i=1}^n t_i \quad (1.48)$$

при $\sum_{i=1}^n t_i a_{ij} \geq 1, j = \overline{1, m}, i = \overline{1, n}$.
 $t_i \geq 0$,

Аналогично может быть сформулирована задача для выбора оптимальной смешанной стратегии игрока B с использованием (1.45):

найти $\max \sum_{j=1}^m u_j \quad (1.49)$

при $\sum_{j=1}^m u_j a_{ij} \leq 1, i = \overline{1, n}$;

$$u_j \geq 0, u_j = \frac{q_j}{v}, j = \overline{1, m}.$$

Как видно из соотношения (1.48) и (1.49) для решения игры имеем пару двойственных задач линейного программирования. Используя свойство симметричности, можно решить одну из них, требующую меньших вычислений, а решение второй задачи найти, используя результаты теории двойственности (см. главу 2).

Пример 1.7. [25]

Рассматривается игра в хоккей. B одном из периодов состояния команд A и B характеризуется следующим образом:

- у команды A — 4 играющих пятёрки,
- у команды B — 3 играющих пятёрки.

Матрица игрового преимущества имеет вид:

$$A = \begin{pmatrix} & \overbrace{\beta_1 \quad \beta_2 \quad \beta_3} \\ \alpha_1 & \begin{pmatrix} 8 & 2 & 4 \\ 6 & 3 & 6 \\ 2 & 7 & 6 \\ 1 & 7 & 3 \end{pmatrix} \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{pmatrix}.$$

Задача (1.48) представлена в виде следующей модели:

$$\min (t_1 + t_2 + t_3 + t_4)$$

при

$$\left. \begin{array}{l} 8t_1 + 6t_2 + 2t_3 + t_4 \geq 1 \\ 2t_1 + 3t_2 + 7t_3 + 7t_4 \geq 1 \\ 4t_1 + 6t_2 + 6t_3 + 3t_4 \geq 1 \end{array} \right\} t_i \geq 0.$$

Оптимальное решение задачи методом линейного программирования (см. главу 2) выглядит следующим образом:

$$t_1^* = 1/32; \quad t_2^* = 3/32; \quad t_3^* = 3/32; \quad t_4^* = 0; \quad v^* = 7/32.$$

Таким образом, цена игры равна $32/7$, а оптимальные вероятности использования чистых стратегий в смешанной стратегии тренеров команды A определились следующим образом:

$$p_1^* = 1/7; \quad p_2^* = 3/7; \quad p_3^* = 3/7; \quad p_4^* = 0.$$

Поэтому в данном игровом периоде тренеру команды A необходимо реже выпускать первую пятёрку, а четвёртую вообще не включать в игру.

Аналогично может быть сформулирована оптимальная стратегия и для команды B .

Содержание предыдущих разделов убеждает в том, что среди математических методов, используемых в теории принятия решений, особую роль играют методы решения оптимизационных задач. Они составляют фундамент системы математического обеспечения проблем принятия решений. Этим вопросам будет посвящено содержание следующего раздела.

1.8. ПРИМЕРЫ ПОСТРОЕНИЯ ОПЕРАЦИОННЫХ МОДЕЛЕЙ

Пример 1.8.1. Линейные модели

Транспортная задача

Предположим, что однородный продукт должен быть перевезен из m пунктов производства в n пунктов потребления, причем из каждого пункта производства вывозится установленное количество и в каждый

пункт потребления ввозится также установленное количество продукта, так что общие спрос и предложение равны. Пусть из i -го пункта производства ($i = 1, 2, \dots, m$) вывозится a_i продукта, а в j -й пункт потребления должно быть ввезено b_j продукта, причем величины эти известны. Тогда:

$$\sum_{i=1}^m a_i = \sum_{j=1}^n b_j , \quad (1.50)$$

где $a_i, b_j \geq 0$ для всех $i \neq j$.

Пусть x_{ij} — неизвестное количество продукта, которое должно быть перевезено из i -го пункта производства в j -й пункт потребления.

Тогда:

$$\sum_{j=1}^n x_{ij} = a_i, \quad i = \overline{1, m}, \quad (1.51)$$

$$\sum_{i=1}^m x_{ij} = b_j, \quad j = \overline{1, n}, \quad (1.52)$$

где $x_{ij} \geq 0$ для всех i и j .

Пусть стоимость перевозки из i -го пункта производства в j -й пункт потребления равна c_{ij} . Эти величины также заданы. Задача состоит в том, чтобы найти x_{ij} , удовлетворяющие приведенным выше ограничениям (1.51), (1.52) и минимизирующие

$$\sum_{j=1}^n \sum_{i=1}^m c_{ij} x_{ij} . \quad (1.53)$$

Задача поставщика

Владелец кафе знает, что в связи с наступлением праздников в течение одной следующей недели ему в j -й день ($i = \overline{1, 7}$) потребуется r_j чистых салфеток, см. табл. 1.6.

Таблица 1.6

1 Понед.	2 Вторник	3 Среда	4 Четверг	5 Пятница	6 Суббота	7 Воскр.
5	6	7	8	7	9	10

На стирку обычно требуется три дня, т. е. грязная салфетка отправленная в стирку в j -й день, вернется из стирки и может быть использована в $(j+3)$ -й день. Однако в прачечной существует ускоренное об-

служивание по повышенному тарифу, при котором салфетки возвращаются из стирки через два дня. Не имея на руках или в прачечной готовых к употреблению салфеток, владелец будет вынужден покупать салфетки по 25 рублей за штуку.

Стоимость стирки 10 и 15 рублей соответственно при обычном обслуживании и при ускоренном обслуживании. Как должен вести дело владелец кафе, чтобы удовлетворить потребность кафе в салфетках и минимизировать издержки за неделю?

Построим математическую модель. Пусть $x_i, (i = \overline{1, 7})$ количество новых салфеток, купленных в соответствующий день недели, Аналогично, пусть $y_i, z_i, (i = \overline{1, 7})$ — количество салфеток, отправленных в стирку по повышенному или по обычному тарифу. Наконец, пусть $u_i, (i = \overline{1, 7})$ — количество грязных салфеток, не отправленных в стирку в соответствующий день.

Так как необходимо минимизировать общую сумму, требуемую для поддержания количества салфеток на должном уровне, не следует покупать салфеток больше, чем требуется в данный день, или отправлять салфетку в прачечную, если она не будет использована в дальнейшем.

В первый день, т. е. в понедельник, необходимо купить такое количество салфеток, какое требуется именно в этот день, т. е. $x_1 = 5$.

В этот день можно выбирать: послать ли все или некоторые из пяти грязных салфеток в быструю, медленную прачечную или оставить их в ящике для грязных салфеток. То, что происходит с этими пятью салфетками можно представить уравнением:

$$y_1 + z_1 + u_1 = 5. \quad (1.54)$$

Общая стоимость всех операций первого дня:

$$25x_1 + 15y_1 + 10z_1. \quad (1.55)$$

Выпишем остальные ограничения данной модели, помня о том, что в нашей системе выстиранные салфетки возвращаются через 2 или 3 дня в зависимости от того, какой прачечной мы воспользовались.

Во вторник необходимо купить некоторое количество салфеток, а именно $x_2 = 6$. После того, как их используют, мы поступим с ними и u_1 грязными салфетками согласно уравнению:

$$y_2 + z_2 + u_2 = 6 + u_1. \quad (1.56)$$

Оно выражает тот факт, что у нас скопилось u_1 грязных салфеток, оставшихся с понедельника и еще 6, оставшихся со вторника. Можно их

отправить в стирку $(y_2 + z_2)$, либо оставить в ящике с грязными салфетками (u_2) . Стоимость всех операций во вторник:

$$25x_2 + 15y_2 + 10z_2. \quad (1.57)$$

В среду нам потребуется 7 салфеток. Это первый день, когда выстиранные салфетки из быстрой стирки можно употребить в дело. Поэтому нужные 7 салфеток можно получить, купив их или взяв их из того количества салфеток, которые были сданы в стирку в понедельник. Имеем:

$$x_3 + y_1 = 7. \quad (1.58)$$

Так же, как и ранее

$$y_3 + z_3 + u_3 = 7 + u_2, \quad (1.59)$$

а стоимость операции в среду равна:

$$25x_3 + 15y_3 + 10z_3. \quad (1.60)$$

Необходимые в четверг 8 салфеток могут быть либо новыми, либо выбранными из числа тех, которые мы посыпали в быструю прачечную во вторник или в медленную прачечную в понедельник. Это можно записать как:

$$x_4 + y_2 + z_1 = 8. \quad (1.61)$$

$$y_4 + z_4 + u_4 = 8 + u_3, \quad (1.62)$$

а стоимость операций в четверг составит:

$$25x_4 + 15y_4 + 10z_4.$$

Предположим, что нас интересует только одна неделя и, следовательно, мы не будем отдавать салфетки в стирку, если они не вернутся назад к воскресенью.

Для пятницы имеем:

$$x_5 + y_3 + z_2 = 7. \quad (1.63)$$

$$y_5 + u_5 = 7 + u_4 \quad (1.64)$$

а стоимость равна:

$$25x_5 + 15y_5. \quad (1.65)$$

Для субботы:

$$x_6 + y_4 + z_3 = 9, \quad (1.66)$$

$$u_6 = 9 + u_5,$$

а стоимость равна $25x_6$.

Для воскресенья имеем:

$$x_7 + y_5 + z_4 = 10, \quad (1.67)$$

$$u_7 = 10 + u_6,$$

а стоимость равна $25x_7$.

Объединяя полученные выше уравнения, получаем математическую модель операции:

$$\min_{(x_i, y_i, z_i, u_i)} \left[25 \sum_{i=1}^7 x_i + 15 \sum_{i=1}^5 y_i + 10 \sum_{i=1}^4 z_i \right]$$

при условиях:

$$\begin{cases} x_1 = 5; \\ x_2 = 6; \\ x_3 + y_1 = 7; \\ x_4 + y_2 + z_1 = 8; \\ x_5 + y_3 + z_2 = 7; \\ x_6 + y_4 + z_3 = 9; \\ x_7 + y_5 + z_4 = 10; \\ y_1 + z_1 + u_1 = 5; \\ y_2 + z_2 + u_2 = 6 + u_1; \\ y_3 + z_3 + u_3 = 7 + u_2; \end{cases} \quad \begin{cases} x_i \geq 0; \\ y_i \geq 0; \\ u_i \geq 0; \\ z_i \geq 0. \end{cases} \quad (1.68)$$

Пример 1.8.2. Нелинейные модели

Задача оптимального управления с непрерывным временем

Рассмотрим задачу о запуске ракеты с поверхности земли до высоты, которую ракета должна достичь за время T . Обозначим через $y(t)$ высоту, достигнутую ракетой за время t , а через $u(t)$ — силу действующую на ракету в вертикальном направлении в момент t . Пусть масса ракеты равна m . Тогда уравнение движения имеет вид:

$$m\ddot{y}(t) + mg = u(t), \quad t \in [0, T], \quad (1.69)$$

где $\ddot{y}(t)$ — ускорение движения ракеты в момент времени t , g — ускорение свободного падения.

Будем считать, что максимальная сила, прилагаемая к ракете в любой момент, не превосходит b . Требуется вычислить минимальную энергию, которую необходимо затратить для выведения ракеты на высоту \bar{y} за время T .

Эта задача может быть сформулирована следующим образом:

$$\min \int_0^T |u(t)| dt;$$

при условиях:

$$\begin{cases} \ddot{y}(t) + g = \frac{1}{m} u(t), \quad t \in [0, T]; \\ |u(t)| \leq b, \quad t \in [0, T]; \\ y(T) = \bar{y}; \\ y(0) = 0. \end{cases} \quad (1.70)$$

Уравнение $\ddot{y}(t) + g = \frac{1}{m} u(t)$, эквивалентно системе уравнений:

$$\begin{aligned} y_1 &= y_2; \\ \dot{y}_2 + g &= \frac{1}{m} u. \end{aligned} \quad (1.71)$$

Таким образом, задача (1.70) принимает вид:

$$\min \int_0^T |u(t)| dt$$

при условиях:

$$\begin{cases} \dot{y}_1 = y_2, \quad t \in [0, T]; \\ \dot{y}_2(t) = \frac{1}{m} u(t) - g, \quad u(t) \leq b, \quad t \in [0, T]; \\ y_1(0) = y_2(0) = 0, \quad y_1(T) = \bar{y}, \quad y_2(T) = 0. \end{cases}$$

Разделим отрезок $[0, T]$ на K интервалов длины Δ . Чтобы упростить запись, будем считать, что $\Delta = 1$. Обозначим силу, действующую на ракету, высоту полета над поверхностью и скорость ракеты в конце k -го периода соответственно через u_k , y_{1k} , y_{2k} . Тогда задача оптимального управления полетом ракеты можно аппроксимировать следующей задачей нелинейного программирования:

$$\min \sum_{i=1}^k |z_k|$$

при условиях:

$$\begin{cases} y_{1k} - y_{1k-1} = y_{2k-1}, \quad k = \overline{1, K}; \\ y_{2k} - y_{2k-1} = \frac{1}{m} u_k - g, \quad k = \overline{1, K}; \\ |u_k| \leq b; \\ y_{10} = y_{20} = 0; \\ y_{1K} = \bar{y}, \quad y_{2K} = 0. \end{cases} \quad (1.72)$$

Несмотря на простоту данного примера наглядно видны принципы и приемы сведения задач оптимального управления к задачам математического программирования.

Пример 1.8.3. Целочисленные модели

Задача о размещении

Рассмотрим какую-нибудь отрасль промышленности, выпускающую товары широкого потребления, которые прямо с заводов отправляются в торговые предприятия. Требуется ответить на вопрос, где разместить каждое из промышленных предприятий и каковы должны быть их производственные мощности. При этом известно местонахождение каждой торговой точки, занимающейся розничной продажей товаров данной отрасли промышленности и известны потребности каждой из этих точек.

Пусть b_1, b_2, \dots, b_n — объемы продукции, необходимые для покрытия потребностей « n » имеющихся торговых точек,

a_1, a_2, \dots, a_m — производственные мощности « m » заводов, о размещении которых идет речь.

Допустим, что известна стоимость каждого из заводов: стоимость строительства i -го завода обозначим через f_i . Расходы на доставку единицы продукции от i -го завода до j -й торговой точки равны c_{ij} . Предположим, что i -м заводом j -й торговой точке поставляется x_{ij} единиц продукции. Введем переменную y_i , такую, что $y_i = 1$, если i -й завод решено построить; $y_i = 0$, если i -й завод решено не строить.

Задача заключается в том, чтобы минимизировать сумму расходов на строительство заводов и доставку товара торговым точкам при полном удовлетворении спроса потребителей.

Математическая модель задачи выглядит следующим образом:

$$\begin{aligned} & \min \sum_{i=1}^m [f_i y_i + \sum_{j=1}^n c_{ij} x_{ij}] \\ & \sum_{i=1}^m x_{ij} \geq b_j, \quad j = \overline{1, n}; \\ & \sum_{j=1}^n x_{ij} \leq a_i y_i, \quad i = \overline{1, m}; \\ & \text{при условиях: } \\ & y_i = 0 \text{ или } 1, \quad i = \overline{1, m}; \\ & x_{ij} > 0, \quad (i = \overline{1, m}, \quad i = \overline{1, n}). \end{aligned} \tag{1.73}$$

Если было бы заранее известно, какие заводы и где решено построить, все свелось бы к решению обычной транспортной задачи (пример 1.8.1).

Задача о водопроводчике

Водопроводчик получил наряд на установку вентилей на нескольких трубах, проложенных под землей, покрытой тяжелыми квадратными плитами (рис. 1.3). Он может установить механизмы перекрытия в любом месте трубы, но, разумеется, лишь по одному шлюзовому затвору на каждой трубе.

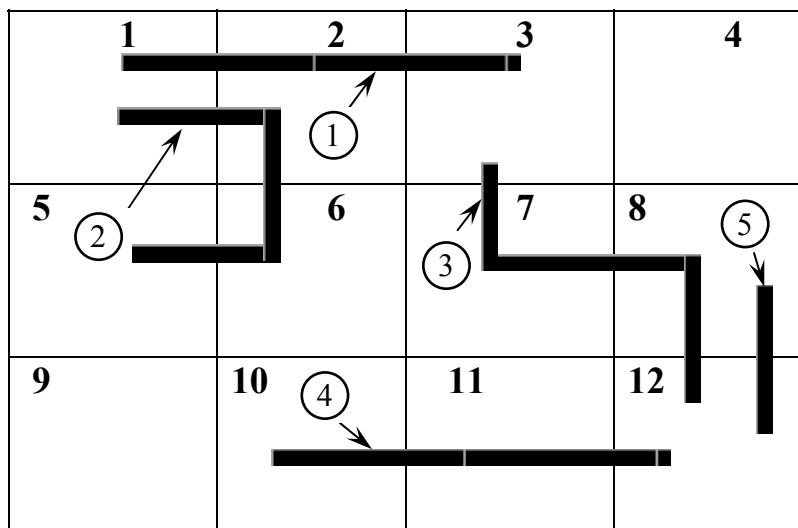


Рис. 1.3.

Для минимизации трудозатрат водопроводчику надо определить минимальное число плит, которые требуется приподнять, чтобы установить по одному затвору на каждой трубе.

Пронумеруем плиты от 1 до 12. Пусть $x_i = 0$, если выбранное решение состоит в том, чтобы не приподнимать i -ю плиту, $x_i = 1$, если принимается решение приподнять i -ю плиту ($i = 1, 12$). Каждой трубе соответствует одно ограничение, которое означает, что для получения к ней доступа необходимо приподнять, по крайней мере, одну из плит.

Математическая модель задачи выглядит следующим образом:

$$\min \sum_{i=1}^{12} x_i$$

при условиях:

$$\begin{cases} x_1 + x_2 + x_3 \geq 1; \\ x_1 + x_2 + x_5 + x_6 \geq 1; \\ x_3 + x_7 + x_8 + x_{12} \geq 1; \\ x_{10} + x_{11} + x_{12} \geq 1; \\ x_8 + x_{12} \geq 1; \\ x_j = 0 \text{ или } 1; \quad i=1,12. \end{cases} \quad (1.74)$$

Эта задача попадает в класс так называемых задач о «покрытии множества», для которого характерны разнообразные практические приложения.

Пример 1.5.4. Стохастические модели

Задача о загрузке судна запасными деталями

Пусть рассматривается три типа запасных деталей, имеющих объемы 1, 2, и 2 единицы. Общий объем склада равен 10 единицам. Штрафные затраты π_j , возникающие при неудовлетворении потребности, равны соответственно 800, 600, и 1300. Будем предполагать, что спрос v_j на каждую из запасных деталей подчинен пуассоновскому распределению со средними значениями $\mu_j = (4; 2; 1)$. Соответственно требуется определить, каким должен быть запас деталей каждого типа, чтобы средние штрафные издержки были минимальными. В математической формулировке мы приходим к следующей модели:

найти $\min_{(X)} \sum_{j=1}^3 \pi_j \sum_{v_j=x_j}^{\infty} (v_j - x_j) P(v_j, \mu_j)$ (1.75)

при условиях: $x_j \geq 0$, целые, $j = 1, 2, 3$;

$$x_1 + 2x_2 + 2x_3 \leq 10,$$

где x_j — запасы деталей j -го типа.

Выражение $P(v_j, \mu_j) = \frac{\mu_j^{v_j}}{v_j!} e^{-\mu_j}$ (1.76)

в функции (1.75) представляет собой вероятность случайной величины v_j , подчиненной закону Пуассона, а μ_j есть ожидаемая средняя потребность в деталях j -го типа: $\mu_j = \sum_{v_j=0}^{\infty} v_j P(v_j, \mu_j)$.

Из (1.76) следует, что

$$\begin{aligned} v_j P(v_j, \mu_j) &= \mu_j P(v_j - 1, \mu_j) \\ v_j &\geq 1 \end{aligned} . \quad (1.77)$$

Таким образом, можно записать:

$$\sum_{v_j=x_j}^{\infty} (v_j - x_j) P(v_j, \mu_j) = \begin{cases} \mu_j P(x_j - 1, \mu_j) - x_j P(x_j, \mu_j), & x_j \geq 1 \\ 0, & \mu_j, x_j = 0, \end{cases} \quad (1.78)$$

где

$$P(x_j, \mu_j) = \sum_{v_j=x_j}^{\infty} P(v_j, \mu_j).$$

Соотношение (1.78) дает возможность без труда вычислить средний спрос на j -ю деталь, возникающий в то время, когда запас их исчерпан, пользуясь таблицами для распределения Пуассона.

ЗАДАЧИ

1. Диспетчерская служба имеет минимальные потребности в количестве диспетчеров в различное время суток (табл. 1.7). При этом нужно иметь в виду, что период 1 следует сразу же за периодом 6. Каждый диспетчер ежедневно приступает к работе в начале определенного периода и работает восемь часов без перерыва. Требуется составить расписание на каждые сутки таким образом, чтобы обойтись минимальным числом диспетчеров, не нарушая, при этом, сформулированных выше требований.

Таблица 1.7

Порядковый номер периода	1	2	3	4	5	6
Время суток час.	2 – 6	6 – 10	10 – 14	14 – 18	18 – 22	22 – 2
Минимальное число диспетчеров, требуемое в указанный период	20	50	80	100	40	30

2. В обработку поступили две партии досок для изготовления комплектов из трех деталей, причем первая партия содержит 50 досок длиной по 6,5 м, вторая содержит 200 досок длиной 4 м. Каждый комплект состоит из двух деталей по 2 м каждая и одной детали по 1,25 м. Как распилить доски, чтобы получить наибольшее число комплектов?

3. На заводе предстоит решить, какое количество x_1 чистой стали и какое количество x_2 металлома следует использовать для приготовления (из соответствующего сплава) литья для одного из своих заказчиков. Пусть производственные затраты на 1 т стали составляют 3 усл. ед., а затраты на 1 т металлома 5 усл. ед. (последняя цифра больше предыдущей, т.к. использование металлома связано с его предварительной очисткой).

Заказ предусматривает поставку не менее 5 т литья. Предположим, что предназначенные для литья запасы чистой стали составляют 4 т, а металлома — 6 т. Отношение веса металлома к весу чистой стали в сплаве не должно превышать 7/8.

Производственно-технологические условия таковы, что на процессы плавки и литья может быть отведено не более 18 час., при этом на 1 т стали затрачивается от 2,5 до 3 час., а на 1т металлома — от 1,5 до 2 час.

Цель завода — выполнить заказ с минимальными производственными затратами.

4. Предприятие выпускает радиоприемники трех различных моделей: модель **A**, модель **B** и модель **C**. Каждое изделие указанных моделей приносит доход в размере 8, 15, 25 единиц стоимости соответственно. Необходимо, чтобы фирма выпускала не менее 100 приемников модели **A**, 150 приемников модели **B** и 75 приемников модели **C**.

Каждая модель характеризуется определенным временем, необходимым для изготовления соответствующих деталей, сборки изделия и его упаковки. Так, в частности, в расчете на 10 приемников модели **A** требуется 3 ч. Для изготовления соответствующих деталей, 4 ч. На сборку и 1 ч. На упаковку. Соответствующие показатели на 10 приемников модели **B** равняются 3, 5; 5 и 1,5 ч., а на 10 приемников модели **C** — 5, 8 и 3.

В течение ближайшей недели фирма может израсходовать на производство — 150 ч., на сборку — 200 ч. И на упаковку — 60 ч.

Составить производственный план.

5. На предприятии требуется произвести раскрой рулона материала шириной 60 см. Данные о заказах недели представлены в табл. 1.8.

Таблица 1.8

Требуемая ширина (см)	28	20	15
Требуемое количество рулонов	30	60	48

Предполагается, что количество рулонов с шириной 60 см достаточно для того, чтобы удовлетворить все недельные заказы. Найти план раскроя, минимизирующий общие суммарные потери.

6. В небольшом населенном пункте A имеется школа, которую посещает некоторое число учеников, при этом место жительства 72 учеников находится вне населенного пункта, что приводит к необходимости организовать их доставку к школе на автобусах. Имеются две основные автобусные остановки B , C (B находится между A и C). Число учеников, нуждающихся в доставке к школе на автобусе равняется 42 на остановке C , 6 — между C и B , 20 на остановке B и 4 — между B и A (рис. 1.4).

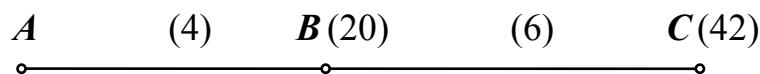


Рис. 1.4

Транспортное агентство, обслуживающее пункт A , располагает двумя типами автобусов: автобусы на 35 и 50 мест.

Агентством установлены следующие цены проездных билетов для каждого из отрезков пути в зависимости от типа автобуса: табл. 1.9.

Таблица 1.9

Отрезок пути	BA	CA	CB
35 мест	39,0	54,0	45,0
50 мест	50,5	68,0	57,5

Необходимо определить, какого типа автобусы следует использовать на каждом участке пути так, чтобы суммарные издержки школьников были минимальными.

7. Определить максимальный поток f из пункта O в пункт M при ограниченной пропускной способности путей. Граф путей представлен на рис. 1.5.

Числа над дугами графа определяют верхнюю границу потока на соответствующих путях.

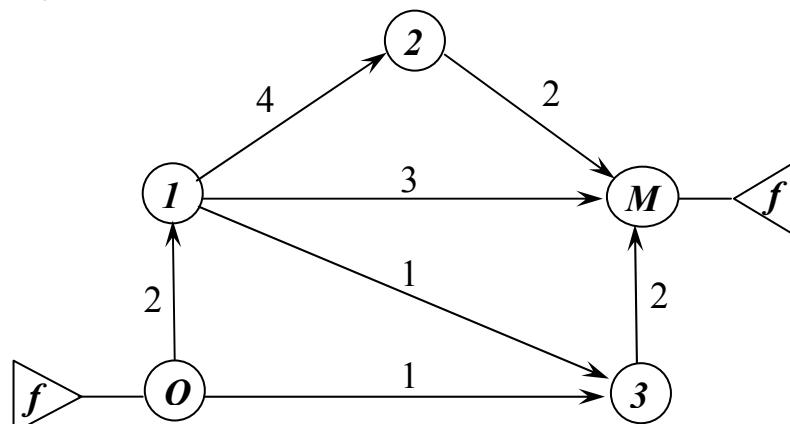


Рис. 1.5

8. Информация о проекте задана перечнем работ, их продолжительностью и последовательностью выполнения (табл. 1.10):

Таблица 1.10

Работа	1	2	3	4	5	6	7	8
Каким работам предшествует	4, 5, 6	4, 5, 6	5, 6	8	8	7	8	—
Продолжительность мес.	20	10	8	20	10	5	5	10

В i -ю работу можно вложить средства, где $x_i \leq c_i$, при этом время выполнения работы уменьшится до $\tau_i(1 - b_i x_i)$.

Пусть: $b_1 = 0,2$; $c_1 = 2,0$; $b_4 = 0,3$; $c_4 = 2,0$; $b_8 = 0,1$; $c_8 = 5$.

Определить размер вложенных средств x_1, x_4, x_8 в 1-ю, 4-ю, 8-ю работы так, чтобы время завершения всего комплекса работ не превышало 40 месяцев.

Сформулировать общую задачу минимизации суммы вложений, необходимых для того, чтобы обеспечить завершение всего комплекса работ к заданному моменту времени T , как задачу линейного программирования.

9. Участник экспедиции укладывает рюкзак, и ему требуется решить, какие положить продукты. В его распоряжении имеются мясо, мука, сухое молоко и сахар. В рюкзаке для продуктов осталось лишь 45 дм^3 объема, и нужно, чтобы суммарная масса продуктов не превосходила 35 кг. Врач экспедиции рекомендовал, чтобы мяса (по массе) было больше муки по крайней мере в два раза, муки не меньше молока, а молока по крайней мере в восемь раз больше, чем сахара. Сколько и каких продуктов нужно положить в рюкзак, с тем, чтобы суммарная калорийность продуктов была наибольшей? Характеристики продуктов приведены в табл.1.11.

Таблица 1.11

Характеристики	Продукты			
	мясо	мука	молоко	сахар
Объем ($\text{дм}^3/\text{кг}$)	1,0	1,5	2,0	1,0
Калорийность (ккал/кг)	1500	5000	5000	4000

10. Организация хранит различную информацию, записанную на магнитных носителях. Организации необходимо составить статистические данные по различным демографическим характеристикам (например, как

распределение населения по возрастным группам, распределение доходов, обеспеченность жильем и т. п. по выбранным восьми городским районам.

Предположим, что вся требуемая информация хранится в n массивах. Примем, что $\tau_j, (j=1, 2, \dots, n)$ — время поиска ЭВМ нужной информации в массиве j и что время поиска не зависит от числа характеристик, выбираемых из массива. Некоторые из m видов данных записаны более чем в одном из n массивов, т.е. различные массивы содержат информацию, относящуюся к одной и той же характеристике. Расположение информации задано матрицей $A = \{a_{ij}\}_m^n$, где $a_{ij} = 1$, если данные по i -й характеристике записаны в j -м массиве и $a_{ij} = 0$ в противном случае.

Постройте модель для определения того, в каких из n массивов производить поиск, чтобы выбрать данные, относящиеся ко всем требуемым статистическим характеристикам за минимальное время.

Объясните, как следует изменить модель, если время поиска в массиве j равно T_j плюс t_{ij} в случае, когда в результате поиска получают характеристику с индексом i .

11. Для производства комплексной продукции требуется изготовить два вида изделий. Их изготовление может быть поставлено на каждом из пяти типов предприятий. Производственная мощность предприятия и количество предприятий каждого типа даны в табл. 1.12.

Определить, сколько предприятий каждого типа надо поставить на производство первого и сколько на производство второго изделия, чтобы обеспечить максимальный выпуск комплектов, если в каждый комплект должно входить два изделия первого вида и одного второго.

Таблица 1.12

Тип предприятия	Число предприятий	Производственная мощность	
		по изделиям № 1 (тыс.)	по изделиям № 2 (тыс.)
№ 1	5	100,0	15,0
№ 2	3	400,0	200,0
№ 3	40	20,0	2,5
№ 4	9	200,0	50,0
№ 5	2	600,0	250,0

12. На мебельной фабрике требуется раскроить 5000 прямоугольных листов фанеры размером 4×5 м каждый с тем, чтобы получить два вида прямоугольных деталей: деталь **A** должна иметь размер 2×2 м, деталь **B** — размер 1×3 м. Необходимо, чтобы деталей **A** оказалось не меньше, чем деталей **B**. Каким образом следует произвести раскрой, чтобы получить минимальное (по площади) количество отходов?

13. Построить математическую модель для выбора расписания при следующих условиях:

задан состав операций:

a_1, a_2, \dots, a_{10} ;

заданы временные функции:

t_1, t_2, \dots, t_{10} ;

заданы ограничения на порядок следования: $a_1 \prec a_2, a_8 \prec a_5, a_7 \prec a_8$;

некоторые операции нельзя выполнять одновременно: $(a_1 \text{ и } a_2), (a_5 \text{ и } a_7)$;

целевая функция — минимизация общей длительности.

14. Для строительства домов на 100 строительных площадках выбраны 5 типовых проектов. По каждому из проектов известны длительность закладки фундаментов, возведения стен и крыш, отделка, а также жилая площадь дома. Возможно, исходя из трудовых, сырьевых и технических ресурсов одновременное ведение закладки 10 фундаментов и строительства и отделки 15 зданий (табл. 1.13).

Составить план строительства, максимизирующий ввод жилой площади в течение года (300 рабочих дней) при условии, что домов II типа должно быть построено не меньше 10.

Составить план строительства, максимизирующий ввод жилой площади в течение года (300 рабочих дней) при условии, что домов II типа должно быть построено не меньше 10.

Таблица 1.13

Вид работ	Длительность ведения работ (дни)				
	I	II	III	IV	V
Закладка фундамента	20	30	35	30	40
Строительство	30	15	40	25	15
Отделка	10	5	20	10	10
Жилая площадь м^2	3000	2000	5000	4000	6000

15. Автозавод выпускает две модели автомобилей: модель **A** и более дешевую модель **B**. На заводе работает 1000 неквалифицированных и 800 квалифицированных рабочих, каждому из которых оплачивается 40 часов

в неделю. Для выпуска модели A требуется 30 ч. неквалифицированного и 50 ч. квалифицированного труда. Для модели B — 40 ч. и 20 ч. соответственно.

Каждая из моделей требует затрат на сырье в размере 500 руб. и 1500 руб. Суммарные затраты не должны превышать 900 000 руб. в неделю. Рабочие, осуществляющие доставку, работают по 5 дней в неделю и могут забрать с завода не более 210 машин в день. Каждая модель A (B) приносит прибыль 1000 (500) руб.

Каким должен быть объем выпуска каждой модели?

16. Фирма производит 2 модели A и B сборных книжных полок. Их производство ограничено наличием сырья (досок) и временем машинной обработки. Для каждого изделия A требуется 3 м^2 досок, для B — 4 м^2 . Фабрика может получить от поставщиков 1700 м^2 досок в неделю. Для каждого изделия A (B) требуется 12 (30) мин. машинного времени. В неделю можно использовать не более 160 час. машинного времени.

Определить план недельного выпуска, максимизирующий прибыль, если изделие A (B) приносит денежную прибыль 2 (4) усл. ед.

17. В некоторой местности в пунктах A и B имеются потребности в дополнительном транспорте. В пункте A требуется 5 дополнительных автобусов, в пункте B — 7. Известно, что 3, 4, 5 автобусов соответственно могут быть получены из гаражей G_1 , G_2 , G_3 .

Как следует распределить эти автобусы между пунктами A и B , чтобы минимизировать их суммарный пробег. Расстояния между гаражами и пунктами A и B заданы табл. 1.14.

Таблица 1.14

Гараж	Расстояние до пунктов	
	A	B
G_1	3	4
G_2	1	3
G_3	4	2

18. Фирма производит два продукта A и B , продаваемых соответственно по 8 и по 15 руб. за упаковку. Рынок сбыта для каждого из них не ограничен.

Продукт A обрабатывается на машине 1, продукт B — на машине 2, затем оба упаковываются на фабрике. 1 кг сырья стоит 6 копеек. Машина 1 (2) обрабатывает 5000 (4000) кг сырья в час с потерями 10 % (20 %). Ма-

шина 1 (2) работает 6 (5) часов в день, ее использование стоит 288 (336) руб/час. Упаковка продукта A (B) весит $1/4$ ($1/3$) кг. Фабрика может работать 10 час. в день, производя в 1 час продукцию стоимостью 360 руб. За 1 час можно упаковать 12 000 продуктов A и 8 000 продуктов B .

Компания хочет определить такие значения x_1 и x_2 потребления сырья для продуктов A и B (в тыс. кг), при которых дневная прибыль максимальна.

19. Для обеспечения автоперевозок в j -й день недели требуется a_j автомашин. Машины после поездки должны пройти профилактический ремонт. Обычный ремонт длится 4 дня при затратах 20 руб. на машину, срочный ремонт длится 2 дня при затратах 30 руб. на машину. Кроме того, можно использовать машины для перевозок, сняв их с другого участка, что приводит к потерям 50 руб. на машину. Определить оптимальную недельную программу подготовки машин, минимизирующую суммарные затраты автобазы, если потребность в машинах характеризуется следующими данными (табл. 1.15).

Таблица 1.15

Дни недели	1	2	3	4	5	6	7
a_j	50	40	70	60	80	40	50

Для решения использовать пример задачи о поставщике.

2. ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

2.1. ЗАДАЧА ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ И ПРИВЕДЕНИЕ ЕЕ К КАНОНИЧЕСКОЙ ФОРМЕ

Задача линейного программирования (ЗЛП) формулируется как задача поиска экстремума линейной функции при наличии системы линейных неравенств, ограничивающих область изменения аргументов этой функции: найти максимум (минимум)

$$f(X) = \sum_{j=1}^n c_j x_j \quad (2.1)$$

при наличии системы ограничений, заданных в следующих вариантах или их комбинации:

$$\text{а)} \quad \sum_{j=1}^n a_{ij} x_j \leq b_i, \quad i = \overline{1, m_1}; \quad (2.2)$$

$$\text{б)} \quad \sum_{j=1}^n a_{ij} x_j \geq b_j, \quad i = \overline{m_1 + 1, m_2}; \quad (2.3)$$

$$\text{в)} \quad \sum_{j=1}^n a_{ij} x_j = b_i, \quad i = \overline{m_2 + 1, m_3}; \quad (2.4)$$

$$\text{г)} \quad \alpha_j \leq x_j \leq \beta_j, \quad j = \overline{1, n_1}; \quad (2.5)$$

$$\text{д)} \quad -\infty < x_j < \infty, \quad j = \overline{n_1 + 1, n}. \quad (2.6)$$

Процедура решения ЗЛП начинается с приведения ее к канонической форме, то есть к стандартной форме задания, ориентированной на разработанный именно для этой формы метод решения.

Наибольшее распространение получила приведенная ниже в скалярном (2.7), (2.8) и матричном (2.9) виде каноническая форма ЗЛП, к которой всегда можно свести задачу поиска экстремума линейной функции при ограничениях типа (2.2) – (2.6):

$$\left\{ \begin{array}{l} \max \sum_{j=1}^n c_j x_j; \\ \sum_{j=1}^n a_{ij} x_j = b_i, \quad b_i \geq 0, \quad i = \overline{1, m}; \\ x_j \geq 0, \quad j = \overline{1, n}. \end{array} \right. \quad (2.8)$$

$$\left\{ \begin{array}{l} \max C^T X; \\ AX = b; \\ X \geq 0, \quad b \geq 0. \end{array} \right. , \quad (2.9)$$

где $C = (c_1, c_2, \dots, c_n)^T$; $X = (x_1, x_2, \dots, x_n)^T$;

$b = (b_1, b_2, \dots, b_n)^T$; $A = \{a_{ij}\}$, $i = \overline{1, m}$; $j = \overline{1, n}$.

Если исходная задача сформулирована как задача минимизации, ее всегда можно представить как задачу поиска максимума той же функции, умноженной на (-1) .

Приведение системы ограничений, заданных в форме неравенств (2.2), (2.3), (2.6) к канонической форме (2.8) равенств осуществляются посредством соответствующего увеличения размерности вектора X с учетом обязательной неотрицательности всех его составляющих:

$$\sum_{j=1}^n a_{ij} x_j \leq b_i, \quad i = \overline{1, m} \rightarrow \sum_{j=1}^n a_{ij} x_j + x_{n+i} = b_i, \quad x_{n+i} \geq 0,$$

$$\sum_{j=1}^n a_{ij} x_j \geq b_i, \quad i = \overline{1, m} \rightarrow \sum_{j=1}^n a_{ij} x_j - x_{n+i} = b_i, \quad x_{n+i} \geq 0,$$

$$\alpha_j \leq x_j \leq \beta_j \rightarrow \begin{cases} x'_j \geq 0, & \text{где } x'_j = \beta_j - x_j; \\ x''_j \geq 0, & \text{где } x''_j = x_j - \alpha_j; \end{cases}$$

$$-\infty \leq x_j \leq \infty \rightarrow x_j = x_{j_1} - x_{j_2}, \quad x_{j_1}, x_{j_2} \geq 0.$$

2.2. СОДЕРЖАНИЕ АЛГОРИТМА СИМПЛЕКС-МЕТОДА

Задача линейного программирования в канонической форме имеет смысл при условии $n > m$. В этом случае система (2.8) описывает область допустимых решений (ОДР) ЗЛП, геометрически являющуюся выпуклым

многогранником в евклидовом пространстве R^n [1, 2, 3]. Выпуклая фигура, как известно, характеризуется тем свойством, что, если две точки X_1 и X_2 принадлежат этой фигуре, то и весь отрезок X_1X_2 также принадлежит ей. Кроме того, доказано, что оптимальное решение ЗЛП всегда лежит на границе ОДР [1, 2, 3], поэтому справедлив вывод о том, что, по крайней мере, одна из угловых (опорных) точек выпуклого многогранника ОДР является точкой оптимума. Для того чтобы определить координаты опорной точки, все множество переменных $X = \{x_j\}, j = \overline{1, n}$ необходимо разделить на два подмножества $X = X^B \cup X^{\bar{B}}$:

- подмножество базисных переменных $X^B = (x_{j1}, \dots, x_{jm})^T$, при этом число m базисных переменных равно числу уравнений системы (2.8) при условии, что уравнения являются линейно-независимыми;
- подмножество $X^{\bar{B}}$ остальных $n - m$ свободных (или внебазисных) переменных $\{x_j\}, j \notin B$.

Количество возможных вариантов разделения переменных на базисные и свободные (число базисов) равно C_n^m .

Приравняем к нулю все свободные переменные, выделим в матрице A , используемой в уравнении (2.9), столбцы, соответствующие базисным переменным, и составим из них матрицу $P_{[m \times m]}$. С учетом (2.9) можем записать

$$PX^B = b \quad (2.10)$$

и, если система (2.10) имеет решение, то

$$X^B = P^{-1}b . \quad (2.11)$$

Оценим местонахождение точки с координатами

$$X^B = P^{-1}b, X^{\bar{B}} = 0 .$$

Эти координаты удовлетворяют уравнению (2.9), и если они также неотрицательны, т. е. $X^B = P^{-1}b \geq 0$, то данная точка принадлежит ОДР, а соответствующий ей набор базисных переменных (базис) называется допустимым. Тот факт, что в этой точке все свободные переменные принимают нулевые (т. е. граничные с учетом условия $X \geq 0$) значения, характеризует ее как лежащую на границе. Будем называть такие точки опорными.

Таким образом, признаком допустимости базиса является условие:

$$X^B = P^{-1}b \geq 0 .$$

Наиболее очевидный метод решения ЗЛП состоит в том, чтобы для

каждого из C_n^m базисов найти координаты соответствующих опорных точек, выделить из них точки, принадлежащие ОДР, а затем из них, в свою очередь, выбрать ту, координаты которой максимизируют целевую функцию (2.1). В отличие от этого метода, реализующего по сути, идею полного перебора опорных точек ОДР, может быть предложен более эффективный так называемый симплекс-метод решения ЗЛП.

В основе симплекс-метода лежит подход, включающий:

- выбор опорной точки, принадлежащей ОДР (выбор начального допустимого базиса);
- проверку опорной точки на оптимальность;
- выбор нового базиса, позволяющего минимизировать число опорных точек на траектории в случае невыполнения условий оптимальности.

2.3. ПОИСК ДОПУСТИМОГО БАЗИСА

Поиск допустимого базиса начинается с анализа столбцов матрицы $A = (A_1, \dots, A_j, \dots, A_n)$, используемой в записи ограничения (2.9) канонической формы ЗЛП. В качестве базисных следует выбирать такие m переменных, которым соответствует набор столбцов A_{j_i} , $j_i \in B_0$, позволяющий составить единичную матрицу $P_0 = (A_{j_1}^{(0)}, \dots, A_{j_m}^{(0)})$.

Индекс “0” обозначает номер начального базиса.

Итак, если $P_0 = E$, то допустимость соответствующего базиса следует из уравнения $AX=b$ для ОДР с учетом неотрицательности вектора b ($b \geq 0$):

$$\begin{aligned} AX &= \sum_{j=1}^n A_j x_j = \sum_{j_i \in B_0} A_{j_i} x_{j_i} + \sum_{j \notin B_0} A_j x_j = \\ &= P_0 X^{B_0} + \sum_{j \notin B_0} A_j x_j = b. \end{aligned}$$

В опорной точке свободные переменные равны нулю ($x_j = 0$, $j \notin B_0$), и базисное решение

$$\tilde{X}^{B_0} = X^{B_0} / x_j = 0, \quad j \notin B_0,$$

определяется как решение уравнения $P_0 \tilde{X}^{B_0} = b$, откуда $\tilde{X}^{B_0} = P_0^{-1}b$. Так как $P_0 = P_0^{-1} = E$, то $\tilde{X}^{B_0} = b \geq 0$, что подтверждает допустимость найденного таким способом базиса X^{B_0} .

Если ОДР исходной ЗЛП задана в форме неравенства типа \leq :

$$\sum_{j=1}^n a_{ij}x_j \leq b_i, \quad b_i \geq 0, \quad i = \overline{1, m},$$

то начальный базис может быть сформирован из дополнительных переменных x_{n+i} , вводимых в систему ограничений с целью приведения ее к канонической форме равенств:

$$\sum_{j=1}^n a_{ij}x_j + x_{n+i} = b_i, \quad b_i \geq 0, \quad x_{n+i} \geq 0, \quad i = \overline{1, m},$$

В этом случае матрица P_0 будет единичной. Если в системе ограничений исходной задачи имеются неравенства типа \geq :

$$\sum_{j=1}^n a_{ij}x_j \geq b_i, \quad b_i \geq 0, \quad (2.12)$$

то базис, сформированный из дополнительных переменных x_{n+i} , $i = \overline{1, m}$ уже не будет допустимым. В этом случае начальный допустимый базис может быть найден с помощью метода искусственных переменных в процессе приведения исходной ЗЛП к каноническому виду.

Суть метода искусственных переменных состоит в том, что каждое неравенство типа (2.12), входящее в систему ограничений исходной задачи, трансформируется в равенство (2.13)

$$\sum_{j=1}^n a_{ij}x_j - x_{n+i} + x_{n+m+i}^* = b_i, \quad (2.13)$$

где x_{n+m+i}^* — так называемая искусственная переменная, которая вводится в систему ограничений специально для того, чтобы стать компонентом начального допустимого базиса.

Количество искусственных переменных равно количеству неравенств типа \geq в ограничениях исходной ЗЛП. Включение искусственных переменных в начальный базис B_0 позволяет сформировать его таким образом, что соответствующая ему матрица P_0 становится единичной, а в этом случае $\tilde{X}^{B_0} = b \geq 0$, т. е. базис B_0 действительно является допустимым.

Ни одна из искусственных переменных не должна войти в оптимальный базис. В противном случае исходная ЗЛП не имеет решения. Поэтому очевидно, что искусственные переменные, введенные в начальный базис исключительно с целью обеспечения его допустимости, должны выводиться из выбранного базиса на последующих шагах поиска оптимального решения. Для исключения искусственных переменных из базиса рассматриваемый метод предполагает следующую модификацию целевой функции:

$$f(X) = \sum_{j=1}^n c_j x_j - M \sum_{k=1}^l x_{n+k}^*,$$

где x_{n+k}^* — искусственная переменная, а $M >> |c_j|$, $j = \overline{1, n}$.

Наличие составляющей в целевой функции $-M \sum_{k=1}^l x_{n+k}^*$, приводит

к существенному уменьшению последней, которое тем значительнее, чем больше искусственных переменных в базисе. Поэтому требование максимизации $f(x)$ при переходе к очередной опорной точке естественным образом совпадает с необходимостью выведения искусственных переменных из выбранного базиса. Следует отметить, что модификация целевой функции не изменяет оптимального решения исходной ЗЛП, так как ни одна из искусственных переменных не входит в оптимальный базис, и дополнительная составляющая $-M \sum_{k=1}^l x_{n+k}^*$ в точке оптимума принимает нулевое значение.

2.4. ТАБЛИЧНАЯ ФОРМА АЛГОРИТМА СИМПЛЕКС-МЕТОДА

Заполнение симплекс-таблицы

Пусть задача линейного программирования сформулирована в канонической форме:

$$\max \sum_{j=1}^n c_j x_j; \quad (2.14)$$

$$\begin{aligned} \sum_{j=1}^n a_{ij} x_j &= b_i, \quad b_i \geq 0, \quad i = \overline{1, m}; \\ x_j &\geq 0, \quad j = \overline{1, n}. \end{aligned} \quad (2.15)$$

Выберем m базисных переменных и обозначим их y_1, \dots, y_m . Остальные $n - m = k$ свободных переменных обозначим x_1, \dots, x_k .

Разрешив систему (2.15) относительно базисных переменных, получим так называемое базисное решение, в котором базисные переменные выражены через свободные:

$$y_i = b'_i + \sum_{j=1}^k a'_{ij} x_j, \quad i = \overline{1, m}. \quad (2.16)$$

С учетом (2.16) целевую функцию можно также представить как ли-

нейную функцию свободных переменных:

$$f(X) = \sum_{j=1}^k c'_j x_j + f_0, \quad (2.17)$$

где x_j — свободные переменные.

Систему соотношений (2.16), (2.17) запишем в форме так называемой симплекс-таблицы [3], (табл. 2.1), опустив для простоты штрихи в обозначениях a'_{ij} , c'_j , b'_i .

Таблица 2.1

	x_1	x_2	...	x_j	...	x_r	...	x_k	b
y_1	a_{11}	a_{12}	...	a_{1j}	...	a_{1r}	...	a_{1k}	b_1
y_2	a_{21}	a_{22}	...	a_{2j}	...	a_{2r}	...	a_{2k}	b_2
\vdots	\vdots	\vdots	\ddots	\ddots	\ddots	\ddots	\ddots	\vdots	\vdots
y_s	a_{s1}	a_{sr}	...	a_{sj}	...	a_{sr}	...	a_{sk}	b_s
\vdots	\vdots	\vdots	\ddots	\vdots	\ddots	\ddots	\ddots	\vdots	\vdots
y_m	a_{m1}	a_{m2}	...	a_{mj}	...	a_{mr}	...	a_{mk}	b_m
$f(x)$	c_1	c_2	...	c_j	...	c_r	...	c_k	F

Составленная симплекс-таблица соответствует начальному базису и начальной опорной точке ОДР. Переход к очередной опорной точке в процессе поиска оптимального решения сопровождается составлением новой симплекс-таблицы.

Каждая симплекс-таблица анализируется по критериям допустимости и оптимальности базиса.

Признак допустимости базиса

В опорной точке в соответствии с (2.16) $y_i = b_i$, $i = \overline{1, m}$, поэтому признак допустимости базиса формулируется как условие

$$b_i \geq 0, i = \overline{1, m}.$$

Признак оптимальности базиса

1. Если для $\forall j \notin B, c_j < 0$, то найденное решение оптимально и единственno.
2. Если для $\forall j \notin B, c_j \leq 0$, то найденное решение оптимально, но не единственno.
3. Если $\exists c_j > 0$, то решение не оптимально. В последнем случае поиск оптимального решения продолжается и необходимо перейти к новой

опорной точке.

В соответствии с симплекс-методом новая опорная точка выбирается только среди соседних, т. е. новый базис лишь в одной переменной отличается от прежнего. Таким образом, формирование нового базиса осуществляется на базе прежнего посредством выведения из него одной из базисных переменных (y_s) и введения одной из свободных переменных (x_r).

Выбор переменной x_r

Выбор переменной x_r осуществляется по результатам анализа коэффициентов c_j симплекс-таблицы. Найдем

$$c_r = \max_{(j)} \{ c_j \},$$

тогда перемещение из данной опорной точки в направлении увеличения x_r , будет сопровождаться максимальным увеличением целевой функции.

Столбец, который соответствует переменной x_r в симплекс-таблице, будем называть разрешающим.

Выбор переменной y_s

Выбор переменной y_s производится по результатам анализа коэффициентов a_{ir} , $i = \overline{1, m}$ разрешающего столбца.

1. Если $\forall a_{ir} \geq 0$, $i = \overline{1, m}$, это означает, что ОДР такова, что неограниченное увеличение свободной переменной x_r приводит к неограниченному возрастанию целевой функции (ОДР не замкнута). Естественно, что анализ элементов a_{ij} симплекс-таблицы необходимо проводить для всех ее столбцов, в которых коэффициенты $c_j \geq 0$.

2. Если $\exists a_{ir} < 0$, то соответствующие базисные переменные y_i получают отрицательные приращения при увеличении x_r . Среди этих переменных y_i необходимо отыскать y_s , достигающую нуля при минимальном значении приращения x_r . Так как при переходе к новой опорной точке лишь одна из свободных переменных прежнего базиса получает приращение (x_r), а остальные свободные переменные остаются нулевыми, значение переменных y_i прежнего базиса определяется при этом в соответствии с уравнением

$$y_i = a_{ir}x_r + b_i,$$

где

$$x_r = \min_{(i)} \left\{ -\frac{b_i}{a_{ir}} \right\} = \frac{b_s}{a_{sr}},$$

$$a_{ir} < 0$$

Строчку, которая соответствует переменной y_s в симплекс-таблице, будем называть разрешающей; элемент a_{sr} — разрешающим элементом симплекс-таблицы.

Выбор разрешающего элемента завершает формирование нового набора X^E_1 базисных переменных, отличающихся от прежнего базиса одной переменной x_r :

$$X^E_1 = (y_1, \dots, y_{s-1}, x_r, y_{s+1}, \dots, y_m)^T.$$

Для нового базиса (новой опорной точки) снова заполняется симплекс-таблица, в которой новые базисные переменные выражены через новые свободные.

Пересчет симплекс-таблиц

Из соотношения (2.16) следует

$$y_s = \sum_{j=1}^k a_{sj}x_j + a_{sr}x_r + b_s. \quad (2.18)$$

Отсюда

$$x_r = \sum_{j=1, j \neq r}^k \left(-\frac{a_{sj}}{a_{sr}} x_j \right) + \frac{y_s}{a_{sr}} - \frac{b_s}{a_{sr}}. \quad (2.19)$$

Подставив (2.19) в (2.16), получим

$$y_i = \underbrace{b_i - \frac{a_{ir}}{a_{sr}} b_s}_{b_i^*} + \underbrace{\frac{a_{ir}}{a_{sr}} y_s}_{a_{ir}^*} + \sum_{j=1, j \neq r}^k \underbrace{\left(a_{ij} - \frac{a_{ir}}{a_{sr}} a_{sj} \right)}_{a_{ij}^*} x_j. \quad (2.20)$$

Используя (2.18),(2.19), целевую функцию можно также выразить через новые свободные переменные:

$$f(x) = \underbrace{\sum_{j=1, j \neq r}^k \left(c_j^* - \frac{a_{sj}}{a_{sr}} c_r \right) x_j}_{c_j^*} + \underbrace{\frac{c_r}{a_{sr}} y_s}_{c_r^*} - \underbrace{\left(f_0 - \frac{b_s}{a_{sr}} c_r \right)}_{f_o^*}. \quad (2.21)$$

Обозначения b_i^* , a_{ij}^* , c_j^* , f_o^* , используемые в (2.20) и (2.21), относятся к элементам новой симплекс-таблицы. Сформулируем на основе (2.19),(2.20) и (2.21) правило пересчета симплекс-таблицы.

Правило пересчета

1. Разрешающий элемент заменяется на 1.
2. Элементы разрешающего столбца за исключением a_{sr} переписываются без изменений.
3. Элементы разрешающей строки за исключением a_{sr} изменяют знак на противоположный.
4. Оставшиеся элементы новой симплекс-таблицы вычисляются согласно правилу прямоугольника: произведение соответствующего элемента прежней таблицы на разрешающий a_{sr} минус произведение элементов, находящихся на другой диагонали получившегося прямоугольника. В соответствии с этим правилом имеем

$$\begin{aligned} a'_{ij} &= a_{ij} a_{sr} - a_{sj} a_{ir}; \\ b'_i &= b_i a_{sr} - b_s a_{ir}; \\ c'_j &= c_j a_{sr} - c_r a_{sj}; \\ f'_o &= f_o a_{sr} - c_r b_s, \end{aligned}$$

а расположение элементов, входящих в правые части формул, поясняется в табл. 2.2.

Таблица 2.2

	x_j		x_r		b
y_s		a_{sj}	a_{sr}		b_s
y_i		a_{ij}	a_{ir}		b_i
$f(x)$		c_j	c_r		f_o

5. Все элементы полученной таблицы необходимо разделить на разрешающий элемент a_{sr} : $a_{ij}^* = \frac{a'_{ij}}{a_{sr}}$, $b_i^* = \frac{b'_i}{a_{sr}}$, $c_j^* = \frac{c'_j}{a_{sr}}$, $f_o^* = \frac{f'_o}{a_{sr}}$.

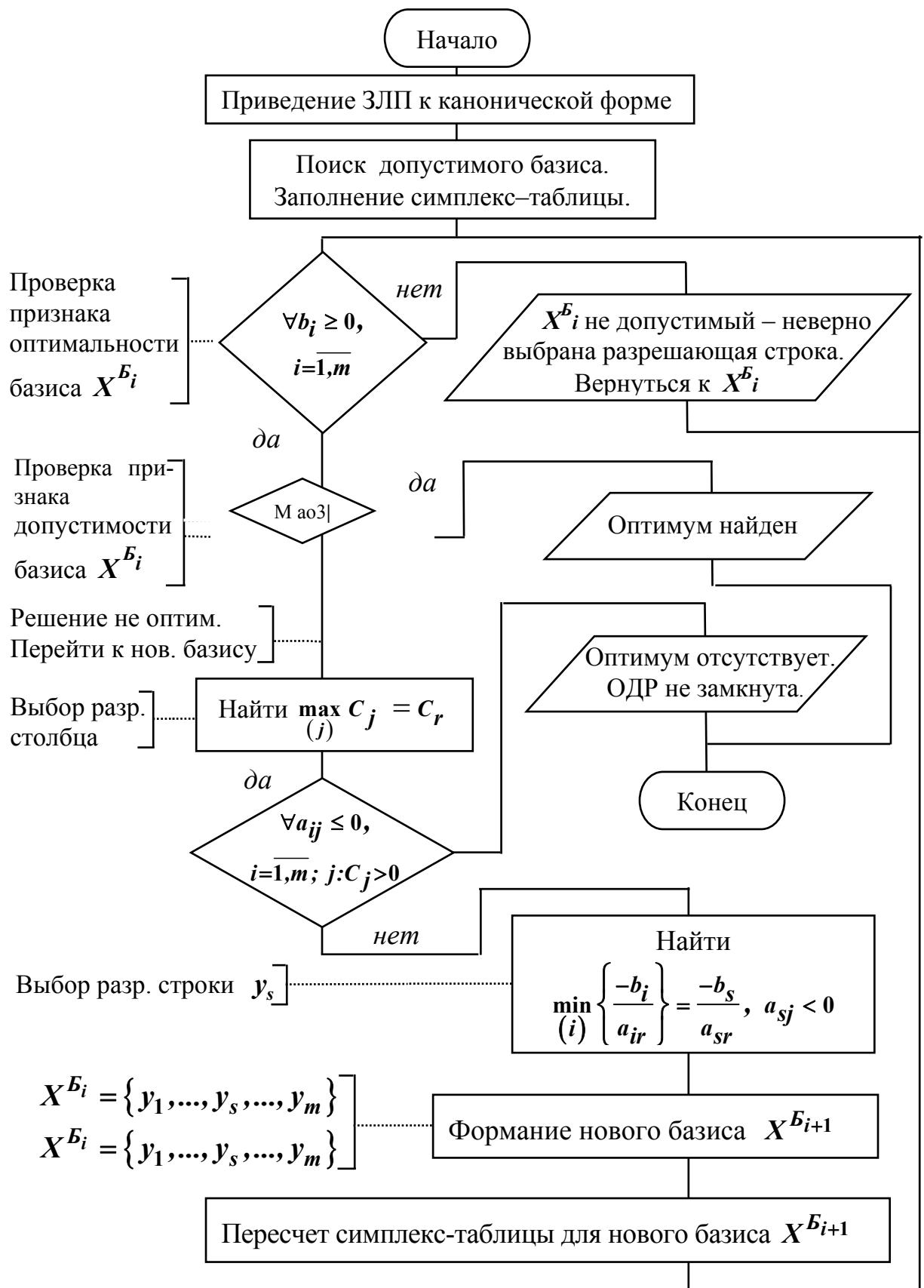


Рис. 2.1

Изложенная последовательность действий составляет описание стандартного шага симплекс-метода, начинающегося с анализа очередной симплекс-таблицы и завершающегося заполнением новой. Процедура поиска оптимального решения ЗЛП с помощью симплекс-метода в табличной форме реализуется в соответствии с алгоритмом, приведенным на рис. 2.1.

2.5. ПОНЯТИЕ ДВОЙСТВЕННОСТИ В ЛИНЕЙНОМ ПРОГРАММИРОВАНИИ

Двойственный симплекс метод

Любой задаче линейного программирования может быть поставлена в соответствие другая задача линейного программирования, называемая двойственной [1, 2]. По отношению к двойственной ЗЛП исходная задача называется прямой. И наоборот, если исходной задачей считать двойственную, то прямая задача является двойственной по отношению к исходной. Поэтому имеет смысл говорить о паре двойственных задач, примеры которых приведены в табл. 2.3.

Таблица 2.3

Прямая задача	Двойственная задача
$\min C^T X,$ $AX \geq b,$ $X \geq 0.$	$\max b^T Y,$ $A^T Y \leq C,$ $Y \geq 0.$
$\max C^T X,$ $AX \leq b,$ $X \geq 0.$	$\min b^T Y,$ $A^T Y \geq C,$ $Y \geq 0.$
$\min C^T X,$ $AX = b,$ $X \geq 0.$	$\max b^T Y,$ $A^T Y \leq C.$
$\max C^T X,$ $AX = b,$ $X \geq 0.$	$\min b^T Y,$ $A^T Y \geq C.$

Совокупность прямой и двойственной задач линейного программирования характеризуется рядом важных свойств [1, 2, 4].

Если множество допустимых значений одной из задач пусто, то значение целевой функции в другой двойственной задаче не ограничено [4].

Если существует оптимальное решение одной из них, то и другая задача имеет оптимальное решение, причем оптимальные значения целевых функций пары двойственных задач совпадают: $C^T X_{opt} = b^T Y_{opt}$. Между оптимальными решениями двойственных задач имеет место строгое соответствие, позволяющее по оптимальному решению одной из них, например X_{opt} , находить оптимальное решение другой.

Поскольку одна из двойственных задач часто решается проще, чем другая, отмеченное свойство является конструктивным и используется при построении эффективных алгоритмов в решении задач линейного программирования.

Существенным мотивом введения двойственной задачи линейного программирования является использование ее при анализе чувствительности исходной задачи. Доказано, что компоненты вектора \mathbf{Y}_{opt} представляют собой численные значения производных целевой функции исходной задачи по соответствующим компонентам вектора \mathbf{b} [2]. Другими словами, решение \mathbf{Y}_{opt} можно интерпретировать как вектор теневых цен ресурсов \mathbf{b}_i . Ниже рассматривается один из методов линейного программирования, основанный на использовании понятия двойственности – двойственный симплекс-метод.

Понятие псевдоплана

Рассмотрим пару двойственных задач.

<p><i>прямая задача</i></p> $\max C^T X \quad (2.22)$ $AX = b, \quad X \geq 0 \quad (2.23)$	<p><i>двойственная задача</i></p> $\min b^T Y; \quad (2.24)$ $A^T Y \geq C, \quad (2.25)$
---	---

Введем некоторые понятия и обозначения.

1. Базис $A_B[m \times m] = \{A_i\}$, $i \in B$ — система m линейно-независимых вектор-столбцов матрицы $A_B[m \times m]$ ограничений прямой задачи.
 2. $X^B[m \times 1] = \{x_i\}$, $i \in B$ — вектор, компонентами которого являются переменные, соответствующие столбцам A_i базиса $\{A_i\}$, $i \in B$.
 3. $C_{B[m \times 1]} = \{C_i\}$, $i \in B$ — вектор, составленный из соответствующих базису $\{A_i\}$ компонентов вектора $C[n \times 1]$.
 4. Базис $\{A_i\}$, $i \in B$ будем называть сопряженным или базисом двойственной задачи, если ее базисное решение $Y^B = [A_B^T]^{-1} C_B$, определяемое из уравнения

$$A_E^T Y = C_E, \quad (2.26)$$

удовлетворяет условию (2.25), т. е. является допустимым решением двойственной задачи.

Принципиальным свойством пары двойственных задач является со-
пряженность их оптимальных базисов. Отсюда следует возможность нахо-
ждения оптимального решения одной из них по известному решению дру-
гой.

В частности, пусть $\{A_i\}, i \in B_{opt}$ — оптимальный базис прямой задачи, вектор C_i составлен из соответствующих этому набору базисных переменных компонентов вектора C . Тогда оптимальное решение двойств-

венной задачи является решением следующего уравнения:

$$A_i^T Y = C_i$$

Базисное решение \tilde{X}^B прямой задачи, полученное относительно сопряженного базиса $\{A_i\}$, $i \in B$, будем называть псевдопланом прямой задачи. Название “псевдоплан” подчеркивает тот факт, что в решении \tilde{X}^B может быть нарушено условие неотрицательности компонентов вектора X .

Ниже приводится ряд преобразований системы (2.23), обосновывающих вывод формулы (2.27) для определения псевдоплана \tilde{X}^B . $AX = b$, или в скалярной форме:

$$\begin{aligned} \sum_{j=1}^n A_j x_j &= b, \quad \sum_{j=1}^n A_j x_j = \sum_{i \in B} A_i x_i + \sum_{j \notin B} A_j x_j = \\ &= A_B X^B + \sum_{j \notin B} A_j x_j = b. \end{aligned}$$

Умножим левую и правую части уравнения на A_B^{-1} , тогда

$$X^B = A_B^{-1} b - \sum_{j \notin B} A_B^{-1} A_j x_j. \quad (2.27)$$

Так как в опорной точке свободные переменные равны нулю, базисное решение \tilde{X}^B прямой задачи определяется следующим образом:

$$\tilde{X}^B = A_B^{-1} b \quad (2.28)$$

Выразим целевую функцию $f(x)$ через свободные переменные x_j , $j \notin B$, не вошедшие в псевдоплан X^B .

$$f(X) = C^T X = \sum_{i \in B} c_i x_i + \sum_{j \notin B} c_j x_j = C_B^T X^B + \sum_{j \notin B} c_j x_j. \quad (2.29)$$

Подставим (2.27) в (2.29):

$$\begin{aligned} f(x) &= C_B^T \left\{ A_B^{-1} b - \sum_{j \notin B} A_B^{-1} A_j x_j \right\} + \sum_{j \notin B} c_j x_j = \\ &= C_B^T A_B^{-1} b - \sum_{j \notin B} \underbrace{[C_B^T A_B^{-1} A_j - c_j]}_{\Delta_j} x_j = C_B^T A_B^{-1} b - \sum_{j \notin B} \Delta_j x_j. \quad (2.30) \end{aligned}$$

Формула (2.30) составляет содержание последней строки симплекстаблицы, в которой $f_0 = C_B^T A_B^{-1} b$, а коэффициенты c_j , $j \notin B$ представляют собой взятые с обратным знаком характеристические разности Δ_j ,

$j \notin \mathcal{B}$ (см. табл.2.1). Псевдоплан $\tilde{X}^{\mathcal{B}}$ обладает следующим свойством: в симплекс-таблице, построенной для псевдоплана $\tilde{X}^{\mathcal{B}}$ прямой задачи, все коэффициенты c_j удовлетворяют условию:

$$c_j \leq 0, \quad j \notin \mathcal{B} \quad \text{или} \quad \Delta_j \geq 0, \quad j \notin \mathcal{B}. \quad (2.31)$$

Выполнение этого условия для псевдоплана $\tilde{X}^{\mathcal{B}}$ означает, что соответствующее ему базисное решение $\tilde{Y}^{\mathcal{B}}$, определяемое из (2.26), является допустимым решением двойственной задачи. Если среди составляющих псевдоплана $\tilde{X}^{\mathcal{B}}$ нет отрицательных, то псевдоплан является оптимальным решением прямой задачи.

При необходимости двойственное решение $\tilde{Y}^{\mathcal{B}}$ может быть получено из уравнения (2.26), где $A_{\mathcal{B}}$ и $C_{\mathcal{B}}$ соответствуют X_{opt} . С учетом сформулированных выше требований, предъявляемых к начальному псевдоплану, и свойств оптимального псевдоплана перейдем к описанию последовательности этапов двойственного симплекс-метода.

1. Поиск начального псевдоплана и заполнение симплекс-таблицы.
2. Анализ псевдоплана по признакам допустимости и оптимальности.

При выполнении условия допустимости псевдоплана здесь возможны три случая:

а) базисное решение $\tilde{X}^{\mathcal{B}} \geq 0$. В соответствии с признаком оптимальности полученное решение является оптимальным.

б) в базисном решении $\tilde{X}^{\mathcal{B}}$ имеются отрицательные компоненты $\tilde{x}_i^{\mathcal{B}} < 0$, но среди элементов a_{ij} симплекс-таблицы отсутствуют положительные: $a_{ij} < 0, \quad j \notin \mathcal{B}$. В этом случае ни прямая, ни обратная задачи не имеют решения;

в) базисное решение $\tilde{X}^{\mathcal{B}}$ содержит отрицательные компоненты, но для каждой из них имеется хотя бы один элемент

$$a_{ij} > 0, \quad j \notin \mathcal{B}.$$

3. В случае “в” псевдоплан не является оптимальным, и следует перейти к другому сопряженному базису, т. е. к новому псевдоплану, для которого, в свою очередь, заполняется симплекс-таблица и повторяется описанная выше процедура анализа.

Рассмотрим подробнее содержание этапов двойственного симплекс-метода.

Заполнение симплекс-таблицы

Предварительно сформировав сопряженный базис $\{A_i\}, i \in \mathcal{B}_l$, необходимо, используя уравнение (2.15), выразить все m составляющих x_i псевдоплана и целевую функцию $f(x)$ через оставшиеся $n - m$ свободных составляющих x_j вектора X :

$$x_i = b_i + \sum_{j=1}^k a_{ij} x_j, \quad i \in \mathcal{B}_l, \quad j \notin \mathcal{B}_l \quad (2.32)$$

Симплекс-таблица составляется в той же форме, что и для обычного симплекс-метода (см. табл. 2.1).

Анализ псевдоплана

Признаком допустимости псевдоплана $X^{\mathcal{B}_l}$ является неположительность коэффициентов последней строки симплекс-таблицы:

$$c_j \leq 0, \quad j \notin \mathcal{B}_l.$$

Признаком оптимальности базисного решения $X^{\mathcal{B}_l}$ является условие его неотрицательности, т. е. неотрицательности элементов b_i последнего столбца симплекс-таблицы.

При невыполнении этого условия следует анализировать знаки элементов a_{ij} в тех строках симплекс-таблицы, в которых $b_i < 0$, и если:

- а) $\forall a_{ij} \leq 0, b_i < 0$, то дальнейший поиск оптимума не имеет смысла, так как ни прямая, ни двойственная задачи не имеют решения;
- б) $\exists a_{ij} > 0, b_i < 0$, то данный псевдоплан не является оптимальным и необходим переход к новому псевдоплану.

Переход к новому псевдоплану

Прежний $X^{\mathcal{B}_l}$ и новый $X^{\mathcal{B}_{l+1}}$ псевдопланы отличаются между собой только одной переменной. Поэтому переход к новому псевдоплану так же как и к новому базису обычного симплекс-метода, состоит в выборе переменной x_s прежнего псевдоплана, исключаемой из нового псевдоплана, и выборе свободной переменной x_r прежнего псевдоплана, включаемой в новый псевдоплан:

$$\begin{cases} x_s \in X^{\mathcal{B}_l} \\ x_s \notin X^{\mathcal{B}_{l+1}} \end{cases}; \quad \begin{cases} x_r \notin X^{\mathcal{B}_l} \\ x_r \in X^{\mathcal{B}_{l+1}} \end{cases}.$$

Выбор переменной x_s

Переменная x_s – переменная псевдоплана X^{B_1} , принимающая наименьшее отрицательное значение в опорной точке:

$$x_s = b_s = \min_{(i)} b_i, \quad b_i < 0.$$

Строчку симплекс-таблицы, соответствующую x_s , будем называть *разрешающей*.

Выбор переменной x_r

Правило выбора переменной x_r формулируется следующим образом: среди строго положительных элементов a_{sj} разрешающей строки найти элемент a_{sr} такой, что

$$-\frac{c_r}{a_{sr}} = \min_{(j)} \left\{ \frac{c_j}{a_{sj}}, \quad a_{sj} > 0 \right\}, \quad (2.33)$$

тогда в новый псевдоплан должна войти переменная x_r . Столбец симплекс-таблицы, соответствующий x_r , будем называть *разрешающим*.

Приведенному выше правилу можно дать следующее обоснование. Поскольку переменная x_s не останется в новом псевдоплане, а перейдет в группу свободных переменных, то в новой опорной точке x_s должна принять нулевое значение. Таким образом, переменная x_s при переходе от прежней опорной точки, где она была отрицательной, к новой опорной точке получит положительное приращение в соответствии с уравнением (2.32):

$$x_s = b_s + a_{sj} x_j, \quad a_{sj} > 0. \quad (2.34)$$

Отсюда ясно, что положительное приращение x_s обусловлено положительным приращением свободной переменной x_j , $a_{sj} > 0$. Так как все коэффициенты c_j не положительные (это условие допустимости псевдоплана), увеличение свободной переменной x_j , как правило, сопровождается уменьшением целевой функции $f(x)$, поэтому целесообразно выбирать переменную x_r так, чтобы минимизировать уменьшение $f(x)$, связанное с обнулением x_s в новой опорной точке.

Итак, в новой опорной точке $x_s = 0$. В соответствии с (2.34)

$$x_s = b_s + a_{sj} x_j, \quad a_{sj} > 0, \quad \text{откуда } x_j = -\frac{b_s}{a_{sj}}.$$

Тогда целевая функция получает следующее отрицательное приращение при переходе к новой опорной точке:

$$\Delta f(X) = f(\tilde{X}^{B_{l+1}}) - f(\tilde{X}^{B_l}) = -c_j \frac{b_s}{a_{sj}}.$$

Очевидно, что разумный выбор переменной x_r состоит в том, чтобы обеспечить $\min |\Delta f(X)| = \min_{(j)} \left\{ c_j \frac{b_s}{a_{sj}}, a_{sj} > 0 \right\}$, что будет иметь место именно при выполнении условия (2.33).

Пересчет симплекс-таблицы

После того как новый псевдоплан $X^{B_{l+1}}$ сформирован, необходимо заполнить новую симплекс-таблицу. Пересчет симплекс-таблицы осуществляется в полном соответствии с правилом, изложенным выше при описании обычного симплекс-метода.

Анализ формулы (2.21) пересчета элементов c_j симплекс-таблицы позволяет сделать вывод о том, что условие допустимости псевдоплана (допустимости сопряженного решения двойственной задачи) сохраняется при переходе к новой опорной точке. Покажем, что при выполнении условия $c_j \leq \mathbf{0}$ выполняется и условие $c_j^* \leq \mathbf{0}$. Из (2.21) следует, что

$$c_r^* = \frac{c_r}{a_{sr}} \leq \mathbf{0}, \text{ так как } c_r \leq \mathbf{0}, \text{ а } a_{sr} > 0. \text{ С учетом (2.21)}$$

$$c_j^* = c_j - \frac{a_{sj}}{a_{sr}} c_r, \quad j \neq r. \text{ Если } a_{sj} \leq 0, \text{ то } c_j^* \leq c_j \leq \mathbf{0}. \text{ Если } a_{sj} > 0, \text{ то со-}$$

отношение $-\frac{c_r}{a_{sr}} \leq -\frac{c_j}{a_{sj}}$, которое следует из условия (2.33), означает, что

$$c_j^* \leq \mathbf{0}.$$

Заполнение симплекс-таблицы для нового псевдоплана завершает стандартный шаг двойственного симплекс-метода. На рис. 2.2 приведен алгоритм двойственного симплекс метода, объединяющий в логической последовательности рассмотренные выше этапы.

Применение двойственного симплекс-метода к задачам с возрастающим количеством условий

Предположим, что после отыскания оптимального базиса задачи (2.7) линейного программирования появились дополнительные ограничения, меняющие допустимое множество решений. Применение двойствен-

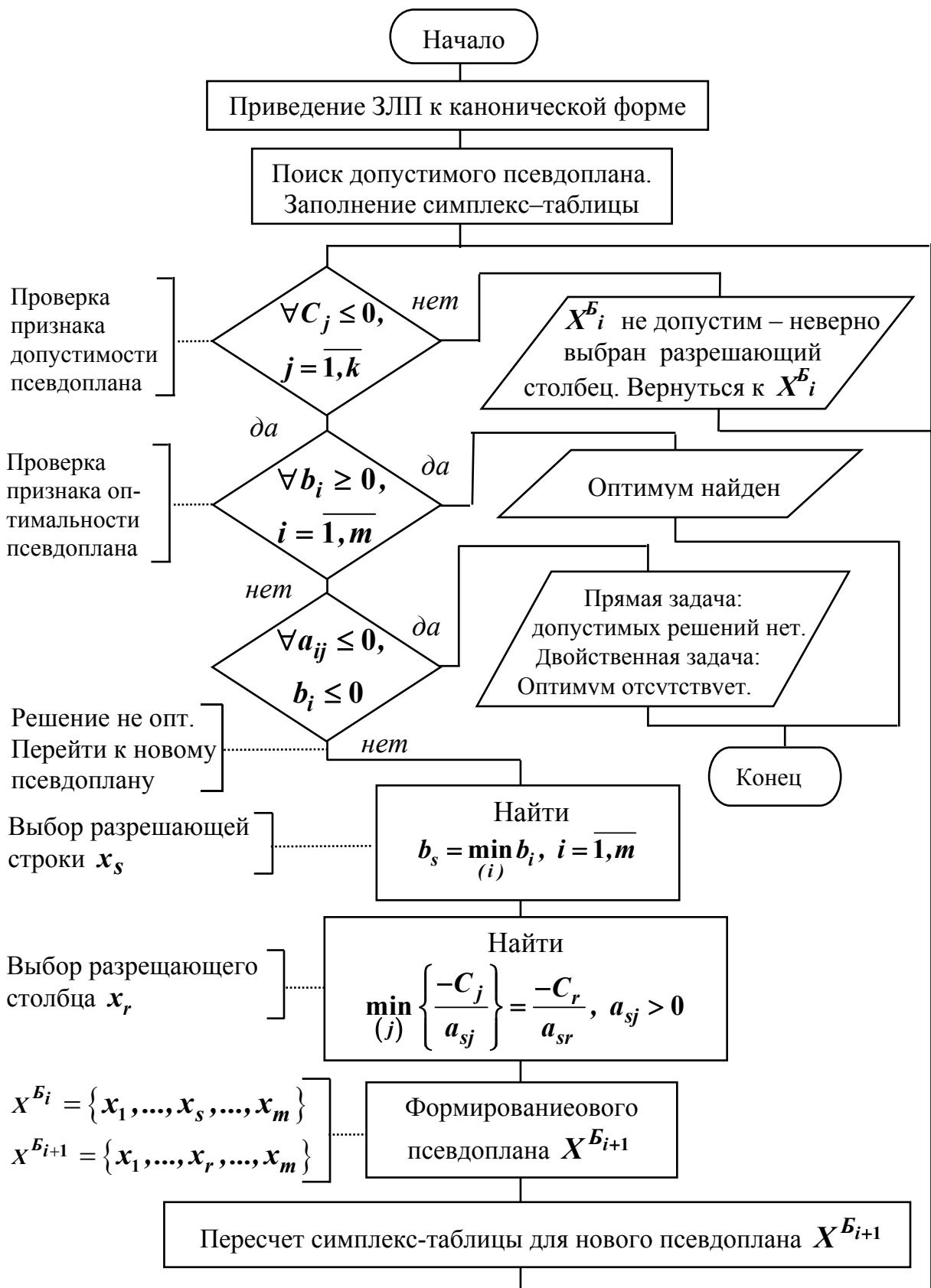


Рис.2.2

ого симплекс-метода в этом случае позволяет максимально полно использовать результаты решения прежней задачи.

Пусть найдено оптимальное решение задачи

$$\max C^T X;$$

$$\begin{aligned} AX &= b; \\ X &\geq 0 \end{aligned}$$

и составлена соответствующая ему симплекс-таблица в форме табл. 2.1.

Появление в этой задаче нового ограничения

$$\sum_{j=1}^n a_j x_j \leq b_i \quad \text{или} \quad \sum_{j=1}^n a_j x_j \geq b_i,$$

($b_i \geq 0$) означает добавление уравнения

$$\sum_{j=1}^n a_j x_j + x_{n+1} = b_i$$

или

$$\sum_{j=1}^n a_j x_j - x_{n+1} = b_i, \quad x_{n+1} \geq 0$$

к системе $AX = b$.

Увеличение размерности вектора X приводит к появлению базисной переменной x_{n+1} , число же свободных переменных остается прежним. Видоизмененная симплекс-таблица отличается от прежней дополнительной строкой для еще одной базисной переменной x_{n+1} . Если в этой строке $b_{n+1} \geq 0$, то прежнее оптимальное решение является таковым и для новой задачи. В противном случае ($b_{n+1} < 0$) базис $(x_1, \dots, x_i, \dots, x_m, \dots, x_{n+1})$ не является допустимым, но может рассматриваться как допустимый псевдоплан (так как $\forall c_j \leq 0, j \notin B_0$), что и предполагает дальнейшее решение задачи двойственным симплекс-методом.

2.6. ПРИМЕРЫ РЕШЕНИЯ ЗАДАЧ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

Пример 2.1

Решение ЗЛП симплекс-методом в табличной форме

$$\max \{ 3x_1 + 4x_2 \},$$

$$\begin{cases} x_1 + 2x_2 \leq 8; \\ 2x_1 + x_2 \leq 10; \\ x_1, x_2 \geq 0. \end{cases}$$

1. Приведение ЗЛП к канонической форме:

<p style="text-align: center;">скалярная</p> $\max \{ 3x_1 + 4x_2 \};$ $\begin{cases} x_1 + 2x_2 + x_3 = 8; \\ 2x_1 + x_2 + x_4 = 10; \\ x_1, x_2, x_3, x_4 \geq 0. \end{cases}$	<p style="text-align: center;">матричная</p> $\max C^T X; \quad AX = b; \quad X \geq 0,$ <p>где $b = (8, 10)^T$, $X = (x_1, x_2, x_3, x_4)^T$,</p> $C = (3, 4, 0, 0)^T, \quad A = \begin{bmatrix} 1 & 2 & 1 & 0 \\ 2 & 1 & 0 & 1 \end{bmatrix}.$
--	--

2. Выбор допустимого начального базиса

$X^{B_0} = (x_3, x_4)^T$, так как столбцы A_3, A_4 матрицы A образуют единичную матрицу.

3. Заполнение симплекс-таблицы (табл. 2.4)

Таблица 2.4

	x_1	x_2	b
x_3	-1	-2	8
x_4	-2	-1	10
f	3	4	0

Таблица 2.5

	x_1	x_3	b
x_2	1	1	-8
x_4	3	-1	-12
f	-2	4	-32

Таблица 2.6

	x_1	x_3	b
x_2	-1/2	-1/2	4
x_4	-3/2	1/2	6
f	1	-2	16

Таблица 2.7

	x_4	x_3	b
x_2	-1/2	1	-3
x_1	1	-1/2	-6
f	1	5/2	-30

Таблица 2.8

	x_4	x_3	b
x_2	1/2	-2/3	2
x_1	-2/3	1/3	4
f	-2/3	-5/3	20

4. Проверка признака допустимости базиса

В опорной точке выполняется условие

$$\tilde{X}^{B_0} \geq 0: \quad \tilde{X}^{B_0} = (8, 10)^T.$$

5. Проверка признака оптимальности базиса

Базис (x_3, x_4) не оптимальный, так как $c_1 = 3 > 0$ и $c_2 = 4 > 0$.

6. Выбор разрешающего столбца

$\max\{c_1, c_2\} = c_2 = 4$, следовательно, в базис X^{B_1} должна войти переменная x_2 .

7. Выбор разрешающей строки

При увеличении x_2 переменные x_3, x_4 получают отрицательные приращения $a_{32}x_2, a_{42}x_2$ соответственно (так как $a_{32} = -2 < 0$ и $a_{42} = -1 < 0$). Причем первой нулевого значения достигает переменная x_3 , так как

$$\min_{\substack{(i) \\ a_{ir} < 0}} \left\{ -\frac{b_i}{a_{ir}} \right\} = \min \left\{ \frac{8}{2}, \frac{10}{1} \right\} = 4.$$

Поэтому x_3 не войдет в базис X^{B_1} . В табл. 2.4 помечены разрешающий столбец x_2 , разрешающая строка x_3 и разрешающий элемент $a_{32} = -2$. Новый базис $X^{B_1} = (x_2, x_4)^T$.

8. Пересчет симплекс-таблицы

Осуществляется в соответствии с правилом, приведенным выше (см. п. 2.4). В табл. 2.5 зафиксирован результат выполнения первых четырех пунктов правила пересчета, а в табл. 2.6 представлена окончательно заполненная симплекс-таблица для базиса $X^{B_1} = (x_2, x_4)^T$.

Базис X^{B_1} , как следует из анализа табл. 2.6, допустимый, но не оптимальный. Разрешающий элемент $a_{41} = -3/2$ (табл. 2.6) определяет необходимость перехода к базису $X^{B_2} = (x_2, x_1)^T$. В табл. 2.7 и 2.8 приведены промежуточный (после выполнения первых четырех пунктов правила пересчета таблиц) и окончательный результаты пересчета симплекс-таблицы для базиса X^{B_2} .

Анализ табл. 2.8 позволяет сделать вывод о допустимости и оптимальности базиса X^{B_2} .

Пример 2.2

Симплекс-метод с использованием искусственных переменных в начальном базисе

$$\max\{3x_1 + 4x_2\};$$

$$\begin{cases} x_1 + 2x_2 \leq 8; \\ 2x_1 + x_2 \geq 10; \\ x_1, x_2 \geq 0. \end{cases}$$

1. Приведение ЗЛП к канонической форме и выбор допустимого базиса

Каноническая форма ЗЛП:

$$\begin{aligned} & \max\{3x_1 + 4x_2\}; \\ & \begin{cases} x_1 + 2x_2 + x_3 = 8; \\ 2x_1 + x_2 - x_4 = 10; \\ x_1, x_2, x_3, x_4 \geq 0. \end{cases} \end{aligned}$$

Начальный базис $X^{B_0} = (x_3, x_4)^T$, составленный из дополнительных переменных, будет недопустимым ($\tilde{x}_3 = 8$, $\tilde{x}_4 = -10$).

Для построения допустимого базиса воспользуемся методом искусственных переменных и сформулируем модифицированную ЗЛП:

$$\begin{aligned} & \max\{3x_1 + 4x_2 - Mx_5\}, M \gg 4; \\ & \begin{cases} x_1 + 2x_2 + x_3 = 8; \\ 2x_1 + x_2 - x_4 + x_5 = 10; \\ x_1, x_2, x_3, x_4, x_5 \geq 0. \end{cases} \end{aligned}$$

Начальный базис $X^{B_0} = (x_3, x_5)^T$, включающий искусственную переменную x_5 , является допустимым (табл. 2.9).

Для заполнения последней строки табл. 2.9 выразим x_5 через свободные переменные и подставим в выражение для целевой функции:

$$\begin{aligned} x_5 &= 10 + x_4 - 2x_1 - x_2, \\ f(x) &= 3x_1 + 4x_2 - M[10 + x_4 - 2x_1 - x_2] = \\ &= [2M + 3]x_1 + [M + 4]x_2 - Mx_4 - 10M. \end{aligned}$$

2. Анализ симплекс-таблицы и выбор разрешающего элемента

Эти действия осуществляются в соответствии с общим правилом. Симплекс-таблицы 2.10, 2.11 и 2.12 составлены для очередных опорных точек, последовательно входящих в траекторию поиска оптимального решения.

Табл. 2.12 соответствует оптимальному решению модифицированной задачи. Искусственная переменная x_5 не вошла в оптимальный базис (x_4, x_1), следовательно, исходная ЗЛП действительно имеет решение. Оптимальное решение модифицированной задачи:

$$x_{opt} = (8, 0, 0, 6, 0) = (x_{1opt}, x_{2opt})^T = (8, 0)^T, f(x_{opt}) = 24.$$

На рисунке 2.3 построены области допустимых решений для примеров 1 и 2 (ОДР₁ и ОДР₂) и отмечены опорные точки, соответствующие оптимальным решениям (opt_1 , opt_2).

Таблица 2.9

	x_1	x_2	x_4	b
x_3	-1	-2	0	8
x_5	-2	-1	1	10
f	$2M+3$	$M+4$	$-M$	$-10M$

Таблица 2.10

	x_5	x_2	x_4	b
x_3	$1/2$	$-3/2$	$-1/2$	3
x_1	$-1/2$	$-1/2$	$1/2$	5
f	$-M-3/2$	$5/2$	$5/2$	15

Таблица 2.11

	x_5	x_3	x_4	b
x_2	$1/3$	$-2/3$	$-1/3$	2
x_1	$-2/3$	$-1/3$	$2/3$	4
f	$-M-2/3$	$-5/3$	$2/3$	20

Таблица 2.12

	x_5	x_3	x_2	b
x_4	1	-2	-3	6
x_1	0	-1	-2	8
f	$-M$	-3	-2	24

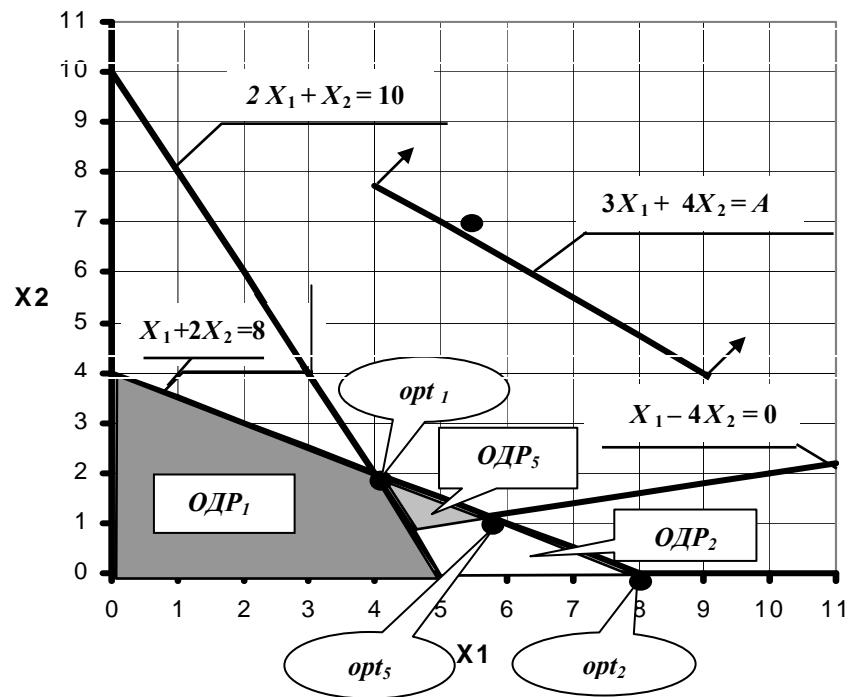


Рис. 2.3.

Пример 2.3*Решение ЗЛП двойственным симплекс- методом*

$$\max \{-x_1 - x_2\};$$

$$\begin{cases} x_1 + 2x_2 \leq 8; \\ 2x_1 + x_2 \geq 10; \\ x_1, x_2 \geq 0. \end{cases}$$

1. Приведение ЗЛП к канонической форме

$$\max\{-x_1 - x_2\};$$

$$\begin{cases} x_1 + 2x_2 + x_3 = 8; \\ 2x_1 + x_2 - x_4 = 10; \\ x_1, x_2, x_3, x_4 \geq 0. \end{cases}$$

В начальный псевдоплан X^{B_0} войдут переменные x_3, x_4 :
 $X^{B_0} = (x_3, x_4)^T$. Для псевдоплана X^{B_0} составлена симплекс-таблица 2.13.

2. Проверка допустимости псевдоплана

$C_1 = -1 < 0, C_2 = -1 < 0$ — условие допустимости X^{B_0} выполняется.

3. Проверка оптимальности псевдоплана

Так как $\exists b_i < 0$ ($b_4 = -10$), псевдоплан не является оптимальным и необходимо переходить к новому псевдоплану.

4. Выбор разрешающей строки

Разрешающая строка соответствует переменной x_4 , так как $b_4 = -10 < 0$.

5. Выбор разрешающего столбца

Так как $\min\left\{-\frac{c_1}{a_{41}} = \frac{1}{2}, -\frac{c_2}{a_{42}} = 1\right\} = -\frac{c_1}{a_{41}} = \frac{1}{2}$,

то разрешающий столбец — x_1 . Разрешающий элемент $a_{41} = -2$, а новый псевдоплан включает переменные x_3, x_1 и $X^{B_1} = (x_3, x_1)^T$.

6. Пересчет симплекс-таблицы

Для нового псевдоплана пересчет осуществляется по общему правилу (см. п. 2.4). Анализ таблицы 2.14 для псевдоплана X^{B_1} выявляет его допустимость ($c_4 = c_2 = -1/2 < 0$) и оптимальность

$$(b_3 = 3 > 0, b_1 = 5 > 0); x_{opt} = (5, 0, 3, 0)^T, f(x_{opt}) = -5.$$

Располагая оптимальным решением, найдем оптимальное решение двойственной задачи.

Таблица 2.13

	x_1	x_2	b
x_3	-1	-2	8
x_4	2	1	-10
$f(x)$	-1	-1	0

Таблица 2.14

	x_4	x_2	b
x_3	-1/2	-3/2	3
x_1	1/2	-1/2	5
$f(x)$	-1/2	-1/2	-5

Пример 2.4

Проверка сопряженности решений прямой и двойственной задачи

Сформулируем прямую задачу в канонической форме:

$$\left. \begin{array}{l} \text{в скалярном виде} \\ \max \{-x_1 - x_2\}; \\ \left\{ \begin{array}{l} x_1 + 2x_2 + x_3 = 8; \\ 2x_1 + x_2 - x_4 = 10; \\ x_1, x_2, x_3, x_4 \geq 0. \end{array} \right. \end{array} \right| \begin{array}{l} \text{в матричном виде} \\ \max \mathbf{C}^T \mathbf{X}; \quad \mathbf{A} \mathbf{X} = \mathbf{b}; \quad \mathbf{X} \geq \mathbf{0}, \\ \text{где } \mathbf{b} = (8, 10)^T, \quad \mathbf{X} = (x_1, x_2, x_3, x_4)^T, \\ \mathbf{C} = (-1, -1, 0, 0)^T, \quad \mathbf{A} = \begin{bmatrix} 1 & 2 & 1 & 0 \\ 2 & 1 & 0 & -1 \end{bmatrix}. \end{array}$$

Тогда двойственная к ней задача будет сформулирована так:

$$\min \{ 8y_1 + 10y_2 \};$$

$$\left\{ \begin{array}{l} y_1 + 2y_2 \geq -1; \\ 2y_1 + y_2 \geq -1; \\ y_1 \geq 0; \\ -y_2 \geq 0. \end{array} \right.$$

Оптимальный псевдоплан прямой задачи известен. Тогда оптимальное решение двойственной задачи может быть найдено по уравнению (2.25): $\mathbf{A}_i^T \mathbf{Y} = \mathbf{C}_i$, где \mathbf{A}_i — сопряженный с оптимальным псевдопланом прямой задачи базис, т. е. матрица, составленная из столбцов матрицы \mathbf{A} , соответствующих оптимальному псевдоплану (x_1, x_3) .

$$\mathbf{A}_i = \begin{bmatrix} 1 & 1 \\ 2 & 0 \end{bmatrix}.$$

Вектор \mathbf{C}_i составлен из соответствующих псевдоплану компонентов вектора \mathbf{C} : $\mathbf{C}_i = (-1, 0)^T$.

Таким образом, оптимальное решение \mathbf{Y}_{opt} двойственной задачи есть решение следующей системы уравнений:

$$\begin{cases} y_1 + 2y_2 = -1; \\ y_1 = 0; \end{cases} \quad \mathbf{Y}_{opt} = (0, -\frac{1}{2})^T.$$

Отметим, что оптимальное значение целевых функций прямой и двойственной задач совпадают:

$$8y_{1opt} + 10y_{2opt} = -x_{1opt} - x_{2opt} = -5.$$

На рис. 2.4 и 2.5 приведены графические решения прямой (пример 2.3) и двойственной (пример 2.4) задач. Следует обратить внимание на

то, что начальная опорная точка \tilde{X}^{B_0} прямой задачи не принадлежит ОДР (и в этом отличие двойственного симплекс-метода от обычного), а соответствующая ей опорная точка \tilde{Y}^{B_0} принадлежит ОДР.

Пример 2.5

Дополнительные условия в исходной задаче

Изменим формулировку примера 2 и добавим дополнительное ограничение (см. рис. 2.3.) $x_1 - 4x_2 \leq 0$.

Новая задача может быть решена двойственным симплекс-методом с использованием решения прежней задачи (см. табл. 2.12).

Приведем ограничение к канонической форме

$$x_1 - 4x_2 + x_6 = 0$$

и дополним окончательную симплекс-таблицу примера 2 (табл. 2.12) строкой для дополнительной базисной переменной x_6 , выразив ее через свободные переменные x_5, x_3, x_2 (табл. 2.15):

$$x_6 = 4x_2 - x_1 = x_3 + 6x_2 - 8.$$

Таблица 2.15

	x_5	x_3	x_2	b
x_4	1	-2	-3	6
x_1	0	-1	-2	8
x_6	0	1	6	-8
f	$-M$	-3	-2	24

Таблица 2.16

	x_5	x_3	x_6	b
x_4	1	-1,5	-0,5	2
x_1	0	-0,67	-0,33	5,33
x_2	0	-0,17	0,17	1,33
f	$-M$	-2,67	-0,33	21,33

Базис (x_4, x_1, x_6) не является допустимым, но может быть использован в качестве допустимого псевдоплана при дальнейшем решении задачи двойственным методом. Оптимальное решение найдено в результате всего одной итерации (табл. 2.16).

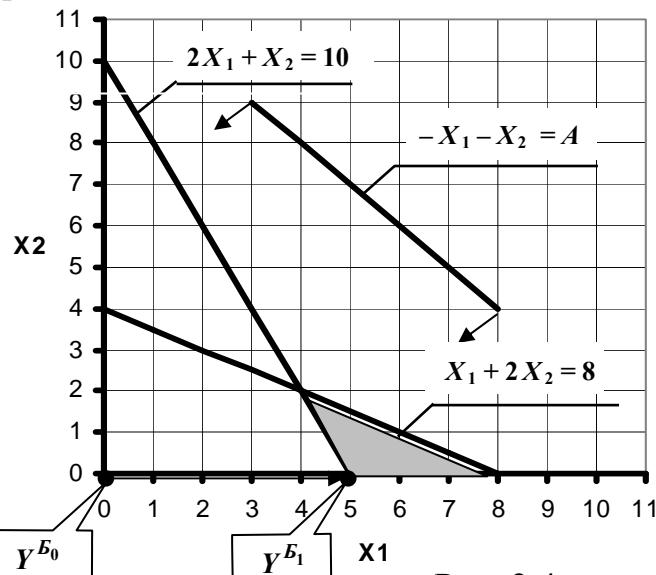


Рис. 2.4

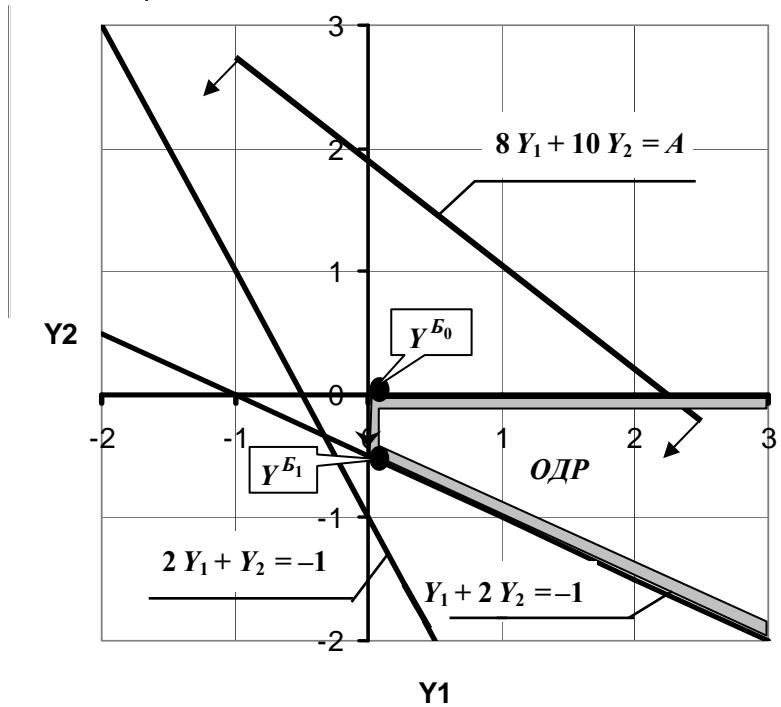


Рис. 2.5

3. ЗАДАЧИ НЕЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ (ПРИ ОТСУТСТВИИ ОГРАНИЧЕНИЙ)

3.1. ОСНОВНЫЕ ПОЛОЖЕНИЯ

1. Формулировка задачи. Общая задача безусловной оптимизации формулируется как задача определения вектора $X = (x_1, \dots, x_n)^T$, доставляющего экстремум (максимум или минимум) скалярной функции $f(X)$ векторного аргумента X , принадлежащего евклидову пространству:

$$f(X) \rightarrow \max, \quad X \in R^n, \quad (3.1)$$

$$f(X) \rightarrow \min, \quad X \in R^n, \quad (3.2)$$

где $f(X)$ — целевая функция.

2. Глобальный экстремум. Точка $X^* \in R^n$ называется точкой глобального экстремума $f(X)$, а именно:

$$\text{максимума, если } f(X^*) \geq f(X) \text{ при всех } X \in R^n; \quad (3.3)$$

$$\text{минимума, если } f(X^*) \leq f(X) \text{ при всех } X \in R^n; \quad (3.4)$$

3. Локальный экстремум. Точка $X^* \in R^n$ называется точкой локального экстремума, а именно:

$$\text{максимума, если } f(X^*) \geq f(X) \text{ при всех } X \in U_\epsilon(X^*); \quad (3.5)$$

$$\text{минимума, если } f(X^*) \leq f(X) \text{ при всех } X \in U_\epsilon(X^*); \quad (3.6)$$

где ϵ — любое число, строго большее нуля ($\epsilon > 0$), а

$$U_\epsilon(X^*) = \{X : \|X - X^*\| < \epsilon\} \text{ — шар радиуса } \epsilon \text{ с центром в } X^*.$$

4. Строгий экстремум. Если неравенства (3.3), (3.4) или (3.5),(3.6) выполняются как строгие при $X = X^*$, то говорят, что X^* — точка строгого экстремума, глобального или локального соответственно.

5. Изменение типа экстремума. Задача минимизации (3.2) функции $f(X)$ эквивалентна задаче максимизации функции $(-f(X))$:

$$(-f(X)) \rightarrow \max, \quad X \in R^n. \quad (3.7)$$

6. Условия существования экстремума $f(x)$ в точке X^ .*

Введем обозначения:

$$f'(X) = \text{grad } f(X) = \nabla f(X) = \left[\frac{\partial f}{\partial x_1}(X), \dots, \frac{\partial f}{\partial x_n}(X) \right]^T - \text{ вектор}$$

первых частных производных (градиент) функции $f(X)$ в точке X ,

$$f''(X) = H(X) = \left\{ \frac{\partial^2 f}{\partial x_i \partial x_j}(X) \right\}_{i,j=1,n} - \text{ вторая производная функции}$$

$f(X)$ в точке X , задаваемая квадратной матрицей (матрицей Гессе).

Теорема 1. Необходимое условие максимума первого порядка

Пусть $f(X)$ дифференцируема в точке $X^* \in R^n$. Тогда, если X^* — локальное решение задачи (3.1), то

$$f'(X^*) = 0, \quad (3.8)$$

а точка, удовлетворяющая этому условию, называется стационарной точкой функции $f(X)$.

Теорема 2. Достаточное условие максимума второго порядка

Пусть $f(X)$ дважды дифференцируема в точке $X^* \in R^n$. Тогда, если $f'(X^*) = 0$, матрица $H(X^*)$ отрицательно определена (полуопределенна), то X^* — строгое (нестрогое) локальное решение задачи (3.1).

Знакоопределенность квадратной матрицы A обусловлена свойством ее квадратичной формы $X^T A X = \langle AX, X \rangle$:

A положительно определена (полуопределенна), если для любого

$$X \in R^n, X \neq 0, \langle AX, X \rangle > 0 \quad (\langle AX, X \rangle \geq 0);$$

A отрицательно определена (полуопределенна), если для любого

$$X \in R^n, X \neq 0, \langle AX, X \rangle < 0 \quad (\langle AX, X \rangle \leq 0);$$

A не определена, если для ряда значений

$$X \in R^n, X \neq 0. \quad \langle AX, X \rangle > 0, \text{ а для остальных } \langle AX, X \rangle < 0.$$

Признаки знакоопределенности квадратной матрицы A. Матрица A положительно определена (полуопределенна), если все ее главные миноры (получающиеся при вычеркивании любого числа одноименных строк и столбцов из матрицы A) положительны (неотрицательны).

Матрица A отрицательно определена (полуопределенна), если матрица $(-A)$ является положительно определенной (полуопределенной). Матрица не определена, если отсутствуют признаки положительной или отри-

цательной определенности или полуопределенности.

Условия существования глобального решения задачи безусловной оптимизации. Если функция относится к классу вогнутых функций

$$f(\lambda X_1 + (1-\lambda)X_2) \geq \lambda f(X_1) + (1-\lambda)f(X_2)$$

или выпуклых функций

$$f(\lambda X_1 + (1-\lambda)X_2) \leq \lambda f(X_1) + (1-\lambda)f(X_2),$$

где, $X_1 \neq X_2$, $0 \leq \lambda \leq 1$, то сформулированные выше условия (теоремы 1 и 2) характеризуют глобальный максимум для вогнутых функций и глобальный минимум для выпуклых функций.

3.2. КЛАССИФИКАЦИЯ МЕТОДОВ БЕЗУСЛОВНОЙ ОПТИМИЗАЦИИ

Обобщенный алгоритм численного метода безусловной оптимизации, реализующий итеративную процедуру построения траектории, начинаящейся в точке $X^{(0)}$ и сходящейся к точке X^* максимума, представлен следующим соотношением:

$$X^{(i+1)} = X^{(i)} + t^{(i)} K^{(i)}, \quad (3.9)$$

где $K^{(i)}$ — вектор направления; $t^{(i)}$ — длина шага из точки.

Существенными классификационными признаками, позволяющими упорядочить множество возможных методов безусловной оптимизации, являются:

- метод выбора направления;
- метод определения длины шага.

По первому и наиболее важному признаку следует выделить пять классификационных групп:

- релаксационные методы;
- градиентные методы;
- методы сопряженных направлений;
- метод Ньютона;
- квазиньютоновские методы.

Отмеченные группы методов имеют некоторые общие свойства. Часть методов (градиентные, сопряженных направлений и квазиньютоновские) используют информацию о градиенте $f'(X^{(i)})$, метод Ньютона использует данные о $f'(X^{(i)})$ и матрице вторых частных производных $f''(X^{(i)})$. Кроме того, метод Ньютона и квазиньютоновские методы объединяет общий подход к определению эффективного направления, основанный на квадратичной аппроксимации целевой функции в точке $X^{(i)}$.

Подробное описание и сравнительный анализ методов приводятся ниже.

В зависимости от способов определения длины шага методы можно разделить на следующие четыре группы:

- с постоянной длиной шага;
- с регулировкой длины шага;
- с оптимизацией длины шага в выбранном направлении;
- с использованием методов одномерного поиска для определения длины шага.

Хотя приведенная классификация учитывает ограниченный набор признаков, она позволяет выделить основополагающие методы безусловной оптимизации. Более детальные варианты классификации приведены в [5, 6].

3.3. МЕТОДЫ ВЫБОРА НАПРАВЛЕНИЯ

Ниже рассматриваются выделенные в классификации методы выбора направления, ориентированные на решение задачи поиска максимума.

Релаксационные методы

Эта группа методов предполагает выбор направлений поиска, параллельных координатным осям. Таким образом, задача оптимизации в n -мерном пространстве \mathbb{R}^n сводится к совокупности задач одномерной по-координатной оптимизации. На каждом шаге оптимизации реализуются n итераций, на каждой из которых изменению подвергается лишь одна из координат вектора, поэтому приращение Δf целевой функции на l -й итерации ($i+1$)-го шага можно представить как функцию приращения скалярного аргумента $\Delta x_l^{(i+1)} = x_l^{((i+1),l)} - x_l^{(i,l)}$:

$$\Delta f = f(X^{((i+1),l)}) - f(X^{((i+1),(l-1))}) = \Delta f(\Delta x_l).$$

Оптимальным значением Δx_l является решение задачи

$$\Delta x_l = \arg \max_{(\Delta x_l)} \Delta f(\Delta x_l),$$

которое может быть получено любым из возможных методов, например, одним из методов одномерного поиска, не требующих вычисления частных производных, либо аналитически с использованием квадратичной аппроксимации $f(X)$.

В последнем случае получим вариант рассмотренного ниже метода

Ньютона, где вместо градиента имеем $\frac{\partial f}{\partial x_l}(X^{(i,(l-1))})$, а вместо

$$f''(X^{(i,(l-1))})^{-1} = H^{-1}(X^{(i,(l-1))}) \Rightarrow \left[\frac{\partial^2 f}{\partial x_l^2}(X^{(i,(l-1))}) \right]^{-1}.$$

Тогда $x_l^{(i,l)} = x_l^{(i,(l-1))} - \frac{\partial f}{\partial x_l}(X^{(i,(l-1))}) \cdot \left[\frac{\partial^2 f}{\partial x_l^2}(X^{(i,(l-1))}) \right]^{-1}$. (3.10)

Градиентные методы

Эти методы основаны на линейной аппроксимации целевой функции $f(X)$ в каждой точке $X^{(i)}$ траектории поиска. В этом случае несложно показать, что оптимальным направлением поиска максимума $f(X)$ явится градиентное направление.

Воспользуемся линейной аппроксимацией целевой функции в окрестности точки:

$$f(X^{(i+1)}) \approx f(X^{(i)}) + \langle f'(X^{(i)}), (X^{(i+1)} - X^{(i)}) \rangle.$$

Так как $X^{(i+1)} = X^{(i)} + t^{(i)} K^{(i)}$, то

$$\Delta f = f(X^{(i+1)}) - f(X^{(i)}) = t^{(i)} \langle f'(X^{(i)}), (X^{(i+1)} - X^{(i)}) \rangle, \quad (3.11)$$

а задача выбора оптимального вектора направления $K^{(i)}$ с учетом (3.11) сводится к задаче условной оптимизации в следующей формулировке:

$$\max_{(K^{(i)})} \left\{ \sum \frac{\partial f}{\partial x_k}(X^{(i)}) K_k^{(i)} \right\}, \quad (3.12)$$

при условии

$$\sum_{k=1}^n [K_k^{(i)}]^2 = 1,$$

где $K^{(i)} = (K_1^{(i)}, \dots, K_n^{(i)})^T$.

Нормирующее условие, связывающее величины составляющих $K_k^{(i)}$ вектора $K^{(i)}$, необходимо, так как задача безусловной оптимизации линейной функции $\sum_{k=1}^n \frac{\partial f}{\partial x_k}(X^{(i)}) K_k^{(i)}$ не имеет решения.

Решив задачу (3.12) любым из методов, например, методом неопределенных множителей Лагранжа, получим

$$K_k^{(i)} = \frac{\frac{\partial f}{\partial x_k}(X^{(i)})}{\sqrt{\sum_{k=1}^n \left[\frac{\partial f}{\partial x_k}(X^{(i)}) \right]^2}},$$

или в векторной форме

$$\mathbf{K}^{(i)} = \frac{\mathbf{f}'(X^{(i)})}{\|\mathbf{f}'(X^{(i)})\|}.$$

Метод Ньютона

В этом методе выбор направления основан на использовании квадратичной аппроксимации целевой функции $f(X)$ в точках $X^{(i)}$ траектории поиска. С учетом квадратичной аппроксимации целевой функции в окрестности $X^{(i)}$ можем записать:

$$f(X^{(i+1)}) \approx f(X^{(i)}) + \langle \mathbf{f}'(X^{(i)}), (X^{(i+1)} - X^{(i)}) \rangle + \\ + \frac{1}{2} \langle \mathbf{f}''(X^{(i+1)})(X^{(i+1)} - X^{(i)}), (X^{(i+1)} - X^{(i)}) \rangle.$$

Отметим, что $\Delta f = f(X^{(i+1)}) - f(X^{(i)}) = \Delta f(\Delta X)$,

где $\Delta X = X^{(i+1)} - X^{(i)}$ и решим задачу поиска

$$\Delta X = \arg \max_{(\Delta X)} \{ \Delta f(\Delta X) \}.$$

Для этого используем необходимое условие оптимизации:

$$\mathbf{f}'(X^{(i+1)}) = \mathbf{f}'(X^{(i)}) + \mathbf{f}''(X^{(i)})(X^{(i+1)} - X^{(i)}) = \mathbf{0}, \text{ отсюда} \\ \mathbf{f}''(X^{(i)})(X^{(i+1)} - X^{(i)}) = -\mathbf{f}'(X^{(i)}). \quad (3.13)$$

Умножим правую и левую части полученного равенства на $(\mathbf{f}''(X^{(i)}))^{-1}$ и получим решение задачи:

$$X^{(i+1)} - X^{(i)} = -(\mathbf{f}''(X^{(i)}))^{-1} \mathbf{f}'(X^{(i)}) \quad (3.14)$$

при условии, что матрица вторых производных $H(X^{(i)}) = \mathbf{f}''(X^{(i)})$ от-

рицательно определенная и $\det H(X^{(i)}) \neq 0$.

Если $f(X)$ квадратичная функция, то решение (3.14) позволяет найти максимум $f(X)$ за один шаг. Таким образом, независимо от расположения начальной точки $X^{(0)}$ точка

$$X^{(1)} = X^{(0)} - (f''(X^{(0)}))^{-1} f'(X^{(0)})$$

становится координатой максимума целевой функции.

Если $f(X)$ – неквадратичная функция, то правая часть (3.14) определяет метод выбора направления из точки $X^{(i)}$:

$$K^{(i)} = -(f''(X^{(i)}))^{-1} f'(X^{(i)}). \quad (3.15)$$

Метод сопряженных направлений

Метод ориентирован на решение задач с квадратичными функциями и основан на построении векторов направлений поиска, обладающих свойством сопряженности относительно матрицы Гессе.

Пусть задана квадратичная целевая функция

$$f(x) = A + b^T X + \frac{1}{2} X^T H X, \quad (3.16)$$

где H — симметричная отрицательно определенная матрица. Вектор $X \in R^n$ всегда может быть представлен в виде линейной комбинации линейно независимых векторов t_j :

$$X = \sum_{j=1}^n t_j z_j = TZ, \quad (3.17)$$

где T — матрица ($n \times n$), составленная из столбцов t_j и характеризуемая тем, что квадратичная форма $Q(X) = \langle HX, X \rangle = X^T H X$ функции (3.16), выраженная с использованием преобразования (3.17) как $Q(Z)$ приводится к сумме полных квадратов:

$$Q(X) = Q(Z) = \sum_{j=1}^n d_{jj} z_j^2. \quad (3.18)$$

В этом случае задача отыскания оптимума (3.16) может быть сведена к n задачам одномерного поиска по преобразованным координатным направлениям t_j . С учетом (3.17)

$$Q(X) = X^T H X = Z^T T^T H T Z = Z^T D Z. \quad (3.19)$$

Для того, чтобы обеспечить представление $Q(Z)$ в форме (3.18),

матрица $\mathbf{D} = \mathbf{T}^T \mathbf{H} \mathbf{T}$ должна быть диагональной.

Элементы d_{jk} матрицы \mathbf{D} с учетом (3.19) вычисляют по формуле

$$d_{jk} = \mathbf{t}_j^T \mathbf{H} \mathbf{t}_k = \langle \mathbf{H} \mathbf{t}_k, \mathbf{t}_j \rangle, \quad (3.20)$$

и если \mathbf{D} — диагональная матрица, то

$$d_{jk} = \begin{cases} d_{jj}, & k = j; \\ 0, & k \neq j. \end{cases}$$

Таким образом, свойство сопряженности направлений \mathbf{t}_j поиска относительно матрицы \mathbf{H} , выраженное в форме равенства $\langle \mathbf{H} \mathbf{t}_k, \mathbf{t}_j \rangle = 0$, является основополагающим в методе сопряженных направлений.

Существуют различные подходы к построению сопряженных направлений. Один из них, метод сопряженных градиентов, предлагает наиболее простую алгоритмическую схему решения этой задачи.

Метод сопряженных градиентов (метод Флетчера-Ривса)

В этом методе последовательность взаимно-сопряженных направлений начинается с градиентного направления из точки $\mathbf{X}^{(0)}$:

$$\mathbf{K}^{(0)} = \mathbf{f}'(\mathbf{X}^{(0)}).$$

На всех последующих шагах вектор направления определяют по рекуррентной формуле

$$\mathbf{K}^{(i)} = \mathbf{f}'(\mathbf{X}^{(i)}) + \omega_i \mathbf{K}^{(i-1)}, \quad (3.21)$$

где скалярный коэффициент ω_i находят из условия $\langle \mathbf{H} \mathbf{K}^{(j)}, \mathbf{K}^{(i)} \rangle = 0$, $j = \overline{0, i-1}$ сопряженности направления $\mathbf{K}^{(i)}$ со всеми ранее выработанными направлениями поиска и вычисляют следующим образом [5, 6]:

$$\omega_i = \frac{\| \mathbf{f}'(\mathbf{X}^{(i)}) \|^2}{\| \mathbf{f}'(\mathbf{X}^{(i-1)}) \|^2} \quad (3.22)$$

Из принципа выбора направлений поиска в методе сопряженных направлений следует, что при оптимизации квадратичной функции число итераций не превышает числа компонентов вектора \mathbf{X} .

Квазиньютоновские методы (методы переменной метрики)

Эти методы, так же, как и метод Ньютона, основаны на квадратичном приближении целевой функции, но в отличие от него предполагают

аппроксимацию матрицы Гессе или обратной к ней, осуществляющую с использованием только первых производных целевой функции.

Таким образом, в методах переменной метрики точки траектории поиска экстремума $f(X)$ определяют по следующему правилу:

$$X^{(i+1)} = X^{(i)} + t^{(i)} K^{(i)} = X^{(i)} - t^{(i)} \eta(X^{(i)}) f'(X^{(i)}), \quad (3.23)$$

где $\eta(X^{(i)}) \approx (f''(X^{(i)}))^{-1} = H^{-1}(X^{(i)})$. (3.24)

Рассмотрим основные соотношения, позволяющие сформулировать метод построения аппроксимации $\eta(X^{(i)})$.

Используя квадратичную аппроксимацию целевой функции

$$\begin{aligned} f(X) \approx & f(X^{(i)}) + \langle f'(X^{(i)}), (X - X^{(i)}) \rangle + \\ & + \frac{1}{2} \langle f''(X^{(i)})(X - X^{(i)}), (X - X^{(i)}) \rangle, \end{aligned}$$

можем записать $f'(X^{(i+1)}) = f'(X^{(i)}) + f''(X^{(i)})(X^{(i+1)} - X^{(i)})$

или $f'(X^{(i+1)}) - f'(X^{(i)}) = f''(X^{(i)})(X^{(i+1)} - X^{(i)})$.

Умножим правую и левую части равенства на $(f''(X^{(i)}))^{-1}$ и получим систему n линейных уравнений, которым должны удовлетворять элементы аппроксимирующей матрицы

$$(X^{(i+1)} - X^{(i)}) = (f''(X^{(i)}))^{-1} [f'(X^{(i+1)}) - f'(X^{(i)})]. \quad (3.25)$$

Основываясь на соотношении (3.25), связывающем на каждом шаге поиска приращение аргумента и градиента целевой функции с элементами матрицы $(f''(X^{(i)}))^{-1}$, можно поставить задачу формирования аппроксимирующей матрицы $\eta(X^{(i)})$ по известным к началу i -го шага значениям

$$X^{(i-1)}, X^{(i)}, f'(X^{(i-1)}), f'(X^{(i)}).$$

Пусть $\eta(X^{(i)}) = \eta(X^{(i-1)}) + \Delta\eta(X^{(i-1)})$. Тогда, введя обозначения

$$\eta(X^{(i)}) = \eta^{(i)}; \quad \Delta\eta(X^{(i)}) = \Delta\eta^{(i)}; \quad \Delta X^{(i-1)} = X^{(i)} - X^{(i-1)};$$

$$\Delta g^{(i-1)} = f'(X^{(i)}) - f'(X^{(i-1)}),$$

с учетом (3.25) можно записать :

$$\Delta X^{(i-1)} = \eta^{(i)} \Delta g^{(i-1)} = i [\eta^{(i-1)} + \Delta\eta^{(i-1)}] \Delta g^{(i-1)}, \text{ откуда}$$

$$\Delta\eta^{(i-1)} \Delta g^{(i-1)} = \Delta X^{(i-1)} - \eta^{(i-1)} \Delta g^{(i-1)}. \quad (3.26)$$

Формируемая на i -м шаге поправка $\Delta\eta^{(i-1)}$ должна удовлетворять уравнению (3.26). Анализ (3.26) показывает, что множество матриц $\Delta\eta^{(i-1)}$, построенных по формуле

$$\Delta\eta^{(i-1)}i = i \frac{\Delta X^{(i-1)}Y^T}{Y^T \Delta g^{(i-1)}} - \frac{\eta^{(i-1)} \Delta g^{(i-1)}Z^T}{Z^T \Delta g^{(i-1)}}, \quad (3.27)$$

где Y и Z — произвольные векторы размерности $(n \times 1)$, является решением уравнения (3.26).

Выбор векторов Y и Z определяет метод переменной метрики.

Метод Бройдена

В этом методе

$$Y = Z = \Delta X^{(i-1)} - \eta^{(i-1)} \Delta g^{(i-1)},$$

матрица $\Delta\eta^{(i-1)}$ имеет ранг 1 и вычисляется по формуле:

$$\begin{aligned} \Delta\eta^{(i-1)} &= [(\Delta X^{(i-1)}) - \eta^{(i-1)} (\Delta g^{(i-1)})] \times \\ &\times \frac{[(\Delta X^{(i-1)} - \eta^{(i-1)} (\Delta g^{(i-1)}))^T]}{[\Delta X^{(i-1)} - \eta^{(i-1)} (\Delta g^{(i-1)})]^T (\Delta g^{(i-1)})}. \end{aligned} \quad (3.28)$$

Метод Дэвидона – Флемчера – Пауэла (ДФП)

В этом методе

$$Y = \Delta X^{(i-1)}, \quad Z = \eta^{(i-1)} \Delta g^{(i-1)},$$

матрица $\Delta\eta^{(i-1)}$ имеет ранг 2 и вычисляется по формуле

$$\Delta\eta^{(i-1)} = A^{(i-1)} - B^{(i-1)}, \quad (3.29)$$

где

$$A^{(i-1)} = \frac{(\Delta X^{(i-1)})(\Delta X^{(i-1)})^T}{(\Delta X^{(i-1)})^T (\Delta g^{(i-1)})}; \quad (3.30)$$

$$B^{(i-1)} = \frac{\eta^{(i-1)} (\Delta g^{(i-1)}) (\Delta g^{(i-1)})^T (\eta^{(i-1)})^T}{(\Delta g^{(i-1)})^T (\eta^{(i-1)})^T (\Delta g^{(i-1)})}. \quad (3.31)$$

В качестве начального приближения в методах переменной метрики обычно применяется взятая со знаком минус единичная матрица ($\eta^{(0)} = -E_{[n \times n]}$) с учетом того, что все рассматриваемые методы предназначены для максимизации целевой функции, и начальное приближение

$\eta^{(0)}$ матрицы $(f''(X^{(0)}))^{-1}$ должно удовлетворять требованию отрицательной определенности.

В таблице 3.1 приведены соотношения, определяющие выбор направления поиска для рассмотренных методов.

3.4. МЕТОДЫ ВЫБОРА ДЛИНЫ ШАГА

Ниже рассматриваются основные методы выбора длины шага, ориентированные на решение задачи поиска максимума.

Постоянная длина шага, $t^{(i)} = t = const$

Наиболее простой в алгоритмическом плане метод обладает недостатком, связанным с необходимостью подбора подходящего значения t , обеспечивающего допустимую погрешность определения координат X^* экстремума. В сочетании с выбором градиентного направления поиска (простейший градиентный метод) постоянная длина шага обуславливает медленную сходимость к точке X^* в ϵ -окрестности X^* , так как ΔX мало вследствие малости $f'(X^* \pm \epsilon)$.

Алгоритм регулировки длины шага

$$t_{0 < \delta < 1}^{(i+1)} = \begin{cases} t^{(i)}, & f(X^{(i+1)}) - f(X^{(i)}) \geq \delta \cdot t^{(i)} \langle f'(X^{(i)}), K^{(i)} \rangle; \\ \frac{t^{(i)}}{2}, & f(X^{(i+1)}) - f(X^{(i)}) < \delta \cdot t^{(i)} \langle f'(X^{(i)}), K^{(i)} \rangle. \end{cases} \quad (3.32)$$

Параметрами алгоритма являются $t^{(0)}$ и δ , настройка алгоритма состоит в удачном выборе этих параметров.

Начальные значения, по существу, определяют постоянную длину шагов, удаленных от точки максимума целевой функции.

Выбор параметра δ влияет на скорость и характер сходимости последовательности $\{X^{(i)}\}$ к точке X^* .

Оптимизация длины шага в выбранном направлении поиска с использованием квадратичной аппроксимации целевой функции

Задачу выбора длины $t^{(i)}$ i -го шага можно сформулировать как задачу безусловной оптимизации:

$$t^{(i)} = \arg \max_{(t)} \{\Delta f = f(X^{(i)} + t K^{(i)}) - f(X^{(i)})\}. \quad (3.33)$$

Для решения этой задачи воспользуемся необходимым условием

Таблица 3.1

Методы выбора направления	Определение вектора направления
Релаксационные	$\mathbf{K}^{(i,l)} = (K_1^{(i,l)}, \dots, K_n^{(i,l)})^T,$ <p>где $K_k^{(i,l)} = \begin{cases} \frac{\partial f}{\partial x_l}(X^{(ik)}), & k = l, \\ 0, & k \neq l \end{cases}$</p>
Градиентные	$\mathbf{K}^{(i)} = \mathbf{f}'(X^{(i)})$
Ньютона	$\mathbf{K}^{(i)} = -(\mathbf{f}''(X^{(i)}))^{-1} \mathbf{f}'(X^{(i)})$
Сопряженных градиентов	$\mathbf{K}^{(0)} = \mathbf{f}'(X^{(0)}), \quad \mathbf{K}^{(i)} =$ $= \mathbf{f}'(X^{(i)}) + \left[\ \mathbf{f}'(X^{(i)})\ ^2 / \ \mathbf{f}'(X^{(i-1)})\ ^2 \right] \mathbf{K}^{(i-1)}$
Квазиньютоновские (переменной метрики)	$\mathbf{K}^{(i)} = -\eta^{(i)} \mathbf{f}'(X^{(i)}),$ $\eta^{(i)} = \eta^{(i-1)} + \Delta \eta^{(i-1)}, \quad \eta^{(0)} = -E$
	<p>Метод Бройдена:</p> $\Delta \eta^{(i-1)} =$ $[\Delta X^{(i-1)} - \eta^{(i-1)} (\Delta g^{(i-1)})] [\Delta X^{(i-1)} - \eta^{(i-1)} (\Delta g^{(i-1)})]^T.$ $[\Delta X^{(i-1)} - \eta^{(i-1)} (\Delta g^{(i-1)})]^T (\Delta g^{(i-1)})$
	<p>Метод ДФП:</p> $\Delta \eta^{(i-1)} = \mathbf{A}^{(i-1)} - \mathbf{B}^{(i-1)},$ <p>где $\mathbf{A}^{(i-1)} = \frac{(\Delta X^{(i-1)})(\Delta X^{(i-1)})^T}{(\Delta X^{(i-1)})^T (\Delta g^{(i-1)})},$</p> $\mathbf{B}^{(i-1)} = \frac{\eta^{(i-1)} \Delta g^{(i-1)} (\Delta g^{(i-1)})^T (\eta^{(i-1)})^T}{(\Delta g^{(i-1)})^T (\eta^{(i-1)})^T \Delta g^{(i-1)}}.$
	<p>Для обоих методов:</p> $\Delta X^{(i-1)} = X^{(i)} - X^{(i-1)},$ $\Delta g^{(i-1)} = \mathbf{f}'(X^{(i)}) - \mathbf{f}'(X^{(i-1)}).$

(3.13) существования максимума квадратичной аппроксимации приращения $\Delta f(t)$ целевой функции $f(X)$:

$$\frac{\partial \Delta f}{\partial t} = \frac{\partial \Delta f}{\partial \Delta X} \frac{\partial \Delta X}{\partial t} = \langle f'(X^{(i)}), K^{(i)} \rangle + t \langle f''(X^{(i)}) K^{(i)}, K^{(i)} \rangle = 0,$$

откуда

$$t^{(i)} = -\frac{\langle f'(X^{(i)}), K^{(i)} \rangle}{\langle f''(X^{(i)}) K^{(i)}, K^{(i)} \rangle}. \quad (3.34)$$

Использование (3.34) в сочетании с выбором градиентного направления приводит к методу, известному как метод наискорейшего подъема-спуска.

Если выбор направления поиска осуществляется методом Ньютона ($K^{(i)} = -(f''(X^{(i)}))^{-1} f'(X^{(i)})$), то использование (3.34) приводит к постоянной длине шага, тождественно равной единице. Метод Ньютона может быть использован совместно с другими методами оптимизации длины шага в выбранном направлении, либо с алгоритмами регулировки длины шага [9].

Использование методов одномерного поиска для определения длины шага

Задача (3.33) может быть решена любым из известных методов одномерного поиска.

В формулировке задачи одномерного поиска должен быть задан начальный интервал неопределенности длины шага $t^{(i)}$ аргумента целевой функции, включающей в себя точку $t_*^{(i)}$ оптимума:

$$t_*^{(i)} \in \left[t_a^{(i)}, t_b^{(i)} \right].$$

Рассмотрим этапы поиска начального интервала неопределенности и определения $t_*^{(i)}$ с использованием метода золотого сечения.

Начальный интервал $\left[t_a^{(i)}, t_b^{(i)} \right]$ неопределенности может быть получен с помощью серии возрастающих шагов в направлении $K^{(i)}$. Так как в методе золотого сечения интервал неопределенности уменьшается с каждым шагом приблизительно в 1,62 раза, то именно это соотношение целесообразно использовать при определении длины пробных шагов.

$$\begin{cases} t_0^{(i)} = 0, t_1^{(i)} = 1; \\ t_{k+1}^{(i)} = t_k^{(i)} + 1,62(t_k^{(i)} - t_{k-1}^{(i)}), \quad k = 1, 2, \dots \end{cases}$$

Таким образом, изменение длины пробных шагов осуществляется в соответствии с правилом:

$$t_{k+1}^{(i)} - t_k^{(i)} = (1,62)^k, k = 1, 2, \dots$$

В процессе реализации пробных шагов определяют последовательно анализируемый ряд значений $f(X^{(i)} + t_k^{(i)} K^{(i)})$ целевой функции. В этом ряду необходимо обнаружить точку нарушения монотонного возрастания целевой функции от шага к шагу. Если, например, имеет место следующее соотношение между пробными значениями функции:

$$\begin{aligned} f(X^{(i)} + t_0^{(i)} K^{(i)}) &< f(X^{(i)} + t_1^{(i)} K^{(i)}) < \dots < f(X^{(i)} + t_{r-2}^{(i)} K^{(i)}) < \\ &< f(X^{(i)} + t_{r-1}^{(i)} K^{(i)}) > f(X^{(i)} + t_r^{(i)} K^{(i)}), \end{aligned}$$

то начальным интервалом неопределенности для оптимальной длины $t_*^{(i)}$ шага является интервал $\left[t_{r-2}^{(i)}, t_r^{(i)} \right]$.

Далее начинается процедура уменьшения начального интервала неопределенности в соответствии с методом золотого сечения [5, 6, 10].

При заданной погрешности δ определения оптимальной длины шага и известном интервале неопределенности число циклов r метода золотого сечения определяют по формуле:

$$r = \left\lceil \ln \frac{t_b^{(i)} - t_a^{(i)}}{\delta} / \ln \tau \right\rceil + 1, \quad \tau = 1,62,$$

где обозначение $\lceil a \rceil$ следует понимать как целую часть числа a .

3.5. СРАВНИТЕЛЬНЫЙ АНАЛИЗ МЕТОДОВ БЕЗУСЛОВНОЙ ОПТИМИЗАЦИИ

Сравнение методов нелинейной оптимизации целесообразно проводить по следующим критериям: характеристика скорости сходимости и устойчивости метода (монотонное возрастание целевой функции от шага к шагу), которые могут быть обоснованы теоретически, и особенности алгоритмических схем — простота реализации, степень сложности вычислительных процедур, количество часов, объем памяти ЭВМ, требуемый при программировании метода.

Метод наискорейшего спуска характеризуется устойчивостью и линейной скоростью сходимости [6], но при практической реализации может привести к большому числу шагов, так как длина шага, пропорциональная градиенту функции, становится малой в окрестности точки X^* . Кроме то-

го, число шагов в значительной степени зависит от расположения начальной точки $X^{(0)}$.

Релаксационные методы обладают соизмеримой, а в ряде случаев худшей, чем у градиентных методов, сходимостью. Их достоинство состоит в возможности включения не требующих вычисления частных производных целевой функции методов одномерного поиска в процедуру покординатной оптимизации.

Метод Ньютона характеризуется квадратичной скоростью сходимости [6], и, как уже отмечалось, для квадратичных целевых функций сходится за один шаг независимо от расположения начальной точки $X^{(0)}$. Недостатком метода является его вычислительная сложность, обусловленная необходимостью определения на каждом шаге матрицы Гессе и ее обращения (для неквадратичных функций).

Метод сопряженных градиентов так же как и метод Ньютона, наиболее удобен для оптимизации квадратичных функций. В этом случае метод гарантирует, что число шагов не превысит размерности n вектора $X \in R^n$. Алгоритм, реализующий метод сопряженных градиентов, не требует используемых в методе Ньютона сложных процедур вычисления и обращения матрицы Гессе и представляет невысокий уровень требований к объему памяти ЭВМ.

Квазиньютоновские методы основанные, как и метод Ньютона, на квадратичной аппроксимации целевой функции, обладают положительными свойствами последнего (близкой к квадратичной скоростью сходимости, высокой эффективностью применительно к квадратичным целевым функциям), но не требует вычисления матрицы вторых производных. Недостатком методов является необходимость хранения аппроксимирующей матрицы $\eta_{[n \times n]}^{(i)}$ на каждом шаге.

Метод Ньютона и квазиньютоновские методы имеют общий недостаток, который может проявиться при оптимизации неквадратичных функций: потеря отрицательной определенности матрицы вторых производных $f''(X^{(i)})$ или $\eta^{(i)}$ и, как следствие, потеря устойчивости метода. Поэтому при использовании этих методов целесообразно дополнить алгоритм процедурой модификации матрицы $f''(X^{(i)})$ или $\eta^{(i)}$, обеспечивающей отрицательную определенность матрицы, используемой для вычисления вектора направления [5].

Следует отметить, что абсолютная предпочтительность того или иного метода для любой задачи оптимизации отсутствует. Выбор метода оптимизации должен осуществляться с учетом конкретных особенностей

целевой функции. Сравнение большого числа различных алгоритмов нелинейной оптимизации приведено в [5,6].

Эффективность алгоритма оптимизации зависит не только от метода, положенного в его основу, но и от удачного выбора оператора проверки условий останова. Включение этого оператора в алгоритмическую схему оптимизации функции преследует ту же цель, что и проверка необходимых и достаточных условий существования экстремума в аналитическом решении той же задачи. Конкретное содержание рассматриваемого оператора зависит от специфики алгоритма поиска и, как правило, основано на использовании тех промежуточных данных, которые необходимы для выполнения стандартного шага алгоритма.

Ниже приводятся основные варианты условий останова с краткими комментариями.

а) $\|f'(X^{(i)})\| \leq \delta$ — условие, позволяющее проверить принадлежность точки некоторой окрестности стационарной точки X^* целевой функции.

б) $\|X^{(i+1)} - X^{(i)}\| \leq \varepsilon$ — по мере приближения точек траектории поиска к стационарной точке приращение $\Delta X = X^{(i+1)} - X^{(i)}$ уменьшается, поэтому выполнение такого условия можно рассматривать как свидетельство близости точки $X^{(i+1)}$ к точке X^* .

в) $|f(X^{(i+1)}) - f(X^{(i)})| \leq \delta$ — это условие целесообразно использовать в методах прямого поиска (не использующих значения частных производных $f(X)$).

Отметим, что обычно используемые в алгоритмах поиска условия останова ориентированы на проверку необходимых условий существования экстремума: $f'(X^*) = \mathbf{0}$. Это относится и к приведенным выше вариантам условий останова.

Проверку достаточных условий существований экстремума (знакопределенности матрицы Гессе) для квадратичной целевой функции необходимо проводить заранее (при постановке задачи). Если $f(X)$ функция более высокого порядка, то соответствующая проверка должна быть предусмотрена в алгоритме.

3.6. ПРИМЕР РЕШЕНИЯ ЗАДАЧИ БЕЗУСЛОВНОЙ ОПТИМИЗАЦИИ

Рассмотрим задачу безусловной минимизации функции

$$f(X) = \{8x_1^2 + 4x_1x_2 + 5x_2^2\}, \\ X^{(0)} = (10, 10)^T$$

с использованием описанных выше методов выбора направления и длины шага. Так как методы выбора направления сформулированы как методы максимизации $f(X)$, перейдем к задаче поиска максимума

$$f(X) = -\{8x_1^2 + 4x_1x_2 + 5x_2^2\}.$$

Убедимся в выполнении достаточных условий существования максимума в стационарной точке X^* .

Целевая функция $f(X)$ квадратичная, поэтому матрица Гессе постоянна в любой точке $X \in R^n$:

$$f''(X) = H = \left\{ \frac{\partial^2 f}{\partial x_i \partial x_j} \right\}_{i,j=1,2} = \begin{bmatrix} -16 & -4 \\ -4 & -10 \end{bmatrix}.$$

Матрица $-H$ положительно определенная, так как главные миноры $-H$ положительны, следовательно, матрица H отрицательно определенная и $f(X)$ действительно имеет максимум в точке X^* . Предусмотрим единое правило останова: $\|f'(X^{(i)})\| \leq 0,1$.

Релаксационный метод

Используем формулу (3.10) для расчета координат $X^{(i)}$ траектории поиска максимума $f(X)$.

Шаг 1, итерация 1

$$\begin{aligned} \frac{\partial f}{\partial x_1} &= -16x_1 - 4x_2; & \frac{\partial^2 f}{\partial x_2^2} &= -10; \\ \frac{\partial f}{\partial x_2} &= -10x_2 - 4x_1; & \Delta x_1^{(0,1)} &= -\frac{-160 - 40}{-16} = -12,5; \\ \frac{\partial^2 f}{\partial x_1^2} &= -16; & X^{(1,1)} &= \begin{bmatrix} 10 \\ 10 \end{bmatrix} + \begin{bmatrix} -12,5 \\ 0 \end{bmatrix} = \begin{bmatrix} -2,5 \\ 10,0 \end{bmatrix}. \end{aligned}$$

Шаг 1, итерация 2

$$\Delta x_2^{(0,2)} = -\frac{-100+10}{-10} = -9,0;$$

$$X^{(1,2)} = \begin{bmatrix} -2,5 \\ 10,0 \end{bmatrix} + \begin{bmatrix} 0 \\ -9,0 \end{bmatrix} = \begin{bmatrix} -2,5 \\ 1,0 \end{bmatrix}.$$

Данные по остальным шагам приведены в табл. 3.2, а траектория поиска — на рис. 3.1.

Таблица 3.2

Номер шага i	Номер итерации l	$x_1^{(i,l)}$	$x_2^{(i,l)}$	$f(X^{(i,l)})$	$\ f'(X^{(i,l)})\ $
1	1	-2,5	10,0	450,0	90
	2	-2,5	1,0	45,0	36
2	1	-0,25	1,0	4,5	9
	2	-0,25	0,1	0,45	3,6
3	1	-0,025	0,1	0,045	0,9
	2	-0,025	0,01	0,0045	0,36
4	1	-0,0025	0,01	0,00045	0,09
	2	-0,0025	0,001	0,000045	0,036

Метод наискорейшего подъема-спуска

Шаг 1

$$K^{(0)} = f'(X^{(0)}) = [-200 \ -140]^T,$$

$$t^{(0)} = -\frac{\langle f'(X^{(0)}), f'(X^{(0)}) \rangle}{\langle Hf'(X^{(0)}), f'(X^{(0)}) \rangle} =$$

$$= -\frac{\left\langle \begin{bmatrix} -200 \\ -140 \end{bmatrix}, \begin{bmatrix} -200 \\ -140 \end{bmatrix} \right\rangle}{\left\langle \begin{bmatrix} -16 & -4 \\ -4 & -10 \end{bmatrix} \begin{bmatrix} -200 \\ -140 \end{bmatrix}, \begin{bmatrix} -200 \\ -140 \end{bmatrix} \right\rangle} \cong 0,056,$$

$$X^{(1)} \cong \begin{bmatrix} 10 \\ 10 \end{bmatrix} + 0,056 \begin{bmatrix} -200 \\ -140 \end{bmatrix} = \begin{bmatrix} -1,240 \\ 2,118 \end{bmatrix}.$$

Данные по остальным шагам приведены в табл. 3.3, а траектория поиска — на рис. 3.1.

Таблица 3.3

Номер шага i	$x_1^{(i)}$	$x_2^{(i)}$	$f(X^{(i)})$	$\ f'(X^{(i)})\ $
1	-1,2453	2,1283	-24,4528	19,8990
2	0,1438	0,1438	-0,3517	3,5116
3	-0,0179	0,0306	-0,0051	0,2862
4	0,0021	0,0021	-0,0001	0,0505

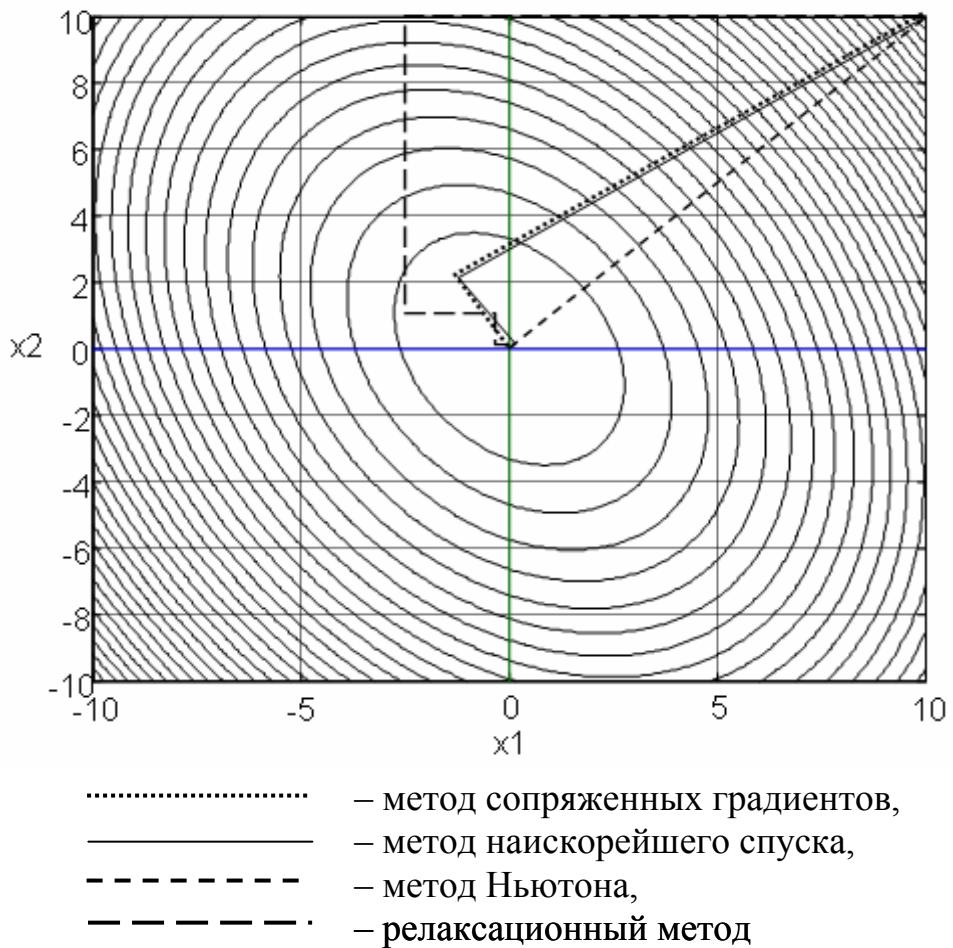


Рис. 3.1

Метод сопряженных градиентов

Шаг 1. Этот шаг полностью совпадает с первым шагом в методе наискорейшего подъема-спуска.

Шаг 2.

$$X^{(2)} = X^{(1)} + t^{(1)} K^{(1)} ;$$

$$K^{(1)} = f'(X^{(1)}) + \frac{\|f'(X^{(1)})\|^2}{\|f'(X^{(0)})\|^2} f'(X^{(0)}) \cong$$

$$\cong \begin{bmatrix} 11,3724 \\ -16,2198 \end{bmatrix} + \frac{(11,3724)^2 + (16,2198)^2}{200^2 + 140^2} \begin{bmatrix} -200 \\ -140 \end{bmatrix} \cong \begin{bmatrix} 10,0914 \\ -17,2326 \end{bmatrix};$$

$$t^{(1)} = -\frac{\langle f'(X^{(1)}), K^{(1)} \rangle}{\langle HK^{(1)}, K^{(1)} \rangle} \cong 0,1237;$$

$$X^{(2)} \cong \begin{bmatrix} 0,0010 \\ 0,0000 \end{bmatrix}; \quad \|f'(X^{(2)})\| \cong 0,017.$$

Метод Ньютона

Шаг 1

$$K^{(0)} = -H^{-1}f'(X^{(0)}), \quad t^{(0)} \equiv 1, \quad H^{-1} \cong \begin{bmatrix} -0,0694 & 0,0278 \\ 0,0278 & -0,1111 \end{bmatrix},$$

$$\begin{aligned} X^{(1)} &= X^{(0)} - H^{-1}f'(X^{(0)}) \cong \\ &\begin{bmatrix} 10 \\ 10 \end{bmatrix} - \begin{bmatrix} -0,0694 & 0,0278 \\ 0,0278 & -0,1111 \end{bmatrix} \begin{bmatrix} -200 \\ -140 \end{bmatrix} \cong \begin{bmatrix} 0,0000 \\ 0,0000 \end{bmatrix}, \\ f'(X^{(1)}) &= O. \end{aligned}$$

Метод Бройдена

Шаг 1. $K^{(0)} = -\eta^{(0)}f'(X^{(0)}), \quad \eta^{(0)} = -E, \quad K^{(0)} = f'(X^{(0)}),$

поэтому первый шаг полностью совпадает с первым шагом в методе наискорейшего подъема-спуска.

Шаг 2

$$\Delta\eta^{(0)} = \frac{(\Delta X^{(0)} + \Delta g^{(0)})(\Delta X^{(0)} + \Delta g^{(0)})^T}{(\Delta X^{(0)} + \Delta g^{(0)})^T \Delta g^{(0)}},$$

$$\text{где } \Delta X^{(0)} \cong \begin{bmatrix} -1,2403 \\ 2,1181 \end{bmatrix} - \begin{bmatrix} 10 \\ 10 \end{bmatrix} = \begin{bmatrix} -11,2403 \\ -7,8819 \end{bmatrix};$$

$$\Delta g^{(0)} \cong \begin{bmatrix} 11,3724 \\ -16,2198 \end{bmatrix} - \begin{bmatrix} -200 \\ -140 \end{bmatrix} = \begin{bmatrix} 211,3724 \\ 123,7802 \end{bmatrix};$$

$$\Delta X^{(0)} + \Delta g^{(0)} \cong (200,1321; 115,8983)^T;$$

$$\Delta \eta^{(0)} \cong \frac{\begin{bmatrix} 200,1321 \\ 115,8983 \end{bmatrix} [200,1321 \ 115,8983]}{[200,1321 \ 115,8983] \begin{bmatrix} 211,3724 \\ 123,7802 \end{bmatrix}} \cong \begin{bmatrix} 0,7070 & 0,4095 \\ 0,4095 & 0,2371 \end{bmatrix};$$

$$K^{(1)} = \eta^{(1)} f'(X^{(1)}) = \begin{bmatrix} -0,2930 & 0,4095 \\ 0,4095 & -0,7629 \end{bmatrix} \begin{bmatrix} 11,3724 \\ -16,2198 \end{bmatrix} = \begin{bmatrix} 9,9729 \\ -17,0303 \end{bmatrix}$$

$$t^{(1)} = - \frac{\langle f'(X^{(1)}), K^{(1)} \rangle}{\langle HK^{(1)}, K^{(1)} \rangle} \cong 0,1243;$$

$$X^{(2)} = X^{(1)} + t^{(1)} K^{(1)} \cong \begin{bmatrix} 0,0000 \\ 0,0000 \end{bmatrix}, \quad \|f'(X^{(2)})\| = 0.$$

$$\eta^{(1)} = -E + \Delta \eta^{(0)} = \begin{bmatrix} -0,2930 & 0,4095 \\ 0,4095 & -0,7629 \end{bmatrix};$$

Метод ДФП

Шаг 1. Полностью совпадает с первым шагом в методе Бройдена.

Шаг 2. $\Delta \eta^{(0)} = A^{(0)} - B^{(0)}$;

$$A^{(0)} = \frac{(\Delta X^{(0)})(\Delta X^{(0)})^T}{(\Delta X^{(0)})^T (\Delta g^{(0)})} \cong \frac{\begin{bmatrix} -11,2403 \\ -7,8819 \end{bmatrix} [11,2403 \ -7,8819]}{[-11,2403 \ -7,8819] \begin{bmatrix} 211,3724 \\ 123,7802 \end{bmatrix}} \cong$$

$$\cong \begin{bmatrix} -0,0377 & -0,0264 \\ -0,0264 & -0,0185 \end{bmatrix};$$

$$\begin{aligned}
B^{(0)} &= - \frac{(\Delta g^{(0)})(\Delta g^{(0)})^T}{(\Delta g^{(0)})^T (\Delta g^{(0)})} \cong - \frac{\begin{bmatrix} 211,3724 \\ 123,7802 \end{bmatrix} \begin{bmatrix} 211,3724 & 123,7802 \end{bmatrix}}{\begin{bmatrix} 211,3724 & 123,7802 \end{bmatrix} \begin{bmatrix} 211,3724 \\ 123,7802 \end{bmatrix}} \cong \\
&\cong \begin{bmatrix} -0,7446 & -0,4361 \\ -0,4361 & -0,2554 \end{bmatrix}; \\
\Delta \eta^{(0)} &\cong \begin{bmatrix} 0,7069 & 0,4096 \\ 0,4096 & 0,2368 \end{bmatrix}; \\
\eta^{(1)} &= -E + \Delta \eta^{(0)} \cong \begin{bmatrix} -0,2931 & 0,4096 \\ 0,4096 & -0,7632 \end{bmatrix}; \\
K^{(1)} &= -\eta^{(1)} f'(X^{(1)}) \cong \begin{bmatrix} 9,9769 \\ -17,0370 \end{bmatrix}; \\
t^{(1)} &= -\frac{\langle f'(X^{(1)}), K^{(1)} \rangle}{\langle HK^{(1)}, K^{(1)} \rangle} \cong 0,1243.
\end{aligned}$$

В методах переменной метрики аппроксимирующая матрица $\eta^{(i)} \rightarrow H^{-1}$ и в точке X^* $\eta(X^*) = H^{-1}$. Убедимся в этом.

Определение $\eta(X^)$ в методе Бройдена*

Мы уже установили, что $X^* = (0,0)^T$, поэтому

$$\begin{aligned}
\Delta X^{(1)} &= X^* - X^{(1)} \cong \begin{bmatrix} 1,2403 \\ -2,1181 \end{bmatrix}; \\
\Delta g^{(1)} &= f'(X^*) - f'(X^{(1)}) = -f'(X^{(1)}) \cong [-11,9724 \ 16,2198]^T; \\
\Delta \eta^{(1)} &= \frac{[\Delta X^{(1)} - \eta^{(1)}(\Delta g^{(1)})] [\Delta X^{(1)} - \eta^{(1)}(\Delta g^{(1)})]^T}{[\Delta X^{(1)} - \eta^{(1)}(\Delta g^{(1)})]^T (\Delta g^{(1)})}; \\
\Delta X^{(1)} - \eta^{(1)}(\Delta g^{(1)}) &\cong \begin{bmatrix} 1,2403 \\ -2,1181 \end{bmatrix} - \begin{bmatrix} -0,2930 & 0,4095 \\ 0,4095 & -0,7629 \end{bmatrix} \begin{bmatrix} -11,9724 \\ 16,2198 \end{bmatrix} \cong \\
&\cong \begin{bmatrix} -8,7326 \\ 14,9122 \end{bmatrix};
\end{aligned}$$

$$\Delta\eta^{(1)} \cong \frac{\begin{bmatrix} -8,7326 \\ 14,9122 \end{bmatrix} \begin{bmatrix} -8,7326 & 14,9122 \end{bmatrix}}{\begin{bmatrix} -8,7326 & 14,9122 \end{bmatrix} \begin{bmatrix} -11,9724 \\ 16,2198 \end{bmatrix}} \cong \begin{bmatrix} 0,2235 & -0,3817 \\ -0,3817 & 0,6518 \end{bmatrix};$$

$$\begin{aligned} \eta(X^*) = \eta^{(2)} = \eta^{(1)} + \Delta\eta^{(1)} &\cong \begin{bmatrix} -0,2930 & 0,4095 \\ 0,4095 & -0,7629 \end{bmatrix} + \\ &+ \begin{bmatrix} 0,2235 & -0,3817 \\ -0,3817 & 0,6518 \end{bmatrix} \cong \begin{bmatrix} -0,0695 & 0,0278 \\ 0,0278 & -0,1111 \end{bmatrix} \cong H^{-1}. \end{aligned}$$

Определение $\eta(X^)$ в методе ДФП*

$$\eta(X^*) = \eta^{(2)} = \eta^{(1)} + \Delta\eta^{(1)};$$

$$\Delta\eta^{(1)} = A^{(1)} - B^{(1)};$$

$$A^{(1)} = \frac{(\Delta X^{(1)})(\Delta X^{(1)})^T}{(\Delta X^{(1)})^T(\Delta g^{(1)})} \cong \frac{\begin{bmatrix} 1,2403 \\ -2,1181 \end{bmatrix} \begin{bmatrix} 1,2403 & -2,1181 \end{bmatrix}}{\begin{bmatrix} 1,2403 & -2,1181 \end{bmatrix} \begin{bmatrix} -11,9724 \\ 16,2198 \end{bmatrix}} \cong$$

$$\cong \begin{bmatrix} -0,0317 & 0,0542 \\ 0,0542 & -0,0926 \end{bmatrix};$$

$$\begin{aligned} B^{(1)} &= \frac{\eta^{(1)}(\Delta g^{(1)})(\Delta g^{(1)})^T(\eta^{(1)})^T}{(\Delta g^{(1)})^T(\eta^{(1)})^T(\Delta g^{(1)})} \cong \\ &\cong \frac{\begin{bmatrix} -0,2931 & 0,4096 \\ 0,4096 & -0,7632 \end{bmatrix} \begin{bmatrix} -11,9724 \\ 16,2198 \end{bmatrix} \begin{bmatrix} -11,9724 \\ 16,2198 \end{bmatrix}^T \begin{bmatrix} -0,2931 & 0,4096 \\ 0,4096 & -0,7632 \end{bmatrix}^T}{\begin{bmatrix} -11,9724 & 16,2198 \end{bmatrix} \begin{bmatrix} -0,2931 & 0,4096 \\ 0,4096 & -0,7632 \end{bmatrix} \begin{bmatrix} -11,9724 \\ 16,2198 \end{bmatrix}} \cong \end{aligned}$$

$$\cong \begin{bmatrix} -0,2553 & 0,4359 \\ 0,4359 & -0,7443 \end{bmatrix};$$

$$\eta(X^*) = \eta^{(1)} + A^{(1)} - B^{(1)} \cong \begin{bmatrix} -0,0694 & 0,0278 \\ 0,0278 & -0,1111 \end{bmatrix} \cong H^{-1}.$$

В рассмотренном примере осуществлялась оптимизация квадратичной функции, поэтому методы переменной метрики привели к системе сопряженных направлений, полностью совпадающих (с учетом некоторой разницы в масштабах и внесенных в расчеты погрешностей вычислений) с направлениями, полученными при реализации метода сопряженных градиентов.

4. ЗАДАЧИ НЕЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ ПРИ НАЛИЧИИ ОГРАНИЧЕНИЙ

4.1. ОСНОВНЫЕ ПОЛОЖЕНИЯ

Постановка задачи условной оптимизации

Формулировка общей задачи. Общая задача условной оптимизации формулируется как задача определения вектора $X^* = (x_1^*, \dots, x_n^*)^T$, обеспечивающего максимум скалярной функции $f(X)$ векторного аргумента X , принадлежащего допустимому множеству D

$$\begin{aligned} & \max f(X), \\ & X \in D \subset \mathbf{R}^n. \end{aligned} \tag{4.1}$$

Основная задача математического программирования формулируется следующим образом:

$$\begin{aligned} & \max f(X), \\ & g_j(X) \leq 0, \quad j = \overline{1, J}, \\ & X \in \mathbf{R}^n \end{aligned} \tag{4.2}$$

и является частным случаем задачи (4.1).

Понятие активных ограничений

Ограничение $g_j(X) \leq 0$ называется активным в точке \bar{X} , если $g_j(\bar{X}) = 0$, и неактивным, если $g_j(\bar{X}) < 0$.

Понятие конуса направлений

Непустое множество K_0 векторов, принадлежащих \mathbf{R}^n , называется конусом с вершиной в точке \bar{X} , если из того, что $(\bar{X} + K) \in K_0$ следует, что $(\bar{X} + tK) \in K_0$ для всех $t \geq 0$.

Конусы возможных и эффективных направлений

Конус возможных направлений. Пусть D — непустое множество в \mathbf{R}^n , а точка $\bar{X} \in D$. Конусом возможных направлений в точке \bar{X} называ-

ется такое множество $K_\epsilon(\bar{X})$ векторов K , что для каждого из них находится $\delta > 0$, обеспечивающее выполнение условия:

$$(\bar{X} + tK) \in D, \quad t \in (0, \delta), \quad K \neq 0. \quad (4.3)$$

Понятие конуса возможных направлений применительно к основной задаче математического программирования (4.2)

Пусть точка

$$X \in D = \left\{ X : g_j(X) \leq 0, \quad j = \overline{1, J} \right\},$$

а множество $I = \left\{ i : g_i(\bar{X}) = 0 \right\}$ — есть множество активных ограничений задачи (4.2) в точке \bar{X} . Тогда условию (4.3), определяющему конус возможных направлений $K_\epsilon(\bar{X})$, эквивалентно условие:

$$\langle g'_i(\bar{X}), K \rangle < 0 \quad (4.4)$$

для всех $i \in I$.

Обоснование. Так как функции $g_i(X)$, $i \in I$ дифференцируемы в точке \bar{X} , воспользуемся линейной аппроксимацией

$$g_i(X) \approx g_i(\bar{X}) + \langle g'_i(\bar{X}), \Delta X \rangle = g_i(\bar{X}) + t \langle g'_i(\bar{X}), K \rangle.$$

Выполнение условия $g_i(X) \leq 0$ возможно только при выполнении условия $\langle g'_i(\bar{X}), K \rangle < 0$, так как $g_i(\bar{X}) = 0$, а $t > 0$.

Конус эффективных направлений. Конусом эффективных направлений с вершиной в точке \bar{X} называется такое множество $K_0(\bar{X})$ векторов K , что для каждого из них найдется $\delta > 0$, обеспечивающее выполнение неравенства:

$$f(\bar{X} + tK) > f(\bar{X}) \quad (4.5)$$

для всех $t \in (0, \delta)$.

Понятие конуса эффективных направлений применительно к задаче (4.2). Пусть $f(X)$ дифференцируема в допустимой точке \bar{X} , тогда условию (4.5) эквивалентно условие:

$$\langle f'(\bar{X}), K \rangle > 0. \quad (4.6)$$

Обоснование. Воспользуемся линейной аппроксимацией $f(X)$ в

точке \bar{X} , тогда

$$f(\bar{X} + tK) - f(\bar{X}) \approx t\langle f'(\bar{X}), K \rangle,$$

поэтому условие (4.5) выполняется только в том случае, если $\langle f'(\bar{X}), K \rangle > 0$, так как $t > 0$.

Необходимые условия оптимальности для общей задачи условной оптимизации

Теорема 1

Пусть X^* — точка локального максимума задачи (4.1), тогда выполняется условие

$$K_e(X^*) \cap K_s(X^*) = \emptyset, \quad (4.7)$$

где $K_e(X^*), K_s(X^*)$ — конусы возможных и эффективных направлений с вершиной в точке X^* для задачи (4.1).

Рис. 4.1, поясняет необходимость выполнения условия (4.7) в оптимальной точке X^* .

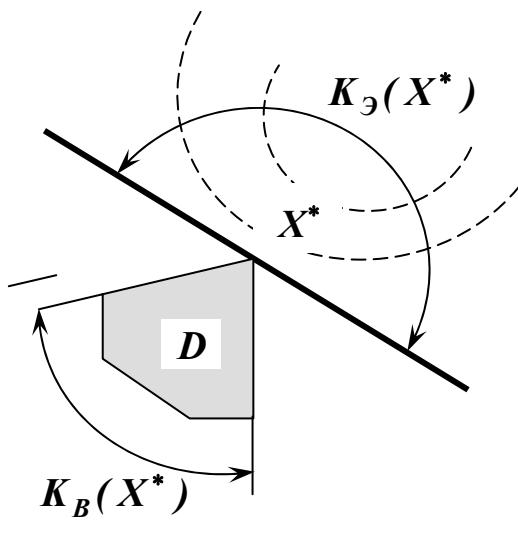


Рис. 4.1

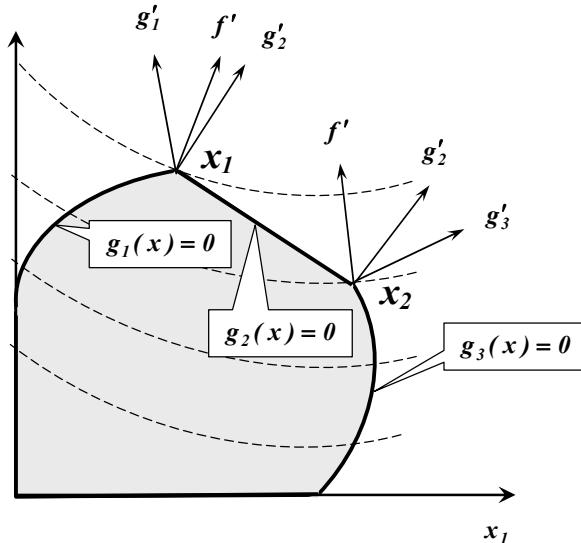


Рис. 4.2

Теорема 2 (Теорема Куна-Таккера)

Она формулирует те же необходимые условия оптимальности (условия Куна-Таккера) для основной задачи математического программирования (4.2), эквивалентные условию (4.7), но выраженные в более конструктивной форме.

Пусть X^* — произвольная допустимая точка задачи (4.2), $I = \{ i : g_i(X^*) = 0 \}$ — множество активных ограничений задачи (4.2).

Пусть функции $f(X)$, $g_i(X)$ дифференцируемы в точке X^* . Пусть также векторы градиентов $g'_i(X^*)$, $i \in I$ линейно независимы.

Тогда, если X^* — точка локального максимума задачи (4.2), то существуют такие числа u_i , $i \in I$, что

$$\begin{aligned} f'(X^*) + \sum_{i \in I} u_i g'_i(X^*) &= O, \\ u_i &\leq 0, \quad i \in I. \end{aligned} \tag{4.8}$$

Примечание. Условия Куна-Таккера в форме (4.8) учитывают только активные ограничения. Эквивалентная форма условий с учетом всех ограничений получается, если положить $u_i = 0$, $i \notin I$:

$$\begin{aligned} f'(X^*) + \sum_{j=1}^J u_j g'_j(X^*) &= O, \\ u_j g_j(X^*) &= 0, \quad j = \overline{1, J}; \\ u_j &\leq 0, \quad j = \overline{1, J}. \end{aligned} \tag{4.9}$$

Геометрическая интерпретация условий Куна-Таккера

Из условий Куна-Таккера следует:

$$f'(X^*) = \sum_{i \in I} (-u_i) g'_i(X^*), \quad u_i \leq 0,$$

что означает возможность представления градиента целевой функции в оптимальной точке X^* в форме линейной комбинации градиентов активных ограничений в той же точке. В этом случае вектор градиента целевой функции принадлежит конусу, натянутому на векторы градиентов активных ограничений. На рис. 4.2 показаны точки X_1 и X_2 , оптимальной из которых является первая, так как только $f'(X_1)$ принадлежит конусу, натянутому на $g'_1(X_1)$ и $g'_2(X_1)$.

Варианты условий оптимальности Куна-Таккера

Для различных задач математического программирования они приведены в табл. 4.1.

Таблица 4.1

Формулировка задачи		Условия Куна -Таккера
Тип экстремума	Ограничения	
$\max f(X)$	$g_j(X) \leq 0, j = \overline{1, J}$	$f'(X^*) + \sum_{j=1}^J u_j g'_j(X^*) = 0$ $u_j g_j(X^*) = 0, u_j \leq 0, j = \overline{1, J}$
$\max f(X)$	$g_j(X) \leq 0, j = \overline{1, J}$ $h_k(X) = 0, k = \overline{1, K}$	$f'(X^*) + \sum_{k=1}^K v_k h'_k(X^*) + \sum_{j=1}^J u_j g'_j(X^*) = 0$ $u_j g_j(X^*) = 0, u_j \leq 0, j = \overline{1, J}$
$\max f(X)$	$g_j(X) \leq 0, j = \overline{1, J_1}$ $g_j(X) \geq 0, j = \overline{J_1+1, J}$ $h_k(X) = 0, k = \overline{1, K}$	$f'(X^*) + \sum_{k=1}^K v_k h'_k(X^*) + \sum_{j=1}^J u_j g'_j(X^*) = 0$ $u_j g_j(X^*) = 0, j = \overline{1, J}; u_j \leq 0, j = \overline{1, J_1};$ $u_j \geq 0, j = \overline{J_1+1, J}$
$\min f(X)$	$g_j(X) \leq 0, j = \overline{1, J_1}$ $g_j(X) \geq 0, j = \overline{J_1+1, J}$ $h_k(X) = 0, k = \overline{1, K}$	$f'(X^*) + \sum_{k=1}^K v_k h'_k(X^*) + \sum_{j=1}^J u_j g'_j(X^*) = 0$ $u_j g_j(X^*) = 0, j = \overline{1, J}; u_j \geq 0, j = \overline{1, J_1};$ $u_j \leq 0, j = \overline{J_1+1, J}$
$\max f(X)$	$g_j(X) \leq 0, j = \overline{1, J}$ $h_k(X) = 0, k = \overline{1, K}$ $X \geq 0$	$f'(X^*) + \sum_{k=1}^K v_k h'_k(X^*) + \sum_{j=1}^J u_j g'_j(X^*) \leq 0$ $\left\langle \left[f'(X^*) + \sum_{k=1}^K v_k h'_k(X^*) + \sum_{j=1}^J u_j g'_j(X^*) \right], X^* \right\rangle = 0$ $u_j g_j(X^*) = 0, u_j \leq 0, j = \overline{1, J}$

Условия оптимальности для недифференцируемых функций

Рассматриваются условия оптимальности, применимые для более общего класса задач оптимизации, допускающих недифференцируемость целевой функции и функций ограничений. В этих условиях используются понятия о седловой точке функции и функции Лагранжа.

Понятие о седловой точке функции

Функция $f(X, Y)$ имеет седловую точку (X^*, Y^*) , если

$$f(X, Y^*) \leq f(X^*, Y^*) \leq f(X^*, Y). \quad (4.10)$$

Это определение подразумевает, что X^* максимизирует $f(X, Y^*)$ по всем X , а Y^* минимизирует $f(X^*, Y)$ по всем Y :

$$f(X^*, Y^*) = \max_X \min_Y f(X, Y) \quad (4.11)$$

Понятие о функции Лагранжа

Функция Лагранжа есть линейная комбинация целевой функции и функции ограничений. В зависимости от формулировки задачи математического программирования функция Лагранжа приобретает соответствующий вид. Пусть задача приведена в форме:

$$\max f(x);$$

$$g_j \leq 0, \quad j = \overline{1, J}; \quad (4.12)$$

$$h_k = 0, \quad k = \overline{1, K};$$

тогда функцию

$$L(X, \Lambda) = L(X, V, U) = f(X) + \sum_{k=1}^K v_k h_k(X) + \sum_{j=1}^J u_j g_j(X) \quad (4.13)$$

называют функцией Лагранжа, а числа v_k , u_j , составляющие вектора Λ , – множителями Лагранжа.

Примечание. Условие Куна-Таккера можно рассматривать как определенные требования, предъявляемые к производной $L'_x(X, \Lambda)$ функции Лагранжа в оптимальной точке.

Необходимые и достаточные условия оптимальности

Теорема 3. Пусть задача математического программирования приведена к форме (4.12), целевая функция вогнута, функции ограничений вы-

пуклы в допустимой области \mathbf{D} , существует точка $\bar{\mathbf{X}} \in \mathbf{D}$, в которой $\mathbf{g}_j(\bar{\mathbf{X}}) < 0$ для всех $j = \overline{1, J}$. Тогда необходимым и достаточным условием существования оптимального решения \mathbf{X}^* задачи (4.12) является существование такого вектора $\Lambda^* = (\mathbf{V}^*, \mathbf{U}^*)$, $\mathbf{U} \leq \mathbf{0}$, что вектор

$$(\mathbf{X}^*, \Lambda^*) = (\mathbf{X}^*, \mathbf{V}^*, \mathbf{U}^*)$$

является седловой точкой функции Лагранжа (4.13):

$$L(\mathbf{X}, \Lambda^*) \leq L(\mathbf{X}^*, \Lambda^*) \leq L(\mathbf{X}^*, \Lambda), \quad (4.14)$$

где $\mathbf{X} \in \mathbf{D} = \left\{ \mathbf{X} : \mathbf{g}_j(\mathbf{X}) \leq \mathbf{0}, j = \overline{1, J}; \mathbf{h}_k = \mathbf{0}, k = \overline{1, K} \right\}, \mathbf{U} \leq \mathbf{0}$.

Примечание. Задача поиска седловой точки функции Лагранжа называется задачей Куна-Таккера. Условия Куна-Таккера определяют решение задачи Куна-Таккера в частном случае дифференцируемости целевой функции и функций ограничений.

4.2. КЛАССИФИКАЦИЯ МЕТОДОВ УСЛОВНОЙ ОПТИМИЗАЦИИ

Большое многообразие методов условной оптимизации требует упорядоченного подхода к их рассмотрению. С этой целью проводится краткая классификация этих методов. Основу любой классификации составляет вектор классификационных признаков, определяющих состав классификационных групп. В данном случае в качестве таких признаков используются характеристики идеи, положенной в основу объединенных в одну группу методов. Выделенные по этому признаку следующие группы методов будут рассмотрены в остальных разделах данной главы.

Методы, основанные на использовании необходимых условий оптимизации

Существует значительная группа методов, в которых в той или иной форме используются необходимые условия оптимальности (условия Куна-Таккера).

В ряде методов условия оптимальности входят в алгоритмическую схему метода в качестве условий останова (метод Била, метод проекции градиента).

Другая часть методов из этой группы (методы Лагранжа, Баракина-Дорфмана, Франка-Вульфа) предполагают замену исходной задачи эквивалентной ей задачей Куна-Таккера поиска седловой точки функции Лагранжа. В этом случае целевой функцией новой задачи становится функция Лагранжа, а условия Куна-Таккера могут рассматриваться как дополнительные

ограничения, накладываемые на допустимую область задачи.

Методы, основанные на обобщении симплексного метода применительно к классу задач математического программирования

Упомянутые выше методы Била, Баранкина-Дорфмана, Франка-Вульфа предназначены для решения задач с квадратичными целевыми функциями и линейными ограничениями и объединены общей идеей, состоящей в линеаризации целевой функции в граничной точке допустимой области и поиске оптимума с использованием модификации симплекс-метода

Методы проекции направлений

Основная идея проективных методов состоит в том, что направление поиска определяется, как проекция эффективного направления, например, градиентного, на линейные или линеаризованные ограничения, образующие линейное многообразие. Все точки траектории поиска принадлежат допустимой области. Поиск оптимального решения заканчивается при выполнении условий Куна-Таккера.

Методы, основанные на выборе эффективного направления в конусе возможных направлений

Основная идея этих методов, в частности, метода возможных направлений Зойтендейка, состоит в том, чтобы на каждой итерации найти эффективное направление, принадлежащее конусу возможных направлений. Выбор эффективного направления осуществляется путем решения вспомогательной задачи, поставленной в форме задачи линейного программирования. Модификации методов могут различаться видом вспомогательной задачи [6].

Методы штрафных и барьерных функций

Поскольку методы безусловной оптимизации значительно проще, легче оптимизируются и более полно разработаны, чем методы условной оптимизации, предложен подход, состоящий в преобразовании исходной задачи условной оптимизации в эквивалентную последовательность задач безусловной оптимизации на основе специально сконструированных барьерных или штрафных функций.

4.3. ИСПОЛЬЗОВАНИЕ НЕОБХОДИМЫХ УСЛОВИЙ

ОПТИМАЛЬНОСТИ

Метод неопределенных множителей Лагранжа

Область применения метода Лагранжа – задачи математического программирования с ограничениями, заданными в форме равенств.

Пусть исходная задача имеет следующий вид:

$$\begin{aligned} & \max f(X), \\ & h_k(X) = 0, \quad k = \overline{1, K}, \end{aligned} \tag{4.15}$$

тогда $L(X, V) = f(X) + \sum_{k=1}^K v_k h_k(X) -$

функция Лагранжа для этой задачи.

Условия Куна-Таккера в данном случае запишем в следующей форме (см. табл. 4.1):

$$f'(X^*) + \sum_{k=1}^K v_k h'_k(X^*) = \mathbf{0},$$

и вместе с системой равенств, определяющих допустимую область задачи (4.15), получим условия

$$\begin{cases} \frac{\partial L}{\partial x_j} = \frac{\partial f}{\partial x_j} + \sum_{k=1}^K v_k \frac{\partial h_k}{\partial x_j} = 0, & j = \overline{1, n}; \\ \frac{\partial L}{\partial v_k} = h_k = 0, & k = \overline{1, K}, \end{cases} \tag{4.16}$$

существования стационарной точки (X^*, V^*) функции Лагранжа.

Таким образом, исходная задача (4.15) условной оптимизации может быть сведена к задаче безусловной оптимизации – задаче поиска стационарной точки функции Лагранжа, являющейся точкой локального максимума функции $L(X, V)$ по аргументу X .

Введем в рассмотрение матрицу $H_L(X, V)$ вторых производных функций $L(X, V)$ по X :

$$H_L(X, V) = \left\{ \frac{\partial^2 L(X, V)}{\partial x_i \partial x_j} \right\}_{i,j=\overline{1,n}}.$$

Для того чтобы стационарная точка (X^*, V^*) функции Лагранжа

являлась точкой локального максимума, должно выполняться достаточное условие максимума второго порядка, состоящее в отрицательной определенности матрицы $H_L(X, V)$.

Пример 4.1

Найти $\max \left\{ -x_1^2 - x_2^2 \right\}$

при ограничении $h(X) = 2x_1 + x_2 - 2 = 0$.

Запишем функцию Лагранжа:

$$L(X, V) = -x_1^2 - x_2^2 + V_1(2x_1 + x_2 - 2).$$

Сформулируем условие стационарности $L(X, V)$ в точке (X^*, V^*) :

$$\begin{cases} \frac{\partial L}{\partial x_1} = -2x_1 + 2V_1 = 0, \\ \frac{\partial L}{\partial x_2} = -2x_2 + V_1 = 0, \\ \frac{\partial L}{\partial V_1} = 2x_1 + x_2 - 2 = 0. \end{cases}$$

Решая систему уравнений, получим

$$(X^*, V^*) = (x_1^*, x_2^*, V_1^*)^T = \left(\frac{4}{5}, \frac{2}{5}, \frac{4}{5}\right)^T.$$

Определим матрицу Гессе $H_L(X, V)$:

$$H_L(X, V) = \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix}$$

и убедимся в ее отрицательной определенности: матрица $-H_L$ положительно определена. Таким образом, в соответствии с условиями второго порядка точка (X^*, V^*) является точкой максимума $L(X, V)$ по X , а точка X^* — решением задачи условной оптимизации.

Сведение нелинейных задач к задачам линейного программирования

Рассматриваемые ниже методы предназначены для решения задач квадратичного программирования — задач с квадратичной целевой функцией и линейными ограничениями и объединены в общую группу по

ряду признаков:

- а) принадлежность единой области применения;
- б) использование (в той или иной форме) необходимых условий оптимальности Куна-Таккера при построении алгоритма оптимизации;
- в) использование симплексного метода в каждой итерации пошагового оптимизационного процесса.

Метод Била

Пусть исходная форма задачи оптимизации следующая:

$$\max \left\{ f(X) = \langle p, X \rangle + \frac{1}{2} \langle HX, X \rangle + C \right\},$$

где

$$AX = b, \quad (4.17)$$

$$X \geq O,$$

$$X = (x_1, x_2, \dots, x_n)^T,$$

$$b = (b_1, \dots, b_m)^T,$$

$$A = \{a_{ij}\}, \quad i = \overline{1, m}; \quad j = \overline{1, n}.$$

Начальный этап метода Била полностью совпадает с начальным этапом линейного симплекс-метода, рассмотренного в главе 2.

Разделим все переменные на две группы: группу m базисных переменных $\{x_i\}$, $i \in \mathcal{B}$, и группу остальных $k = (n - m)$ небазисных (свободных) переменных $\{x_j\}$, $j \notin \mathcal{B}$. Выразим базисные переменные через свободные:

$$x_i = b'_i + \sum_{j=1}^k a'_{ij} x_j, \quad i = \overline{1, m}; \quad i \in \mathcal{B}. \quad (4.18)$$

Условием допустимости базисного решения $\{x_i\}$, $i \in \mathcal{B}$ является неотрицательность его компонентов ($b'_i \geq 0$, $i \in \mathcal{B}$) в опорной точке (в которой все свободные переменные $\{x_j\}$, $j \notin \mathcal{B}$ принимают граничные нулевые значения). Опустим в дальнейшем штрихи в обозначениях b'_i , a'_{ij} .

Предположим, что начальное допустимое базисное решение задачи (4.17) найдено (например, одним из методов, рассмотренных в главе 2).

Представим целевую функцию в виде линейной аппроксимации в точке $X^{(0)}$ как функцию только свободных переменных:

$$f(X) \approx f(X^{(0)}) + \sum_{j \notin B_0} \frac{\partial f}{\partial x_j}(X^{(0)})x_j. \quad (4.19)$$

Сформулируем признак оптимальности опорной точки X^* для задачи (4.17):

$$\frac{\partial f}{\partial x_j}(X^*) \leq 0, \quad j \notin B_*, \quad (4.20)$$

где B_* — обозначение оптимального базиса.

Действительно, если в точке X^* выполняется условие (4.20), то любое допустимое перемещение из точки X^* (приводящее к положительным приращениям свободных переменных) будет способствовать уменьшению целевой функции.

Покажем, что условия (4.20) следуют из условий Куна-Таккера для задачи (4.17).

Используя (4.18), всегда можем выразить $f(X)$ только через свободные переменные, а задачу (4.17) сформулировать следующим образом:

$$\begin{aligned} & \max f(X), \\ & g_j(X) = -x_j \leq 0, \quad j \notin B_*. \end{aligned} \quad (4.21)$$

Условия Куна-Таккера для задачи (4.21) получим, интерпретировав соответствующим образом условия (4.9):

условию $f'(X^*) + \sum u_j g'_j(X^*) = \mathbf{0}$ соответствует условие

$$\frac{\partial f}{\partial x_j}(X^*) - u_j = 0, \quad j \notin B_*,$$

$$\text{откуда } \frac{\partial f}{\partial x_j}(X^*) = u_j, \quad j \notin B_*. \quad$$

Условию $u_j g_j(X^*) = 0, \quad j = \overline{1, J}$ соответствует условие

$$-u_j x_j^* = -x_j^* \frac{\partial f}{\partial x_j}(X^*) = 0, \quad j \notin B_*. \quad$$

Условию $u_j \leq 0, \quad j = \overline{1, J}$ соответствует условие

$$u_j = \frac{\partial f}{\partial x_j}(X^*) \leq 0, \quad j \notin B_*. \quad$$

Анализ этих условий позволяет убедиться в справедливости

принятого признака (4.20) оптимальности опорной точки X^* .

Пусть в начальной опорной точке $X^{(0)}$ некоторые из производных

$$\frac{\partial f}{\partial x_j}(X^{(0)}) > 0, \quad j \notin B_0.$$

Тогда целесообразно увеличивать одну из таких переменных x_r , оставляя другие свободные переменные неизменными и равными нулю и обеспечивая тем самым увеличение целевой функции. Если бы целевая функция была линейной, то есть задача оптимизации относилась к классу задач линейного программирования, увеличение свободной переменной x_r сопровождалось бы таким изменением базисных переменных, что одна из них (x_s) ранее других обратилась бы в нуль. Это означало бы переход к следующей опорной точке и замену x_s на x_r в наборе базисных переменных. В случае нелинейной функции такая ситуация может иметь место, но не является единственно возможной. Производная $\frac{\partial f}{\partial x_r}(X)$

квадратичной функции может обратиться в нуль раньше, чем любая из базисных переменных.

Рассмотрим последовательно каждую из этих ситуаций.

1. Базисная переменная x_s обратится в нуль раньше, чем $\frac{\partial f}{\partial x_r}(X)$.

В этом случае переменная $x_r, r \notin B_0$ включается в новый базис вместо переменной $x_s, s \in B_0$.

В соответствии с (4.18) базисные переменные можно выразить через свободные, а целевую функцию представить в виде (4.19) как функцию только свободных переменных $x_j, j \notin B_1$.

В новой опорной точке $X^{(1)}$ необходимо проверить выполнение признака оптимальности и в случае продолжения поиска произвести выбор свободной переменной, соответствующей максимуму частной производной целевой функции по свободной переменной:

$$x_r \rightarrow \max_j \left\{ \frac{\partial f}{\partial x_j}(X^{(1)}), j \in B_1 \right\} = \frac{\partial f}{\partial x_r}(X^{(1)}).$$

2. Предположим, что при перемещении из $X^{(0)}$ (в связи с

изменением свободной переменной x_r) производная $\frac{\partial f}{\partial x_r}(X)$ обращается в нуль раньше, чем любая из базисных переменных. Это означает, что по переменной x_s мы не выходим за границу допустимой области.

В этом случае метод Била предполагает введение в задачу новой переменной $u_1 = \frac{1}{2} \cdot \frac{\partial f}{\partial x_r}(X)$, выполняющей вместо x_r роль новой свободной переменной. Переменная x_r , свободная переменная прежнего базиса B_0 , становится теперь дополнительной базисной переменной базиса B_1 . Отметим, что новая переменная u_1 , в отличие от переменных x , не ограничена по знаку.

В опорной точке $X^{(1)}$ все свободные переменные, в том числе и u_1 , принимают нулевые значения.

Признаком оптимальности опорной точки $X^{(1)}$ является условие

$$\frac{\partial f}{\partial x_j}(X^{(1)}) \leq 0, \quad j \notin B_1; \quad \frac{\partial f}{\partial u_1}(X^{(1)}) = 0.$$

Признак допустимости — неотрицательность компонентов базисного решения — остается без изменений. Дополнительное условие $\frac{\partial f}{\partial u_1}(X^{(1)}) = 0$ оптимальности $X^{(1)}$ есть условие Куна-Таккера для неограниченной по знаку переменной u_1 .

При переходе к очередной опорной точке следует в первую очередь стремиться к увеличению $f(X)$ за счет изменения не ограниченных по знаку переменных u_j (знак изменения u_j выбирают в зависимости от

знака $\frac{\partial f}{\partial u_j}(X^{(i)})$), и только в том случае, если все

$\frac{\partial f}{\partial u_j}(X^{(i)}) = 0, \quad j \notin B_i$, осуществляется выбор одной из свободных

переменных $x_j, j \notin B_i$, включаемой в новый базис B_{i+1} .

Рассмотрим метод Била в приложении к конкретной задаче квадратичного программирования.

Пример 4.2

Найти $\max f(X) = 10x_1 - x_1^2 + 2x_1x_2 - 2x_2^2$

при ограничениях

$$\begin{cases} x_1 + 2x_2 \leq 10; \\ x_1 + x_2 \leq 6; \\ x_1, x_2 \geq 0. \end{cases}$$

Введем дополнительные переменные x_3, x_4 и запишем систему ограничений в форме равенств

$$\begin{cases} x_1 + 2x_2 + x_3 = 10; \\ x_1 + x_2 + x_4 = 6; \\ x_1, x_2, x_3, x_4 \geq 0. \end{cases}$$

Дальнейшую процедуру решения задачи развернем в виде последовательности шагов.

Шаг 1. Формирование начального базиса B_0

Пусть $B_0 = (3,4)$, тогда $X^{(0)} = (0,0,10,6)^T$ является допустимым базисным решением. Для опорной точки $X^{(0)}$ заполним симплекс-таблицу 4.2, составленную по обычной форме (см. табл. 2.1) с некоторым изменением содержания последней строки, куда запишем формулы для частных производных $\frac{\partial f}{\partial x_j}(X), j \notin B_0$, выраженных только через свободные переменные $x_j, j \notin B_0$.

Шаг 2. Анализ симплекс-таблицы 4.2

Точка $X^{(0)}$ принадлежит допустимой области $(x_3^{B_0} = 10 > 0, x_4^{B_0} = 6 > 0)$, но не является оптимальной:

$$\frac{\partial f}{\partial x_1}(X^{(0)}) = 10 > 0, \quad \frac{\partial f}{\partial x_2}(X^{(0)}) = 0.$$

Шаг 3. Выбор разрешающего столбца

Необходимо найти свободную переменную x_r , соответствующую максимальной производной целевой функции:

$$\frac{\partial f}{\partial x_r}(X^{(0)}) = \max \left\{ \frac{\partial f}{\partial x_1}(X^{(0)}), \frac{\partial f}{\partial x_2}(X^{(0)}) \right\}.$$

Разрешающий столбец — x_1

Таблица 4.2

	x_1	x_2	b
x_3	-1	-2	10
x_4	-1	-1	6
$\frac{\partial f}{\partial x_j}$	$2(5 - x_1 + x_2)$	$2(x_1 - 2x_2)$	

Таблица 4.3

	x_1	x_2	b
u_1	-1	1	5
x_3	-1	-2	10
x_4	-1	-1	6

Таблица 4.4

	u_1	x_2	b
x_1	-1	1	5
x_3	1	-3	5
x_4	1	-2	1
$\frac{\partial f}{\partial x_j}, \frac{\partial f}{\partial u_j}$	$-2u_1$	$2(5 - x_2)$	

Таблица 4.5

	u_1	x_4	b
x_1	$-1/2$	$-1/2$	$11/2$
x_3	$-1/2$	$3/2$	$7/2$
x_2	$1/2$	$-1/2$	$1/2$
$\frac{\partial f}{\partial x_j}, \frac{\partial f}{\partial u_j}$	$\frac{9}{2} - \frac{5}{2}u_1 + \frac{1}{2}x_4$	$-\frac{9}{2} + \frac{1}{2}u_1 - \frac{1}{2}x_4$	

Шаг 4. Оценка ситуации в опорной точке $X^{(0)}$

Найдем соотношение между приращением свободной переменной x_1 и изменениями базисных переменных x_3, x_4 и производной $\frac{\partial f}{\partial x_1}(X)$:

$$\frac{\partial f}{\partial x_1}(X) = 0 \quad \text{при } x_1 = 5;$$

$$x_3 = 0 \quad \text{при } x_1 = 10;$$

$$x_4 = 0 \quad \text{при } x_1 = 6.$$

Производная $\frac{\partial f}{\partial x_1}(X)$ обращается в нуль раньше базисных

переменных, поэтому введем в задачу новую свободную переменную $u_1 = \frac{1}{2} \cdot \frac{\partial f}{\partial x_1} = 5 - x_1 + x_2$, а переменную x_1 включим в новый базис B_1 .

Шаг 5. Формирование базиса $B_1: X^{B_1} = (x_1, x_2, x_3)^T$

Заполнение новой симплекс-таблицы начинаем с составления промежуточной табл. 4.3, отличающейся от табл. 4.2 дополнительной строкой для новой свободной переменной u_1 и отсутствием строки с данными о частных производных. Выделим в табл. 4.3 разрешающий элемент на пересечении столбца x_1 и строки u_1 . Произведем пересчет табл. 4.3 в соответствии с правилом пересчета симплекс-таблиц (гл.2, п. 2.4).

Выразим целевую функцию через новые свободные переменные:

$$f(u_1, x_2) = 25 - u_1^2 + 10x_2 - x_2^2 \quad \text{и дополним полученную}$$

таблицу строкой для частных производных $\frac{\partial f}{\partial u_1}(X), \frac{\partial f}{\partial x_2}(X)$,

также выраженных через свободные переменные u_1, x_2 (табл. 4.4.).

Шаг 6. Анализ симплекс-таблицы 4.4

Точка $X^{(1)}$ принадлежит допустимой области, но не оптимальна, так как

$$\frac{df}{dx_2}(X^{(1)}) = 2(5 - x_2) = 10 > 0, \quad \frac{df}{du_1}(X^{(1)}) = -2u_1 = 0.$$

Разрешающий столбец x_2 соответствует положительной производной $\frac{\partial f}{\partial x_2}(X^{(1)})$, а разрешающая строка x_4 выбирается с учетом требования принадлежности новой опорной точки $X^{(2)}$ допустимой области. Критерий выбора разрешающей строки тот же, что и в задачах линейного программирования.

Шаг 7. Формирование базиса B_2 : $X^{B_2} = (x_1, x_3, x_2)^T$

Произведем пересчет таблицы 4.4 (без последней строки), выразим целевую функцию через новые свободные переменные:

$$f(u_1, x_4) = \frac{119}{4} + \left(\frac{9}{2} + \frac{1}{2} \cdot x_4 \right) u_1 - \frac{5}{4} \cdot u_1^2 - \frac{9}{2} \cdot x_4 - \frac{1}{4} \cdot x_4^2$$

В табл. 4.5 для новой опорной точки выразим $\frac{\partial f}{\partial u_1}(X)$, $\frac{\partial f}{\partial x_4}(X)$ через свободные переменные u_1, x_4 .

Шаг 8. Анализ симплекс-таблицы 4.5

$$\text{Точка } X^{(2)} = (x_1, x_2, x_3)^T = \left(\frac{11}{2}, \frac{7}{2}, \frac{1}{2} \right)^T$$

принадлежит допустимой области. Признак оптимальности не выполняется:

$$\frac{df}{du_1}(X^{(2)}) = \frac{9}{2} \neq 0,$$

поэтому в качестве разрешающего столбца выбираем u_1 . Так как $\frac{\partial f}{\partial u_1}(X^{(1)}) > 0$, то при переходе к новой точке $X^{(3)}$ неограниченная по знаку переменная u_1 получает положительное приращение.

Шаг 9. Оценка ситуации в опорной точке $X^{(2)}$.

Оценим изменения базисных переменных и $\frac{\partial f}{\partial u_1}(X)$,

обусловленные приращением u_1 :

$$x_1 = 0 \quad \text{при } u_1 = 11;$$

$$x_2 = 0 \quad \text{при } u_1 = -1;$$

$$x_3 = 0 \quad \text{при } u_1 = 7;$$

$$\frac{\partial f}{\partial u_1}(X) = 0 \quad \text{при } u_1 = \frac{9}{5};$$

x_2 не обращается в нуль при $u_1 \geq 0$.

Производная $\frac{\partial f}{\partial u_1}(X)$ обнуляется раньше базисных

переменных, поэтому вводим в задачу новую свободную переменную

$$u_2 = \frac{1}{2} \frac{\partial f}{\partial u_1} = \frac{9}{4} - \frac{5}{4} u_1 + \frac{1}{4} x_4,$$

переводя u_1 в группу базисных переменных. Составим промежуточную табл. 4.6 с дополнительной строкой для u_2 и без строки для производных.

Шаг 10. Формирование базиса B_3

$$X^{B_3} = (x_1, x_3, x_2)^T.$$

Произведем пересчет табл. 4.6 и составим окончательную таблицу (табл. 4.7) для новой точки $(X, U)^{(3)}$.

Выразим целевую функцию через новые свободные переменные:

$$f(u_2, x_4) = \frac{169}{5} - \frac{4}{5} \cdot u_2^2 - \frac{18}{5} \cdot x_4 - \frac{1}{5} \cdot x_4^2.$$

Шаг 11. Анализ симплекс-таблицы 4.7

В точке $(X, U)^{(3)} = (u_1, x_1, x_3, x_2)^T = \left(0, \frac{23}{5}, \frac{13}{5}, \frac{7}{5}\right)^T$

выполняется признак допустимости: $X^{(3)} = (x_1, x_3, x_2)^T > 0$

и признак оптимальности:

$$\frac{\partial f}{\partial u_2}(X^{(3)}) = 0; \quad \frac{\partial f}{\partial x_4}(X^{(3)}) = -\frac{18}{15} < 0,$$

следовательно, решение найдено:

$$X_{opt} = X^{(3)}, \quad \max f(X) = \frac{169}{5}.$$

Таблица 4.6

	u_1	x_4	b
u_2	-5/4	1/4	9/4
x_1	-1/2	-1/2	11/2
x_3	-1/2	3/2	7/2
x_2	1/2	-1/2	1/2

Таблица 4.7

	u_2	x_4	b
u_1	-4/5	1/5	9/5
x_1	2/5	-3/5	23/5
x_3	2/5	7/5	13/5
x_2	-2/5	-2/5	7/5
$\frac{\partial f}{\partial x_j}, \frac{\partial f}{\partial u_j}$	$-\frac{8}{5}u_2$	$-\frac{18}{5} - \frac{2}{5}x_4$	

На рис. 4.3 приведена траектория поиска оптимального решения. В отличие от задач линейного программирования не все точки траектории являются опорными точками допустимой области. Еще одной особенностью метода Била является увеличение числа переменных задачи в ходе ее решения.

Метод Баранкина-Дорфмана

Этот метод предполагает замену исходной задачи эквивалентной ей задачей математического программирования, сформулированной в результате интерпретации условий Куна-Таккера.

Предположим, что исходная задача задана в следующей форме:

$$\begin{aligned} \max \left\{ f(X) = \langle p, X \rangle + \frac{1}{2} \langle HX, X \rangle + C \right\}, \\ AX \leq b, \\ X \geq 0. \end{aligned} \quad (4.22)$$

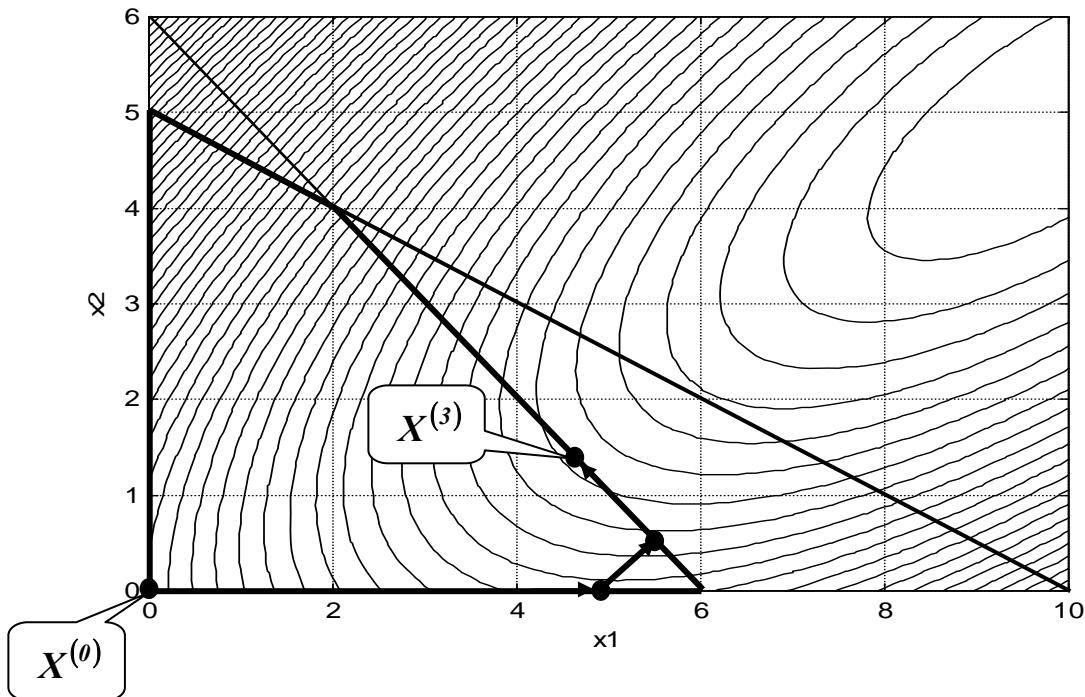


Рис. 4.3

Запишем условия Куна-Таккера, воспользовавшись их общей формулировкой, приведенной в последней строке табл. 4.1. Для исходной задачи (4.22) определим конкретный вид фрагментов выражений, составляющих условия Куна-Таккера:

$$\begin{aligned} f'(X) &= p + HX, \\ g_j(X) &= \sum_{l=1}^n a_{jl} x_l - b_j, \\ \sum_{j=1}^J u_j g'_j(X) &= \left\{ \sum_{j=1}^J a_{jl} u_j \right\}_{l=\overline{1,n}} = A^T U, \end{aligned}$$

где

$$U = (u_1, \dots, u_J)^T, \quad A = \{a_{jl}\}, \quad j = \overline{1, J}; \quad l = \overline{1, n}.$$

Условие $u_j g_j(X) = 0, \quad j = \overline{1, J}$ трансформируется в условие

$$(AX - b)^T U = \mathbf{0}.$$

С учетом приведенного анализа получим следующую формулировку условий Куна-Таккера для задачи (4.22):

$$p + HX^* + A^T U \leq \mathbf{0}; \quad (4.23)$$

$$(p + HX^* + A^T U)^T X^* = \mathbf{0}; \quad (4.24)$$

$$(AX^* - b)^T U = \mathbf{0}; \quad (4.25)$$

$$U \leq \mathbf{0}. \quad (4.26)$$

Введем обозначения:

$$U = -\Lambda; \quad p + HX + A^T U = p + HX - A^T \Lambda = -V; \quad -(AX - b) = Y,$$

тогда с учетом (4.23) – (4.26) для исходной задачи должны выполняться следующие соотношения:

$$\begin{cases} AX + Y = b; \\ HX - A^T \Lambda + V = -p; \\ X \geq \mathbf{0}, \quad V \geq \mathbf{0}; \\ Y \geq \mathbf{0}, \quad \Lambda \geq \mathbf{0}; \end{cases} \quad (4.27)$$

$$X^T V + Y^T \Lambda = \mathbf{0}. \quad (4.28)$$

Условие (4.28) объединяет условия (4.24) и (4.25).

Располагая системой (4.27) и уравнением (4.28), сформулируем следующую задачу, эквивалентную исходной:

найти два n -мерных вектора X, V и два m -мерных вектора Y, Λ , удовлетворяющих условиям (4.27) и минимизирующих квадратичную функцию $X^T V + Y^T \Lambda$

Система (4.27) определяет допустимую область задачи, а минимальное значение целевой функции $X^T V + Y^T \Lambda$ заранее известно и равно нулю в соответствии с условиями Куна-Таккера. Таким образом, оптимальное решение задачи удовлетворяет условиям (4.27) и обеспечивает нулевое значение целевой функции.

Если ввести в рассмотрение векторы

$$Z = (X, Y, V, \Lambda), \quad \tilde{Z} = (V, \Lambda, X, Y)$$

размерности $2(n+m)$, то целевая функция примет вид

$$X^T V + Y^T \Lambda = \frac{1}{2} Z^T \tilde{Z}, \quad (4.29)$$

а система ограничений (4.27)

$$\begin{bmatrix} A & E & O & O \\ H & O & E & -A^T \end{bmatrix} Z = \begin{bmatrix} b \\ -p \end{bmatrix}. \quad (4.30)$$

Для решения этой задачи может быть использован описанный выше алгоритм метода Била, определенным недостатком которого является увеличение числа переменных по ходу решения задачи. Если же строить симплексную процедуру как процедуру анализа только опорных точек допустимой области (4.30), то возможен останов алгоритма при выполнении формальных признаков оптимальности (отрицательности частных производных целевой функции по свободным переменным) в опорной точке, не являющейся точкой оптимума задачи [7]. Метод Франка-Вульфа, реализующий те же идеи, предполагает поиск оптимума целевой функции (4.29) не только на границах, но и внутри допустимой области и лишен отмеченного недостатка [7]. Рассмотрим пример приведения исходной задачи к форме (4.29) – (4.30).

Пример 4.3

Найти $\max \{f(X) = 10x_1 - x_1^2 + 2x_1x_2 - 2x_2^2\}$
 при ограничениях $\begin{cases} x_1 + 2x_2 \leq 10; \\ x_1 + x_2 \leq 6; \\ x_1, x_2 \geq 0. \end{cases}$

Представим целевую функцию в виде

$$f(X) = \langle p, X \rangle + \frac{1}{2} \langle HX, X \rangle + C,$$

где $p = (10, 0)^T$, $C = 0$, $H = \begin{bmatrix} -2 & 2 \\ 2 & -4 \end{bmatrix}$.

Введем векторы

$$Z = (x_1, x_2, y_1, y_2, v_1, v_2, \lambda_1, \lambda_2);$$

$$\tilde{Z} = (v_1, v_2, \lambda_1, \lambda_2, x_1, x_2, y_1, y_2)^T,$$

тогда минимизируемая целевая функция эквивалентной задачи в соответствии с (4.29) будет иметь вид:

$$\frac{1}{2} Z^T \tilde{Z} = x_1 v_1 + x_2 v_2 + y_1 \lambda_1 + y_2 \lambda_2,$$

а система ограничений в соответствии с (4.30)

$$\begin{bmatrix} 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ -2 & 2 & 0 & 0 & 1 & 0 & -1 & -1 \\ 2 & -4 & 0 & 0 & 0 & 1 & -2 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ y_1 \\ y_2 \\ v_1 \\ v_2 \\ \lambda_1 \\ \lambda_2 \end{bmatrix} = \begin{bmatrix} 10 \\ 6 \\ -10 \\ 0 \end{bmatrix}, \quad z \geq 0$$

или в скалярной форме

$$\begin{cases} x_1 + 2x_2 + y_1 = 10; \\ x_1 + x_2 + y_2 = 6; \\ -2x_1 + 2x_2 + v_1 - \lambda_1 - \lambda_2 = -10; \\ 2x_1 - 4x_2 + v_2 - 2\lambda_1 - \lambda_2 = 0; \\ x_1, x_2, y_1, y_2, v_1, v_2, \lambda_1, \lambda_2 \geq 0. \end{cases}$$

4.4. МЕТОД ПРОЕКЦИИ ГРАДИЕНТА

4.4.1. Основные этапы вычислительной процедуры

Этот метод предназначен для решения задач с линейными ограничениями. Пусть задача приведена к следующей форме:

$$\max f(X),$$

$$A_0 X \leq b_0.$$

Рассмотрим общую идею метода. Предположим, что начальная точка $X^{(0)}$ принадлежит допустимой области. Если точка безусловного оптимума также принадлежит допустимой области, то задачу условной оптимизации целесообразно сформулировать как задачу безусловной оптимизации и использовать для ее решения соответствующие методы (рассмотренные во второй главе). Если безусловный оптимум находится вне допустимой области, необходимо любым из методов безусловной оптимизации построить последовательность итераций $\{X^{(i)}\}$, выводящую на границу допустимой области. Далее в действие вводится собственно метод проекции градиента, итерации

$$X^{(i+1)} = X^{(i)} + t^{(i)} K^{(i)}$$

которого осуществляются таким образом, что точка $X^{(i)}$ всегда принадлежит допустимой области. Это свойство метода обеспечивается надлежащим выбором направления $K^{(i)}$ и длины шага $t^{(i)}$.

Дадим краткую характеристику основных этапов стандартной итерации метода: выбора направления, выбора длины шага, проверки условий останова.

Направление $K^{(i)}$ поиска должно быть эффективным и принадлежать конусу возможных направлений. Этим требованиям, которые в принципе можно считать общими для выбора направлений поиска в задачах условной оптимизации (см. п. 4.1), удовлетворяет, в частности, направление, являющееся проекцией градиента на систему активных ограничений.

Длина шага $t^{(i)}$ должна обеспечивать $\max f(X^{(i)} + t^{(i)} K^{(i)})$ при обязательном условии, что точка $X^{(i+1)} = X^{(i)} + t^{(i)} K^{(i)}$ принадлежит допустимой области. Проверка условий останова осуществляется на каждой итерации метода и состоит в проверке выполнения условий Куна-Таккера в точке $X^{(i)}$.

Рассмотрим принцип выбора направления, составляющий основу метода проекции градиента.

4.4.2. Выбор направления $K^{(i)}$

Задача выбора направления формулируется как задача построения оператора проекции градиента на систему активных ограничений и упрощается тем обстоятельством, что в случае линейных ограничений, определяющих допустимую область, оператор проекции на каждой итерации имеет единую форму. Причем на каждой итерации учитываются только активные ограничения, так как вся траектория поиска в данном методе принадлежит границам допустимой области.

Введем обозначения. Пусть A — матрица активных ограничений, сформированная из строк исходной матрицы A_0 , соответствующих тем ограничениям, которые выполняются как строгие равенства в точке $X^{(i)} : AX^{(i)} = b$. Тогда

$$P = E - A^T (AA^T)^{-1} A \quad (4.31)$$

есть оператор проекции вектора на систему $AX = b$ активных ограничений [7].

Формула (4.31) может быть получена в результате решения задачи определения проекции точки Z на активное линейное многообразие $AX = b$. Эта вспомогательная задача условной оптимизации формулируется следующим образом:

$$\begin{aligned} & \min \|X - Z\|, \\ & X \in D = \{X : AX = b\} \end{aligned}$$

и решается методом неопределенных множителей Лагранжа.

Операция проектирования вектора и, в частности, градиента $f'(X^{(i)})$ на систему равенств $AX = b$ есть умножение его слева на матрицу P [7]:

$$K^{(i)} = Pf'(X^{(i)}).$$

Дальнейшее описание метода целесообразно дать в форме изложения пошагового алгоритма стандартной итерации.

4.4.3. Алгоритм стандартной итерации

Пусть $X^{(0)} \in D$, $D = \{X : A_0 X \leq b_0\}$.

Представим матрицу A_0 в следующем виде:

$$A_0 = \begin{bmatrix} A_1 \\ A \end{bmatrix},$$

где $A = \{a_k\}$, $k \in I$ — матрица активных ограничений, составленная из строк a_k , I — индексное множество, соответствующее активным ограничениям, а $A_1 = \{a_l\}$, $l \notin I$.

Шаг 1. Вывод траектории на границу допустимой области

Если $I = \emptyset$, то необходимо любым из методов безусловной оптимизации вывести траекторию поиска на границу допустимой области.

Если, $I \neq \emptyset$ то необходимо убедиться в принадлежности любого из эффективных направлений поиска $K^{(i)}$ (градиентного, ньютоновского и т. д.) конусу возможных направлений:

$$K_e = \{K^{(i)} : AK^{(i)} \leq 0\}.$$

Если выбранное таким образом направление является допустимым, то переход к шагу 2 иначе — к шагу 3.

Шаг 2. Формирование набора активных ограничений: $A = \{a_k\}, k \in I$

Найдем оператор проекции $P = E - A^T (AA^T)^{-1}A$. Определим проекцию градиента $Pf'(X^{(i)})$. Если $Pf'(X^{(i)}) \neq 0$, то $K^{(i)} = Pf'(X^{(i)})$. И переход к шагу 3. Если $Pf'(X^{(i)}) = 0$, то переход к шагу 4.

Шаг 3. Выбор длины $t^{(i)}$ шага в направлении $K^{(i)}$

Сначала необходимо выявить те ограничения, которые в принципе могут быть нарушены при движении из точки $X^{(i)}$ в направлении $K^{(i)}$. Для этого необходимо проанализировать знаки компонентов вектора $A_0 K^{(i)}$ и выделить положительные компоненты, соответствующие именно тем ограничениям, которые могут быть нарушены. Обозначим $I_{npe\partial}$ — множество индексов нарушаемых ограничений, тогда

$$t^{(i)} = \min\{t^*, t_{npe\partial}\}, \quad (4.32)$$

где

$$t^* = \arg \max_{(t)} f(X^{(i)} + t K^{(i)}), \quad (4.33)$$

$$\text{а } t_{npe\partial} = \min_{k \in I_{npe\partial}} \{t_{npe\partial_k}\}, \quad (4.34)$$

$$\text{где } t_{npe\partial_k} = \frac{b_k - a_k X^{(i)}}{a_k K^{(i)}} = \frac{b_k - \sum_{j=1}^n a_{kj} x_j^{(i)}}{\sum_{j=1}^n a_{kj} K_j^{(i)}}. \quad (4.35)$$

Переход к шагу 2.

Шаг 4. Анализ условий останова

Этот анализ необходимо производить в тех случаях, когда $Pf'(X^{(i)}) = 0$, так как данное равенство при выполнении некоторых

дополнительных условий интерпретируется как условие Куна-Таккера существования оптимума в точке $X^{(i)}$. В соответствии с (4.31)

$$\begin{aligned} Pf'(X^{(i)}) &= (E - A^T(AA^T)^{-1}A) f'(X^{(i)}) = \\ &= f'(X^{(i)}) - A^T(AA^T)^{-1}Af'(X^{(i)}). \end{aligned}$$

Произведение $-(AA^T)^{-1}Af'(X^{(i)})$ есть вектор, обозначим его Λ , тогда условие

$$Pf'(X^{(i)}) = f'(X^{(i)}) + A^T\Lambda = 0.$$

является одним из двух условий (4.8) Куна-Таккера для исходной задачи. Выполнение второго условия

$$\Lambda = -(AA^T)^{-1}Af'(X^{(i)}) \leq 0$$

необходимо проверить.

Если $\Lambda \leq 0$, то оптимальное решение найдено: $X^{(i)} = X^*$ и переход к шагу 6.

Если часть элементов λ_k вектора Λ больше нуля, то переход к шагу 5.

Шаг 5. Модификация матрицы активных ограничений

Каждая модифицированная матрица строится из матрицы активных ограничений путем исключения из нее строки, соответствующей положительному значению элемента λ_k вектора Λ . Число модифицированных матриц равно числу положительных элементов вектора Λ .

Для каждой из модифицированных матриц \tilde{A}_k формируют оператор проекции \tilde{P}_k . Далее выбирают самое эффективное направление по следующему критерию:

$$K^{(i)} = \tilde{P}_l f'(X^{(i)}) = \arg \max_{\substack{(k) \\ \lambda_k > 0}} \left\| \tilde{P}_k f'(X^{(i)}) \right\|.$$

Переход к шагу 3.

Шаг 6. Останов.

Пример 4.4

Найти $\max\{f(X) = -(x_1 - 1)^2 - (x_2 - 2)^2\}$ при ограничениях

$$\begin{cases} -x_1 + 2x_2 \leq 2; \\ x_1 + x_2 \leq 4; \\ x_1, x_2 \geq 0, \quad X^{(0)} = (0, 0)^T. \end{cases}$$

Перепишем систему ограничений в следующем виде:

$$\begin{aligned} -x_1 + 2x_2 &\leq 2, \\ x_1 + x_2 &\leq 4, \\ -x_1 &\leq 0, \\ -x_2 &\leq 0, \end{aligned}$$

тогда

$$A_0 = \begin{bmatrix} -1 & 2 \\ 1 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix}.$$

Определим градиент и матрицу Гессе целевой функции:

$$f'(x) = \begin{bmatrix} -2(x_1 - 1) \\ -2(x_2 - 2) \end{bmatrix}, \quad H = \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix},$$

Шаг 1. Определение матрицы активных ограничений в точке

$$X^{(0)} = (0, 0)^T$$

Убедимся в допустимости градиентного направления:

$$Af'(X^{(0)}) = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 2 \\ 4 \end{bmatrix} = \begin{bmatrix} -2 \\ -4 \end{bmatrix} < 0,$$

поэтому

$$K^{(0)} = f'(X^{(0)}) = (2, 4)^T.$$

Шаг 2. Выбор длины шага $t^{(0)}$

Найдем множество $I_{\text{нред}}$ индексов нарушенных ограничений.

$$AK^{(0)} = \begin{bmatrix} -1 & 2 \\ 1 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 2 \\ 4 \end{bmatrix} = \begin{bmatrix} 6 \\ 6 \\ -2 \\ -4 \end{bmatrix}, I_{npe\partial} = \{1, 2\}.$$

Тогда $t^{(\theta)} = \min \{ t^*, t_{npe\partial_1}, t_{npe\partial_2} \};$

$$t^* = \frac{\langle f'(X^{(0)}, K^{(0)}) \rangle}{\langle K^{(0)}, HK^{(0)} \rangle} = \frac{[2;4] \begin{bmatrix} 2 \\ 4 \end{bmatrix}}{[2;4] \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix} \begin{bmatrix} 2 \\ 4 \end{bmatrix}} = \frac{1}{2};$$

$$t_{npe\partial_1} = \frac{2}{[-1;2] \begin{bmatrix} 2 \\ 4 \end{bmatrix}} = \frac{1}{3}; \quad t_{npe\partial_2} = \frac{4}{[1;1] \begin{bmatrix} 2 \\ 4 \end{bmatrix}} = \frac{2}{3};$$

$$t^{(0)} = t_{npe\partial_1} = \frac{1}{3};$$

$$X^{(1)} = X^{(0)} + t^{(0)} K^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \frac{1}{3} \begin{bmatrix} 2 \\ 4 \end{bmatrix} = \frac{1}{3} \begin{bmatrix} 2 \\ 4 \end{bmatrix}.$$

Шаг 3. Формирование матрицы активных ограничений

Определим оператор проекции и направление $K^{(1)}$.

$$A = [-1, 2]; \quad f'(X^{(1)}) = \frac{1}{3} \begin{bmatrix} 2 \\ 4 \end{bmatrix};$$

$$P = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} -1 \\ 2 \end{bmatrix} [-1; 2] \begin{bmatrix} -1 \\ 2 \end{bmatrix}^{-1} [-1; 2] =$$

$$= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \frac{1}{3} \begin{bmatrix} 1 & -2 \\ -2 & 4 \end{bmatrix} = \frac{1}{5} \begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix};$$

$$K^{(1)} = \frac{1}{5} \cdot \frac{1}{3} \begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 4 \end{bmatrix} = \frac{8}{15} \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Шаг 4. Выбор длины шага $t^{(1)}$. Найдем множество $I_{npe\partial}$:

$$A_0 K^{(1)} = \frac{8}{15} \begin{bmatrix} -1 & 2 \\ 1 & 1 \\ 0 & -1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \frac{8}{15} \begin{bmatrix} 0 \\ 3 \\ -1 \\ -2 \end{bmatrix}, \quad I_{npe\partial} = \{2\},$$

тогда

$$t^{(1)} = \min\{t^*, t_{npe\partial_2}\}$$

$$t_{npe\partial_2} = \frac{4 - \left(\frac{2}{3} + \frac{4}{5}\right)}{[1; 1] \begin{bmatrix} 2 \\ 1 \end{bmatrix} \left(\frac{8}{15}\right)} = \frac{5}{4};$$

$$t^{(1)} = t^* = \frac{-\left(\frac{1}{3}\right)\left(\frac{8}{15}\right)[2; 4] \begin{bmatrix} 2 \\ 1 \end{bmatrix}}{\left(\frac{8}{15}\right)^2 [2; 1] \begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix} \begin{bmatrix} 2 \\ 1 \end{bmatrix}} = \frac{1}{2};$$

$$X^{(2)} = \begin{bmatrix} \frac{2}{3} \\ \frac{4}{3} \end{bmatrix} + \frac{1}{2} \cdot \frac{8}{15} \begin{bmatrix} 2 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{18}{15} \\ \frac{24}{15} \end{bmatrix}.$$

Шаг 5. Выбор направления

В точке $X^{(2)}$ матрица активных ограничений и оператор проекции остаются прежними, поэтому

$$K^{(2)} = Pf'(X^{(2)}) = \frac{1}{5} \begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix} \frac{2}{5} \begin{bmatrix} -1 \\ 2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Шаг 6. Проверка условий останова.

Найдем элементы вектора

$$\Lambda = -(AA^T)^{-1}A f'(X^{(2)}).$$

$$\Lambda = -\left\{[-1; 2] \begin{bmatrix} -1 \\ 2 \end{bmatrix}\right\}^{-1} [-1; 2] \left(\frac{2}{5}\right) \begin{bmatrix} -1 \\ 2 \end{bmatrix} = -\frac{2}{5}.$$

Вектор $\Lambda = \{\lambda_1\} = -\frac{2}{5}$ состоит из одного отрицательного элемента, следовательно, точка $X^{(2)} = (18/15; 24/15)^T$ — оптимальное решение исходной задачи. Область допустимых решений и точки траектории поиска приведены на рис. 4.4.

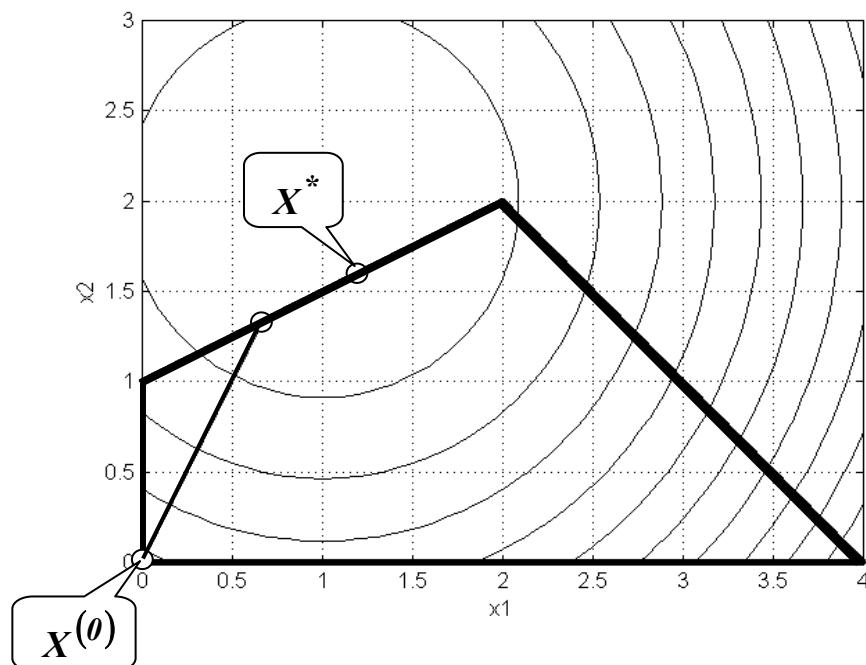


Рис. 4.4

4.5. МЕТОД ВОЗМОЖНЫХ НАПРАВЛЕНИЙ ЗОЙТЕНДЕЙКА

В отличие от метода проекции градиентов, ориентированного на линейные ограничения, рассматриваемый метод применим к общему случаю нелинейных ограничений, когда исходная задача формулируется в следующем виде:

$$\begin{aligned} & \max f(X); \\ & g_j(X) \geq 0, \quad j = \overline{1, J}. \end{aligned} \tag{4.36}$$

Принципиальная особенность метода возможных направлений проявляется в выборе направления, когда траектория поиска выходит на границу области: $g_l(X^{(i)}) = 0$, $l \in I$, где I — множество активных ограничений. Тогда задача выбора направления $K^{(i)} \in K_e(X^{(i)}) \cap K_o(X^{(i)})$, удовлетворяющего требованиям допустимости и эффективности (см. п. 4.1), формулируется как вспомогательная задача линейного программирования в следующем виде:

$$\begin{aligned} & \max u, \\ & \langle f'(X^{(i)}), K^{(i)} \rangle \geq u, \\ & \langle g'_l(X^{(i)}), K^{(i)} \rangle \geq u, \quad l \in I, \\ & u \geq 0, \quad -1 \leq K_j^{(i)} \leq 1, \quad j = \overline{1, n}. \end{aligned} \tag{4.37}$$

Сформулированная задача имеет следующее целевое назначение: максимизировать приращение функций $f(X)$ и $g_l(X)$, $l \in I$ при движении точки траектории по направлению $K^{(i)}$. Тем самым обеспечивается выполнение двух условий: возрастание целевой функции в направлении $K^{(i)}$ и перемещение внутрь области по ограничениям $g_l(X^{(i)}) = 0$, $l \in I$. Ограничения на составляющие $K_j^{(i)}$ вектора $K^{(i)}$ необходимы как условие существования оптимума вспомогательной задачи.

В случае, если решение $u_{opt} > 0$, такое направление $K^{(i)}$ существует, в противном случае ($u_{opt} = 0$) рассматриваемая точка $X^{(i)}$ является оптимальной.

Отметим, что при выборе направления не отдается предпочтения ни одному из двух упомянутых требований: допустимости и эффективности.

Это следует из формулировки вспомогательной задачи – правые части неравенств с градиентами $f'(X^{(i)})$ и $g'(X^{(i)})$ совпадают.

Метод Зойтендейка может быть формализован в виде алгоритма, определяющего последовательность шагов стандартной итерации метода.

Пусть исходная задача приведена к форме (4.36), в точке $X^{(i)} \in D$ множество I активных ограничений не пустое.

Шаг 1. Решение задачи (4.37) линейного программирования. Пусть $K^{(i)}$ и u — результат решения задачи (4.37).

Шаг 2. Анализ решения задачи (4.37). Если $u = 0$, переход к *шагу 5*; если $u > 0$, переход к *шагу 3*.

Шаг 3. Поиск длины шага в направлении $K^{(i)}$. Найти

$$t^{(i)} = \min \left\{ t^*, \left\{ t_{n_{pred_j}} \right\} \right\},$$

где

$$\begin{aligned} t^* &= \arg \max_{(t)} \left\{ f(X^{(i)} + tK^{(i)}) \right\}, \\ \text{а } t_{n_{pred_j}} &= \arg \left\{ g_j(X^{(i)} + tK^{(i)}) = 0 \right\}, \quad j = \overline{1, J}. \end{aligned}$$

Шаг 4. Определение новой координаты $X^{(i+1)}$:

$$X^{(i+1)} = X^{(i)} + t^{(i)} K^{(i)},$$

присвоение i значения $i+1$, переход к *шагу 1*.

Шаг 5. Оптимальное решение найдено, так как дальнейшее увеличение целевой функции в пределах допустимой области невозможно:
 $X^* = X^{(i)}$, *останов*.

При построении конкретного алгоритма, реализующего метод Зойтендейка, необходимо задать точность определения оптимального решения исходной задачи. Поэтому теоретически обоснованное условие оптимальности $u = 0$ трансформируется в признак останова $u \leq \epsilon$, где ϵ — допустимая погрешность определения наступления условий оптимальности. Все точки $X^{(i)}$ траектории поиска принадлежат допустимой области и, если в какой-либо из них отсутствуют активные ограничения, в алгоритм включается любой из методов выбора направления, используемый в задачах без-

условной оптимизации вплоть до появления активных ограничений в очередной точке траектории.

Пример 4.5

Найти максимум $f(X) = -(x_1 - 3)^2 - (x_2 - 3)^2$
 при ограничениях $\begin{cases} g_1(X) = -x_2^2 + 2x_1 - 1 \geq 0; \\ g_2(X) = -0,8x_1^2 - 2x_2 + 9 \geq 0. \end{cases}$

Градиенты $f'(X)$, $g'_1(X)$, $g'_2(X)$ — линейные функции:

$$\begin{aligned} f'(X) &= [2(3-x_1), 2(3-x_2)]^T; \\ g'_1(X) &= [2, -2x_2]^T; \\ g'_2(X) &= [-1,6x_1, -2]^T. \end{aligned}$$

Пусть $X^{(0)} = (1, 1)^T$, $X^{(0)} \in D$, так как $g_1(X^{(0)}) = 0$ и $g_2(X^{(0)}) = 6,2 > 0$. Кроме того, первое ограничение выполняется как равенство:

$$I = \{1\} \neq \emptyset.$$

Шаг 1. Поиск направления $K^{(0)}$

Для этого сформулируем вспомогательную задачу:

$$\begin{aligned} \max u, \\ \begin{cases} 4K_1^{(0)} + 4K_2^{(0)} \geq u; \\ 2K_1^{(0)} - 2K_2^{(0)} \geq u; \\ -1 \leq K_1^{(0)} \leq 1; \\ -1 \leq K_2^{(0)} \leq 1. \end{cases} \end{aligned}$$

Получим решение этой задачи:

$$K^{(0)} \cong (1, -0,333)^T; \quad u \cong 2,666.$$

Шаг 2. Анализ решения вспомогательной задачи.

$u \cong 2,666 > 0$, переход к шагу 3.

Шаг 3. Поиск длины $t^{(0)}$ шага в направлении $K^{(0)}$.

$$t^{(0)} = \min \{t^*, t_{nped_1}, t_{nped_2}\};$$

$$t^* = \arg \max_{(t)} \{f(X^{(0)} + tK^{(0)})\} = 1,200;$$

$$t_{nped_2} = \arg \{g_2(X^{(0)} + tK^{(0)}) = 0\} \cong 2,261;$$

$$t^{(0)} = \min \{1,200; 2,261\} = 1,200;$$

$$t_{nped_1} = \arg \{g_1(X^{(0)} + tK^{(0)}) = 0\} \cong 24,000.$$

Шаг 4. Определение координат точки $X^{(1)}$:

$$X^{(1)} = X^{(0)} + t^{(0)}K^{(0)} \cong$$

$$\cong (1,1)^T + 1,200(1, -0,333)^T \cong (2,200, 0,600)^T.$$

Шаг 5. Поиск направления $K^{(1)}$

Определение множества активных ограничений: $I = \emptyset$.

Так как $I = \emptyset$, выбор $K^{(1)}$ осуществляется в соответствии с методом безусловной оптимизации. Выберем метод Ньютона:

$$K^{(1)} = -H^{-1}(f'(X^{(1)})) = -\begin{bmatrix} -2 & 0 \\ 0 & -2 \end{bmatrix}^{-1} \begin{bmatrix} 1,600 \\ 4,800 \end{bmatrix} = \begin{bmatrix} 0,800 \\ 2,400 \end{bmatrix}.$$

Шаг 6. Поиск длины $t^{(1)}$ шага в направлении $K^{(1)}$.

$$t^{(1)} = \min \{t^*, t_{nped_2}, t_{nped_1}\};$$

$$t^* = 1,000;$$

$$t_{nped_1} = \arg \{g_1(X^{(1)} + tK^{(1)}) = 0\} \cong 0,624;$$

$$t_{nped_2} = \arg \{g_2(X^{(1)} + tK^{(1)}) = 0\} \cong 0,499;$$

$$t^{(1)} = t_{nped_2} = 0,499.$$

Шаг 7. Определение координат точки $X^{(2)}$.

$$X^{(2)} = X^{(1)} + t^{(1)}K^{(1)} \cong (2,200, 0,600)^T + \\ + 0,499(0,800, 2,400)^T \cong (2,599, 1,798)^T.$$

Шаг 8. Определение множества I активных ограничений в точке $X^{(2)}$:

$$I = \{2\}.$$

Шаг 9. Поиск направления $K^{(2)}$.

Формулировка вспомогательной задачи

$$\max u,$$

$$\begin{cases} 1,600K_1^{(2)} + 4,800K_2^{(2)} \geq u; \\ -4,159K_1^{(2)} + 2,000K_2^{(2)} \geq u; \\ -1 \leq K_1^{(2)} \leq 1; \\ -1 \leq K_2^{(2)} \leq 1. \end{cases}$$

Решение задачи: $K^{(2)} \cong (-1,000, 0,847)$, $u \cong 2,465$.

Шаг 10. Поиск длины $t^{(2)}$ шага в направлении $K^{(2)}$.

$$t^{(2)} = \min \{t^*, t_{nped_1}, t_{nped_2}\};$$

$$t^* \cong 0,718;$$

$$t_{nped_1} = \arg \{g_1(X^{(1)} + tK^{(1)}) = 0\} \cong 0,187;$$

$$t_{nped_2} = \arg \{g_2(X^{(1)} + tK^{(1)}) = 0\} \cong 3,081;$$

$$t^{(2)} = t_{nped_1} = 0,187.$$

Шаг 11. Определение координат точки $X^{(3)}$

$$X^{(3)} = X^{(2)} + t^{(2)}K^{(2)} \cong$$

$$\cong (2,599, 1,798)^T + 0,187(-1,000, 0,847)^T \cong (2,412, 1,956)^T.$$

В табл. 4.8 приведены данные о дальнейших шагах решения задачи оптимизации. Итерации метода продолжаются вплоть до достижения максимума целевой функции в окрестности точки $X^* = (2,5; 2,0)^T$.

Анализ полученных результатов позволяет сделать вывод о возможной слабой сходимости метода: для данной задачи значение параметра u довольно сильно отличается от 0 даже вблизи точки X^* . Отметим, что в практической реализации метода Зойтендайка целесообразно применять совместно несколько признаков останова: например, в добавление к уже упомянутому $u < \epsilon$ использовать условие

$$\|X^{(i)} - X^{(i-1)}\| < \delta.$$

На рис. 4.5 показаны допустимая область исходной задачи, линии равного уровня целевой функции и траектория поиска оптимального решения X^* .

Таблица 4.8

	Номер шага i						X_{opt}
	0	1	2	3	4	5	
$x_1^{(i)}$	1,000	2,200	2,599	2,412	2,515	2,491	2,500
$x_2^{(i)}$	1,000	0,600	1,798	1,956	1,970	1,996	2,000
u	2,666	–	2,465	1,463	1,150	1,347	0,000

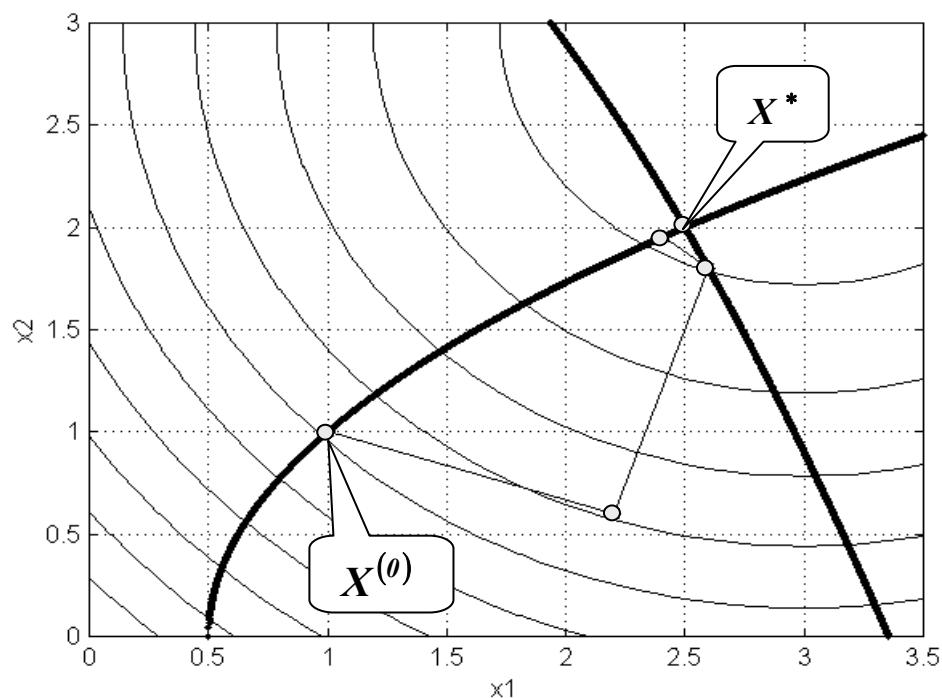


Рис. 4.5

4.6. СВЕДЕНИЕ ЗАДАЧ УСЛОВНОЙ ОПТИМИЗАЦИИ К ЗАДАЧАМ БЕЗУСЛОВНОЙ ОПТИМИЗАЦИИ. МЕТОД ШТРАФНЫХ И БАРЬЕРНЫХ ФУНКЦИЙ

Суть рассматриваемых методов состоит в замене исходной задачи условной оптимизации эквивалентной ей задачей безусловной оптимизации или последовательностью таких задач. Стимулом к такой замене является наличие эффективных, надежных и сравнительно простых методов безусловной оптимизации.

Аналогичный подход используется, как уже отмечалось, в методе Лагранжа, предполагающем замену исходной задачи задачей безусловной оптимизации функции Лагранжа.

В данном случае также формируется специальная функция, относительно которой ставится задача безусловной оптимизации. В зависимости от метода построения специальной функции рассматриваемая идея может быть реализована с использованием штрафных или барьерных функций.

В первом случае специальная функция носит название штрафной и формируется как сумма целевой функции и функции штрафа за нарушение ограничений.

Метод штрафной функции является методом внешней точки, то есть вся траектория поиска располагается вне допустимой области

В методе барьерной функции оптимизируется так называемая барьерная функция — сумма целевой функции и функции барьера. Последняя имеет смысл платы за приближение изнутри допустимой области к ее границам.

Таким образом, метод барьерной функции строится как метод внутренней точки: все точки траектории поиска обязаны принадлежать допустимой области.

Рассмотрим основные аспекты построения и использования излагаемых методов для исходной задачи, заданной в следующем виде:

$$\begin{aligned} & \max f(X); \\ & g_j(X) \leq 0, \quad j = \overline{1, J}; \\ & h_k(X) = 0, \quad k = \overline{1, K}. \end{aligned} \tag{4.38}$$

Метод штрафных функций

Формирование штрафной функции. Штрафная функция $F_0(X, \mu)$ формируется следующим образом:

$$F_0(X, \mu) = f(X) - \mu \left[\sum_{j=1}^J \Psi_0(g_j(X)) + \sum_{k=1}^K \Phi_0(h_k(X)) \right], \quad (4.39)$$

где весовой коэффициент $\mu \geq 0$ а $\Psi_0(y)$ и $\Phi_0(y)$ выполняют роль штрафов за нарушение ограничений исходной задачи и должны удовлетворять следующим требованиям:

$$\Psi_0(y) = 0, \quad y \leq 0; \quad \Psi_0(y) > 0, \quad y > 0,$$

если $y_2 > y_1 > 0$, то $\Psi_0(y_2) > \Psi_0(y_1) > 0$.

$$\Phi_0(y) = 0, \quad y = 0; \quad \Phi_0(y) > 0, \quad y \neq 0,$$

если $|y_2| > |y_1|$, то $\Phi_0(y_2) > \Phi_0(y_1) > 0$.

Приведем типичные формы штрафов Φ_0 и Ψ_0 (рис. 4.6):

$$\Phi_0(y) = |y|, \quad \Phi_0(y) = y^p;$$

$$\Psi_0(y) = \{\max\{0, y\}\}^p,$$

где p — целое положительное число.

Штраф, имеющий смысл «наказания» за удаления от границы, уменьшается по мере приближения к ней из внешних точек. Это свойство функции штрафа обеспечивается специально подобранными формами функций Φ_0 и Ψ_0 .

Параметр μ вводится в штрафную функцию $F_0(X, \mu)$ для усиления чувствительности последней к факту пребывания точки вне допустимой области, так как метод штрафной функции является методом внешней точки.

Обоснование сходимости метода штрафных функций

Если параметр μ задан, то $X^*(\mu)$ — решение задачи безусловной максимизации штрафной функции $F_0(X, \mu)$.

В соответствии с методом штрафной функции генерируется последовательность подзадач

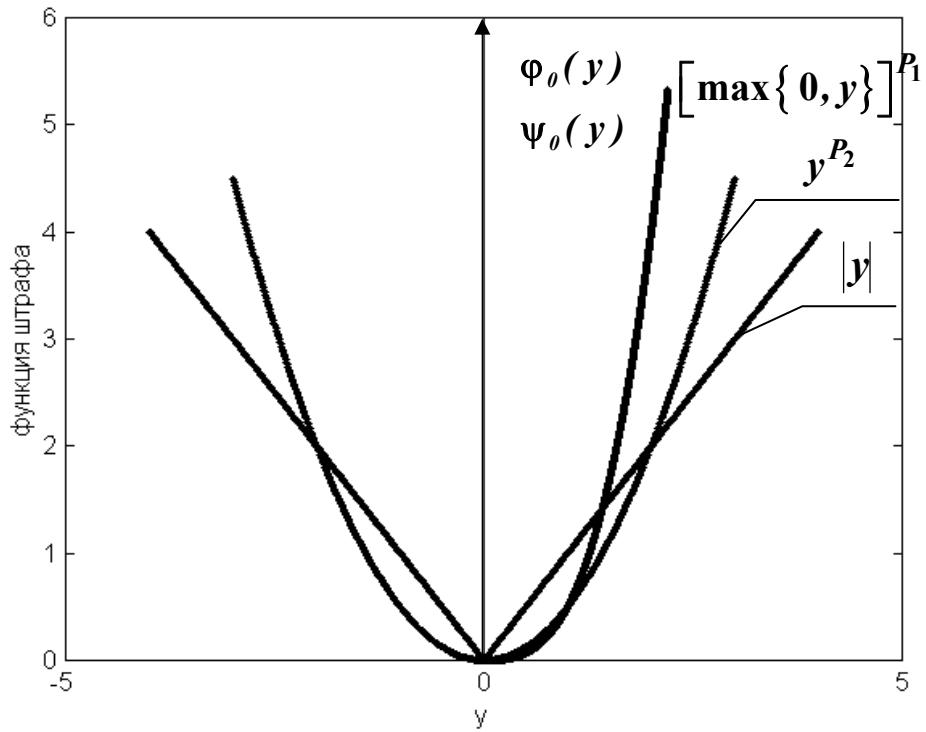


Рис. 4.6

$$X^*(\mu_i) = \arg \max_X F_0(X, \mu_i), \\ X \in (R^n \setminus D), \quad X^{(0)} = X^*(\mu_{i-1}),$$

в которых параметр μ монотонно увеличивается от некоторого начального значения μ_0 : $\mu_i > \mu_{i-1}$, $i = 1, 2, \dots$, а начальной точкой i -й подзадачи является решение $X^*(\mu_{i-1})$ ($i-1$)-й подзадачи. Последовательность подзадач порождает последовательность $\{X^*(\mu)\}$, которая при некоторых условиях сходится к оптимальной точке исходной задачи.

Сходимость метода штрафных функций обоснована в ряде теорем [8], определяющих требования к исходной задаче (в частности, требование непрерывности функций f, g, h в пространстве R^n) и обосновывающих выполнение условия:

$$\lim_{\mu \rightarrow \infty} F_0(X^*(\mu), \mu) = f(X^*)$$

где X^* — оптимальное решение задачи.

Рассмотрим еще один существенный аспект построения алгоритма штрафных функций, непосредственно связанный со сходимостью метода.

Выбор штрафного параметра

Метод штрафных функций основывается на том, что последовательность решений $\{X^*(\mu)\}$ подзадач безусловной оптимизации штрафной функции при устремлении μ к бесконечности имеет своим пределом решение X^* исходной задачи.

Возникает естественный вопрос: для чего необходимо выстраивать последовательность указанных подзадач, а не ограничиться одной из них с достаточно большим значением μ , обеспечивающим приемлемую точность определения X^* .

Для ответа на этот вопрос необходимо выяснить, каким образом значение параметра μ влияет на трудоемкость решения вспомогательной подзадачи. При увеличении μ происходит деформация линий равного уровня штрафной функции, ухудшающая сходимость методов безусловной оптимизации, используемых для решения подзадач, и, что особенно важно, увеличивается зависимость скорости сходимости этих методов от выбора начальной точки.

Постепенное увеличение параметра μ в последовательности подзадач позволяет постепенно приближать начальную точку подзадачи к точке условного экстремума, улучшая тем самым сходимость методов безусловной оптимизации штрафной функции.

Метод барьерных функций

Этот метод предназначен для решения исходной задачи, приведенной к следующей форме:

$$\begin{aligned} & \max f(X); \\ & g_j(X) \leq 0, \quad j = \overline{1, J}; \\ & X \in R^n. \end{aligned}$$

Отсутствие ограничений типа $h_k = 0$, т. е. ограничений-равенств, в формулировке исходной задачи связано с особенностью метода, являющегося методом внутренней точки: все точки траектории поиска должны принадлежать внутренности допустимого множества. Это условие при наличии ограничений-равенств обычно не может быть выполнено.

Барьерная функция $F_1(X, \mu)$ формируется следующим образом:

$$F_1(X, \mu) = f(X) - \mu \sum_{j=1}^J \psi_1(g_j(X)), \quad (4.40)$$

где μ — весовой коэффициент, $\mu > 0$, а $\psi_1(y)$ — функция, выполняющая роль барьера, предотвращающего выход точек траектории поиска из допустимой области.

К функции $\psi_1(y)$ предъявляются следующие требования:

$$\psi_1(y) \geq 0, \quad y < 0;$$

$$\lim_{y \rightarrow 0^-} \psi_1(y) = \infty.$$

Этим условиям удовлетворяют следующие функциональные зависимости, обычно используемые в качестве барьеров, рис. 4.7:

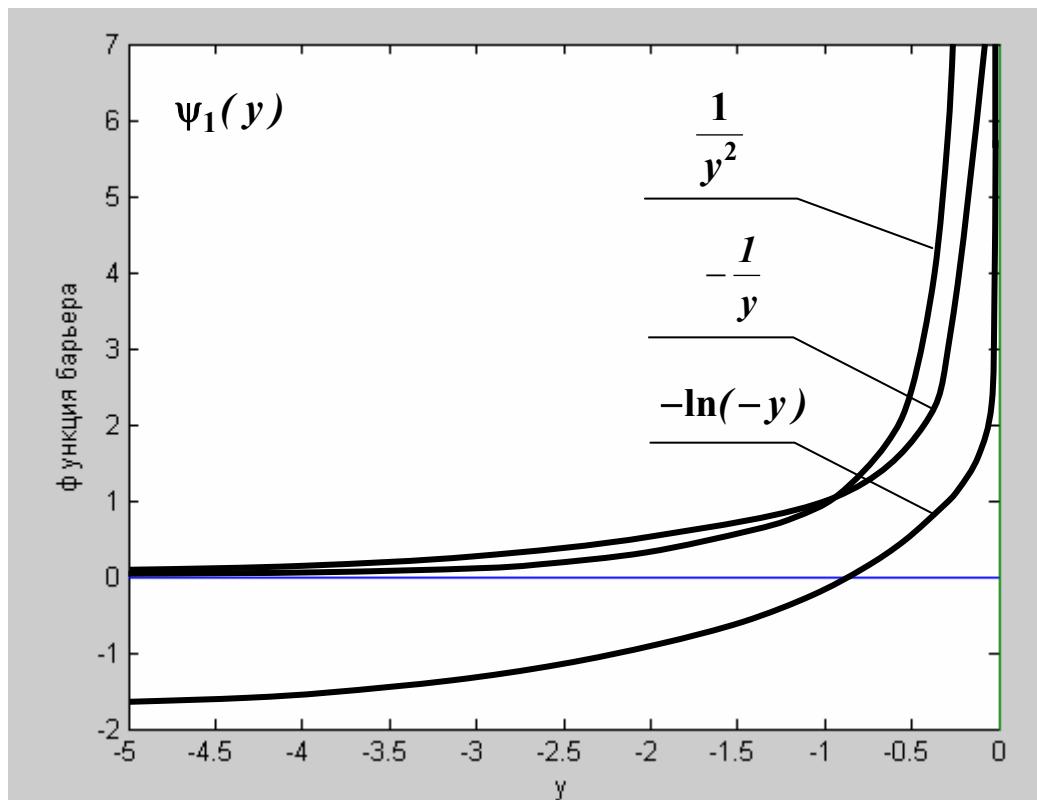


Рис. 4.7

$$\psi_1(y) = \frac{1}{y^p}, \quad \text{где } p \text{ — четное,} \quad \psi_1(y) = -\frac{1}{y^p}, \quad \text{где } p \text{ — нечетное,}$$

$$\psi_1(y) = -\log_a(-y)$$

Метод барьерных функций предполагает решение последовательности задач безусловной максимизации барьерной функции $F_1(X, \mu)$ при монотонном уменьшении параметра μ .

Сходимость метода формулируется как выполнение условия

$$\lim_{\mu \rightarrow 0^+} F_1(X^*(\mu), \mu) = f(X^*)$$

и гарантируется рядом теорем, приведенных и доказанных в [8].

По мере приближения точек последовательности $\{X^*(\mu)\}$ к оптимальному решению X^* и, следовательно, к границе допустимой области происходит возрастание барьера $\Psi_1(y)$, которое компенсируется уменьшением весового коэффициента μ . Эта компенсация осуществляется, чтобы

$$\lim_{\mu \rightarrow 0^+} \mu \sum_{(j)} \Psi_1(g_j(X)) = 0.$$

Выбор последовательности $\{\mu\}$ производится с учетом тех же соображений, связанных со сходимостью методов безусловной оптимизации, рассмотренных выше, при анализе метода штрафной функции.

В ряде случаев используют сочетание методов штрафной и барьерной функции и формируют комбинированную функцию $F(X, \mu)$, в виде суммы целевой функции, штрафной и барьерной добавок:

$$F(X, \mu) = f(X) - \mu \left[\sum_{j=1}^J \Psi_0(g_j(X)) + \sum_{k=1}^K \Phi_0(h_k(X)) \right] - \frac{1}{\mu} \sum_{j=1}^J \Psi_1(g_j(X)),$$

при этом штрафная добавка должна быть равна нулю во всех внутренних точках, а барьерная – во всех внешних точках. Решение исходной задачи в соответствии с единой структурой алгоритма метода получается как предел последовательности $\{X^*(\mu)\}$ решений подзадач максимизации $F(X, \mu)$ при увеличении параметра μ .

Цель такой комбинации рассмотренных методов состоит в том, чтобы получить универсальный метод, применимый для решения исходной задачи, заданной в самой общей форме (4.38), при произвольном расположении точек траектории поиска (как внутри, так и вне допустимой области). Универсальность метода предъявляет определенные требования к комбинированной функции $F(X, \mu)$, которая должна быть дифференцируема в R^n . Это требование обеспечивается соответствующим выбором функций барьеров и штрафов, которые тоже должны быть непрерывны и

дифференцируемы в \mathbf{R}^n , а это, в свою очередь, может приводить к существенно более сложным конструкциям функций штрафов и барьеров.

В качестве иллюстрации возможностей рассмотренной группы методов приведем анализ примера решения задачи условной оптимизации методом барьерных функций.

Пример 4.6

Решить задачу

$$\max [-(x_1 - 4)^2 - (x_2 - 4)^2]$$

при ограничении $g(X) = 5 - x_1 - x_2 \geq 0$.

Изменим форму записи ограничения:

$$g_1(X) = -g(X) = x_1 + x_2 - 5 \leq 0.$$

Используем логарифмический барьер $\psi_1(y) = 1 \ln(-y)$, тогда барьерная функция $F_1(X, \mu)$ примет вид:

$$F_1(X, \mu) = -(x_1 - 4)^2 - (x_2 - 4)^2 + \mu \ln[5 - x_1 - x_2]$$

Меняя параметр μ по закону:

$$\mu_i = 0,1 \mu_{i-1}, \quad i = 1, 2, \dots; \quad \mu_0 = 100,$$

получим последовательность подзадач максимизации барьерной функции:

$$\max \{f(x) + \mu \ln[5 - x_1 - x_2]\}$$

порождающую последовательность $\{X^*(\mu)\}$ приближений к оптимальному решению исходной задачи. В табл. 4.9 приведены результаты решения подзадач. На рис. 4.8 изображена допустимая область и линии равного уровня целевой функции исходной задачи. При уменьшении параметра μ происходит вытягивание линий равного уровня барьерной функции вдоль границы, приводящее к ухудшению сходимости методов безусловной оптимизации, используемых для решения подзадач (рис. 4.9).

В данном случае оптимальные решения подзадач могут быть получены аналитически как корни системы уравнений:

$$\begin{cases} \frac{\partial F_1}{\partial x_1} = -2(x_1 - 4) - \mu \left[\frac{1}{5 - x_1 - x_2} \right] = 0, \\ \frac{\partial F_1}{\partial x_2} = -2(x_2 - 4) - \mu \left[\frac{1}{5 - x_1 - x_2} \right] = 0, \end{cases}$$

откуда $x_1^*(\mu) = x_2^*(\mu) = \frac{13}{4} - \frac{1}{4}\sqrt{9 + 4\mu},$

а оптимальное решение исходной задачи:

$$X^* = \lim_{\mu \rightarrow 0} X^*(\mu) = (2,5; 2,5)^T.$$

Приведенная выше процедура получения аналитического решения отнюдь не является формальным изложением метода барьерных функций, а приведена здесь исключительно для иллюстрации возможностей метода, в частности, достаточно быстрой сходимости последовательности $\{X^*(\mu)\}$ к оптимальной точке X^* . В табл. 4.9 приведены результаты решения подзадач, а на рис. 4.8 изображена допустимая область и линии равного уровня целевой функции исходной задачи. При уменьшении параметра μ происходит вытягивание линий равного уровня барьерной функции вдоль границы, приводящее к ухудшению сходимости методов безусловной оптимизации, используемых для решения подзадач (рис. 4.9).

Таблица 4.9

μ	100	10	1,0	0,1	0,01	0
$x_1^*(\mu) = x_2^*(\mu)$	-1,806	1,500	2,349	2,484	2,498	2,500
$F_1(X, \mu)$	147,890	-5,570	-6,650	-4,940	-4,570	-4,500

4.7. Сравнительный анализ методов условной оптимизации

Задача сравнения методов условной оптимизации имеет два существенных аспекта: выбор критерия оценки методов и методики определения показателей качества по выбранному критерию.

Найти единый универсальный критерий в данном случае не представляется возможным, так как методы нелинейного программирования, с одной стороны, и свойства задач, определяющие область их применения, с другой стороны, характеризуются большим числом параметров. Поэтому

критерий оценки методов является комплексным и предполагает учет следующих наиболее важных факторов.

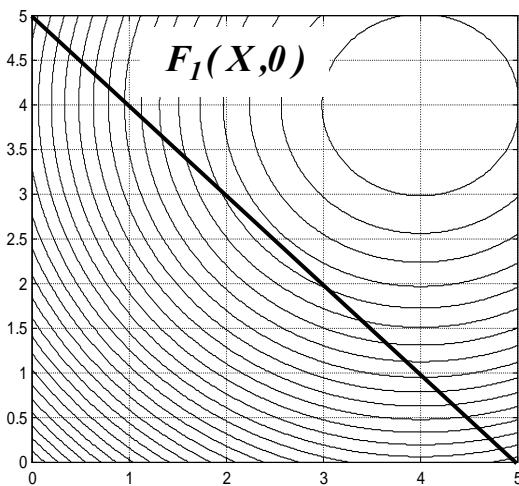


Рис. 4.8

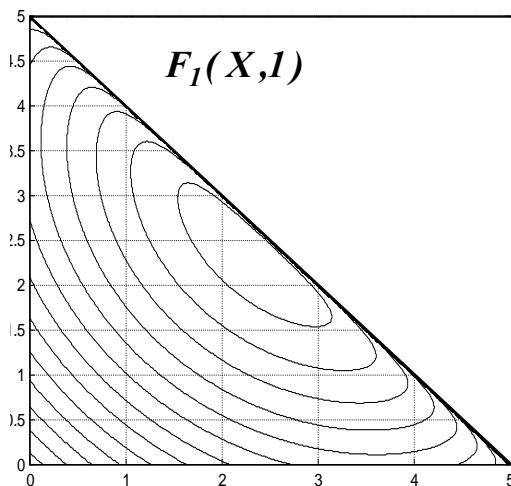


Рис. 4.9

1. Универсальность методов — применимость метода к определенным классам целевых функций и формам ограничений, иными словами, к определенным структурам задач. Эта составляющая критерия наиболее полно рассмотрена при классификации методов условной оптимизации (раздел 4.2).

2. Надежность или устойчивость метода — способность метода обеспечивать решение большинства задач из некоторого класса с разумной точностью.

3. Чувствительность к параметрам алгоритмической схемы метода и начальным данным (например, к расположению начальной точки $X^{(0)}$, выбору длины шага, параметру μ).

4. Вычислительные затраты метода, включающие также предварительную обработку данных.

5. Сходимость метода.

Существуют два подхода к оценке этих или, возможно, несколько иных составляющих критерия эффективности метода: сравнение алгоритмов, реализующих соответствующие методы, на основе пошагового принципа, предполагающего подсчет числа операций в пределах стандартного шага, и сравнение программ на некотором множестве конкретных тестовых примеров.

Оба подхода имеют свои достоинства и недостатки. Анализ стандартных итераций, рассмотренных выше методов приводит, по-видимому, к следующему их ранжированию в порядке возрастания вычислительной

сложности: метод Лагранжа, методы штрафных и барьерных функций, метод возможных направлений, методы квадратичного программирования.

Неточность такого анализа определяется большим разнообразием реальных задач и различной эффективностью используемого программного обеспечения. Поэтому в большинстве исследований используется второй подход, реализуемый в форме сравнения различных программ на определенном наборе задач [5,6]. Результаты, приведенные в [6], позволяют сделать вывод о том, что наиболее эффективные программы основаны на использовании методов квадратичного программирования, проективных методов, метода Лагранжа и методов штрафных и барьерных функций.

5. ЗАДАЧИ ДИСКРЕТНОГО ПРОГРАММИРОВАНИЯ

5.1. СОДЕРЖАНИЕ МЕТОДА ВЕТВЕЙ И ГРАНИЦ

Многие задачи исследования операций, возникающие в различных областях автоматики и вычислительной техники, формализуются в виде математических моделей дискретного программирования. Примеры таких задач: синтез структур информационных связей в больших системах, оптимизация структур и вычислительных процессов, планирование работ при создании АСУ, задачи автоматизации проектирования (выбор оптимальной топологии связей, конструирование системы диагностических тестов) и др. Общая постановка задачи дискретного программирования выглядит следующим образом:

найти $\min f(x_1, x_2, \dots, x_n)$ (5.1)

при $g_i(x_1, x_2, \dots, x_n) \leq 0, i = 1, 2, \dots, m,$

$x = (x_1, x_2, \dots, x_n) \in D,$

где D — некоторое множество, являющееся конечным или счетным.

В задачах дискретного программирования область допустимых решений D является невыпуклой и несвязной. Поэтому отыскание решения таких задач сопряжено со значительными трудностями, для чего необходимы специальные методы. Методы решения задач дискретного программирования по принципу подхода к проблеме делят обычно на три группы: методы отсечения, или отсекающих плоскостей; методы ветвей и границ; методы случайного поиска и эвристические методы. Ввиду широкой распространенности и эффективности метода ветвей и границ для решения широкого круга практических задач в дальнейшем будет рассматриваться только этот подход. Основной принцип поиска вычисления оптимального решения, согласно этому методу включает: определение правил ветвления для построения дерева решений; определение правил оценок.

Выбранная система правил ветвления позволяет провести разбиение как начальной задачи на некоторое число подзадач, так и каждой из полученных подзадач на новые подзадачи. Цель разбиения – снижение размерности задачи и модификация структуры задачи с целью ее упрощения. Основные условия, учитываемые при любых правилах ветвления

$$G_i = \bigcup_{k=1}^L G_{ik}, \quad \bigcap G_{il} = \emptyset,$$

где G_i — множество допустимых решений для i -й задачи; G_{ik}, G_{il} — множества допустимых решений для подзадач с индексами i_k, i_l , полученных в процессе ветвления i -й задачи.

Вычисление оценок на построенном дереве решений позволяет организовать целенаправленный поиск оптимального решения, значительно более эффективный по сравнению с методом перебора. Оценкой некоторого множества G называется некоторая функция $V(G)$, удовлетворяющая следующим свойствам:

$$\begin{aligned} \text{при поиске максимума} \quad & V(G) \leq \max_x f(x), \quad x \in G; \\ & V(G_1) \leq V(G), \quad G_1 \subset G; \end{aligned} \quad (5.2)$$

$$\begin{aligned} \text{при поиске минимума} \quad & V(G) \geq \min_x f(x), \quad x \in G; \\ & V(G_1) \geq V(G), \quad G_1 \subset G. \end{aligned} \quad (5.3)$$

Правило определения оценок дает способ вычисления $V(G)$. Оценка $V(G)$ при условии

$$\max_x f(x) = V(G), \quad (\min_x f(x) = V(G)), \quad x \in G$$

называется точной, в противном случае — грубой. Чаще всего при решении практических задач методом ветвей и границ используются грубые оценки, так как они значительно проще вычисляются. Однако следует учитывать, что чем точнее оценка, тем за меньшее число шагов в среднем заканчивается поиск оптимального решения. Выбор алгоритма построения оценок, во-первых, неоднозначен, а во-вторых, определяется спецификой задачи.

Для раскрытия более подробной структуры общего алгоритма ветвей и границ будем для определенности предполагать, что требуется найти

$$\min_x f(x).$$

Введем некоторый промежуточный одномерный массив переменной, размерность которого $B^{(K)}$ зависит от номера шага (K). Массив содержит значения оценочных функций на подмножествах решений, соответствующих висячим вершинам текущего дерева вариантов.

На первом шаге вычисляем оценку $V(G^{(0)})$ на полном множестве допустимых решений $G^{(0)}$, а далее в соответствии с выбранными принципами ветвления ветвим множество $G^{(0)}$, т. е.

$$G^{(0)} = \bigcup_{i_1} G_{i_1}^{(1)}$$

и находим

$$\min_{i_1} V(G_{i_1}^{(1)}) = V(G_{r_1}^{(1)}).$$

Все оценочные функции $V(G_{i_1}^{(1)})$ для $i_1 \neq r_1$ заносим в массив, производим предварительное упорядочение элементов массива с точки зрения \min т. е.

$$B^{(1)}(1) = \min_j B^{(1)}(j) \text{ и } B^{(1)}(j-1) \leq B^{(1)}(j).$$

На втором шаге производим дополнительное ветвление подмножества $G_{r_1}^{(1)}$

$$G_{r_1}^{(1)} = \bigcup_{i_2} G_{r_1, i_2}^{(2)}$$

вычисляем $V G_{r_1, i_2}^{(2)}$ и находим

$$\min V(G_{r_1, i_2}^{(2)}) = V(G_{r_1, r_2}^{(2)}).$$

Если

$$V(G_{r_1, r_2}^{(2)}) \leq B^{(1)}(1),$$

то оценки на подмножествах $G_{r_1, r_2}^{(2)}$ при $i_2 \neq r_2$ заносятся в массив $B^{(2)}$

и в дальнейшем на третьем шаге производится ветвление подмножества $G_{r_1, i_2}^{(2)}$. Если

$$V(G_{r_1, r_2}^{(2)}) > B^{(1)}(1),$$

то все оценки на подмножествах $G_{r_1, i_2}^{(2)}$ заносятся в массив $B^{(2)}$ и на

третьем шаге производится ветвление подмножества из массива $B^{(1)}$ с индексом 1, причем из массива оно вычеркивается. Производим вновь переупорядочение элементов массива $B^{(2)}$ с учетом старых и вновь включенных элементов.

Переходим к третьему шагу. Многошаговый процесс ветвления и вычисления оценок продолжается до тех пор, пока рассматриваемое на очередном шаге подмножество, соответствующее висячей вершине дерева вариантов, не будет одноэлементным и на нем не будет достигнута нижняя граница оценочной функции по всему множеству висячих (неразветвленных) вершин, полученных к данному шагу алгоритма. Решение X , соответ-

ствующее данному подмножеству, будет оптимальным. Аналогично может быть сформулирован алгоритм решения задач максимизации.

Рассмотрим на конкретных примерах принципы построения оценочных функций и их использования в общем алгоритме ветвей и границ.

5.2. МЕТОД ВЕТВЕЙ И ГРАНИЦ В ЗАДАЧАХ «О КОММИВОЯЖЕРЕ» И «О НАЗНАЧЕНИИ»

Задачи о “коммивояжере” и о “назначении” широко распространены при построении математических оптимизационных моделей различных процессов контроля и управления.

Задача о коммивояжере

Имеется n городов, задана матрица $T = (t_{ij})$ – матрица расстояний между городами ($t_{ii} = \infty$, $i = 1, 2, \dots, n$). Найти маршрут посещения городов коммивояжером, имеющий минимальную длину, при условии, что в каждом городе он побывает только один раз (исключая только начальное место, из которого начинается маршрут).

Задача о назначении

Имеется последовательность n работ и n исполнителей. Задана матрица $T = (t_{ij})$, определяющая время, необходимое для выполнения i -й работы j -м исполнителем ($i = 1, 2, \dots, n$; $j = 1, 2, \dots, n$). Найти оптимальное распределение работ между исполнителями, гарантирующее минимальное суммарное время их выполнения, при условии, что каждый исполнитель может выполнять только одну единственную работу.

Хотя принципы ветвления в указанных задачах различаются, при построении оценочных функций используется один и тот же принцип.

Рассмотрим его на примере алгоритма решения задачи «о коммивояжере» методом ветвей и границ.

Шаг 1. Находим в каждой строке матрицы T минимальный элемент и вычитаем его из всех элементов этой строки. Если в получающейся матрице окажутся столбцы, не содержащие нуля, то в каждом из них находим минимальный элемент и вычитаем его из всех элементов этого столбца. В результате получаем матрицу $T^{(1)}$, каждая строка и каждый столбец которой содержат по меньшей мере один нуль.

Шаг 2. Суммируем все элементы, которые вычитали на шаге 1. Пусть сумма всех элементов равна \mathbf{h} . Очевидно $V(G^{(0)}) = \mathbf{h}$, где \mathbf{h} – нижняя граница целевой функции по полному множеству решений $G^{(0)}$.

Шаг 3. Выбираем переход (k, l) , для которого

$$u(k, l) = \max_{(i, j)} \tau(i, j),$$

$$\tau(i, j) = \min_{j' \neq j} t_{ij'}^{(1)} + \min_{j' \neq j} t_{i'j}^{(1)} \quad (5.4)$$

сумма наименьшего элемента i -й строки (исключая $t_{i,j}^{(1)}$) и наименьшего элемента j -го столбца (исключая $t_{i,j}^{(1)}$), вычисляемая для элемента (i, j) с $t_{ij}^{(1)} = 0$.

Шаг 4. В соответствии с выбранным переходом (k, l) , любое множество G разбивается на два подмножества: $G_{k,l}$ — подмножество всех циклов, содержащих переход (k, l) и $G_{\overline{k},\overline{l}}$ — подмножество всех циклов с запретом на переход (k, l) . Переход (k, l) выбран на шаге 3 таким образом, чтобы подмножество $G_{k,l}$ с наибольшей вероятностью содержало оптимальный маршрут, а подмножество $G_{\overline{k},\overline{l}}$ не содержало его, т. е. чтобы пути, входящие в подмножество $G_{\overline{k},\overline{l}}$, были как можно более длинные.

Шаг 5. Вычисляем $V(G_{\overline{k},\overline{l}}) = V(G) + u(k, l)$.

Шаг 6. Вычертим из матрицы, соответствующей данному шагу алгоритма (матрицы $T^{(1)} = (t_{ij}^{(1)})$) k -ую строку и l -й столбец и заменим на ∞ значения тех элементов в ней, используя которые, можно получить контуры длиной меньшей n , где n — размерность матрицы T .

Шаг 7. Действуем так же, как и на шаге 1, с матрицей, получающейся в результате шага 6.

Шаг 8. Действуем аналогично шагу 2 с матрицей, получающейся в результате шага 7. Прибавляя полученную сумму $\mathbf{h}^{(1)}$ к оценке $V(G)$

для предыдущей вершины, находим значение оценки для подмножества первого ветвления $G_{k,l}$: $V(G_{k,l}) = V(G^{(0)}) + h^{(1)}$.

Шаг 9. Если в результате шага 6 получаем матрицу порядка **1**, то процесс заканчивается. В противном случае переходим к шагу **10**.

Шаг 10. Среди висячих вершин уже построенного дерева выбираем вершину с наименьшим значением оценочной функции (если таковых несколько, выбираем любую из них).

Шаг 11. Если выбранная на шаге **10** вершина соответствует $G_{..., s, u}$, то переходим к шагу 3. В противном случае переходим к шагу **12**.

Шаг 12. Значение элемента (s, u) матрицы, соответствующей висячей вершине, выбранной на шаге **10**, заменяем на ∞ . В s -й строке, а также в u -м столбце находим наименьший элемент и вычитаем его из всех элементов этой строки (столбца).

Затем переходим к шагу 3.

Пример 5.1

Пусть матрица $T = (t_{ij})$ для $n = 5$ представлена в табл. 5.1. Найдем оптимальный маршрут в соответствии с алгоритмом, изложенным выше.

Таблица 5.1

$i \backslash j$	1	2	3	4	5
1	∞	1	4	7	4
2	1	∞	3	7	5
3	4	3	∞	5	6
4	7	7	5	∞	6
5	4	5	6	6	∞

Шаг 1. Приведем матрицу T к виду $T^{(1)}$, представленному в табл. 5.2.

Шаг 2. $h = 1 + 1 + 3 + 5 + 4 + 2 + 1 = 17$; $V(G^{(0)}) = 17$.

Шаг 3. Рассмотрим все нулевые элементы в матрице $T^{(1)}$ (табл. 5.2). Индексы нулевых элементов:

$$(1,2); (2,1); (3,2); (3,4); (4,3); (4,5); (5,1); (5,4).$$

В соответствии с формулой (5.4):

$$\tau(1,2) = \tau(2,1) = \tau(4,3) = \tau(4,5) = 2; \tau(3,2) = \tau(3,4) = \tau(5,1) = \tau(5,4) = 0$$

Шаг 4. $G^{(0)} = G_{1,2} \cup G_{\overline{1,2}}$ (ветвление первого уровня).

Шаг 5. $V(G_{\overline{1,2}}) = V(G^{(0)}) + u(1,2) = 17 + 2 = 19$. Заносим в массив $B^{(1)}$.

Таблица 5.2

$i \backslash j$	1	2	3	4	5	
1	∞	0	3	4	2	1
2	0	∞	2	4	3	1
3	1	0	∞	0	2	3
4	2	2	0	∞	0	5
5	0	1	2	0	∞	4
	0	0	0	2	1	∞

Шаг 6. Вычерткнем первую строку и второй столбец, элемент $t_{2,1}^{(1)}$ примем равным ∞ для исключения цикла (1,2,1); преобразованная матрица представлена в табл. 5.3

Таблица 5.3

	1	3	4	5
2	∞	2	4	3
3	1	∞	0	2
4	2	0	∞	0
5	0	2	0	∞

Таблица 5.4

	1	3	4	5	
2	∞	0	2	1	2
3	1	∞	0	2	0
4	2	0	∞	0	0
5	0	2	0	∞	0
	0	0	0	0	∞

Шаг 7. Преобразуем полученную матрицу (табл. 5.3), так как первая строка ее не содержит нулевых элементов; после преобразования получим матрицу, представленную в табл. 5.4.

Шаг 8. Сумма всех элементов $h^{(1)} = 2$, следовательно,

$$V(G_{1,2}) = V(G^{(0)}) + h^{(1)} = 17 + 2 = 19 \text{ заносим в массив } B^{(1)}.$$

Шаг 9. Размерность полученной на шаге 7 матрицы не равна 1, поэтому переходим к шагу 10.

Шаг 10. Так как в данном шаге список висячих вершин и их оценок включает всего две вершины и оценки их равны $V(G_{1,2}) = V(G_{\overline{1,2}})$, то выбор вершины для последующего ветвления безразличен. Выберем $G_{1,2}$, оценку $V(G_{\overline{1,2}})$ оставим в массиве $B^{(1)}, B^{(1)} = (19)$ (массив $B^{(1)}$ — массив первого уровня ветвления).

Шаг 11. Так как выбрано для ветвления подмножество $G_{1,2}$, то переходим к шагу 3 с целью определения перехода для ветвления в матрице (табл. 5.4).

Дальнейший ход алгоритма является повторением шагов 1–2 на последующих уровнях ветвлений. На втором уровне ветвлений подмножества $G_{1,2}$ выбирается переход (2,3), а соответствующая матрица после необходимых преобразований представлена в табл. 5.5.

Таблица 5.5

	1	4	5
3	∞	0	2
4	2	∞	0
5	0	0	∞

Таблица 5.6

	1	4
3	∞	0
5	0	∞

На третьем, завершающем уровне ветвлений подмножества $G_{(1,2),(2,3)}$ выбирается переход (4,5), а соответствующая преобразованная матрица представлена в табл. 5.6. Оптимальный маршрут включает переходы: (1,2); (2,3); (3,4); (4,5); (5,1). Дерево решений, соответствующее данному примеру, показано на рис. 5.1.

5.3. МЕТОД ВЕТВЕЙ И ГРАНИЦ В ЗАДАЧАХ РАСПРЕДЕЛЕНИЯ РЕСУРСОВ

Рассмотрим особенности реализации алгоритма ветвей и границ применительно к распределению множества последовательно решаемых задач между узлами обработки информации в автоматизированной системе управления (АСУ) с учетом стоимостных и временных ресурсов. Задача

ны множество задач, решаемых в АСУ $i = (\overline{1}, \overline{I})$, и множество узлов системы управления ($j = (\overline{1}, \overline{J})$). Необходимо так распределить задачи по узлам системы, чтобы достигнуть максимальной эффективности распределения, не выходя из области допустимых ограничений. При этом каждая задача решается только в одном узле.

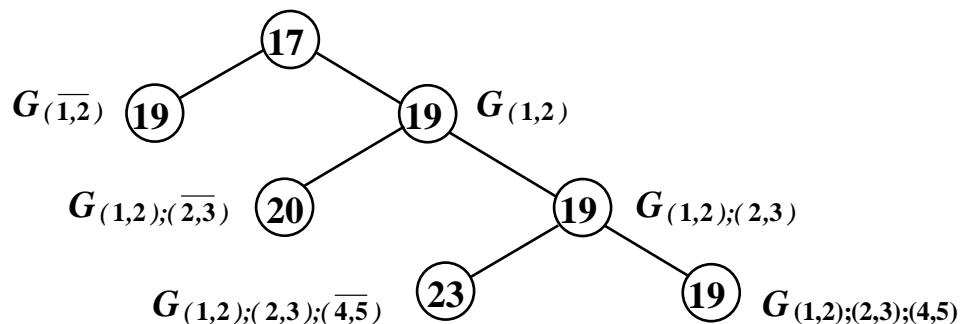


Рис.5.1

Пусть C_{ij} — затраты на реализацию i -й задачи в j -м узле системы; t_{ij} — время решения задачи в j -м узле. В качестве критерия эффективности выберем общие стоимостные затраты; ограничение накладывается на общее время решения задач. Общая математическая постановка задачи выглядит следующим образом: найти

$$\min \sum_{i=1}^I \sum_{j=1}^J C_{ij} x_{ij} \quad \text{при} \quad \sum_{i=1}^I \sum_{j=1}^J t_{ij} x_{ij} \leq T_0; \quad \sum_{j=1}^J x_{ij} = 1, i \in I. \quad (5.5)$$

Переменная $x_{ij} = 1$, если i -я задача решается в j -м узле, $x_{ij} = 0$ в противном случае. Принцип ветвления в данной задаче выглядит следующим образом. Подмножества $G_{j_1}^{(1)}$ первого уровня ветвления формируем, фиксируя соответствие первой задачи различным узлам:

$$G^{(1)} = [G_1^{(1)}, G_2^{(1)}, \dots, G_{j_1}^{(1)}, \dots, G_J^{(1)}].$$

Подмножество $G_{j_1}^{(1)}$ включает все варианты распределения задач по узлам, где первая задача решается в узле j_1 , а распределение остальных задач по узлам произвольное. Аналогично подмножества второго уровня

$G_{j_1, j_2}^{(2)}$ формируем, фиксируя соответствие второй задачи различным узлам:

$$G^{(2)} = [G_{j_1, 1}^{(2)}, G_{j_1, 2}^{(2)}, \dots, G_{j_1, j_2}^{(2)}, \dots, G_{j_1, J}^{(2)}].$$

Подмножество $G_{j_1, j_2}^{(2)}$ включает все варианты распределения, где первая задача решается в узле j_1 , вторая — в узле j_2 , а остальные задачи имеют произвольное распределение по узлам. Аналогичным образом принцип ветвления распространяется на последующие уровни. Для каждого из подмножеств необходимо при использовании метода ветвей и границ построить оценки целевой функции и ограничения. Оценка целевой функции для подмножества вариантов $G_{j_1, j_2, \dots, j_l}^{(l)}$ выглядит следующим образом:

$$V_C(G_{j_1, j_2, \dots, j_l}^{(l)}) = \sum_{i \leq l} t_{ij_i} + \sum_{i > l} \min_j C_{ij}, \quad (5.6)$$

а общее выражение для оценки функции ограничения может быть построено аналогично:

$$V_t(G_{j_1, j_2, \dots, j_l}^{(l)}) = \sum_{i \leq l} t_{ij_i} + \sum_{i > l} \min_j t_{ij}, \quad (5.7)$$

где j_1, j_2, \dots, j_l — множество узлов системы, закрепленных за соответствующими задачами.

Оценка для функции ограничения необходима в данном случае для исключения из процесса ветвления подмножеств заведомо неподходящих вариантов с учетом принятого ограничения на общее время решения задач. При решении такого класса задач для сокращения времени поиска целесообразно сделать следующие преобразования матриц $C = (c_{ij})$ и $T = (t_{ij})$.

1. Так как каждая задача может решаться только в одном узле системы, то

$$\min \sum_{i=1}^I \sum_{j=1}^J c_{ij} x_{ij} = \sum_{i=1}^I \min_{X_{ij}} \sum_{j=1}^J c_{ij} x_{ij}.$$

Используя это условие и принимая во внимание ограничение во времени $\sum_{i=1}^I \sum_{j=1}^J t_{ij} x_{ij} \leq T_0$, можно исключить из матриц C и T элементы, не влияющие на выбор оптимального решения. Такими элементами в каждой

строке являются элементы, удовлетворяющие одновременно двум следующим условиям:

$$C_{ij} > \min_j C_{if} = C_{if}; \quad t_{ij} > t_{if}. \quad (5.8)$$

2. Принимая во внимание ограничение по времени, можно из матриц C и T также исключить элементы, при которых всегда будет нарушено ограничение по времени. Для этого представим ограничение по времени в следующем виде:

$$\sum_{i=1}^{r-1} \min_j t_{ij} + t_{r,j} + \sum_{i=r+1}^I \min_j t_{ij} \leq T_0. \quad (5.9)$$

Изменяя индекс строки $r = (\overline{1, I})$ и просматривая все элементы этой строки в соответствии с условием (5.9), можно вычеркнуть из матриц C и T все элементы, для которых это условие не выполняется. Рассмотрим алгоритм решения задачи на примере.

Пример 5.2

Пусть задана матрица стоимостных затрат C в некоторых условных единицах стоимости (табл.5.7) и матрица временных затрат T в некоторых единицах времени (табл.5.8); ограничение по времени $T_0 = 20$, число задач равно 5, число узлов АСУ равно 4.

Таблица 5.7

3	7	2	2
4	8	1	1
5	9	6	2
6	10	7	1
7	5	3	1

$C =$

Таблица 5.8

1,5	3	2	9
2	6	5	10
3	7	6	11
4	8	7	12
4	9	8	5

$T =$

Шаг 1. Проведем исключение элементов в соответствии с условием (5.8).

Матрицы $C^{(1)}$ и $T^{(1)}$ после исключения представлены в табл. 5.9, 5.10.

Таблица 5.9

3	-	2	2
4	-	1	-
5	9	6	2
6	10	7	1
7	-	-	1

$C^{(1)} =$

Таблица 5.10

1,5	-	2	9
2	-	5	-
3	7	6	11
4	8	7	12
4	-	-	5

$T^{(1)} =$

Шаг 2. Проведем исключение элементов в соответствии с условием (5.9), преобразованные матрицы $C^{(II)}$ и $T^{(II)}$ представлены в табл. 5.11, 5.12.

Таблица 5.11				Таблица 5.12			
$C^{(II)} =$				$T^{(II)} =$			
3	—	2	—	1,5	—	2	—
4	—	1	—	2	—	5	—
5	9	6	—	3	7	6	—
6	10	7	—	4	8	7	—
7	—	—	1	4	—	—	5

Шаг 3. Вычислим оценки для подмножеств первого уровня ветвления в соответствии с формулами (5.6) и (5.7):

$$G^{(1)} = [G_1^{(1)}, G_3^{(1)}],$$

$$V_c(G_1^{(1)}) = 3 + 1 + 5 + 6 + 1 = 16; \quad V_t(G_1^{(1)}) = 1,5 + 2 + 3 + 4 + 4 = 14,5;$$

$$V_c(G_3^{(1)}) = 2 + 1 + 6 + 5 + 1 = 15; \quad V_t(G_3^{(1)}) = 2 + 2 + 3 + 4 + 4 = 15.$$

Так как $V_c(G_3^{(1)}) < V(G_1^{(1)})$ и $V_t(G_3^{(1)}) < 20$, выбираем для ветвления на втором уровне подмножество $G_3^{(1)}$ и заносим в список $B^{(1)}$ оценку $V(G_3^{(1)})$, т. е. $B^{(1)} = 16$. Общее дерево решений представлено на рис. 5.2. Итак, имеем следующее оптимальное распределение задач по узлам: (1,3; 2,3; 3,1; 4,1; 5,4).

Очень часто задача распределения ресурсов формализуется в виде следующей задачи дискретного программирования:

$$\begin{aligned} \text{найти} \quad & \max \sum_{j=1}^n c_j x_j \\ \text{при} \quad & \sum_{j=1}^n a_{ij} x_j \leq b_i, \quad i = 1, 2, \dots, m; \quad 0 \leq x_j \leq d_j, \quad j = 1, 2, \dots, n, \end{aligned} \quad (5.10)$$

где x_j — целое число ($j = 1, \dots, n$); $b_i, d_j, a_{ij} > 0$; $c_j > 0$ — целые числа.

При решении ее методом ветвей и границ для построения оценок используется оптимальное решение вышеуказанной задачи без учета требований целочисленности x_j , $j = 1, 2, \dots, n$.

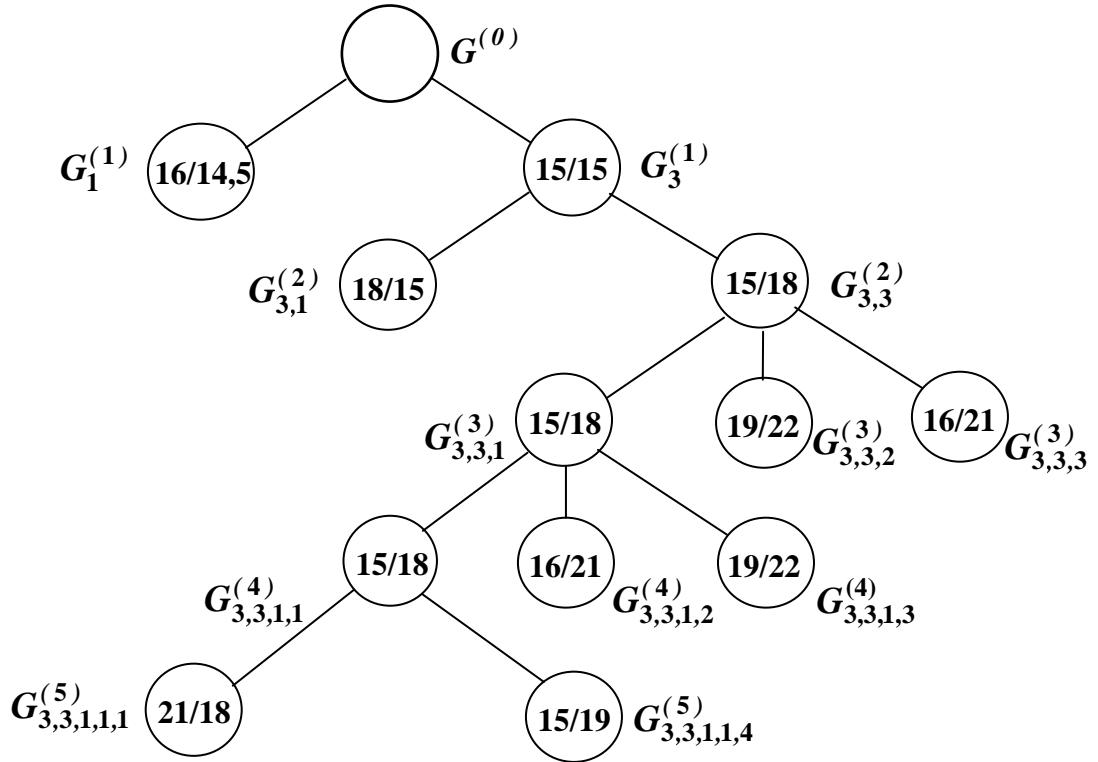


Рис. 5.2

Рассмотрим следующую задачу:

$$\begin{aligned}
 & \text{найти} \quad \max \sum_{j=1}^n c_j x_j \quad \text{при} \quad \sum_{j=1}^n a_j x_j \leq b, \\
 & \quad 0 \leq x_j \leq d_j, \quad (j = 1, \dots, n); \\
 & \quad b > 0, \quad d_j > 0, \quad c_j > 0, \quad a_j > 0.
 \end{aligned} \tag{5.11}$$

Алгоритм определения оптимального решения такой задачи вытекает из следующей теоремы. Пусть

$$\frac{c_1}{a_1} \geq \frac{c_2}{a_2} \geq \dots \geq \frac{c_j}{a_j} \geq \dots \geq \frac{c_n}{a_n}.$$

Тогда оптимальное решение $X = (x_1, x_2, \dots, x_n)$ для задачи максимизации определяется по формулам:

$$x_j = \begin{cases} d_j, & \text{если } j \leq r_0 \\ 0, & \text{если } j > r_0 + 1 \end{cases},$$

$$x_{r_0+1} = (b - \sum_{j \leq r_0} a_j d_j) / a_{r_0+1},$$

где

$$r_0 = \max \left[r / \sum_{j \leq r} a_j d_j \leq b \right].$$

Таким образом, оценка для всего подмножества решений $G^{(0)}$ в задаче (5.11) с учетом целочисленности x_j определяется как

$$V(G^{(0)}) = \sum_{j=1}^{r_0} c_j x_j + [c_{r_0+1}(b - \sum_{j \leq r_0} a_j d_j)] / a_{r_0+1}$$

Аналогичная оценка подмножества $G^{(0)}$ для задачи (5.10) определяется как

$$V(G^{(0)}) = \min_i V_i(G^{(0)}),$$

где $V_i(G^{(0)})$ — оценка, получаемая для задачи (5.11) при использовании только i -го ограничения ($i = 1, 2, \dots, m$).

Рассмотрим более подробно принцип построения и использования оценок в задачах такого класса на примере.

Пример 5.3

Найти $\max \sum_{j=1}^5 c_j x_j$ при $\sum_{j=1}^5 a_{1j} x_j \leq b_1$,

$$\sum_{j=1}^5 a_{2j} x_j \leq b_2, \quad x_j = (0, 1), \quad j = 1, 2, \dots, 5.$$

Исходные данные представлены в табл. 5.13.

Шаг 1. Упорядочим переменные $x_j (j = 1, \dots, 5)$ по критериям

$$c_j/a_{1j} \text{ и } c_j/a_{2j}: \quad x_1, x_2, x_3, x_4, x_5; \quad x_5, x_4, x_3, x_2, x_1.$$

Таблица 5.13

j	1	2	3	4	5	
c_j	15	12	12	15	10	
a_{1j}	3	3	4	5	5	
a_{2j}	5	4	3	3	2	
c_j/a_{1j}	5	4	3	3	2	
c_j/a_{2j}	3	3	4	4	5	

$$\begin{aligned} b_1 &= 13 \\ b_2 &= 11 \end{aligned}$$

Шаг 2. Построим оценку $V(G^{(0)})$:

$$V_1(G^{(0)}) = 15 + 12 + 12 + 15 \cdot 3/5 = 48,$$

$$V_2(G^{(0)}) = 10 + 15 + 12 + 12 \cdot 3/4 = 46,$$

$$V(G^{(0)}) = \min(48, 46) = 46.$$

Общее дерево ветвления показано на рис. 5.3 (\emptyset — недопустимый по ограничениям вариант).

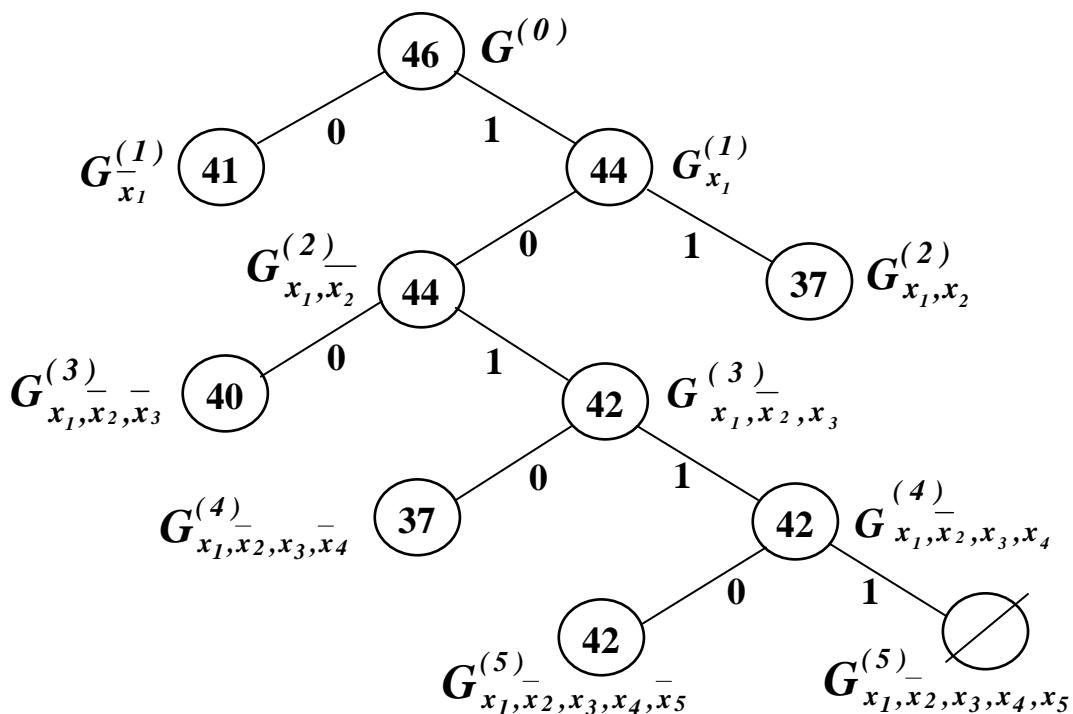


Рис.5.3

В результате получаем оптимальное решение:

$$x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 1, x_5 = 0;$$

максимальное значение целевой функции равно 42.

ЗАДАЧИ

1. Определить оптимальное расписание для контроля параметров в автоматизированной системе контроля (АСК). Проверка проводится в последовательном режиме. Время проверки t_{ij} зависит как от самого параметра t_i , так и от параметра, который предшествовал проверке данного

параметра i . Временные затраты t_{ij} связаны с необходимостью проведения подготовительных операций и являются затратами на переналадку (перенастройку). Каждый параметр контролируется только один раз. Варианты заданий для $n = 6$ и $T = (t_{ji})$ приведены в табл. 5.14 – 5.19.

Таблица 5.14

∞	31	15	19	8	55
19	∞	22	31	7	35
25	43	∞	53	57	16
5	50	49	∞	39	9
24	24	33	5	∞	14
34	26	6	3	36	∞

Таблица 5.15

∞	19	25	11	2	35
37	∞	26	58	21	43
10	50	∞	39	22	3
38	39	24	∞	38	45
27	9	32	9	∞	2
33	48	60	53	1	∞

Таблица 5.16

∞	16	13	35	41	52
19	∞	29	31	26	18
57	51	∞	44	51	7
5	40	32	∞	14	16
33	41	28	3	∞	53
19	54	24	10	41	∞

Таблица 5.17

∞	39	45	2	51	33
30	∞	20	33	40	35
54	16	∞	55	22	56
19	36	25	∞	18	43
29	8	8	12	∞	25
16	47	31	14	8	∞

Таблица 5.18

∞	41	27	54	46	5
42	∞	11	32	58	21
36	5	∞	33	22	33
46	24	59	∞	49	59
48	58	11	44	∞	47
26	50	35	19	27	∞

Таблица 5.19

∞	21	40	28	60	52
58	∞	11	39	22	56
22	12	∞	23	14	19
25	47	51	∞	20	54
47	43	18	42	∞	52
24	49	50	52	29	∞

2. Рассматривается сложная человеко-машинная система управления; n задач управления сложным объектом распределяются между n операторами, причем каждый оператор решает только одну задачу.

Эффективность решения i -й задачи j -м оператором зависит от соответствующей вероятности правильного решения. Общая эффективность

решения всех задач является аддитивной функцией эффективности отдельных операторов по решению соответствующих задач.

Распределить задачи управления между операторами таким образом, чтобы общая эффективность была максимальной. Различные варианты исходных данных представлены в табл. 5.20 – 5.23.

Таблица 5.20

3	10	5	9	16	8	17
6	8	11	8	18	19	20
7	13	10	3	4	14	18
5	9	6	21	12	17	22
5	4	11	6	13	14	11
17	7	12	13	16	17	9
13	0	8	8	10	12	17

Таблица 5.21

5	13	6	10	13	8	9
10	9	7	11	8	12	11
11	5	8	12	4	18	4
12	6	9	8	5	8	5
9	4	4	5	6	6	7
11	8	7	4	7	3	8
6	5	8	12	13	9	14

Таблица 5.22

6	5	9	10	7	12	8
9	7	11	6	8	11	10
8	10	7	8	10	7	4
5	6	10	5	6	11	12
4	9	8	9	4	1	2
5	6	10	11	10	12	5
4	11	5	4	5	12	13

Таблица 5.23

4	5	9	5	6	14	6
8	12	4	13	16	15	16
2	15	8	10	17	7	9
14	8	4	9	5	6	7
3	5	4	12	10	11	13
10	9	11	5	6	12	8
7	13	8	12	8	11	10

3. Выбрать оптимальную совокупность тестов для контроля сложного оборудования, чтобы достигалась максимальная суммарная достоверность D при условии, что общее время контроля T ограничено T_3 . Считать, что достоверность является аддитивной функцией достоверности отдельных тестов D_j . Исходные данные представлены в табл. 5.24.

Таблица 5.24

j	1	2	3	4	5	Ограничение
t_j	5	4	5	4	4	$T_3=13$
D_j	0,225	0,180	0,150	0,120	0,120	

4. Решить задачу 3 при учете ограничений не только по времени T_3 , но и по общей стоимости реализации тестов C_3 . Исходные данные представлены в табл. 5.25 – 5.27.

Таблица 5.25

j	1	2	3	4	5	Ограничение
D_j	0,225	0,180	0,180	0,225	0,150	$T_3 = 13$ $C_3 = 11$
t_j	3	3	4	5	5	
C_j	5	5	5	3	2	

Таблица 5.26

j	1	2	3	4	5	Ограничение
D_j	0,1	0,3	0,2	0,1	0,15	$T_3 = 8$ $C_3 = 13$
t_j	1	2	4	2	3	
C_j	4	5	2	6	4	

Таблица 5.27

j	1	2	3	4	5	Ограничение
D_j	0,2	0,25	0,25	0,1	0,1	$T_3 = 5$ $C_3 = 9$
t_j	1	3	2	1	3	
C_j	4	3	2	3	2	

5. Система состоит из n элементов. Определить максимальное приращение вероятности безотказной работы ΔP при возможном использовании m способов введения избыточности для каждого из рассматриваемых элементов системы. Считать, что $\Delta P \cong \sum_{j=1}^n \Delta P(x_j)$, где $\Delta P(x_j)$ — приращение вероятности безотказной работы j -го элемента при использовании способа введения избыточности x_j ($x_j = \overline{0, m}$; $j = \overline{1, n}$). Учесть ограничения по стоимости C_3 . Данные приведены в табл. 5.28.

6. Решить задачу 5 для системы из четырех элементов при ограничении $C_3 = 5$. Избыточность, используемая при повышении надежности

элементов, — одного типа, $X_j = (\mathbf{0}, \mathbf{1})$. Остальные исходные данные представлены в табл. 5.29.

Таблица 5.28

x_i	$C_j(x_i)$	$P(x_i)$		
		1	2	3
0	0	0	0	0
1	5	0,02	0,05	0,04
2	10	0,04	0,06	0,07
3	14	0,05	0,07	0,09
$n = 3, C_3 = 15$				

Таблица 5.29

j	1	2	3	4
P_j	0,006	0,003	0,008	0,001
C_j	1	3	2	2
$n = 4, C_3 = 5$				

7. Рассматривается вычислительная система, состоящая из n вычислительных машин. Имеется n задач. Задана матрица $T = (t_{ij})$, определяющая время решения i -й задачи на j -й машине. Задачи решаются одновременно, начиная с некоторого момента t_0 . Найти такое распределение задач по вычислительным машинам, чтобы общее время решения всех задач было бы минимальным при условии, что на одной машине может решаться только одна задача.

Предложить способ построения оценочной функции. Данные по матрице T приведены в табл. 5.20 – 5.23.

8. Составить оптимальное расписание для работы трех одинаковых центральных процессоров при выполнении M заданий. Каждое задание может быть выполнено на любом процессоре и если задание погружено в процессор, оно находится в нем до полного завершения (т. е. задания не могут прерываться или разделяться между двумя или более процессорами). Время реализации задания t_i ($i = 1, 2, \dots, M$). Для любого линейного порядка заданий следующее задание из списка выполняется на первом освободившемся процессоре. Задания могут быть выполнены в любом порядке. Пусть $M = 4; t_1 = 3; t_2 = 3; t_3 = 3; t_4 = 6$.

Построить алгоритм ветвей и границ для поиска оптимального расписания.

9. В оперативной памяти ЭВМ должна одновременно храниться информация по N программам для организации мультипрограммной работы. Размер выделяемой зоны для каждой из программ может быть выбран любым из имеющихся K вариантов.

Пусть $N = K = 4$. Соответствие программ и вариантов выделяемых зон задается матрицей A . Столбцы матрицы A соответствуют программам, строки – вариантам выбора зон. При каждом варианте распределения возможно недоиспользование зоны памяти. Каждый вариант характеризуется средним размером недоиспользования памяти, который зависит как от объема программы, так и от варианта и задан таблицей B .

Найти такой вариант статического размещения программ в оперативной памяти ЭВМ, чтобы общий объем недоиспользования памяти был минимальным при условии, что не будет превышен общий объем памяти S_0 . Исходные данные по трем вариантам приведены в табл. 5.30 – 5.35.

Таблица 5.30

	1	2	3	4
1	30	20	10	40
2	35	25	12	45
3	42	28	15	47
4	45	35	17	50
$S_0=115$				

Таблица 5.31

	1	2	3	4
1	2	3	1	3
2	3	2	2	3
3	1	3	1	2
4	2	4	3	1

Таблица 5.32

	1	2	3	4
1	20	50	10	30
2	25	20	15	35
3	40	27	30	50
4	50	30	35	55
$S_0=130$				

Таблица 5.33

	1	2	3	4
1	2	1	3	3
2	5	3	1	4
3	2	2	1	3
4	1	4	2	2

Таблица 5.34

	1	2	3	4
1	40	15	40	50
2	45	20	50	65
3	60	30	60	70
4	80	35	65	80
$S_0 = 180$				

Таблица 5.35

	1	2	3	4
1	3	5	3	5
2	4	1	4	4
3	2	2	4	1
4	1	5	3	4

6. ЗАДАЧИ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

6.1. СОДЕРЖАНИЕ МЕТОДА ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

Динамическое программирование представляет особый математический метод оптимизации решений, специально приспособленный к многошаговым или многоэтапным операциям. Некоторые из таких операций распадаются на отдельные шаги естественным образом, в других разбиение приходится вводить искусственно для того, чтобы их можно было решить методом динамического программирования.

Метод динамического программирования позволяет одну задачу со многими переменными заменять рядом последовательно решаемых задач с меньшим числом переменных. При этом нумерация шагов, как правило, осуществляется от конца к началу. Основным принципом оптимизации многошагового процесса, а также вычислительного метода динамического программирования является принцип оптимальности Р. Беллмана.

В соответствии с данным принципом оптимальное поведение обладает тем свойством, что, каковы бы не были начальное состояние и начальное поведение, последующие решения должны обеспечивать оптимальное поведение относительно уже достигнутого состояния.

Принцип оптимальности имеет конструктивный характер и непосредственно указывает процедуру нахождения оптимального решения. Математически при поиске максимума он записывается в виде следующего уравнения:

$$f_{N-l}(S(N-l)) = \max_{U(N-l)} \{R_l(S(N-l), U(N-l)) + f_{N-l+1}(S(N-l+1))\}, \quad l = \overline{1, N} \quad (6.1)$$

где $S(N-l) = \{S_1(N-l), S_2(N-l), \dots, S_n(N-l)\}$ — состояние системы (процесса) на l -м шаге;

$R_l(S(N-l), U(N-l))$ — непосредственный эффект, достигаемый на l -м шаге;

$U(N-l) = \{U_1(N-l), U_2(N-l), \dots, U_m(N-l)\}$ — решение (управление), выбранное на l -м шаге;

f_{N-l} — условно-оптимальное значение эффекта, достигаемое на l -м шаге;
 N — общее количество шагов (этапов);

Нумерация шагов идет с конца ($l = 1, 2, \dots, N$).

Аналогичное (6.1) соотношение записывается при поиске минимума. Все вычисления, дающие возможность найти оптимальное значение эффекта, достигаемое за N шагов, проводятся по формуле (6.1), которая носит название основного (функционального) уравнения или рекуррентного соотношения Беллмана.

Процесс вычисления значений функции f_{N-l} , $l=1,2,\dots,N$ осуществляется при естественном начальном условии $f_N(S_N) = 0$, которое означает, что за пределами конечного состояния системы эффект равен нулю.

В исследовании операций метод динамического программирования имеет весьма разнообразные приложения. Из них целесообразно выделить:

- 1) поиск максимальных и минимальных путей на графах;
- 2) поиск оптимальных траекторий движения различных объектов;
- 3) задачи надежности, контроля и технической диагностики;
- 4) задачи распределения ресурсов и др.

Рассмотрим примеры конкретных задач исследования операций и особенности использования метода динамического программирования.

6.2. Задачи оптимизации функций на графах

Будем считать, что задан ориентированный граф $G(V)$, не имеющий контуров, где V — множество вершин графа. Вершины графа пронумерованы в соответствии с введенной порядковой функцией на графике.

Целью введения порядковой функции на графике является разбиение множества вершин графа на непересекающиеся подмножества, упорядоченные так, что если какая-то вершина входит в подмножество с номером i , то следующая за ней в графике вершина — в подмножество с номером большим, чем i . Полученные непересекающиеся подмножества называются уровнями. Для построения алгоритма упорядочения используется множественное представление структуры графа, для чего вводятся понятия:

- множества прямого соответствия $G(i)$, включающего все соседние вершины, в которые можно попасть из i -ой вершины;
- множества обратного соответствия $G^{-1}(i)$, включающего все соседние вершины, из которых можно попасть в i -ю вершину.

Алгоритм упорядочения сводится к следующему.

1. В подмножество первого уровня Ω_1 включаются все вершины, у которых \emptyset (пустое множество); проводится последовательная нумерация вершин: $1, 2, \dots, l$.

2. В подмножество второго уровня Ω_2 включаются все вершины I , у которых $G^{-1}(i) \subset \Omega_1$; проводится последовательная нумерация вершин: $l+1, l+2, \dots, l+r$.

3. В подмножество третьего уровня Ω_3 включаются все вершины I , у которых $G^{-1}(i) \subset (\Omega_1 \cup \Omega_2)$; проводится последовательная нумерация вершин: $l+r+1, l+r+2, \dots, l+r+p$.

Данный процесс продолжается до тех пор, пока не будут пронумерованы все вершины в графе.

Формализация многих экстремальных задач на графах требует задания числовой функции. Числовая функция на графике считается заданной, если каждой дуге (или вершине) ставится в соответствие число $q(i, j)$ (или $l(i)$), где i, j — начальная и конечная вершины дуги (i, j) . В некоторых случаях числовая функция задается комбинированным способом как на вершинах, так и на дугах. Многие экстремальные задачи на графах сводятся к определению максимального (минимального) пути на графике при аддитивном

$$q_S = \sum_{(i,j) \in S} q(i,j) \text{ или } l_s = \sum_{i \in S} l_i \quad (6.2)$$

или мультипликативном

$$q_S = \prod_{(i,j) \in S} q(i,j) \text{ или } l_s = \prod_{i \in S} l_i \quad (6.3)$$

способах задания числовой функции на путях S , $S \in S_{\text{don}}$, где S_{don} — множество допустимых путей (например, соединяющих две заданные вершины).

При аддитивном способе задания числовой функции на графике функциональное уравнение для определения максимального пути, аналогичное (6.1), выглядит следующим образом:

$$q_{N-l}^{\max}(j, B) = \max_{i \in G(j)} \left[q_{N-l+1}^{\max}(i, B) + q(j, i) \right], \quad (6.4)$$

где B — конечная вершина пути S (начало — вершина A).

При мультипликативном способе задания числовой функции на графике функциональное уравнение для определения максимального пути записывается так:

$$q_{N-l}^{\max}(j, B) = \max_{i \in G(j)} \left[q_{N-l+1}^{\max}(i, B) \times q(j, i) \right], \quad (6.5)$$

Число шагов N при использовании уравнений (6.4) и (6.5) определяется числом уровней M , определяемым при введении порядковой функции на графе ($N = M - 1$).

Аналогично решаются задачи и при определении минимальных путей на графах.

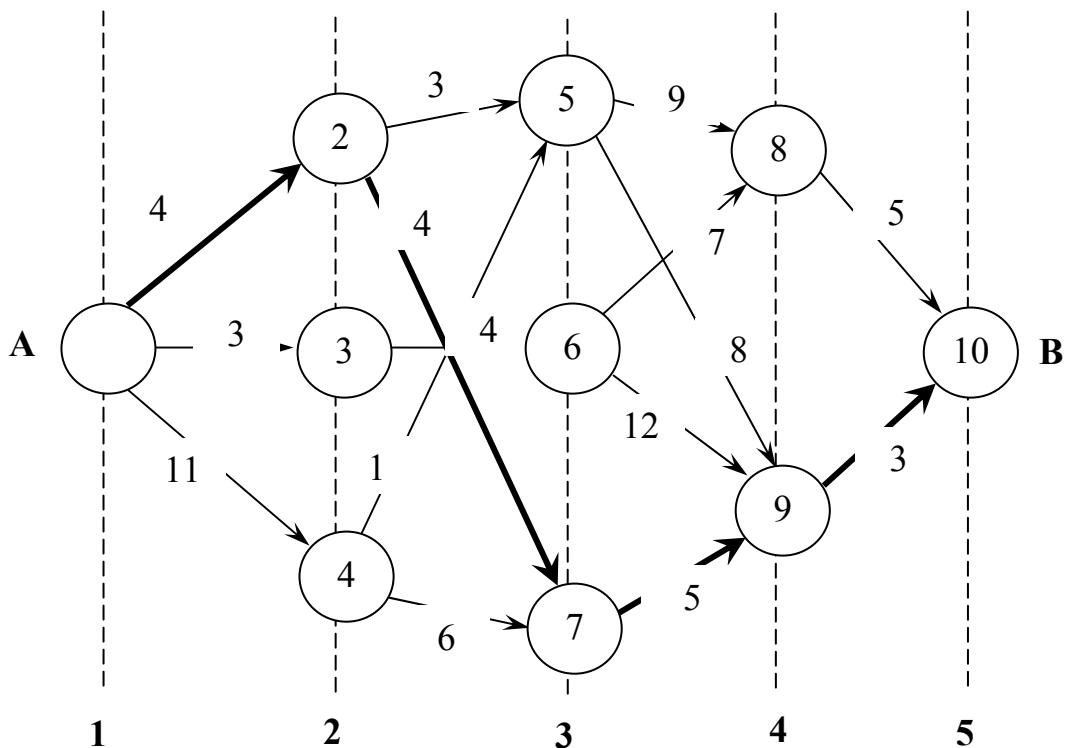


Рис. 6.1

Пример 6.1

Дан граф (рис. 6.1). Найти минимальный путь в графе в соответствии с заданием числовой функции, представленной на рис. 6.1. Структура графа содержит пять уровней.

Шаг 1. $q_N^{\min}(B, B) = 0, N = 4;$

$$q_{N-1}^{\min}(8, 10) = q(8, 10) = 5;$$

$$q_{N-1}^{\min}(9, 10) = q(9, 10) = 3.$$

Шаг 2.

$$q_{N-2}^{\min}(5,10) = \min \left\{ \begin{array}{l} [q_{N-1}^{\min}(8,10) + q(5,8)], \\ [q_{N-1}^{\min}(9,10) + q(5,9)] \end{array} \right\} = 11,$$

$$q_{N-2}^{\min}(6,10) = \min \left\{ \begin{array}{l} [q_{N-1}^{\min}(8,10) + q(6,8)], \\ [q_{N-1}^{\min}(9,10) + q(6,9)] \end{array} \right\} = 12,$$

$$q_{N-2}^{\min}(7,10) = 8.$$

На последующих шагах (3,4) повторяется описанная выше система действия, вытекающая из уравнения (6.4), для множеств вершин второго и первого уровней.

Минимальный маршрут показан на рисунке 6.1, минимальная длительность пути из вершины A в B равна 16.

6.3. Динамическое программирование в задачах поиска неисправностей

Рассмотрим вопросы построения функциональных уравнений в задачах поиска неисправностей и синтеза оптимальных систем тестов.

Задача формулируется следующим образом: найти неисправность в агрегате, состоящем из M элементов. Будем считать, что неисправен ровно один элемент, причем априорная вероятность того, что это именно i -й элемент, известна и равна P_i , $\sum_{i=1}^M P_i = 1$. Для выявления неисправности

используются диагностические тесты. Число тестов равно L . Каждому тесту ставится в соответствие затраты $c(j)$ на использование теста. Требуется построить диагностическую процедуру выявления неисправного элемента, минимизирующую математическое ожидание затрат.

Пусть на некотором l -м шаге рассматривается множество возможных элементов $R(l)$, среди которых может находиться неисправный элемент. Тест с номером j разделяет любое множество R на два подмножества:

$$R = R'(j) \cup R''(j),$$

где $R'(j)$ объединяет все элементы, покрываемые тестом ($x_j = 1$),

$R''(j)$ — все элементы, не покрываемые тестом ($x_j = 0$).

Пусть на $(l-1)$ -м шаге или ранее определена оптимальная последовательность тестов для подмножеств $R'(j)$ и $R''(j)$ и условно оптимальные средние потери:

$$f_{l-1}[R'(j)] \text{ и } f_{l-1}[R''(j)].$$

Тогда условно оптимальным на l -м шаге следует считать тест, для которого справедливо условие:

$$f_l(R) = \min_j [c_l[j] + f_{l-1}(j)] , \quad (6.6)$$

$$\text{где } f_{l-1}(j) = P'_{l-1,j} f_{l-1,j}[R'(j)] + P''_{l-1,j} f_{l-1,j}[R''(j)], \quad (6.7)$$

$$P'_{l-1,j} = \frac{1}{P(R)} \sum_{i \in R'(j)} P_i; \quad P''_{l-1,j} = \frac{1}{P(R)} \sum_{i \in R''(j)} P_i.$$

При решении данной задачи методом динамического программирования в соответствии с полученным функциональным уравнением множество R последовательно пробегает ряд вариантов, число которых равно $\sum_{j=2}^M C_M^j$. Так, например, для $M = 5$ рассматриваются следующие варианты множества R : $(1, 2); (3, 1); (3, 2); (3, 2, 1); \dots; (1, 2, 3, 4, 5)$, см. табл. 6.2.

Поэтому при больших значениях L и M хранение промежуточных результатов требует значительной памяти. Очевидное равенство $\bar{f}_l(R) = 0$ выполняется для множества R , включающего всего один элемент.

Пример 6.2

Пусть $M = 5$, $L = 4$, $c_1 = 46$, $c_2 = 65$, $c_3 = 31$, $c_4 = 82$, а $p_1 = p_2 = p_3 = p_4 = p_5 = 0,2$.

Матрица соответствия тестов и результатов тестирования различных элементов приведена в таблице 6.1.

Таблица 6.1

$i \backslash j$	1	2	3	4
S_1	0	0	1	1
S_2	1	0	1	1
S_3	1	0	0	0
S_4	1	1	1	0
S_5	1	1	1	1

В дальнейшем, так как шаг будет однозначно определяться R , будем записывать $\bar{f}_l(R) = \bar{f}(R)$. Начальные данные для алгоритма

$$\bar{f}(1) = \bar{f}(2) = \bar{f}(3) = \bar{f}(4) = \bar{f}(5) = 0.$$

Шаг 1. $\bar{f}(1,2) = 46$, условно оптимален тест 1.

Шаг 2. $\bar{f}(3,1) = 31$, условно оптимален тест 3.

Шаг 3. $\bar{f}(3,2) = 31$, условно оптимален тест 3.

Шаг 4. $\bar{f}(3,2,1) = [46 + \frac{0,4}{0,6} \times 31, 31 + \frac{0,4}{0,6} \times 46, 82 + \frac{0,4}{0,6} \times 46] = 61,6$.

Условно оптimalен тест 3.

Данные по последующим шагам сведены в таблицу 6.2: в первом столбце выделено множество R , во втором — условно-оптимальное значение \bar{f} , в третьем — условно-оптимальный тест. Оптимальное диагностическое дерево представлено на рис. 6.2.

Таблица 6.2

R	$\bar{f}(R)$	Опти- мальный тест	R	$\bar{f}(R)$	Опти- мальный тест
1; 2	46,0	1	5; 2; 1	89,5	1
3; 1	31,0	3	5; 3	31,0	3
3; 2	31,0	3	5; 3; 1	61,6	3
3; 2; 1	61,6	3	5; 3; 2	74,5	3
4; 1	46,0	1	5; 3; 2; 1	98,0	3
4; 2	65,0	2	5; 4	82,0	4
4; 2; 1	89,5	1	5; 4; 1	85,5	3
4; 3	31,0	3	5; 4; 2	119,5	2
4; 3; 1	61,6	3	5; 4; 2; 1	129,0	2
4; 3; 2	74,0	3	5; 4; 3	85,5	3
4; 3; 2; 1	101,5	1	5; 4; 3; 1	95,5	3
5; 1	46,0	1	5; 4; 3; 2	121,0	3
5; 2	65,0	2	5; 4; 3; 2; 1	134,0	3

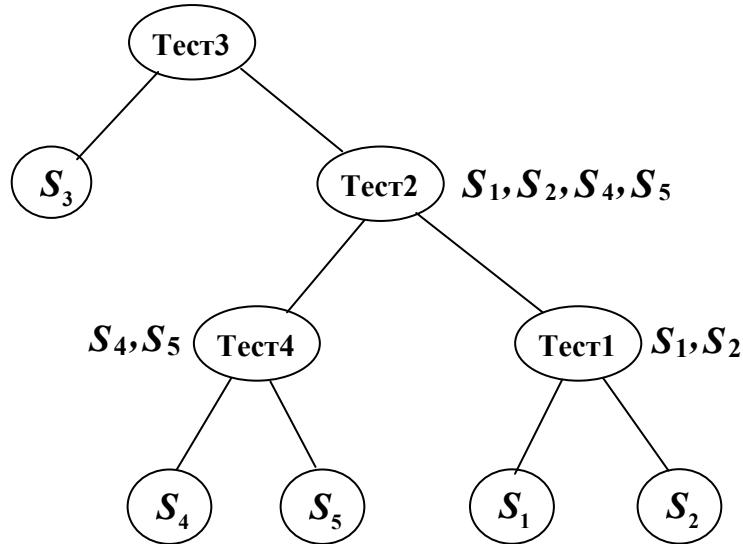


Рис. 6.2

6.4. ДИНАМИЧЕСКОЕ ПРОГРАММИРОВАНИЕ В ЗАДАЧАХ РАСПРЕДЕЛЕНИЯ РЕСУРСОВ

Рассмотрим принцип составления функциональных уравнений для таких случаев на примере задачи с экономической интерпретацией – распределение средств на расширение производства. Пусть группе из n предприятий выделены дополнительные средства на реконструкцию и модернизацию производства. По каждому предприятию известен возможный прирост $g_i(x)$, $i = 1, 2, \dots, n$ выпуска продукции в зависимости от выделенной ему суммы x . Требуется распределить средства « C » между предприятиями, чтобы общий прирост $f_n(c)$ выпуска продукции был максимальным.

Считая процесс оптимального распределения многошаговым, имеем:

$$\text{Шаг 1. } f_1(c) = \max_{0 \leq x \leq c} g_1(x).$$

$$\text{Шаг 2. } f_2(c) = \max_{0 \leq x \leq c} [g_2(x) + f_1(c - x)].$$

Аналогично можно вычислить значение $f_3(c)$, если известны значения $f_2(c)$ и т. д.

Общий вид функционального уравнения Беллмана для данной задачи выглядит следующим образом:

$$\cdot f_n(c) = \max_{0 \leq x \leq c} [g_n(x) + f_{n-1}(c - x)] \quad (6.8)$$

Поясним методику использования данного функционального уравнения на примере.

Пример 6.3

Пусть $n = 4$, $c = 100$ тыс. руб. Функции $g_i(x)$ $i=1,4$ приведены в табл. 6.3. Составить план распределения средств, максимизирующий общий прирост выпуска продукции. В дальнейшем для упрощения расчетов значения x будем принимать кратными 20 тыс. руб. и для большей наглядности результаты оптимизации оформлять в виде таблиц.

Таблица 6.3

c	$g_1(x)$	$g_2(x)$	$g_3(x)$	$g_4(x)$	$x_1=c$	$f_1(c)$
20	10	12	11	16	20	10
40	31	26	36	37	40	31
60	42	36	45	46	60	42
80	62	54	60	63	80	62
100	76	78	77	80	100	76

Шаг 1. ($n = 1$). Оптимальные значения $x_1(c)$ и $f_1(c)$ представлены в табл. 6.3.

Шаг 2. ($n = 2$). Предполагается, что средства вкладываются в два предприятия. Результаты оптимизации представлены в табл. 6.4.

Таблица 6.4

$c \backslash x$	0	20	40	60	80	100	$x_2(c)$	$f_2(c)$
20	0+10	12+0					20	12
40	0+31	12+10	26+0				0	31
60	0+42	12+31	26+10	36+0			20	43
80	0+62	12+42	26+31	36+10	54+0		0	62
100	0+76	12+62	26+42	36+31	54+10	78+0	100	78

Для каждого значения (20, 40, 60, 80, 100) начальной суммы с распределаемых средств в табл. 6.4 предусмотрена отдельная строка, а для каждого возможного значения x (0, 20, 40, 60, 80, 100) распределяемой суммы – столбец. Некоторые клетки таблицы останутся незаполненными, так как соответствуют недопустимым сочетаниям c и x . В каждую клетку таблицы записывается значение суммы $g_2(x) + f_1(c - x)$.

Аналогичным образом проводится процесс оптимизации на 3-м и 4-м шагах. Общие результаты оптимизации для всего многошагового процесса представлены в сводной таблице (табл. 6.5). При $C = 100$ тыс. руб. оптимальное распределение средств между предприятиями следующее: 4 – 40; 3–40; 2–20; 1–0. При таком распределении максимальный прирост выпуска продукции 85 тыс. руб.

Таблица 6.5

$c \backslash x$	$x_1(c)$	$f_1(c)$	$x_2(c)$	$f_2(c)$	$x_3(c)$	$f_3(c)$	$x_4(c)$	$f_4(c)$
0	0	0	0	0	0	0	0	0
20	20	10	20	12	0	12	20	16
40	40	31	0	31	40	36	40	37
60	60	42	20	43	40	48	20	52
80	80	62	0	62	40	67	40	73
100	100	76	100	78	40	79	40	85

Легко обобщить основные принципы использования динамического программирования в задачах распределения ресурсов, представленные в данном примере, на случай нескольких ограничений.

Рассмотрим задачу выбора и обоснования кратности резервирования при повышении надежности элементов, устройств, систем. Устройство состоит из n элементов, вероятность отказа элемента q_j , $j = 1, \dots, n$. Необходимо так выбрать кратности резервирования элементов k_j при ограничениях по стоимости C и весу W , чтобы вероятность безотказной работы устройства в течение заданного времени была максимальной. Формальная математическая постановка данной задачи сводится к следующему:

найти $\max P = \prod_{j=1}^n (1 - q_j^{1+k_j})$; (6.9)

$$\sum_{j=1}^n k_j c_j \leq C; \quad \sum_{j=1}^n k_j w_j \leq W; \quad k_j — \text{целые}, \quad k_j \geq 0.$$

Функциональное уравнение Беллмана для данного случая выглядит следующим образом:

$$P_l = \max \left[(1 - q_l)^{1+k_l} P_{l-1}(w - k_l w_l, c - c_l k_l) \right] \quad (6.10)$$

при условиях:

$$w - k_l w_l \geq 0, \quad c - c_l k_l \geq 0, \quad P_0(w, c) = 1.$$

Используя уравнение (6.10), можно последовательно определить оптимальные кратности резервирования в задаче 6.9.

ЗАДАЧИ

1. Определить кратчайший маршрут передачи данных между пунктами А и В при наличии графа сети линий связи, представленного на рис. 6.3 – 6.7.

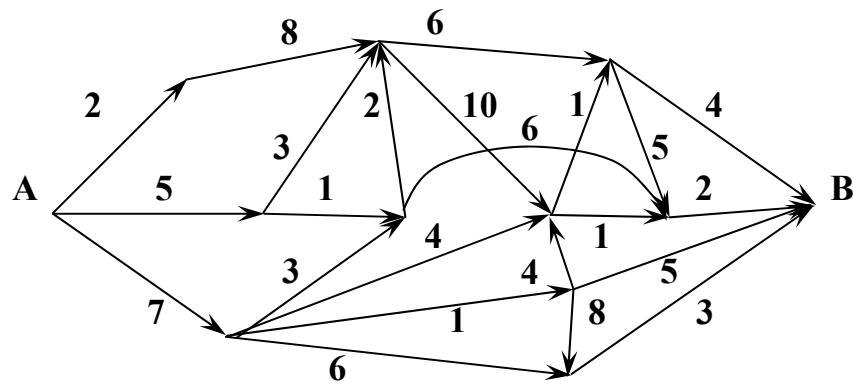


Рис. 6.3

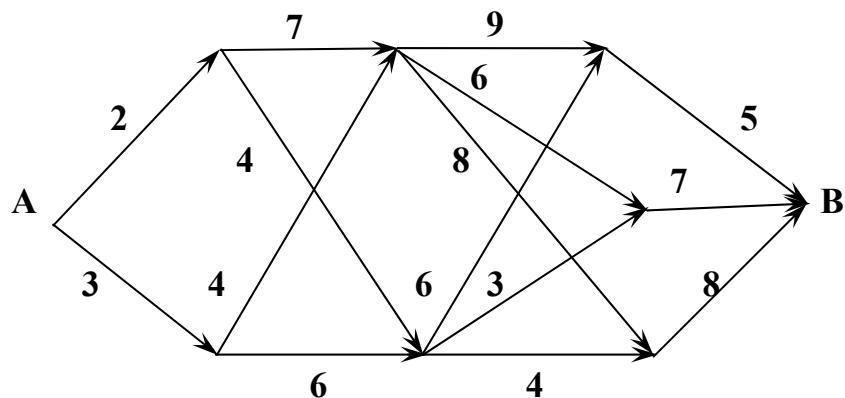


Рис. 6.4

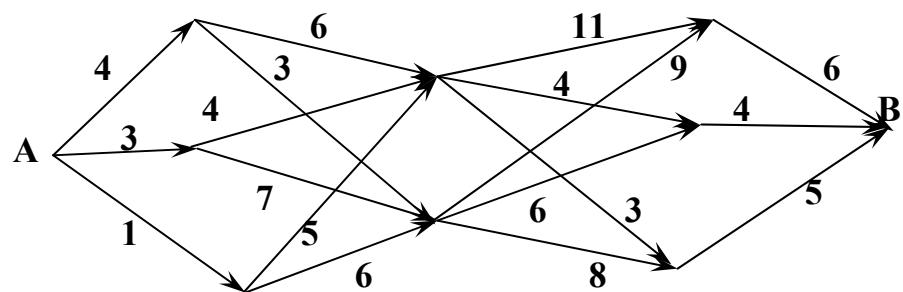


Рис. 6.5

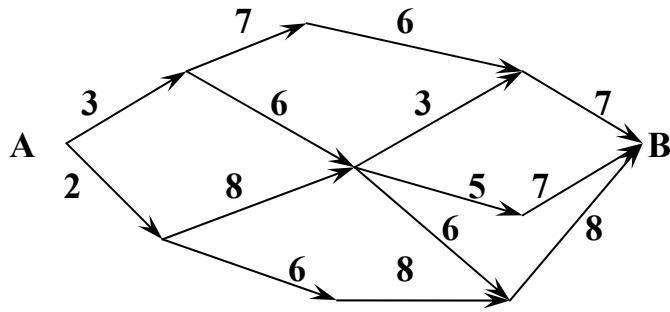


Рис. 6.6

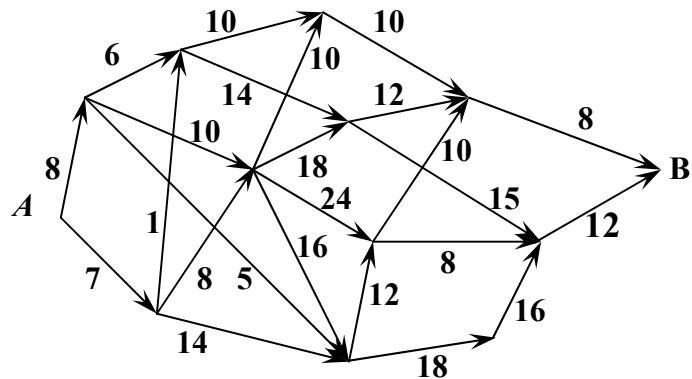


Рис. 6.7

2. Самолет (или другой летательный аппарат), находящийся на высоте H_0 и имеющий скорость V_0 , должен быть поднят на заданную высоту H_k , а скорость его доведена до заданного значения V_k . Известен расход горючего, потребный для подъема аппарата с любой высоты H_1 на любую другую $H_2 > H_1$ при неизменной скорости V ; известен также расход горючего, требуемый для увеличения скорости от любого значения V_1 до любого другого $V_2 > V_1$ при неизменной высоте H .

Требуется найти оптимальный режим набора высоты и скорости, при котором общий расход горючего будет минимальным. Решение задачи провести при следующем допущении: весь процесс набора высоты и скорости разделен на ряд последовательных этапов, и за каждый шаг самолет увеличивает только высоту или только скорость.

Исходные данные для решения приведены на рис.6.8. Составить функциональное уравнение.

3. Решить задачу 2 при наличии возможности диагональных переходов, указанных на рис. 6.8. Составить функциональное уравнение.

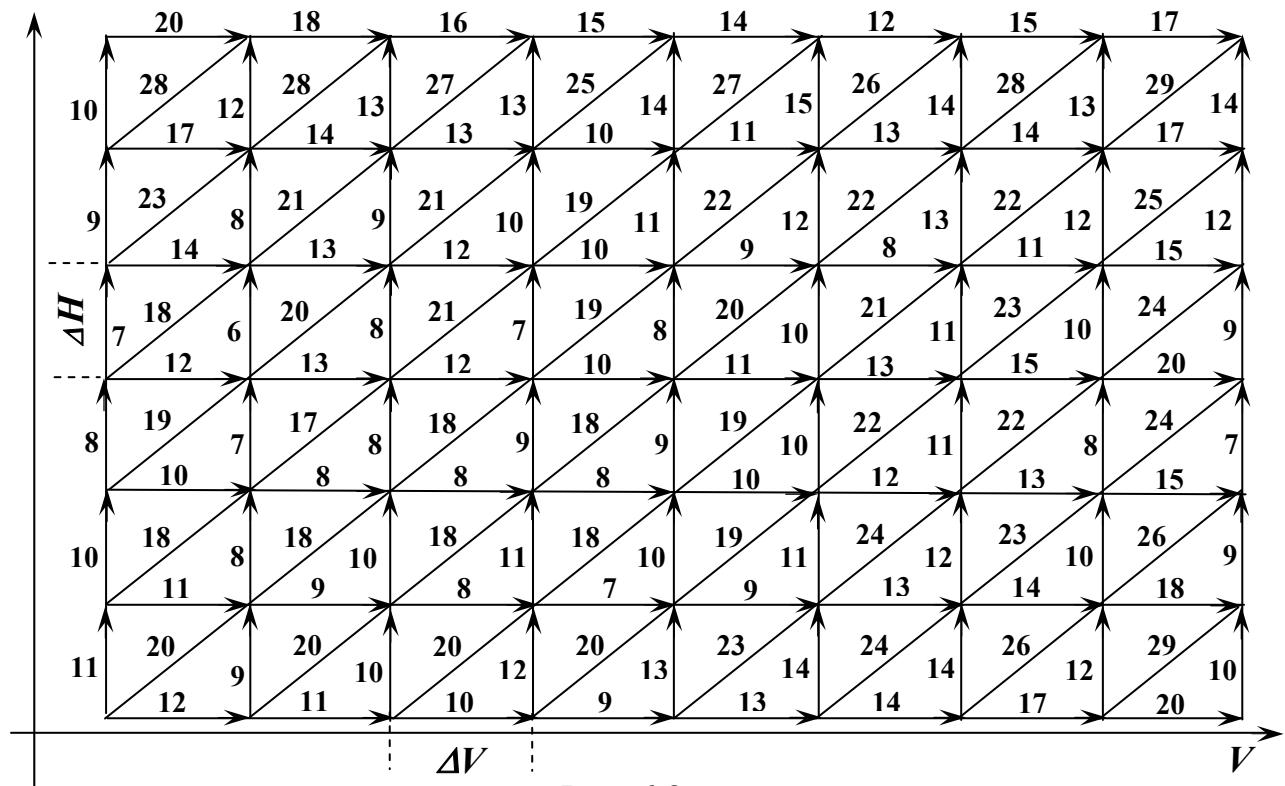


Рис. 6.8

4. Задана последовательность выполнения операций представленная графом на рис. 6.9. Дугам графа сопоставлены длительности выполнения операций. Все операции, соответствующие дугам графа, выходящим из вершины, могут быть начаты только при условии завершения всех операций, сопоставленным дугам графа, входящим в вершину. Найти общую длительность выполнения операций, маршрут выполнения и временные резервы при выполнении отдельных операций.

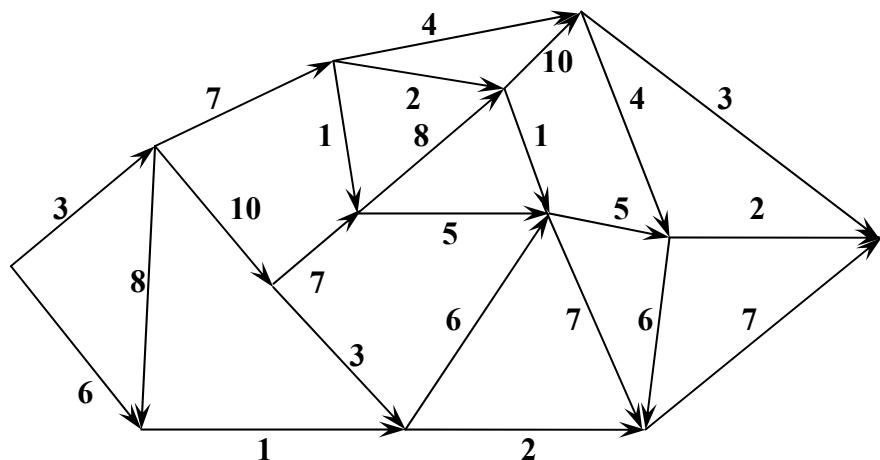


Рис. 6.9

5. Задан граф сети передачи данных (рис. 6.10). Вершины графа – пункты передачи, дуги – каналы. P_{ij} – вероятность правильной передачи сообщения от пункта i к пункту j . Найти путь передачи сообщения с максимальной и минимальной вероятностями правильной передачи от пункта A до пункта B .

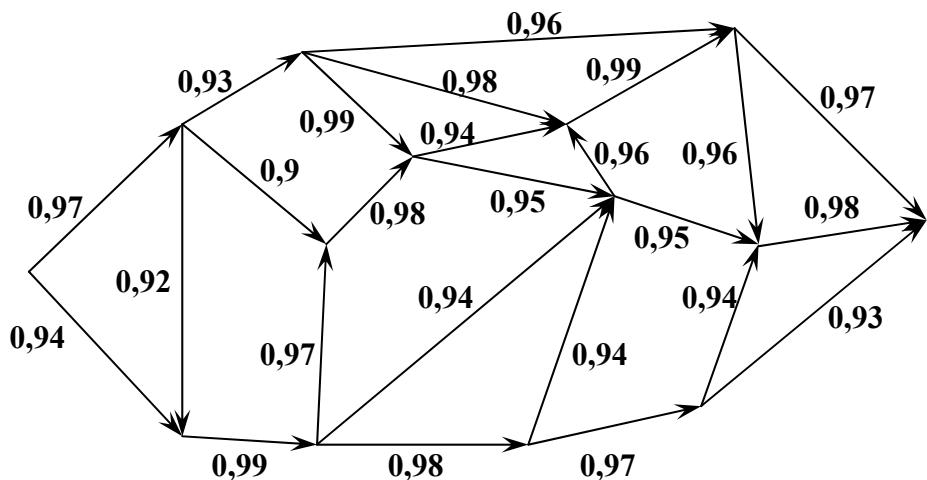


Рис. 6.10

6. Выбрать оптимальную систему тестов в соответствии с содержанием примера 6.2. Варианты исходных данных представлены в табл. 6.6 – 6.9.

Таблица 6.6

$i \backslash j$	Тесты				
	1	2	3	4	P_i
S_1	1	0	1	1	0,1
S_2	0	1	0	0	0,2
S_3	1	0	1	0	0,3
S_4	1	0	0	0	0,1
S_5	0	1	1	1	0,3
C_j	20	15	30	45	

Таблица 6.7

$i \backslash j$	Тесты				
	1	2	3	4	P_i
S_1	0	0	1	1	0,3
S_2	0	1	1	0	0,2
S_3	1	1	0	0	0,3
S_4	0	1	1	1	0,2
C_j	10	15	15	10	

Таблица 6.8

$j \backslash i$	Т е с т ы				
j	1	2	3	4	P_i
S_1	1	0	1	0	0,3
S_2	1	1	1	0	0,2
S_3	1	0	0	0	0,1
S_4	0	0	1	1	0,4
C_j	20	25	5	10	

Таблица 6.9

$j \backslash i$	Т е с т ы				
j	1	2	3	4	P_i
S_1	0	0	1	1	0,15
S_2	0	1	1	1	0,15
S_3	1	0	0	1	0,2
S_4	0	0	0	1	0,3
S_5	1	1	1	0	0,2
C_j	10	15	15	10	

Таблица 6.10

Пред- приятие	x , тыс. руб.	20	40	60	80	100	Вариант
1	$g1(x)$	9	18	24	38	50	1
		9	17	29	38	47	2
		7	29	37	41	59	3
		9	20	35	44	57	4
		9	18	29	41	60	5
		11	21	40	54	62	6
2	$g2(x)$	11	19	30	44	59	1
		11	34	46	53	75	2
		9	19	28	37	46	3
		12	25	34	46	57	4
		8	19	30	47	58	5
		13	20	42	45	61	6
3	$g3(x)$	16	32	40	57	70	1
		13	28	37	49	61	2
		17	27	37	48	66	3
		11	20	32	48	61	4
		12	25	51	58	69	5
		12	22	34	55	60	6
4	$g4(x)$	13	27	44	69	73	1
		12	35	40	54	73	2
		16	30	42	65	81	3
		14	23	40	50	58	4
		7	15	52	59	60	5
		10	27	33	57	69	6

7. Найти оптимальный план распределения средств (100 тыс. руб.) на расширение производства между предприятиями ($n = 4$). Различные варианты исходных данных представлены в табл. 6.10. (см. пример 6.3).

8. Одной из функций программы-диспетчера многопрограммной ЭВМ является управление вводом в оперативную память (ОЗУ) программ и подпрограмм решаемых задач. По мере выполнения требуемых расчетов часть программ, записанных в ОЗУ, может заменяться новыми. Однако нередко возникают ситуации, когда в ходе решения больших задач, включающих в себя много подпрограмм, объем освобождаемой памяти постепенно сокращается за счет накопления в ОЗУ промежуточных и служебной информации. Ввод в ОЗУ новых порций программной информации производится диспетчером через фиксированные интервалы времени – циклы. Производительность ЭВМ зависит от объема хранящейся в ОЗУ программной информации, а также от специфики решаемых задач, которая влияет прежде всего на интенсивность обращений к внешним устройствам. Пусть ЭВМ должна решать две большие задачи. Ее средняя производительность во время цикла определяется выражением:

$$F = \frac{1}{2} [g_1(x) + g_2(y)] = \frac{1}{2} (1 - e^{-x} + 1 - e^{-y}),$$

где x, y — доли общего объема ОЗУ, выделенные для программ первой и второй задач в начале цикла.

К концу цикла часть этой памяти $f_1 = 0,5x$ (для 1-й задачи), $f_2 = 0,2y$ (для 2-й задачи) освобождается и может перераспределяться для новых позиций программ. Остальная же часть $x - f_1(x) = 0,5x$ (для 1-й задачи) и $y - f_2(y) = 0,8y$ (для 2-й задачи) остается занятой до конца решения обеих задач. Будем считать, что на любом цикле в ОЗУ могут вводиться порции программ разного объема, т. е. освободившаяся часть памяти может использоваться полностью. Необходимо найти такое распределение памяти объемом $S_0 = 10$ условных единиц между обеими задачами, которое обеспечит максимальную среднюю производительность в течение четырех циклов.

9. Выбрать оптимальную кратность резервирования трех элементов системы, соединенных последовательно в смысле надежности. Величины q_j, c_j, w_j определяют вероятность отказа, стоимость и вес j -го элемента-

системы. Заданы ограничения по стоимости $\sum_{j=1}^3 c_j \leq 4$ и по весу

$$\sum_{j=1}^3 w_j \leq 14. \text{ Остальные данные приводятся в табл. 6.11.}$$

Таблица 6.11

<i>j</i>	1	2	3
<i>q_j</i>	0,1	0,2	0,3
<i>w_j</i>	3	5	4
<i>c_j</i>	1	1	1

Таблица 6.12

<i>j</i>	1	2	3	4
<i>c_j</i>	2	3	1,5	1
<i>q_j</i>	0,2	0,3	0,2	0,3

Таблица 6.13

<i>j</i>	1	2	3	4
<i>c_j</i>	1	2	3	1
<i>q_j</i>	0,4	0,3	0,2	0,5
<i>K_j</i>	3	2	2	4

10. Решить задачу 9 при минимизации стоимостных затрат и при ограничениях на требуемую вероятность безотказной работы $P \geq 0,99$. Ограничение по весу не учитывается. Исходные данные по элементам представлены в табл. 6.12.

11. Решить задачу 9 при ограничениях на затраты $\sum_{j=1}^4 c_j \leq 10$ и на

предельное количество K_j резервных подсистем каждого типа. Данные по элементам представлены в табл. 6.13.

12. Решить задачи 3 – 5 из главы 5 методом динамического программирования.

13. Имеется начальное количество средств K_0 , которое нужно распределить в течение m лет между двумя отраслями производства 1 и 2. Средства x , вложенные в i -ю отрасль, приносят доход $f_i(x)$, однако уменьшаются при этом до $g_i(x) < x$. По истечении года оставшиеся средства заново распределяются между отраслями. Средства извне не поступают, и в производство вкладываются все оставшиеся в наличии средства; доход не вкладывается, а накапливается отдельно. Требуется найти способ управления ресурсами, при котором суммарный доход обеих отраслей за m лет будет максимальным.

Решить задачу в следующих случаях:

a) $k_0 = 10, m = 5, f_1(x) = x^2, g_1(x) = 0,75x,$

$$f_2(x) = 2x^2, g_2(x) = 0,3x;$$

$$\text{б) } k_0 = 2, \ m = 5, \ f_1(x) = 1 - e^{-x}, \ g_1(x) = 0,75x, \\ f_2(x) = 1 - e^{-2x}, \ g_2(x) = 0,3x.$$

14. Известно, что промышленность может выпускать n различных типов станков. Пусть $b_k, k=1, \dots, n$ — потребность в каждом из этих типов станков. Станок типа k может выполнять работу станка типа $k+1$ и всех последующих типов. Задана функция стоимости $f_k(m)$ изготовления m станков типа k . Требуется определить, какие типы станков и в каком количестве необходимо выпускать, чтобы удовлетворить заданной потребности при наименьшей сумме затрат на производство.

а) Решить задачу при

$$n = 4, \ b_k = k, \ f_k(m) = a_k m^{0,85},$$

$$a_1 = 1900, \ a_2 = 1700, \ a_3 = 1300, \ a_4 = 1100;$$

б) Найти решение задачи, если функции стоимости

$$f_k(m) = a_k m \quad \text{— линейные.}$$

15. Рассмотрим электронную систему, состоящую из четырех блоков, каждый из которых обязательно должен функционировать для работы всей системы. Надежность системы может быть улучшена путем включения в блоки параллельно работающих компонент. В таблице 6.14 приведены вероятности функционирования блоков, если они содержат 1;2 или 3 параллельно включенные компоненты. Вероятность успешного функционирования всей системы равна произведению вероятностей надежной работы блоков. Стоимость включения 1, 2 или 3 параллельных компонент в блоки приведена в табл. 6.15. На создание всей системы выделено не более 100 руб. Определить необходимое число компонент в каждом из четырех блоков, для максимизации успешной работы всей системы.

Таблица 6.14

Число компонент	Вероятность функционирования блока			
	1	2	3	4
1	0,7	0,5	0,7	0,6
2	0,8	0,7	0,9	0,7
3	0,9	0,8	0,95	0,9

Таблица 6.15

Число компонент	Стоимость блока			
	1	2	3	4
1	10	20	10	20
2	20	40	30	30
3	30	50	40	40

16. Пусть имеются два месторождения полезных ископаемых A , B , запасы которых соответственно равны x и y . Для добычи ископаемых используется одна машина, которая либо с определенной вероятностью добывает часть запаса, либо выходит из строя и в дальнейшем не используется. Если машина работает на месторождении A , то с вероятностью p_1 она добывает часть r_1 имеющегося запаса и с вероятностью $(1 - p_1)$ выходит из строя. Если машина работает на месторождении B , то она добывает часть r_2 имеющегося запаса с вероятностью p_2 и с вероятностью $(1 - p_2)$ выходит из строя. В какой последовательности следует использовать машину на месторождениях, чтобы общее количество полезных ископаемых добытых до выхода машины из строя было максимальным. Рассмотреть трехэтапный процесс, если

$$x = 400; \quad y = 200; \quad p_1 = 0,7; \quad r_1 = 0,6; \quad p_2 = 0,8; \quad r_2 = 0,8.$$

7. РЕШЕНИЕ ЗАДАЧ ОПТИМИЗАЦИИ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕКИ OPTIM СИСТЕМЫ MATLAB-5

Среди множества интегрированных систем автоматизации математических и научных расчетов наиболее приспособленной и удобной для решения задач оптимизации (а также многих других задач) является система MATLAB.

Это связано не только с наличием специальной библиотеки MATLAB APPLICATION TOOLBOX OPTIM, включающей функции оптимизации, достижения цели и аппроксимации кривых (основанные на решении оптимационных задач), но и с простотой написания собственных функций со своими алгоритмами оптимизации, с использованием готовых функций для наглядного графического представления результатов и созданием новых моделей для исследования оптимационных задач с современной организацией интерфейса (графического интерфейса пользователя с системой меню и подменю и т. п.).

Кроме того, имеется возможность интегрирования функций MATLAB и функций, написанных на языке Си. Язык MATLAB является смесью языка Фортрана с Си и вместе с системой MATLAB постоянно совершенствуется.

В состав системы MATLAB 5 входит операционная среда с множеством интерфейсов, позволяющих осуществлять связь с внешним миром, современный редактор/отладчик, рабочая область с возможностью просмотра, редактор активных путей и др.

В MATLAB 5 добавлены возможность работы со структурами, многомерными массивами, массивами записей и ячеек, объектно-ориентированное программирование, новые средства визуализации и многое другое, познакомиться подробнее с новыми возможностями системы в целом и языка можно в соответствующей литературе.

В данном разделе рассматриваются только вопросы, связанные с применением системы MATLAB для решения задач оптимизации: основные функции библиотеки оптимизации, их применение, даются примеры написания собственных систем моделирования для решения задач оптимизации с использованием библиотечных функций.

Предполагается, что основы работы в среде MATLAB и ее базовые возможности (такие как, система команд, программирование и т. п.) уже известны или будут известны из предварительного ознакомления с литературой [15, 16, 17, 18, 20].

7.1. СОСТАВ И НАЗНАЧЕНИЕ ФУНКЦИЙ БИБЛИОТЕКИ ОРТИМ

Состав и назначение функций библиотек-расширений (MATLAB APPLICATION TOOLBOX) можно найти в файлах contents.m, обычно расположенных в соответствующих каталогах (например, для библиотеки оптимизации OPTIM это каталог C:\MATLAB\TOOLBOX\OPTIM).

В табл. 7.1 дана краткая характеристика основных функций (синтаксис и соответствующие математические модели задач оптимизации). Формат обращения к той или иной функции (задаваемые аргументы и возвращаемые значения) в значительной степени зависит от класса конкретной решаемой задачи (например, наличие или отсутствие ограничений), требуемой точности решения и детализации результатов оптимизации.

Таблица 7.1

Тип оптимизации	Математическая модель	Синтаксис
Оптимизация скалярных функций	$\min_a f(a), a_1 < a < a_2$	$a = fmin('f', a_1, a_2)$
Линейное программирование	$\min_x c^T x \text{ при } Ax \leq b$	$x = lp(c, A, b)$
Квадратичное программирование	$\min_x \left(\frac{1}{2} x^T H x + c^T x \right),$ при $Ax \leq b$	$x = qp(H, c, A, b)$
Безусловная оптимизация	$\min_x f(x)$	$x = fminu('f', x_0)$ $x = fmins('f', x_0)$
Условная оптимизация	$\min_x f(x)$ при $G(x) \leq 0$	$x = constr('fG', x_0)$
Достижение цели (многокритериальная оптимизация)	$\min_x \varphi$ при $F(x) - w\varphi \leq goal$	$X =$ $attgoal('F', x, goal, w)$
Задача минимакса	$\min_x \max_{\{F_i\}} \{F_i(x)\},$ при $G(x) \leq 0$	$x =$ $minimax('FG', x_0)$
Задача минимизации при наличии параметрических ограничений	$\min_x f(x) \text{ при } G(x) \leq 0$ и $K(x, w) \leq 0 \text{ для всех } w$	$x =$ $semiinf('fgk', n, x_0)$

7.2. ДЕМОНСТРАЦИОННЫЕ ПРИМЕРЫ

Как и в большинстве библиотек-расширений системы MATLAB в пакете OPTIM имеются демонстрационные примеры, ориентированные на иллюстрацию возможностей готовых функций, и, кроме того, имеется обучение (tutorial) основным приемам программирования оптимизируемых функционалов, связанных с ними ограничений, начальных условий и использованию возможностей готовых функций. Состав демонстрационных примеров:

- **optdemo** вызов меню демонстрационных примеров по оптимизации;
- **bandemo** — минимизация функции вида

$$f(x) = 100 * (x(2) - x(1)^2)^2 + (1 - x(1))^2, \text{ при } G(x) \leq 0$$

с выбором различных методов оптимизации (Banana function);

- **datdemo** — аппроксимация (**Data fitting**);
- **goaldemo** — пример использования функции достижения цели (**Goal Attainment**);
- **tutdemo** — пошаговое обучение (**Tutorial**).

Запуск демонстрации с пошаговым обучением может быть осуществлен с помощью команды demo и выбора соответствующего подменю окна MATLAB Demo Windows (TOOLBOX — OPTIMIZATION — COMMAND LINE DEMOS — TUTORIAL), командой optdemo и выбора LINE DEMOS — TUTORIAL или непосредственно командой tutdemo.

7.2.1. Задачи безусловной оптимизации (unconstrained optimization)

Функция fminu и вектор параметров оптимизации options

Рассмотрим задачу нахождения $X=[x_1, x_2]$, обеспечивающего минимум следующей функции

$$f(x) = \exp(x_1)(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1).$$

Отметим, что там, где это соответствует контексту, функциональные зависимости, выражения, значения переменных и функций приводятся в формате, принятом в среде MATLAB.

Решение задачи

1. Определение оптимизируемой функции (можно написать М-файл (примеры 7.2.5 – 7.2.6) или работать в режиме прямых вычислений в командной строке среды MATLAB (примеры 7.2.1 – 7.2.4)):

`fun = exp(x(1)) * (4 * x(1) ^ 2 + 2 * x(2) ^ 2 + 4 * x(1) * x(2) + 2 * x(2) + 1);`

2. Задание начальной точки:

$$x0 = [-1 \ 1];$$

3. Получение решения:

$$x = fminu (fun, x0).$$

После 36 шагов оптимизации будет найдено решение: $x = 0.5000 - 1.00$. Можно получить и значение функции в оптимальной точке: `fun=1.3029e-10`

Если у функции существует более чем один локальный минимум, то выбор начального приближения (начальной точки) влияет как на число шагов поиска оптимального решения, так и на значение оптимальной точки.

При использовании функции **fminu** включение в состав параметров вектора `options` позволяет дополнительно влиять на ход оптимизационного процесса и получать детальную информацию о результатах оптимизации:

$$[x, options] = fininu (fun, x0, options).$$

Если задать `options = []`, то будут использоваться параметры оптимизации по умолчанию, а результаты будут доступны в соответствующих ячейках `options`.

Назначение компонентов вектора параметров `options` и значения, установленные по умолчанию представлены в табл. 7.2.

Если при вызове функций оптимизации вектор параметров пуст, то функция **foptions** генерирует значения по умолчанию; если только некоторые компоненты не определены или равны нулю, то они устанавливаются по умолчанию.

Некоторые компоненты используются только для возврата информации о ходе выполнения оптимизации в точку вызова соответствующих функций, эти компоненты не имеют начальных значений (помечены как недоступные).

Таблица 7.2

N	Назначение	Значение по умолчанию	Примечание
1	Признак вывода на экран промежуточных результатов	0	0 — не выводить, 1 — выводить таблицу результатов, –1 — подавление предупреждающих сообщений.
2	Точность, с которой должно быть определено значение X	1.e–4	Критерий останова — требуемая точность в худшем случае для независимых переменных X , процесс оптимизации будет продолжаться до тех пор, пока не будет достигнута требуемая точность по всем критериям останова.
3	Точность, с которой должно быть определено значение функции в решении	1.e–4	Точность представления f .
4	Точность вычисления ограничений	1.e–7	Критерий останова, используемый в функциях attgoal , constr , minimax , seminf — точность приближения к ограничениям.
5	Алгоритм решения. Стратегия поиска.	0	Выбор основного алгоритма оптимизации.
6	Алгоритм решения. Оптимизатор поиска.	0	Выбор алгоритма поиска направления.

Таблица 7.2 (продолжение)

N	Назначение	Значение по умолчанию	Примечание
7	Алгоритм решения. Алгоритм определения длительности шага	0	Выбор алгоритма поиска шага
8	Значение функции в найденной оптимальной точке (Lambda при достижении цели).	Недоступно	Для функций attgoal и minimax — признак достижения цели
9	Проверка аналитического соотношения для градиента, заданного пользователем	0	1 — на первых шагах оптимизации осуществляется проверка заданных пользователем аналитических соотношений для градиентов и численных приближений (при этом должна быть определена функция градиента). 0 — проверка отключена.
10	Количество вычислений значений целевой функции и функции ограничений	Недоступно	Счетчик числа шагов (итераций)
11	Количество вычислений градиента функции	Недоступно	Число вычислений градиента функции или конечных разностей для получения значения градиента.
12	Количество вычислений функции ограничений.	Недоступно	Общее число вычислений градиента функций ограничений или конечных разностей для получения значения градиента.

Таблица 7.2 (окончание)

N	Назначение	Значение по умолчанию	Примечание
13	Количество ограничений типа равенства.	0	Число ограничений типа равенства. Ограничения типа равенства должны размещаться в первых элементах переменной g .
14	Максимальное количество итераций	$100n$	Максимальное число вычислений функции. Значение по умолчанию зависит от размерности задачи n — числа независимых переменных. Для функции fmins — значение по умолчанию $200n$, а для fmin — $500n$.
15	Число целевых функций (n), приближения которых к целевым значениям (goal) необходимо достичнуть как можно точнее	0	Используется при поиске решения в функции attgoal . Наиболее значимые целевые функции должны располагаться в первых n элементах вектора f (в общем случае возвращаемого функцией fun.m).
16	Минимальные приращения переменных для вычисления градиентов	$1e-8$	Действительное значение приращения переменных изменяется в пределах минимального и максимального значений приращений переменных.
17	Максимальные приращения переменных для вычисления градиентов	0,1	Действительное значение приращения переменных изменяется в пределах минимального и максимального значений приращений переменных.
18	Длина шага	Недоступно	На первой итерации длина шага равняется 1 или меньше, в зависимости от производной.

7.2.2. Задачи условной оптимизации с ограничениями типа \leq , \geq и без ограничений на пределы изменения X (*constrained problem*). Функция *constr*

Найти

$$\min f(x)$$

$$f(x) = \exp(x_1)(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1)$$

при наличии следующих ограничений:

$$\begin{aligned} x_1 x_2 - x_1 - x_2 &\leq -1,5 \\ -x_1 x_2 &\leq 10. \end{aligned}$$

После приведения ограничений к каноническому виду:

$$\begin{aligned} 1,5 + x_1 x_2 - x_1 - x_2 &\leq 0 \\ -x_1 x_2 - 10 &\leq 0 \end{aligned}$$

можно приступать к решению задачи.

1. Определение функции с учетом ограничений:

```
funf = 'f = exp(x(1))*(4*x(1)^2 + 2*x(2)^2 + 4*x(1)*x(2) + 2*x(2) + 1);';
fung = 'g = [1,5 + x(1)*x(2) - x(1) - x(2), -x(1)*x(2) - 10];';
fun = [funf fung];
```

2. Задание начальной точки:

$$x0 = [-1 1];$$

3. Получение решения:

$$x = \text{constr}(fun, x0).$$

После 29 шагов оптимизации будет найдено решение:

$$x = -9,5547 \quad 1,0474x.$$

Можно получить и значение функции в оптимальной точке:

$$fun = 0,0263.$$

Можно определить активность ограничений в полученной точке:

$$g = [1,5 + x(1)*x(2) - x(1) - x(2), -x(1)*x(2) - 10];$$

$$g = 1.0e-015 *$$

$$\begin{aligned} g = & \\ & 1.0e-015 * \\ & -0.8882 \\ & 0 \end{aligned}$$

7.2.3. Задачи условной оптимизации с ограничениями на область допустимых значений X (*bounded example*)

К рассмотренным выше ограничениям в п. 7. 2. 2 добавим ограничения типа

$$x_1 \geq 0,$$

$$x_2 \geq 0.$$

Тогда на третьем этапе кроме начальной точки необходимо определить следующие переменные, позволяющие учесть ограничения снизу (**vlb**) и сверху (**vub**) на значения x :

```
vlb = [0, 0]; % нижняя граница X >= 0  
vub = [ ]; % ограничений сверху нет, пустая матрица  
options = [ ]; % значения по умолчанию
```

Формат вызова функции минимизации здесь:

```
x = constr(fun, x0, options, vlb, vub).
```

Необходимо отметить, что в этом случае обязательным является аргумент **options**, который должен быть определен (можно использовать установки по умолчанию, как в примере).

Через 10 шагов оптимизации будет получено решение:

```
x =  
    1.5000
```

Аналогично предыдущим примерам можно получить значение функции в найденной точке и проверить действуют ли ограничения:

```
fun = exp(x(1))*(4*x(1)^2 + 2*x(2)^2 + 4*x(1)*x(2) + 2*x(2) + 1)  
fun =  
    8.5000
```

```
g = [1.5 + x(1)*x(2) - x(1) - x(2), - x(1)*x(2) - 10];  
g =  
    0  
   -10
```

С учетом того, что верхнего ограничения нет, можно использовать и сокращенный набор аргументов:

```
x = constr(fun, x0, options, vlb).
```

Такой способ обращения к функции минимизации целесообразен, если заданы ограничения в виде $x_i \leq U$ (или $x_i \geq L$), преобразованные к виду $x_i - U \leq 0$ (или $-x_i + L \leq 0$).

При выполнении оптимизации в функциях MATLABа производятся вычисления значений градиентов с применением различных конечных аппроксимаций. Для нахождения более точного решения и повышения эффективности вычислительных процедур можно в исходные данные включить описание градиента оптимизируемой функции и ограничений в аналитической форме:

```
dfdx1 = 'exp(x(1))*(4*x(1)^2 + 2*x(2)^2 + 4*x(1)*x(2) + 2*x(2) + 1) +
         + 4*exp(x(1))*(2*x(1) + x(2));';
dfdx2 = '4*exp(x(1))*(x(1) + x(2) + 0.5);';
% сохранено обозначение авторов Matlab
% по смыслу это dg = [dgdx1, dgdx2]
dfdg = '[x(2)-1, x(2), x(1)-1, -x(1)];';
grad = ['df = [, dfdx1, dfdx2, ]; dg = ', dfdg];
```

Обращение к функции в этом случае (в этом примере предполагается, что ограничения на X отсутствуют, а остальные переменные определены в соответствии с примерами выше):

```
[x, options] = constr(fun, x0, options, [], [], grad);
```

Через 11 шагов будет получено решение:

```
x =
    - 9.5474   1.0474
```

```
fun = exp(x(1))*(4*x(1)^2 + 2*x(2)^2 + 4*x(1)*x(2) + 2*x(2) + 1)
```

```
fun =
    0.0236
```

```
g = [1.5 + x(1)*x(2) - x(1) - x(2), -x(1)*x(2) - 10]
g =
    1.0e-014 *
        0.11110
        - 0.1776
```

Кроме того, имеется возможность включить проверку правильности задания градиента в аналитической форме. Для этого необходимо установить

```
options(9) = 1;
```

тогда на первом цикле оптимизации будет проводиться сравнение градиентов, полученных численно и с использованием заданного соотношения. При обнаружении значительного расхождения будет выдано предупреждение.

7.2.4. Задачи условной оптимизации с ограничениями типа (*equality constrained example*)

Найти

$$\min_x f(x)$$

$$f(x) = \exp(x_1)(4x_1^2 + 2x_2^2 + 4x_1x_2 + 2x_2 + 1).$$

при наличии следующих ограничений:

$$x_1x_2 - x_1 - x_2 \leq -1,5$$

$$-x_1x_2 \leq 10$$

$$x_1 + x_2 = 1.$$

Канонический вид ограничений:

$$1,5 + x_1x_2 - x_1 - x_2 \leq 0$$

$$-x_1x_2 - 10 \leq 0$$

$$x_1 + x_2 - 1 = 0.$$

1. Определение минимизируемой функции и ограничений (например, в файле fun.m):

```
function [f,g] = fun(x)
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
g(1) = x(1) + x(2) - 1; % Вначале ограничения типа равенства
g(2) = 1.5 + x(1) * x(2) - x(1) - x(2); % Затем ограничения типа
g(3) = -x(1) * x(2) - 10; % неравенств
```

2. Задание начального приближения и условий оптимизации:

```
x0 = [-1,1];
options(13) = 1; % Имеется одно ограничение типа равенства
```

3. Получение решения:

```
x = constr('fun',x0,options)
```

Через 22 шага будет получено решение:

```
x =  
-2.7016 3.7016
```

Значение функции и активность ограничений:

```
[f,g] = fun(x)  
f =  
1.6775  
g =  
- 0.0000 - 9.5000 0.0000
```

Если имеется n ограничений типа равенства, то их размещают первыми в векторе ограничений $\mathbf{g}(\mathbf{x})$ и задают **options(13) = n**.

7.2.5. Использование вектора параметров оптимизации **options**: изменение установок по умолчанию (*changing the default settings*)

Рассмотрим задачу безусловной оптимизации следующей функции:

$$f(x) = \exp(x_1) (4x_1^2 + 2x_2^2 + 4x_1 x_2 + 2x_2 + 1).$$

Решение задачи оформим в виде M-файла:

```
function f = fun(x)  
% определение функционала  
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);  
x0 = [-1,1]; % задание начального приближения  
options(2) = 1e-8; % критерий останова — точность приближения  
options(3) = 1e-8; % критерий останова — точность вычисления  
% оптимального значения fun(x)  
  
x = fminu('fun',x0,options) % вызов функции минимизации
```

После 61 шага оптимизации можно получить решение:

```
x =  
0.5000 -1.0000
```

и значение функции в оптимальной точке можно найти через заданную ранее функцию:

```
fun(x)  
ans =  
3.5145e-14
```

или через вектор параметров:

```
options(8)ns =  
3.4412e-014
```

число шагов оптимизации находится в 10-м элементе вектора options:

```
options(10)  
ans =  
61
```

Если хотим получить отчет о ходе выполнения оптимизации то, необходимо задать

```
options(1) = 1; % отображение результатов каждого шага на экране
```

Затем вызвать функцию оптимизации:

```
[x, options] = fminu (fun, x0, options);
```

Будут получены следующие результаты, включающие таблицу шагов табл. 7.3. и другие сообщения системы MATLAB.

Таблица 7.3

f-COUNT	FUNCTION	STEP-SIZE	GRAD/SD
номер значения функции	значение функции	длина шага одномерного поиска	градиент в направлении поиска
4	1.8394	1	- 0.677
9	1.72427	0.361834	- 0.00354
17	0.362233	5.90886	- 0.295
20	0.221775	3	0.436
25	0.0171251	0.529597	0.00204
30	6.31008e-005	0.25069	- 2.92e-005
36	6.21504e-008	1.37782	1.27e-009

Optimization Terminated Successfully — успешное завершение оптимизации.

Search direction less than 2*options(2) — найденное значение меньше, чем 2*options(2).

Gradient in the search direction less than 2*options(3) — градиент в найденном направлении меньше, чем 2*options(3).

NUMBER OF FUNCTION EVALUATIONS = 36 — Число шагов оптимизации = 36.

Получение оптимального решения:

```
x =  
    0.5000 - 1.0000
```

Значение функции в оптимальной точке:

```
options(8)  
ans =  
    1.3029e-010
```

Для функций **constr** и **seminf** (см. табл. 7.1) заголовок таблицы шагов оптимизации выглядят несколько иначе:

f-COUNT	FUNCTION	MAX{g}	STEP	Procedures
---------	----------	--------	------	------------

f — **COUNT FUNCTION** — те же, что и для **fminu**,
MAX{g} — максимальное нарушающее ограничение,
STEP — длина шага одномерного поиска,
Procedures — сообщения об изменениях матрицы Гессе и QR подзадач.

Для функций **attgoal** и **minimax**, заголовок таблицы шагов оптимизации выглядит так же, как для функции **constr** за исключением столбца **FUNCTION**, который отсутствует, так как **MAX{g}** дает максимальное приближение к цели для **attgoal** и максимальное значение функции для **minimax**.

Один из возможных путей задания параметров оптимизации — использование вспомогательной функции **foptions**. Функция **foptions**, вызванная без аргументов, возвращает набор параметров по умолчанию. Если ее вызвать с вектором аргументов, то она возвращает набор параметров, установленных по умолчанию за исключением тех компонент, где были установлены ненулевые значения, например:

```
options = foptions  
options =  
    0 0.0001 0.0001 0.0000 0 0 0 0 0 0 0 0 0 0.0000 0.1000 0  
  
options = foptions([0 1e-2])  
options =  
    0 0.0100 0.0001 0.0000 0 0 0 0 0 0 0 0 0 0.0000 0.1000 0
```

7.3. ОПИСАНИЕ ОСНОВНЫХ ФУНКЦИЙ ДЛЯ РЕШЕНИЯ ЗАДАЧ ОПТИМИЗАЦИИ (ПОРЯДОК СЛЕДОВАНИЯ — ПО АЛФАВИТУ)

Функция attgoal

Функция **attgoal** решает задачу достижения цели (один из вариантов формализации задачи многокритериальной оптимизации):

$$\min_x \phi, \text{ при } F(x) - w \phi \leq goal,$$

где $F(x)$ — функция возвращающая вектор значений;

w — вектор весов,

$goal$ — вектор целей,

ϕ — скалярная переменная.

Формат вызова функции:

x = attgoal('fun',x0,goal,w) — решение задачи начинается с точки **x0** для функции, определенной в М-файле **fun.m** с заданными векторами **w** и **goal**;

x = attgoal('fun',x,goal,w,options) — с подключением вектора параметров оптимизации **options** с целью изменения установок по умолчанию (см. табл. 7.2);

x = attgoal('fun',x,goal,w,options,vlb,vub) — с определением допустимых пределов для **x** посредством задания матриц **vlb** и **vub**: введение ограничения **vlb <= x <= vub**;

x = attgoal('fun',x0,goal,w,options,vlb,vub,'grad') — с аналитическим заданием градиента в М-файле **grad.m** (вызов функции **grad** для вычисления частных производных оптимизируемой функции и ограничений).

x = attgoal('fun',x0,goal,w,options,vlb,vub,'grad',p1,p2,...) — с передачей параметров **p1**, **p2** и т. д., определяющих задачу, прямо в функции **fun** и **grad**;

[x,options] = attgoal('fun',x0, ...) — с возвращением параметров используемых при оптимизации в точку вызова (назначение компонентов см. в табл. 7.2).

Аргументы функции:

fun — строка содержащая имя функции, вычисляющей целевую функцию в зависимости от значений x . Функция возвращает один аргумент — вектор значений f ,

$$f = \text{fun}(x).$$

Для определения оптимизируемой функции можно использовать строки-выражения с x :

$$x = \text{attgoal}('sin(x.^*x)', x0, goal, w).$$

goal — вектор значений, которых целевая функция пытается достигнуть.

w — весовой вектор для управления достижимостью целей.

p1, p2, ... — дополнительные аргументы, передаваемые в функцию **fun** и **grad**, посредством вызовов:

$$[f, g] = \text{fun}(x, p1, p2, \dots)$$

$$[df, dg] = \text{grad}(x, p1, p2, \dots).$$

Использование этой возможности позволяет применять один и тот же М-файл для решения ряда близких задач с различными параметрами без необходимости определения глобальных переменных. Для того, чтобы воспользоваться дополнительными аргументами необходимо при вызове функции оптимизации задать весь список аргументов, предшествующий **p1, p2**, и т. д., однако так как пустые матрицы могут использоваться в качестве аргументов, то корректным является следующий формат вызова:

$$x = \text{constr}('fun', x0, [], [], [], p1, p2, \dots)$$

для параметров **vlb**, **vub** и ‘**grad**’ будут использованы значения по умолчанию.

Функция constr

Функция **constr** ищет безусловный минимум функции нескольких переменных:

$$\min_x f(x), \text{ при } G(x) \leq 0.$$

Формат вызова и аргументы функции:

x = constr ('fun', x0) начинает поиск с точки **x0** и находит безусловный минимум функции, описанной в “**FUN**” (обычно это M-file: **fun.M**).

Функция `fun` должна возвращать два значения: скалярное значение целевой функции f и матрицу ограничений g :

$$[f, g] = \text{fun}(x).$$

Функция F минимизируется таким образом, чтобы выполнялось условие $G < \text{zeros}(G)$.

`x = constr('fun', x0, options)` — с заданием вектора управляющих параметров “options”. (см. табл. 7.2)

`x = constr('fun', x, options, vlb, vub)` — с установкой допустимых пределов для переменных x ; в этом случае решение всегда лежит в диапазоне $vlb < x < vub$.

`x= constr('fun', x, options, vlb, vub, 'gradfun')` — позволяет задать в качестве входного параметра функции `constr` функцию `gradfun`, которая возвращает частные производные целевой функции и функции ограничений в точке x :

$$[gf, gc] = \text{gradfun}(x).$$

В этом случае поиск решения будет осуществляться эффективнее и точнее.

`x = constr('fun', x, options, vlb, vub, gradfun, p1, p2,...)` — можно указать дополнительные входные параметры `pi`, которые задаются функции `fun:[f,g]=fun (x, p1, p2,...).`

Пустые аргументы `[]` для функции `constr` игнорируются.

Функция `constr` может выдавать предупреждающие сообщения вида:

'Warning: No feasible solution found.'

(Предупреждение: допустимое решение не найдено)

"Warning: Maximum number of iterations exceeded"

'increase OPTIONS(14)'

(Предупреждение: превышено максимально допустимое количество итераций: для выполнения большего числа итераций увеличьте `OPTIONS(14).`)

Функция `fmin`

Функция `fmin` минимизирует функцию одной переменной:

$$\min_x f(x), \text{ при } x_1 < x < x_2.$$

Формат вызова функции и ее аргументы:

`a = fmin('f', x1, x2)` — возвращает значение x , которое является точкой локального минимума функции $f(x)$ на интервале $x_1 < x < x_2$. ' f ' — это строка, которая является именем целевой функции.

a = fmin('f', x1, x2, options) — с использованием вектора управляющих параметров options (см. табл. 7.2).

a = fmin('f', x1, x2, options, p1, p2,...) позволяет указывать до 10 дополнительных входных параметров **p_i**, которые задаются целевой функции: **f(x, p1, p2,...)**.

[a,options] = fmin('f',a1,a2) — с возвратом параметров, используемых при оптимизации.

Пример 7.1

fmin('cos', 3, 4) вычисляет число π с точностью до нескольких знаков после запятой.

fmin('cos', 3, 4, [1,1.e-12]) выводит на экран последовательность шагов, проходимых для вычисления π с точностью до 12 знаков после запятой.

В функции **fmin** используется алгоритм одномерной минимизации, использующий на разных шагах метод золотого сечения и метод параболической интерполяции.

Функция **fmin** может выдавать предупреждающее сообщение вида:

*'Warning: Maximum number of iterations has been exceeded'
' -- increase options(14) for more iterations.'*

(Предупреждение: превышено максимально допустимое количество итераций: для выполнения большего числа итераций увеличьте **OPTIONS(14)**.)

Функция fmins

Функция **fmins** ищет минимум функции нескольких переменных:

$$\min_x f(x).$$

Форматы вызова функции и ее аргументы:

x = fmins('f', x0) — возвращает значение **x**, которое является точкой локального минимума функции **f(x)** вблизи начального вектора **x0**. **'f'** — это строка, которая является именем целевой функции. **f(x)** должна быть скалярной функцией от нескольких переменных.

x = fmins('f', x0, options) — использует вектор управляющих параметров (см. табл. 7.2).

[x,options]= fminu('fun',x0) — с возвратом параметров, используемых при оптимизации.

x=fmins('f', x1, x2, options, [], p1, p2,...) — позволяет указывать до 10 дополнительных входных параметров **pi**, которые задаются целевой функции: **f(x, p1, p2,...)**.

x=fmins('fun', x0, options, 'gradfun') позволяет задать в качестве входного параметра функции fminu функцию “**gradfun**”, которая возвращает вектор частных производных целевой функции **df/dx**, в точке **x**: **gf = gradfun(x)**. В этом случае поиск решения будет осуществляться эффективнее и точнее.

Функция **fmins** использует метод “симплексного поиска”.

Функция **fmins** может выдавать предупреждающее сообщение вида:

*'Warning: Maximum number of iterations has been exceeded'
' – increase options(14) for more iterations.'*

(Предупреждение: превышено максимально допустимое количество итераций: для выполнения большего числа итераций увеличьте **OPTIONS(14)**.)

Функция fminu

Функция **fminu** ищет минимум функции нескольких переменных и отличается от **fmins** только используемым алгоритмом оптимизации (формат вызова и назначение аргументов соответствует функции **fmins**).

Алгоритм, использующийся в функции **fminu**, зависит от значения компонента **options(6)** входного управляющего вектора “**options**”:

- **options(6) = 0** — используется квазиньютоновский метод Брайдена-Флетчера-Голдфарба-Шенно (по умолчанию);
- **options(6) = 1** — используется метод Дэвидона-Флетчера-Пауэлла;
- **options(6) = 2** — используется метод наискорейшего спуска.

Кроме того, задав различные значения компонента **options(7)** входного управляющего вектора “**options**”, можно использовать различные методы определения длительности шага:

- **options(7) = 0** — используется метод со смешанными квадратичными и кубическими функциями определения длительности шага (квадратичная и кубическая интерполяция).
- **options(7) = 1** — используется метод только с кубической функцией определения длительности шага (кубическая интерполяция).

Функция **fminu** может выдавать предупреждающее сообщение вида:

*'Warning: Maximum number of iterations has been exceeded'
' - increase options(14) for more iterations.'*

(Предупреждение: превышено максимально допустимое количество итераций: для выполнения большего числа итераций увеличьте OPTIONS(14))

Функция lp

Функция **lp** решает задачу линейного программирования:

$$\min_x c^T x, \text{ при } Ax \leq b$$

Форматы вызова функции **lp** и аргументы функции: полностью совпадают с форматами и аргументами функции **qr**.

Решение задачи линейного программирования внутри функции **lp** осуществляется путём вызова **qr** с пустой матрицей **H** в качестве параметра. Таким образом, при решении линейной задачи возможно появление всех сообщений, которые выдаются в квадратичном случае. Функция **lp**, охватывая лишь подмножество задач, решаемых **qr**, не выдаёт никаких дополнительных сообщений.

Функция minimax

Функция **minimax** минимизирует наихудшее значение из набора функций многих переменных:

$$\min_x \max_{\{F_i\}} \{F_i(x)\}, \text{ при } G(x) \leq 0,$$

где x — вектор переменных, $F(x)$ и $G(x)$ — функции; возвращающие значения; $F_i(x)$ — значение i -го элемента вектора, возвращаемого функцией $F(x)$.

$G(x)$ может быть использована для определения ограничений типа неравенств или равенств.

Формат вызова функции:

x = minimax('fun',x0) — начало оптимизации с точки x_0 и нахождение решения для функции, определенной в М-файле **fun.m**.

x = minimax('fun',x0,options)

x = minimax('fun',x,options,vlb,vub)

x = minimax('fun',x0,options,vlb,vub,'grad')

x = minimax('fun',x0,options,vlb,vub,'grad',p1,p2,...)

[x,options] = minimax('fun',x0, ...).

Назначение каждого из возможных вариантов вызовов соответствует, описанным в функции **attgoal**.

Аргументы функции

(Описаны только те, назначение которых отличается от аргументов функции **attgoal**, или требует дополнительных пояснений):

fun — строка, содержащая имя функции, в которой вычисляется целевая функция и ограничения в точке **x**. Функция **fun** возвращает два аргумента: скалярное значение функции **f** и вектор значений ограничений **g**:

$$[f,g] = \text{fun}(x).$$

Можно задать целевую функцию и ограничения по отдельности (в разных функциях **fun(x)** и **cstr(x)**):

$$x = \text{minimax}'f = \text{fun}(x); g = \text{cstr}(x);', x0.$$

Если имеются ограничения типа неравенств, целевая функция **f** минимизируется таким образом, что **g <= zeros(size(g))**.

Если есть ограничения типа равенств, то они должны располагаться в первых элементах вектора **g** и **options(13)** должен содержать число таких ограничений.

Для минимизации наихудшего абсолютного значения любой из функций вектора **F(x)** (т. е., **minimax abs{F(x)}**), необходимо разместить эти целевые функции в первых элементах **F(x)** и установить в **options(15)** число таких функций.

grad — строка содержащая имя функции, которая вычисляет значение градиента функции и градиента ограничений в точке **x**. Функция имеет следующий формат:

$$[df,dg] = \text{grad}(x).$$

Переменная **df** — вектор, содержащий частные производные **f** по **x**, переменная **dg** — матрица, колонки которой содержат частные производные для каждого из ограничений соответственно (**i**-й столбец **dg** содержит частные производные **i**-го ограничения по каждому элементу **x**).

x0, p1, p2, ..., vlb, vub — такие же как и для функции **constr**.

Пример 7.2

Найти значение x , обеспечивающее минимум максимального значения следующих функций:

$$[f_1(x), f_2(x), f_3(x), f_4(x), f_5(x)],$$

где

$$f_1(x) = 2x_1^2 + x_2^2 - 48x_1 - 40x_2 + 304;$$

$$f_2(x) = -x_2^2 - 3x_2^2;$$

$$f_3(x) = x_1 + 3x_2 - 18;$$

$$f_4(x) = -x_1 - x_2;$$

$$f_5(x) = x_1 + x_2 - 8.$$

Решение.

1. Напишем следующий M-файл:

```
function [f,g] = fun(x)
```

```
% целевые функции
```

```
f(1)=2*x(1)^2+x(2)^2-48*x(1)-40*x(2)+304;
```

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

```
f(3)= x(1) + 3*x(2) -18;
```

```
f(4)= -x(1)- x(2);
```

```
f(5) = x(1) + x(2) - 8;
```

```
% без ограничений
```

```
g = [ ];
```

2. Определение условий оптимизации:

```
% начальное приближение
```

```
x0 = [0.1,0.1];
```

```
x = minimax('fun',x0)
```

Через 29 шагов будет получено решение:

```
x =
```

```
4.0000 4.0000
```

```
fun(x)
```

```
ans =
```

```
0.0000 -16.0000 -2.0000 -8.0000 0.000bsemi
```

Функция qp

Функция **qp** решает задачу квадратичного программирования:

$$\min_x \left(\frac{1}{2} x^T H x + c^T x \right) \text{ при } Ax \leq b.$$

Формат вызова функции:

x = qp(H, c, A, b)

x = qp(H, c, A, b, vlb, vub) — устанавливает допустимые пределы для переменных **x**; в этом случае решение всегда лежит в диапазоне **vlb < x < vub**.

x = qp(H, c, A, b, vlb, vub, x0) — указывает начальную точку **x0**, с которой необходимо начать поиск.

x = qp(H, c, A, b, vlb, vub, x0, n) — указывает, что первые **n** ограничений, определённых матрицами **A** и **b** — это ограничения типа равенств.

[x, lambda] = qp (H, c, A, b) — возвращает набор множителей Лагранжа **lambda** в найденной точке оптимума.

x=qp(H,c,A,b,vlb,vub,x0,n,display) — функция **qp** выводит предупреждающие сообщения в тех случаях, когда не найдено допустимое решение, или когда функция неограниченно возрастает в области допустимых решений. Вывод предупреждающих сообщений может быть отключён, для этого **display = -1**

[x, lambda, how] = qp(H, c, A, b,...) — позволяет в вызывающей программе проанализировать результаты оптимизации. Возможны следующие значения переменной **how**.

1). ‘**ok**’ — оптимальная точка успешна найдена. При этом функция **qp** может выдать на экран предупреждение следующего вида.

‘Warning: QP problem is -ve semi-definite.’

(Предупреждение: QP-задача неточно сформулирована.)

Выдача такого сообщения указывает на то, что заданная квадратичная функция не имеет безусловного минимума (а имеет безусловный максимум), что соответствует некорректной постановке задачи.

2). ‘**infeasible**’ — не существует допустимого решения.

При этом функция **qp** может выдать на экран одно из следующих предупреждений.

*'Warning: The equality constraints are overly stringent;'
'there is no feasible solution.'*

(Предупреждение: ограничения типа равенств слишком строгие — допустимого решения не существует.)

*'Warning: The constraints or bounds are overly stringent;'
'there is no feasible solution.'
'Equality constraints have been met.'*

(Предупреждение: ограничения или допустимые пределы переменных слишком строгие — допустимого решения не существует.
Встретились ограничения типа равенств.)

*'Warning: The constraints are overly stringent;
'there is no feasible solution.'*

(Предупреждение: ограничения слишком строгие: допустимого решения не существует.)

3). ‘**overly constrained**’ — ограничения слишком строгие.

4). ‘**unbounded**’ — оптимизируемая функция неограниченно убывает в области допустимых значений. При этом функция **qp** выдаёт на экран следующее предупреждение.

*'Warning: The solution is unbounded and at infinity;
'the constraints are not restrictive enough.'*

(Предупреждение: Решение не ограничено и стремится к бесконечности; ограничения недерживают убывание функции.)

5). ‘**ill posed**’ — задача плохо сформулирована. При этом функция **qp** выдаёт на экран следующее предупреждение.

'Warning: The search direction is close to zero; the problem is ill posed.'

*'The gradient of the objective function may be zero'
'or the problem may be badly conditioned.'*

(Предупреждение: направление поиска близко к нулю — задача плохо сформулирована.

Возможно, градиент целевой функции равен нулю, или задача плохо обусловлена.)

6).‘**unreliable**’ — полученное решение ненадёжно, т. к. задача плохо обусловлена. При этом функция **qp** выдаёт на экран следующее предупреждение.

*'Warning: The problem is badly conditioned;
'the solution is not reliable'*

(Предупреждение: задача плохо обусловлена — решение ненадежно.)

Аргументы функции

H, c — матрица Гессе и вектор **c** — коэффициенты квадратичной целевой функции. **H** должна быть симметричной.

A, b — матрица и вектор коэффициентов линейных ограничений. Коэффициенты ограничений типа равенства должны быть помещены в первые элементы матрицы **A** и вектора **b**.

vlb, vub — нижний и верхний пределы **x**.

x0 — начальная точка. (Функция **qp** по умолчанию начинает поиск с точки **zeros (size(x))**).

n — число ограничений типа равенства.

display — флаг, определяющий необходимость отображения предупреждений (по умолчанию — 0, отображать предупреждения, установка значения — 1 позволяет подавить предупреждения.)

lambda — вектор результата — набор коэффициентов Лагранжа в результирующей точке. Размерность вектора **lambda** — суммарное число элементов трех векторов:

$$\text{length}(\mathbf{b}) + \text{length}(\mathbf{vlb}) + \text{length}(\mathbf{vub});$$

последовательность коэффициентов: первыми — для **A**, затем **vlb**, затем **vub**.

Функция seminf

Функция **seminf** находит минимум нелинейной функции многих переменных при наличии параметрических ограничений:

$$\begin{aligned} & \min_x f(x), \\ & \text{при } G(x) \leq 0 \quad \text{и} \quad K_1(x, w_1) \leq 0 \\ & \quad K_2(x, w_2) \leq 0, \\ & \quad \dots \\ & \quad K_n(x, w_n) \leq 0 \end{aligned}$$

где **x** и **G(x)** — вектора; **f(x)** — скалярная функция.

G(x) может использоваться для определения как ограничений типа неравенств, так и ограничений типа равенств.

$K_n(x, w_n) \leq 0$ — вектора или матрицы непрерывных функций переменной x и дополнительного набора переменных w_1, w_2, \dots, w_n .

Задача состоит в отыскании минимума функции $f(x)$ таким образом, чтобы обеспечить выполнение ограничений для всех возможных значений w_i ($w_i \in R^1$) — одномерный случай, или $w_i \in R^2$ — двумерный случай). Так как невозможно просчитать все возможные значения $K_n(x, w_n)$, то для w_i задается определенная область, в пределах которой осуществляется поиск подходящих значений x .

Формат вызова функции

x = seminf('fun',n,x0) — оптимизация с начальной точки **x0** и поиск минимума для функции и ограничений, включая полуограничения, определенные в файле **fun.m**.

```

x = seminf('fun',n,x0,options)
x = seminf('fun', n, x,options,vlb, vub)
x = seminf('fun',x0,options,vlb,vub,p1,p2,...)
[x,options] = seminf('fun',n,x0) .

```

Назначение каждого из возможных вариантов вызовов соответствует, описанным в функции **attgoal**.

Аргументы функции

Описаны только те, назначение которых отличается от аргументов функции **attgoal**, или требует дополнительных пояснений.

fun — строка, содержащая имя функции, в которой вычисляется целевая функция и ограничения в точке x .

Функция **fun** возвращает скалярное значение функции **f**, вектор значений ограничений **g**, полубесконечные ограничения K_1, K_2, \dots, K_n , вычисляемые для набора значений независимых переменных w_1, w_2, \dots, w_n соответственно.

Две колонки матрицы **s** содержат рекомендуемые интервалы для значений w_1, w_2, \dots, w_n , которые используются при вычислении K_1, K_2, \dots, K_n .

I-я строка **s** содержит рекомендованный интервал для K_i :

$$[f, g, K1, K2, ..., Kn, s] = fun(x, s)$$

Пример 7.3

Найти x , минимизирующий

$$f(x) = (x_1 - 0,5)^2 + (x_2 - 0,5)^2 + (x_3 - 0,5)^2$$

при

$$K_1(x, w_1) = \sin(w_1 x_1) \cos(w_1 x_2) - \frac{1}{1000} (w_1 - 50)^2 - \sin(w_1 x_3) - x_3 \leq 1$$

$$K_2(x, w_2) = \sin(w_2 x_2) \cos(w_2 x_1) - \frac{1}{1000} (w_2 - 50)^2 - \sin(w_2 x_3) - x_3 \leq 1$$

для всех значений w_1 и w_2 из интервала:

$$0 \leq w_1 \leq 100$$

$$0 \leq w_2 \leq 100$$

($w_i \in R^1$ — одномерный случай).

Решение

1. Определение М-файла

```
function [f,G,K1,K2,s] = fun(X,s)
if isnan(s(1,1)),s = [0.2 0; 0.2 0]; end % начальное значение
w1 = 1:s(1,1):100; % интервал изменения
w2 = 1:s(2,1):100;

% параметрические ограничения
K1 = sin(w1*X(1)).*cos(w1*X(2)) - 1/1000*(w1-50).^2 -...
sin(w1*X(3))-X(3)-1;
K2 = sin(w2*X(2)).*cos(w2*X(1)) - 1/1000*(w2-50).^2 -...
sin(w2*X(3))-X(3)-1;
G = []; % других ограничений нет целевая функция
f = sum((X-0.5).^2);

% построение параметрических ограничений
plot(w1,K1,w2,K2),title('Semi-infinite constraints')
```

2. Результаты оптимизации

```
x0 = [0.5,0.2,0.3]; % начальная точка
x = seminf('fun',2,x0)
```

Через 37 шагов будет получено решение:

$$x = 0.6956 \quad 0.3052 \quad 0.4261$$

$$[f,G,K1,K2] = \text{fun}(x,\text{NaN});$$

$$f = 0.0817$$

$$\max(K1)$$

$$ans = -1.0617e-04$$

$$\max(K2)$$

$$ans = -0.0023$$

Пример 7.4

Найти \mathbf{x} , минимизирующий

$$f(\mathbf{x}) = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 + (x_3 - 0.5)^2$$

при

$$K_1(\mathbf{x}, w) = \sin(w_1 x_1) \cos(10 w_2 x_2) - \frac{1}{1000} (w_1 - 50)^2 - \sin(w_1 x_3) + \\ - x_3 + \sin(w_2 x_2) \cos(w_1 x_1) - \frac{1}{1000} (w_2 - 50)^2 - \sin(w_2 x_3) - x_3 \leq 1.5$$

для всех значений w_1 и w_2 из интервала:

$$0 \leq w_1 \leq 100$$

$$0 \leq w_2 \leq 100$$

($w_i \in \mathbb{R}^2$ — одномерный случай).

Решение

1. Определение M-файла

```
function [f,G,K1,s] = fun(X,s)
```

% начальные значения интервала

```
if isnan(s(1,1)),s = [2 2]; end
```

% диапазон изменения

```
w1 = 1:s(1,1):100;
```

```
w2 = 1:s(1,2):100;
```

```
[wx,wy] = meshdom(w1,w2);
```

% полуограничения

```
K1 = sin(wx*X(1)).*cos(wy*X(2))-1/1000*(wx-50).^2-...
```

```
sin(wx*X(3))-X(3)+sin(wy*X(2)).*cos(wx*X(1))-...
```

```
1/1000*(wy-50).^2-sin(wy*X(3))-X(3)-1.5;
```

```
% других ограничений нет
G = [];

% целевая функция
f = sum((X - 0.2).^2);

% графические построения
mesh(K1), title('Semi-infinite constraint')

2. Результаты оптимизации
x0 = [0.2,0.2,0.2]; % начальное приближение
x = seminf('fun',1,x0)
```

Через 65 шагов будет получено решение:

$x = 0.2033 \quad 0.2034 \quad 0.1930$

[f,G,K1] = fun(x,NaN);

max(max(K1))

$ans = -0.0273$

7.4. ПРИМЕР СОЗДАНИЯ ПОДСИСТЕМЫ МОДЕЛЕЙ ДЛЯ РЕШЕНИЯ ЗАДАЧ ОПТИМИЗАЦИИ

Для проведения исследований, связанных с изучением эффективности применения тех или иных методов решения одной и той же задачи (или разных задач одного класса), определением степени влияния начальных условий и других параметров на качество решения задач, представляет интерес разработка подсистемы моделей с использованием готовых функций системы MATLAB. Такая подсистема позволит задавать в удобной форме функции, подлежащие оптимизации и систему ограничений, которым должна удовлетворять оптимальная точка. Кроме того, результаты оптимизации можно отображать в графической форме и сохранять шаги оптимизации в файле.

Для запуска любой программы в системе MATLAB должны быть определены пути к каталогам, в которых находятся основные и вспомогательные функции. Например, если вся подсистема моделей располагается в каталоге C:\matlab5\toolbox\optim\opt, то для использования программы Lqp должны быть заданы следующие пути:

C:\matlab5\toolbox\optim\opt,

C:\matlab5\toolbox\optim\opt\lqp,

C:\matlab5\toolbox\optim\opt\lqp\examples;

для программы Unconmiin:

C:\matlab5\toolbox\optim\opt,

C:\matlab5\toolbox\optim\opt\unconmin,
C:\matlab5\toolbox\optim\opt\unconmiin\examples,
C:\matlab5\toolbox\optim\opt\unconmiin\datas;

для программы Conmin:

C:\matlab5\toolbox\optim\opt,
C:\matlab5\toolbox\optim\opt\conmiin,
C:\matlab5\toolbox\optim\opt\conmin\examples,
C:\matlab5\toolbox\optim\opt\conmiin\datas.

7.4.1. Решение задач линейного и квадратичного программирования (программа “Lqp”)

Назначение программы

Программа предназначена для решения задач линейного и квадратичного программирования — нахождения минимума функции вида

$$0,5x^T H x + F^T x,$$

(в случае линейной задачи $H = \mathbf{O}$), при наличии ограничений следующего вида:

$$\begin{aligned} A \cdot x &= b; \\ x &> \mathbf{VLB}; \\ x &< \mathbf{VUB}, \end{aligned}$$

Программа “Lqp”, использующая функции **lp** и **qp** библиотеки “Optim”, позволяет задавать в удобной форме линейные или квадратичные функции, а также систему ограничений. Результаты оптимизации выводятся на экран, а после этого осуществляется построение графиков, иллюстрирующего решаемую задачу.

Кроме того, имеется возможность последовательного увеличения числа активных ограничений, что позволяет проследить степень влияния на нахождение оптимального решения каждого из ограничений. Например, вначале найти оптимум при наличии только ограничений типа 1, а затем подключить ограничения типа 2 и типа 3.

Описание входных данных программы

Оптимизируемая с помощью программы “Lqp” функция, а также система ограничений, которые будут использоваться должны быть описаны в отдельном файле. Этот файл должен содержать определения следующих глобальных переменных:

- 1) \mathbf{H} — квадратная матрица размером $M \times M$,
где M — размерность задачи;

2) \mathbf{F} — столбец свободных членов размером $\mathbf{M} \times 1$;

3) \mathbf{A} — матрица ограничений размером $\mathbf{P} \times \mathbf{M}$,

где \mathbf{P} — общее число ограничений;

4) \mathbf{b} — столбец коэффициентов в правой части ограничений;

5) \mathbf{VLB} — столбец ограничений типа $\mathbf{x} > \mathbf{VLB}$ размером $\mathbf{M} \times 1$;

6) \mathbf{VUB} — столбец ограничений типа $\mathbf{x} < \mathbf{VUB}$ размером $\mathbf{M} \times 1$;

7) $\mathbf{X_0}$ — начальная точка;

8) \mathbf{N} — число ограничений типа неравенства в решаемой задаче.

Из рассмотренного списка обязательному определению во входном файле подлежат только те переменные, которые необходимы для корректной формулировки решаемой задачи и для проведения исследования (включение в формулировку задачи тех или иных ограничений). Так, например, если решается задача линейного программирования, то необязательно задавать в исходном файле матрицу \mathbf{H} .

Задавая различные параметры в исходном файле, можно решать задачи поиска оптимального решения при всех возможных комбинациях ограничений.

Так, если необходимо решить задачу с ограничениями типа равенств, не задавая начальную точку, то во входном файле надо определить переменную $\mathbf{X_0}$ следующим образом:

$$\mathbf{X_0} = [];$$

После запуска программы **Iqr** файл, содержащий описанные выше параметры, можно выбрать при помощи окна поиска, которое формируется программой. Затем по результатам анализа исходных данных можно уточнить тип решаемой задачи (линейное или квадратичное программирование) и выбрать тип ограничений, учитываемых при решении данной задачи. Далее вызываются функции оптимизации **Ip** или **qp** с необходимым списком параметров. Результаты выполнения этих функций анализируются в программе и выводятся на экран.

Опции меню, предлагаемые программой

1. Quit. Выход из программы.
2. Only inequality constraints: $\mathbf{Ax} \leq \mathbf{b}$. Решение задачи условной оптимизации при наличии только ограничений типа $\mathbf{Ax} \leq \mathbf{b}$.
3. Extra constraints: $\mathbf{x} > \mathbf{VLB}$. Решение задачи условной оптимизации при наличии только ограничений типа $\mathbf{Ax} \leq \mathbf{b}$ и $\mathbf{x} > \mathbf{VLB}$.
4. Extra constraints: $\mathbf{x} < \mathbf{VUB}$. Решение задачи условной оптимизации при наличии только ограничений типа $\mathbf{Ax} \leq \mathbf{b}$, $\mathbf{x} > \mathbf{VLB}$ и $\mathbf{x} < \mathbf{VUB}$.

5. With the initial starting point. Решение задачи условной оптимизации при наличии ограничений типа $\mathbf{Ax} \leq \mathbf{b}$, $\mathbf{x} > \mathbf{VLB}$ и $\mathbf{x} < \mathbf{VUB}$. Поиск решения начинается с точки $\mathbf{X_0}$.

6. Extra N equality constraints: $\mathbf{Ax} = \mathbf{b}$.

Решение задачи условной оптимизации при наличии ограничений типа $\mathbf{Ax} \leq \mathbf{b}$, $\mathbf{x} > \mathbf{VLB}$ и $\mathbf{x} < \mathbf{VUB}$ начинается с точки $\mathbf{X_0}$. Первые N ограничений типа $\mathbf{Ax} \leq \mathbf{b}$ рассматриваются как ограничения типа равенств $\mathbf{Ax} = \mathbf{b}$.

Пример 7.5

Решение задачи квадратичного программирования

1. Файл с определением проблемы программирования

```
%* TASKLQP7.M
%*
%* Задание #1 по летней практике по курсу
%* "Системный анализ".
%* Пример решения задачи квадратичного
%* программирования.
%* Авторы: Ильин А.Н., Никитин А.В.
%* Группа 4081/4
%*****+
%* Функция TASK #7 to "Lqp" function
%*
function tasklqp7
global H F A b VLB VUB N Xo
H = [ 2, 0; 0, 2 ]';
F = [ 0 0 ]';
A = [ 1, -1; 1, 1; -1, -1 ];
b = [ 6 2 2 ]';
VLB = [ 0.5 -3 ]';
VUB = [ 4 -0.5 ]';
Xo = [];
N = 1;
%*
%* Конец файла tasklqp7.m
Таким образом, минимизируется функция вида
f( X ) = 2*( x12 + x22)
при следующих ограничениях типа неравенств:
x2 >= x1 - 6
```

$$\begin{aligned}x_2 &\leq -x_1 + 2 \\x_2 &\geq -x_1 - 2\end{aligned}$$

7.4.2. Решение задач безусловной оптимизации (программа “Unconmin”)

Для решения задач безусловной оптимизации (поиска локального минимума функции) с использованием функций fmin, fminu и fmins, входящих в оптимизационный пакет программ “Optim”, предназначена программа “Unconmin”.

Она позволяет в описывать решаемую задачу, использовать при нахождении безусловного минимума различные методы поиска, а также представлять результаты работы программы: координаты найденной оптимальной точки, число пройденных шагов и предупреждения, выдаваемые оптимизирующими процедурами.

Существует возможность создания файла отчёта, содержащего аналогичную информацию. Кроме того, в двумерном случае можно построить трёхмерные и двухмерные графики, иллюстрирующие процесс поиска решения.

Описание входных и выходных данных программы

Описание решаемой задачи безусловной минимизации должно содержаться во входном файле. В этом файле необходимо определить следующие глобальные переменные.

1. **func** — название целевой функции — имя файла MATLAB с определением функции, имеющей формат

function Y = f(X),

где **X** — вектор (точка, в которой вычисляется значение функции),
Y — скаляр (вычисленное значение).

2. **dim** — размерность задачи.

3. **GRD** — название функции, возвращающей значение градиента целевой функции — имя файла MATLAB с определением функции, имеющей формат

function Y = g(X),

где **X** — вектор (точка, в которой вычисляется градиент);
Y — вектор частных производных (градиент);

4. **Xo** — матрица, каждая строка которой представляет собой значение начальной точки, из которой можно осуществить поиск оптимума.

5. **TrueOptPt** — матрица, каждая строка которой является известной точкой локального минимума целевой функции;

6. **Values** — вектор значений целевой функции в известных точках локального минимума;

7. **AB** — матрица, каждая строка которой является возможным интервалом поиска минимума в одномерном случае.

8. **xeps** — точность, с которой должно быть определено значение **X** (по умолчанию: 1.e-4);

feps — точность, с которой должно быть определено значение функции в решении (по умолчанию: 1.e-4);

steps — это максимальное количество итераций (по умолчанию: 500).

Обязательными переменными, которые должны быть определены в исходном файле, являются переменные **func** и **dim**. Переменные **Xo**, **TrueOptPt**, **Values** и **AB** служат для указания начальных условий и для представления в более удобном виде результаты работы программы. Определение переменной **GRD** позволяет увеличить количество методов, которые могут быть применены для решения задачи.

После запуска программы **Unconmin** осуществляется поиск файла, содержащего указанные определения. Для этого на экран выводится соответствующее окно поиска.

После нахождения исходного файла он исполняется, в результате программы **Unconmin** становятся “известны” необходимые входные параметры. Далее по результатам проверки полученных параметров выдаются сообщения об ошибках, или предлагается выбрать один из методов оптимизации. На экран выводится меню, содержащее несколько пунктов.

В программе **Unconmin** можно использовать различные методы поиска точки локального минимума:

- метод Бройдена-Флетчера-Голдфарба-Шенно;
- метод Дэвидона-Флетчера-Пауэлла;
- метод наискорейшего спуска;
- метод “симплексного поиска”;
- метод одномерной минимизации, комбинирующий метод золотого сечения и метод параболической интерполяции.

Выбор необходимого метода оптимизации осуществляется путём выбора определённого пункта меню. После этого осуществляется вызов соответствующей процедуры с необходимыми входными параметрами. Особенностью трёх первых указанных алгоритмов оптимизации, которые

используют процедуру **fminu**, является возможность использования в них различных методов определения длительности шага. Для выбора одного из двух возможных методов на экран выводится меню, после чего необходимым образом определяются компоненты входного вектора **OPTIONS**.

После выбора одного из предложенных методов минимизации производится задание начальной точки (или интервала поиска в случае, если функция одной переменной минимизируется с помощью процедуры **fmin**).

Задание начальной точки может осуществляться следующими способами (меню "начальные условия").

1. Начальные точки, предложенные в верхней части меню. Значения этих точек сохранились от предыдущего запуска программы или они были заданы в текущем сеансе работы программы **Unconmin**.

2. **Input from the keyboard.** Ввод начальной точки с клавиатуры.

3. **Graphical input.** Графический ввод точки (он возможен в случае, если открыто окно, соответствующее выбранному методу оптимизации).

4. Начальные точки, предложенные в нижней части меню. Значения этих точек были указаны во входном файле программы.

Введённые одним из указанных методов начальные точки сразу "не забываются", т. е. существует возможность использования этих точек при повторном запуске программы "**Unconmin**". Начальная точка, которая не была описана во входном файле, а была задана в процессе работы программы, появляется в предлагаемом списке начальных точек. Таким образом, указанный список постепенно увеличивается при проведении экспериментов над функциями одинаковой размерности.

Максимальное количество точек, которые могут находиться в этом списке, ограничено восемью. Если достигнут этот предел, то при задании очередного начального значения точка, которая дольше всего не использовалась, будет удалена из списка. Точки, заданные во входном файле не могут быть удалены из списка (т. е. они являются более приоритетными). Для выделения таких начальных точек они указываются в нижней части предлагаемого меню, после пунктов **Input from the keyboard** и **Graphical input**.

После задания начальной точки вызывается одна из процедур **fmin**, **fminu** или **fmins** с необходимыми входными параметрами, осуществляется анализ возвращённых её значений, и на экран выводятся результаты поиска, может быть выведено предупреждение от программ оптимизации. Затем предлагается провести построение иллюстрирующей процесс решения картинки (в двумерном случае). Выбор одного из пунктов меню "построить" приводит к выполнению следующих действий.

1. **Only the path of the solving** — построение только траектории поиска решения.
2. **With the 3-D mesh surface** — построение только трёхмерного графика целевой функции.
3. **Only the 3-D mesh surface** — построение и траектории поиска решения и трёхмерного графика целевой функции.
4. **Don't draw** — не строится ничего.

Если пользователь запросил построение траектории поиска решения, следующим действием производится определение способа ее построения.

Возможны следующие режимы построения траектории поиска:

1. **Step by step** — построение по шагам, после каждого шага ожидается нажатие клавиши.
2. **Slow drawing** — построение в динамике, производится “медленное” построение пути — с небольшой паузой после каждого шага.
3. **All at once** — построение окончательного вида траектории поиска.

После этого производится построение всех необходимых графиков, Если построенные графики не достаточно наглядно иллюстрируют результаты работы, существует возможность перестроить эти графики в других пределах. Выбор одного из пунктов меню означает следующее:

1. **Input from the keyboard** — задание новых пределов с клавиатуры.
2. **Graphical input** — графическое задание новых пределов (указываются две точки на графике или за его пределами).
3. **Auto limits** — пределы, устанавливаются программой (первоначальный вид графиков).

После выхода из режима построения графиков при помощи пункта меню **Don't Redraw** программа **Unconmin** предлагает сохранить последние результаты работы процедур оптимизации. При этом создаётся или дополняется уже существующий файл отчёта. Файл отчёта имеет имя, совпадающее с именем входного файла программы, но имеет расширение “out”. Выбор одного из пунктов меню «**сохранить**» приводит к следующим действиям:

1. **Overwrite file** — создание нового файла отчёта.

2. **Add to file** — добавление результатов в конец существующего файла.
3. **Don't save** — не сохранять последние результаты.

Пример 7.6

Решение задачи безусловной оптимизации при помощи программы Unconmin

Файл с определением решаемой задачи

```
%* TASK1CHK.M
%*
function task1chk
global func GRD Xo dim TrueOptPt Values AB xeps yeps
dim = 1;
func = 'func1chk';
GRD = 'grad1chk';
AB = [-23 67377];
Xo = [3];
TrueOptPt = [3];
Values = [4];

xeps = 1.e-12;
yeps = 1.e-30;
```

Файл с описанием целевой функции

```
%* FUNC1CHK.M
function y = func1chk( x )
y = (x-3)^2 + 4;
```

Файл с описанием градиента целевой функции

```
%* GRAD1CHK.M
%*
function y = grad1chk( x )
y = (x-3)*2;
```

Фрагмент созданного программой “Unconmin” файла отчёта
 Unconstrained minimization of the function described in the file
 "func1chk.m"

The known points of local minimum:
 $X_1 = [3]' \quad f(X_1) = 4$

----- 1 -----

The optimization method: Function FMIN.

OPTIONS = [1 1e-007 0 0 0 0 0 0 0 0 0 0 0 0 500]'

The search interval [A,B] = [-3 78]'

Used optimizer:

Xopt = fmin(FUNC, AB(1), AB(2), OPTIONS);

The solution is: X = 3

Value of the function at the solution: 4

----- 2 -----

The optimization method : Broyden-Fletcher-Goldfarb-Shanno.

The line search method : Mixed Polynomial Interpolation.

OPTIONS = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 200]'

The starting point: Xo = [56]'

Used optimizer:

[Xopt, options] = fminu(FUNC, Xo, OPTIONS, GRAD);

The solution is: X = 3

Value of the function at the solution: 4

Number of function evaluations: 9

Number of gradient evaluations: 3

----- 3 -----

----- 8 -----

The optimization method : Simplex Search.

OPTIONS = [0 0 0 0 0 0 0 0 0 0 0 0 0 0 300]'

The starting point: Xo = [7.457e+007]'

Used optimizer:

[Xopt, options] = fmins(FUNC, Xo, OPTIONS);

The solution is: X = 3

Value of the function at the solution: 4

Number of function evaluations: 118

Number of gradient evaluations: 0

7.4.3. Решение задач условной оптимизации (программа “*Conmin*”)

Назначение программы

Программа **Conmin** предназначена для решения задач условной оптимизации, т. е. для поиска локального минимума функции

$$\min_x f(x)$$

при наличии системы нелинейных ограничений:

$G(X) < 0$, $G(X)$ — матрица ограничений в точке X ;
 $X > VLB$;
 $X < VUB$.

Программа **Conmin** обладает всеми дополнительными возможностями программы **Unconmin**.

Описание входных и выходных данных программы

Описание решаемой задачи условной минимизации должно содержаться во входном файле. В этом файле должны быть определены следующие глобальные переменные.

1. **FUN** — название функции, возвращающей значения целевой функции и вектор функции ограничений — имя файла MATLAB с определением функции, имеющей формат

function [F, G] = FUN(X),

где X — вектор (точка, в которой вычисляется значение функции);

F — скаляр (вычисленное значение);

G — матрица ограничений ($G(x) < 0$).

2. **dim** — размерность задачи.

3. **GRADFUN** — название функции, возвращающей частные производные целевой функции и функции ограничений — имя файла MATLAB с определением функции, имеющей следующий формат:

function Y= g(X),

где X — вектор (точка, в которой вычисляется градиент);

Y — вектор частных производных (градиент).

4. **Xo** — матрица, каждая строка которой представляет собой координаты начальной точки, из которой можно осуществить поиск оптимума.

5. **VLB** — столбец ограничений типа $x > VLB$ размером **dim x1**.
6. **VUB** — столбец ограничений типа $x < VUB$ размером **dim x1**.
7. **N** — число ограничений типа неравенства в решаемой задаче.
8. **xeps** — точность, с которой должно быть определено значение **X** (по умолчанию: $1.e-4$).
9. **feps** — точность, с которой должно быть определено значение функции в решении (по умолчанию: $1.e-4$).
10. **steps** — максимальное количество итераций (по умолчанию: 500).
11. **TrueOptPt** — матрица, каждая строка которой является известной точкой локального минимума целевой функции.
12. **Values** — вектор значений целевой функции в известных точках локального минимума.

Обязательными переменными, которые должны быть определены в исходном файле, являются переменные **FUN** и **dim**. Переменные **Xo**, **TrueOptPt**, **Values** и **AB** позволяют указать интересующие начальные условия и представить результаты работы программы. Переменные **VLB** и **VUB** позволяют наложить дополнительные ограничения на диапазон, в котором должна лежать оптимальная точка. Определение переменной **GRADFUN** позволяет улучшить эффективность поиска решения процедурой **constr**.

Особенностью файла, содержащего описание целевой функции и функции ограничений, является то, что в случае двумерной задачи, при необходимости построить иллюстрирующие графики, в конце этого файла (перед выходом) надо вставить строку:

putptf(X);

Эта команда будет помещать очередную точку в массив, содержащий траекторию поиска оптимальной точки.

После запуска программы, прежде всего, осуществляется поиск файла, содержащего указанные **Conmin** определения. Для этого на экран выводится окно поиска.

После нахождения исходного файла он исполняется, в результате чего программе **Conmin** становятся “известны” необходимые входные параметры.

По результатам проверки параметров выдаются сообщения об ошибках, или предлагается выбрать один из режимов работы программы.

1. **Quit** — Выход из программы.
2. **Only constraints such as : $G < \text{zeros}(G)$** — решение задачи условной оптимизации только при наличии ограничений типа $G(X) < 0$.
3. **Extra constraints:** $x > VLB$ — дополнительные ограничения типа $G(X)$ и $x > VLB$.
4. **Extra constraints:** $x < VUB$ — дополнительные ограничения типа $G(X)$, $x > VLB$ и $x < VUB$.
5. **With GRADFUN definition** — с ограничениями, аналогичными, заданным в определении функции **GRADFUN**.

*Решение задачи условной оптимизации
при наличии ограничений типа*

$$Ax \leq b, x > VLB \text{ и } x < VUB.$$

Поиск решения начинается с точки **X₀**.

Задание начальной точки осуществляется таким же образом, как и в программе **Unconmin** (см. меню "начальные условия" в программе **Unconmin**). После задания начальной точки вызывается процедура **constr** с необходимыми входными параметрами, затем осуществляется анализ возвращённых значений, и на экран выводятся результаты поиска и могут быть выведены предупреждения от процедуры **constr**.

Затем предлагается сохранить последние результаты работы процедуры оптимизации, возможности такие же, как и в программе "**Unconmin**" (см. меню "сохранить").

После сохранения результатов программа **Conmin** предлагает провести построение иллюстрирующей процесс решения картинки (в двумерном случае) с выбором одного из пунктов меню, которое соответствует меню "построить" программы **Unconmin**. Также, как в описанной выше программе, есть возможность перестроить эти графики в других пределах.

После выхода из режима построения графиков при помощи пункта меню "Don't Redraw" появляется основное меню выбора режима работы программы. Выбор пункта "Quit" этого меню приводит к завершению работы программы.

Пример 7.7

Решение задачи безусловной оптимизации при помощи программы “Conmin”

1. Файл с определением задачи программирования

%* TASKCHK.M

function taskchk

global FUN Xo VLB VUB GRADFUN dim N TrueOptPt

Valuesdim = 2;

N = 1;

FUN = 'funcchk';

GRADFUN = 'gradchk';

Xo = [10 -27];

TrueOptPt = [];

VLB = [0 0]';

VUB = [0.5 1.5]';

2. Файл с описанием целевой функции и системы ограничений

%* FUNCCHK.M

%*

function [F , G] = funcchk(X)

F = (X(1) - 1) ^ 2 + (X(2) - 1) ^ 2;

g1 = (X(1)) ^ 2 + (X(2)) ^ 2 - 4;

g2 = (X(1)) - (X(2)) ^ 2 + 1;

G = [g1 g2]';

putptf(X);

3. Файл отчёта, созданный программой “Conmin”

Constrained minimization of the function described in the file
"funcchk.m"

----- 1 -----
The starting point : Xo = [10 -27]'

Used optimizer :

[Xopt,options,lambda,HESS] = constr(FUN,Xo,OPTIONS,VLB,VUB);

The solution is : X = [0.8229 1.823]'

Value of the function at the solution : 0.7085

8. ТЕОРИЯ МАССОВОГО ОБСЛУЖИВАНИЯ. ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ

Основные обозначения

λ — интенсивность потока;

μ — интенсивность обслуживания;

$\rho = \lambda / \mu$ — коэффициент загрузки в системе **G/G/1**;

x — длительность интервала обслуживания (в общем случае случайная величина);

$B(t) = P(x \leq t)$ — интегральный закон распределения длительности обслуживания;

$b(t)$ — плотность распределения длительности обслуживания;

$v^{(1)} = \bar{x}$; $v^{(2)} = \bar{x^2}$ — первый и второй начальные моменты длительности обслуживания;

$A(t) = P(y \leq t)$ — интегральный закон распределения длительности интервалов между требованиями во входном потоке;

$a(t)$ — плотность распределения длительности интервалов между требованиями;

\bar{y} — средняя длительность интервалов между требованиями;

K — число каналов обслуживания в системе;

$\rho_c = \frac{\lambda}{K\mu}$ — суммарный коэффициент загрузки в системе **G/G/K**;

m — емкость накопителя;

n — число требований в конечном источнике;

\bar{j} — среднее число требований в системе;

\bar{n}_0 — среднее число требований в очереди;

P_j — вероятность того, что в системе находится j требований ($j = 0, 1, 2, \dots$)

P_0 — вероятность простого системы;

\bar{t}_c — среднее время пребывания требования в системе;

σ_j^2 — дисперсия числа требований в системе;

$\bar{t}_{ож}$ — среднее время ожидания требования в очереди;

\bar{K}_3 — среднее число занятых каналов в системе;

η_3 — коэффициент загрузки системы;

$P_{отк}$ — вероятность отказа в обслуживании;

$P(n_0 > k)$ — вероятность того, что число требований в очереди больше k ;

$P(t_{ож} < t)$ — вероятность того, что время ожидания в очереди меньше t ;

$\Lambda = \{ \lambda_{ij} \}$ — матрица интенсивностей передач на графе состояний;

$P = \{ P_{ij} \}$ — матрица вероятностей передач на графе состояний.

8.1. КЛАССИФИКАЦИЯ СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ

Модели функционирования объектов, представленных в виде систем массового обслуживания (СМО), имеют структуру, показанную на рис. 8.1.

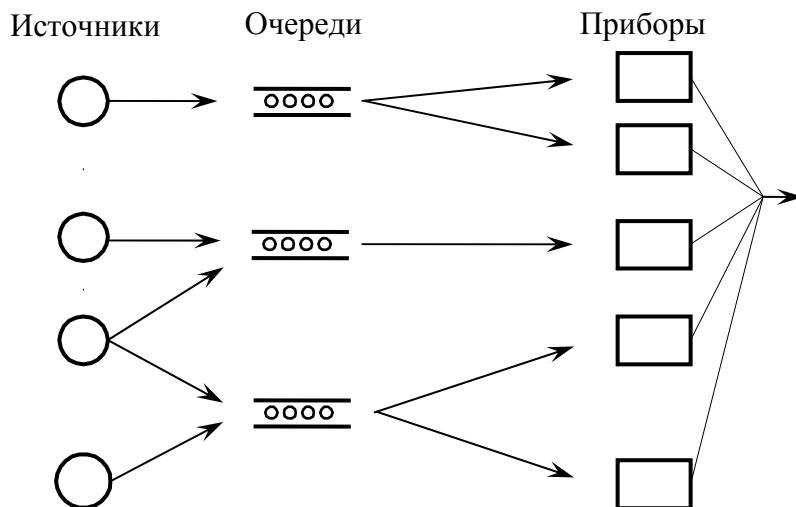


Рис. 8.1

Введем основные определения.

Последовательность событий будем называть потоком. Поток, состоящий из требований на обслуживание, назовем потоком требований.

Первопричину возникновения требований независимо от их физической природы будем называть источником.

Совокупность очередей и приборов обслуживания будем называть обслуживающей системой.

Алгоритм взаимодействия обслуживающих приборов с очередью назовем дисциплиной обслуживания.

Рассмотрим основные классификационные категории СМО.

По характеру источника требований различают источники с бесконечным и конечным числом требований.

По отсутствию или наличию возможностей ожидания для окончательного завершения обслуживания требований в системе выделяют: *системы с потерями* (по причине ограниченной емкости буферного накопителя или ограничения времени ожидания в очереди) и *системы без потерь*.

По числу приборов (каналов) различают: *одноканальные и многоканальные обслуживающие системы*.

По числу этапов (фаз) обслуживания в обслуживающей системе различают *однофазные и многофазные* (в том числе и сети массового обслуживания).

По правилу формирования очереди различают *системы массового обслуживания с общей очередью* и *системы массового обслуживания с несколькими очередями*.

По дисциплине обслуживания различают *системы массового обслуживания: с бесприоритетными дисциплинами и приоритетными* (относительный или абсолютный приоритет). В случае относительного приоритета требование занимает обслуживающий прибор без прерывания фазы обслуживания, а при абсолютном — с прерыванием.

Рассмотренная классификация СМО является далеко не полной, но достаточной для рассмотрения моделей, представленных в данном учебном пособии.

Для изложения дальнейшего содержания данного пособия целесообразно воспользоваться принятой в теории массового обслуживания системой обозначений [29].

В их основе лежит трехбуквенное обозначение вида **A/B/K**, где **A** и **B** описывают, соответственно, распределение промежутков времени между требованиями и распределение времени их обслуживания, а **K** — число обслуживающих приборов.

A и **B** принимают значения из следующего набора символов, интерпретация которых дается распределениями, указанными в круглых скобках: **M** (показательное или иначе экспоненциальное), **E_r** (распределение Эрланга порядка **r**), **D** (детерминированное), **G** (распределение общего вида).

Иногда приходится указывать также емкость накопителя системы, которую обозначим через **m**, и число источников нагрузки (**n**). В этом случае будем использовать пятибуквенное обозначение: **A/B/K/m/n**.

В случае отсутствия одного из двух последних индексов предполагается, что его значение сколь угодно велико, т. е. равно бесконечности.

В дальнейшем эта терминология будет использована при формализации и анализе технических характеристик систем, представляемых в виде систем массового обслуживания.

8.2. ПОТОКИ СОБЫТИЙ И ИХ СВОЙСТВА

8.2.1. Классификация потоков

Поток — последовательность событий, происходящих одно за другим в случайные или детерминированные моменты времени. Физическая природа потоков: поступления заявок на решение задачи, выдача результата, посадка или взлет самолета, отказ аппаратуры, поступление вызовов на АТС.

Различают потоки:

однородные, в которых события характеризуются только моментом времени появления;

неоднородные, в которых события дополняются другими качественными и количественными характеристиками, такими как вид и уровень приоритета, время обработки и т. п.

В дальнейшем, пока будем говорить об однородных потоках.

С потоком связаны две группы характеристик случайных величин:

плотность распределения длительности интервалов $f(\tau_j)$:

$$\tau_j = t_{j+1} - t_j, \quad j = 0, 1, 2, \dots ;$$

закон распределения числа событий $n(t, T)$ на интервале T , отсчитываемом от момента t , $P_n(t, T)$, где $n = 0, 1, 2, \dots$.

В первом случае имеем последовательность непрерывных случайных величин; во втором — дискретных случайных величин. По характеру величины τ_j выделяют потоки:

- случайные, если величины τ_j случайные;
- детерминированные, если величина τ_j постоянная, т. е. $\tau_j = \tau_0$;
- случайные с дискретным временем: $\tau_j = k_j T_0$, где T_0 — шаг дискретного времени, k_1, k_2, \dots, k_j — возможные точки появления событий;
- регулярные потоки, в которых события появляются обязательно в точках $\tau_j = k_j T_0$.

Определим основные модели распределений, рассматриваемые в данном пособии:

$$1) \quad f(\tau) = \lambda \cdot e^{-\lambda\tau}, \quad \tau \geq 0, \quad M — \text{показательное, (Markovian);}$$

$$2) \quad f_r(\tau) = \frac{r\lambda(r\lambda\tau)^{r-1}}{(r-1)!} e^{-r\lambda\tau}, \quad E_r — \text{эрланговское, (Erlangian);}$$

$$3) \quad f_r(\tau) = \sum_{i=1}^r \alpha_i \lambda_i e^{-\lambda_i \tau}, \quad (\sum_{i=1}^r \alpha_i = 1), \quad H_r — \text{гиперэкспоненциальное (Hyperexponential);}$$

$$4) \quad f(\tau) = \delta(\tau - \tau_0); \quad \tau_j = \tau_0, \quad D — \text{постоянное, (Deterministic);}$$

$$5) \quad f(\tau), \quad G — \text{произвольное, (General).}$$

Статистические свойства случайных величин $n(t, T)$ определяют важнейшие свойства потоков.

1. Стационарность: $P_n(t, T) = P_n(T), \quad n = 0, 1, 2, \dots;$

(распределение не зависит от положения интервала T на оси времени и зависит только от длительности T).

2. Отсутствие последействия:

$$P_{n_1 n_2 \dots n_i}(T_1 T_2 \dots T_i) = \prod_{k=1}^i P_{n_k}(T_k), \quad (8.1)$$

(независимость числа событий в неперекрывающихся интервалах T_i). Данное свойство с большим трудом поддается экспериментальной проверке.

3. Ординарность: вероятность появления более одного события на бесконечно малом интервале имеет порядок малости выше, чем вероятность появления одного события на этом интервале, т. е.

$$\lim_{\Delta t \rightarrow 0} \frac{P_n(\Delta t)}{P_1(\Delta t)} = 0, \quad n = 2, 3, \dots$$

или

$$\lim_{\Delta t \rightarrow 0} \frac{P_{n>1}(\Delta t)}{\Delta t} = 0, \quad n = 2, 3, \dots. \quad (8.2)$$

Можно показать, что свойство ординарности вытекает из свойств стационарности и отсутствия последействия.

Поток, характеризующийся стационарностью, отсутствием последействия и ординарностью, называется простейшим потоком.

Стационарные потоки с независимыми интервалами, распределенными по произвольным законам $F_1(t), F_2(t), \dots, F_i(t)$,

где $F_i(t) = P(\tau_i < t)$ — функция распределения i -го интервала в потоке, называются потоками Пальма.

Если все интервалы распределены одинаково, то такие потоки называются рекуррентными.

Поток Пальма с экспоненциальным распределением длительности интервала в потоке является простейшим.

Из отсутствия последействия вытекает независимость интервалов, но независимость интервалов не означает автоматически отсутствия последействия.

8.2.2. Простейший поток и его свойства

1. Закон распределения длительности интервала выводится из свойств стационарности и отсутствия последействия.

$$\begin{aligned} f(\tau) &= \lambda e^{-\lambda \tau}, \quad \tau > 0, \\ P(\tau < t) &= F(t) = 1 - e^{-\lambda t}, \end{aligned} \quad (8.3)$$

где λ — параметр потока.

Закон распределения числа событий на интервале T .

$$P_n(T) = \frac{(\lambda T)^n}{n!} e^{-\lambda T}.$$

2. Определение параметра потока λ .

Сначала найдем среднюю интенсивность потока на интервале $[t, t+T]$:

$$\lambda(t, T) = \frac{M[n(t, T)]}{T}, \quad (8.4)$$

где $M[n(t, T)]$ — среднее число событий на интервале $[t, t+T]$.

Мгновенная интенсивность потока определяется так:

$$\lambda(t) = \lim_{\Delta t \rightarrow 0} \frac{M[n(t, \Delta t)]}{\Delta t}, \quad (8.5)$$

причем: $\lambda(t, T) = \frac{1}{T} \int_t^{t+T} \lambda(x) dx.$

Тогда для стационарного потока:

$$\lambda(t) = \lim_{\Delta t \rightarrow 0} \frac{M[n(t, \Delta t)]}{\Delta t} = \lim_{\Delta t \rightarrow 0} \frac{M[n(\Delta t)]}{\Delta t} = \lambda = const, \quad (8.6)$$

а среднее число событий на интервале T будет равно

$$M[n(T)] = \lambda T. \quad (8.7)$$

Тот же результат можно получить с учетом подчинения событий простейшего потока закону Пуассона:

$$M[n(T)] = \sum_{n=0}^{\infty} n \frac{(\lambda T)^n}{n!} e^{-\lambda T} = \lambda T \sum_{n=1}^{\infty} \frac{(\lambda T)^{n-1}}{(n-1)!} e^{-\lambda T} = \lambda T,$$

$$\text{т. е. } \lambda = \frac{M[n(T)]}{T}.$$

Отсюда следует, что параметр потока равен интенсивности потока (средняя интенсивность равна мгновенной).

3. Законы распределения, экспоненциальный $F(t) = 1 - e^{-\lambda t}$ и Пуассона $P_n(T) = \frac{(\lambda T)^n}{n!} e^{-\lambda T}$, связаны между собой:

$$P_0(T) = e^{-\lambda T} = P(\tau > T), \text{ тогда } P(\tau < T) = 1 - e^{-\lambda T},$$

а это и есть распределение $F(t) = 1 - e^{-\lambda t}$.

4. Численные характеристики экспоненциального закона:

$$M[\tau] = \int_0^{\infty} \tau \lambda e^{-\lambda \tau} d\tau = \frac{1}{\lambda}, \quad (8.8)$$

$$D[\tau] = M[\tau^2] - (M[\tau])^2 = \frac{1}{\lambda^2}, \quad (8.9)$$

$$M[\tau] = \sigma[\tau]. \quad (8.10)$$

Соотношение (8.10) — основное для выдвижения гипотезы о простейшем потоке при обработке экспериментальных данных о наблюдаемых потоках.

Закон распределения остатка интервала T_1 между событиями в простейшем потоке не зависит от прошедшего времени T (рис. 8.2):

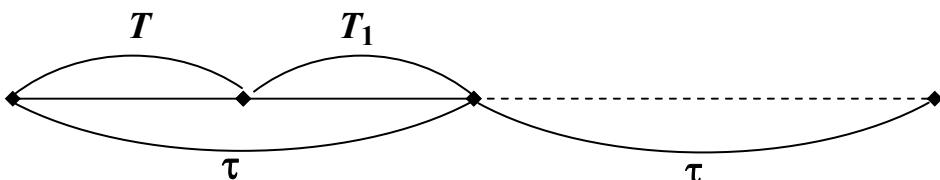


Рис. 8.2

$$P(\tau < t) = 1 - e^{-\lambda t},$$

$$\begin{aligned} F(T_1 < t) &= P[T_1 \leq t / \tau \geq T] = \frac{P[T_1 \leq t, \tau \geq T]}{P[\tau \geq T]} = \\ &= \frac{P[T \leq \tau \leq T+t]}{P[\tau \geq T]} = \frac{1 - e^{-\lambda(T+t)} - 1 + e^{-\lambda T}}{e^{-\lambda T}} = 1 - e^{-\lambda t}. \end{aligned}$$

Остаток интервала T_1 в простейшем потоке распределен так же, как и полный интервал τ . Этот результат, известный также, как парадокс остаточного времени, является наиболее полным выражением сущности свойства отсутствия последействия.

6. Закон распределения суммы интервалов $\tau = \sum_{i=1}^r \tau_i$, где каждый интервал имеет экспоненциальное распределение с параметром λ , равен:

$$f_r(\tau) = \frac{\lambda(\lambda\tau)^{r-1}}{(r-1)!} e^{-\lambda\tau}, \quad \tau \geq 0,$$

а численные характеристики определяются как:

$$M[\tau] = \frac{r}{\lambda}, \quad D[\tau] = \frac{r}{\lambda^2}.$$

Данный закон описывает распределение между событиями в потоке Эрланга r -го порядка.

7. Преобразования простейшего потока.

A. Регулярное прореживание с параметром r :

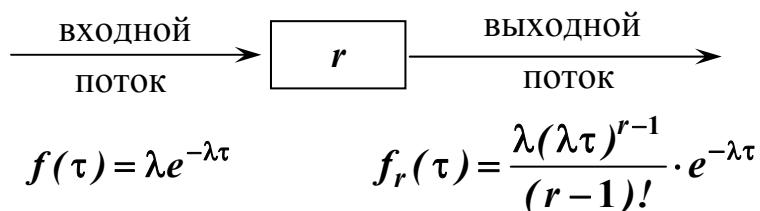


Рис. 8.3

B. Случайное просеивание с параметром p :

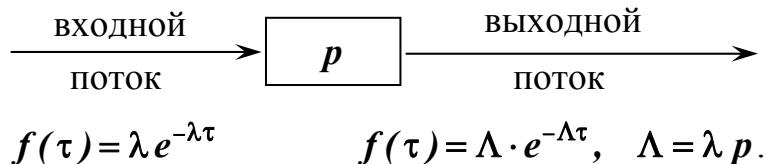


Рис. 8.4

В. Суммирование потоков:

$$f(\tau_i) = \lambda_i e^{-\lambda_i \tau_i} \quad \left| \begin{array}{l} \lambda_1 \rightarrow \\ \lambda_2 \rightarrow \\ \vdots \\ \lambda_i \rightarrow \\ \vdots \\ \lambda_n \rightarrow \end{array} \right. \quad \Lambda = \sum_i^n \lambda_i, \quad f(\tau) = \Lambda e^{-\Lambda \tau}.$$

Рис. 8.5

8. Моделирование влияния последействия в потоке.

Поток Эрланга имеет характеристики:

$$f_r(\tau) = \frac{\lambda(\lambda\tau)^{r-1}}{(r-1)!} \cdot e^{-\lambda\tau}; \quad M[\tau] = \frac{r}{\lambda}, \quad D[\tau] = \frac{r}{\lambda^2}.$$

$$\text{Пусть } M[\tau] = \frac{r}{\lambda} = \text{const} = c_0. \text{ Если } \lambda = \frac{r}{c_0}, \text{ то } D[\tau] = \frac{r \cdot c_0^2}{r^2} = \frac{c_0^2}{r}.$$

Отсюда следует:

- a) при $r \rightarrow \infty \quad D[\tau] = 0$;
- б) при $r \rightarrow 1 \quad D[\tau] = c_0^2$.

В случае а) имеем максимальное последействие — приближаемся к регулярному потоку.

В случае б) имеем отсутствие последействия, т. е. простейший поток.

8.2.3. Потоки Пальма. Рекуррентные потоки

Потоки Пальма — потоки с независимыми интервалами, распределенными по произвольным законам (рис. 8.6): $F_1(t), F_2(t), \dots, F_i(t), \dots$. Рекуррентные потоки: $F_i(t) = F(t)$.

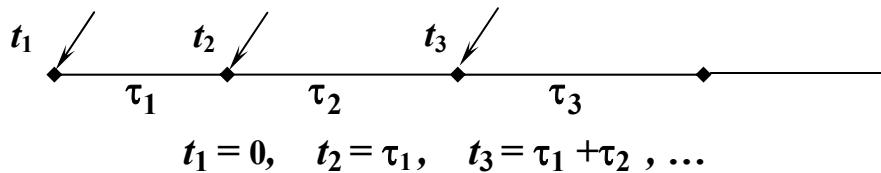


Рис. 8.6

Пусть $n(t)$ — число событий рекуррентного потока, появившихся до момента t , т. е. на интервале $[0, t]$. Математическое ожидание $M[n(t)]$ называется функцией восстановления и обозначается $H(t)$:

$$H(t) = M[n(t)] = \sum_{n=1}^{\infty} n \cdot P[n(t) = n]. \quad (8.11)$$

Показано в [30], что:

$$H^*(s) = \frac{F^*(s)}{1 - F^*(s)}, \quad (8.12)$$

где $F^*(s)$ и $H^*(s)$ — преобразования Лапласа $F(t)$ и $H(t)$.

Для иллюстрации рассмотрим пример.

Простейший поток:

$$F(t) = 1 - e^{-\lambda t}, \quad F^*(s) = \frac{\lambda}{\lambda + s}, \quad \text{т. е. } H^*(s) = \frac{\lambda}{s} \quad \text{и} \quad H(t) = \lambda \cdot t.$$

Рекуррентный поток в задачах теории надежности выглядит следующим образом (рис. 8.7):

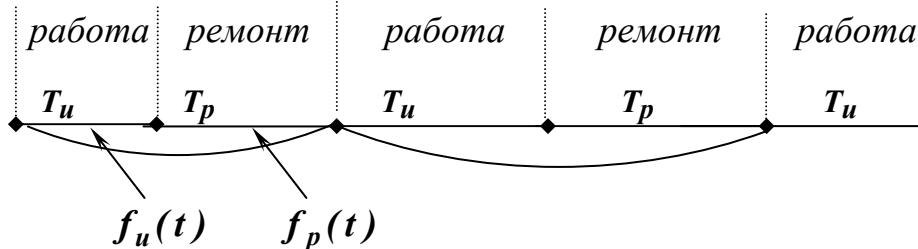


Рис. 8.7

$$T_u = T_u + T_p, \quad f_u(t) \rightarrow F_u(s), \quad f_p(t) \rightarrow F_p(s),$$

где T_u — длительность цикла.

Случайные величины T_u и T_p независимы, поэтому:

$$F_u^*(s) = F_u^*(s) * F_p^*(s) \quad \text{и} \quad H_u^*(s) = \frac{F_u^*(s) \cdot F_p^*(s)}{1 - F_u^*(s) \cdot F_p^*(s)}, \quad (8.13)$$

где $H_u^*(s)$ — изображение по Лапласу скорости изменения среднего числа циклов «работа – ремонт».

Как было показано ранее, простейший поток обладает свойством отсутствия последействия в любой момент, т. е. по всей оси времени. Что касается рекуррентных потоков и вообще потоков Пальма, как показано в

[30], они обладают последействием всюду, кроме точек, в которых появляются события потока.

Время, произошедшее с момента появления предыдущего события, влияет на закон распределения интервала до ближайшего следующего события, но не влияет на закон распределения последующих полных интервалов. Область проявления последействия в потоках Пальма ограничена пределами одного интервала. Это объясняет происхождение термина «потоки с ограниченным последействием», который иногда используется применительно к потокам Пальма.

8.3. ОБЩИЕ РЕЗУЛЬТАТЫ ТЕОРИИ СМО

В данном разделе приводятся результаты, справедливые для уставившегося режима в СМО независимо от законов распределения длительности интервалов входящего потока, а также длительности обслуживания, т. е. справедливые для систем типа **G/G/1** и **G/G/K**.

Коэффициент использования ρ для систем **G/G/1** определяется как:

$$\rho = \lambda \cdot \bar{x} = \frac{\lambda}{\mu}, \quad (8.14)$$

т. е. произведение средней скорости λ поступления требований в систему на среднее время обслуживания $\bar{x} = 1/\mu$ каждого требования.

Аналогично, для многоканальных систем типа **G/G/K** данное определение приобретает вид:

$$\rho_c = \frac{\lambda \cdot \bar{x}}{K} = \frac{\lambda}{K \cdot \mu}. \quad (8.15)$$

Стабильной или устойчивой системой (т. е. такой системой, которая имеет конечные средние задержки и длины очередей) является система, для которой:

$$\rho < 1 \text{ или } \rho_c < 1 \quad (8.16)$$

Чем ближе ρ или ρ_c к единице, тем больше очереди и время ожидания, тем более существенно отражается изменение характеристик СМО при изменении средней нагрузки.

Среднее время \bar{t}_c пребывания требования в обслуживающей системе определяется как:

$$\bar{t}_c = \bar{t}_{ож} + \bar{x}, \quad (8.17)$$

где \bar{x} — среднее время обслуживания, $\bar{t}_{ож}$ — среднее время ожидания в очереди.

Одной из самых замечательных формул теории массового обслуживания является закон Литтла [29]:

$$\bar{j} = \lambda \cdot \bar{t}_c, \quad (8.18)$$

т. е. среднее число требований в системе равно произведению средней скорости поступления требований на среднее время пребывания требования в системе для случая разомкнутых систем массового обслуживания.

Аналогично для среднего числа требований, ожидающих в очереди \bar{n}_0 и среднего времени ожидания в очереди имеем:

$$\bar{n}_0 = \lambda \cdot \bar{t}_{ож}. \quad (8.19)$$

Интерпретация данных соотношений для конечного источника, т. е. для замкнутой системы выглядит следующим образом:

$$\begin{aligned} \bar{j} &= \lambda(n - \bar{j}) \cdot \bar{t}_c, \\ \bar{n}_0 &= \lambda(n - \bar{j}) \cdot \bar{t}_{ож}. \end{aligned} \quad (8.20)$$

Для многоканальных систем с бесконечным источником имеем:

$$\bar{j} = \bar{n}_0 + K \rho_c = \lambda \bar{t}_{ож} + \rho, \quad (8.21)$$

а для многоканальных систем с конечным источником:

$$\bar{j} = \bar{\lambda}(n - \bar{j})\bar{t}_{ож} + \bar{K}_3, \quad (8.22)$$

где \bar{K}_3 — среднее число занятых каналов в обслуживающей системе из K каналов.

8.4. ПРИМЕРЫ ФОРМАЛИЗАЦИИ ЗАДАЧ В ТЕРМИНАХ СМО

Пример 8.1

Рассматривается многопроцессорная система, составляющая вычислительный комплекс (ВК) обработки информации. Пусть число процессоров равно двум. На вход системы поступает поток задач. Если задача застает оба процессора занятыми, то она теряется. В процессе решения в результате сбоя может возникнуть ошибка. С целью повышения достоверности результатов счета рассматриваются два режима работы.

1. Использование временной избыточности: каждый процессор работает независимо от другого, т. е. решает свою задачу и для повышения дос-

троверности результата использует принцип голосования 2 из 3-х. На каждом из процессоров задача решается два раза. Если результаты совпали, то требование считается обслуженным; в противном случае, если возникла ошибка, задача решается третий раз. В случае совпадения двух результатов решения задачи из трех, требование считается обслуженным, в противном случае — потерянным.

2. Использование структурной избыточности: каждую поступающую задачу оба процессора обрабатывают совместно, т. е. решают параллельно. Если результаты счета на обоих процессорах совпали, то требование считается обслуженным, в противном случае, требование считается потерянным.

Провести сравнительный анализ двух режимов работы.

Рассмотрим формализацию данной задачи в терминах и понятиях СМО.

1. Источник требований — внешняя среда, генерирующая поток задач на обработку в ВК (источник — бесконечный).

2. Буфер для накопления требований о потоке поступающих задач в случае занятости процессоров отсутствует.

3. Система обработки информации, представленная ВК, включает: два канала, работающих автономно в случае временной избыточности; один канал в случае структурной избыточности при совместной параллельной обработке одной задачи двумя процессорами.

4. Дисциплина обслуживания в обоих случаях: первым пришел — первым обслужен.

5. Принимая модель простейшего потока задач, поступающих в систему, а также показательный закон распределения длительности однократной обработки каждой задачи имеем:

- в случае временной избыточности: модель СМО типа $M/E_2/2/∞$;
- при структурной избыточности: модель СМО типа $M/M/1/0/∞$.

При этом для упрощения принято, что в случае временной избыточности длительность обработки каждой задачи в среднем в два раза превосходит длительность обработки для случая структурной избыточности.

6. Указанные модели СМО относятся к классу систем с потерями и поэтому при сравнительном анализе качества обслуживания в качестве основного показателя целесообразно выбрать вероятность потерь P_{nom} .

Пример 8.2

Рассматривается автоматизированная система контроля (АСК), работающая в режиме параллельного выполнения n программ контроля. Каждая программа представляет собой последовательность чередующихся фаз

работы процессора и специализированных внешних функциональных устройств (ФУ) (генераторов, преобразователей, коммутаторов, устройств ввода–вывода, регистраторов, внешних запоминающих устройств), управление которыми ведется от процессора. Средняя длительность фазы $\tau_{\Phi U}$ работы функциональных устройств значительно превосходит среднюю длительность фазы обработки команды в процессоре ($\bar{\tau}_{np}$).

Обосновать целесообразный уровень совмещения в выполнении программ при условиях.

1. Конфликты при обращении к функциональным устройствам не учитываются.

2. Дисциплина обслуживания программ процессором: первый ответил — первым обслужен.

Проведем формализацию данной задачи в терминах и понятиях СМО.

1. Источник требований — конечный и представляет собой совокупность n параллельно выполняемых программ.

2. Обслуживающий прибор — процессор.

3. Дисциплина обслуживания программ: первым пришел — первым обслужен.

4. Принимая модели показательных законов распределения длительностей фаз работы ФУ и процессора $\tau_{\Phi U}, \tau_{np}$, получим модель данной задачи в виде СМО типа $M/M/1/\infty/n$. Знак ∞ означает, что физические ограничения для длины очереди в системе отсутствуют.

При оценке качества совмещения программ необходимо учесть как загрузку процессора (коэффициент загрузки обслуживающего прибора), так и задержки в выполнении программ (среднее время ожидания требований в очереди).

ЗАДАЧИ

Рассмотреть содержательную интерпретацию следующих задач в терминах и понятиях моделей систем массового обслуживания.

1.1. Обслуживание рабочим или бригадой рабочих парка станков на производстве.

1.2. Работа мастерской бытового обслуживания на вокзале.

1.3. Обслуживание разговоров на телефонной станции.

1.4. Работа пользователей за терминалами в дисплейном классе.

1.5. Управление взлетом — посадкой самолетов в аэропорту.

1.6. Обслуживание пациентов в поликлинике.

- 1.7. Погрузка и разгрузка судов в порту.
- 1.8. Работа производственного конвейера.
- 1.9. Обслуживание пассажиров в железнодорожных кассах.
- 1.10. Работа мультипрограммной и мультипроцессорной вычислительной системы.
- 1.11. Прохождение пассажиров через таможню в аэропорту.
- 1.12. Работа автотранспортной фирмы по перевозке грузов.
- 1.13. Обслуживание пассажиров на стоянках такси.
- 1.14. Комплектование штата секретарей-машинисток.
- 1.15. Работа управляющей вычислительной системы.
- 1.16. Обслуживание в столовой (буфете) с самообслуживанием.
- 1.17. Работа телеграфа по доставке телеграмм.
- 1.18. Обслуживание пассажиропотоков в метрополитене.

2. Опишите характер управляющих решений, которые могли бы иметь место в результате проведения анализа каждой из предложенных выше задач.

3. Выберите для указанных задач (п. 1) варианты формирования потока, очереди, закона обслуживания из следующего набора.

- 3.1. Требования поступают в обслуживающую систему по одному.
- 3.2. Требования поступают в систему пачками.
- 3.3. Порядок поступления требований известен заранее.
- 3.4. Порядок поступления требований – случайный.
- 3.5. Источник требований – бесконечный.
- 3.6. Источник требований имеет ограниченную мощность.
- 3.7. В очередь становится каждое из поступивших требований.
- 3.8. Возможен отказ от присоединения к очереди со стороны некоторых из поступивших требований.
- 3.9. Дисциплина обслуживания: первым пришел – первым обслужен.
- 3.10. Дисциплина обслуживания: пришел последним – обслужен первым.
- 3.11. Имеет место случайный порядок обслуживания.
- 3.12. Учет определенной системы приоритетов.
- 3.13. Присоединившиеся к очереди требования не покидают систему до тех пор, пока их не обслужат.
- 3.14. Некоторые из требований могут покинуть систему, не дождавшись пока их обслужат (т. е. ожидают обслуживания лишь ограниченное время).
- 3.15. Имеется единственная очередь.
- 3.16. Имеется несколько параллельных очередей.

- 3.17. Обслуживающие приборы функционируют параллельно.
- 3.18. Обслуживающие приборы расположены последовательно.
- 3.19. Скорость обслуживания не зависит от длины очереди.
- 3.20. Скорость обслуживания зависит от длины очереди.

4. Определите четыре – пять важных операционных характеристик, которые необходимо измерить количественно, с тем, чтобы оценить эффективность и качество работы соответствующих функциональных систем.

9. МАРКОВСКИЕ СИСТЕМЫ МАССОВОГО ОБСЛУЖИВАНИЯ

9.1. МАРКОВСКИЕ СЛУЧАЙНЫЕ ПРОЦЕССЫ И ИХ КЛАССИФИКАЦИЯ

Данный класс случайных процессов достаточно подробно исследован в работах выдающегося русского математика Андрея Андреевича Маркова, поэтому и получил такое название. В последнее время такие понятия как цепи Маркова, марковские процессы, свойства марковости стали достаточно привычными и не сходят со страниц отечественных и иностранных книг, диссертаций и статей.

Случайный процесс $x(t)$ называется марковским, если для любого момента t' при фиксированном значении $x(t')$ (каково бы ни было x) значения процесса $x(t)$ при $t > t'$ не зависят от значений процесса $x(t)$ при $t < t'$.

Иначе такие процессы называются процессами без последействия. Случайный процесс $x(t)$ считается заданным, если для любых t_1, t_2, \dots, t_n известен закон распределения (плотность) $f(x_1, x_2, \dots, x_n)$.

Для марковских процессов характерно задание только плотности $f(x_i / x_{i-1})$, так как $f(x_i / x_1, x_2, \dots, x_{i-1}) = f(x_i / x_{i-1})$.

Тогда:

$$f(x_1, x_2, \dots, x_n) = f(x_n / x_{n-1}) \cdot f(x_{n-1} / x_{n-2}) \cdot \dots \cdot f(x_2 / x_1) \cdot f(x_1). \quad (9.1)$$

Для выделения классификационных групп марковских процессов введем $x \in X, t \in T$, а также вектор $Z = (z_x, z_T)$, где z_x принимает значение $\mathbf{0}$, если множество X дискретно и 1 , если множество X непрерывно. Аналогичные значения принимает z_T в отношении множества T . В соответствии с этим имеем следующие классы процессов:

- 00** — процессы с дискретным числом состояний и дискретным временем (дискретные марковские цепи или последовательности);
- 01** — процессы с дискретным числом состояний и непрерывным временем (непрерывные марковские процессы);
- 10** — процессы с независимыми приращениями;
- 11** — диффузионные процессы.

В дальнейшем нас будут интересовать только первые две группы процессов, так как они активно используются для описания поведения

СМО, хотя это и условно. В [31] показано, что при построении приближенных оценок показателей качества могут использоваться и модели процессов типа **10** и **11**.

9.2. ДИСКРЕТНЫЕ МАРКОВСКИЕ ПОСЛЕДОВАТЕЛЬНОСТИ

В этом случае процесс $x(t)$ переходит в последовательность $x(n) = x_n$, где $n=0, 1, 2 \dots$; $x \in X$, а X — счетное или конечное множество.

Дискретная марковская последовательность считается заданной, если:

- определены вероятности

$$P_{i,j}^{n, n+1} = P\{x_{n+1} = j / (x_n = i)\}, \text{ где } i = 0, 1, 2, \dots; j = 0, 1, 2, \dots;$$

- определены начальные вероятности

$$P\{x_0 = i\} = P_i(0), \quad \sum_i P_i(0) = 1, \quad \text{где } i = 1, 2, \dots.$$

Если переходные вероятности не зависят от n , то $P_{i,j}^{n, n+1} = P_{i,j}$, и такая марковская цепь называется однородной и задается матрицей переходов $P = [P_{i,j}]$ и вектором распределения начального состояния $P^T(0) = [P_1(0) P_2(0) \dots P_i(0) \dots]$. Если множество X конечно, то цепь конечная, если нет, то цепь бесконечная.

При конечном или счетном множестве значений X удобно ввести другую интерпретацию марковской цепи. Сопоставим множеству X множество состояний $S : X \Leftrightarrow S$. Если $x=j$, то имеем состояние S_j . В этом случае марковская цепь представлена последовательностью переходов из состояния S_i в состояние S_j .

Проведем классификацию состояний.

1. Состояния S_i и S_j называются *взаимно-связанными*, если S_i связано с S_j и S_j связано с S_i (не обязательно непосредственно).

2. Состояние S_i называется *возвратным*, если $\sum_{n=1}^{\infty} P_{ii}(n) = 1$,

где $P_{ii}(n)$ — вероятность возврата в i -е состояние на n -м шаге. Если

$\sum_{n=1}^{\infty} P_{ii}(n) < 1$, то S_i — невозвратное состояние. Марковская цепь, со-

состоящая из взаимно-связанных (возвратных) состояний называется эргодической.

3. *Поглощающим* (концевым или тупиковым) называется состояние, которое процесс не может покинуть, но в которое он может перейти хотя бы из одного другого состояния.

4. Марковская цепь, которая содержит хотя бы одно поглощающее состояние, называется *поглощающей*.

9.2.1. Эргодические однородные марковские цепи

Исследование таких цепей обычно проводится в двух режимах: переходном и установившемся.

Переходный режим

В переходном режиме требуется найти вектор вероятностей состояний $P(n)$ в n -й момент дискретного времени, где $n = 1, 2, \dots$, при условии, что начальный вектор $P(0)$ известен. В этом случае при известной матрице переходов P имеем:

$$\begin{aligned} P^T(1) &= P^T(0) \cdot P; \\ P^T(2) &= P^T(1) \cdot P = P^T(0) \cdot P^2; \\ &\dots \\ P^T(n) &= P^T(0) \cdot P^n; \end{aligned} \tag{9.2}$$

При этом возможно также и аналитическое решение для $P(i)$ с использованием Z -преобразования (аналога преобразования Лапласа для решетчатых функций). Умножаем каждое уравнение из системы (9.2) на Z^i , где $i = 1, 2, \dots$ и суммируем, используя определение производящей функции,

$$P^T(z) = \sum_{i=0}^{\infty} z^i \cdot P^T(i), \quad (9.3)$$

получим следующее векторное уравнение:

$$P^T(z) - P^T(0) = z \cdot P^T(z) \cdot P \quad . \quad (9.4)$$

Отсюда при существовании обратной матрицы $[E - z \cdot P]^{-1}$,

где E — единичная матрица, получим:

$$\mathbf{P}^T(\mathbf{z}) = \mathbf{P}^T(\mathbf{0}) [E - \mathbf{z} \cdot \mathbf{P}]^{-1}. \quad (9.5)$$

Используя обратное Z -преобразование, получим оценку вектора $P^T(n)$, где $n = 1, 2, \dots$.

Установившийся режим

Условия существования установившегося режима.

1. Если множество состояний S конечно, то необходимым условием является эргодичность структуры графа переходов.
2. Если S счетно, то необходимым условием является эргодичность графа, а также наличие некоторых дополнительных условий, накладываемых на матрицу переходов P .

Рассмотрим конечный случай, тогда из (9.2) имеем:

$$P^T(n) = P^T(n-1) \cdot P. \quad (9.6)$$

Реализуя предельный переход

$$\lim_{n \rightarrow \infty} P^T(n) = \pi^T, \quad (9.7)$$

$$\lim_{n \rightarrow \infty} P^T(n-1) = \pi^T,$$

получим:

$$\pi^T = \pi^T \cdot P; \quad \pi^T(E - P) = O^T. \quad (9.8)$$

Система (9.8) имеет нетривиальное решение при условии $\sum_i \pi_i = 1$ и

замены одного из уравнений системы (9.8) на нормировочное условие $\sum_i \pi_i = 1$.

9.2.2. Неэргодические цепи с поглощающими состояниями

В этом случае матрица переходов имеет следующую структуру:

$$P = \begin{matrix} k & l \\ k & Q & R \\ l & O & E \end{matrix}, \quad (9.9)$$

где k — число невозвратных состояний, l — число поглощающих состояний, $k + l = n$, Q — матрица переходов в пределах множества невозвратных состояний, R — матрица переходов из невозвратных состояний в поглощающие, E — единичная матрица, O — нулевая матрица.

Исследуем поведение матрицы $P^n = P(\infty)$ при $n \rightarrow \infty$:

$$P^n = \left[\frac{Q^n}{O} \left| \frac{(Q^{n-1} + Q^{n-2} + \dots + Q + E) \cdot R}{E} \right. \right], \quad (9.10)$$

где $(Q^{n-1} + Q^{n-2} + \dots + Q + E) \cdot R = (E - Q)^{-1} \cdot (E - Q^n) \cdot R$,

(аналогия с рядом $\sum_{i=0}^{n-1} q^i = \frac{1-q^n}{1-q}$).

Так как $\lim_{n \rightarrow \infty} Q^n = 0$, то имеем:

$$P(\infty) = \left[\frac{O}{O} \left| \frac{(E - Q)^{-1} \cdot R}{E} \right. \right]. \quad (9.11)$$

Обозначим: $U = (E - Q)^{-1} \cdot R$, $(E - Q) \cdot U = R$, $U = R + Q \cdot U$. При введенном обозначении U получим:

$$P(\infty) = P \cdot P(\infty), \quad (9.12)$$

где $P = \begin{bmatrix} Q & R \\ O & E \end{bmatrix}$ из (9.2), а $P(\infty) = \begin{bmatrix} O & U \\ O & E \end{bmatrix}$.

Разрешая уравнение (9.12), можно также найти матрицу $P(\infty)$.

Для поглощающих цепей матрица $N = (E - Q)^{-1}$ называется в [32] фундаментальнойной, так как позволяет найти целый ряд полезных в исследовании СМО характеристик.

1. Вектор средних времен до выхода в любое из поглощающих состояний:

$$\bar{\tau}_1 = N \cdot \zeta, \text{ где } \zeta = \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}. \quad (9.13)$$

2. Вектор дисперсий времен достижения любого поглощающего состояния:

$$D(\tau) = \bar{\tau}_2 = (2 \cdot N - E) \cdot \bar{\tau}_1 - \tau_{sq}, \quad (9.14)$$

где τ_{sq} — вектор-столбец квадратов элементов вектора $\bar{\tau}_1$.

3. Матрица вероятностей того, что процесс, выйдя из невозвратного состояния S_i , остановится в поглощающем состоянии S_j :

$$B = \{b_{ij}\} = N \cdot R. \quad (9.15)$$

Пример 9.1

Задача с «игрушечным делом» мастером [32].

Мастер открывает новое производство игрушек. Он может находиться в одном из двух состояний S_1, S_2 :

S_1 — игрушка, которую мастер делает, получит большой спрос;

S_2 — игрушка не найдет спроса и мастер экспериментирует с новыми игрушками.

Предполагается, что каждую неделю мастер делает одну новую игрушку. Задана матрица P и вектор начальных состояний $P(0)$:

$$P = \begin{bmatrix} 1/2 & 1/2 \\ 2/5 & 3/5 \end{bmatrix}, \quad P(0) = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

Переходный режим:

$$P^T(n) = P^T(0) \cdot P^n.$$

Вероятности последовательных состояний мастера, начавшего с удачной игрушки, представлены в табл. 9.1, а для неудачной игрушки — в табл. 9.2.

Таблица 9.1

n	0	1	2	3	4	5
$P_1(n)$	1	0,5	0,45	0,445	0,4445	0,44445
$P_2(n)$	0	0,5	0,55	0,555	0,5555	0,55555

Таблица 9.2

n	0	1	2	3	4	5
$P_1(n)$	0	0,4	0,45	0,445	0,4445	0,44444
$P_2(n)$	1	0,6	0,56	0,556	0,5556	0,55556

Установившийся режим:

$$\pi^T = \pi^T \cdot P; \quad \sum_i \pi = 1.$$

Имеем в данном случае:

$$\begin{cases} \pi_1 = \frac{1}{2}\pi_1 + \frac{2}{5}\pi_2; \\ \pi_2 = \frac{1}{2}\pi_1 + \frac{3}{5}\pi_2; \end{cases} \quad \pi_1 + \pi_2 = 1.$$

Решая приведенную систему уравнений, получим:

$$\pi_1 = 4/9, \quad \pi_2 = 5/9.$$

Пример 9.2

Задана поглощающая марковская цепь со следующим графом переходов:

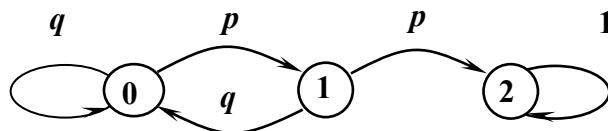


Рис. 9.1

Матрица переходов P выглядит следующим образом:

$$P = \left[\begin{array}{cc|c} q & p & 0 \\ q & 0 & p \\ \hline 0 & 0 & 1 \end{array} \right] = \left[\begin{array}{c|c} Q & R \\ \hline O & E \end{array} \right].$$

Фундаментальная матрица N определяется следующим образом:

$$N = [E - Q]^{-1} = \left[\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \begin{pmatrix} q & p \\ q & 0 \end{pmatrix} \right]^{-1} = \left[\begin{matrix} 1-q & -p \\ -q & 1 \end{matrix} \right]^{-1} = \left[\begin{matrix} \frac{1}{p^2} & \frac{p}{p^2} \\ \frac{q}{p^2} & \frac{(1-q)}{p^2} \end{matrix} \right].$$

Матрица U предельных вероятностей выхода в поглощающее состояние равна:

$$U = (E - Q)^{-1} \cdot R = \left[\begin{matrix} \frac{1}{p^2} & \frac{p}{p^2} \\ \frac{q}{p^2} & \frac{(1-q)}{p^2} \end{matrix} \right] \cdot \begin{pmatrix} 0 \\ p \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$

Очевидно, вероятность попадания в поглощающее состояние равна единице независимо от того, каким было начальное состояние системы.

Вычислим усредненные характеристики цепи:

- вектор средних времен до попадания в поглощающее состояние равен:

$$\bar{\tau}_1 = N \cdot \xi = \begin{pmatrix} \frac{1}{p^2} & \frac{p}{p^2} \\ \frac{q}{p^2} & \frac{(1-q)}{p^2} \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{1+p}{p^2} \\ \frac{1}{p^2} \end{pmatrix};$$

- вектор дисперсий времен до попадания в поглощающее состояние равен:

$$D(\tau) = (2 \cdot N - E) \cdot \bar{\tau}_1 - \tau_{sq} =$$

$$= \left[2 \cdot \begin{pmatrix} \frac{1}{p^2} & \frac{p}{p^2} \\ \frac{q}{p^2} & \frac{1-q}{p^2} \end{pmatrix} - \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right] \cdot \begin{pmatrix} \frac{1+p}{p^2} \\ \frac{1}{p^2} \end{pmatrix} - \begin{pmatrix} \frac{(1+p)^2}{p^4} \\ \frac{1}{p^4} \end{pmatrix} = \begin{pmatrix} \frac{1-p^3+2pq}{p^4} \\ \frac{1+2pq-p^2}{p^4} \end{pmatrix}.$$

9.3. ОБЩИЕ УРАВНЕНИЯ НЕПРЕРЫВНОГО МАРКОВСКОГО ПРОЦЕССА

Для описания поведения системы в классе непрерывных марковских процессов необходимо:

- определить понятие состояния;
- составить полное множество состояний, в которых может находиться система;
- составить график состояний, т. е. указать пути возможных переходов системы из состояния в состояние;
- указать, в каком состоянии находится система в начальный момент времени, или задать распределение начальных состояний;
- для каждого возможного перехода указать соответствующую интенсивность перехода $\lambda_{i,j}(t)$ потока событий, переводящих систему из состояния i в состояние j :

$$\lambda_{i,j}(t) = \lim_{\Delta t \rightarrow 0} \frac{P_{i,j}(t, t + \Delta t)}{\Delta t}. \quad (9.16)$$

В дальнейшем рассматриваются только случаи, когда $\lambda_{i,j}(t) = \lambda$, т. е. интенсивности переходов не зависят от времени, что справедливо для

однородных марковских процессов. Общая система дифференциальных уравнений, описывающая поведение марковского процесса, выглядит следующим образом [43]:

$$P'_j(t) = - \left(\sum_{\substack{i=1 \\ i \neq j}}^n \lambda_{ji} \right) P_j(t) + \sum_{\substack{i=1 \\ i \neq j}}^n P_i(t) \lambda_{ij}, \quad (9.17)$$

где i, j — индексы состояния, $j = 0, 1, \dots, n$, $i = 0, 1, \dots, n$; $P_j(t)$ — вероятность того, что система в момент t находится в состоянии j .

Данная система дифференциальных уравнений составляется по графу состояний. Левая часть каждого уравнения содержит производную вероятности состояния, а правая — столько членов, сколько дуг связано с данным состоянием.

Если дуга направлена из состояния, соответствующий член имеет знак минус; если дуга направлена в состояние — знак плюс. Каждый член равен произведению интенсивности перехода, соответствующей данной стрелке, и вероятности того состояния, из которого выходит дуга.

При решении данной системы дифференциальных уравнений необходимо учитывать, что

$$\sum_{j=1}^n P_j(t) = 1,$$

т. е. одно из уравнений отбрасывается.

Если число состояний конечно и из каждого состояния графа можно перейти за то или иное число шагов в любое другое, то существует установившийся режим, т. е. предельные вероятности состояний:

$$P_j = \lim_{t \rightarrow \infty} P_j(t), \quad j = \overline{0, n},$$

причем их значения не зависят от начального состояния системы.

Система дифференциальных уравнений (9.17) переходит в систему линейных алгебраических уравнений, решение которой с учетом условия $\sum_{j=0}^n P_j = 1$ позволяет найти все предельные вероятности P_j .

Для некоторых видов графов, описывающих весьма широкий класс поведения систем, удается снизить размерность системы уравнений (9.17) при анализе характеристик в установившемся режиме.

Пусть график динамики системы выглядит, как показано на рис. 9.2.

Использование обычного способа расчета стационарных вероятностей состояний приводит к системе (9.17) с числом уравнений, равным $(m+1) \times (n+1)$. Агрегирование, т. е. объединение состояний, позволяет перейти к системе $(n+1)$ алгебраических уравнений в матричной форме.

Для каждого i -го агрегированного состояния ведем транспонированный вектор-строку $P_i^T, i = \overline{0, m}$:

$$P_i^T = (p_{i0}, p_{i1}, \dots, p_{in}).$$

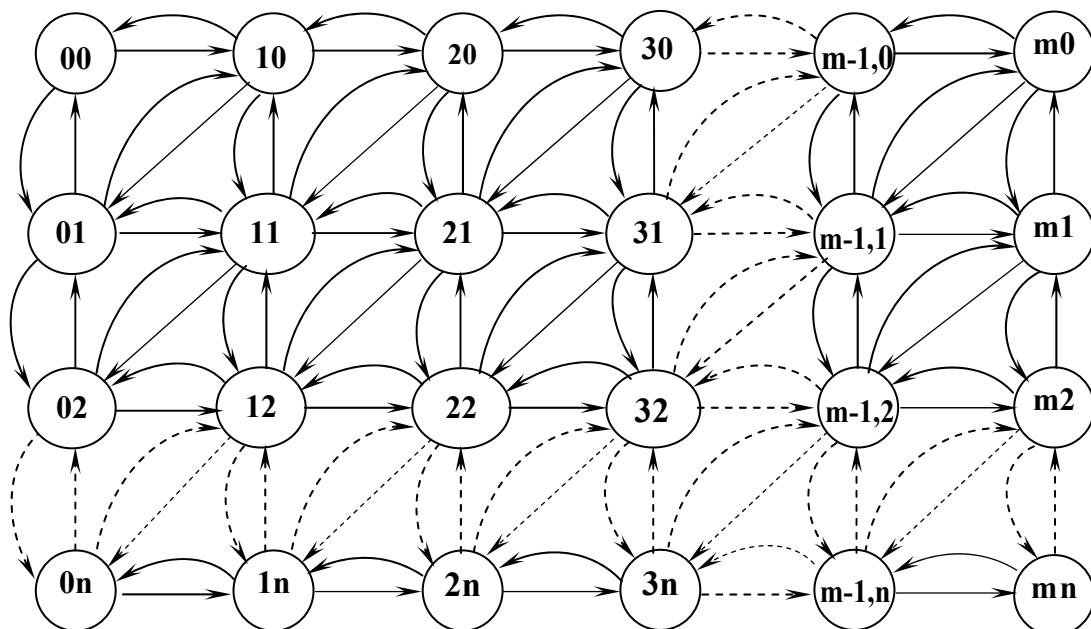


Рис. 9.2

Тогда обобщенная система алгебраических уравнений в случае установившегося режима выглядит следующим образом:

$$\left. \begin{aligned} A_0 P_0 + C_1 P_1 &= 0, \\ B_0 P_0 + A_1 P_1 + C_2 P_2 &= 0, \\ \dots & \\ B_{i-1} P_{i-1} + A_i P_i + C_{i+1} P_{i+1} &= 0, \\ \dots & \\ B_{m-2} P_{m-2} + A_{m-1} P_{m-1} + C_m P_m &= 0, \\ B_{m-1} P_{m-1} + A_m P_m &= 0. \end{aligned} \right\} \quad (9.18)$$

Коэффициентами системы уравнений (9.18) служат матрицы интенсивностей переходов A_i, B_i, C_i размерности $(n+1) \times (n+1)$. Принцип построения матриц A_i, B_i, C_i для графа состояний регулярной структуры типа «решетка» излагается в Приложении 3. Отметим, что часто для рассматриваемого графа, по крайней мере, одна из матриц B_i или C_i при всех i может оказаться диагональной. Предположим, что матрицы C_i диагональные. Тогда ищем решение системы уравнений (9.18) в виде:

$$P_i = W_i \cdot P_0, \quad (9.19)$$

где W_i — некоторые неизвестные матрицы.

Подставляя (9.19) в (9.18), получим рекуррентные соотношения для вычисления матриц:

$$\begin{cases} W_0 = E, & W_1 = -C_1^{-1} A_0, \\ W_2 = -C_2^{-1} (A_1 W_1 + B_0 W_0), \dots, \\ W_{i+1} = -C_{i+1}^{-1} (A_i W_i + B_{i-1} W_{i-1}), \dots, & i = \overline{1, m-1}, \end{cases} \quad (9.20)$$

где E — единичная матрица,

а последнее уравнение системы (9.18) перепишем, введя матрицу W^* :

$$B_{m-1} P_{m-1} + A_m P_m = (B_{m-1} W_{m-1} + A_m W_m) P_0 = W^* P_0, \quad (9.21)$$

причем:

$$W^* P_0 = O. \quad (9.22)$$

Для определения P_0 из (9.22) необходимо учесть условие нормировки:

$$\mathbf{1}^T \cdot \sum_{i=0}^m W_i P_0 = 1, \quad (9.23)$$

где $\mathbf{1}^T = (\underbrace{1, 1, 1, \dots, 1}_{n+1})$.

Составим теперь неоднородную систему уравнений с матрицей коэффициентов W , состоящей из элементов матрицы W^* , с первой строкой, замененной на элементы вектора $I^T \cdot \sum_{i=0}^n W_i$, и вектором L свободных

членов $L^T = \underbrace{(1, 0, 0 \dots 0)}_{n+1}$. Тогда:

$$W \cdot P_0 = L. \quad (9.24)$$

Матрица W невырожденная, так как получена из исходной системы уравнений (9.18) методом последовательного исключения неизвестных и заменой одного уравнения на нормировочное условие. Таким образом, из (9.24) находим P_0 , а далее, используя (9.19) или (9.18), можно найти всё распределение вероятностей.

Предположим теперь, что B_i — диагональные. В этом случае решение для P_i ищем в виде:

$$P_i = V_i P_m, \quad (9.25)$$

где V_i — матрицы той же размерности $(n+1) \times (n+1)$.

Тогда для V_i справедливы следующие рекуррентные соотношения:

$$\begin{cases} V_m = E, & V_{m-1} = -B_{m-1}^{-1} A_m, \\ \dots, \\ V_{i-1} = -B_{i-1}^{-1} (A_i V_i + C_{i+1} V_{i+1}), & i = \overline{1, m-1}, \\ \dots, \\ V^* = -A_0 V_0 + C_1 V_1. \end{cases} \quad (9.26)$$

$$V^* P_m = 0. \quad (9.27)$$

Условие нормировки в этом случае выглядит следующим образом:

$$1^T \cdot \sum_{i=0}^m V_i P_m = 1. \quad (9.28)$$

Аналогично, как и в первом случае, заменив в (9.27) первую строку соотношением (9.28), получим систему уравнений для определения P_m , а следовательно, используя (9.25), и всех матриц P_i . Если оказывается, что обе матрицы B_i , C_i диагональные, то можно выбрать любой из алгоритмов. Обратимся к примеру.

Пример 9.3

Рассматривается двухфазная система периферийных технических средств, входящая в комплекс автоматизированных систем управления обслуживанием.

Первая фаза включает накопитель емкостью n_1 и K_1 каналов обслуживания, вторая — K_2 каналов и накопитель емкостью $n_2 = K_1 - K_2 \geq 0$.

В систему поступает пуассоновский поток требований с интенсивностью λ и обслуживается в каналах 1-й фазы с интенсивностью μ_1 , 2-й фазы — с интенсивностью μ_2 , как показано на рис. 9.3.

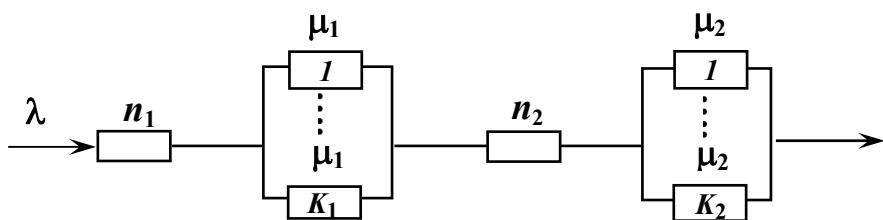


Рис. 9.3

Если при этом на второй фазе имеется свободный канал, требование, обслуженное в первой фазе, немедленно переходит на этот канал, а канал первой фазы принимает, но не обслуживает новые требования до тех пор, пока обслуженное им требование не покинет вторую фазу.

Так как перед прибором второй фазы имеется накопитель емкостью $n_2 = K_1 - K_2$, то потери внутри системы невозможны.

Построим математическую модель функционирования данной системы.

Пространство состояний марковского процесса:

$$Z = [(i, j), i = \overline{0, K_1 + n_1}; j = \overline{0, K_2 + n_2}],$$

где i — число требований на первой фазе, j — число требований на второй фазе. Граф марковского процесса для $n_1 = 2$; $K_1 = 2$; $n_2 = 1$; $K_2 = 1$ показан на рис. 9.4.

Рассмотрим более подробно схему вычислений предельных вероятностей для случая $n_1 = 1$, $K_1 = 1$, $K_2 = 1$.

Граф процесса представлен на рис. 9.5.

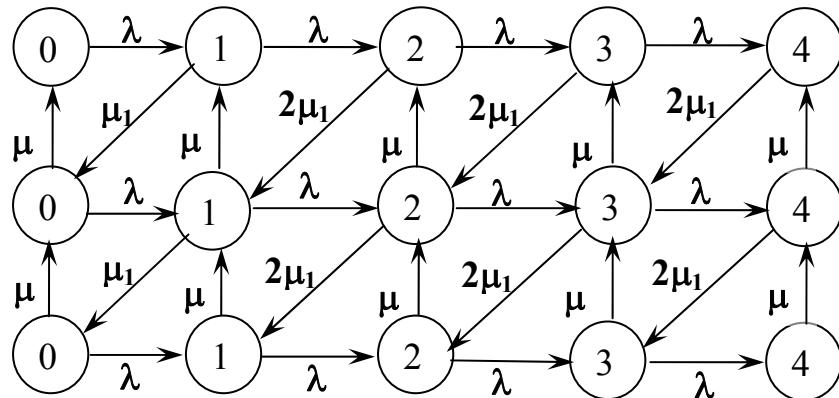


Рис. 9.4

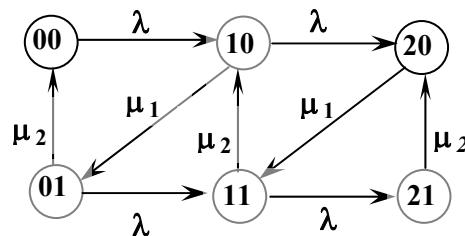


Рис. 9.5

Введем вектора $P_i, i = \overline{0, 2}$ состояний: $P_i^T = (p_{i0}, p_{i1})$.

Система уравнений (9.18) для данного примера выглядит следующим образом:

$$\underbrace{\begin{bmatrix} -\lambda & \mu_2 \\ 0 & -(\lambda + \mu_2) \end{bmatrix}}_{A_0} \cdot P_0 + \underbrace{\begin{bmatrix} 0 & 0 \\ \mu_1 & 0 \end{bmatrix}}_{C_1} \cdot P_1 = O;$$

$$\underbrace{\begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}}_{B_0} \cdot P_0 + \underbrace{\begin{bmatrix} -(\lambda + \mu_1) & \mu_2 \\ 0 & -(\lambda + \mu_2) \end{bmatrix}}_{A_1} \cdot P_1 + \underbrace{\begin{bmatrix} 0 & 0 \\ \mu_1 & 0 \end{bmatrix}}_{C_2} \cdot P_2 = O;$$

$$\underbrace{\begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}}_{B_1} \cdot P_1 + \underbrace{\begin{bmatrix} -\mu_1 & \mu_2 \\ 0 & -\mu_2 \end{bmatrix}}_{A_2} \cdot P_2 = O.$$

Так как матрицы B_i диагональные, ищем решение в виде:

$$P_i = V_i P_2, \quad i = 0, 1, 2, \quad V_2 = E,$$

где E — единичная матрица.

Определим матрицы V_0, V_1 :

$$\begin{aligned} V_1 &= -B_1^{-1} A_2 = -\begin{bmatrix} 1/\lambda & 0 \\ 0 & -1/\lambda \end{bmatrix} \cdot \begin{bmatrix} -\mu_1 & \mu_2 \\ 0 & -\mu_2 \end{bmatrix} = \begin{bmatrix} \mu_1/\lambda & -\mu_2/\lambda \\ 0 & \mu_2/\lambda \end{bmatrix}; \\ V_0 &= -B_0^{-1} \cdot (A_1 \cdot V_1 + C_2 \cdot V_2) = \\ &= -\begin{bmatrix} 1/\lambda & 0 \\ 0 & 1/\lambda \end{bmatrix} \cdot \left\{ \begin{bmatrix} -(\lambda + \mu_1) & \mu_2 \\ 0 & -(\lambda + \mu_2) \end{bmatrix} \cdot \begin{bmatrix} \frac{\mu_1}{\lambda} & -\frac{\mu_2}{\lambda} \\ 0 & \frac{\mu_2}{\lambda} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \mu_1 & 0 \end{bmatrix} \right\} = \\ &= \begin{bmatrix} \frac{(\lambda + \mu_1) \cdot \mu_1}{\lambda^2} & \frac{-(\lambda + \mu_1) \cdot \mu_2 + \mu_2^2}{\lambda^2} \\ -\frac{\mu_1}{\lambda} & \frac{(\lambda + \mu_2) \cdot \mu_2}{\lambda} \end{bmatrix}. \end{aligned}$$

Примем в дальнейшем, что $\lambda = 1, \mu_1 = 2, \mu_2 = 1$. Тогда:

$$V_1 = \begin{bmatrix} 2 & -1 \\ 0 & 1 \end{bmatrix}, \quad A_0 = \begin{bmatrix} -1 & 1 \\ 0 & 2 \end{bmatrix}, \quad V_0 = \begin{bmatrix} 6 & -4 \\ -2 & 2 \end{bmatrix}, \quad C_1 = \begin{bmatrix} 0 & 0 \\ 2 & 0 \end{bmatrix}.$$

Первое матричное уравнение приобретает следующий вид:

$$V^* P_2 = (A_0 V_0 + C_1 V_1) P_2 = \begin{bmatrix} -8 & 6 \\ 8 & -6 \end{bmatrix} P_2 = 0. \quad (9.29)$$

Условие нормировки:

$$(1, 1)^T [V_2 P_2 + V_1 P_2 + V_0 P_2] = 1,$$

т. е.

$$(1,1)^T \left\{ \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} P_2 + \begin{bmatrix} 2 & -1 \\ 0 & 1 \end{bmatrix} P_2 + \begin{bmatrix} 6 & -4 \\ -2 & 2 \end{bmatrix} P_2 \right\} = 1.$$

В результате преобразований получим:

$$7P_{20} - P_{21} = 1. \quad (9.30)$$

Вычеркнув первое уравнение в (9.29) и, заменив его на (9.30), получим систему двух уравнений, разрешение которой дает значение вектора P_2 :

$$P_2 = \begin{pmatrix} 3/17 \\ 4/17 \end{pmatrix}.$$

Тогда остальные векторы P_1 и P_0 определяются в соответствии с матрицами V_0 и V_1 :

$$P_1 = \begin{bmatrix} 2 & -1 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 3/17 \\ 4/17 \end{bmatrix} = \begin{bmatrix} 2/17 \\ 4/17 \end{bmatrix};$$

$$P_0 = \begin{bmatrix} 6 & -4 \\ -2 & 2 \end{bmatrix} \cdot \begin{bmatrix} 3/17 \\ 4/17 \end{bmatrix} = \begin{bmatrix} 2/17 \\ 2/17 \end{bmatrix}.$$

Для рассматриваемой задачи вероятность потерь требований определяется вероятностью P_{21} , равной $4/17$.

Пример 9.4

Рассматривается вычислительная система, представленная на рис. 9.6.

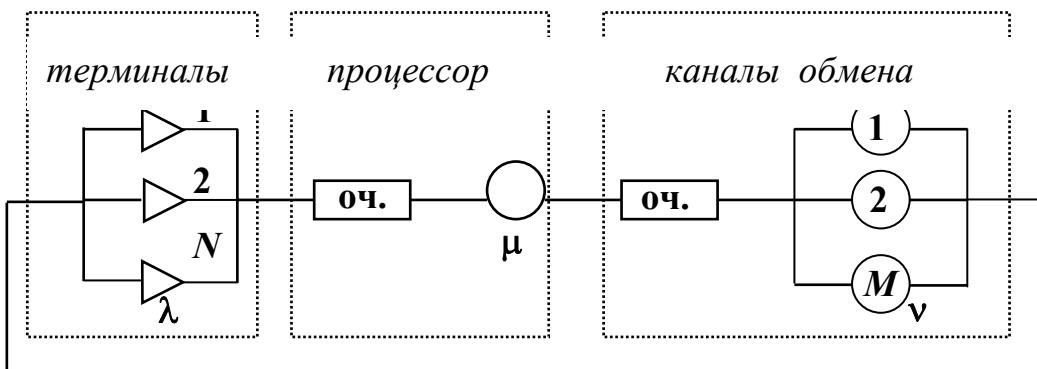


Рис. 9.6

Она включает:

- терминалы (среднее время обдумывания результатов $\bar{T}_{\text{обо}} = 1/\lambda$);
- процессор (среднее время решения задачи в процессоре $\bar{T}_{\text{реш}} = 1/\mu$);
- каналы обмена с памятью, включающие селекторные каналы и накопители на магнитных дисках (среднее время обмена $\bar{T}_{\text{обм}} = 1/v$).

Найти среднее время реакции (время пребывания заявки в системе). Будем считать, что случайные величины $T_{\text{обо}}, T_{\text{реш}}, T_{\text{обм}}$ подчиняются показательному закону распределения с соответствующими параметрами.

Для описания поведения системы в классе марковских процессов примем следующее определение состояния:

$$J(t) = [j_1(t) \ j_2(t)],$$

где $j_1(t)$ — число заявок на терминальной фазе (число пользователей, получивших ответ системы и непосредственно работающих за монитором); $j_2(t)$ — число заявок на процессорной фазе. Тогда число заявок на фазе каналов обмена равно $N - j_1(t) - j_2(t)$.

Граф соответствующего марковского процесса представлен на рис. 9.7 при значениях параметров системы $M = 2$, $N = 3$.

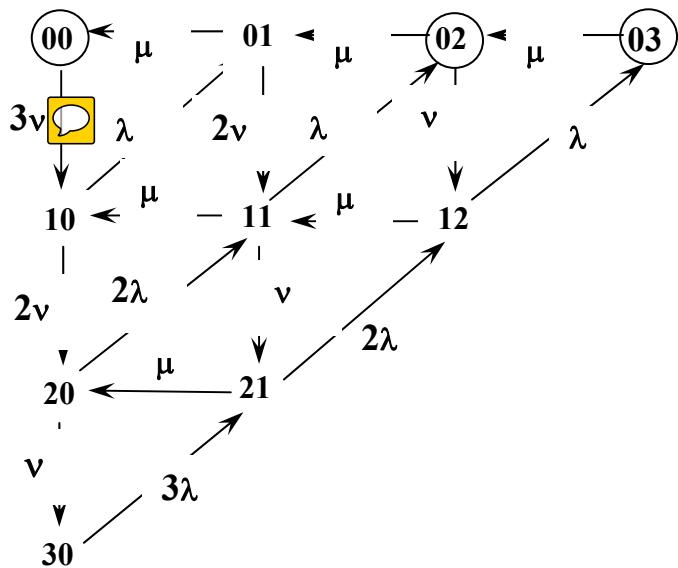


Рис. 9.7

В результате решения соответствующей системы уравнений типа

(9.17) находим вероятности состояний P_{ij} .

Среднее число занятых терминалов равно:

$$\bar{N}_{\text{терм}} = 1 \cdot (P_{10} + P_{11} + P_{12}) + 2 \cdot (P_{20} + P_{21}) + 3 \cdot P_{30}.$$

Используя закон Литтла, найдем среднее время реакции:

$$\bar{t}_p = \frac{N - \bar{N}_{\text{терм}}}{\bar{N}_{\text{терм}} \cdot \lambda}.$$

9.4. ПРОЦЕСС ГИБЕЛИ И РАЗМНОЖЕНИЯ

Граф поведения марковского процесса такого вида представляет собой цепь состояний, в которой каждое из состояний ($j = 1, n - 1$) связано прямой и обратной связями с каждым из соседних состояний, а состояния $j = n$ и $j = 0$ — только с одним из соседних состояний. Если n бесконечно, то цепь может быть продолжена на неограниченное число звеньев в направлении увеличения.

Общая система дифференциальных уравнений, описывающая динамику такого процесса, имеет следующий вид [43]:

$$\begin{aligned} P'_0(t) &= -\lambda_0 P_0(t) + \mu_1 P_1(t), \quad j = 0; \\ &\dots\dots\dots\dots\dots\dots\dots\dots; \\ P'_j(t) &= -(\lambda_j + \mu_j) P_j(t) + \lambda_{j-1} P_{j-1}(t) + \mu_{j+1} P_{j+1}(t), \quad j \geq 1; \quad (9.31) \\ &\dots\dots\dots\dots\dots\dots\dots\dots; \\ P_0(0) &= 1, \quad P_0(j) = 0, \quad j = 1, 2, \dots . \end{aligned}$$

Вопросы выяснения существования и единственности решения полученной системы дифференциальных уравнений подробно рассмотрены в [30].

Для того, чтобы найденные вероятности $P_j(t)$ удовлетворяли условию: $\sum_{j=1}^{\infty} P_j(t) = 1$, достаточно иметь ряд $\sum_{j=1}^{\infty} \prod_{k=1}^j \mu_k / \lambda_{k-1}$ расходящимся.

Если при этом оказывается, что ряд

$$\sum_{j=1}^{\infty} \prod_{k=1}^j \lambda_{k-1}/\mu_k$$

сходится, то существуют пределы вероятностей

$$P_j = \lim_{t \rightarrow \infty} P_j(t), \quad j = 0, 1, 2, \dots$$

Последнее условие, в частности, выполняется во всех случаях, когда, начиная с некоторого k , выполнено неравенство $\lambda_{k-1}/\mu_k < 1$, что для большинства практических случаев выполняется.

Предельные вероятности определяются следующим образом:

$$P_0 = \left[1 + \sum_{j=1}^{\infty} \prod_{k=1}^j \lambda_{k-1}/\mu_k \right]^{-1}, \quad P_j = P_0 \prod_{k=1}^j \lambda_{k-1}/\mu_k. \quad (9.32)$$

Пример 9.5

Рассматривается двухпроцессорная управляющая система, на вход которой поступают три простейших входящих потока с интенсивностями:

$$\lambda_1 = 6c^{-1}, \quad \lambda_2 = 15c^{-1}, \quad \lambda_3 = 9c^{-1}.$$

Процессоры считаются однотипными с быстродействием $B = 50 \cdot 10^3$ опер/с.

Обслуживание требования заключается в выполнении на любом из процессоров соответствующей прикладной программы, причем средняя трудоемкость всех трех прикладных программ принимается примерно одинаковой $\bar{\Theta} = 2,5 \cdot 10^3$ операций.

От заявки к заявке конкретная трудоемкость меняется случайным образом. Будем для простоты считать закон ее распределения экспоненциальным.

Для хранения заявок, которые не могут быть немедленно поставлены на обслуживание, выделена буферная зона памяти емкостью в восемь ячеек; служебная информация об одной заявке занимает две ячейки.

Время пребывания заявки в системе не должно превышать случайной величины τ_δ , распределенной экспоненциально с математическим ожиданием $\bar{\tau}_\delta = 0,1 \text{ с}$.

Операционная система реализует бесприоритетные дисциплины ожидания и обслуживания. В ее же функции входит удаление “устаревших” заявок из системы. Критерий эффективности управляющей системы

имеет вид:

$$E = \lambda \cdot (e_{OTK} P_{OTK} + e_y P_y) + e_H (K - \bar{K}_3),$$

где e_{OTK} — штраф за отказ СМО принять заявку, $e_{OTK} = 3$ усл.ед./заявка; P_{OTK} — вероятность отказа в обслуживании; e_y — штраф за устаревшие заявки во время обработки ее системой; $e_y = 2$ усл. ед./заявка; P_y — вероятность ухода требования по причине устаревания, e_H — штраф за неиспользование одного канала обслуживания; $e_H = 10$ усл. ед./кан.; \bar{K}_3 — среднее число занятых каналов.

Определить E .

Формализация в терминах и понятиях СМО приводит к тому, что мы имеем дело с разомкнутой системой массового обслуживания **M/M/K/m**, где $K = 2$, $m = 4$.

Интенсивность обслуживания $\mu = B/\Theta = 50 \cdot 10^3 / 2,5 \cdot 10^3 = 20 c^{-1}$.

Интенсивность ухода заявок из очереди и обслуживающего прибора $\alpha = 1/\tau_\delta = 10 c^{-1}$.

Общая интенсивность потока при бесприоритетном обслуживании равна:

$$\lambda = \lambda_1 + \lambda_2 + \lambda_3 = 30 c^{-1},$$

при этом $\rho = \lambda / \mu = 30 / 20 = 1,5$; $\rho_c = 0,75 < 1$.

Граф процесса гибели и размножения для данной модели представлен на рис. 9.8.

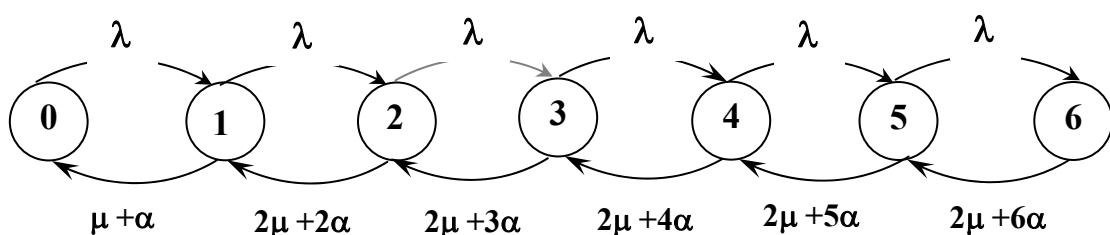


Рис. 9.8

Граф построен с учетом того, что переходы от $(j+1)$ -го состояния к j -му возможны как за счет завершения обслуживания, так и ухода “нетерпеливых” заявок из очереди и из обслуживающих приборов. Используя соотношения (9.32), можно найти вероятностные характеристики состояний:

$$P_0 = 0,3534; \quad P_1 = 0,3534; \quad P_2 = 0,1767; \quad P_3 = 0,0757; \\ P_4 = 0,0284; \quad P_5 = 0,0095; \quad P_6 = 0,029.$$

Среднее число занятых каналов:

$$\bar{K}_3 = P_1 + 2P_2 + 2(1 - P_0 - P_1 - P_2) = 0,9398.$$

Средняя длина очереди:

$$\bar{n} = 1P_3 + 2P_4 + 3P_5 + \dots + 4P_6 = 0,1726.$$

Коэффициент загрузки:

$$\eta_3 = \bar{K}_3 / K = 0,4699.$$

Вероятность отказов по причине ограниченности буфера:

$$P_{отк} = P_6 = 0,0029.$$

Вероятность ухода заявки во время обслуживания:

$$P_y' = \bar{K}_3 \cdot \alpha / \mu = 0,9398 \cdot 10 / 30 = 0,3133.$$

Вероятность ухода заявки из очереди:

$$P_y'' = \bar{n}_0 \cdot \alpha / \lambda = 1726 \cdot 10 / 30 = 0,0575.$$

Общая вероятность ухода:

$$P_y = P_y' + P_y'' = 0,3133 + 0,0575 = 0,3708.$$

Значение критерия эффективности E в соответствии с выбранными данными равно

$$E = 30 \cdot (3 \cdot 0,0029 + 2 \cdot 0,3708) + 10 \cdot (2 - 0,9398) = 33,111 \text{ усл. ед.}$$

9.5. МЕТОД ЧАСТИЧНОГО УКРУПНЕНИЯ МОДЕЛЕЙ ПРИ ИСПОЛЬЗОВАНИИ МАРКОВСКИХ ПРОЦЕССОВ

Использование описанной процедуры анализа с использованием системы уравнений (9.17) в многовариантных задачах может оказаться затруднительным при большой размерности пространства состояний, поэтому представляются необходимыми методы снижения размерности.

Одним из них является метод частичного укрупнения состояний [33]. Идея метода заключается в том, чтобы анализ сложной по структуре модели системы осуществлять по частям с помощью совокупности частично укрупненных моделей. В каждой из таких моделей подробно представлена некоторая часть системы, а остальная часть отображается обобщенным параметром (параметром связи). В итоге получается система уравнений, описывающих изменение состояния каждой из частично укрупненных моделей.

лей.

При реализации метода частичного укрупненного (агрегирования) можно использовать принцип эквивалентности потоков. В этом случае часть системы заменяется агрегированным узлом — обслуживающим аппаратом (ОА) с очередью. Интенсивность обслуживания в таком ОА (параметр связи) зависит от числа заявок в узле.

Принцип эквивалентности потоков состоит в том, что агрегированный узел при любом числе заявок в нем должен обеспечивать такой же поток во внешнюю сеть, как и заменяемая им система. Рассмотрим применение этой идеи для анализа модели, представленной в примере 9.4.

Пример 9.6

Вместо модели, представленной на рис. 9.5, рассмотрим агрегированную модель АМ 1, в которой подсистема, содержащая процессор и каналы обмена, заменена агрегированным узлом с интенсивностью обслуживания, зависящей от числа заявок. Состояние системы в некоторый момент времени есть число заявок в агрегированном узле. Структура модели АМ 1 и граф переходов представлены на рис. 9.9 а), б).

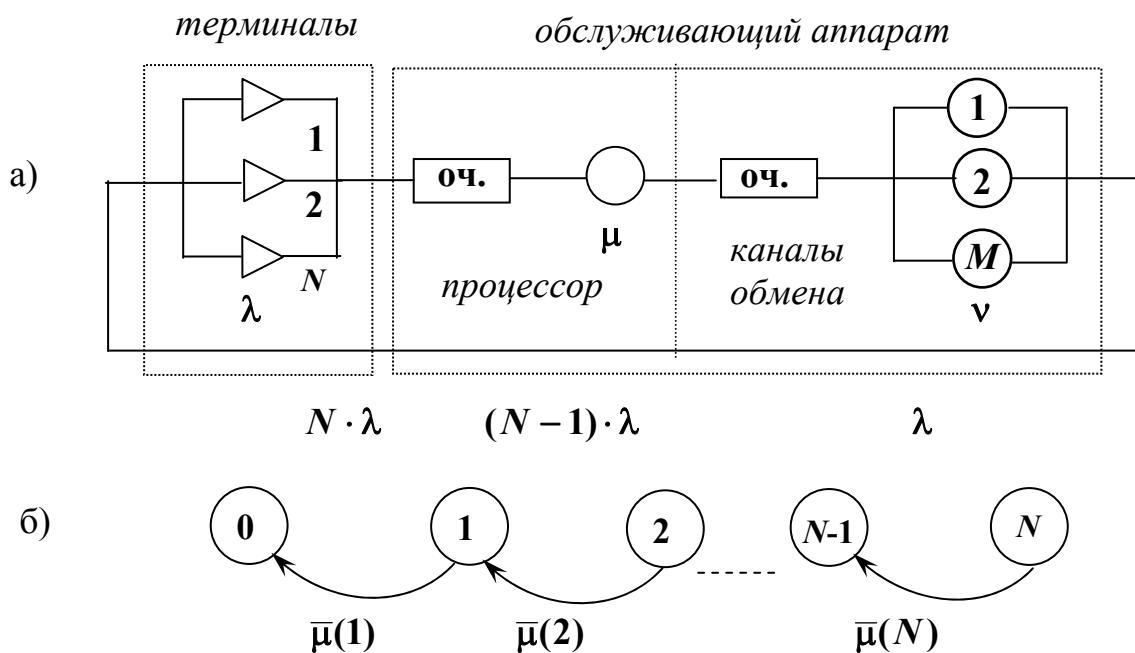


Рис. 9.9

Так как данный процесс является процессом гибели-размножения, то стационарное распределение вероятностей записывается в соответствии с (9.32) следующим образом:

$$\begin{cases} P_0 = \left(1 + \frac{N \cdot \lambda}{\bar{\mu}(1)} + \frac{N \cdot (N-1) \cdot \lambda^2}{\bar{\mu}(1) \cdot \bar{\mu}(2)} + \dots \right)^{-1}; \\ P_1 = P_0 \cdot \frac{N \cdot \lambda}{\bar{\mu}(1)}; \\ P_2 = P_0 \cdot \frac{N \cdot (N-1) \cdot \lambda^2}{\bar{\mu}(1) \cdot \bar{\mu}(2)}; \\ \dots \dots \dots \end{cases} \quad (9.33)$$

При этом среднее число заявок в агрегированном узле равно:

$$\bar{N} = \sum_{j=1}^N j \cdot P_j, \quad (9.34)$$

а среднее время реакции из закона Литтла определяется следующим образом:

$$\bar{t}_p = \frac{\bar{N}}{(N - \bar{N}) \cdot \lambda}. \quad (9.35)$$

Чтобы воспользоваться формулами (9.34) и (9.35) необходимо знать параметры связи $\bar{\mu}(j), j = \overline{1, N}$. Для этого используется модель АМ 2, структура и график переходов для которой представлены на рис. 9.10 а), б).

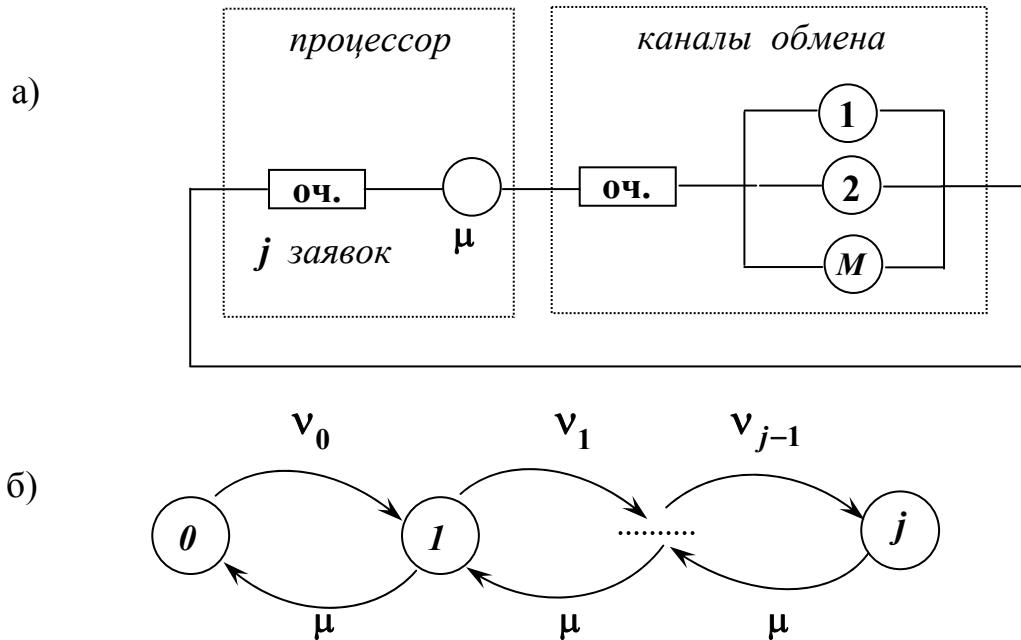


Рис. 9.10

За состояние в графе переходов принято число заявок i на процессорной фазе. Тогда интенсивность переходов ν_i в графе определяется как:

$$\nu_i = \nu \cdot \min [M, j-i], \quad i = \overline{0, j-1}. \quad (9.36)$$

Стационарное распределение представляется соотношениями:

$$\left\{ \begin{array}{l} \pi_{0j} = \left(1 + \frac{\nu_0}{\mu} + \frac{\nu_0 \cdot \nu_1}{\mu^2} + \dots \right)^{-1}; \\ \pi_{1j} = \pi_{0j} \cdot \frac{\nu_0}{\mu}; \\ \pi_{2j} = \pi_{0j} \cdot \frac{\nu_0 \cdot \nu_1}{\mu^2}; \\ \dots \end{array} \right. \quad (9.37)$$

Тогда параметры связи $\bar{\mu}(j)$ определяются как:

$$\bar{\mu}(j) = \mu \cdot (1 - \pi_{0j}), \quad j = \overline{1, N}. \quad (9.38)$$

Последовательность расчета следующая:

1. Для $j = \overline{1, N}$ по формулам (9.37), (9.38) определяются $\bar{\mu}(j)$.
2. По формулам (9.33), (9.34), (9.35) определяется среднее время реакции.
3. Процедуру частичного укрупнения моделей можно использовать многократно, при этом анализируемая модель последовательно упрощается.

9.6. ОСНОВНЫЕ СООТНОШЕНИЯ ДЛЯ ПРОСТЕЙШИХ СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ

В данном параграфе в табл. 9.3 – 9.9 приводятся без вывода базисные расчетные соотношения для основных показателей качества обслуживания в элементарных СМО, описываемых в классе процессов гибели и размножения. Во всех рассматриваемых системах предполагается:

- поток простейший с параметром λ ;
- обслуживание в канале по показательному закону с параметром μ ;
- дисциплина обслуживания — бесприоритетная, т. е. первым пришел — первым обслужен.

Таблица 9.3

Показатели	M/M/1 — одноканальная система с неограниченной очередью
P_θ	$1-\rho; \rho = \frac{\lambda}{\mu}$
P_j	$\rho^j (1-\rho); j \geq 1$
\bar{j}	$\frac{\rho}{(1-\rho)}$
\bar{n}_θ	$\frac{\rho^2}{(1-\rho)}$
$\bar{t_{ож}}$	$\frac{\rho}{(1-\rho)\mu}$
$\bar{t_C}$	$\frac{1}{\mu(1-\rho)}$
$P(j > k)$	ρ^{K+1}
$P(n_\theta > k)$	ρ^{K+2}
$P(t_{ож} < t)$	$1 - \rho e^{-\mu(1-\rho)t}; t \geq 0$
$P(t_C < t)$	$1 - e^{-\mu(1-\rho)t}; t \geq 0$
σ_j^2	$\frac{\rho}{(1-\rho)^2}$

Таблица 9.4

Показатели	M/M/1/m — одноканальная система с ограниченной очередью
P_0	$\frac{1-\rho}{1-\rho^{m+2}}$
P_j	$P_0 \rho^j$
\bar{n}_0	$\frac{\rho^2 P_0}{(1-\rho)^2} [1 - \rho^m (m + 1 - m\rho)]$
\bar{j}	$\bar{n}_0 + \frac{\rho - \rho^{m+2}}{1 - \rho^{m+2}}$
$\bar{t}_{ож}$	$\frac{\rho P_0}{\mu (1-\rho)^2} [1 - \rho^m (m + 1 - m\rho)]$
P_{OTK}	$\frac{\rho^{m+1} (1-\rho)}{1 - \rho^{m+2}}$
\bar{t}_c	$\bar{t}_{ож} + \frac{1}{\mu} (1 - P_{OTK})$
Относительная пропускная способность	
$q = 1 - P_{отк}$	
Абсолютная пропускная способность	
$A = \lambda \cdot q$	

Таблица 9.5

Показатели	M/M/K — многоканальная система с неограниченной очередью
P_0	$\left[1 + \sum_{j=1}^K \rho^j \frac{1}{j!} + \rho^{K+1} \frac{1}{K!(K-\rho)} \right]^I.$
P_j	$\begin{cases} \rho^j P_0 / j! & ; \quad j \leq K; \\ \rho^j P_0 / K! K^{j-K} & ; \quad j > K. \end{cases}$
\bar{n}_0	$\frac{\rho^{K+1} \cdot P_0}{K \cdot K! (1 - \rho_c)^2}; \quad \rho_c = \frac{\lambda}{K\mu}.$
\bar{K}_3	ρ
\bar{j}	$\bar{n}_0 + \bar{K}_3$
$\bar{t}_{o\&c}$	$\frac{\rho^K \cdot P_0}{K \cdot \mu \cdot K! (1 - \rho_c)^2}$
\bar{t}_c	$\bar{t}_{o\&c} + \frac{1}{\mu}$
$P(t_{o\&c} < t)$	$1 - \pi e^{-\mu(K-\rho)t}; \quad \pi = \sum_{j=K}^{\infty} P_j.$

Таблица 9.6

Показатели	M/M/∞ — многоканальная система с неограниченным числом каналов K .
$P_0(t)$	$e^{-\frac{\lambda}{\mu}(1-e^{-\mu t})}$
$P_0 = \lim_{t \rightarrow \infty} P_0(t)$	$e^{-\frac{\lambda}{\mu}}$
$P_j(t)$	$\frac{1}{j!} \cdot \left(\frac{\lambda}{\mu}\right)^j \cdot (1-e^{-\mu t})^j \cdot P_0(t)$
$P_j = \lim_{t \rightarrow \infty} P_j(t)$	$\frac{1}{j!} \cdot \left(\frac{\lambda}{\mu}\right)^j e^{-\frac{\lambda}{\mu}}$
$K_3(t)$	$\frac{\lambda}{\mu}(1-e^{-\mu t})$
$\overline{K_3} = \lim_{t \rightarrow \infty} K_3(t)$	$\frac{\lambda}{\mu}$

Таблица 9.7

Показатели	M/M/K/m — многоканальная система с числом каналов K и ограниченной очередью m
P_0	$\left[1 + \sum_{j=1}^K \rho^j \frac{1}{j!} + \frac{\rho^{K+1} (1 - \rho_c^m)}{K \cdot K! (1 - \rho_c)} \right]^{-1}; \quad \rho_c = \frac{\lambda}{K\mu}.$
P_j	$\begin{cases} \rho^j \frac{1}{j!} P_0, & 1 < j \leq K; \\ \rho^j \frac{1}{K!} \frac{1}{K^{j-K}} P_0; & K < j \leq K+m. \end{cases}$
P_{omk}	$\frac{\rho^{K+m}}{K^m \cdot K!} \cdot P_0$
\bar{K}_3	$\rho (1 - P_{omk})$
\bar{n}_0	$\frac{\rho^{K+1} \cdot P_0}{K \cdot K!} \left\{ \frac{1 - \rho_c^m / (m + 1 - m\rho_c)}{(1 - \rho_c)^2} \right\}$
$\bar{t}_{o\text{жс}}$	\bar{n}_0 / λ
\bar{j}	$\bar{n}_0 + \bar{K}_3$
\bar{t}_c	$\bar{t}_{o\text{жс}} + \frac{1}{\mu} (1 - P_{omk})$

Таблица 9.8

Показатели	M/M/1/n — одноканальная система с конечным числом требований
P_0	$\left[\sum_{j=0}^n \frac{n!}{(n-j)!} \rho^j \right]^{-1}$
P_j	$\frac{n!}{(n-j)!} \rho^j P_0 ; \quad j = \overline{1, n}$
\bar{j}	$n - \frac{1 - P_0}{\rho}$
\bar{n}_0	$n - \frac{1 + \rho}{\rho} (1 - P_0)$
$\bar{t}_{o\kappa c}$	$\frac{1}{\mu} \left[\frac{n}{1 - P_0} - \frac{1 + \rho}{\rho} \right]$
\bar{t}_c	$\frac{1}{\mu} \left[\frac{n}{1 - P_0} - \frac{1}{\rho} \right]$

Таблица 9.9

Показатели	M/M/K/n — многоканальная система с числом каналов K и с конечным числом требований n , $n > K$
P_0	$\left[1 + \sum_{j=1}^K \rho^j C_n^j + \sum_{j=K+1}^n \rho^j C_n^j \frac{j!}{K! K^{j-K}} \right]^{-1}$
P_j	$\begin{cases} \rho^j \frac{n!}{(n-j)! j!} P_0; & 1 \leq j \leq K, \\ \rho^j \frac{n!}{(n-j)! K! K^{j-K}} P_0; & K < j \leq n. \end{cases}$
\bar{n}_θ	$\sum_{j=K+1}^n (j - K) P_j$
\bar{K}_3	$\sum_{j=1}^{K-1} j P_j + K \sum_{j=K}^n P_j$
\bar{j}	$\bar{n}_\theta + \bar{K}_3$
\bar{t}_c	$\frac{\bar{j}}{\lambda(n - \bar{j})}$
$\bar{t}_{ож}$	$\bar{t}_c - \frac{1}{\mu}$

Пример 9.7

Провести сравнительный анализ двух структур организации системы обработки информации (рис. 9.11 а, б).

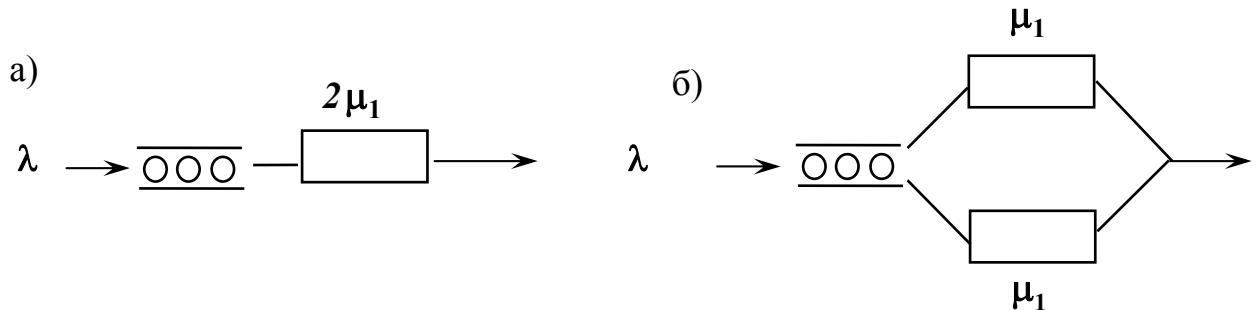


Рис. 9.11

В качестве показателя для проведения сравнительного анализа выберем среднюю длину очереди в системе \bar{n}_0 . Для структуры на рис 9.11 (а) в соответствии с моделью типа **M/M/1** имеем:

$$\bar{n}_0^{(a)} = \frac{\rho^2}{1-\rho} = \frac{(\lambda/2\mu_1)^2}{1-\lambda/2\mu_1} = \frac{1}{2} \frac{\rho_*^2}{(2-\rho_*)}, \quad \rho_* = \frac{\lambda}{\mu_1}.$$

Для структуры на рис. 9.11 (б) в соответствии с моделью типа **M/M/K**:

$$\bar{n}_0^{(b)} = \frac{\rho^{K+1} P_0}{K \cdot K! (1-\rho_c)^2},$$

где $P_0 = \left[1 + \sum_{j=1}^K \rho^j \frac{1}{j!} + \rho^{K+1} \frac{1}{K! (K-\rho)} \right]^{-1}$.

При $K = 2$ получим:

$$\bar{n}_0^{(b)} = \rho_*^3 P_0 \cdot \frac{1}{(2-\rho_*)^2};$$

$$P_0 = \left[1 + \rho_* + \rho_*^2 \frac{1}{2!} + \rho_*^3 \frac{1}{2} \frac{1}{(2-\rho_*)} \right]^{-1} = \frac{2-\rho_*}{2+\rho_*};$$

$$\bar{n}_0^{(b)} = \rho_*^3 \frac{1}{(2-\rho_*)(2+\rho_*)} = \frac{\rho_*^3}{4-\rho_*^2}.$$

Выражение для коэффициента предпочтительности одной структуры

по сравнению с другой по данному показателю имеет следующий вид:

$$F_1(\rho_*) = \frac{\bar{n}_0^{(a)}}{\bar{n}_0^{(b)}} = \frac{1}{2} \frac{\rho_*^2}{(2-\rho_*)} \cdot \frac{4-\rho_*^2}{\rho_*^3} = \frac{1+\rho_*/2}{\rho_*} > 1.$$

Отсюда видно, что во всей возможной области изменения $0 < \rho_* < 2$ средняя длина очереди в первой структуре на рис. 9.11 (а) больше, чем во второй (рис. 9.11 (б)), при одинаковой средней производительности обслуживающей системы.

Рассмотрим другой показатель качества обслуживания, среднее время \bar{t}_c пребывания требования в системе. Величина \bar{t}_c в соответствии с моделью **M/M/1** равна:

$$\bar{t}_c^{(a)} = \frac{1}{\mu} \cdot \frac{1}{1-\rho} = \frac{1}{2\mu_1} \cdot \frac{1}{1-\lambda/2\mu_1} = \frac{1}{\mu_1} \cdot \frac{1}{2-\rho_*}.$$

Аналогично для \bar{t}_c в соответствии с моделью **M/M/K** имеем:

$$\bar{t}_c^{(b)} = \frac{1}{\mu} + \frac{\rho_*^K \cdot P_0}{K \mu K! (1-\rho_c)^2}.$$

При $K=2$ получим:

$$\bar{t}_c^{(b)} = \frac{1}{\mu_1} + \frac{\rho_*^2 \cdot \frac{2-\rho_*}{2+\rho_*} \cdot 2}{2\mu_1 \cdot 2 \left(1 - \frac{\lambda}{2\mu_1}\right)^2} = \frac{1}{\mu_1} \cdot \frac{4}{4-\rho_*^2}.$$

Коэффициент предпочтительности:

$$F_2(\rho_*) = \frac{\bar{t}_c^{(a)}}{\bar{t}_c^{(b)}} = \frac{2+\rho_*}{4} < 1.$$

Отсюда видно, что в области $0 < \rho_* < 2$ среднее время пребывания в системе согласно рис. 9.11 (а) меньше, чем в случае рис. 9.11 (б).

Из данного примера видно, что от выбора показателя качества обслуживания зависит и выбор структуры СМО.

10. НЕМАРКОВСКИЕ СИСТЕМЫ МАССОВОГО ОБСЛУЖИВАНИЯ

10.1. КЛАССИФИКАЦИЯ МЕТОДОВ ИССЛЕДОВАНИЯ НЕМАРКОВСКИХ СМО

Как было показано выше, допущение о более общих законах распределения длительности интервалов между требованиями в потоке и длительности обслуживания нарушает марковское свойство и требует более сложных методов исследования.

В [29] проведена классификация основных, перечисленных ниже подходов к исследованию немарковских СМО:

- метод фаз (этапов) Эрланга;
- введение избыточной переменной;
- метод вложенных цепей Маркова;
- использование полумарковских процессов;
- интегральный подход.

10.1.1. Метод фаз (этапов)

Метод фаз разработан в самом начале прошлого столетия Эрлангом. А.К. Эрланг заметил, что ряд распределений могут быть разложены в набор показательных распределений. Это в первую очередь касается распределения:

$$f_k(x) = \frac{k\mu(k\mu x)^{k-1} e^{-k\mu x}}{(k-1)!},$$

которое при замене $k\mu=\mu^*$ переходит в гамма-распределение с параметром μ^* :

$$f_k(x) = \frac{\mu^*(\mu^* x)^{k-1} e^{-\mu^* x}}{(k-1)!}. \quad (10.1)$$

В дальнейшем будем считать распределение (10.1) базовым в моделях СМО типа **M/E_k/1**, **E_k/M/1**.

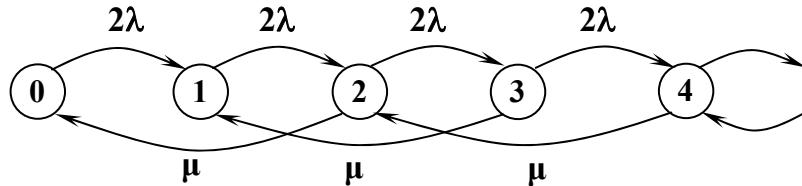
В соответствии с этим длительность обслуживания в системе типа **M/E_k/1** раскладывается в набор k фаз, длительность каждой из которых имеет плотность распределения:

$$f(x) = k\mu e^{-k\mu x}, \quad x \geq 0.$$

Аналогичную ситуацию имеем для системы типа $E_k/M/1$.

На рис. 10.1 представлены графы переходов для случая $k=2$ систем $E_2/M/1$ и $M/E_2/1$. Номер состояния $N(t)$ марковского процесса есть число этапов обслуживания находящихся в системе заявок.

а) Система типа $E_2/M/1$:



б) Система типа $M/E_2/1$:

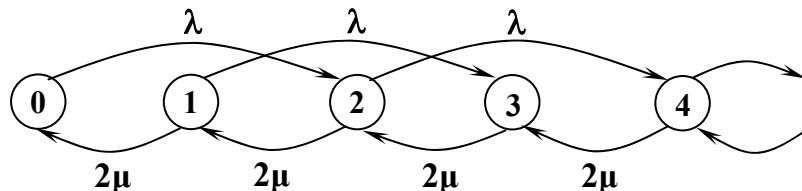


Рис. 10.1. Графы переходов для случая $k = 2$

Так как распределение длительности пребывания в каждом из состояний графов рис.10.1 является показательным, обладающим отсутствием последействия, то соответствующие процессы являются марковскими и поэтому к ним может быть применен рассмотренный выше в разделе 9.3 метод исследования с использованием общих уравнений непрерывного марковского процесса.

Этот метод связан с увеличением размерности графа состояний и невыгоден тем, что он дает только процедуру решения, но не позволяет получить решение в явном виде.

10.1.2. Метод введения избыточной переменной

В этом случае одним из возможных способов приведения немарковского процесса к марковскому является восполнение недостающей информации с помощью дополнительной переменной.

Метод был предложен Кендаллом и дальнейшее развитие получил в работах Кокса и Смита [29]. Для иллюстрации основного принципа рассмотрим систему типа $M/G/1$. Как ранее было показано для системы типа $M/M/1$, описываемой в классе процессов гибели и размножения, в качестве состояния процесса достаточно было выбрать число требований j , находящихся в системе. В случае системы $M/G/1$ с произвольным законом рас-

пределения длительности обслуживания этого недостаточно, и приходится расширить размерность состояния процесса за счет введения в него дополнительной переменной x , времени, прошедшего в системе от начала обслуживания текущей заявки до рассматриваемого момента t . Таким образом, состоянием процесса становится вектор (j, x) и описание дисциплины такого векторного процесса может быть получено на основе уравнений в частных производных.

Естественным является стремление к упрощению такого описания за счет рассмотрения не любых моментов t наблюдения за процессом, а только определенных в соответствии со специально подобранный последовательностью точек на оси времени. Если для этих моментов сохраняются марковские свойства для последовательности переходов в графе состояний, то такая последовательность называется *вложенной марковской цепью*.

10.1.3. Метод вложенных марковских цепей

Для системы **M/G/1** такой последовательностью являются последовательность моментов ухода требований из обслуживающего прибора, при этом состояние процесса определяется числом требований, остающихся в эти моменты в системе. Так как состояние процесса в эти моменты может изменяться только за счет поступающих требований из пуассоновского потока, не обладающего последействием, то соответствующая цепочка переходов будет обладать марковскими свойствами. Аналогичное рассмотрение можно провести и для системы типа **G/M/1**.

В соответствии с данным подходом в [29] выводится известная формула Поллячека-Хинчина для среднего числа требований в системе **M/G/1**:

$$\bar{j} = \rho + \rho^2 \frac{1 + \sigma_x^2 / (\bar{x})^2}{2(1-\rho)}, \quad (10.2)$$

где σ_x^2 и \bar{x} определяются для конкретного закона обслуживания.

Частным случаем использования вложенных цепей Маркова является полумарковский процесс (ПМП).

10.1.4. Использование полумарковских процессов (ПМП)

Формально полумарковский процесс полностью определяется заданием:

множества состояний и начального состояния;

вектора закона распределения длительностей пребывания ПМП в каждом состоянии — $F_i(t), (i = \overline{1, n})$;

матрицей вероятностей непосредственных переходов в моменты,

соответствующие окончанию пребывания ПМП в своих состояниях, $\mathbf{P} = \{P_{ij}\}$.

Определим основные соотношения, аналогичные (9.17) для определения предельных вероятностей ПМП. Пусть $\pi_j (j = \overline{1, n})$ есть вероятность пребывания ПМП в состоянии j в стационарном режиме, только с учетом переходов. Вероятности π_j связаны следующими соотношениями:

$$\pi_j = \sum_i \pi_i P_{ij}, \quad j = \overline{1, n}. \quad (10.3)$$

Определим формулы для расчета предельных вероятностей ПМП P_j как с учетом переходов, так и длительностей пребывания процесса в соответствующем состоянии. Рассмотрим достаточно длинную реализацию ПМП, содержащую большое число L различных переходов.

Пусть случайная величина T_c — длина этой реализации, а T_{cj} — длительность пребывания ПМП в состоянии j . Очевидно, что

$$\begin{aligned} M[T_c] &= L \sum_i \pi_i M[T_i], \\ M[T_{cj}] &= L \pi_i M[T_i], \end{aligned}$$

где $M[T_j]$ — среднее значение длительности пребывания процесса в j -м состоянии. Тогда:

$$\begin{aligned} P_j &= \lim_{L \rightarrow \infty} \frac{M[T_{cj}]}{M[T_c]} = \frac{\pi_j M[T_j]}{\sum_i \pi_i M[T_i]}, \text{ откуда} \\ \pi_j &= \frac{P_j}{M[T_j]} \sum_i \pi_i M[T_i]. \end{aligned} \quad (10.4)$$

Подставляя выражение (10.4) в систему уравнений (10.3), получим:

$$\frac{P_j}{M[T_j]} = \sum_i \frac{P_i}{M[T_i]} P_{ij}, \quad (j = \overline{1, n}). \quad (10.5)$$

Так как уравнения системы (10.5) являются линейно-зависимыми, то при их решении одно из уравнений следует заменить соответствующим

условием нормировки: $\sum_{j=1}^n P_j = 1$.

Иногда удобно рассматривать систему (10.5) в обозначениях $\frac{P_j}{M[T_j]} = U_j$,

т. е.

$$\begin{cases} U_j = \sum_i U_i P_{ij}, \\ \sum_j U_j M[T_j] = 1, \quad (j = \overline{1, n}). \end{cases} \quad (10.6)$$

Рассматривая полумарковский процесс «гибели и размножения» на конечном множестве состояний и вводя для каждого из состояний подобно λ_i и μ_i вероятности переходов q_i и S_i , ($q_i + S_i \leq 1$), получим из (10.5) соотношения для предельных вероятностей:

$$P_j = M[T_j] \prod_{k=1}^j \left(\frac{q_{k-1}}{S_k} \right) \left[\sum_{i=0}^n M[T_i] \cdot \prod_{k=1}^i \left(\frac{q_{k-1}}{S_k} \right) \right]^{-1}, \quad (j = \overline{i, n}).$$

Пример 10.1

Рассмотрим обобщенный цикл решения задачи пользователем на вычислительном центре:

подготовка программы → ввод → счет → анализ результатов.

На каждом из первых трех этапов возможны ошибки; выявление ошибок может производиться как на этапе ввода и счета, так и на этапе анализа результатов. Общий граф прохождения задачи представлен на рис. 10.2.

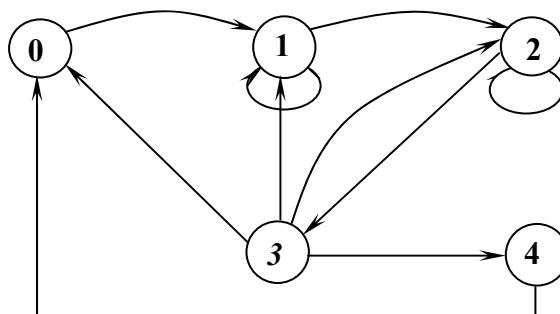


Рис. 10.2

Предполагается, что заданы значения:

$$M[T_i], \quad (i = \overline{0, 4}),$$

где **0** — фаза подготовки программы, **1** — фаза ввода, **2** — фаза счета, **3** —

фаза анализа результатов, 4 — фаза генерации новой задачи.

Матрица вероятностей переходов P соответствует рис. 10.2:

$$P = \begin{pmatrix} & 0 & 1 & 2 & 3 & 4 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & P_{11} & 1 - P_{11} & 0 & 0 \\ 2 & 0 & 0 & P_{22} & 1 - P_{22} & 0 \\ 3 & P_{30} & P_{31} & P_{32} & 0 & P_{34} \\ 4 & 1 & 0 & 0 & 0 & 0 \end{pmatrix},$$

где P_{11} , P_{22} — вероятности обнаружения ошибки; P_{30} , P_{31} , P_{32} — вероятности обнаружения ошибок по результатам анализа, дифференцированных по фазам прохождения задачи; P_{33} — вероятность необнаруженных ошибок, равная 0; P_{34} — вероятность правильного решения задачи ($\sum_{j=0}^4 P_{3j} = 1$).

Система уравнений (10.5) для данного конкретного случая выглядит следующим образом:

$$\begin{cases} u_0 = P_{30} u_3 + P_{40} u_4; \\ u_1 = P_{11} u_1 + P_{31} u_3 + P_{01} u_0; \\ u_2 = P_{12} u_1 + P_{22} u_2 + P_{32} u_3; \\ u_3 = P_{23} u_2; \\ u_4 = P_{34} u_3; \\ \sum_{i=0}^4 u_i M[T_i] = 1. \end{cases}$$

Разрешение полученной системы уравнений позволяет найти все интересующие нас вероятности P_j .

10.1.5. Интегральный метод

Все рассмотренные выше методы были основаны на использовании марковских свойств за счет применения тех или иных преобразований пространства состояний или выбора необходимой последовательности наблюдений. При изучении более сложных систем типа G/G/1 и G/G/k приходится отказаться от многих упрощений, вытекающих из марковских

свойств и искать новые методы анализа. Рассмотрим основную идею подхода на примере системы типа **G/G/1**.

Все заявки имеют естественный порядок поступления их в систему. Обозначим через C_n заявку с порядковым номером n ($n = 1, 2, \dots$), а через W_n — время ожидания ее в очереди. Тогда, если на обслуживание этой заявки требуется ровно x_n времени, то длительность ее пребывания в системе от момента поступления до завершения обслуживания на приборе определяется:

$$S_n = x_n + W_n. \quad (10.7)$$

Случайное время между приходом заявок C_{n-1} и C_n обозначим t_n .

Установим рекуррентную связь между этими величинами. Каждая вновь появляющаяся заявка может застать обслуживающий прибор свободным или занятым. Если прибор свободен, то заявка сразу начнет обслуживаться и $W_n = 0$; если же прибор оказывается занятым, то заявка C_n прежде чем начать обслуживаться, должна ждать завершения обслуживания заявки C_{n-1} . В результате время ожидания заявки C_n оказывается меньше времени пребывания заявки C_{n-1} на интервал t_n между поступлением в систему, т. е. $W_n = S_{n-1} - t_n$.

Объединяя эти условия, находим, что

$$W_n = \max[0, S_{n-1} - t_n] \quad (10.8)$$

и время S_n пребывания в системе заявки C_n равно

$$S_n = \max[0, S_{n-1} - t_n] + x_n. \quad (10.9)$$

Введем функции распределения случайной величины времени ожидания в очереди и пребывания в системе:

$$W_n(x) = P[W_n < x]; \quad S_n(x) = P[S_n \leq x]. \quad (10.3)$$

Тогда из (10.9) и формулы полной вероятности на основании независимости величин t_n и S_n получим:

$$\begin{aligned} W_n(x) &= P\{W_n = \max[0, S_{n-1} - t_n] \leq x\} = \\ &= \sum_{\Delta\xi} P\{t_n \in (\xi + \Delta\xi)\} P\{S_{n-1} < x + \xi\} \Big|_{\xi=\overline{0,\infty}} = \int_0^\infty S_n(x + \xi) dA(\xi) \end{aligned} \quad (10.11)$$

где $A(x)$ — функция распределения длительности интервала в потоке.

$$S_n(x) = P\{S_n(x) = W_n(x) + x_n < x\} = \int_0^x W_n(x - \xi) dB(\xi), \quad (10.12)$$

где $B(x)$ — функция распределения длительности обслуживания.

Переходя к пределу при $n \rightarrow \infty$ в выражениях (10.11) и (10.12), получим, что в установившемся режиме функции $W(x)$ и $S(x)$ связаны системой двух интегральных уравнений Винера-Хопфа:

$$W(x) = \int_0^\infty S(x + \xi) dA(\xi), \quad S(x) = \int_0^x W(x - \xi) dB(\xi), \quad (10.13)$$

разрешение которых позволяет получить $W(x)$ и $S(x)$ для заданных законов $A(x)$ и $B(x)$. Так, например, для системы типа **M/M/1** и законов

$$A(t) = 1 - e^{-\lambda t} \quad \text{и} \quad B(t) = 1 - e^{-\mu t}$$

$$\text{получим:} \quad W(t) = 1 - \rho e^{-\mu(1-\rho)t}, \quad t \geq 0;$$

$$S(t) = 1 - e^{-\mu(1-\rho)t}, \quad t \geq 0.$$

10.2. Характеристики простейших систем массового обслуживания типа **M/D/1**, **M/E_K/1**, **M/G/1**, **G/M/1**

Важнейшим результатом использования вложенных цепей Маркова для анализа характеристик СМО является формула Поллячека-Хинчина для определения среднего числа требований в одноканальной системе в момент ухода обслуженного требования при произвольном законе распределения длительности обслуживания, т. е. для систем **M/G/1**:

$$\bar{j} = \rho + \rho^2 \frac{\left(1 + \frac{\sigma_x^2}{(\bar{x})^2}\right)}{2(1-\rho)}, \quad (10.14)$$

где σ_x^2 и \bar{x} определяются для конкретного закона обслуживания.

В соответствии с (10.14) имеем:

— для системы **M/M/1**

$$\sigma_x^2 = \frac{1}{\mu^2}, \quad \bar{x} = \frac{1}{\mu}, \quad \bar{j} = \frac{\rho}{1-\rho}; \quad (10.15)$$

— для системы **M/D/1**

$$\sigma_x^2 = 0, \quad \bar{x} = \frac{1}{\mu}, \quad \bar{j} = \rho + \frac{\rho^2}{2(1-\rho)} = \frac{\rho}{1-\rho} - \frac{\rho^2}{2(1-\rho)}; \quad (10.16)$$

что свидетельствует о том, что система **M/D/1** в среднем на $\rho^2/2(1-\rho)$ содержит меньше требований, чем система **M/M/1**;

– для системы **M/E_k/1**

$$\bar{j} = \rho + \frac{\rho^2 \left(1 + \frac{1}{k} \right)}{2(1-\rho)}, \quad (10.17)$$

что свидетельствует о том, что система **M/E_k/1** при $k \rightarrow \infty$ приближается к системе **M/D/1**.

Из формулы (10.14) и закона Литтла (7.18) можно получить соотношения, определяющие время ожидания требования в очереди:

$$\bar{t}_{ож} = \frac{\rho \bar{x} \left(1 + \frac{\sigma_x^2}{\bar{x}^2} \right)}{2(1-\rho)}; \quad (10.18)$$

для системы **M/D/1**

$$\bar{t}_{ож} = \frac{\rho \frac{1}{\mu}}{2(1-\rho)} = \frac{\rho}{2\mu(1-\rho)}; \quad (10.19)$$

для системы **M/E_k/1**

$$\bar{t}_{ож} = \frac{\rho \left(1 + \frac{1}{k} \right)}{2\mu(1-\rho)}. \quad (10.20)$$

Использование принципа вложенных марковских цепей позволяет получить не только характеристики моментов, но и законы распределения времени пребывания и времени ожидания в системе.

Если обозначить через $W^*(s)$ и $S^*(s)$ преобразование Лапласа для плотностей распределения $t_{ож}$ и t_c , а через $B^*(s)$ — преобразование Лапласа для закона распределения времени обслуживания, то из [29] известно:

$$S^*(x) = B^*(s) \frac{s(1-\rho)}{s - \lambda + \lambda B^*(s)}, \quad (10.21)$$

$$W^*(s) = \frac{s(1-\rho)}{s - \lambda + \lambda B^*(s)}. \quad (10.22)$$

Реализуя обратное преобразование Лапласа величин $W^*(s)$ и $S^*(s)$

для системы **M/M/1**, получим:

$$P(t_c < t) = 1 - e^{-\mu(1-\rho)t}; \quad (10.23)$$

$$P(t_{ож} < t) = 1 - \rho e^{-\mu(1-\rho)t}. \quad (10.24)$$

Заметим, что для экспоненциального закона распределения времени обслуживания в системе **M/M/1**:

$$B^*(s) = \frac{\mu}{s + \mu}.$$

Одной из важных характеристик систем **M/G/1** является среднее остаточное время до завершения обслуживания требования, находящегося в системе, \bar{T}_o определяемое как:

$$\bar{T}_o = \frac{1}{2} \lambda v^{(2)}, \quad (10.25)$$

где $v^{(2)}$ — второй начальный момент распределения времени обслуживания требования в системе [29]. Данная характеристика является весьма важной при анализе приоритетных систем массового обслуживания и будет использована в разделе 10.

Применение метода вложенных цепей согласно [29] позволяет получить для систем типа **G/M/1** вероятности состояний P_j в следующем виде:

$$P_j = (1 - \sigma) \sigma^k, \quad k = 0, 1, 2, \dots \quad (10.26)$$

где σ — единственное решение уравнения

$$\sigma = A^*(\mu - \mu\sigma) \quad (10.27)$$

при условии, что $0 < \sigma < 1$ (гарантия установившегося режима), где $A^*(s)$ — преобразование Лапласа для плотности распределения $b(y)$ длительности интервалов в потоке

Среднее время ожидания $\bar{t}_{ож}$ равно:

$$\bar{t}_{ож} = \frac{\sigma}{\mu(1 - \sigma)}, \quad (10.28)$$

а вероятность $P(t_{ож} < t)$ определяется как:

$$P(t_{ож} < t) = 1 - \sigma e^{-\mu(1-\sigma)t}. \quad (10.29)$$

Для иллюстрации данного подхода рассмотрим примеры.

Пример 10.2

Рассматривается система **M/M/1**. Тогда:

$$A^*(s) = \int_0^\infty e^{-st} \lambda e^{-\lambda t} dt = \frac{\lambda}{\lambda + s}, \quad (10.30)$$

а уравнение (10.27) выглядит следующим образом:

$$\sigma = \frac{\lambda}{\mu - \mu\sigma + \lambda}. \quad (10.31)$$

Единственное решение уравнения (10.31), удовлетворяющее условию $0 < \sigma < 1$, является $\sigma = \rho$. В результате получаем известные характеристики для системы **M/M/1**, представленные в табл. 9.3.

Пример 10.3

Определить реальную пропускную способность системы передачи данных (СПД) с учетом допустимого времени хранения сообщений перед выдачей. Входной поток — простейший, время обслуживания — постоянное, равное x_3 . При неограниченной очереди лимитирующим фактором является только допустимое время ожидания t_o . При этом возможны две дисциплины обслуживания требований, для которых время ожидания превысит допустимое.

1. Требование обслуживается, но считается, что система загружена им непроизводительно, и при определении реальной пропускной способности передача таких сообщений не учитывается.

2. Требование выбывает из очереди, если его время ожидания до начала обслуживания превышает допустимое время ожидания.

Рассмотрим дисциплину №1. Используя подход для определения плотности распределения времени ожидания при постоянном времени обслуживания, основанный на соотношениях (10.21), (10.22), получим, что вероятность превышения времени ожидания t_o равна:

$$P(t_{ож} > t_o) = (1 - \rho) \sum_{m=r+1}^{\infty} e^{-\rho} \left(\frac{t_o}{x_3} - m \right) \frac{\left[-\rho \left(\frac{t_o}{x_3} - m \right) \right]^m}{m!};$$

где r — наибольшее целое число, удовлетворяющее условию $r \leq t_o / x_3$, ρ — коэффициент загрузки системы.

Данное выражение громоздко и весьма неудобно для расчета из-за необходимости суммировать знакопеременный ряд. Однако имеется при-

ближенное выражение, приведенное в [35], которое позволяет получить численное значение вероятности превышения заданного времени задержки:

$$P(t_{ож} > t_o) = \frac{(1-\rho)e^{-\frac{t_o}{x_3}(u-\rho)}}{u-1},$$

где u определяется в соответствии с табл. 10.1.

Таблица 10.1

ρ	0,7	0,8	0,9	0,95	0,98
u	1,375	1,231	1,107	1,052	1,020

Следовательно, полезная нагрузка системы для дисциплины №1 определяется как:

$$\rho_{СПД} = \rho [1 - P(t_{ож} > t_o)].$$

Значение $\rho_{СПД}$ для некоторых значений t_o/x_3 и ρ представлены в табл. 10.2.

Таблица 10.2

t_o/x_3	ρ		
	0,7	0,8	0,9
5	0,67	0,72	0,61
10	0,70	0,80	0,78
40	0,70	0,80	0,90

Для дисциплины №2 определение вероятности потерь полностью эквивалентно дисциплине с ограниченной очередью величины $r = [t_o/x_3]$, где квадратные скобки означают целую часть дроби.

Определение вероятности потерь при постоянном времени обслуживания может быть проведено следующим образом. Из (10.14) следует, что среднее число требований в очереди при экспоненциальном законе обслуживания вдвое больше, чем при постоянном времени обслуживания и при той же загрузке системы.

Тогда вероятность потерь заявок при пуассоновском потоке и постоянном времени обслуживания можно приблизенно рассчитать, так же, как и при экспоненциальном законе обслуживания, увеличив объем r накопителя в два раза:

$$P_{omk} = \frac{(1-\rho)\rho^{2r+1}}{1-\rho^{2r+2}},$$

где $r = [t_o/x_3]$.

Тогда величина $\rho'_{СПД}$, т. е. реальная загрузка СПД равна:

$$\rho'_{СПД} = \rho(1 - P_{omk}).$$

Значения $\rho'_{СПД}$ для тех же значений t_o/x_3 и ρ представлены в табл. 10.3.

Таблица 10.3

t_o/x_3	ρ		
	0,7	0,8	0,9
5	0,70	0,79	0,86
10	0,70	0,80	0,89
40	0,70	0,80	0,9

Из сравнения данных табл. 10.1 и 10.2 видно, что $\rho_{СПД} \leq \rho'_{СПД}$ для одних и тех же значений загрузки ρ , что указывает на целесообразность использования дисциплины № 2, и следовательно на эффективность применения ограниченной буферной памяти, соответствующей максимальному времени ожидания.

10.3. ПРИБЛИЖЕННЫЕ ОЦЕНКИ ХАРАКТЕРИСТИК СМО

Теория массового обслуживания — довольно сложная дисциплина и получение точных результатов в ней во многих случаях оказывается затруднительным. Более того, точные результаты, которые можно получить даже для относительно простых систем, нередко настолько сложны, что становятся малопригодными для практических приложений.

Многие практические задачи массового обслуживания часто не удовлетворяют тем допущениям, которые делаются в большинстве работ по теории массового обслуживания. Поэтому большой интерес вызывает приближенное построение решений и оценок качества СМО [31].

Для систем **G/G/1** в [31] приведена верхняя граничная оценка для среднего времени ожидания:

$$\bar{t}_{ож}^{(b)} = \frac{\sigma_a^2 + \sigma_b^2}{2\bar{t}(1-\rho)}, \quad (10.32)$$

где σ_a^2 и σ_b^2 — соответственно, дисперсии длительности интервала в потоке и интервала обслуживания, \bar{t} — средняя длительность интервала между требованиями.

При этом данная верхняя граница является приближением среднего времени ожидания при большой нагрузке ($\rho \rightarrow 1$). Полученная верхняя граница, по существу, не зависит от закона распределения. Она определяется только первыми двумя моментами распределения промежутков между требованиями и времени обслуживания.

Аналогичная граничная оценка получена Кингманом и Брумелем для систем типа **G/G/K** и приведена в [29]:

$$\bar{t}_{ож} \leq \frac{\sigma_a^2 \frac{1}{K} \sigma_b^2 + \frac{K-1}{K^2} \bar{x}^2}{2\bar{t}(1-\rho)}, \quad (10.33)$$

где K — число каналов; \bar{x} — первый начальный момент длительности распределения времени обслуживания.

Рассмотрим, как соотносится граничная оценка (10.32) с истинным значением $\bar{t}_{ож}$ для систем типа **M/M/1**:

$$\begin{aligned} \bar{t}_{ож} &= \frac{\rho}{(1-\rho)\mu} \\ \bar{t}_{ож}^{(b)} &= \frac{\sigma_a^2 + \sigma_b^2}{2\bar{t}(1-\rho)} = \frac{\rho^2 + 1}{\lambda 2(1-\rho)} = \bar{t}_{ож} \frac{\rho^2 + 1}{2\rho^2}. \end{aligned}$$

Оценка коэффициента $A = \frac{\rho^2 + 1}{2\rho^2}$:

$$\rho = 0,9; \quad A = \frac{1,81}{1,62} \approx 1;$$

$$\rho = 0,5; \quad A = \frac{1,25}{0,5} \approx 2,5;$$

$$\rho = 0,1; \quad A = \frac{1,01}{2 \cdot 0,01} \approx 50.$$

Отсюда видно, что чем выше коэффициент загрузки, тем ближе граничная оценка к истинному значению среднего времени ожидания.

В [29, 31] приведен обзор различных направлений в построении приближенных оценок:

- построение границ для точных решений,
- дискретная аппроксимация работы СМО,
- непрерывное приближение работы СМО с использованием усредненных характеристик потоков и законов обслуживания,
- использование диффузионного приближения, при котором процесс поступления требований и процесс ухода их из системы, аппроксимируется гауссовыми непрерывными процессами с независимыми приращениями.

Хочется отметить, что разработка хороших приближений для сложных вероятностных процессов, описывающих СМО, дает мощный толчок развитию теории массового обслуживания и ее применений.

11. СИСТЕМЫ МАССОВОГО ОБСЛУЖИВАНИЯ С НЕОДНОРОДНЫМ ПОТОКОМ

11.1. ОДНОКАНАЛЬНЫЕ СМО С БЕСПРИОРИТЕТНЫМ ОБСЛУЖИВАНИЕМ

Рассматривается одноканальная СМО, в которую поступает многомерный поток с интенсивностями $\lambda_1, \lambda_2, \dots, \lambda, \dots, \lambda_n$, при этом каждый из входящих потоков — простейший. Длительность обслуживания требований каждого из потоков задается функцией распределения $B_i(t)$ с математическим ожиданием $\bar{x}_i = M\{x_i\}$ и вторым начальным моментом $M(x_i^2) = v_i^{(2)}$. Обслуживание — бесприоритетное, интенсивность обслуживания требований i -го потока — μ_i .

Использование обычного марковского подхода для описания поведения систем подобного типа приводит в случае неограниченного источника к бесконечным системам линейных уравнений. Поэтому в данном случае анализ качества обслуживания более эффективно проводить специальным методом — методом средних значений.

Базовой характеристикой качества обслуживания является среднее время ожидания $t_{ож}$. Поясним кратко основную идею метода [35, 36].

Пусть t_o — момент поступления требования из i -го потока, а t_i — момент, когда данное требование будет принято на обслуживание. Тогда

$$t_{ожi} = t_i - t_o.$$

При бесприоритетном обслуживании

$$\bar{t}_{ожi} = \bar{T}_o + \sum_{i=1}^n \bar{T}_i, \quad i = \overline{1, n}, \quad (11.1)$$

где T_o — время, необходимое для завершения обслуживания ранее выбранного требования; T_i — время обслуживания требований из i -го потока, поступивших в систему ранее рассматриваемого.

Проводя усреднение в (11.1), получим:

$$\bar{t}_{ожi} = \bar{T}_o + \sum_{i=1}^n \bar{T}_i, \quad (11.2)$$

где

$$\bar{T}_o = \frac{1}{2} \sum_{i=1}^n \lambda_i v_i^{(2)} \quad (11.3)$$

в соответствии с формулой (3.25), а

$$\bar{T}_i = \bar{x}_i \bar{n}_{oi} = \bar{x}_i \bar{\lambda}_i \bar{t}_{o\text{ж}i},$$

при этом \bar{n}_{oi} — среднее число требований типа i в очереди.

С учетом указанных выше формул, получим:

$$\bar{t}_{o\text{ж}i} = \bar{T}_o + \sum_{i=1}^n \bar{x}_i \bar{\lambda}_i \bar{t}_{o\text{ж}i} = \bar{T}_o + \sum_{i=1}^n \rho_i \bar{t}_{o\text{ж}i}. \quad (11.4)$$

Отсюда следует, что для различных типов требований среднее время ожидания при бесприоритетной дисциплине обслуживания одинаково, т. е.

$$\bar{t}_{o\text{ж}} = \bar{T}_o / (1 - R), \quad (11.5)$$

где

$$R = \sum_{i=1}^n \rho_i, \quad \rho_i = \bar{x}_i \bar{\lambda}_i,$$

причем в установившемся режиме коэффициент загрузки R меньше единицы.

Для различных законов распределения времени обслуживания значение \bar{T}_o зависит в соответствии с (11.3) от величины $v_i^{(2)}$:

$$\begin{aligned} M &\rightarrow v_i^{(2)} = 2(1/\mu_i)^2; \\ D &\rightarrow v_i^{(2)} = (1/\mu_i)^2; \\ E_{k_i} &\rightarrow v_i^{(2)} = (1/\mu_i)^2(1/k_i + 1). \end{aligned} \quad (11.6)$$

11.2. ОДНОКАНАЛЬНЫЕ СМО С ПРИОРИТЕТАМИ

11.2.1. Относительный приоритет

Рассматривается дисциплина обслуживания с относительным приоритетом, причем, чем меньше номер потока, тем выше относительный приоритет.

Подобно (11.1) время ожидания требования из i -го потока

$$t_{o\text{ж}i} = T_o + \sum_{k=1}^i T'_k + \sum_{k=1}^{i-1} T''_k, \quad i = \overline{1, n}, \quad (11.7)$$

где T_o определяется в соответствии с (11.3), (11.6);

$\sum_{k=1}^i T'_k$ — время обслуживания всех требований с более высоким приоритетом или таким же приоритетом, поступивших в систему ранее рассматриваемого;

$\sum_{k=1}^{i-1} T''_k$ — время обслуживания всех требований с более высоким приоритетом, чем i , поступивших за время ожидания $t_{ожi}$ рассматриваемого требования.

Рассуждая так же, как и при выводе уравнения (11.4), получим систему уравнений:

$$\bar{t}_{ожi} = \bar{T}_o + \sum_{k=1}^i \rho_k \bar{t}_{ожk} + \sum_{k=1}^{i-1} \rho_k \bar{t}_{ожi}, \quad i = \overline{1, n}. \quad (11.8)$$

$$\text{Отсюда: } \bar{t}_{ожi} = \frac{\bar{T}_o + \sum_{k=1}^i \rho_k \bar{t}_{ожk}}{1 - R_{i-1}}, \quad i = \overline{1, n}. \quad (11.9)$$

Используя метод математической индукции для решения системы уравнений (11.9), получим:

$$\bar{t}_{ожi} = \frac{\bar{T}_o}{(1 - R_{i-1})(1 - R_i)}, \quad i = \overline{1, n}, \quad (11.10)$$

где $R_0 = 0$, $R_{i-1} = \sum_{k=1}^{i-1} \rho_k$.

11.2.2. Абсолютный приоритет

Расчет СМО с абсолютным приоритетом подобен приведенному выше, но включает дополнительные трудности, связанные с описанием того, как прерванное требование возвращается к обслуживанию.

Возможны три варианта дисциплины.

1. Требование возвращается к обслуживанию с того места, где оно было прервано. Такая дисциплина называется *дисциплиной с абсолютным приоритетом и дообслуживанием*.

2. Требование при возвращении обслуживается с начала с тем же самым временем. Такая дисциплина называется *дисциплиной с абсолютным приоритетом и повторным обслуживанием без повторного выбора времени обслуживания*.

3. При повторном обслуживании время обслуживания определяется заново. Такой случай называется обслуживанием *с абсолютным приоритетом и с повторными обслуживанием и выбором времени обслуживания*.

В дальнейшем рассматривается первый случай. Соотношение (11.7) в данном случае выглядит следующим образом:

$$t_{ож\ i} = T_i^H + T_i^n ,$$

где T_i^H — время ожидания начала обслуживания;

T_i^n — время ожидания в прерванном состоянии.

Переходя к математическим ожиданиям, получим

$$\bar{t}_{ож\ i} = \bar{T}_i^H + \bar{T}_i^n , \quad i = \overline{1, n}. \quad (11.11)$$

Для определения \bar{T}_i^n будем рассуждать следующим образом. За среднее время обслуживания \bar{x}_i заявок типа i в систему поступит в среднем $\lambda_k \bar{x}_i$ заявок типа k , имеющих более высокий приоритет, чем i , которые прервут обслуживание рассматриваемой заявки и будут обслуживаться в течение времени, в среднем равному $\tau_k^{(1)} = \bar{x}_k \lambda_k \bar{x}_i$.

Среднее время обслуживания заявок всех типов с более высоким абсолютным приоритетом, чем i , равно:

$$T_i^{(1)} = \sum_{k=1}^{i-1} \tau_k^{(1)} = \sum_{k=1}^{i-1} \bar{x}_k \lambda_k \bar{x}_i = \bar{x}_i R_{i-1},$$

где $R_{i-1} = \sum_{k=1}^{i-1} \bar{x}_k \lambda_k = \sum_{k=1}^{i-1} \rho_k$ — загрузка системы со стороны заявок 1, 2,

..., $i-1$, имеющих приоритет более высокий, чем i .

За это время в систему поступят еще заявки с более высоким приоритетом, чем i , в количестве:

$$\lambda_k T_i^{(1)} = \lambda_k \bar{x}_i R_{i-1},$$

которые будут обслужены за время:

$$\tau_k^{(2)} = \bar{x}_k \lambda_k \bar{x}_i R_{i-1} = \rho_k \bar{x}_i R_{i-1}.$$

Общее среднее время обслуживания заявок всех типов с более высоким приоритетом, чем i , будет равно:

$$T_i^{(2)} = \sum_{k=1}^{i-1} \tau_k^{(2)} = \sum_{k=1}^{i-1} \rho_k \bar{x}_i R_{i-1} = \bar{x}_i R_{i-1}^2.$$

Рассматривая данный процесс в динамике на l -м шаге, имеем:

$$T_i^{(l)} = \bar{x}_i R_{i-1}^l, \quad (l=1,2,\dots).$$

Общее среднее время ожидания заявок типа i будет равно среднему времени обслуживания всех заявок с более высоким приоритетом, которые поступают в систему за время обслуживания рассматриваемой заявки:

$$\bar{T}_i^n = \sum_{l=1}^{\infty} T_i^{(l)} = \sum_{l=1}^{\infty} \bar{x}_i R_{i-1}^l = \frac{\bar{x}_i R_{i-1}}{1 - R_{i-1}}, \quad (i=1,\dots,n), \quad (11.12)$$

где $R_{i-1} = \sum_{k=1}^{i-1} \rho_k < 1$.

Время ожидания начала обслуживания i -го требования:

$$T_i^H = \sum_{k=1}^i T_{ok} + \sum_{k=1}^i T'_k + \sum_{k=1}^{i-1} T''_k + \sum_{k=1}^i T'''_k, \quad (11.13)$$

где $\sum_{k=1}^i T_{ok}$ — время, необходимое для завершения обслуживания ранее выбранного требования с более высоким или таким же приоритетом i ;

$\sum_{k=1}^i T'_k$ — время обслуживания требований, которые поступили в систему ранее рассматриваемой заявки и имеют более высокий или такой же приоритет;

$\sum_{k=1}^{i-1} T''_k$ — время обслуживания требований, поступивших в систему за время ожидания начала обслуживания рассматриваемого требования типа i , имеющих более высокий приоритет и, следовательно, принятых на обслуживание ранее рассматриваемого требования;

$\sum_{k=1}^i T'''_k$ — время обслуживания всех требований с более высоким или таким же приоритетом, которые на момент поступления рассматриваемого находились в прерванном состоянии.

Переходя к математическим ожиданиям из (11.13) получим:

$$\bar{T}_i^H = \sum_{k=1}^i \bar{T}_{ok} + \sum_{k=1}^i \bar{T}'_k + \sum_{k=1}^{i-1} \bar{T}''_k + \sum_{k=1}^i \bar{T}'''_k. \quad (11.14)$$

Как следует из (11.3),

$$\bar{T}_{ok} = \bar{t}_k = \frac{1}{2} \lambda_k V_k^{(2)} = \rho_k V_k^*, \quad (11.15)$$

где $V_k^* = \frac{1}{2} \bar{x}_k (1+u_k^2)$ — среднее время дообслуживания требований k -го типа, учитываящее, что на дообслуживании находится требование типа k ; $u_k = \sigma_k / \bar{x}_k$;

$\rho_k = \lambda_k \bar{x}_k$ — загрузка со стороны требований k -го типа, которая в данном случае имеет смысл вероятности того, что в произвольный момент времени в системе находится заявка типа k .

Среднее время обслуживания требований типа k , поступивших в систему ранее рассматриваемого, равно

$$\bar{T}'_k = \bar{x}_k \lambda_k \bar{T}_k^H = \rho_k \bar{T}_k^H, \quad (11.16)$$

где $\lambda_k \bar{T}_k^H$ — среднее число заявок типа k , ожидающих начала обслуживания.

$$\text{Аналогично: } \bar{T}''_k = \bar{x}_k \lambda_k \bar{T}_i^H, \quad (11.17)$$

где $\lambda_k \bar{T}_i^H$ — среднее число требований типа k , которые поступают за время ожидания начала обслуживания рассматриваемого требования типа i .

Среднее время дообслуживания требований типа k , находящихся в прерванном состоянии, определяется как:

$$\bar{T}'''_k = V_k^* \lambda_k \bar{T}_k^n, \quad (11.18)$$

где $\lambda_k \bar{T}_k^n$ — число заявок k -го типа, находящихся в прерванном состоянии.

С учетом формулы (11.13):

$$\bar{T}'''_k = V_k^* \lambda_k \bar{T}_k^n = \frac{V_k^* \lambda_k \bar{x}_k R_{k-1}}{1 - R_{k-1}} = \frac{R_{k-1} \rho_k V_k^*}{1 - R_{k-1}}. \quad (11.19)$$

Подставляя (11.16), (11.17), (11.19) в соотношение (11.14), получим:

$$\bar{T}_i^H = \sum_{k=1}^i \rho_k V_k^* + \sum_{k=1}^i \rho_k \bar{T}_k^H + \bar{T}_i^H R_{i-1} + \sum_{k=1}^i \frac{R_{k-1} \rho_k V_k^*}{1 - R_{k-1}}.$$

Введем следующую подстановку из (11.15) $\bar{T}_{ok} = \rho_k V_k^*$ и после некоторых преобразований получим:

$$\bar{T}_i^H = \frac{\sum_{k=1}^i \frac{\bar{T}_{ok}}{1-R_i} + \sum_{k=1}^{i-1} \rho_k \bar{T}_k^H}{1-R_i}, \quad (i = \overline{1, n}). \quad (11.20)$$

Применив метод математической индукции к последнему выражению, найдем, что среднее время ожидания начала обслуживания требований с приоритетом i :

$$\bar{T}_i^H = \frac{\sum_{k=1}^i \bar{T}_{ok}}{(1-R_i)(1-R_{i-1})}, \quad (i = \overline{1, n}).$$

Подставляя (11.20) и (11.12) в (11.11), где $\bar{T}_{ok} = \frac{1}{2} \lambda_k v_k^{(2)}$, получим среднее время ожидания требований при использовании дисциплины с абсолютными приоритетами:

$$\bar{t}_{ож,i} = \frac{R_{i-1} \bar{x}_i}{1-R_{i-1}} + \frac{\sum_{k=1}^i \lambda_k v_k^{(2)}}{2(1-R_i)(1-R_{i-1})}, \quad (i = \overline{1, n}). \quad (11.21)$$

Зная среднее время ожидания требования, легко найти по известным зависимостям остальные характеристики обслуживания требований.

11.3. ОДНОКАНАЛЬНЫЕ СИСТЕМЫ МАССОВОГО ОБСЛУЖИВАНИЯ СО СМЕШАННЫМИ ПРИОРИТЕТАМИ

В некоторых системах массового обслуживания необходимо выполнить жесткие ограничения на время ожидания отдельных режимов, что требует присваивания им абсолютных приоритетов. В результате этого время ожидания отдельных низкоприоритетных требований может оказаться недопустимо большим, хотя отдельные требования и имеют запас по времени ожидания. Чтобы выполнить ограничения по всем видам требований, можно наряду с абсолютными приоритетами некоторым требованиям присвоить относительные приоритеты, а остальные требования обслуживать без приоритетов. Такая дисциплина обслуживания называется смешанной.

Рассмотрим далее смешанную дисциплину обслуживания с тремя классами требований, когда требованиям l_1, \dots, l_1 присвоены абсолютные приоритеты, требованиям $l_1 + 1, \dots, l_1 + l_2$ относительные приоритеты, а требования $l_1 + l_2 + 1, \dots, l$ обслуживаются на основе бесприоритетной дисциплины, т. е. в порядке поступления. Требования первого класса облада-

ют абсолютным приоритетом по отношению к требованиям второго и третьего класса, а требования второго класса — относительным приоритетом по отношению к третьему классу. Определим средние времена ожидания требований различных типов.

Для требований первого класса, обладающих абсолютным приоритетом по отношению ко всем остальным, характерна независимость среднего времени ожидания от характеристик обслуживания с более низкими приоритетами. В этом случае среднее время ожидания требований первого класса будет таким же, как при использовании чистой дисциплины с абсолютными приоритетами, т. е. будет определяться выражением (11.21).

Требования второго и третьего классов могут рассматриваться как требования с относительными приоритетами, время ожидания которых увеличено за счет прерывания требованиями первого класса, причем требования третьего класса, обладают самым низким относительным приоритетом.

С учетом этого среднее время ожидания требований этих классов можно представить в виде:

$$\bar{t}_{ож,i} = \bar{t}_{ож,i}^{(0)} + \bar{t}_{ож,i}^{(n)},$$

где $\bar{t}_{ож,i}^{(0)}$ — среднее время ожидания требований с относительными приоритетами без учета прерываний со стороны требований первого класса;

$\bar{t}_{ож,i}^{(n)}$ — среднее время ожидания, вызванное прерыванием рассматриваемого требования требованиями $1, 2, \dots, l_1$, относящимися к первому классу.

По аналогии с выводом выражения (11.21), легко показать, что:

$$\bar{t}_{ож,i}^{(n)} = \frac{\bar{x}_i R_{l_1}}{1 - R_{l_1}}, \quad (11.22)$$

где \bar{x}_i — среднее время обслуживания требований типа $i = \overline{l_1 + 1, l};$

$R_{l_1} = \sum_{i=1}^{l_1} \rho_i$ — загрузка со стороны требований первого класса.

Для требований второго класса величина $\bar{t}_{ож,i}^{(0)}$ будет определяться, как и в случае дисциплины с относительными приоритетами, выражением (11.10):

$$\bar{t}_{ож,i}^{(0)} = \frac{\sum_{k=1}^l \lambda_k v_k^{(2)}}{2(1 - R_{i-1})(1 - R_i)}. \quad (11.23)$$

После несложных преобразований это выражение может быть использовано и для требований третьего класса.

Величина R_{i-1} означает загрузку со стороны всех требований более высокого относительного приоритета, чем рассматриваемое требование, R_i — загрузка со стороны требований более высокого приоритета, включая рассматриваемую заявку типа i . Так как требования третьего класса рассматриваются как требования с самым низким относительным приоритетом, т. е. требования типов $1, 2, \dots, l_1 + l_2$ обладают более высоким приоритетом, то загрузка со стороны этих требований равна:

$$R_{l_1+l_2} = \sum_{i=1}^{l_1+l_2} \rho_i.$$

Загрузка же с учетом требований третьего класса представляет собой суммарную загрузку системы:

$$R = \sum_{i=1}^l \rho_i.$$

Таким образом, для требований третьего класса:

$$\bar{t}_{oжc,i}^{(0)} = \frac{\sum_{i=1}^l \lambda_i v_i^{(2)}}{2(1-R_{l_1+l_2})(1-R)}. \quad (11.24)$$

Окончательно для системы со смешанными приоритетами и тремя классами требований выражения для средних значений времени ожидания различных типов требований выглядят следующим образом:

$$\bar{t}_{oжc,i} = \begin{cases} \frac{R_{l_1} \bar{x}_i}{1-R_{i-1}} + \frac{\sum_{k=1}^i \lambda_k v_k^{(2)}}{2(1-R_{i-1})(1-R_i)}; & i = 1, 2, \dots, l_1; \\ \frac{R_{l_1} \bar{x}_i}{1-R_{i-1}} + \frac{\sum_{k=1}^i \lambda_k v_k^{(2)}}{2(1-R_{i-1})(1-R_i)}; & i = l_1 + 1, \dots, l_1 + l_2; \\ \frac{R_{l_1} \bar{x}_i}{1-R_{i-1}} + \frac{\sum_{k=1}^i \lambda_k v_k^{(2)}}{2(1-R_{l_1+l_2})(1-R_i)}; & i = l_1 + l_2 + 1, \dots, l. \end{cases} \quad (11.25)$$

Из рассмотренной дисциплины обслуживания со смешанным приоритетом и тремя классами требований легко могут быть получены как частные случаи следующих дисциплин:

- 1) с абсолютными приоритетами, ($l_2=l_3=0$);
- 2) с относительными приоритетами, ($l_1=l_3=0$);
- 3) с абсолютными и с относительными приоритетами, ($l_3=0$);
- 4) с абсолютными и без приоритетов, ($l_2=0$);
- 5) с относительными и без приоритетов, ($l_1=0$).

Пример 11.1

Провести сравнительный анализ следующих дисциплин обслуживания для двумерного потока $\lambda_1=0,5c^{-1}$; $\lambda_2=0,2c^{-1}$; $\mu_1=1,0c^{-1}$; $\mu_2=2,0c^{-1}$:

- бесприоритетного обслуживания;
- относительного приоритета за первым потоком;
- относительного приоритета за вторым потоком;
- абсолютного приоритета за первым потоком;
- абсолютного приоритета за вторым потоком;
- циклического обслуживания.

Бесприоритетное обслуживание

В соответствии с формулами (11.3), (11.5) и (11.6) имеем:

$$\bar{t}_{ож} = \bar{T}_o / (1 - R);$$

$$\bar{T}_o = \frac{1}{2} \sum_{i=1}^n \lambda_i v_i^{(2)};$$

$$R = \sum_{i=1}^2 \rho_i = \frac{0,5}{1} + \frac{0,2}{2} = 0,5 + 0,1 = 0,6;$$

$$\bar{T}_o = \frac{1}{2} \sum_{i=1}^2 \lambda_i v_i^{(2)} = \frac{1}{2} (0,5 \cdot 2 + 0,2 \cdot 2 \cdot \frac{1}{4}) = 0,5 \text{ с};$$

$$\bar{t}_{ож} = \frac{\bar{T}_o}{(1 - R)} = \frac{0,5}{1 - 0,6} = 1,26 \text{ с.}$$

*Обслуживание с относительным приоритетом
(приоритет за первым источником)*

В соответствие с формулой (11.10):

$$\bar{t}_{o\mathcal{K}1} = \frac{\bar{T}_o}{(1-R_o)(1-R_1)} = \frac{0,5}{0,5} = 1,0 \text{ c};$$

$$\bar{t}_{o\mathcal{K}2} = \frac{\bar{T}_o}{(1-R_1)(1-R_2)} = \frac{0,5}{0,5 \cdot 0,4} \approx 2,5 \text{ c}.$$

*Обслуживание с относительным приоритетом
(приоритет за вторым источником)*

$$\bar{t}_{o\mathcal{K}1} = \frac{0,5}{0,9} = 0,55 \text{ c};$$

$$\bar{t}_{o\mathcal{K}2} = \frac{0,5}{(1-0,1)(1-0,6)} = 1,4 \text{ c}.$$

*Обслуживание с абсолютным приоритетом
(приоритет за первым источником)*

В соответствии с (11.21) имеем:

$$\bar{t}_{o\mathcal{K},1} = \frac{R_o \bar{x}_1}{1-R_o} + \frac{\lambda_1 v_1^{(2)}}{2(1-R_o)(1-R_1)} = \frac{0,5 \cdot 2}{2(1-0,5)} = 1 \text{ c};$$

$$\bar{t}_{o\mathcal{K},2} = \frac{R_1 \bar{x}_2}{1-R_1} + \frac{\sum_{k=1}^2 \lambda_k v_k^{(2)}}{2(1-R_1)(1-R_2)} = \frac{0,5 \cdot 0,5}{0,5} + \frac{1}{2 \cdot (1-0,5)(1-0,6)} = 3 \text{ c}.$$

Обслуживание с абсолютным приоритетом за вторым источником

$$\bar{t}_{o\mathcal{K},1} = \frac{0,2 \cdot 2 \cdot 0,25}{2(1-0,1)} = 0,055 \text{ c};$$

$$\bar{t}_{o\mathcal{K},2} = \frac{0,1}{0,9 \cdot 2} + \frac{1}{2 \cdot (1-0,1)(1-0,6)} = 1,46 \text{ c}.$$

Циклическое обслуживание

Аналитическое исследование циклических дисциплин обслуживания требований при произвольном количестве входных потоков является весьма сложной задачей.

В данном случае речь идет о следующей интерпретации циклической дисциплины. Начавшееся обслуживания требований i -го потока продолжается до тех пор, пока в очереди не останется ни одной заявки этого потока. При этом на обслуживание принимаются как уже имеющиеся в очереди требования, так и требования, поступающие в период обслуживания i -й очереди. Переключение на обслуживание заявок следующего ($i+1$)-го потока осуществляется лишь после того, как в очереди не останется ни одного требования i -го типа. Обычно этот режим называют режимом пакетной обработки информации. Для случая $n=2$ в [34] получены следующие выражения для среднего времени ожидания заявок в очереди при условии, что время обслуживания требований имеет произвольное распределения:

$$\bar{t}_{ож,1} = \frac{\lambda_1 v_1^{(2)}}{2(1-\rho_1)} + \frac{\lambda_2 v_2^{(2)}(1-\rho_1)^2 + \lambda_1 v_1^{(2)}\rho_2^2}{2(1-\rho_1)(1-\rho)(1-\rho+2\rho_1\rho_2)};$$

$$\bar{t}_{ож,2} = \frac{\lambda_2 v_2^{(2)}}{2(1-\rho_2)} + \frac{\lambda_1 v_1^{(2)}(1-\rho_2)^2 + \lambda_2 v_2^{(2)}\rho_1^2}{2(1-\rho_2)(1-\rho)(1-\rho+2\rho_1\rho_2)}.$$

В соответствии с этим, если цикл начинается с первого потока, то

$$\bar{t}_{ож,1} = \frac{0,5 \cdot 2 \cdot 1}{2(1-0,5)} + \frac{0,2 \cdot 2 \cdot \frac{1}{4}(1-0,5)^2 + 0,5 \cdot 1 \cdot 0,01}{2(1-0,5)(1-0,6)(1-0,6+2 \cdot 0,1 \cdot 0,5)} \approx 1,2 \text{ с};$$

$$\bar{t}_{ож,2} = \frac{0,2 \cdot 2 \cdot \frac{1}{4}}{2(1-0,1)} + \frac{0,5 \cdot 2 \cdot 1 \cdot (1-0,1)^2 + 0,2 \cdot 2 \cdot \frac{1}{4} \cdot (0,5)^2}{2(1-0,1)(1-0,6)(1-0,6+2 \cdot 0,1 \cdot 0,5)} = 4 \text{ с.}$$

Используя соответствующие результаты по различным дисциплинам обслуживания и требованиям к допустимому времени ожидания заявок из различных потоков, можно выбрать соответствующий способ управления качеством обслуживания.

Пример 11.2

Для мультипроцессорной вычислительной системы с числом процессоров $n = 3$ и двумя независимыми модулями памяти, связанными между собой общей шиной, предложить эффективный алгоритм разрешения конфликтов на магистрали. Конфликты возникают в том случае, если несколько процессоров обратились к магистрали. Каждая программа представляет чередующуюся последовательность фаз счета и обмена данными с памятью. Средние значения длительностей фаз счета и обмена с памятью равны:

$$\bar{\tau}_i = \frac{1}{\lambda_i}, \quad \bar{t}_i = \frac{1}{\mu_i}, \quad i = \overline{1, 3}.$$

Определить коэффициент загрузки магистрали и временные задержки в выполнении программ при условии, что выбрана бесприоритетная дисциплина разрешения конфликтов, т. е. первый обратился – первым обслужился.

Из содержания задачи следует, что источник требований конечный, содержит три требования, обслуживающим прибором является магистраль обмена.

Предполагаем, что длительности фаз счета и обмена являются случайными величинами, подчиненными экспоненциальному закону распределения с параметрами λ_i и μ_i , $i = \overline{1, 3}$.

Для решения поставленной задачи используем общие уравнения марковского процесса. Состояние марковского процесса, описывающего поведение данной системы, представим вектором:

$$z^T = (z_1, z_2, z_3),$$

где каждая переменная z_i принимает нулевое значение или значение номера программы, обратившейся к магистрали (с учетом очередности поступления запросов на обмен).

Общий граф марковского процесса, описывающего поведение системы, представлен на рис. 11.1.

Разрешение системы уравнений типа (9.17) в установившемся режиме позволяет найти все вероятности состояний, в том числе и $P_{000} = 1 - \eta$, где η — коэффициент загрузки системы.

При определении времени задержек программ из-за конфликтов магистрали используем вероятностные характеристики состояний P_{ijk} .

Среднее время ожидания, например, первой программы определяем по правилу Литтла:

$$\bar{t}_{o_{\text{ок}1}} = \frac{\bar{n}_1}{\bar{\lambda}_1} = \frac{P_{210} + P_{310} + P_{231} + P_{213} + P_{312} + P_{321}}{\lambda_1(P_{000} + P_{200} + P_{300} + P_{230} + P_{320})}.$$

Аналогично определяются $\bar{t}_{o_{\text{ок}2}}$ и $\bar{t}_{o_{\text{ок}3}}$. Введение относительных приоритетов программ модифицирует систему возможных связей только между последним и предпоследним уровнем графа.

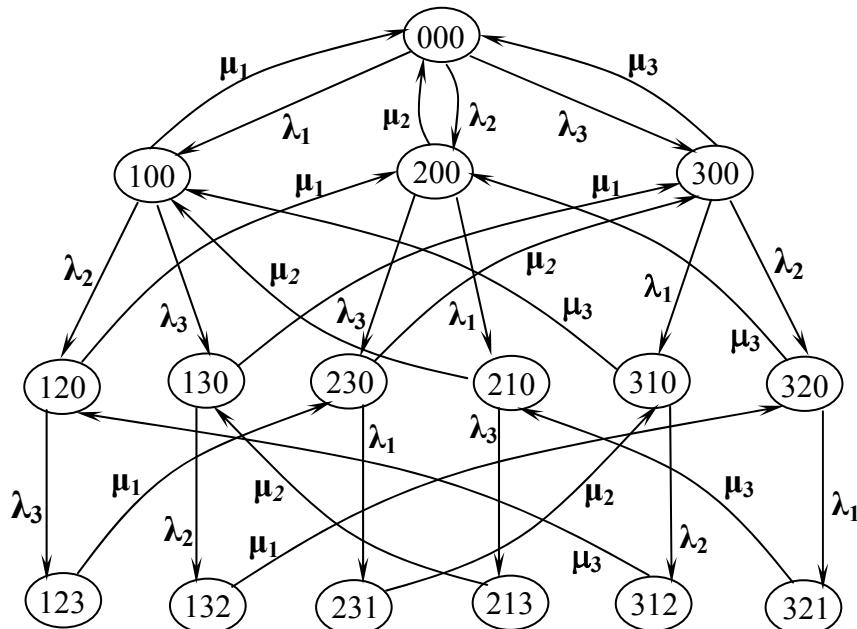


Рис. 11.1

Предположим, что введена система приоритетов: 1, 2, 3, меньший номер соответствует большему относительному приоритету, тогда график взаимодействий между указанными уровнями выглядит следующим образом (рис. 11.2):

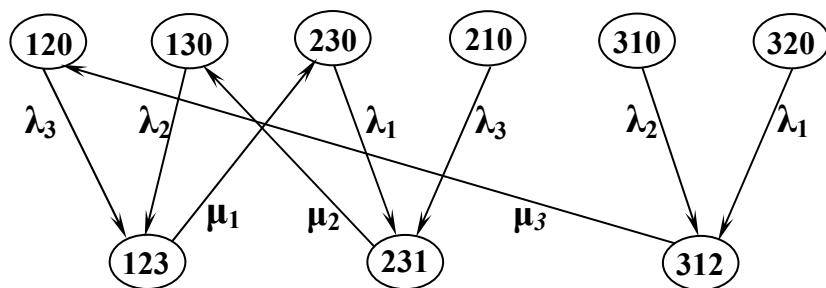


Рис. 11.2

Определив соответствующие вероятности состояний, можно оценить коэффициент загрузки и временные задержки программ по указанной выше методике, тем самым проведем анализ эффективности введения данной дисциплины относительно бесприоритетной [37].

ЗАДАЧИ

1. В процессе проектирования завода, необходимо определить, каким количеством испытательных стендов, предназначенных для контроля качества готовой продукции, он должен обладать. Пусть предъявляются следующие требования:

- вероятность того, что все стены заняты, равна 0,9;
- вероятность того, что готовое изделие ждет начала контроля больше смены равно 0,01.

Считать поток изделий, поступающих на контроль, простейшим, при этом среднее число изделий, изготавливаемых за смену, равно 5.

Время, затрачиваемое на контроль одного изделия, подчинено показательному закону распределения и в среднем равно двум сменам. Найти необходимое число стендов и построить закон распределения времени ожидания.

2. В информационную систему поступает непрерывный поток сообщений. При занятости системы очередное сообщение записывается в буферную память, рассчитанную на хранение m сообщений. При этом информация, которая содержится в каждом сообщении, теряет свою ценность через 2 минуты после его получения. Поток сообщений простейший с интенсивностью $\lambda = 10\text{мин}^{-1}$. В среднем за минуту система обрабатывает 20 сообщений, реальное время обработки подчинено показательному закону.

п.1. Оценить вероятность того, что поступившее сообщение не будет своевременно обработано и, следовательно, потеряно и необходимый объем буферной памяти.

п.2. Определить необходимые параметры (число устройств обработки и объем буферной памяти) при повышении интенсивности потока сообщений в 1.5 раза, обеспечивающие тот же уровень вероятности несвоевременной обработки сообщений, что и в п. 1.

Рассмотреть два варианта работы:

- просроченное сообщение удаляется,
- просроченное сообщение остается в системе и обрабатывается.

3. Рассматривается система типа **M/M/K/0**, предназначенная для обслуживания суммы двух пуассоновских потоков требований λ_i , а время обслуживания распределено по показательному закону с интенсивностью μ_i ($i = 1, 2$).

Первый поток является ординарным, поэтому каждое последующее требование занимает точно один из обслуживающих приборов; если все k приборов заняты, то вновь поступающее требование первого класса теряется. Для обслуживания каждого требования второго класса требуется одновременно k_0 приборов (и оно занимает все эти приборы одновременно на одно и то же показательно распределенное время со средним значением $1/\mu_2$). Если в момент поступления требования второго класса в системе имеется меньше, чем k_0 свободных приборов, это требование также теряется. Найти:

- долю потерянных требований первого и второго классов при $k_0 = k = 2$, и построить зависимость от λ_i , μ_i , ($i = 1, 2$),
- выяснить, насколько изменится процент потерянных требований по сравнению со случаем, когда потоки ординарны и $\mu_i = \mu$, $i = 1, 2$.

4. Рассматривается вычислительная система, представленная на рис. 11.3, включающая:

терминалы (среднее время обдумывания $\bar{T}_{об\delta} = 1/\lambda$);

процессор (среднее время решения задачи процессором $\bar{T}_{реш} = 1/\mu$);

каналы обмена с памятью, включающие селекторные каналы и накопители на магнитных дисках (среднее время обмена $\bar{T}_{обм} = 1/v$).

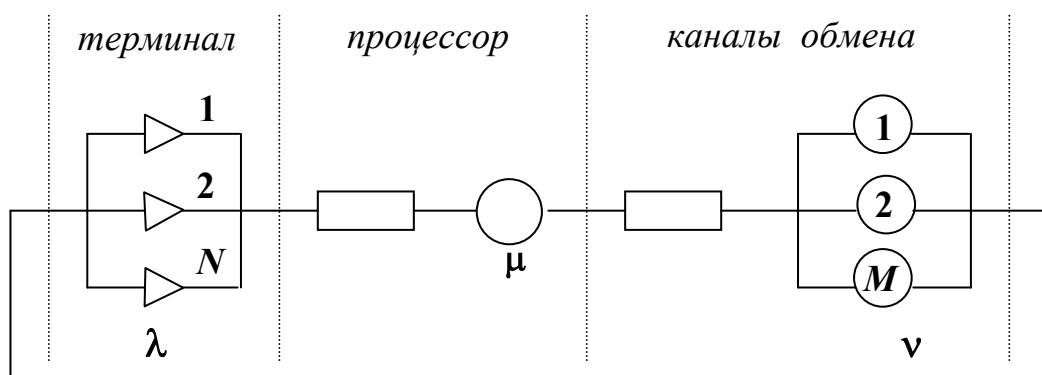


Рис. 11.3

Вычислить среднее время реакции (среднее время пребывания требования в системе) при следующих условиях:

– считать случайные величины $T_{об\delta}$, $T_{реш}$, $T_{обм}$ подчиняющимися показательному закону распределения с соответствующими параметрами;

– описать поведение системы марковским случайным процессом, при этом принять в качестве состояния вектор $\xi(t) = [\xi_1(t), \xi_2(t)]$, где $\xi_1(t)$ — число заявок на терминальной фазе (число пользователей, получивших ответ системы и непосредственно работающих за экраном), $\xi_2(t)$ — число заявок на процессорной фазе. Тогда число заявок на канальной фазе равно $N - \xi_1(t) - \xi_2(t)$. Построить график марковского процесса.

Использовать для оценки времени реакции закон Литтла, предварительно вычислив среднее число занятых терминалов, на основе построенного графа марковского процесса. Для оценки использовать следующие модельные данные

$$N = 3; \quad M = 2; \quad \lambda = 5 \text{ c}^{-1}; \quad \mu = 15 \text{ c}^{-1}; \quad v = 50 \text{ c}^{-1}.$$

Использовать для оценки времени реакции метод частичного укрупнения состояний, при этом считать, что подсистема, содержащая процессор и каналы заменена агрегированным узлом с интенсивностью обслуживания, зависящей от числа заявок.

Сравнить полученные результаты с результатами п. 3.

5. Рассматривается СМО, состоящая из двух неодинаковых каналов, с производительностью $\mu_1 > \mu_2$. Входящий пуассоновский поток имеет интенсивность λ . Требование, поступившее в простаивающую систему с вероятностью ϕ выбирает 1-й канал и с вероятностью $(1 - \phi)$ — 2-й канал.

Требуется:

– построить график марковского процесса, приняв все необходимые допущения;

– составить систему уравнений для установившегося режима и найти оценки вероятностей в общем виде, используя обозначения:

$$\rho = \frac{\lambda}{\mu_1 + \mu_2}; \quad \alpha = \frac{\mu_2}{\mu_1};$$

– проверить правильность полученных формул, используя для системы типа $M/M/2/\infty/\infty$ известные формулы при $\alpha = 1$,

– получить оценку для среднего числа требований в системе и сравнить со следующим выражением:

$$\bar{j} = \rho(1 + \alpha) \frac{1 + (1 + \alpha)\rho - (1 - \alpha)\phi}{\alpha(1 + 2\rho) + \rho[1 + (1 + \alpha)^2\rho - (1 - \alpha^2)\phi]},$$

– исследовать зависимость $j = f(\alpha, \varphi)$ при $\rho = 0,6$ в следующих диапазонах изменения параметров φ, α :

$$0 < \varphi < 1; \quad 0 < \alpha < 1;$$

– выяснить, при каких условиях можно использовать приближенную модель $M/M/2/\infty/\infty$, где каждый из каналов имеет интенсивность

$$\mu = \frac{\mu_1 + \mu_2}{2}.$$

6. Перед управляющим нотариальной конторой всталась проблема комплектования штата. Для простоты предположим, что выбор ограничен двумя вариантами:

– принять на работу двух опытных машинисток, каждая из которых способна печатать и оформлять 20 документов в день;

– принять на работу четырех неопытных машинисток, каждая из которых печатает и оформляет лишь 10 документов в день.

В упомянутой нотариальной конторе в среднем печатается 36 документов в день. Необходимо произвести следующие расчеты:

– оценить средние продолжительности периодов занятости машинисток в течение дня, а также средние продолжительности пребывания документа у машинистки для каждого из вариантов.

– рассмотреть два режима работы машинисток: централизованный — поступающие документы ожидают своей очереди в общей папке; автономный — машинистки территориально рассредоточены и у каждой своя папка с входящими документами.

Перечислить дополнительные соображения, которые следовало бы принять во внимание при решении проблемы комплектования штатов и предложить способы их учета при моделировании.

7. Система обработки информации (СОИ) обрабатывает информацию, которая поступает в случайные моменты времени со средней интенсивностью λ (файл/мин).

Учитывая, что объем каждого файла и сложность его обработки различны, можно считать, что время обработки одной порции случайно и распределено по показательному закону с параметром μ (файл/мин). СОИ имеет память для хранения поступающей информации объемом до m файлов. Если очередная группа информации застанет всю память занятой, то она теряется. Одновременно может обрабатываться два файла информации. Со временем поступившая информация теряет свою ценность и в

среднем через t_3 мин. после поступления, если она не была обработана, становится практически ненужной.

Провести следующий анализ. Какой процент информации теряется из-за того, что пропускная способность системы не позволяет своевременно обрабатывать всю информацию и найти зависимость вероятности потерь от временного ограничения t_3 ($0,1 \text{ мин.} \leq t_3 \leq 1 \text{ мин.}$).

Исходные данные представлены в табл. 11.1.

Таблица 11.1

Вариант	λ	μ	m
1	10	10	5
2	10	6	2
3	10	10	1

Построить зависимость объема накопителя $m = f(t_3)$ для обеспечения вероятности потерь 0,01 для системы типа M/M/N.

8. Провести сравнительный анализ организации систем обработки информации для следующих вариантов структур (рис. 11.4).

Параметры и показатели для проведения сравнительного анализа указаны в табл. 11.2.

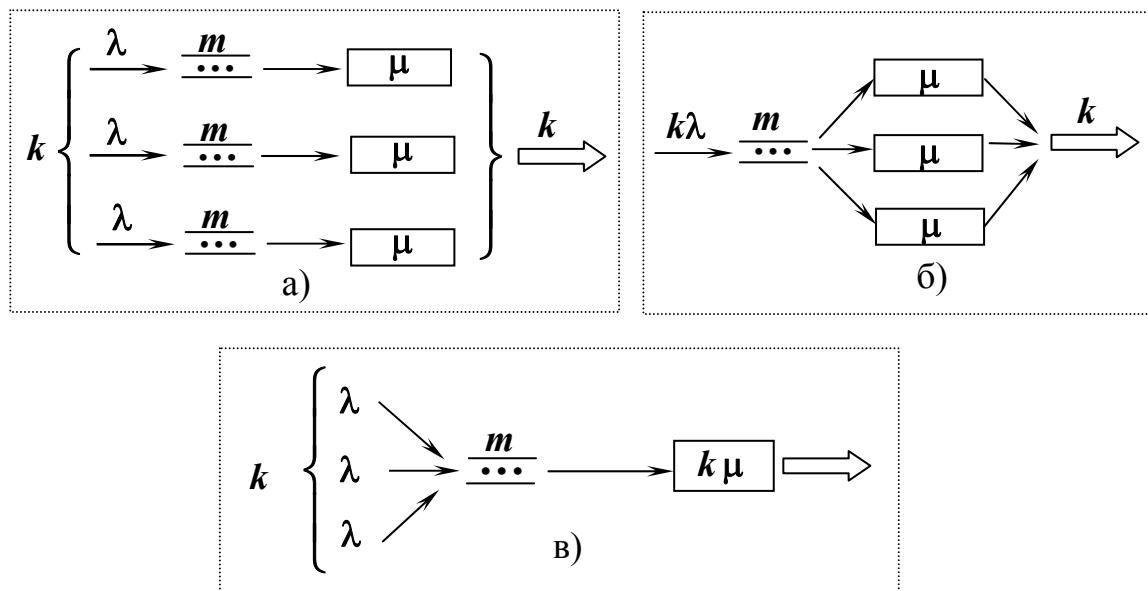


Рис. 11.4

Таблица 11.2

Вариант	k	m	Показатели
1	2	∞	$\bar{t}_{ож}, \bar{t}_c, \bar{n}_o, \bar{j}, \bar{K}_3$
2	3	∞	$\bar{t}_{ож}, \bar{t}_c, \bar{n}_o, \bar{j}, \bar{K}_3$
3	2	0	$\bar{t}_{ож}, \bar{t}_c, \bar{n}_o, \bar{j}, \bar{K}_3, P_{отк}$
4	3	0	$\bar{t}_{ож}, \bar{t}_c, \bar{n}_o, \bar{j}, \bar{K}_3, P_{отк}$
5	2	1	$\bar{t}_{ож}, \bar{t}_c, \bar{n}_o, \bar{j}, \bar{K}_3, P_{отк}$
6	3	1	$\bar{t}_{ож}, \bar{t}_c, \bar{n}_o, \bar{j}, \bar{K}_3, P_{отк}$

Построить зависимость указанных показателей в функции от $\beta = \lambda/\mu; \beta = 0,1 \div 0,9$. Сравнительный анализ предполагает модели типа **M/M/1** для вариантов «а» и «в» и **M/M/K** для варианта «б».

9. Провести сравнительный анализ вариантов организации системы многоэтапной обработки информации для следующих вариантов структур (рис. 11.5).

Параметры и показатели для проведения сравнительного анализа приведены в табл. 11.3.

Построить зависимость указанных показателей в функции от $\beta = \lambda/\mu; \beta = 0,1 \div 0,9$. Сравнительный анализ предполагает использование моделей типа **M/M/1** и **M/M/K**.

Таблица 11.3

Вариант	κ	m_1	m_2	Показатели
		1	2	
1	2	∞	∞	$\bar{t}_c, \bar{n}_o, \bar{j}$
2	3	∞	∞	$\bar{t}_c, \bar{n}_o, \bar{j}$
3	2	∞	∞	$\bar{t}_c, \bar{n}_o, \bar{j}$
4	2	0	0	$\bar{t}_c, \bar{n}_o, \bar{j}$

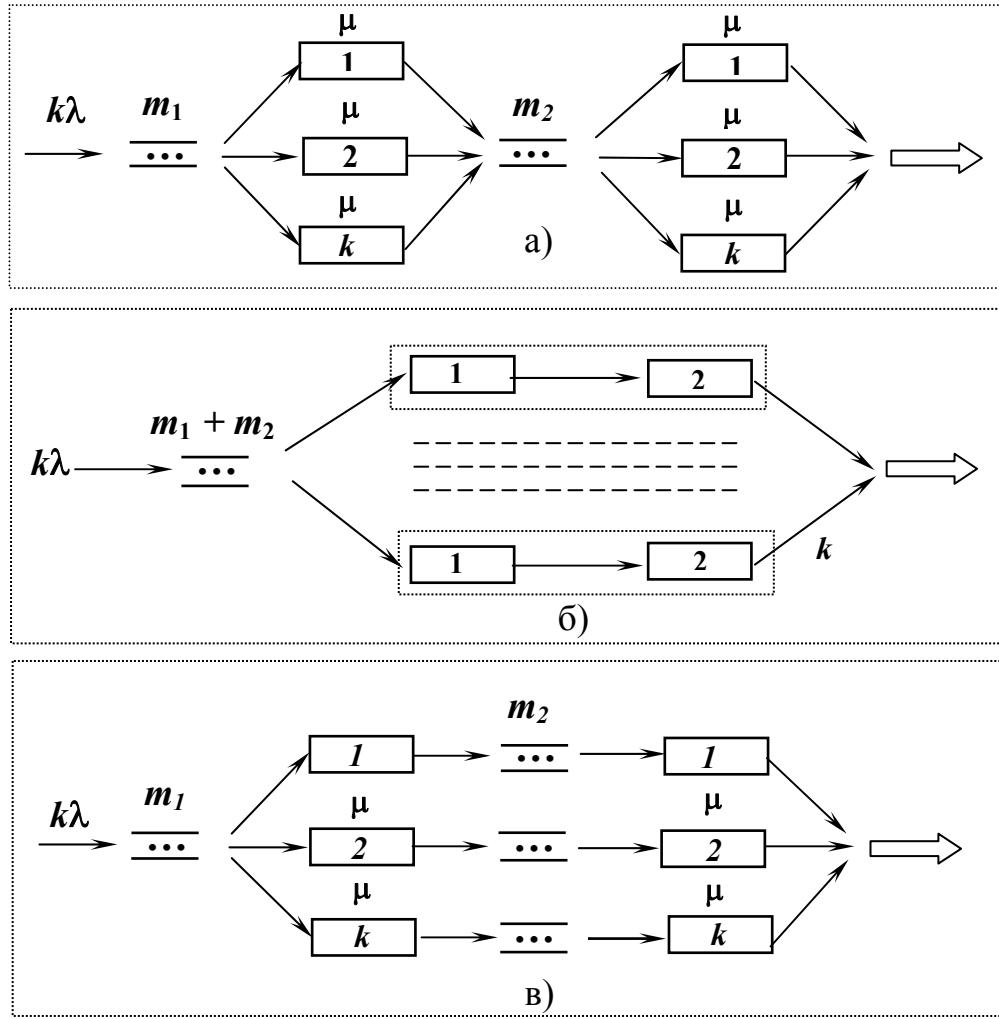


Рис. 11.5

10. Для мультипроцессорной вычислительной системы с числом процессоров n и с двумя независимыми модулями памяти, связанными между собой общей шиной, предложить эффективный алгоритм разрешения конфликтов на магистрали. Известно, что каждый из процессоров работает по программе независимой от другого. Каждая программа представляет чередующуюся последовательность фаз счета и обмена с памятью. Средние значения длительностей фаз счета и обмена заданы: $\bar{\tau}_i$ и \bar{t}_i , ($i = \overline{1, n}$). Структура системы представлена на рис. 11.6.

Определить коэффициент загрузки магистрали и временные задержки в выполнении каждой из программ при следующих условиях:

– выбрать бесприоритетную дисциплину разрешения конфликтов и провести анализ при различных характеристиках программ и усредненных характеристиках;

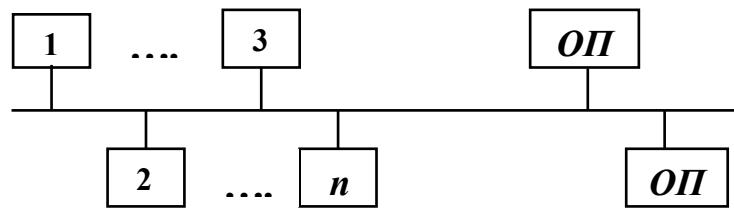


Рис. 11.6

– проанализировать указанные приоритетные правила разрешения конфликтов (приоритет убывает с номером задачи), приоритет относительный и абсолютный.

Исходные данные представлены в табл. 11.4 (время измеряется в относительных единицах).

Таблица 11.4

Варианты	Программы						Приоритеты		
	1		2		3				
	τ	t	τ	t	τ	t	1	2	3
1	30	5	40	5	50	5	1	2	3
							3	2	1
2	30	5	40	10	50	20	1	2	3
							3	2	1
3	30	20	40	10	50	5	1	2	3
							3	2	1
4	30	5	30	10	30	20	1	2	3
							3	2	1

11. Рассматривается двухфазная система периферийных технических средств в системе обработки данных с параметрами первой фазы k_1 , μ_1 и второй фазы m , k_2 , μ_2 . Структура системы представлена на рис. 11.7.

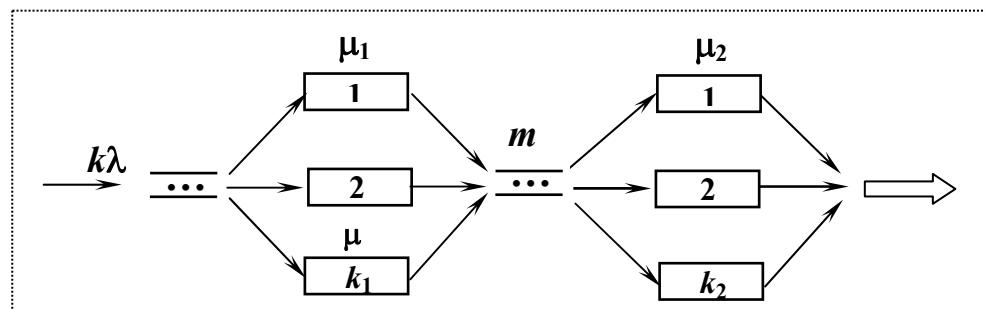


Рис. 11.7

В первой фазе очередь не ограничена, во второй фазе очередь ограничена объемом накопителя m . Потери внутри системы невозможны, так как, если приборы второй фазы заняты и накопитель заполнен, происходит последовательная блокировка первой фазы. Для режима пик-нагрузок (на входе первой фазы всегда существует очередь):

- определить предельную пропускную способность λ_n ;
- для $\lambda = 0,9\lambda_n$ определить среднее время пребывания требования в системе, включая и очередь перед первой фазой;
- для $\lambda = 0,2\lambda_n$ определить время пребывания требования в системе, используя принцип декомпозиции, при этом не учитывается ограниченность объема накопителя m — считается, что очередь во второй фазе не достигает своего предела.

Исходные данные представлены в табл. 11.5.

Таблица 11.5

Вариант	k_1	μ_1	m	k_2	μ_2
1	2	15	1	2	10
2	3	10	2	2	15
3	3	30	2	1	20
4	2	40	1	3	10

12. Рассмотреть задачу 4 при условии ограниченной очереди на входе системы (длина очереди l).

Приборы первой фазы могут полностью завершить обслуживание требования с вероятностью p или передать на окончательное обслуживание во вторую фазу с вероятностью $1-p$, причем, если приборы второй фазы заняты, то принятое требование не обслуживается и ожидает в канале первой фазы, т. е. данный канал первой фазы переходит в состояние блокировки (не обслуживает требования, поступающие в систему).

Необходимо:

- определить вероятность потерь при исходных данных задачи 4 и дополнительных данных по p и l ($p = 0,5$, $l = 1$).
- провести сравнительный анализ результатов, полученных в п. «а» с результатами исследования при $p = 0$.

13. Рассматривается автоматизированная система контроля (АСК), работающая в мультипрограммном режиме, т. е. в режиме параллельного выполнения программ контроля. Каждая программа представляет собой

последовательность чередующихся фаз работы процессора и N специализированных внешних функциональных устройств, управление которыми ведется от процессора (генераторов, преобразователей, коммутаторов, устройств ввода-вывода, регистраторов, внешних запоминающих устройств). Характеристики однородных программ заданы в виде отношения

$$\alpha = \frac{\bar{\tau}_{np}}{\bar{\tau}_{fy}},$$

где $\bar{\tau}_{np}$ — средняя длительность фазы счета, $\bar{\tau}_{fy}$ — средняя длительность фазы работы функционального устройства.

Необходимо:

1) провести сравнительный анализ двух дисциплин обслуживания программ процессором для $n = 2$:

- дисциплина — первый пришел, первый обслужен;
- циклическая дисциплина, т. е. процессор последовательно переключается с одной программы на другую, причем, переключившись на очередную программу в случае отсутствия требования на работу по данной программе, процессор ожидает его поступления, обслуживает и только после этого переключается на следующую;

2) провести сравнительный анализ качества обслуживания при заданном n (дисциплина обслуживания программ: первым пришел – первым обслужился) в двух случаях:

- программа обращается к любому свободному функциональному устройству из N ;
- выбор программой любого функционального устройства равновероятен.

Рассмотреть представление АСК в виде сети массового обслуживания.

При оценке качества учитывать как загрузку процессора, так и задержки в выполнении программ. Исходные данные представлены в табл. 11.6.

Таблица 11.6

Вариант	n	N	a
1	5	4	0,2
2	8	10	0,1
3	3	5	0,3
4	3	3	0,4
5	4	8	0,3

14. Рассматривается вычислительный комплекс, состоящий из двух процессоров (П) и секционированного блока оперативной памяти (ОП), включающего две секции. Предполагается, что каждый процессор может равновероятно обратиться к любой секции. Требуется оценить производительность двухпроцессорного комплекса при условии, что среднее время выполнения команды в процессоре T_0 , интенсивность обращения к секции, $\nu/2$ интенсивность обслуживания памяти — μ , независимо от количества секций.

Для оценки производительности использовать марковский граф со следующим множеством состояний:

- ни один П не обратился к оперативной памяти (S_0), $|S_0| = 1$;
- один П обратился к ОП (S_1), $|S_1| = 4$:
 $\Pi_1 \rightarrow \text{ОП}_1, \quad \Pi_1 \rightarrow \text{ОП}_2, \quad \Pi_2 \rightarrow \text{ОП}_1, \quad \Pi_2 \rightarrow \text{ОП}_2$;
- оба процессора обратились к разным блокам памяти (S_2), $|S_2| = 2$:
 $(\Pi_1 \rightarrow \text{ОП}_1, \quad \Pi_2 \rightarrow \text{ОП}_2 \quad \text{и} \quad \Pi_1 \rightarrow \text{ОП}_2, \quad \Pi_2 \rightarrow \text{ОП}_1)$;
- оба процессора обратились к одному блоку памяти (S_3), $|S_3| = 4$:
 $\Pi_1, \Pi_2 \rightarrow \text{ОП}_1; \quad \Pi_2, \Pi_1 \rightarrow \text{ОП}_1; \quad \Pi_1, \Pi_2 \rightarrow \text{ОП}_2; \quad \Pi_2, \Pi_1 \rightarrow \text{ОП}_2$.

Здесь $|S_i|$ — мощность множества S_i , равная числу его элементов.

Сопоставить полученную оценку производительности со случаем, когда секционирование памяти не проводится.

15. Рассматривается одноканальная СМО типа **M/M/1**, $\lambda = 0,8$, $\mu = 1$.

Определить для системы **M/E₂/1** интенсивность входящего потока, при которой среднее количество требований в ней будет таким же, как и в системе **M/M/1**.

Определить для системы **E₂/M/1** интенсивность обслуживания, при которой среднее количество требований в ней будет таким же, как в системе типа **M/M/1**.

Провести анализ влияния последействия в обслуживании для систем типа **M/M/1**, **M/E₂/1**, **M/D/1** при одинаковой средней длительности обслуживания $\tau_{обсл} = 1$.

16. С целью экономии расходов на телефонные переговоры фирма решила арендовать три линии междугородной телефонной связи (МГТС) с тем, чтобы работники ее подразделений, расположенных в различных городах, могли вести между собой деловые переговоры.

Предположим, что стоимость аренды одной линии МГТС составляет C за час. Будем считать, что частота, с которой возникает необходимость использования линии МГТС, характеризуется пуассоновским распределением с $\lambda = 10$ вызовов в час, а длительности переговоров имеют экспоненциальное распределение со средним значением, равным 15 мин.

Обозначим через ω стоимость минуты ожидания работником соединения с абонентом (т. е. ожидания момента освобождения хотя бы одной из трех линий). Пусть $C = 1$.

Требуется определить диапазон значений ω , при которых решение арендовать упомянутые выше три линии МГТС будет оптимальным.

17. На вход управляющей вычислительной машины (УВМ) поступают данные из пяти каналов передачи данных. Запросы поступают и обслуживаются с интенсивностями λ_i и μ_i соответственно, ($i = \overline{1, 5}$).

Определить эффективную дисциплину обслуживания потоков, сочетающая абсолютные и относительные приоритеты, а также бесприоритетное обслуживание, т. е. используя смешанные приоритеты.

Какие показатели эффективности УВМ нечувствительны к выбору дисциплины обслуживания из множества предложенных.

Исходные данные в относительных единицах приведены в табл. 11.7.

Таблица 11.7

Варианты	Источники									
	λ_1	μ_1	λ_2	μ_2	λ_3	μ_3	λ_4	μ_4	λ_5	μ_5
1	2	5	1	10	3	40	4	40	1	10
2	3	15	10	40	15	30	1	100	2	100
3	4	10	1	15	1	15	1	15	2	10
4	1	20	2	40	10	100	3	20	1	10

18. Сеть автоматизированных рабочих мест (АРМ) организована на основе трех персональных ЭВМ, выполняющих функции интеллектуальных терминалов. АРМ имеет доступ к центральной ЭВМ через сервер. Задача на каждом терминале формируется в течение времени, распределенного по экспоненциальному закону, со средним $t = 10$ сек, причем на выполнение в центральную ЭВМ направляется 25 % задач. Остальные задачи выполняются в автономном режиме в среднем за 8 сек.

Время обслуживания на сервере и центральной ЭВМ имеет экспоненциальное распределение с интенсивностями $\mu_1 = 1 c^{-1}$ и $\mu_2 = 10 c^{-1}$, соответственно. После выполнения задачи в центральной ЭВМ она возвра-

щается через сервер на соответствующий терминал, инициируя формирование новой задачи.

Определить среднее время реакции системы.

19. Вычислительная система имеет две разнотипные ЭВМ (ЭВМ–1 и ЭВМ–2), которые обслуживают сеть активных терминалов. Задачи пользователей образуют пуассоновский поток с интенсивностью $\lambda = 0,2 \text{ зад/с}$, а время выполнения заданий в ЭВМ имеет экспоненциальное распределение с математическим ожиданием $\bar{t}_1 = 8 \text{ с.}$ (ЭВМ–1) и $\bar{t}_2 = 12 \text{ с.}$ (ЭВМ–2).

Задачи пользователей выполняются в мультипрограммном режиме, причем область памяти каждой ЭВМ разделена на $m = 10$ блоков (одно задание занимает один блок памяти).

Если поступившее задание застает ЭВМ–1 занятой, то оно направляется на ЭВМ–2. После выполнения в ЭВМ 25% всех заданий обслуживается сетевым принтером, причем распечатка одного листа занимает $12 \text{ с} \pm 8 \text{ с.}$

Определить среднее время пребывания задания в системе.

20. Определить оптимальное количество аппаратов автоматического контроля качества деталей.

Если очередная деталь, двигающаяся по конвейеру, застает все контролирующие приборы занятыми, то она проходит на отгрузку без контроля. Цена аппарата 10000 рублей , эксплуатационные расходы на содержание работающего аппарата 200 рублей , а простаивающего — $100 \text{ рублей/сутки.}$

Потери по рекламации от возможного получения потребителем бракованной детали — 20 руб. Время контроля одной детали распределено по экспоненциальному закону с параметром $\mu = 0,2 \text{ мин}^{-1}$. Поток является простейшим с параметром $\lambda = 0,8 \text{ мин}^{-1}$. Срок службы аппарата равен 100 суткам. Вероятность появления бракованной детали на входе равна 10^{-2} .

12. СТОХАСТИЧЕСКИЕ СЕТЕВЫЕ МОДЕЛИ СМО

12.1. ПРЕДПОСЫЛКИ И ЦЕЛИ ПРИМЕНЕНИЯ АППАРАТА СЕТЕЙ СМО

Методы теории массового обслуживания, основанные на представлении моделируемой системы в пространстве дискретных состояний, являются универсальными и активно используются для исследования различных объектов. К ним могут относиться не только технические системы, такие как вычислительные системы, системы автоматизации или компьютерные сети, но и организационно-производственные системы широкого назначения. Однако усложнение структур и режимов реальных систем затрудняет применение классических методов теории массового обслуживания ввиду возрастающей размерности решаемой задачи (в частности, порядка уравнений Чепмена-Колмогорова для случая протекания марковского дискретного стохастического процесса). Одним из возможных путей преодоления размерности является использование моделей в форме сетей массового обслуживания.

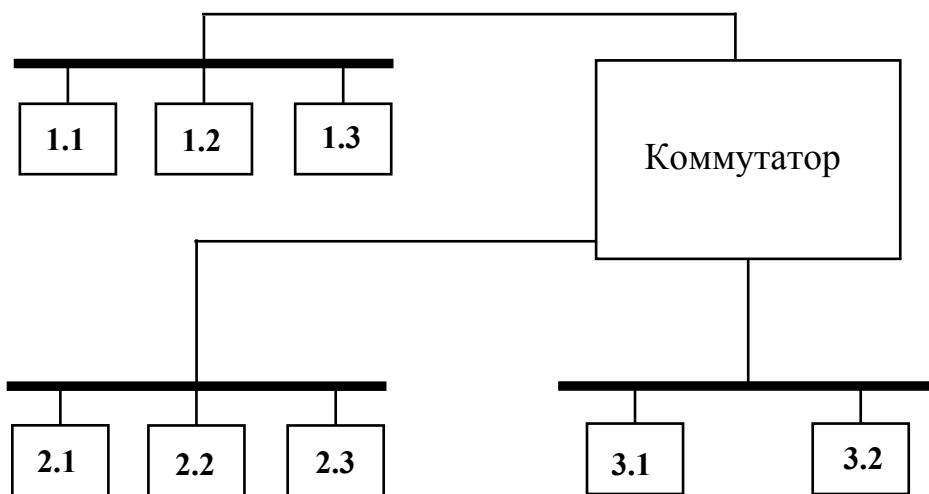


Рис. 12.1

Сеть массового обслуживания представляет собой совокупность конечного числа обслуживающих узлов, в которой циркулируют заявки, переходящие в соответствии с маршрутной матрицей из одного узла в другой. Узел всегда является разомкнутой СМО.

В качестве примера может быть рассмотрена упрощенная структура локальной вычислительной сети (ЛВС), в состав которой входят коммути-

тор и три сегмента, объединяющих 8 рабочих станций (рис. 12.1). Рабочие станции нагружают сегменты, генерируя, независимо друг от друга, пакеты, которые перемещаются по элементам ЛВС в соответствии с вероятностной матрицей передач. Если магистраль сегмента организована по стандарту ETHERNET, то в каждый момент времени в ней может находиться не более одного пакета.

Для построения сети СМО необходимо определить, что является заявкой, в чем состоит «обслуживание заявки», какими ресурсами моделируемой системы оно осуществляется. Ответы на эти вопросы не всегда очевидны.

Топология сети СМО может не повторять топологию моделируемой системы. Адекватность сетевой модели, в первую очередь, определяется точностью воспроизводимых ею временных задержек.

В данном случае пакеты интерпретируются как заявки, а относительная независимость рабочих станций при их генерации от загруженности ЛВС означает присутствие в модели внешнего источника заявок. На рис. 12.2 приведена сеть СМО для рассматриваемого примера.

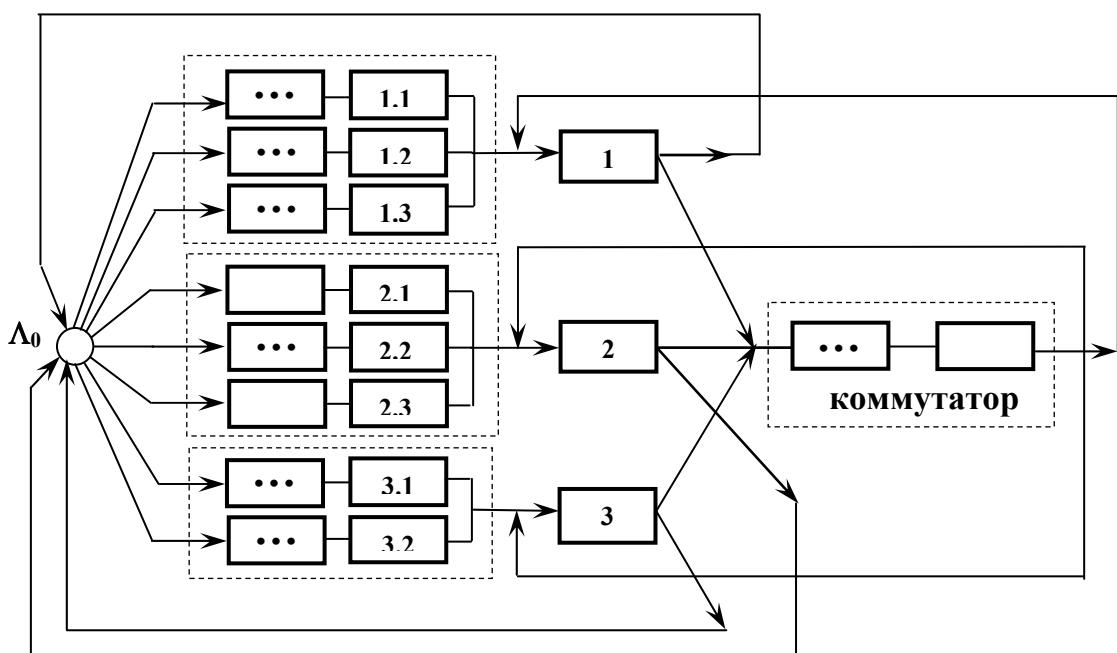


Рис. 12.2

Заявки попадают в сеть из внешнего источника, причем его интенсивность равна суммарной интенсивности пакетов-заявок, поступающих в ЛВС от всех рабочих станций.

Как только заявка получает полное обслуживание в сети СМО, то есть пакет принимается адресатом, она покидает сеть и возвращается в ис-

точник Λ_0 . Каждому ресурсу ЛВС в сети массового обслуживания поставлен в соответствие узел — СМО определенного типа, а именно:

- рабочей станции — одноканальный узел с ограниченным накопителем, позволяющим хранить некоторое количество заявок-пакетов, ожидающих отправки;
- ETHERNET-магистрали сегмента — одноканальный узел без накопителя;
- коммутатору — одноканальный узел с ограниченным накопителем.

Если число выполняемых заданий в моделируемой системе характеризуется постоянством, то для неё, скорее всего, будет построена замкнутая сеть СМО, в которой отсутствует внешний источник заявок. Естественно, это не единственный, (хотя и важный) признак, по которому различаются сети массового обслуживания. Приведенная ниже классификация сетей массового обслуживания позволяет оценить спектр используемых сетевых моделей и их возможности адекватного представления реальных систем.

Цель классификации состоит в определении типа сети, задающего способы расчета параметров её функционирования как основанных на аналитических методах, так и использующих методы имитационного моделирования. В основу классификации, представленной на рис. 12.3, положены две группы признаков: одна из которых определяет маршрутизацию заявок, другая — характеристики их обслуживания в узлах сети.

12.2. ОДНОРОДНЫЕ ЭКСПОНЕНЦИАЛЬНЫЕ СЕТИ СМО

12.2.1. Разомкнутые сети СМО. Анализ их характеристик

Аналитическое исследование возможно для стохастических сетей со следующими допущениями:

- сеть формируют узлы, объединяющие каналы с показательным законом обслуживания;
- сеть нагружена пуассоновским потоком однородных заявок;
- в каждом узле сети, очередь не ограничена, дисциплина обслуживания: первым пришёл — первым обслужен;
- сеть состоит из узлов типа $M/M/K$;
- сеть является линейной, т. е. вероятность поступления заявки в узел C_j за интервал времени $[t, t + \Delta t]$ является линейной комбинацией с постоянным коэффициентом π_{ij} вероятностей выхода заявок из различных узлов сети, включая и узел с номером j .



Рис. 12.3

***Формальное задание
линейных экспоненциальных разомкнутых сетей СМО***

Линейная экспоненциальная разомкнутая сеть СМО задаётся кортежем

$$\left\{ \lambda_0; M; \pi = \left\{ p_{ij} \right\}, \quad i, j = \overline{0, M}; \quad \vec{\mu} = \left\{ \mu_1, \dots, \mu_M \right\}; \quad \vec{m} = \left\{ m_1, \dots, m_M \right\} \right\}$$

где λ_0 — интенсивность внешнего пуссоновского источника заявок,

M — число узлов сети,

$\pi = \left\{ p_{ij} \right\}, \quad i, j = \overline{0, M}$ — стохастическая матрица передач,

$\vec{\mu} = \left\{ \mu_1, \dots, \mu_M \right\}$ — вектор, i -м компонентом которого является интенсивность обслуживания заявки в канале i -го узла сети,

$\vec{m} = \left\{ m_1, \dots, m_M \right\}$ — вектор, i -м компонентом которого является число каналов в i -м узле сети.

Для линейных экспоненциальных сетей будем рассматривать только установившийся режим, как наиболее часто встречающийся при решении практических задач. При наличии установившегося режима в разомкнутой сети справедлива следующая закономерность: интенсивность среднего суммарного потока на входе любой подсистемы равна средней суммарной интенсивности выходного потока из данной подсистемы, т. е. справедлива следующая система уравнений:

$$\lambda_j = \sum_{i=0}^M \lambda_i p_{ij}, \quad j = \overline{0, M}, \quad (12.1)$$

где λ_i — средняя суммарная интенсивность на выходе i -й подсистемы; λ_j — суммарная интенсивность на выходе j -й подсистемы; p_{ij} — вероятность поступления требований из i -й подсистемы по окончании обслуживания в j -ю подсистему.

В матричной форме можно записать данную систему уравнений следующим образом:

$$\pi' \lambda = \lambda, \quad (12.2)$$

где π' — транспонированная матрица передач; λ — вектор-столбец интенсивностей λ_i потоков требований, проходящих через подсистемы сети в установившемся режиме. Уравнение (12.2) можно представить в несколько

ином виде, используя понятие единичной матрицы E :

$$(\pi' - E)\lambda = \mathbf{0}.$$

Решение данного матричного уравнения существует, если определитель $|\pi' - E| = 0$. Так как

$$\sum_{j=0}^M p_{ij} = 1, \quad i = \overline{0, M}$$

и учитывая, что прибавление к элементам какой-либо строки элементов любой другой строки, умноженных на одно и то же число, не изменяет значения определителя, можно показать, что данное условие выполняется. Замыкание в разомкнутой сети происходит через источник требований, отображающий внешнюю среду, следовательно, можно считать, что каждый узел сети является возвратным, т. е. из любой подсистемы через некоторое количество шагов можно попасть в любую другую подсистему, в том числе, и в источник требований. Поэтому в такой сети существует установленный режим и ранг матрицы $\pi' - E$ равен M , т. е. для каждого значения интенсивности λ_0 источника требований существует единственное решение системы уравнений (12.1):

$$\lambda_j = \alpha_j \lambda_0, \quad j = \overline{1, M},$$

где α_j — коэффициент передачи от источника требований к j -му узлу.

Условие предотвращения образования бесконечной очереди в каждой из подсистем очевидно:

$$\alpha_j \lambda_0 < m_j \mu_j, \quad j = \overline{1, M},$$

где m_j — число каналов в j -м узле.

Это же условие определяет ограничение на интенсивность требований, поступающих из источника

$$\lambda_0 < \min_{(j)} \left(\frac{m_j \mu_j}{\alpha_j} \right). \quad (12.3)$$

Для иллюстрации применения системы уравнений (12.1) рассмотрим пример.

Пример 12.1

Определить стационарные вероятности стохастической сети, представленной вместе с матрицей передач на рис. 12.4. Пусть интенсивность

источника $\lambda_0 = 5 \text{ c}^{-1}$. Тогда в соответствии с (12.1) имеем следующую систему уравнений:

$$\begin{cases} \lambda_0 = \lambda_1 p_{10} \\ \lambda_1 = \lambda_0 + \lambda_2 + \lambda_3 \\ \lambda_2 = \lambda_1 p_{12} \\ \lambda_3 = \lambda_1 p_{13} \end{cases},$$

откуда следует, что. $\alpha_1 = [p_{10}]^{-1}$, $\alpha_2 = p_{12}[p_{10}]^{-1}$, $\alpha_3 = p_{13}[p_{10}]^{-1}$.

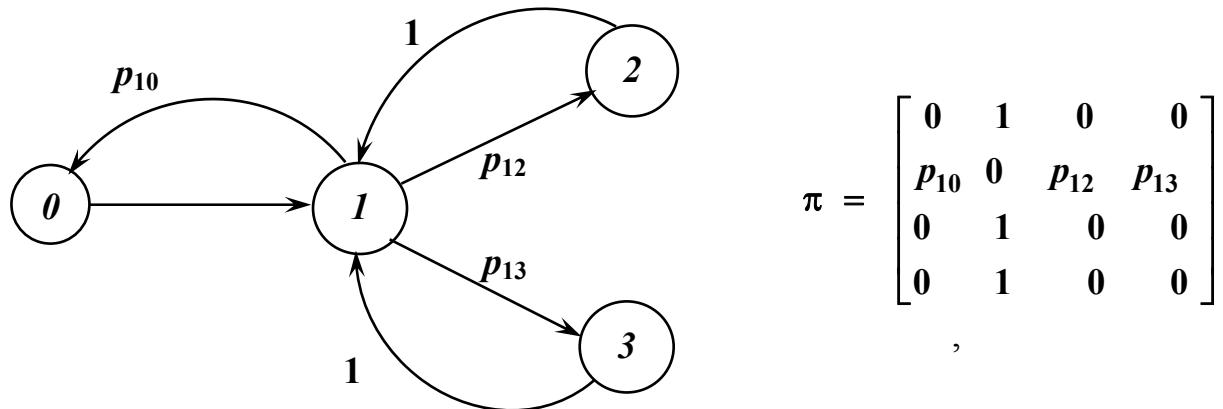


Рис. 12.4

Рассмотрим основные характеристики разомкнутых сетей в целом и их отдельных узлов в установившемся режиме. При определении характеристик сети используется понятие состояния сети.

Под состоянием сети понимается вектор $n = (n_1, n_2, \dots, n_M)$, в соответствии с которым в узле 1 содержится n_1 требований, в узле 2 — n_2 требований, в узле M — n_M требований. Поэтому наиболее полной характеристикой сети является вероятность $P(n_1, n_2, \dots, n_M)$. Если разомкнутая сеть представляет собой совокупность узлов с неограниченными очередями, то множество состояний $\{n_i\}$ каждого узла и множество состояний $S = \{n\}$ сети бесконечны.

Важным аналитическим инструментом анализа характеристик сетей является теорема разбиения Джексона [29], содержащая достаточные условия, при выполнении которых разомкнутая сеть общего вида является совокупностью независимых полнодоступных систем с очередями. Ниже эта теорема приводится без доказательства, которое можно найти в [29].

Теорема Джексона

Если для рассматриваемой сети выполнены условия:

- все входящие потоки являются пуассоновскими,
- все маршруты переходов требований между подсистемами управляются
- вероятностной матрицей передач,
- все длительности обслуживания имеют экспоненциальное распределение,
- дисциплина выбора заявок из очереди в каждой подсистеме не зависит от длительностей обслуживания и маршрута заявок, то

$$P(n_1, n_2, \dots, n_j, \dots, n_M) = \prod_{j=1}^M P_j(n_j), \quad (12.4)$$

где

$$P_j(n_j) = P_j(0) \prod_{i=1}^{n_j} \frac{\lambda_j}{\mu_j(i)}; \quad (12.5)$$

$$\lambda_j = \alpha_j \lambda_0; \quad \mu_j(i) = i \mu_j, \quad i \leq m_j; \quad \mu_j(i) = m_j \mu_j, \quad i > m_j.$$

Интерпретация соотношений (12.5) для разомкнутой сети, когда все узлы сети являются одноканальными системами массового обслуживания, может быть представлена следующим образом:

где

$$P_j(n_j) = P_j(0) (\rho_j)^{n_j},$$

$$\rho_j = \frac{\alpha_j \lambda_0}{\mu_j}, \quad P_j(0) = 1 - \rho_j, \quad (12.6)$$

а для многоканальных систем:

$$P_j(n_j) = \begin{cases} P_j(0) \cdot \left(\frac{\lambda_j}{\mu_j}\right)^{n_j} \frac{1}{n_j!}, & n_j \leq m_j, \\ P_j(0) \cdot \left(\frac{\lambda_j}{\mu_j}\right)^{n_j} \frac{1}{m_j! (m_j)^{n_j-m_j}}, & n_j > m_j; \end{cases} \quad (12.7)$$

$$P_j(0) = \left[\sum_{n_j=0}^{m_j-1} \frac{1}{n_j!} \left(\frac{\lambda_j}{\mu_j}\right)^{n_j} + \frac{\rho_j^{m_j}}{m_j! (1-\rho_j/m_j)} \right];$$

$$\text{где } \rho_j = \frac{\lambda_j}{\mu_j} = \frac{\alpha_j \lambda_0}{\mu_j}.$$

В соответствии с рассмотренным принципом декомпозиции задачи анализа сети на задачи анализа отдельных подсистем получим следующие формулы:

– среднее число требований, находящихся в сети

$$\bar{n} = \sum_{j=1}^M \bar{n}_j, \quad (12.8)$$

– среднее число требований, ожидающих обслуживания в сети

$$\bar{l} = \sum_{j=1}^M \bar{l}_j, \quad (12.9)$$

– среднее время пребывания требования в сети

$$\bar{t}_c = \sum_{j=1}^M \alpha_j \bar{t}_j, \quad (12.10)$$

– среднее время ожидания требования в сети

$$\bar{t}_{ож} = \sum_{j=1}^M \alpha_j \bar{t}_{ожj}, \quad (12.11)$$

где $\alpha_j = \lambda_j / \lambda_0$ определяет среднее число прохождений требования через j -й узел, соответствующее одному требованию, поступившему из 0-го узла в сеть.

Формулы для $\bar{n}_j, \bar{l}_j, \bar{t}_j, \bar{t}_{ожj}$ и других параметров, характеризующих отдельные подсистемы, сведены в табл. 12.1.

Рассмотренные этапы анализа однородных экспоненциальных разомкнутых сетей СМО можно кратко изложить в виде методики.

Методика анализа однородных экспоненциальных разомкнутых сетей СМО

1. Расчёт интенсивностей потоков заявок в узлах сети по формуле (12.1).
2. Проверка условий (12.3) наличия установившегося режима в сети.
3. «Разрезание» сети — расчёт локальных показателей узлов сети по формулам (12.6), (12.7) и соотношениям табл. 12.1.

4. «Сборка» сети — расчёт интегральных показателей сети по теореме Джексона (формулы (12.4), (12.5) и (12.6), (12.7)) и соотношениям (12.8) – (12.11).

Таблица 12.1

Основные соотношения для расчёта параметров узлов		
Вид параметра	Узел $M/M/1$	Узел $M/M/m$
P_0	$1 - \rho$	$\left[1 + \sum_{j=1}^m \frac{\rho^j}{j!} + \frac{\rho^{m+1}}{m!(m-\rho)} \right]^{-1}$
\bar{l}	$\frac{\rho^2}{1 - \rho}$	$\frac{P_0 \rho^{m+1}}{m \left(1 - \frac{\rho}{m}\right)^2 m!}$
\bar{n}	$\frac{\rho^2}{1 - \rho}$	$\bar{l} + \rho$
$\bar{t}_{ож}$	$\frac{\rho}{1 - \rho}$	$\frac{P_0 \rho^m}{m \cdot \mu \cdot m! \left(1 - \frac{\rho}{m}\right)^2}$
\bar{t}	$\frac{1}{\mu - \lambda}$	$\bar{t}_{ож} + \frac{1}{\mu}$

Пример 12.2

Определить среднее время пребывания заявки в разомкнутой сети, структура которой представлена на рис. 12.5. Структура данной сети отображает работу вычислительной системы (ВС) в составе процессора и двух внешних однородных запоминающих устройств, управляемых посредством мультиплексорного канала. Пусть задана следующая матрица передач:

$$\pi = \begin{bmatrix} 0 & 1 & 0 \\ 0,2 & 0 & 0,8 \\ 0 & 1 & 0 \end{bmatrix},$$

$$\lambda_0 = 0,1 \text{ c}^{-1}; \quad \mu_1 = 2 \text{ c}^{-1}; \quad \mu_2 = 1 \text{ c}^{-1}.$$

Тогда систему уравнений (12.1) можно записать так:

$$\begin{cases} \lambda_0 = 0,2 \cdot \lambda_1; \\ \lambda_1 = \lambda_0 + \lambda_2; \\ \lambda_2 = 0,8 \cdot \lambda_1, \end{cases}$$

$$\lambda_1 = 5\lambda_0 = 0,5 \text{ c}^{-1},$$

откуда

$$\lambda_2 = 0,8 \cdot 5\lambda_0 = 0,4 \text{ c}^{-1}.$$

В рассматриваемой сети стационарный режим существует, так как условие (12.3) выполняется: $0,1 < \min\left(\frac{2}{5}, \frac{1}{2}\right)$.

Среднее время пребывания требования в 1-м узле есть сумма средних времен ожидания в очереди и обслуживания:

$$\bar{t}_1 = \frac{1}{\mu_1} + \frac{\lambda_1}{\mu_1(\mu_1 - \lambda_1)} = \frac{1}{\mu_1 - \lambda_1} = \frac{1}{2 - 0,5} = 0,67 \text{ c.}$$

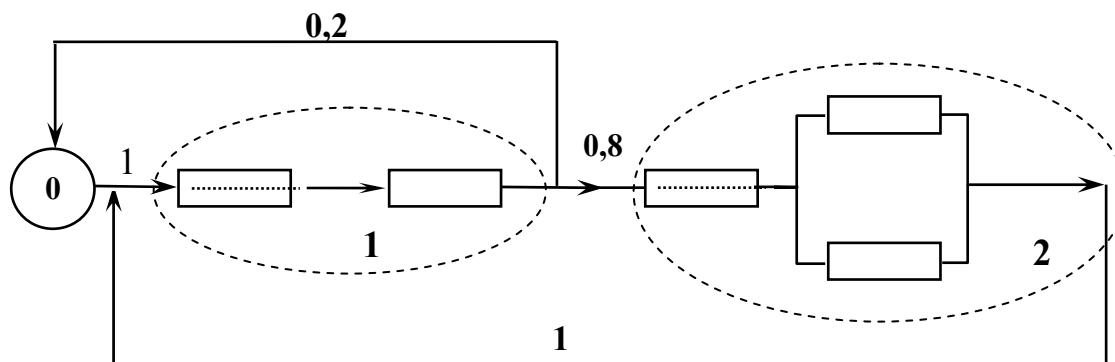


Рис. 12.5. Представление ВС в виде разомкнутой сети

Среднее время пребывания требования во 2-м узле определяется аналогично, но с учетом 2-х каналов обслуживания:

$$\bar{t}_2 = \frac{1}{\mu_2} + \frac{\rho_2^2 P_0}{2\mu_2 \cdot 2! \left(1 - \frac{\rho_2}{2}\right)^2},$$

где $P_0 = \left[1 + \rho_2 + \rho_2^2 \cdot \frac{1}{2!} + \frac{\rho_2^3}{2!(2-\rho_2)} \right]^{-1} = 0,67,$

$$\bar{t}_2 = 1 + \frac{(0,4)^2}{2 \cdot 1 \cdot 2! \left(1 - \frac{0,4}{2}\right)^2} \cdot 0,67 = 1,04 \text{ с.}$$

Среднее время пребывания требования в сети по формуле (10.11) равно:

$$\bar{t}_c = \sum_{j=1}^2 \alpha_j \bar{t}_j = 5 \cdot 0,67 + 4 \cdot 1,04 = 7,51 \text{ с.}$$

Анализ характеристик систем, структура которых отображается в виде многофазных систем массового обслуживания, иногда можно рассматривать как частный случай разомкнутых сетей массового обслуживания.

12.2.2. Замкнутые сети СМО. Анализ их характеристик

В замкнутых сетях всегда имеется конечное число требований, которое сохраняется постоянным при изменении их распределения по отдельным узлам. Мощность $|S|$ множества $S(N, M)$ состояний замкнутой сети определяется по правилам комбинаторики как число вариантов размещения N заявок по M узлам:

$$|S| = C_{N+M-1}^N \quad (12.8)$$

Формальное задание линейных экспоненциальных замкнутых сетей СМО

Линейная экспоненциальная замкнутая сеть СМО задаётся кортежем:

$$\left\{ N; M; \pi = \left\{ p_{ij} \right\}, i, j = \overline{1, M}; \vec{\mu} = \left\{ \mu_1, \dots, \mu_M \right\}; \vec{m} = \left\{ m_1, \dots, m_M \right\} \right\},$$

где N — постоянное число заявок в сети, M — число узлов сети,

$\pi = \left\{ p_{ij} \right\}, i, j = \overline{1, M}$ — стохастическая матрица передач,

$\vec{\mu} = \left\{ \mu_1, \dots, \mu_M \right\}$ — вектор, i -м компонентом которого является интенсивность обслуживания заявки в канале i -го узла сети, $\vec{m} = \left\{ m_1, \dots, m_M \right\}$ — вектор, i -м компонентом которого является число каналов в i -м узле сети.

В замкнутой сети обязательно должно выполняться условие:

$$\pi_{i0} = 0; \pi_{0j} = 0, i = \overline{0, M}, j = \overline{0, M},$$

так как требования не покидают сеть, а просто перераспределяются по её узлам. В этом случае матрица передач имеет размерность $M \times M$ и система уравнений, эквивалентная системе уравнений (12.1), выглядит следующим образом:

$$\lambda_j = \sum_{i=1}^M \lambda_i p_{ij}, \quad j = \overline{1, M}. \quad (12.9)$$

Следовательно, для существования решения определитель матрицы $\pi' - E$ должен быть равен нулю. Можно показать так же, как и для разомкнутых систем, что в этом случае ранг матрицы равен $M - 1$, т. е. в установленном режиме справедлива следующая система решений:

$$\lambda_j = \alpha_j \lambda_1, \quad \alpha_j = \frac{\lambda_j}{\lambda_1},$$

при условии, что интересующие нас интенсивности λ_j выражены через интенсивность λ_1 .

Система (12.9) имеет бесконечное множество решений $\{\lambda_1, \lambda_2, \dots, \lambda_M\}$, получаемых с точностью до мультипликативной константы. Это означает, что точное решение возможно только для относительных интенсивностей $\alpha_j, j = \overline{1, M}$:

$$\begin{cases} \alpha_j = \sum_{i=1}^M \alpha_i p_{ij}, \quad j = \overline{1, M}; \\ \alpha_j = 1. \end{cases} \quad (12.10)$$

Так как в замкнутой сети содержится конечное число требований, то в любом узле может образоваться только конечная очередь. Поэтому анализ решения системы (12.10) позволяет выявить только наиболее нагруженную подсистему, характеризуемую условием:

$$\max_{(j)} \left\{ \alpha_j / m_j \mu_j \right\}.$$

Для определения вероятностей состояний замкнутой сети используется, тот же подход, что и для разомкнутой сети, строгое доказательство которого приведено в [31] вероятности состояний сети выражаются в мультипликативной форме через вероятности состояний составляющих сеть систем массового обслуживания. Однако полная декомпозиция задачи анализа характеристик замкнутых сетей невозможна, так как в сети содержит-

жится конечное число требований. Для учёта этого факта вводится дополнительный множитель A , смысл которого может быть пояснён следующим образом.

Представим себе разомкнутую сеть, состоящую из тех же самых узлов, что и рассматриваемая замкнутая сеть. Пусть интенсивность входного источника и матрица передач таковы, что на входы узлов разомкнутой сети заявки поступают с той же средней интенсивностью, что и на аналогичные входы замкнутой сети. Отличие двух сетей состоит в том, что число состояний разомкнутой сети бесконечно, а замкнутой конечно и соответствует (12.8). Выделим подмножество состояний разомкнутой сети, характеризуемое условием:

$$\sum_{j=1}^M n_j = N,$$

где n_j — число заявок в j -м узле сети, и определим в соответствии с формулой (12.4) сумму вероятностей всех C_{N+M-1}^N интересующих нас состояний «эквивалентной» разомкнутой сети. Величина, обратная этой сумме, и есть множитель A .

В соответствии с этим:

$$P(n_1, n_2, \dots, n_j, \dots, n_M) = A \cdot \prod_{j=1}^M P_j(n_j). \quad (12.11)$$

Так как $\sum_{n \in S(N,M)} P(n_1, n_2, \dots, n_j, \dots, n_M) = 1$,

то:

$$A = \left[\sum_{n \in S(N,M)} \prod_{j=1}^M P_j(n_j) \right]^{-1},$$

$$P(n_1, n_2, \dots, n_j, \dots, n_M) = \frac{\prod_{j=1}^M P_j(n_j)}{\sum_{n \in S(N,M)} \prod_{j=1}^M P_j(n_j)}. \quad (12.12)$$

Из (12.6) и (12.7) следует, что вероятности $P_j(n_j)$ для одноканальных и многоканальных систем могут быть представлены произведением:

$$P_j(n_j) = d_j(n_j) P_j(0),$$

причем для одноканальных систем

$$d_j(n_j) = (\rho)^{n_j}, \quad n_j = \overline{1, N}, \quad (12.14)$$

а для многоканальных систем:

$$d_j(n_j) = \begin{cases} (\rho_j)^{n_j} \frac{1}{n_j!}, & n_j \leq m_j; \\ (\rho_j)^{n_j} \frac{1}{m_j! m_j^{(n_j-m_j)}}, & n_j > m_j. \end{cases} \quad (12.15)$$

На основании (12.14) и (12.15) можно записать:

$$d_j(n_j) = \frac{\lambda_j^{n_j}}{\prod_{k=1}^{n_j} \mu_j(k)}, \quad \text{где } \mu_j(k) = \begin{cases} k \mu_j, & j < m_j, \\ m_j \mu_j, & j \geq m_j. \end{cases} \quad (12.16)$$

С учётом этих преобразований из формулы (12.12) имеем:

$$P(n_1, n_2, \dots, n_j, \dots, n_M) = \frac{\prod_{j=1}^M P_j(0) \prod_{j=1}^M d_j(n_j)}{\prod_{j=1}^M P_j(0) \sum_{n \in S(N, M)} \prod_{j=1}^M d_j(n_j)}. \quad (12.17)$$

Можно показать, что вероятность $P(n_1, \dots, n_M)$ не зависит от абсолютного значения интенсивности λ_1 при условии, что все другие интересующие интенсивности λ_i выражены через интенсивность λ_1 :

$$P(n_1, \dots, n_M) = \frac{\lambda_1^N \prod_{j=1}^M C_j(n_j)}{\lambda_1^N \cdot \sum_{\sum n_j=N} \prod_{j=1}^M C_j(n_j)}, \quad (12.18)$$

причём для одноканальных систем:

$$C_j(n_j) = (\alpha_j / \mu_j)^{n_j}, \quad (12.19)$$

а для многоканальных систем:

$$C_j(n_j) = \begin{cases} \left(\frac{\alpha_j}{\mu_j}\right)^{n_j} \cdot \frac{1}{n_j!}, & n_j \leq m_j, \\ \left(\frac{\alpha_j}{\mu_j}\right)^{n_j} \cdot \frac{1}{m_j! m_j^{(n_j-m_j)}}, & n_j > m_j \end{cases} . \quad (12.20)$$

По аналогии с (12.16):

$$C_j(n_j) = \frac{\alpha_j^{n_j}}{\prod_{k=1}^{n_j} \mu_j(k)}, \quad (12.21)$$

$$\text{где } \mu_j(k) = \begin{cases} k \mu_j, & j < m_j \\ m_j \mu_j, & j \geq m_j \end{cases} .$$

Анализ проведённых преобразований позволяет сделать вывод: структура формулы (12.18) такова, что параметр α_j достаточно определить с точностью до мультипликативной константы. Поэтому в соотношениях (12.19) – (12.21) можно вместо параметра α_j ввести параметр ω_j , отличающийся от него постоянным множителем. По смыслу ω_j представляет собой вероятность поступления в j -й узел некоторого помеченного требования при его очередном переходе из узла в узел замкнутой сети:

$$\omega_j = \frac{\alpha_j}{\sum_{i=1}^M \alpha_i} .$$

Вероятности ω_j удовлетворяют нормирующему условию:

$$\sum_{j=1}^V \omega_j = 1.$$

Можно показать, что значения ω_j могут быть определены непосредственно из системы уравнений, подобной (12.10):

$$\omega_j = \sum_{i=1}^M \omega_i \cdot p_{ij}, \quad j = \overline{1, M}, \quad (12.22)$$

с учётом нормирующего условия $\sum_{j=1}^M \omega_j = 1$.

Замена λ_j на ω_j приводит к следующей записи формулы для определения вероятности состояния замкнутой сети:

$$P(n_1, \dots, n_M) = \frac{1}{G_M(N)} \cdot \prod_{i=1}^M Z_i(n_i), \quad (12.23)$$

где

$$Z_i(n_i) = \frac{\omega_i^{n_i}}{\prod_{j=1}^{n_i} \mu_i(j)}. \quad (12.24)$$

Здесь $G_M(N)$ — нормирующая константа замкнутой сети с M узлами и N заявками:

$$G_M(N) = \sum_{n \in S(N, M)} \prod_{i=1}^M Z_i(n_i). \quad (12.25)$$

Пример 12.3

Определить коэффициент загрузки процессора в вычислительной системе (ВС), работающей в мультипрограммном режиме, структура которой представлена на рис. 12.6.

Структура сети СМО отображает взаимодействие процессора C_1 с устройствами внешней памяти C_2 и C_3 . Постоянное число N заявок равно числу выполняемых программ: $N = 3$. Число узлов M равно числу элементов ВС: $M = 3$. Все узлы являются одноканальными. Матрица π описывает передачу управления между элементами ВС в рамках выполнения программы.

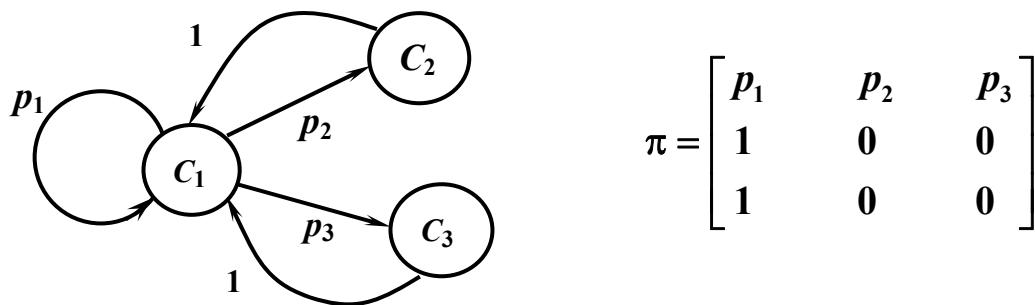


Рис. 12.6

Решив систему уравнений (12.22), найдём вероятности ω_1 и ω_2 :

$$\begin{cases} \omega_2 = \omega_1 p_2; & \omega_3 = \omega_1 p_3 \\ \omega_1 = \omega_1 p_1 + \omega_2 + \omega_3 \end{cases}$$

С учётом нормирующего условия $\sum_{i=1}^3 \omega_i = 1$

$$\omega_1 = \frac{1}{p_2 + p_3 + 1}, \quad \omega_2 = \frac{p_2}{p_2 + p_3 + 1}, \quad \omega_3 = \frac{p_3}{p_2 + p_3 + 1}.$$

Интенсивности обслуживания μ_j в одноканальных узлах сети считаются заданными.

В соответствии с (12.24) $Z_i(n_i) = \left(\frac{\omega_i}{\mu_i}\right)^{n_i}$.

Коэффициент η загрузки процессора определяется как разность между единицей и суммой вероятностей состояний сети, в которых имеет место простой узла C_1 :

$$\eta = 1 - (P(0,3,0) + P(0,0,3) + P(0,1,2) + P(0,2,1)).$$

В соответствии с (12.23) получим:

$$\eta = 1 - \frac{\left(\frac{\omega_2}{\mu_2}\right)^3 + \left(\frac{\omega_3}{\mu_3}\right)^3 + \left(\frac{\omega_2}{\mu_2}\right) \cdot \left(\frac{\omega_3}{\mu_3}\right)^2 + \left(\frac{\omega_2}{\mu_2}\right)^2 \cdot \left(\frac{\omega_3}{\mu_3}\right)}{G_3(3)},$$

где $G_3(3) = \sum_{(n_1+n_2+n_3=3)} \left(\frac{\omega_1}{\mu_1}\right)^{n_1} \cdot \left(\frac{\omega_2}{\mu_2}\right)^{n_2} \cdot \left(\frac{\omega_3}{\mu_3}\right)^{n_3}$.

Число произведений, суммируемых при расчёте нормирующей константы $G_3(3)$, равно числу состояний сети: $C_{N+M-1}^N = C_5^3 = 10$.

12.2.3. Определение основных показателей качества обслуживания в узлах замкнутой сети СМО

Введем следующие обозначения для средних значений показателей качества обслуживания в j -м узле сети при условии пребывания в ней N заявок:

$\overline{n_j}(N)$ — среднее число заявок в узле;

$\overline{n_j^{ож}}(N)$ — среднее число заявок в очереди на обслуживание;

$\overline{n_j^{обсл}}(N)$ — среднее число занятых приборов;

$\overline{t_j}(N)$ — среднее время пребывания заявки в узле;

$\overline{t_j^{ож}}(N)$ — среднее время ожидания заявкой обслуживания;

$\overline{t_j^{обсл}}(N)$ — среднее время обслуживания заявки (равное $\frac{1}{\mu_j}$).

Исходными данными, необходимыми для определения любых показателей качества обслуживания в узлах замкнутой сети являются вероятности её состояний, определяемые в соответствии с формулой (12.23). Это в полной мере относится к среднему числу $\overline{n_j}(N)$ заявок в j -м узле, рассчитываемому по формуле для математического ожидания дискретной случайной величины:

$$\overline{n_j}(N) = \sum_{n \in S(N, M)} n_j P(n_1, \dots, n_j, \dots, n_M) \quad (12.26)$$

Тот же подход используется для определения показателей $\overline{n_j^{ож}}(N)$ и $\overline{n_j^{обсл}}(N)$:

$$\overline{n_j^{ож}}(N) = \sum_{\substack{n \in S(N, M) \\ n_j > m_j}} (n_j - m_j) P(n_1, \dots, n_j, \dots, n_M), \quad (12.27)$$

$$\overline{n_j^{обсл}}(N) = \sum_{\substack{n \in S(N, M) \\ n_j < m_j}} n_j P(n_1, \dots, n_j, \dots, n_M) + m_j \left[\sum_{\substack{n \in S(N, M) \\ n_j > m_j}} P(n_1, \dots, n_j, \dots, n_M) \right] \quad (12.28)$$

Определение временных показателей обслуживания в узле замкнутой сети

Заявки, находящиеся в узле сети, заняты ожиданием либо обслуживанием, поэтому:

$$\overline{t_j}(N) = \overline{t_j^{ож}}(N) + \overline{t_j^{обсл}}(N), \quad (12.29)$$

$$\overline{n_j}(N) = \overline{n_j^{ож}}(N) + \overline{n_j^{обсл}}(N). \quad (12.30)$$

При наличии стационарного режима в сети справедливо следующее

тождество, представляющее собой вариант записи правила Литтла:

$$\frac{\bar{t}_j(N)}{n_j(N)} = \frac{\bar{t}_j^{ож}(N)}{n_j^{ож}(N)} = \frac{\bar{t}_j^{обсл}(N)}{n_j^{обсл}(N)} = \frac{I/\mu_j}{n_j^{обсл}(N)}. \quad (12.31)$$

Каждый член равенства (12.31) есть средняя интенсивность потока заявок в j -м узле, одинаковая также и для его элементов: очереди и обслуживающего прибора. Из (12.31) следует:

$$\bar{t}_j(N) = \frac{n_j(N)}{n_j^{обсл}(N)} \cdot \frac{1}{\mu_j}, \quad (12.32)$$

$$\bar{t}_j^{ож}(N) = \frac{n_j^{ож}(N)}{n_j^{обсл}(N)} \cdot \frac{1}{\mu_j}. \quad (12.33)$$

Подводя промежуточный итог, можно отметить основные этапы расчёта однородных экспоненциальных замкнутых сетей СМО в форме упрощенной методики.

Методика анализа однородных экспоненциальных замкнутых сетей СМО

1. Расчёт вероятностей ω_j , $j = \overline{1, M}$ посещения заявками узлов замкнутой сети по формуле (12.22).
2. Расчёт вероятностей состояний сети по формуле (12.23).
3. Расчёт среднего числа заявок в узлах, очередях и каналах узлов сети по известным вероятностям её состояний.
4. Расчёт средних времён пребывания заявок в узлах, очередях и каналах узлов сети по данным п.3 с использованием правила Литтла (формулы (12.32), (12.33)).

Пример 12.4

Задана одноканальная сеть с двумя узлами (рис. 12.7), число требований в сети $N = 1$. Определить для каждого из узлов коэффициент загрузки и среднее время пребывания в нём заявки.

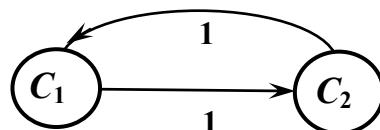


Рис. 12.7

Решение.

1. Расчет вероятностей ω_j : $\omega_1 = \omega_2 = 0,5$.
2. Расчет коэффициентов η_1 и η_2 загрузки узлов:

$$\eta_1 = 1 - \eta_2 = P(1,0) = \frac{\omega_1/\mu_1}{\omega_1/\mu_1 + \omega_2/\mu_2} = \frac{1/\mu_1}{1/\mu_1 + 1/\mu_2}.$$

3. Расчет среднего числа заявок в узле сети

$$\overline{n_1(1)} = 1 - \overline{n_2(1)} = P(1,0)$$

4. Расчет среднего времени пребывания заявки в узле сети:

$$\overline{t_1(1)} = \frac{\overline{n_1(1)}}{\overline{n_1^{обсл}}(1)} \cdot \frac{1}{\mu_1} = \frac{\overline{n_1(1)}}{\eta_1} \cdot \frac{1}{\mu_1} = \frac{P(1,0)}{\eta_1} \cdot \frac{1}{\mu_1},$$

$$\overline{t_2(1)} = \frac{\overline{n_2(1)}}{\overline{n_2^{обсл}}(1)} \cdot \frac{1}{\mu_2} = \frac{\overline{n_2(1)}}{\eta_2} \cdot \frac{1}{\mu_2} = \frac{P(0,1)}{1 - \eta_1} \cdot \frac{1}{\mu_2}.$$

Строго говоря, последовательность расчёта замкнутых сетей СМО, начиная с п.2 методики может быть изменена. Преобразования исходной формулы (12.23) позволяют вывести соотношения, обеспечивающие расчёт локальных характеристик узлов сети без предварительного определения вероятностей её состояний. Эти результаты, имеющие важное практическое значение, приводятся в разделе 12.2.5.

12.2.4. Расчёт нормирующей константы однородной замкнутой сети СМО

Расчёт нормирующей константы одноканальной замкнутой сети

В основу процедуры расчёта нормирующей константы положена приведенная выше формула (12.25):

$$G_M(N) = \sum_{n \in S(N,M)} \prod_{i=1}^M Z_i(n_i)$$

Для одноканальных сетей $Z_i(n_i) = \left(\frac{\omega_i}{\mu_i}\right)^{n_i} = x_i^{n_i}$ и выражение для

$G_r(k)$ можно представить в следующем виде:

$$\begin{aligned}
G_r(k) &= \sum_{n \in S(k,r)} \prod_{i=1}^r x_i^{n_i} = \sum_{\substack{n \in S(k,r) \\ n_r=0}} \prod_{i=1}^r x_i^{n_i} + \sum_{\substack{n \in S(k,r) \\ n_r>0}} \prod_{i=1}^r x_i^{n_i} = \\
&= G_{r-1}(k) + x_r G_r(k-1),
\end{aligned} \tag{12.34}$$

где

$$\begin{cases} G_1(k) = x_1^k, & k = \overline{0, N}, \\ G_r(0) = 1, & r = \overline{1, M}. \end{cases}$$

С учётом полученных соотношений можно дать следующее формальное изложение процедуры расчёта нормирующей константы:

Начальные значения параметров: $r = 2, k = 1$.

Шаг 1. $G_r(k) = G_{r-1}(k) + x_r G_r(k-1)$,

Если $k = N$, идти к Шагу 2,

Иначе $k = k + 1$, идти к Шагу 1.

Шаг 2. Если $r = M$, Конец,

Иначе $r = r + 1, k = \boxed{k} + 1$, идти к Шагу 1.

В этой процедуре помимо $G_M(N)$ вычисляются также константы $G_M(r), r = \overline{1, N-1}$, используемые при расчёте ряда характеристик замкнутой сети.

Пример 12.5

Рассмотрим пример использования процедуры расчёта нормирующей константы для задачи, изложенной в примере 12.2.

Шаг 1. $G_2(1) = G_1(1) + x_2 G_2(0) = x_1 + x_2$;

Шаг 2. $G_2(2) = G_1(2) + x_2 G_2(1) = x_1^2 + x_2(x_1 + x_2)$;

Шаг 3. $G_2(3) = G_1(3) + x_2 G_2(2) = x_1^3 + x_2 x_1^2 + x_2^2 x_1 + x_2^3$;

Шаг 4. $G_3(1) = G_2(1) + x_3 G_3(0) = x_1 + x_2 + x_3$;

Шаг 5. $G_3(2) = G_2(2) + x_3 G_3(1) = x_1^2 + x_2(x_1 + x_2) + x_3(x_1 + x_2 + x_3)$;

Шаг 6. $G_3(3) = G_2(3) + x_3 G_3(2) = x_1^3 + x_2 x_1^2 + x_2^2 x_1 + x_2^3 + x_3 x_1^2 + x_3 x_2(x_1 + x_2) + x_3^2(x_1 + x_2 + x_3)$.

Расчёт нормирующей константы многоканальной замкнутой сети

Для общего случая многоканальной замкнутой сети формула (12.25) может быть представлена в следующем виде:

$$\begin{aligned}
 G_M(N) &= \sum_{\substack{n \in S(N, M) \\ n_M = 0}} \prod_{i=1}^M Z_i(n_i) + \sum_{\substack{n \in S(N, M) \\ n_M = 1}} \prod_{i=1}^M Z_i(n_i) + \\
 &+ \sum_{\substack{n \in S(N, M) \\ n_M = 2}} \prod_{i=1}^M Z_i(n_i) + \sum_{\substack{n \in S(N, M) \\ n_M = 3}} \prod_{i=1}^M Z_i(n_i) + \dots + \sum_{\substack{n \in S(N, M) \\ n_M = N}} \prod_{i=1}^M Z_i(n_i) = \\
 &= \sum_{l=0}^N Z_M(l) \sum_{n \in S(N-l, M-1)} \prod_{i=1}^{M-1} Z_i(n_i) = \sum_{l=0}^N Z_M(l) G_{M-1}(N-l), \quad (12.35)
 \end{aligned}$$

где $Z_i(n_i) = \frac{\omega_i^{n_i}}{\prod_{j=1}^{n_i} \mu_i(j)}$, а $\mu_i(j) = \begin{cases} j \mu_i, & j < m_i \\ m_i \mu_i, & j \geq m_i \end{cases}$,

$$G_1(k) = Z_1(k) = \frac{\omega_1^k}{\prod_{j=1}^k \mu_1(j)}, \quad k = \overline{0, N},$$

$$\begin{aligned}
 G_r(0) &= 1, \quad r = \overline{1, M}, \\
 Z_i(0) &= 1, \quad i = \overline{1, M}.
 \end{aligned}$$

Тогда с учётом (12.35) получим следующее выражение:

$$G_r(k) = \sum_{l=0}^k Z_r(l) G_{r-1}(k-l). \quad (12.36)$$

Эта формула составляет основу приведенной ниже процедуры расчёта нормирующей константы многоканальной замкнутой сети.

Начальные значения параметров: $r = 2, k = 1$.

Шаг 1. $G_r(k) = \sum_{l=0}^k Z_r(l) G_{r-1}(k-l),$

Если $k = N$, идти к Шагу 2,
Иначе $k = k + 1$, идти к Шагу 1.

Шаг 2. Если $r = M$, Конец,

Иначе $r = r + 1$,  ~~$k + 1$~~ , идти к Шагу 1.

Здесь так же, как и в случае одноканальной сети справедливо замечание о полезности промежуточных результатов процедуры — значений констант $G_M(r)$, $r = \overline{1, N - 1}$ — для расчёта ряда характеристик многоканальных замкнутых сетей.

Отметим обстоятельство, которое может оказаться существенным для определения характеристик узлов замкнутой сети.

Процедура расчёта нормирующей константы предполагает последовательное включение узлов в состав сети. Нумерация узлов соответствует порядку их внедрения в сеть ($r = 1, 2, \dots, M$).

При этом указанный порядок, естественно, не имеет значения для результата расчёта нормирующей константы $G_M(N)$, но влияет на смысл и значение получаемых промежуточных констант $G_r(k)$.

С учётом сказанного принято выделять M -й узел сети, получивший название граничного узла.

Использование операции свёртки векторов для расчёта нормирующей константы

Пусть свёрткой двух $(R + 1)$ -мерных векторов

$$A = \{a(0), a(1), \dots, a(R)\} \quad \text{и} \\ B = \{b(0), b(1), \dots, b(R)\}$$

считается $(R + 1)$ -мерный вектор $C = \{c(0), c(1), \dots, c(R)\}$,

где $c(i) = \sum_{j=0}^i a(j) \cdot b(i-j)$. (12.37)

Будем обозначать операцию свёртки следующим образом:

$$C = A * B.$$

Введём в рассмотрение вектор $Z_i = \{Z_i(0), Z_i(1), \dots, Z_i(N)\}$,

где $Z_i(k) = \frac{\omega_i^k}{\prod_{j=1}^k \mu_i(j)}, \quad Z_i(k) = \frac{Z_i(k-1)\omega_i}{\mu_i(k)}, \quad Z_i(0) = 1.$

Введём в рассмотрение систему векторов $\mathbf{G}_0, \mathbf{G}_1, \dots, \mathbf{G}_M$ одинаковой размерности $(N+1) \times 1$.

Пусть $\mathbf{G}_0 = (1, 0, \dots, 0)^T$. В соответствии с (12.36) нормирующая константа $\mathbf{G}_r(k)$ есть не что иное, как $(k+1)$ -й компонент вектора \mathbf{G}_r , являющегося свёрткой $\mathbf{Z}_r * \mathbf{G}_{r-1}$ векторов \mathbf{Z}_r и \mathbf{G}_{r-1} :

где $\mathbf{G}_r = \mathbf{Z}_r * \mathbf{G}_{r-1}, \quad (12.38)$

$$\mathbf{G}_r(k) = \sum_{l=0}^k \mathbf{Z}_r(l) \mathbf{G}_{r-1}(k-l), \quad k = \overline{0, N}.$$

Если последовательно раскрывать \mathbf{G}_r в соответствии с (12.38), то в конечном итоге будет получено следующее соотношение:

$$\mathbf{G}_r = \mathbf{G}_0 * \mathbf{Z}_1 * \mathbf{Z}_2 * \mathbf{Z}_3 * \dots * \mathbf{Z}_i * \dots * \mathbf{Z}_{r-1} * \mathbf{Z}_r,$$

аналогично для \mathbf{G}_M :

$$\mathbf{G}_M = \mathbf{G}_0 * \mathbf{Z}_1 * \mathbf{Z}_2 * \mathbf{Z}_3 * \dots * \mathbf{Z}_i * \dots * \mathbf{Z}_{M-1} * \mathbf{Z}_M.$$

Отметим, что искомая нормирующая константа $\mathbf{G}_M(N)$ является $(N+1)$ -м компонентом вектора \mathbf{G}_M .

12.2.5. Расчётные соотношения, используемые для определения характеристик узлов однородной замкнутой сети СМО

Маргинальным называется распределение отдельного компонента системы случайных величин. Применительно к замкнутой сети СМО число n_j заявок в j -м узле является компонентом вектора $(n_1, \dots, n_j, \dots, n_M)$, задающего состояние всей сети в целом.

Используя формулу (12.23), можно вывести общее соотношение для вероятности $P_M(n, N)$ состояния граничного M -го узла, характеризуемого наличием n заявок при общем числе заявок в сети, равном N :

$$P_M(n, N) = \sum_{\substack{n \in S(N, M), \\ n_M = n}} \frac{1}{G_M(N)} \prod_{i=1}^M Z_i(n_i) =$$

$$= Z_M(n) \frac{\left\{ \sum_{n \in S(N-n, M-1)} \prod_{i=1}^{M-1} Z_i(n_i) \right\}}{G_M(N)} = Z_M(n) \frac{G_{M-1}(N-n)}{G_M(N)}. \quad (12.39)$$

*Маргинальное распределение
числа заявок в узле однородной замкнутой сети СМО*

Выражение (12.39) является ключевым для получения ряда эффективных формул расчёта характеристик замкнутых сетей. Эти формулы, вывод которых можно найти в [47], приводятся ниже.

Маргинальное распределение числа заявок в одноканальном узле однородной замкнутой сети СМО:

$$P_i(n, N) = \frac{x_i^n [G_M(N-n) - x_i G_M(N-n-1)]}{G_M(N)}. \quad (12.40)$$

Интенсивность на выходе i -го узла однородной замкнутой сети СМО:

$$\lambda_i(N) = \frac{\omega_i G_M(N-1)}{G_M(N)}. \quad (12.41)$$

Среднее число заявок в граничном узле однородной замкнутой сети СМО:

$$\bar{n}_M(N) = \frac{\sum_{n=1}^N n Z_M(n) G_{M-1}(N-n)}{G_M(N)}. \quad (12.42)$$

Среднее число заявок в одноканальном узле однородной замкнутой сети СМО:

$$\bar{n}_i(N) = \frac{\sum_{n=1}^N x_i^n G_M(N-n)}{G_M(N)}. \quad (12.43)$$

Среднее время пребывания заявки в одноканальном узле однородной замкнутой сети СМО:

$$\bar{t}_i(N) = \frac{\sum_{n=1}^N x_i^n G_M(N-n)}{\omega_i G_M(N-1)}. \quad (12.44)$$

12.2.6. Метод анализа средних значений характеристик узлов однородной замкнутой сети СМО

В предыдущих разделах рассмотрен традиционный подход к определению характеристик узлов замкнутых сетей, основанный на использовании теоремы Джексона и предполагающий обязательный расчёт нормирующей константы.

Последнее обстоятельство привносит комбинаторно возрастающую вычислительную сложность в анализ замкнутых сетей с увеличением числа их состояний.

Описываемый ниже итерационный метод не требует предварительного вычисления нормирующей константы, обеспечивая, тем не менее, точный расчёт важных характеристик узлов замкнутой сети СМО, таких как средние времена ожидания, длины очередей, коэффициенты загрузки узлов и т. д.

Одноканальные однородные замкнутые сети СМО

Рекуррентная процедура расчёта среднего времени пребывания заявки в узле одноканальной однородной замкнутой сети

Приводится итерационная процедура расчёта среднего времени $T_i(N)$ пребывания заявки в i -м узле, основанная на использовании формулы (12.43) в сочетании с правилом Литтла.

Начальные значения параметров: $r = 1$, $\bar{n}_i(0) = 0$, $i = \overline{1, M}$.

$$\text{Шаг 1. } \bar{t}_i(r) = \frac{1}{\mu_i} [1 + \bar{n}_i(r-1)], \quad i = \overline{1, M}$$

$$\text{Шаг 2. } \lambda_1(r) = \frac{r}{\left[\sum_{i=1}^M \frac{\omega_i \bar{t}_i(r)}{\omega_1} \right]}.$$

$$\text{Шаг 3. } \bar{n}_i(r) = \frac{\omega_i}{\omega_1} \lambda_1(r) \bar{t}_i(r), \quad i = \overline{1, M}.$$

Шаг 4. Если $r = N$, Конец,

Иначе, $r = r + 1$, идти к Шагу 1.

На выходе процедуры можно получить следующие характеристики сети.

1. Коэффициент загрузки $\rho_i(N)$ i -го узла:

$$\rho_i(N) = \frac{\lambda_i(N)}{\mu_i} = \frac{\bar{n}_i(N)}{\bar{t}_i(N)\mu_i}. \quad (12.45)$$

2. Вероятность $q_i(N)$ пребывания заявки в i -м узле сети:

$$q_i(N) = \frac{\omega_i \bar{t}_i(N)}{\sum_{j=1}^M \omega_j \bar{t}_j(N)}. \quad (12.46)$$

3. Среднее время $\bar{t}_i^{o\text{ж}}(N)$ ожидания в i -м узле сети:

$$\bar{t}_i^{o\text{ж}}(N) = \bar{t}_i(N) - \frac{1}{\mu_i}. \quad (12.47)$$

4. Среднее число $\bar{n}_i^{o\text{ж}}(N)$ заявок в очереди i -го узла сети:

$$\bar{n}_i^{o\text{ж}}(N) = \bar{n}_i(N) \frac{\bar{t}_i^{o\text{ж}}(N)}{\bar{t}_i(N)}. \quad (12.48)$$

5. Среднее время \bar{V}_i цикла для i -го узла сети:

$$\bar{V}_i = \frac{[N - \bar{n}_i(N)]}{\lambda_i(N)}. \quad (12.49)$$

Многоканальные однородные замкнутые сети СМО

Рекуррентная процедура вычисления $\bar{t}_i(N)$ и маргинального распределения числа заявок в узлах однородной замкнутой сети

Начальные значения параметров:

$$P_i(0, 0) = 1, \quad i = \overline{1, M}; \quad r = 1$$

$$\text{Шаг 1. } \bar{t}_i(r) = \sum_{n=1}^r \frac{n}{\mu_i(n)} P_i(n-1, r-1), \quad i = \overline{1, M} \quad (12.51)$$

$$\text{Шаг 2. } \lambda_1(r) = \frac{r}{\sum_{j=1}^M \frac{\omega_j}{\omega_1} \bar{t}_j(r)} \quad (12.52)$$

Шаг 3. Для каждого i -го узла сети, $i = \overline{1, M}$, определить набор вероятно-

стей $P_i(n, r)$, $n = \overline{0, r}$:

$$\begin{cases} P_i(n, r) = \frac{\omega_i \lambda_1(r)}{\omega_1 \mu_i(n)} P_i(n-1, r-1), & n = \overline{1, r}, \\ P_i(0, r) = 1 - \sum_{n=1}^r P_i(n, r), \end{cases} \quad (12.53)$$

где $\mu_i(n) = \begin{cases} n \mu_i, & n < m_i \\ m_i \mu_i, & n \geq m_i \end{cases}$, (12.54)

где m_i — число каналов в i -м узле.

Шаг 4.

Если $r = N$, Конец,
Иначе, $r = r + 1$, идти к Шагу 1.

На выходе процедуры формируются значения следующих характеристик:

$$\bar{t}_i(N), i = \overline{1, M}; \quad \{P_i(n, N)\}, i = \overline{1, M}; \quad n = \overline{0, N}.$$

Примечание: Вместо формулы (12.51) может быть использована следующая эквивалентная формула:

$$\bar{t}_i(r) = \frac{1}{\mu_i} \left[1 + \frac{1}{m_i} \sum_{n=m_i}^{r-1} (n - m_i + 1) P_j(n, r-1) \right], \quad i = \overline{1, M}. \quad (12.55)$$

Ряд существенных характеристик узлов многоканальной однородной замкнутой сети так же, как и ранее может быть получен с помощью формул (12.45) – (12.49).

12.3. НЕОДНОРОДНЫЕ СЕТИ СМО

12.3.1. Общие сведения о неоднородных сетях СМО

Рассматриваются разомкнутые линейные экспоненциальные неоднородные сети СМО. Неоднородность сети обусловлена принадлежностью заявок различным классам, определяющим разную маршрутизацию и обслуживание в узлах.

Состояние неоднородной сети задается матрицей S :

$$S = \begin{Bmatrix} \rightarrow \\ \overrightarrow{\mathbf{n}_1} \\ \rightarrow \\ \overrightarrow{\mathbf{n}_2} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \rightarrow \\ \overrightarrow{\mathbf{n}_M} \end{Bmatrix} = \begin{Bmatrix} \mathbf{n}_{11}, \mathbf{n}_{12}, \dots, \mathbf{n}_{1V} \\ \mathbf{n}_{21}, \mathbf{n}_{22}, \dots, \mathbf{n}_{2V} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \mathbf{n}_{M1}, \mathbf{n}_{M2}, \dots, \mathbf{n}_{MV} \end{Bmatrix}, \quad (12.56)$$

где $\vec{\mathbf{n}}_i = (\mathbf{n}_{i1}, \mathbf{n}_{i2}, \dots, \mathbf{n}_{iV})$ — вектор состояния i -го узла сети,

M — число узлов сети, V — число классов заявок.

Основная часть представленных ниже аналитических решений для разомкнутых и замкнутых сетей справедлива в том случае, если узлы сети принадлежат одному из четырех типов [31, 47].

Узел типа 1. Дисциплина обслуживания FIFO (в порядке поступления). Длительность обслуживания заявок всех классов в i -м узле имеет одно и то же экспоненциальное распределение с интенсивностью $\mu_i(\mathbf{n}_i)$, зависящей от числа \mathbf{n}_i заявок в узле.

Узел типа 2. Обслуживание заявок в одноканальном узле осуществляется в соответствии с дисциплиной PS¹. Длительность обслуживания распределена по закону Кокса [31, 47]².

Узел типа 3. Многоканальный узел с бесконечным числом каналов и дисциплиной IS (без ожидания). Длительность обслуживания задается так же, как и для узла типа 2.

Узел типа 4. Одноканальный узел с дисциплиной обслуживания LIFO (в порядке, обратном поступлению) и с прерыванием обслуживания. Длительность обслуживания задается так же, как и для узла типа 2.

12.3.2. Разомкнутые неоднородные сети СМО

Разомкнутые сети массового обслуживания характеризуются наличием внешнего источника требований. Требования (заявки) поступают в сеть, обслуживаются в узлах сети и покидают ее. В неоднородной сети

¹ Дисциплина Processor Sharing [31] предполагает предоставление бесконечно малого кванта времени обслуживания каждой заявке. Не успевшие обслужиться заявки становятся на дообслуживание в хвост очереди.

² Показательный закон является частным случаем закона Кокса.

имеет место несколько классов заявок, различающихся маршрутизацией и обслуживанием в узлах.

Рассмотрим общий класс разомкнутых неоднородных сетей СМО: заявки меняют класс при переходе из узла в узел. Это означает, что матрица передач π в такой сети приобретает следующую форму:

$$\pi = \{p_{ik,jv}\} \quad i, j = \overline{0, M}; \quad k, v = \overline{1, V}, \quad (12.57)$$

где M — число узлов сети, V — число классов заявок, $p_{ik,jv}$ — вероятность того, что заявка k -го класса, завершившая обслуживание в i -м узле, поступит в j -й узел и станет заявкой v -го класса.

Модель входного источника для неоднородной разомкнутой сети может быть задана следующим образом:

$\Lambda_0 = \{\lambda_{0i}\}$, $i = \overline{1, V}$, где λ_{0i} — интенсивность внешнего источника заявок i -го класса.

Если сеть линейная, то для каждого узла справедливо следующее уравнение:

$$\lambda_{iv} = \sum_{i=1}^M \sum_{k=1}^V \lambda_{ik} p_{ik,jv}, \quad j = \overline{1, M}, \quad k, v = \overline{1, V}. \quad (12.58)$$

Источник заявок удобно представить дополнительным (**0**-м) узлом сети, тогда, замкнув сеть через источник, можно считать каждый узел взаимоизменяющимся.

В этом случае, учитывая, что $\sum_{(jv)} p_{ik,jv} = 1$, то есть сумма элементов

в каждой строке с номером ik матрицы π равна 1, можно утверждать, что существует единственное решение системы (12.58):

$$\lambda_{jv} = \alpha_{jv} \lambda_{0r}, \quad j = \overline{1, M}, \quad v = \overline{1, V}; \quad (12.59)$$

где λ_{0v} — средняя интенсивность внешнего источника v -го класса заявок, α_{jv} — коэффициент передачи от источника требования v -го класса к j -му узлу.

Если заявки не меняют класса, то каждому из них соответствует своя матрица передач $\pi^{(v)}$:

$$\pi^{(v)} = \{p_{iv,jv}\} = \{p_{ij}^{(v)}\}, \quad i, j = \overline{0, M}; \quad v = \overline{1, V}, \quad \text{а система уравнений}$$

(12.57) приобретает следующий вид:

$$\lambda_{jv} = \sum_{i=0}^M \lambda_{iv} p_{ij}^{(v)}, \quad j = \overline{1, M}, \quad v = \overline{1, V}. \quad (12.60)$$

Условие существования установившегося режима состоит в том, что суммарный коэффициент загрузки R_j каждого узла должен быть меньше единицы:

$$R_j = \sum_{v=1}^V \frac{\alpha_{jv} \lambda_{0v}}{\mu_{jv}} < 1, \quad j = \overline{1, M}, \quad (12.61)$$

где μ_{jv} — средняя интенсивность обслуживания заявок v -го класса в j -м узле.

Сделаем важный вывод. В разомкнутых линейных экспоненциальных сетях массового обслуживания в установившемся режиме интенсивности потоков в узлах зависят только от матрицы передач и входных потоков, и их значения определяются системой (12.59). Данное решение справедливо в условиях (12.61).

Распределение числа заявок в сети СМО без разделения на классы

Рассмотрим задачу определения вероятностей агрегированных состояний неоднородных разомкнутых сетей СМО. В этом случае состояние S сети задается вектором: (n_1, n_2, \dots, n_M) , а

выражение для стационарного распределения состояний сети принимает следующий вид:

$$P(S) = P_1(n_1)P_2(n_2) \dots P_M(n_M), \quad \text{где} \quad (12.62)$$

$$P_i(n_i) = \begin{cases} (1 - R_i) R_i^{n_i} e^{-R_i} & \text{для узлов типов 1, 2, 4;} \\ \frac{R_i^{n_i} e^{-R_i}}{n_i!} & \text{для узлов типа 3.} \end{cases} \quad (12.63)$$

Отметим, что для узла сети, принадлежащего типу 1, (12.63) предполагает наличие только одноканальных узлов с однородным обслуживанием заявок разных классов ($\mu_{ir} = \mu_i$).

Распространение метода средних значений на случай неоднородных одноканальных разомкнутых сетей СМО

Метод средних значений применительно к одноканальной СМО типа **M/G/1** используется для определения среднего времени ожидания в очереди

ди и рассмотрен в разделе 11.

Полученные результаты, и в частности, формула (11.25) могут быть использованы для расчета локальных временных показателей неоднородных одноканальных разомкнутых сетей СМО [37].

Исходные предпосылки расчёта

1. На вход сети поступает неоднородный пуссоновский поток заявок.
2. Все узлы сети являются СМО типа $M/M/1$ с неоднородным потоком заявок на входе.
3. Возможные дисциплины обслуживания заявок в узле:
 - бесприоритетная дисциплина FIFO,
 - введение относительных приоритетов,
 - введение абсолютных приоритетов.
4. Параметры распределения длительности обслуживания в i -м узле: μ_{iv} — интенсивность обслуживания заявки v -го класса в i -м узле.

Формализация дисциплины обслуживания заявок разных классов в i -м узле сети задается матрицей-диспетчером $D_i = \left\{ d_{vr}^{(i)} \right\}, v, r = \overline{1, V}$.

Каждый элемент $d_{vr}^{(i)}$ матрицы D_i определяет дисциплину обслуживания при конфликте поступающей в узел заявки v -го класса с находящейся в узле заявкой r -го класса. Таким образом, квадратная матрица D_i имеет число строк, равное числу классов заявок, посещающих узел i . Элемент $d_{vr}^{(i)}$ в нашем рассмотрении может принимать следующие значения:

- A , если заявки класса v имеют абсолютный приоритет по отношению к заявкам класса r ,
- \bar{A} — при обратном назначении приоритетов,
- R , если заявки класса v имеют относительный приоритет по отношению к заявкам класса r ,
- \bar{R} — при обратном назначении приоритетов,
- F , если заявки классов v и r обслуживаются в порядке поступления.

Разумно предположить, что заявки одного класса обслуживаются в порядке поступления. Тогда отсюда следует свойство симметрии главной диагонали матрицы D_i : $d_{rr}^{(i)} = F, r = \overline{1, V}$. Можно выделить и некоторые

свойства элементов, симметричных относительно главной диагонали. Так, например, если

$$d_{vr}^{(i)} = A \text{ или } d_{vr}^{(i)} = \bar{A}, \text{ то } d_{rv}^{(i)} = \bar{A} \text{ или } d_{rv}^{(i)} = A.$$

То же самое относится и к дисциплине с относительными приоритетами. Если $d_{vr}^{(i)} = R$ или $d_{vr}^{(i)} = \bar{R}$, то $d_{rv}^{(i)} = \bar{R}$ или $d_{rv}^{(i)} = R$.

Полная симметрия соблюдается для дисциплины FIFO.

$$\text{Если } d_{vr}^{(i)} = F, \text{ то } d_{rv}^{(i)} = F.$$

Заметим, что одна v -я строка матрицы D_i полностью определяет порядок обслуживания заявок v -го класса в i -м узле по отношению к заявкам других классов. Ниже приводится пример матрицы диспетчера.

A означает абсолютный приоритет,

R — относительный приоритет,

F — бесприоритетное обслуживание,

\bar{A} и \bar{R} — наличие более приоритетных потоков заявок, чем данный с абсолютным и относительным приоритетом:

$$D = \begin{matrix} & \begin{matrix} 1 & 2 & 3 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \end{matrix} & \left[\begin{matrix} F & A & A \\ \bar{A} & F & R \\ \bar{A} & \bar{R} & F \end{matrix} \right] \end{matrix}.$$

С учетом сказанного разомкнутая неоднородная одноканальная сеть, удовлетворяющая условиям стационарности и прошедшая анализ интенсивностей потоков в узлах в соответствии с (12.59) или (12.60), может быть задана следующим набором характеристик:

$$(M, V, \{\lambda_{iv}\}, \{\mu_{iv}\}, \{D_i\}), i = \overline{1, M}, v = \overline{1, V}. \quad (12.64)$$

Использование набора формул (4.25) для сети, представленной в форме (12.64), позволяет определить среднее время \bar{t}_{iv} пребывания заявки v -го класса в i -м узле как сумму среднего времени ожидания в очереди и среднего времени обслуживания.

$$\bar{t}_{iv} = \bar{t}_{ож_{iv}} + \bar{x}_{iv} \quad (12.65)$$

Приведенное ниже выражение определяет алгоритм формирования формулы расчета \bar{t}_{iv} с учетом содержания матрицы диспетчера D_i i -го узла.

В формуле (12.66) символ суммирования \sum_X с некоторым индексом X означает суммирование только таких значений r , для которых элемент d_{vr} матрицы-диспетчера D_i равен X : $d_{vr} = X$.

$$\begin{aligned} \overline{t_{iv}} &= \frac{1}{\mu_{iv}} \left[\sum_{(A)} \frac{\lambda_{ir}}{\mu_{ir}} \right] \left[1 - \sum_{(A)} \frac{\lambda_{ir}}{\mu_{ir}} \right]^{-1} + \\ &+ \left[\sum_{(F, \bar{A}, R, \bar{R})} \frac{\lambda_{ir}}{\mu_{ir}^2} \right] \left[1 - \sum_{(F, \bar{R}, \bar{A})} \frac{\lambda_{ir}}{\mu_{ir}} \right]^{-1} \left[1 - \sum_{(\bar{R}, A)} \frac{\lambda_{ir}}{\mu_{ir}} \right]^{-1} + \frac{1}{\mu_{iv}}. \end{aligned} \quad (12.66)$$

Поясним кратко конструкцию формулы (12.66). Первое слагаемое определяет время ожидания заявки v -го класса в связи с поступлением заявок с большим абсолютным приоритетом на интервале её пребывания в i -м узле. Второе слагаемое формулы — оставшаяся часть времени ожидания заявки v -го класса в i -м узле. Третье слагаемое формулы — среднее время обслуживания заявки v -го класса в i -м узле.

12.3.3. Замкнутые неоднородные сети СМО

Замкнутые сети массового обслуживания характеризуются постоянным числом заявок. В неоднородной замкнутой сети циркулируют различающиеся маршрутизацией и/или обслуживанием в узлах заявки нескольких классов с постоянным числом N_v , ($v = \overline{1, V}$) в каждом из них.

Неоднородная замкнутая экспоненциальная сеть может быть задана следующим набором параметров:

$$\left(\mathbf{M}, \vec{\mathbf{N}}, \left\{ \pi^{(v)} \right\}, \left\{ \mu_{iv} \right\}, \vec{\mathbf{m}}, \mathbf{D} \right), \quad (12.67)$$

где \mathbf{M} — число узлов сети,

$\vec{\mathbf{N}} = (N_1, N_2, \dots, N_v, \dots, N_V)$ — вектор распределения общего числа заявок сети по классам, $v = \overline{1, V}$;

$\left\{ \pi^{(v)} \right\}$, $v = \overline{1, V}$ — набор стохастических матриц передач, в котором

$\pi^{(v)} = \left\{ p_{ij}^{(v)} \right\}$, $i, j = \overline{1, M}$ — матрица передач заявок v -го класса;

$\{\mu_{iv}\}, i = \overline{1, M}, v = \overline{1, V}$; — множество, задающее интенсивности обслуживания неоднородных заявок в узлах сети, в котором элемент множества μ_{iv} есть интенсивность обслуживания заявки v -го класса в канале i -го узла сети;

$\vec{m} = \{m_1, \dots, m_M\}$ — вектор, i -м компонентом которого является число каналов в i -м узле сети;

D — матрица диспетчера, определяющая порядок обслуживания заявок в узлах.

Мощность $|S|$ множества $S(\vec{N}, M)$ состояний неоднородной замкнутой сети определяется по правилам комбинаторики как число вариантов размещения заявок сети по узлам с учетом их принадлежности к разным классам:

$$|S| = \prod_{v=1}^V C_{N_v + M - 1}^{N_v}. \quad (12.68)$$

Замкнутые сети с неоднородными по маршрутам циркуляции заявками

Пусть в замкнутой экспоненциальной сети с числом узлов M циркулируют заявки V классов. Число заявок каждого класса v равно $N_v, v = 1, 2, \dots, V$, а маршруты циркуляции заявок v -го класса задаются матрицей передач $\pi^{(v)} = \{p_{ij}^{(v)}\}, v = 1, 2, \dots, V$. Обслуживание всех заявок пусть остается однородным в том смысле, что интенсивность обслуживания в каналах узлов не зависит от класса заявки.

В отношении дисциплин обслуживания оставим те, которые соответствуют четырем типам узлов, введенным в разделе 12.3.1.

Тогда оказывается, что рассмотренный выше алгоритм расчета однородных замкнутых сетей может быть обобщен и на случай сетевых моделей с несколькими классами заявок, отличающимися маршрутами циркуляции. При этом, правда, значительно возрастают сложность и объем вычислений. Для иллюстрации сказанного приведем основные расчетные соотношения.

Аналогично тому, как это сделано для однородных сетей, введем относительные частоты ω_{iv} , удовлетворяющие системам уравнений

$$\omega_{jv} = \sum_{i=1}^M \omega_{iv} p_{jv}^{(v)}, \quad i, j = \overline{1, M}; v = \overline{1, V}. \quad (12.69)$$

Состояние i -го узла сети будем задавать вектором $\vec{n}_i = \left\{ n_{iv} \right\}_{v=1, \overline{V}}$,

тогда состояние сети определяется матрицей $\left\{ n_{iv} \right\}_{i=\overline{1, M}; v=\overline{1, V}}$.

Возможными состояниями сети будут такие, для которых

$$\sum_{i=1}^M n_{iv} = N_v, \quad v = \overline{1, V},$$

где $n_{iv} \geq 0, i = \overline{1, M}; v = \overline{1, V}$.

Вероятность $P \left(\left\{ n_{iv} \right\} \right)$ состояния сети $\left\{ n_{iv} \right\}$ имеет мультипликативную форму:

$$P \left(\left\{ n_{iv} \right\} \right) = \frac{1}{G} \prod_{i=1}^M Z_i \left(n_{i1}, n_{i2}, \dots, n_{iV} \right) = \frac{1}{G} \prod_{i=1}^M Z_i \left(\vec{n}_i \right) \quad (12.70)$$

где в зависимости от типа узла i выделим три варианта вычисления сомножителей $Z_i \left(\vec{n}_i \right)$:

$$1) \quad Z_i \left(\vec{n}_i \right) = \frac{n_i!}{\prod_{k=1}^{n_i} \mu_i(k)} \prod_{v=1}^V \frac{\omega_{iv}^{n_{iv}}}{n_{iv}!} \text{ для многоканальных узлов типа 1; } \quad (12.71)$$

$$2) \quad Z_i \left(\vec{n}_i \right) = n_i! \prod_{v=1}^V \frac{\left(\frac{\omega_{iv}}{\mu_{iv}} \right)^{n_{iv}}}{n_{iv}!} \text{ для одноканальных узлов типов 1,2,4; } \quad (12.72)$$

$$3) \quad Z_i \left(\vec{n}_i \right) = \prod_{v=1}^V \frac{1}{n_{iv}!} \left(\frac{\omega_{iv}}{\mu_{iv}} \right)^{n_{iv}} \text{ для узлов типа 3; } \quad (12.73)$$

в приведенных соотношениях

$$n_i = \sum_{v=1}^V n_{iv}, \quad i = \overline{1, M}$$

обозначает общее число заявок всех классов в узле i , G — нормирующая константа. Отметим, что $Z_i \left(0, 0, \dots, 0 \right) \equiv 1$.

Таким образом, так же как и в случае однородных замкнутых сетей возможность и эффективность точного анализа будет зависеть от наличия и эффективности алгоритма вычисления нормирующей константы \mathbf{G} . Покажем возможность использования так называемого алгоритма «свертки» для вычисления нормирующей константы.

*Использование операции свертки множеств
для расчета нормирующей константы
в случае замкнутых неоднородных сетей СМО*

Пусть теперь Z_i есть множество значений функции $Z_i(n_{i1}, n_{i2}, \dots, n_{iV})$, определенных в пространстве размерности V , причем v -я координата n_{iv} может принимать значения $0, 1, 2, \dots, N_v$. Мощность множества Z_i определяется вектором

$$\vec{N} = (N_1, N_2, \dots, N_v, \dots, N_V) : |Z_i| = \prod_{v=1}^V (N_v + 1).$$

Для двух множеств A и B со структурой, аналогичной структуре множества Z_i , определим свертку $C = A * B$ как множество C такой же структуры, элементы которого вычисляются по формуле

$$c(n_1, \dots, n_V) = \sum_{m_V=0}^{m_V=n_V} \cdots \sum_{m_1=0}^{m_1=n_1} a(m_1, \dots, m_V) \cdot b(n_1 - m_1, \dots, n_V - m_V). \quad (12.74)$$

Введем M множеств $G_i, i = \overline{1, M}$, связанных соотношением $G_i = G_0 * Z_1 * Z_2 * \dots * Z_i$ или, иначе, $G_i = G_{i-1} * Z_i$, где G_0 — множество, содержащее все нулевые элементы кроме $G_0(0, 0, \dots, 0) = 1$. Нормирующая константа \mathbf{G} определяется аналогично рассмотренной ранее в разделе 12.2.4 константе $G_M(N)$ и равна $G_M(N_1, N_2, \dots, N_V)$.

Отметим, что порядок включения множеств Z_i в свертку, приводящую к расчету $G_M(N_1, N_2, \dots, N_V)$, может быть произвольным.

*Расчет некоторых характеристик
замкнутых неоднородных сетей СМО*

Так же, как и в случае однородных сетей, теперь можно вывести ряд важных расчетных соотношений. Например, вероятность нахождения

в узле номер M ровно n_v заявок каждого класса v , $v=1,2,\dots,V$, равна

$$P_M(n_1, \dots, n_V) = Z_M(n_1, \dots, n_V) \frac{G_{M-1}(N_1 - n_1, \dots, N_V - n_v)}{G}, \quad (12.75)$$

интенсивность λ_{Mv} прохождения заявок класса v через узел с номером M :

$$\lambda_{Mv} = \omega_{Mv} \frac{G_M(N_1, N_2, \dots, N_{v-1}, N_v - 1, N_{v+1}, \dots, N_V)}{G}, \quad (12.76)$$

среднее число заявок класса v в M -м узле типа 1, 2 или 4 равно

$$\overline{n}_{Mv} = \frac{1}{G} \sum_{n_v=0}^{N_v} \cdots \sum_{n_1=0}^{N_1} n_v Z_M(n_1, \dots, n_V) G_{M-1}(N_1 - n_1, \dots, N_V - n_v), \quad (12.77)$$

а для узла типа 4 можно воспользоваться формулой Литтла:

$$\overline{n}_{Mv} = \lambda_{Mv} / \mu_i.$$

Среднее время пребывания заявок класса v в M -м узле также может быть определено с помощью формулы Литтла:

$$\overline{t}_{Mv} = \overline{n}_{Mv} / \lambda_{Mv}.$$

Аналогичные характеристики для любого другого узла могут быть получены после перенумерации узлов. Вывод приведенных соотношений можно найти, например, в [47].

Таким образом, усложнение процедур численного анализа сетевых моделей с несколькими классами заявок, отличающимися маршрутами циркуляции, происходит главным образом за счет необходимости оперировать множествами, определенными на многомерном пространстве.

Расчёт средних характеристик замкнутых неоднородных одноканальных сетей СМО

Пусть в замкнутой экспоненциальной модели циркулируют заявки V классов, которые могут отличаться не только маршрутами прохождения по сети, но и характеристиками их в обслуживания в узлах сети: интенсивностью и дисциплиной обслуживания. Причем назначение дисциплин (приоритетов между классами) может иметь произвольный характер в каждом отдельном узле. Точный анализ подобных моделей довольно затруднителен. В данном разделе предлагаются расчетные соотношения для при-

ближенного анализа, который, по существу, является обобщением метода средних значений на случай неоднородных замкнутых одноканальных сетей СМО.

Зададим сеть следующим набором характеристик:

$$\left(M, \vec{N}, \Omega = \left\{ \omega_{iv} \right\}, \left\{ \mu_{iv} \right\}, D \right), \quad (12.78)$$

где M — число узлов сети, $i = \overline{1, M}$,

$\vec{N} = (N_1, N_2, \dots, N_v, \dots, N_V)$ — вектор распределения заявок сети по классам, $v = \overline{1, V}$;

$\Omega = \left\{ \omega_{iv} \right\}_{\substack{i=1, M \\ v=1, V}}$ — матрица частот прохождения заявок v -го класса через

i -й узел, определяемая через вероятности переходов:

$$\omega_{iv} = \sum_{j=1}^M \omega_{jv} p_{jv, iv}, \quad (12.79)$$

$D_i = \left\{ d_{vr}^{(i)} \right\}$, $v, r \in \overline{1, 2, \dots, V}$ — матрица-диспетчер, задающая дисциплины обслуживания заявок разных классов в i -м узле.

Эта матрица, введенная ранее для разомкнутых неоднородных сетей, имеет тот же смысл, но может быть дополнена ещё одной дисциплиной, задающей обслуживание в порядке, обратном поступлению (LIFO). Для дисциплины LIFO элемент $d_{vr}^{(i)} = d_{rv}^{(i)}$ получает обозначение L .

Среднее время пребывания заявок v -го класса в i -м узле складывается из среднего времени ожидания в очереди $\overline{t_{ож_{iv}}}$ и среднего времени обслуживания:

$$\overline{t_{iv}} = \overline{t_{ож_{iv}}} + \frac{1}{\mu_{iv}}. \quad (12.80)$$

Среднее время ожидания заявки v -го класса в i -м узле можно представить как функцию времен $\overline{t_{ir}}$ пребывания заявок всех имеющихся классов во всех узлах сети: $i = \overline{1, M}$, $r = \overline{1, V}$. Предварительно обозначим через q_{iv} вероятность пребывания заявок v -го класса в i -м узле

$$q_{iv} = \frac{\omega_{iv} \bar{t}_{iv}}{\sum_{j=1}^M \omega_{jv} \bar{t}_{jv}}, \quad (12.81)$$

а через λ_{iv} — интенсивность прохождения заявок v -го класса через i -й узел, которую в дальнейшем будем вычислять по формуле Литтла:

$$\lambda_{iv} = \frac{N_v q_{iv}}{y_{iv}}, \quad i = \overline{1, M}, \quad v = \overline{1, V}. \quad (12.82)$$

Теперь перепишем (12.71), подробно представляя каждую составляющую среднего времени ожидания:

$$\begin{aligned} \bar{t}_{iv} = & \left[\sum_{F, A, R, r \neq v} \frac{N_r q_{ir}}{\mu_{ir}} \right] + \frac{[N_v - 1] q_{iv}}{\mu_{iv}} + \left[\sum_{R, L} \frac{\lambda_{ir}}{\mu_{ir}^2} \right] + \\ & + \left[\bar{t}_{iv} \sum_A \frac{\lambda_{ir}}{\mu_{ir}} \right] + \left(\left[\bar{t}_{iv} - \frac{1}{\mu_{iv}} \right] \sum_{R, L} \frac{\lambda_{ir}}{\mu_{ir}} \right) + \frac{1}{\mu_{iv}}, \quad i = \overline{1, M}, \quad v = \overline{1, V}, \end{aligned} \quad (12.83)$$

где символ суммирования \sum_X с нижним индексом X обозначает наличие

только таких слагаемых в сумме, которые удовлетворяют условию $d_{vr} = X$.

Поясним смысл каждого слагаемого. Первое слагаемое отображает долю времени ожидания, обусловленную обслуживанием заявок, уже находившихся в i -м узле к моменту прихода туда заявки v -го класса, и перед которыми заявка v -го класса не имеет приоритета. Второе слагаемое соответствует времени ожидания обслуживания заявок «своего же» v -го класса. Третье — времени ожидания дообслуживания заявки менее приоритетной, чем v -я, но уже находившейся на обслуживании. Четвертое слагаемое учитывает возможность прихода в i -й узел заявок, имеющих абсолютный приоритет по отношению к v -му классу, за время нахождения там заявки v -го класса. Пятое слагаемое соответствует времени обслуживания заявок, имеющих относительный приоритет по отношению к v -му классу и пришедших в i -й узел за время нахождения там заявки v -го класса в очереди.

Подставив теперь выражения (12.81) в (12.82), а затем (12.81) и (12.82) в (12.83), получим систему $M \cdot V$ нелинейных алгебраических уравнений вида

$$\overline{t_{iv}} = f_{iv} \left(\left\{ \overline{t_{ir}} \right\}_{\substack{i=1,M \\ r=1,V}} \right) + \frac{1}{\mu_{iv}}, \quad i = \overline{1, M}; \quad v = \overline{1, V}, \quad (12.84)$$

где символ $f_{iv}(\cdot)$ используется для обозначения функциональной зависимости $\overline{t_{iv}}$ от совокупности значений $\left\{ \overline{t_{ir}} \right\}_{\substack{i=1,M \\ r=1,V}}$ в правой части каждого уравнения системы.

Можно доказать существование решения системы уравнений (12.84) и возможность ее решения итерационным способом, например методом простых итераций. При этом имеет значение выбор начальных приближений. Установлено, что всегда будет иметь место сходимость «сверху», хотя для большинства случаев наблюдается хорошая сходимость и «снизу», и «сверху». Таким образом, для обеспечения сходимости целесообразно выбрать начальные значения $\overline{t_{iv}}$ относительно большими, например, $\overline{t_{iv}} = h/\mu_{iv}$, где коэффициент $h \gg 1$ и выбирается исходя из общего числа заявок в сети, разброса значений μ_{iv} и т. п.

Решение системы уравнений (12.84) и дальнейшее использование соотношений (12.81), (12.82) дает приближенную оценку параметров функционирования сетевой модели.

Погрешность оценки объясняется не только погрешностями численных методов, которые обычно приходится применять для решения системы (12.84), но и методической неточностью соотношений (12.83), в которых для простоты и в правой, и в левой частях уравнений используются параметры, соответствующие одному и тому же числу заявок в замкнутой сети.

Строго говоря, в правые части этих уравнений необходимо подставлять значения q_{jr} , λ_{jr} , $j = \overline{1, M}$, $r = \overline{1, V}$, соответствующие сетям с числом заявок v -го класса, меньшим на единицу. Учет этого обстоятельства приводит к сложной рекуррентной процедуре. В то же время погрешности оценок, получаемых путем решения системы уравнений (12.84), составляют единицы процентов и уменьшаются с ростом числа заявок в сети. Исходя из всего сказанного, можно сделать вывод, что для многих практических случаев вполне применим предложенный выше приближенный метод решения системы (12.84).

Рассмотрим, тем не менее, содержание рекуррентной процедуры расчета времени $\overline{t_{iv}}$, свободной от упомянутой методической неточности.

Для этого введем следующие обозначения:

$$N = \sum_{v=1}^V N_v — суммарное число заявок в сети,$$

$$\mathbf{1}_v = \begin{pmatrix} 0, 0, \dots, 1_v, \dots, 0 \end{pmatrix} — вектор, v-й компонент которого равен 1, а$$

остальные принимают нулевые значения.

Рекуррентный способ использования зависимости (12.84) предполагает последовательное увеличение числа n заявок в сети от 1 до N , сопровождаемое расчетом набора времен $\left\{ \overline{t}_{iv} \right\}_{\substack{i=1,M \\ v=1,V}}$ для каждого s -го варианта распределения n заявок по V классам из числа возможных.

Указанное распределение задается множеством $K(n)$ векторов $\vec{K}_s(n)$: $K(n) = \left\{ \vec{K}_s(n) = (K_{s1}(n), \dots, K_{sV}(n)) \right\}$,

где $K_{sv}(n)$ — число заявок, помещенных для s -го варианта в v -й класс из общего числа n .

Принимая во внимание очевидный факт, состоящий в том, что заявка v -го класса, поступающая в любой узел сети, не может войти в конфликт сама с собой и, следовательно, «имеет дело» с заявками, распределенными по сети в соответствии с вектором $\vec{K} - \mathbf{1}_v$, систему (12.84) можно представить так:

$$\overline{t}_{iv}(\vec{K}) = f_{iv} \left(\overline{t}_{iv}(\vec{K} - \mathbf{1}_v), \left\{ q_{ir}(\vec{K} - \mathbf{1}_v), \lambda_{ir}(\vec{K} - \mathbf{1}_v) \right\}_{\substack{i=1,M \\ r=1,V}} \right). \quad (12.85)$$

Используя прямую зависимость значений q_{ir} и λ_{ir} от множества значений $\left\{ \overline{t}_{ir} \right\}$, $i = \overline{1, M}; r = \overline{1, V}$, демонстрируемую (12.81) и (12.82), можем записать

$$\overline{t}_{iv}(\vec{K}) = f_{iv} \left(\left\{ \overline{t}_{ir}(\vec{K} - \mathbf{1}_v) \right\}_{\substack{i=1,M \\ r=1,V}} \right) \quad (12.86)$$

Процедура расчёта времени $\overline{t}_{iv}(\vec{K})$

С учетом сказанного рекуррентная процедура может быть представлена в следующей форме:

Шаг 1.

$n = 1$,

$$\overline{t}_{iv}(1_v) = \frac{1}{\mu_{iv}}, \quad i = \overline{1, M}; \quad v = \overline{1, V}$$

Шаг 2.

Если $n = N$, Конец,
Иначе, $n = n + 1$, Идти к Шагу 3

Шаг 3.

Найти множество $K(n) = \left\{ \vec{K}_s(n) = (K_{s1}(n), \dots, K_{sV}(n)) \right\}$ векторов, задающих варианты распределения n заявок по V классам с учетом ограничений, налагаемых вектором $\vec{N} = (N_1, N_2, \dots, N_v, \dots, N_V) : K_{sv}(n) \leq N_v$.

Для каждого вектора $\vec{K}_s(n)$ найти

$$\overline{t}_{iv}(\vec{K}_s(n)) = f_{iv} \left(\left\{ \overline{t}_{ir}(\vec{K}_s(n) - 1_v) \right\}_{\substack{i=1, M \\ r=1, V}} \right).$$

Идти к Шагу 2.

Метод средних значений для неэкспоненциальных замкнутых неоднородных одноканальных сетей СМО

Рассмотрим одноканальную сеть СМО, в узлах которой время обслуживания распределено по произвольному закону с математическим ожиданием \overline{X}_{ir} и вторым начальным моментом \overline{X}_{ir}^2 .

Среднее время \overline{t}_{iv} пребывания заявки v -го класса в узле данной сети можно представить в виде суммы следующих трех составляющих:

- а) времени обслуживания заявок уже находящихся в очереди,
- б) собственного времени обслуживания и
- в) времени дообслуживания той заявки, которая на момент поступ-

ления пришедшей занимала обслуживающий прибор.

«Неэкспоненциальность» сети проявляется в последнем компоненте, зависящем от закона распределения времени обслуживания.

Формула расчета t_{iv} в рассматриваемой сети с дисциплиной обслуживания FIFO, учитывая указанные составляющие, представлена в [64]. Следуя [47], можно получить аналогичную формулу для той же сети, допускающей ряд других дисциплин обслуживания.

Пусть сеть задана следующим набором характеристик:

$$\left(M, \vec{N}, \Omega = \{\omega_{ir}\}, \left\{ \overline{x}_{ir}, \overline{x_{ir}^2} \right\}, D \right), \text{ тогда}$$

$$\begin{aligned} \overline{t_{iv}} &= \sum_{F,A,R,r \neq v} \overline{X_{ir}} \left[N_r q_{ir} - \overline{X_{ir}} \right] + \overline{X_{iv}} \left[(N_v - 1) q_{iv} - \overline{X_{iv}} \right] + \\ &+ \frac{1}{2} \left[\sum_{R,F,A,\bar{A},L} \lambda_{ir} \overline{X_{ir}^2} \right] + \left[\overline{t_{iv}} \sum_A \overline{X_{iv}} \lambda_{ir} \right] + \left(\left[\overline{t_{iv}} - \overline{X_{iv}} \right] \sum_{R,L} \overline{X_{ir}} \lambda_{ir} \right) + \overline{X_{iv}}, \\ i &= \overline{1, M}, \quad v = \overline{1, V} \end{aligned} \tag{12.87}$$

Составляющие правой части (12.87) по смыслу совпадают с аналогичными составляющими (12.83).

Формула (12.87) предполагает наличие пуассоновского потока на входах в узлах сети. Известно, что смешение потоков приводит к суммарному потоку, близкому к пуассоновскому. Этим объясняется решение в пользу допустимости использования (12.87) для расчёта сетей большой размерности.

Рекомендации по практическому использованию (12.83) для экспоненциальных сетей в полной мере распространяются и на (12.87).

Пример 12.6

Приведем сначала пример, иллюстрирующий характер влияния неоднородности маршрутов циркуляции заявок на временные параметры сетевой модели. Пусть в простейшей сетевой модели со структурой, показанной на рис. 12.8, циркулируют заявки двух классов. Маршруты циркуляции заданы матрицами передач $\pi^{(1)}$ и $\pi^{(2)}$.

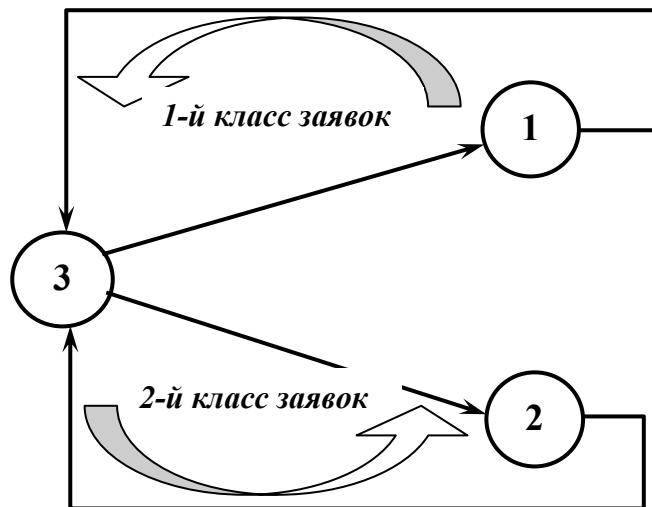


Рис. 12.8

$$\pi^{(1)} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \pi^{(2)} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

Пусть во всех узлах модели заявки обслуживаются в порядке поступления, и интенсивности обслуживания μ_i , $i = 1, 2, 3$, не зависят от класса заявок, т. е. маршруты заявок разных классов пересекаются только в узле 3.

Рассмотрим случай, когда в каждом классе всего по одной заявке. Среднее время пребывания заявки v -го класса в i -м узле можно представить как сумму времени ожидания и времени обслуживания:

$$\overline{t}_{iv} = \overline{t}_{iv}^{ож} + 1/\mu_i, \quad i = 1, 2, 3, \quad v = 1, 2.$$

Для получения некоторых предельных характеристик предположим, что $\mu_1 \ll \mu_3 \ll \mu_2$. Тогда вероятность пребывания заявки первого класса в первом узле стремится к единице: $q_{11} \rightarrow 1$ и, соответственно, $q_{31} \rightarrow 0$. Для второго класса можно утверждать, что $q_{32} \rightarrow 1$, а $q_{22} \rightarrow 0$. Таким образом, заявка первого класса в узле номер 3 практически отсутствует и $t_{32}^{ож} \rightarrow 0$.

Заявка второго класса, напротив, почти все время находится в 3-м узле, и, следовательно,

$$\overline{t_{31}^{ож}} \rightarrow 1/\mu_3.$$

Теперь можно заключить, что времена пребывания заявок разных классов в узле номер 3 отличаются примерно в два раза:

$$\overline{t_{31}} \approx 2/\mu_3, \quad \overline{t_{32}} \approx 1/\mu_3,$$

хотя заявки в этом узле обслуживаются в порядке поступления.

Приведенный пример наглядно иллюстрирует, что наличие хотя бы одной неоднородности (в данном случае неоднородности маршрутов циркуляции заявок) может привести к существенным различиям временных характеристик обслуживания заявок разных классов в узлах сети.

Пример 12.7

Данный пример служит иллюстрацией использования процедуры «свертки» при расчете характеристик сетевой модели.

Пусть структура сети та же, что и в предыдущем примере, и пусть по-прежнему в сети циркулируют две заявки разных классов в соответствии с матрицами передач $\pi^{(1)}$ и $\pi^{(2)}$. Таким образом, хотя рассматриваемая модель относительно проста, она все же является сетью с неоднородными заявками, и для ее анализа необходимо использовать наиболее общие расчетные соотношения (12.69) – (12.73).

Пусть для конкретности $\mu_1 = 1$, $\mu_2 = 0,5$, $\mu_3 = 2$. Определим в этих условиях интенсивности прохождения заявок 1-го и 2-го класса через узел номер 3.

Начнем расчет с нахождения относительных частот прохождения заявок через узлы сети ω_{iv} , где $i = 1, 2, 3$, $v = 1, 2$, в соответствии с (12.69). Поскольку любое решение системы (12.69) пригодно для использования, выберем конкретное значение ω_{iv} исходя из простоты дальнейших вычислений:

$$\begin{aligned}\omega_{11} &= 1, \quad \omega_{21} = 0, \quad \omega_{31} = 1 \\ \omega_{12} &= 0, \quad \omega_{22} = 1, \quad \omega_{32} = 1\end{aligned}$$

Значения компонентов множества G_0 заданы его определением, а значения компонентов множеств Z_1 , Z_2 , Z_3 вычислим в соответствии с (12.70) – (12.73), табл. 12.10:

Таблица 12.10

G_0	Z_1	Z_2	Z_3
$G_0(0,0) = 1$	$Z_1(0,0) = 1$	$Z_2(0,0) = 1$	$Z_3(0,0) = 1$
$G_0(0,1) = 0$	$Z_1(0,1) = 0$	$Z_2(0,1) = 2$	$Z_3(0,1) = 0,5$
$G_0(1,0) = 0$	$Z_1(1,0) = 1$	$Z_2(1,0) = 0$	$Z_3(1,0) = 0,5$
$G_0(1,1) = 0$	$Z_1(1,1) = 0$	$Z_2(1,1) = 0$	$Z_3(1,1) = 0,5$

Легко убедиться, что результат первой свертки $G_1 = G_0 * X_1 = X_1$, а потому перейдем к вычислению второй, а затем и третьей свертки, руководствуясь соотношением (12.74), табл. 12.11:

Таблица 12.11

$G_2 = G_1 * Z_2$	$G_3 = G_2 * Z_3$
$G_2(0,0) = 1$	$G_3(0,0) = 1$
$G_2(0,1) = 1 \cdot 2 + 0 \cdot 1 = 2$	$G_3(0,1) = 1 \cdot 0,5 + 2 \cdot 1 = 2,5$
$G_2(1,0) = 1 \cdot 0 + 1 \cdot 1 = 1$	$G_3(1,0) = 1 \cdot 0,5 + 1 \cdot 1 = 1,5$
$G_2(1,1) = 1 \cdot 0 + 1 \cdot 1 + 1 \cdot 2 + 0 \cdot 1 = 2$	$G_3(1,1) = 1 \cdot 0,5 + 2 \cdot 0,5 + 1 \cdot 0,5 + 2 \cdot 1 = 4$

Таким образом, нормирующая константа G , определяемая как последний компонент множества G_3 , равна 4. Теперь можно определить исходные интенсивности, пользуясь формулой (12.76):

$$\lambda_3(1) = \frac{\omega_3(1)G_3(0,1)}{G} = \frac{1 \cdot 2,5}{4} = 0,625 \frac{1}{c},$$

$$\lambda_3(2) = \frac{\omega_3(2)G_3(1,0)}{G} = \frac{1 \cdot 1,5}{4} = 0,375 \frac{1}{c}.$$

В заключение примера заметим, что увеличение числа узлов сетевой

модели, числа классов заявок, количества самих заявок не внесет никаких принципиальных затруднений в вычисления, так как основное расчетное соотношение свертки двух множеств легко поддается программированию на ЭВМ.

Пример 12.8

Система автоматического контроля осуществляет проверку сложного объекта в режиме совмещенного выполнения N программ контроля. Процесс контроля по каждой программе сводится к поочередной обработке запросов программ в центральном устройстве обработки (ЦУО) и функциональных устройствах (ФУ).

Пусть число ФУ равно 4. Таким образом, структура системы может быть представлена рис. 12.9.

Времена обработки запросов во всех устройствах случайны и распределены по экспоненциальному закону, а их средние значения соответственно равны: $\overline{t_{uyo}} = 0,2c$; $\overline{t_{fy1}} = \overline{t_{fy2}} = \overline{t_{fy3}} = 1c$; $\overline{t_{fy4}} = 0,5c$.

После усреднения характеристик программ контроля будем считать, что обращения к функциональным устройствам носят случайный характер. В данном примере вероятности этих обращений принимают равные значения:

$$p_1 = p_2 = p_3 = p_4 = 0,25.$$

Во всех устройствах системы программы обслуживаются в соответствии с относительными приоритетами. Программы контроля разделены на три приоритетных класса.

Пусть в первом, наиболее приоритетном, классе имеется одна программа, в следующем, втором — три программы, в третьем — три программы. Общее число программ, таким образом, равно семи.

Требуется определить:

1) интенсивности прохождения заявок программ каждого класса через ЦУО;

2) средние времена обслуживания программ каждого класса в каждом устройстве;

3) коэффициенты загрузки каждого устройства программами каждого класса.

Процесс функционирования рассматриваемой системы может быть сведен к марковскому процессу с дискретными состояниями, число состояний которого можно определить по известной комбинаторной формуле

$$C_{(N+M-1)}^N = \frac{(N+M-1)!}{N!(M-1)!} = 330,$$

где N — число программ, M — число ресурсов, $N=7$, $M=5$. Если использовать традиционный подход, то для получения требуемого результата пришлось бы решать систему уравнений Колмогорова с числом уравнений 330, что представляет значительные трудности и в составлении, и в решении системы.

Перейдем к сетевой стохастической модели. Структура графа сети практически совпадает с рис. 12.9. Пронумеруем узлы сети в порядке 1, 2, 3, 4, 5, начиная с ЦУО.

Моделью исследуемой системы будет замкнутая стохастическая сеть массового обслуживания с постоянным и равным семи числом заявок, обслуживаемых в соответствии с относительными приоритетами между классами и в порядке поступления внутри классов.

Используя введенные выше обозначения, вектор состава заявок может быть записан как $\vec{N} = (1,3,3)$. Матрица передач в нашем случае примет вид

$$\pi = \begin{pmatrix} 0 & 0,25 & 0,25 & 0,25 & 0,25 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

для любого класса заявок. Решив систему уравнений (12.22), получим значения относительных частот: $\omega_1 = 0,5$; $\omega_2 = \omega_3 = \omega_4 = \omega_5 = 0,125$.

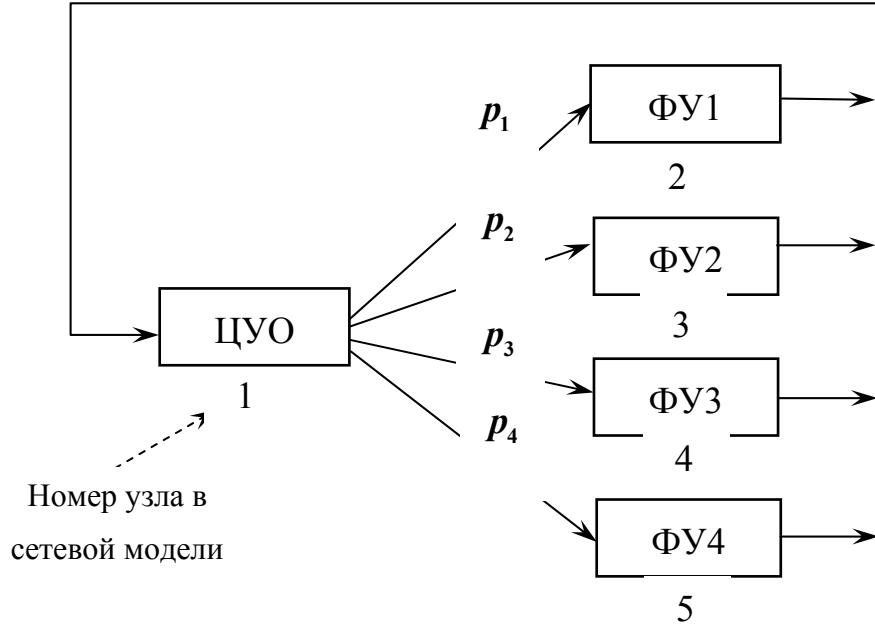


Рис.12.9

В соответствии с заданной дисциплиной обслуживания определим матрицы-диспетчеры как

$$D_1 = D_2 = D_3 = D_4 = D_5 = \begin{Bmatrix} F & R & R \\ \bar{R} & F & R \\ \bar{R} & \bar{R} & F \end{Bmatrix}.$$

Заметим, что только в нашем простом примере все матрицы-диспетчеры одинаковы, в общем случае они могут быть разными для каждого узла.

Теперь можно составить основное расчетное соотношение для нашего случая на основе (12.83). С учетом относительных приоритетов среднее время пребывания заявок v -го класса в i -м узле может быть выражено как

$$\overline{t}_{iv} = \sum_{j=1}^{j=v-1} \frac{q_{ij} N_j}{\mu_j} + [N_v - 1] \frac{q_{iv}}{\mu_i} + \sum_{j=v+1}^V \frac{\lambda_{ij}}{\mu_i^2} + \left[\overline{t}_{iv} - \frac{1}{\mu_i} \right] \sum_{j=1}^{j=v-1} \frac{\lambda_{ij}}{\mu_i} + \frac{1}{\mu_i}. \quad (12.88)$$

Выражение для вероятности пребывания заявок v -го класса в i -м узле и их интенсивности прохождения имеют вид соответственно

$$q_{iv} = \frac{\overline{t_{iv}} \cdot \omega_i}{\sum_{j=1}^5 \overline{t_{jv}} \cdot \omega_j}; \quad (12.89)$$

$$\lambda_{iv} = \frac{N_v \cdot q_{iv}}{\overline{t_{iv}}}. \quad (12.90)$$

Всего необходимо составить 15 уравнений вида (12.88), так как в нашей модели пять узлов и три класса заявок. Полученную систему из 15 уравнений следует решать итерационным численным методом с учетом (12.89) и (12.90).

Приведем уравнения для средних времен пребывания заявок каждого из трех классов в двух первых узлах сетевой модели (т. е. шесть уравнений из 15 входящих в систему):

$$\overline{t_{11}} = \frac{\lambda_{12} + \lambda_{13}}{\mu_1^2} + \frac{1}{\mu_1};$$

$$\overline{t_{12}} = \frac{q_{11} + 2q_{12}}{\mu_1} + \frac{\lambda_{13}}{\mu_1^2} + \left[\overline{t_{12}} - \frac{1}{\mu_1} \right] \frac{\lambda_{11}}{\mu_1} + \frac{1}{\mu_1};$$

$$\overline{t_{13}} = \frac{q_{11} + 3q_{12}}{\mu_1} + \frac{2q_{13}}{\mu_1} + \left[\overline{t_{13}} - \frac{1}{\mu_1} \right] \frac{\lambda_{11} + \lambda_{12}}{\mu_1} + \frac{1}{\mu_1};$$

$$\overline{t_{21}} = \frac{\lambda_{22} + \lambda_{13}}{\mu_2^2} + \frac{1}{\mu_2};$$

$$\overline{t_{22}} = \frac{q_{21} + 2q_{22}}{\mu_2} + \frac{\lambda_{23}}{\mu_2^2} + \left[\overline{t_{22}} - \frac{1}{\mu_2} \right] \frac{\lambda_{21}}{\mu_2} + \frac{1}{\mu_2};$$

$$\overline{t_{23}} = \frac{q_{21} + 3q_{22}}{\mu_2} + \frac{2q_{23}}{\mu_2} + \left[\overline{t_{23}} - \frac{1}{\mu_2} \right] \frac{\lambda_{21} + \lambda_{22}}{\mu_2} + \frac{1}{\mu_2}.$$

Остальные девять уравнений легко могут быть составлены по анало-

гии и потому здесь не приводятся.

Для решения составленной системы уравнений может быть применен любой итерационный метод, в том числе и наиболее доступный для реализации «метод простых итераций».

Суть последнего состоит в том, что начальные приближения для \bar{t}_{iv} (а также вычисленные на основе \bar{t}_{iv} значения q_{iv} и λ_{iv}), $i = 1, 2, 3, 4, 5$, $v = 1, 2, 3$, подставляются в правые части уравнений и вычисляются новые значения \bar{t}_{iv} , которые опять подставляются в правые части уравнений и т. д. Результаты решения системы уравнений относительно \bar{t}_{iv} , полученные при использовании метода простых итераций и начальных приближений

$$\bar{t}_{iv} = \frac{1}{\mu_i}, \quad i = 1, 2, 3, 4, 5; \quad v = 1, 2, 3,$$

приводятся в табл. 12.12.

Используя соотношения (12.89) и (12.90), можно определить:

$$\begin{aligned}\lambda_{11} &= 0,6184; \quad \lambda_{12} = 1,4157; \quad \lambda_{13} = 0,8047; \\ \lambda_{21} &= \lambda_{31} = \lambda_{41} = 0,1546; \quad \lambda_{22} = \lambda_{32} = \lambda_{42} = 0,3539; \\ \lambda_{23} &= \lambda_{33} = \lambda_{43} = 0,2012; \\ \lambda_{51} &= 0,1546; \quad \lambda_{52} = 0,3539; \quad \lambda_{53} = 0,2012 \text{ } c^{-1}.\end{aligned}$$

Коэффициенты загрузки рассчитываются по формуле $\rho_{iv} = \lambda_{iv}/\mu_{iv}$.

Таким образом, с помощью сетевой модели удается преодолеть трудности, связанные с большой размерностью марковской модели, и получить искомые характеристики путем сравнительно несложных вычислений.

Таблица 12.12

Средние времена	Шаг итерации						
	1	2	3	4	5	6	7
\bar{t}_{11}	0,423	0,299	0,295	0,291	0,290	0,289	0,289
\bar{t}_{12}	0,423	0,372	0,364	0,359	0,356	0,355	0,354
\bar{t}_{13}	0,423	0,482	0,513	0,522	0,525	0,526	0,525
$\bar{t}_{21}, \bar{t}_{31}, \bar{t}_{41}$	2,395	1,617	1,593	1,570	1,562	1,557	1,555
$\bar{t}_{22}, \bar{t}_{32}, \bar{t}_{42}$	2,395	2,191	2,169	2,142	2,128	2,120	2,115
$\bar{t}_{23}, \bar{t}_{33}, \bar{t}_{43}$	2,395	3,051	3,508	3,761	3,913	4,002	4,054
\bar{t}_{51}	0,849	0,654	0,648	0,642	0,640	0,639	0,639
\bar{t}_{52}	0,849	0,726	0,711	0,703	0,700	0,698	0,697
\bar{t}_{53}	0,849	0,833	0,825	0,814	0,809	0,806	0,805

Пример 12.9

Определим среднее время пребывания заявок в узлах разомкнутой сети, представленной на рис. 12.10. Источник (нулевой узел) генерирует заявки двух классов с разной интенсивностью $\lambda_{01} = 0,1c^{-1}, \lambda_{02} = 0,2c^{-1}$.

Матрица передач $\pi^{(v)}, v = 1, 2$, выглядит следующим образом:

$$\pi^{(v)} = \begin{Bmatrix} 0 & 1 & 0 & 0 \\ 0,2 & 0 & 0,3 & 0,5 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{Bmatrix}.$$

Структура сети на рис. 12.10 отображает работу мультипроцессорной системы, работающей в реальном масштабе времени и обрабатывающей информацию от двух объектов [60].

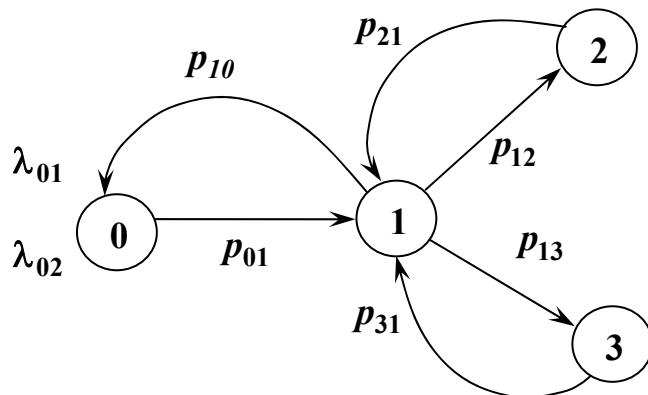


Рис. 12.10

Информация от первого объекта поступает с меньшей интенсивностью, но нуждается в более быстрой обработке. Поэтому, переходя к модели, назначим приоритеты заявок первого класса по отношению к заявкам второго класса: в узле 1 — абсолютный, в узле 2 — относительный. Тогда матрицы-диспетчеры задаются следующим образом:

$$D_1 = \begin{Bmatrix} F & A \\ A & F \end{Bmatrix}, \quad D_2 = \begin{Bmatrix} F & R \\ R & F \end{Bmatrix}, \quad D_3 = \begin{Bmatrix} F & F \\ F & F \end{Bmatrix}. \quad (12.91)$$

Интенсивность обслуживания в узлах не зависит от принадлежности классу: $\mu_1 = 2 c^{-1}$, $\mu_2 = \mu_3 = 1 c^{-1}$.

По матрице передач определим интенсивность прохождения заявок через узлы в соответствии с (12.60)

$$\begin{cases} \lambda_{01} = 0,2\lambda_{11} \\ \lambda_{11} = \lambda_{01} + \lambda_{21} + \lambda_{31} \\ \lambda_{21} = 0,3\lambda_{11} \\ \lambda_{31} = 0,5\lambda_{11} \end{cases}. \quad (12.92)$$

Отсюда $\lambda_{11} = 0,5 c^{-1}$, $\lambda_{21} = 0,15 c^{-1}$, $\lambda_{31} = 0,25 c^{-1}$. Аналогичная (12.92) система равенств может быть составлена и для λ_{i2} , $i=1,2,3$. Другой способ расчета λ_{i2} , $i=1,2,3$ — использование очевидной линейной зависимости $\lambda_{i2} = \frac{\lambda_{02}}{\lambda_{01}}\lambda_{i1}$. Тогда $\lambda_{i2} = 2\lambda_{i1}$, $i=1,2,3$.

Итак, найдены интенсивности прохождения заявок через узлы. Теперь можно перейти к определению времен пребывания заявки в узлах в

соответствии с соотношениями (12.66) и матрицами-диспетчерами (12.91):

а) для первого класса

$$\begin{cases} \overline{t_{11}} = \left[\frac{\lambda_{11}}{\mu_1^2} \right] \left[1 - \frac{\lambda_{11}}{\mu_1} \right]^{-1} [1 - 0]^{-1} + \frac{1}{\mu_1} \\ \overline{t_{21}} = \left[\frac{\lambda_{21}}{\mu_2^2} + \frac{\lambda_{22}}{\mu_2^2} \right] \left[1 - \frac{\lambda_{21}}{\mu_2} \right]^{-1} [1 - 0]^{-1} + \frac{1}{\mu_2} \\ \overline{t_{31}} = \left[\frac{\lambda_{31}}{\mu_3^2} + \frac{\lambda_{32}}{\mu_3^2} \right] \left[1 - \frac{\lambda_{31} + \lambda_{32}}{\mu_3} \right]^{-1} [1 - 0]^{-1} + \frac{1}{\mu_3} \end{cases}, \quad (12.93)$$

б) для второго класса

$$\begin{cases} \overline{t_{12}} = \frac{1}{\mu_1} \frac{\lambda_{11}}{\mu_1} \left[1 - \frac{\lambda_{11}}{\mu_1} \right]^{-1} + \\ + \left[\frac{\lambda_{11}}{\mu_1^2} + \frac{\lambda_{12}}{\mu_1^2} \right] \left[1 - \frac{\lambda_{11} + \lambda_{12}}{\mu_1} \right]^{-1} \left[1 - \frac{\lambda_{11}}{\mu_1} \right]^{-1} + \frac{1}{\mu_1} \\ \overline{t_{22}} = \left[\frac{\lambda_{21}}{\mu_2^2} + \frac{\lambda_{22}}{\mu_2^2} \right] \left[1 - \frac{\lambda_{21} + \lambda_{22}}{\mu_2} \right]^{-1} \left[1 - \frac{\lambda_{21}}{\mu_2} \right]^{-1} + \frac{1}{\mu_2} \\ \overline{t_{32}} = \left[\frac{\lambda_{31}}{\mu_3^2} + \frac{\lambda_{32}}{\mu_3^2} \right] \left[1 - \frac{\lambda_{31} + \lambda_{32}}{\mu_3} \right]^{-1} [1 - 0]^{-1} + \frac{1}{\mu_3} \end{cases}. \quad (12.94)$$

Расчёты по формулам (12.93), (12.94) позволяют определить времена пребывания заявок двух классов во всех узлах, табл. 12.13. Как видно из результатов, введение приоритетов позволило уменьшить время пребывания заявок первого класса по отношению ко второму в 3,7 раза для абсолютного и в 1,3 раза для относительного. Напомним, что интенсивность потока заявок второго класса выше, чем первого, в два раза.

Рассмотрим теперь влияние интенсивностей обслуживания на временные характеристики сети. Пусть сеть остается прежней (см. рис. 12.10), но интенсивности обслуживания теперь зависят от принадлежности классу: $\mu_{i1} = 2\mu_i$, $\mu_{i2} = \mu_i$, $i = \overline{1,3}$, и во всех узлах заявки обслуживаются в порядке поступления. Соотношения (12.66) примут вид:

$$\overline{t_{iv}} = \left[\frac{\lambda_{i1}}{\mu_{i1}^2} + \frac{\lambda_{i2}}{\mu_{i2}^2} \right] \left[1 - \frac{\lambda_{i1}}{\mu_{i1}} - \frac{\lambda_{i2}}{\mu_{i2}} \right]^{-1} + \frac{1}{\mu_{iv}}, \quad v=1,2; \quad i=1,2,3. \quad (12.95)$$

Подставив значения параметров в (12.95), получим результат, представленный в последних двух столбцах табл. 12.13.

Таблица 12.13

Номер узла i	$\mu_{i1} = \mu_{i2} = \mu_i$		$\mu_{i1} = 2\mu_i, \mu_{i2} = \mu_i$	
	Номер класса		Номер класса v	
	1	2	1	2
1	0,67	2,67	1,00	1,25
2	1,53	1,96	1,04	1,54
3	4,00	4,00	2,00	2,50

Двойное увеличение интенсивности обслуживания заявок первого класса уменьшило время пребывания заявок обоих классов в 1,27–2,14 раза почти во всех узлах. Исключение составляют заявки первого класса в первом узле, лишившиеся абсолютного приоритета.

Мы оценивали относительное изменение средних времен пребывания заявок. Для оценки абсолютных значений этих изменений используется обычно среднее время цикла сети, которое равно

$$\overline{T_{uv}} = \sum_{i=1}^M \alpha_{iv} \overline{t_{iv}}, \quad v=1,2, \quad \text{и представляет собой среднее время про-}$$

хождения заявки через сеть.

Для варианта с приоритетным обслуживанием

$$\overline{T_{u1}} = 0,67 \cdot 5 + 1,53 \cdot 1,5 + 4,00 \cdot 2,5 = 15,64 \text{ с.}$$

$$\overline{T_{u2}} = 2,67 \cdot 5 + 1,96 \cdot 1,5 + 4,00 \cdot 2,5 = 26,29 \text{ с.}$$

Для второго варианта, где увеличена интенсивность обслуживания заявок первого класса,

$$\overline{T_{u1}} = 1,00 \cdot 5 + 1,04 \cdot 1,5 + 2,00 \cdot 2,5 = 11,56 \text{ с.}$$

$$\overline{T_{u2}} = 1,25 \cdot 5 + 1,54 \cdot 1,5 + 2,50 \cdot 2,5 = 14,81 \text{ с.}$$

Сравнивая эти два варианта, можно отметить, что введение приори-

тетов существенно перераспределяет доступ к обслуживающему прибору в узле и тем самым уменьшает время прохождения заявки по сети. Увеличение интенсивности обслуживания заявок одного из классов ведет к уменьшению времени цикла заявок всех классов, но в большей степени данного класса.

В заключение приведем для сравнения расчетные соотношения и результаты анализа однородной разомкнутой сети той же структуры.

Если сеть однородна, то $\lambda_0 = \lambda_{01} + \lambda_{02} = 0,3 c^{-1}$, времена пребывания определяются по формуле

$$\bar{t}_i = \frac{\lambda_i}{\mu_i^2} \left[1 - \frac{\lambda_i}{\mu_i} \right]^{-1} + \frac{1}{\mu_i}, \quad i = \overline{1, M},$$

а характеристики сети принимают следующие значения:

$$\bar{t}_1 = 2 \text{ с}, \quad \bar{t}_2 = 1,82 \text{ с}, \quad \bar{t}_3 = 4,00 \text{ с}, \quad \bar{T_u} = 22,73 \text{ с}.$$

ЗАДАЧИ

1. Задана сеть массового обслуживания, рис. 12.11.

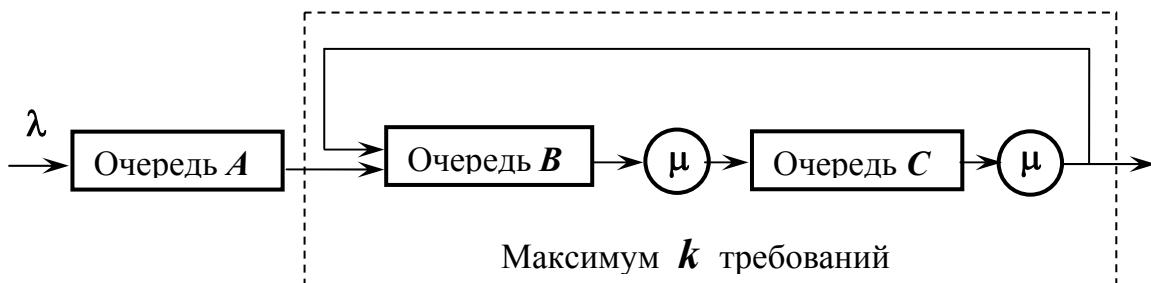


Рис. 12.11

На входе сети — пуассоновский поток с интенсивностью λ . При поступлении, требование проходит в блок, выделенный штриховой линией только, если число таких требований меньше k . В противном случае оно становится в очередь A , из головы которой требования поступают в штриховой блок.

Поступив в штриховой блок, требование становится в очередь B и со временем в соответствии с дисциплиной «первым пришел — первым обслужен» получает обслуживание с показательным распределением времени интенсивностью μ .

Затем оно переходит в очередь C и получает аналогичное обслуживание на независимом приборе с такими же характеристиками. После этого требование покидает штриховой блок (и систему) с вероятностью P или возвращается в очередь B с вероятностью $(1 - P)$.

Предполагая, что в системе существует установившийся режим, найти:

- а) среднее число требований во всей системе (в системе «очередь + штриховой блок»);
- б) среднее время \bar{T} пребывания требования во всей системе для значений k , равных 1 и ∞ ;
- в) условия стационарности для значений k , равных 1 и ∞ ;
- г) для общей области стационарности доказать, что

$$\bar{T}(k=1) \geq \bar{T}(k=\infty).$$

2. Задана сеть массового обслуживания с M узлами и матрицей передач R , ($i, j = \overline{1, M}$). Требование покидает сеть с вероятностью $1 - \alpha_M$. Интенсивность внешнего источника требований равна λ_0 .

$$R = \begin{pmatrix} \alpha_1 & 1 - \alpha_1 & 0 & 0, & \dots & \dots, & 0 \\ \alpha_2 & 0 & 1 - \alpha_2 & 0, & \dots & \dots, & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \alpha_{M-1} & 0 & 0 & 0, & \dots & \dots, & 1 - \alpha_{M-1} \\ \alpha_M & 0 & 0 & 0, & \dots & \dots, & 0 \end{pmatrix}$$

Найти следующие показатели системы:

- а) трафик λ_i , $i = \overline{1, M}$, каждого узла сети в явном виде.
- б). предполагая, что все узлы являются одноканальными, с интенсивностью обслуживания μ_i , $i = \overline{1, M}$, определить:
 - условия установившегося режима;
 - соотношение между μ_i , обеспечивающее один и тот же показа-

тель загрузки обслуживающего прибора.

в). Характеристики сети для случая

$$\alpha_i = \alpha, \quad \mu_i = \mu, \quad i = \overline{1, M}.$$

3. Задана замкнутая сеть массового обслуживания с пятью узлами, $M = 5$, где i -й узел имеет m_i обслуживающих приборов, каждый из которых требует $\frac{1}{\mu_i}$ с. обслуживания при показательном законе распределения. Пусть $m_5 = M$, $m_1 = m_2 = m_3 = m_4 = 1$ и 5-й узел является терминальным. Сеть содержит M требований, циркулирующих в соответствии с матрицей передач R .

$$R = \begin{pmatrix} 0 & 1/2 & 1/2 & 0 & 0 \\ 0 & 0 & 1/2 & 1/2 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Пусть $\mu_1 = 3 c^{-1}$, $\mu_2 = 2 c^{-1}$, $\mu_3 = \mu_4 = 3 c^{-1}$, $\mu_5 = 2/17 c^{-1}$.

Требуется:

- а) построить граф сети;
- б) определить отношения λ_i/λ_5 для $i = \overline{1, 4}$;
- в) определить наиболее нагруженный узел;
- г) определить среднее время T ответа (от момента ухода требования из 5-го узла до момента возвращения).

4. Задана разомкнутая сеть массового обслуживания, включающая 4 узла и источник с интенсивностью $\lambda_0 = 0,25 c^{-1}$. В дальнейшем источник будем считать нулевым узлом. Матрица R передач задана:

$$R = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0,02 & 0 & 0,18 & 0,8 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix},$$

$$m_1 = 1, \quad m_2 = m_3 = m_4 = 2,$$

где m_i — число каналов в i -м узле, а интенсивности обслуживания в узлах,

$$\mu_1 = 20 \text{ c}^{-1}, \quad \mu_2 = 1 \text{ c}^{-1}, \quad \mu_3 = 7 \text{ c}^{-1}, \quad \mu_4 = 9 \text{ c}^{-1}.$$

Требуется:

- а) выяснить, может ли данная сеть работать в установившемся режиме;
- б) определить наиболее нагруженный узел;
- в) определить среднее время реакции (среднее время пребывания требования в сети).

5. Задана разомкнутая сеть массового обслуживания (рис. 12.12).

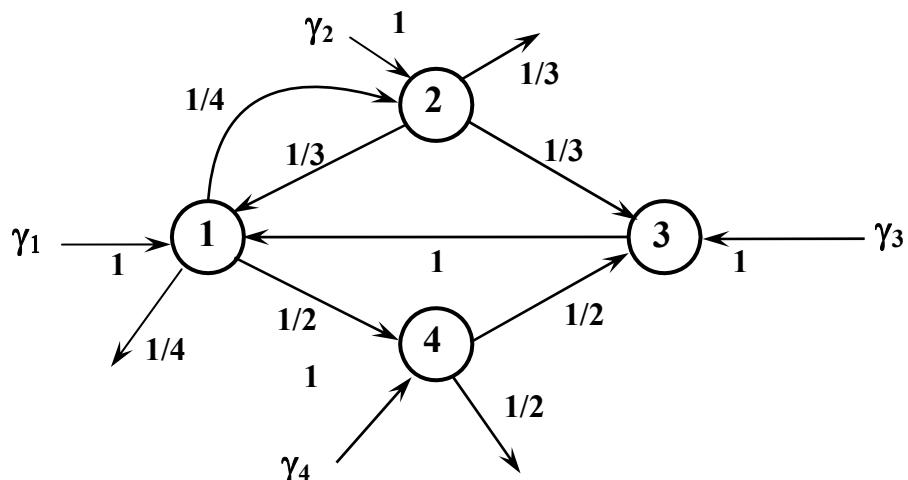


Рис. 12.12

Интенсивности γ_i внешних источников требований, поступающих в сеть, равны: $\gamma_1 = 1, \quad \gamma_2 = 0, \quad \gamma_3 = 5, \quad \gamma_4 = 2$.

Требуется:

- а) найти интенсивность требований для i -го узла;
- б) найти характеристики узлов при следующих условиях:

$$m_1 = m_3 = 1, \quad m_2 = m_4 = 2 ;$$

$$\mu_1 = 20 \text{ c}^{-1}, \quad \mu_2 = 1 \text{ c}^{-1}, \quad \mu_3 = 7 \text{ c}^{-1}, \quad \mu_4 = 9 \text{ c}^{-1} ;$$

- в) определить среднее время пребывания требований в сети для различных внешних источников.

6. Задана разомкнутая неоднородная сеть с числом узлов, равным трём, $M = 3$ и таким же числом $L = 3$ внешних источников требований, трафик которых представлен на рис. 12.13. а), б), в).

Предполагается равновероятное распределение требований в трафике, т. е.

$$P_j^{(1)} = 1/3, \quad j = \overline{1, 3} \text{ — (вариант а, рис. 12.13.а));}$$

$$P_j^{(2)} = P_j^{(3)} = 1/2, \quad j = 1, 2 \text{ — (варианты б и в, рис. 12.13.а) и в).}$$

Выполнить следующее задание.

1). Найти интенсивность неоднородных потоков заявок в узлах сети и сформулировать условия наличия установившегося режима в сети с одноканальными узлами.

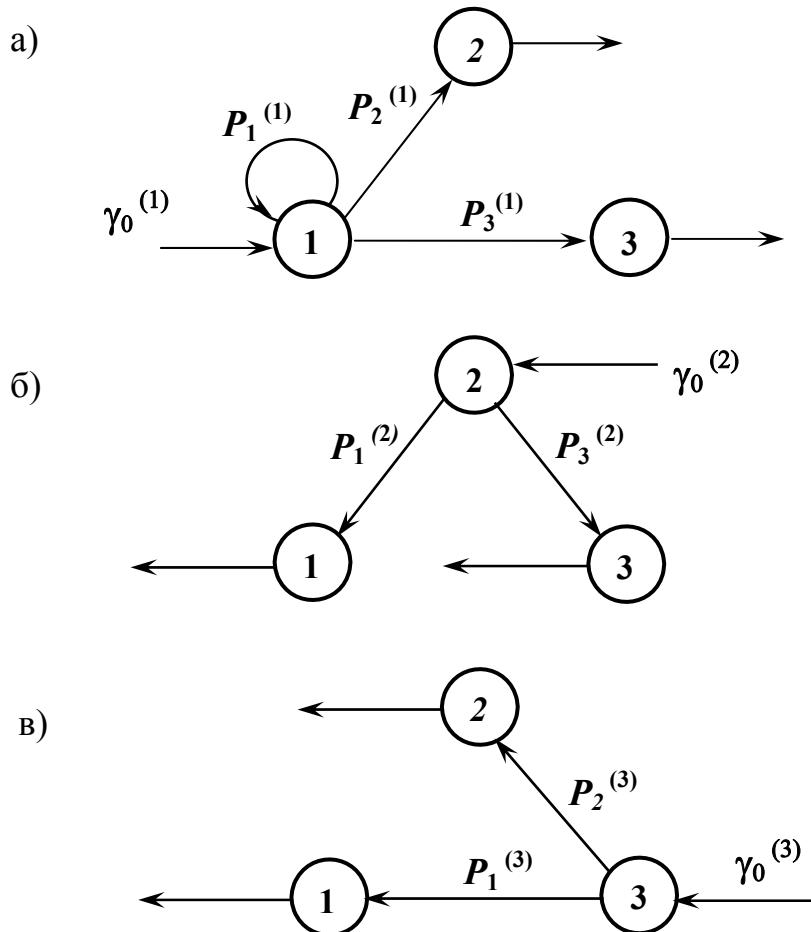


Рис. 12.13

2). Для заданных векторов интенсивностей обслуживания в узлах:

$$[\mu^{(1)}]^T = (1, 2, 3);$$

$$[\mu^{(2)}]^T = (3, 2, 1);$$

$$[\mu^{(3)}]^T = (1, 3, 2)$$

найти интенсивности входных потоков $\gamma_0^{(i)}$, $i = \overline{1, 3}$, максимизирующих суммарную пропускную способность при наличии установленного режима в сети.

3). Проанализировать и сравнить следующие дисциплины обслуживания в узлах сети:

- бесприоритетное обслуживание в порядке поступления,
- обслуживание с относительными приоритетами для всех 3! возможных вариантов их назначения.

7. Задана сеть массового обслуживания, включающая три узла, $M = 3$. Число каналов обслуживания в узлах определяется вектором $m^T = (1 \ 1 \ 1)$, интенсивности обслуживания — вектором

$$\mu^T = (2 \text{ c}^{-1}, \ 0,8 \text{ c}^{-1}, \ 0,1 \text{ c}^{-1}).$$

В сети циркулируют N заявок в соответствии с матрицей передач R :

$$R = \begin{pmatrix} 0,1 & 0,4 & 0,5 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$

Требуется:

- а) определить характеристики узлов и сети в целом ($N = 3$);
- б) сопоставить характеристики узлов указанной сети с сетью, где узлы 2 и 3 объединены в один узел с числом каналов, равным двум и усреднённой интенсивностью $0,45 \text{ c}^{-1}$.

8. Рассматривается сетевая модель вычислительной системы, представленная на рис. 12.14, где узел 1 моделирует обслуживание пакета N программ в процессоре, а узлы 2 и 3 — обслуживание в различных ступенях внешней памяти.

Параметры узлов:

$$m_1 = 1, \ m_2 = 1, \ m_3 = 1,$$

$$\mu_1 = 0,5 \text{ c}^{-1}; \ \mu_2 = 0,6 \text{ c}^{-1}; \ \mu_3 = 1 \text{ c}^{-1}.$$

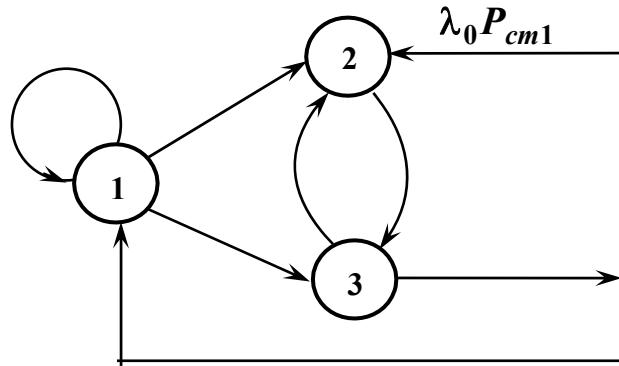


Рис. 12.14

Число программ N равно 2. Заданы следующие варианты структуры сети с соответствующими матрицами передач:

Вариант 1:

$$P = \begin{pmatrix} 0,2 & 0,3 & 0,5 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix};$$

Вариант 2:

$$P = \begin{pmatrix} 0,2 & 0,8 & 0 \\ 0,37 & 0 & 0,63 \\ 0 & 1 & 0 \end{pmatrix};$$

Вариант 3:

$$P = \begin{pmatrix} 0 & 0,5 & 0,5 \\ 1 & 0 & 0 \\ 0,4 & 0,6 & 0 \end{pmatrix};$$

Вариант 4:

$$P = \begin{pmatrix} 0,35 & 0,55 & 0,1 \\ 0,5 & 0 & 0,5 \\ 0,5 & 0,5 & 0 \end{pmatrix}.$$

Требуется:

а) Провести сравнительный анализ различных структур по следующей системе показателей: коэффициент загрузки узла и среднее время пребывания требования в узле.

б) Провести сравнительный анализ различных структур при условии, что $N = 0$, а в первый узел поступает поток внешних заявок с интенсивностью λ_0 . Значение λ_0 выбрать исходя из обеспечения коэффициента загрузки 1-го узла на уровне **0,8** с учетом наличия установленного режима в сети.

9. Рассматривается вычислительная система с перекрёстными связями, состоящая из равного числа n процессоров и модулей оперативной памяти (рис. 12.15.).

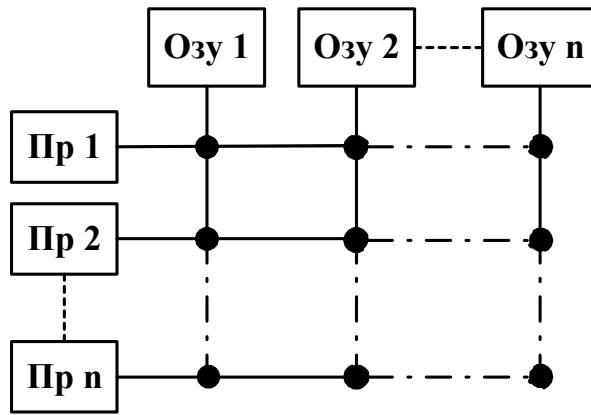


Рис. 12.15

Обращение процессора к любому модулю памяти равновероятно. Потоки запросов на ОЗУ имеют интенсивность λ_i , обслуживание происходит с интенсивностью μ_i , ($i = \overline{1, n}$).

Выполнить следующее задание:

- определить характеристики занятости магистралей и задержки в выполнении программ;
- определить зависимость вероятности простоя из-за занятости магистралей в зависимости от изменения параметра λ_i на $\pm 50\%$ при постоянных значениях μ_i , ($i = \overline{1, n}$) и от изменения параметра μ_i на $\pm 50\%$ при постоянных значениях λ_i , ($i = \overline{1, n}$);
- сравнить указанные зависимости с такими же при наличии одной магистрали обмена при тех же условиях. Исходные данные приведены в табл. 12.14.

Таблица 12.14

Вариант	n	λ_1	λ_2	λ_3	μ_1	μ_2	μ_3
1	2	2	3	–	2,5	3	–
2	2	10	15	–	20	30	–
3	3	2	3	5	3	4	8
4	3	20	30	25	50	30	40

10. Объектом исследования является мультипрограммная вычислительная система, функционирующая в режиме оперативной обработки информации. Система предназначена для решения заданного набора L

задач. Задачи, поступающие от пользователей через мультиплексный канал (МК), передаются в подсистему ОП – ПР (оперативная память – процессор).

Каждая задача характеризуется набором следующих параметров:

- интенсивностью λ_{0i} поступления запросов на её решение (табл. 12.15);
- трудоёмкостью Θ_{0i} (тыс. опер.) процессорных операций (табл. 12.16.);
- трудоёмкостью операций обмена с внешней памятью, задаваемой средним числом K_{ij} обращений к файлу F_j в процессе решения i -й задачи (табл. 12.16).

Множество файлов $\{F_j\}$, $j = \overline{1, N}$, где N – мощность множества, используемое в процессе решения данного набора задач, размещается во внешней памяти системы.

Внешние запоминающие устройства (ВЗУ) состоят из накопителей двух типов: накопителей на магнитных дисках (НМД) и накопителей на магнитных лентах (НМЛ), которые связаны с подсистемой (ОП – ПР) через селекторные каналы связи (СК), рис. 12.16.

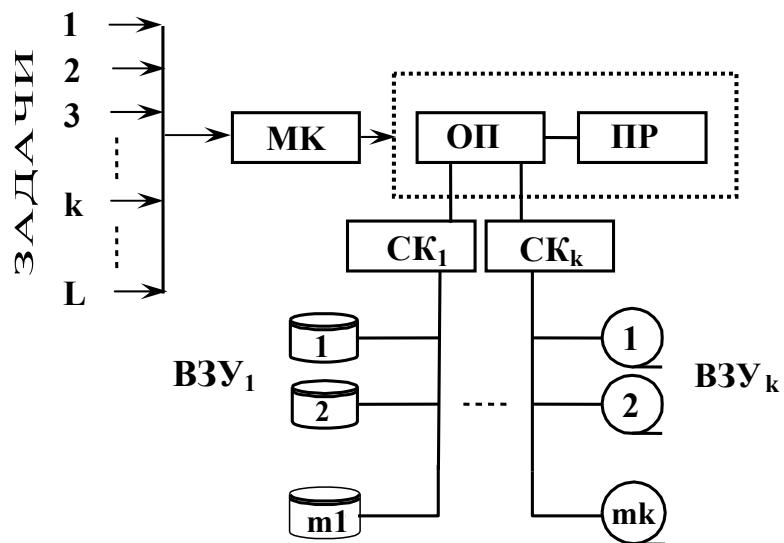


Рис. 12.16

Файл F_j характеризуется:

- длиной G_j (Мбайт), (табл. 12.17);
- средней длиной записи g_j (Кбайт), (табл. 12.17).

Накопители (НМД, НМЛ) характеризуются следующими параметрами:

рами:

- средним временем доступа к данным, размещенным в НМД ($\overline{t_{MD}}$) и в НМЛ ($\overline{t_{ML}}$), (табл. 12.18);
- скоростью передачи данных при обмене через селекторный канал (V_{MD} и V_{ML}), (табл. 12.18);
- емкостью накопителя (G_{MD} и G_{ML}) (табл. 12.18).

Построение модели предполагает следующие допущения.

1). Процесс решения задачи представляется произвольной последовательностью строго чередующихся этапов счёта и обращения к файлам (обмена данными между внешней и оперативной памятью системы). Причем этап счёта всегда начинает и завершает решение задачи.

2). Любое устройство вычислительной системы отображается в модели сети массового обслуживания одноканальным или многоканальным узлом.

3). Потоки задач простейшие и время обслуживания в узлах сети подчиняется экспоненциальному закону.

4). Дисциплина обслуживания в узлах сети бесприоритетная — в порядке поступления заявок.

Требуется:

- привести неоднородный поток задач к однородному;
- предложить варианты размещения файлов во внешних накопителях;
- выбрать число накопителей с учётом заданной ёмкости и требований по обеспечению условий установившегося режима в сети;
- выбрать производительность процессора, обеспечивающую коэффициент загрузки на уровне 0,9;
- выбрать необходимое количество селекторных каналов исходя из необходимости обеспечения установившегося режима в сети;
- вычислить среднее время реакции (среднее время пребывания задачи в системе) и выявить наиболее загруженный узел.

Интенсивности поступления задач и их иные параметры представлены в табл. 12.15 – 12.18.

11. Оценка эффективности управления трафиком в виртуальном канале с помощью механизма «скользящего окна». Виртуальный канал представляет собой последовательное объединение нескольких каналов с разной пропускной способностью.

Пакеты передаются от источника получателю по виртуальному каналу. Пакет считается успешно принятным на стороне получателя в случае доставки источнику подтверждающей квитанции.

Таблица 12.15

№ варианта	Состав задач							
	№ задачи	λ_{0i}	№ задачи	λ_{0i}	№ задачи	λ_{0i}	№ задачи	λ_{0i}
1	1	0,10	5	0,02	10	0,01	9	0,04
2	7	0,04	4	0,03	1	0,05	10	0,02
3	5	0,01	6	0,03	4	0,02	3	0,04
4	2	0,09	1	0,02	5	0,01	4	0,02
5	3	0,01	2	0,01	6	0,05	7	0,03

Таблица 12.16

№ задачи	Трудоёмкость процессорной операции Θ_{0i} (тыс. опер.)	Среднее число K_{ij} обращений к файлам									
		F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
1	100	10	5	—	—	—	—	2	1	—	—
2	200	—	8	5	3	—	—	—	—	3	—
3	300	—	—	10	—	5	—	—	—	—	2
4	400	12	4	—	—	—	3	—	2	—	—
5	500	—	15	8	—	6	—	4	—	—	—
6	600	8	—	8	—	—	7	—	—	3	1
7	700	10	—	—	5	—	—	1	—	2	—
8	800	—	12	6	—	8	—	—	2	—	2
9	900	10	5	—	9	—	—	—	—	—	3
10	1000	—	15	—	—	—	10	3	—	4	—

Таблица 12.17

Файл F_j	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
Длина G_j файла (Мбайт)	0,5	1,0	1,0	1,5	1,5	2,0	2,5	3,0	4,0	5,0
Средняя длина g_j записи (Кбайт)	5	8	15	6	14	18	10	15	20	25

Таблица 12.18

№ варианта	Среднее время доступа к данным (сек)		Скорость передачи данных (кбайт/сек)		Ёмкость накопителя (мбайт)	
	t_{MD}	t_{ML}	V_{MD}	V_{ML}	G_{MD}	G_{ML}
1	0,05	1,0	200	100	4,0	10
2	0,06	1,5	190	90	5,0	12
3	0,07	2,0	180	80	5,5	15
4	0,08	2,5	170	75	6,0	20
5	0,09	3,0	160	70	6,5	22

Как только число неподтвержденных пакетов становится равным предельному значению N (размеру «окна»), виртуальный канал блокируется для передачи новых пакетов. Методология моделирования трафика в виртуальном канале рассмотрена в [48].

Требуется:

- а) предложить модель трафика в виртуальном канале;
 - б) определить вероятность блокировки виртуального канала и среднее время передачи пакета;
 - в) определить «узкое место» в виртуальном канале.
- Варианты заданий приведены в табл. 12.19.

Таблица 12.19

№ задания	Интенсивность источника λ_0	Размер «окна» N	Состав виртуального канала		
			Канал 1	Канал 2	Канал 3
			μ_1	μ_2	μ_3
1	40	15	70	120	130
2	50	20	90	130	110
3	60	10	160	150	140
4	70	25	180	170	150
5	80	30	200	210	190

12. Исследование маршрутизатора глобальной компьютерной сети, обеспечивающего обмен данными по протоколу. Входящие пакеты записываются в буферную память маршрутизатора, обрабатываются процессором и передаются по каналу в соответствующем направлении. Копия

принятого пакета хранится в буферной памяти до тех пор, пока не будет получена квитанция о безошибочной доставке пакета адресату. Отсутствие квитанции-подтверждения в течение времени time-out вынуждает маршрутизатор к повторной передаче пакета по тому же каналу в направлении адресата. При отсутствии места в буферной памяти входящие в маршрутизатор пакеты получают отказ.

Требуется построить модель маршрутизатора в форме сети массового обслуживания. При этом следует иметь в виду, что поступивший в маршрутизатор пакет занимает буферную память в течение следующих интервалов времени:

- 1) ввод в маршрутизатор и запись в буфер со скоростью канала связи,
- 2) ожидание в очереди на обработку в процессоре,
- 3) обработка в процессоре,
- 4) ожидание освобождения канала в направлении выхода их маршрутизатора,
- 5) передача из маршрутизатора по выходному каналу,
- 6) ожидание квитанции об удачной доставке с возможным повторением п. 5 (по истечении интервала time-out).

Каждая из отмеченных временных задержек пакета в маршрутизаторе обеспечивается ожиданием и обслуживанием в соответствующем узле сети массового обслуживания. Условия задачи приводят к разомкнутой сети с блокировками, которая может быть заменена стандартной замкнутой сетью массового обслуживания, характеризуемой тем же марковским процессом перехода из состояния в состояние [47].

Определить следующие характеристики маршрутизатора:

- а) вероятность отказа в приёме пакета в буферную память,
- б) среднее время пребывания пакета в буферной памяти,
- в) среднее число пакетов в маршрутизаторе,
- г) среднюю интенсивность потока пакетов, занимающих буферную память маршрутизатора,
- д) зависимость указанных показателей от размера буферной памяти маршрутизатора.

Варианты заданий приведены в табл. 12.20.

Примечание. Предполагается равновероятная передача пакета по любому из L каналов, среднее время time-out принимается равным $2(\mu_{nep})^{-1}$, время успешной доставки квитанции равно $0,1(\mu_{nep})^{-1}$. Все временные задержки в маршрутизаторе, в том числе интервал time-out, и время доставки квитанции считаются случайными величинами, распределёнными по показательному закону. Входной поток пакетов — простейший.

Таблица 12.20

№ задания	Интенсивность λ_0 поступления пакетов	Размер буфера N	Число каналов L	Интенсивность μ_{nep} передачи пакетов по каналу	Интенсивность μ_{pr} обработки пакета в процессоре	Вероятность неуспешной передачи пакета по каналу
1	20	50	20	1	10	0,2
2	15	40	18	1	10	0,1
3	5	30	10	0,5	8	0,15
4	7	20	8	0,5	10	0,2
5	9	25	9	0,8	15	0,1

13. Исследование сегмента локальной сети организованной по типу ETHERNET. Построить модель сегмента в форме сети массового обслуживания в соответствии со следующими рекомендациями.

В модели сегмента необходимо обеспечить формирование всех временных задержек, реально сопутствующих процессу передачи кадра по магистрали в соответствии с методом МДКН/ОС (множественным доступом с контролем несущей и обнаружением столкновений).

Перечислим эти задержки:

а) t_1 — время обнаружения столкновения кадров, находящееся в интервале от 0 до величины $T = 2L/V$, где L — длина сегмента, а V — скорость распространения сигнала в физической среде;

б) t_2 — время передачи кадра после обнаружения столкновения для фиксации его другими станциями;

в) τ_i — задержка в передаче кадра при обнаружении i -го для этого кадра столкновения: $\tau_i = \epsilon T$,

где $\epsilon = RANDOM[0, 2^k - 1]$, а $k = \min(i, 10)$;

г) $t_{\text{передачи кадра}}$ — время передачи кадра, определяемое его длиной.

д) t_{mz} — время межкадровой задержки, необходимое для завершения переходных процессов в магистрали.

Таким образом, для $i \leq 10$ среднее значение τ_i определим как

$$\bar{\tau}_i \approx \frac{2^i}{2} \frac{2L}{V} = 2^{i-1} \frac{2L}{V},$$

то есть $\bar{\tau}_1 = \frac{1}{2} \left(\frac{2L}{V} \right)$, а далее $\bar{\tau}_i$ удваивается с каждой попыткой.

Начиная с 11-й попытки $\bar{\tau}_i$ не меняется: $\bar{\tau}_i = \bar{\tau}_{10}$, $i \geq 10$.

Вероятность столкновения P_{col} определяется как вероятность появления кадра от абонента сегмента на входе в магистраль на интервале $[0, T]$: $P_{col} = P\{ t_u < T \} = 1 - e^{-\lambda T}$, где $f(t_u) = \lambda e^{-\lambda t_u}$, а λ — интенсивность потока кадров на входе в магистраль сегмента.

В соответствии с приведенной интерпретацией стандарта ETHERNET модель сегмента может быть представлена в форме неоднородной разомкнутой сети с заявками, меняющими класс при переходе из узла в узел. Структура сети для предельного числа попыток посылки кадра после столкновений, равного двум, представлена на рис. 12.17.

Для выполнения задания необходимо самостоятельно задать необходимые количественные параметры сегмента сети:

- длину L сегмента;
- скорость распространения V сигнала в физической среде;
- время t_2 передачи кадра после обнаружения столкновения для фиксации его другими станциями (интервал Jam_time);
- время $t_{передачи\ кадра}$ передачи кадра, определяемое его длиной;
- время $t_{мз}$ межкадровой задержки, необходимое для завершения переходных процессов в магистрали; длину и интенсивность поступающих в сегмент кадров.

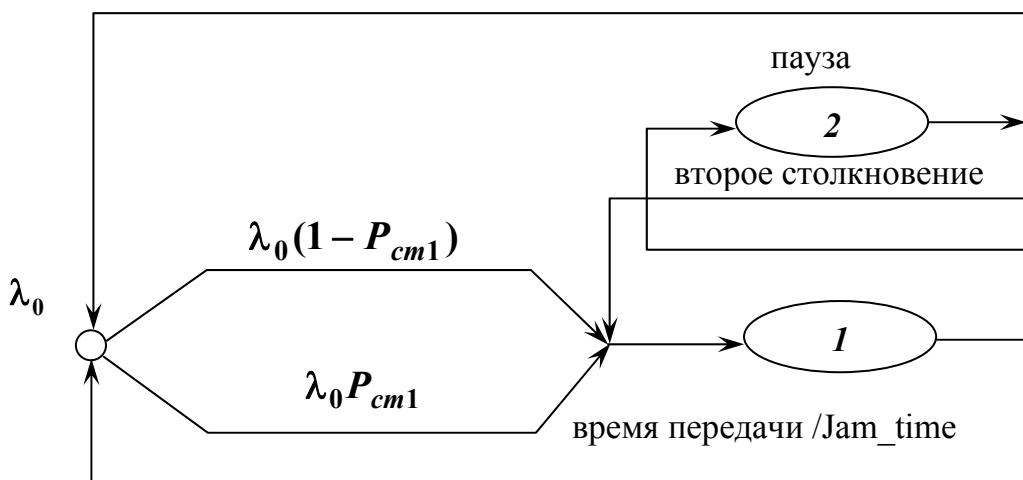


Рис. 12.17

Требуется определить следующие характеристики сегмента:

- вероятность отказа в передаче кадра;
- среднее время пребывания кадра в магистрали;
- вероятность столкновения кадров;
- провести исследование зависимости задержки в передаче кадра от интенсивности λ внешнего потока.

14. Модификация условий задачи 12. На вход маршрутизатора поступают пакеты двух типов: «длинные» и «короткие», занимающие выделенные для них области буферной памяти и характеризуемые разными значениями средних времён передачи по каналу. Исходные данные для решения задачи приведены в табл. 12.21.

Таблица 12.21

№ задания	Интенсивность поступления пакетов		Размер буфера		Число каналов L	Интенсивность μ_{nep} передачи пакетов по каналу		Интенсивность μ_{npr} обработки пакета в процессоре	Вероятность успешной передачи пакета по каналу
	λ_{01}	λ_{02}	N_1	N_2		μ_{nep}^1	μ_{nep}^2		
1	8	12	20	30	20	0,5	1,0	10	0,20
2	7	8	15	25	18	0,5	1,0	10	0,10
3	2	3	10	20	10	0,2	0,5	8	0,15
4	3	4	8	12	8	0,2	0,5	10	0,20
5	3	6	8	17	9	0,3	0,8	15	0,10

Примечание. Предполагается равновероятная передача пакета по любому из L каналов, среднее время time-out принимается равным $2(\mu_{nep})^{-1}$, время успешной доставки квитанции — $0,1(\mu_{nep})^{-1}$. Все временные задержки в маршрутизаторе, в том числе интервал time-out, и время доставки квитанции считаются случайными величинами, распределёнными по показательному закону. Входные потоки пакетов — простейшие.

13. Имитационное моделирование

13.1. Понятие об имитационном моделировании. Основные его этапы и особенности

Имитационной моделью (ИМ) сложной системы будем называть машинные алгоритмы и программы, позволяющие имитировать на ЭВМ поведение системы или ее отдельных компонентов и связей между ними в течение заданного времени [38].

Можно выделить следующие основные этапы имитационного моделирования.

1. Формулировка проблемы: определение цели моделирования в виде вопросов, требующих ответа, либо гипотез, которые необходимо проверить, либо воздействий, которые надо оценить. В соответствии с выбранной целью определяются показатели, по которым оцениваются результаты имитации.

2. Построение концептуальной модели, обеспечивающей переход от неструктурированного содержательного описания системы к описанию с элементами формализации.

3. Построение математической модели с определением входных, выходных, управляющих переменных и раскрытием их взаимосвязи в общем алгоритме функционирования системы с целью оценки значений выбранных показателей. Для имитации математическая модель чаще всего представляется в виде алгоритмического описания моделируемого процесса.

4. Программирование моделирующего алгоритма, т. е. разработка самой имитационной модели (ИМ).

5. Отладка, тестирование и проверка адекватности ИМ.

6. Планирование экспериментов, предполагающее решение следующих задач:

- выбор способов ускорения сходимости статистических оценок интересующих показателей к истинным значениям;
- определение объема имитационных экспериментов;
- составление плана проведения экспериментов, что особенно актуально при решении задач оптимизации на основе имитации.

7. Проведение экспериментов и обработка результатов с целью формулирования выводов по результатам имитации и обоснования их точности.

Существенной проблемой ИМ является необходимость имитации параллельного поведения отдельных элементов системы.

Чтобы обеспечить имитацию параллельных событий системы на конечном множестве моментов времени T , в ИМ используется специальная

переменная t , называемая системным модельным временем или просто модельным временем (МВ).

Модельное время t следует отличать от других используемых при ИМ систем типов времени, таких как:

t_p — реальное время системы S , функционирование которой имитируется;

t_m — машинное время имитации, отражающее затраты ресурса времени ЭВМ на организацию имитационного моделирования.

Существуют два способа формирования конечного множества моментов системного времени T , известных как « Δt » и « Δx » организации изменения модельного времени.

Принцип « Δt » (time slicing) заключается в изменении МВ с дискретным шагом Δt .

Принцип « Δx » (event scheduling) заключается в изменении МВ при скачкообразном изменении вектора состояний x системы S . Скачкообразные изменения состояний системы S происходят при наступлении таких «особых» событий как поступление управляющих сигналов, внешних воздействий, выдача выходных сигналов и др. Применительно к системам массового обслуживания состояние (число требований в системе) изменяется скачкообразно в зависимости от прихода требований в систему или от завершения обслуживания в обслуживающих приборах.

Для систем с дискретными событиями, исследуемых в пособии, принцип « Δx » является более предпочтительным по следующим причинам: во-первых, он характеризуется меньшими затратами машинного времени, во-вторых, обеспечивает большую точность аппроксимации фазовой траектории, описывающей поведение системы S в фазовом пространстве.

В зависимости от подхода к отображению временного поведения системы различают три основные формы процедурной организации моделирующей программы, предписывающие представление последней в виде упорядоченного набора описаний активностей, событий или процессов.

Активность — единица работы, характеризуемая условиями начала и временем выполнения.

Событие — мгновенное изменение состояния системы вследствие внешних или внутренних причин.

Процесс — логически взаимосвязанная последовательность событий

Эти понятия не являются обособленными. Так активность может сама по себе представлять последовательность активностей очень близкую по смыслу к процессу. Начало и окончание активности, по сути, являются событиями. В определении понятия «процесс» используется понятие «со-

бытие». Выбор формы представления модели — это выбор определенного взгляда (точки зрения) на поведение системы. На рис. 13.1 – 13.3 приведены структуры моделирующих программ, составленных в соответствии с указанными подходами

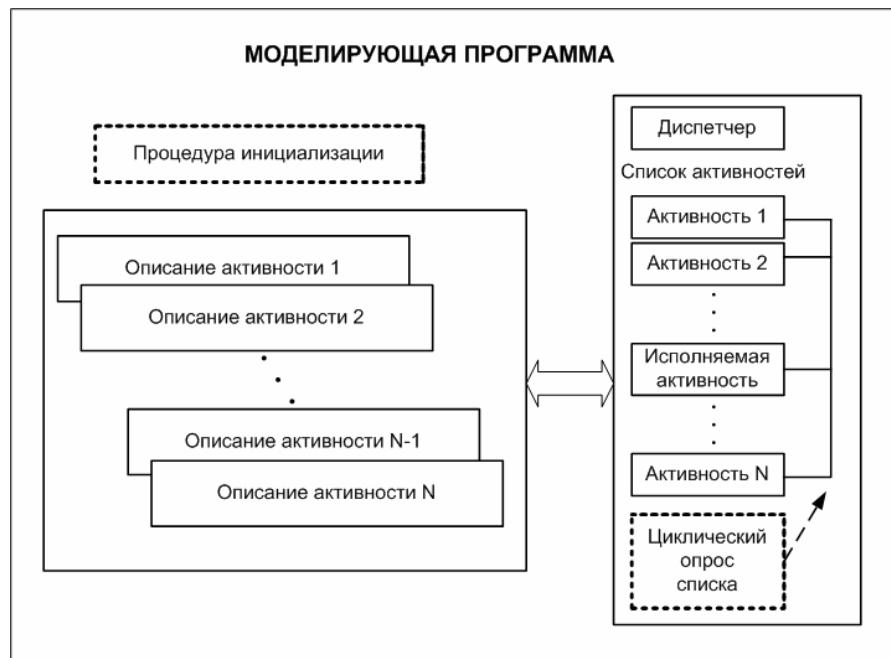


Рис. 13.1

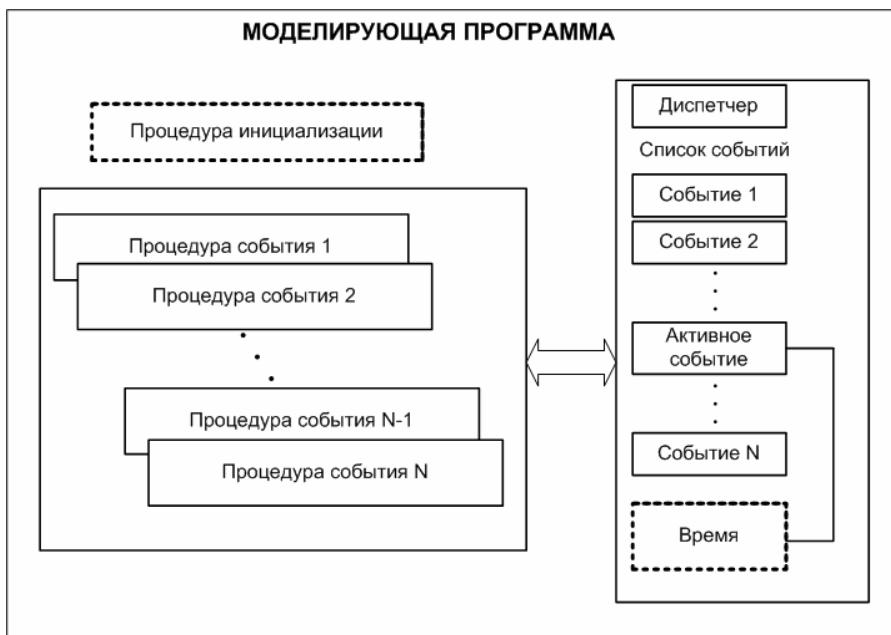


Рис. 13.2

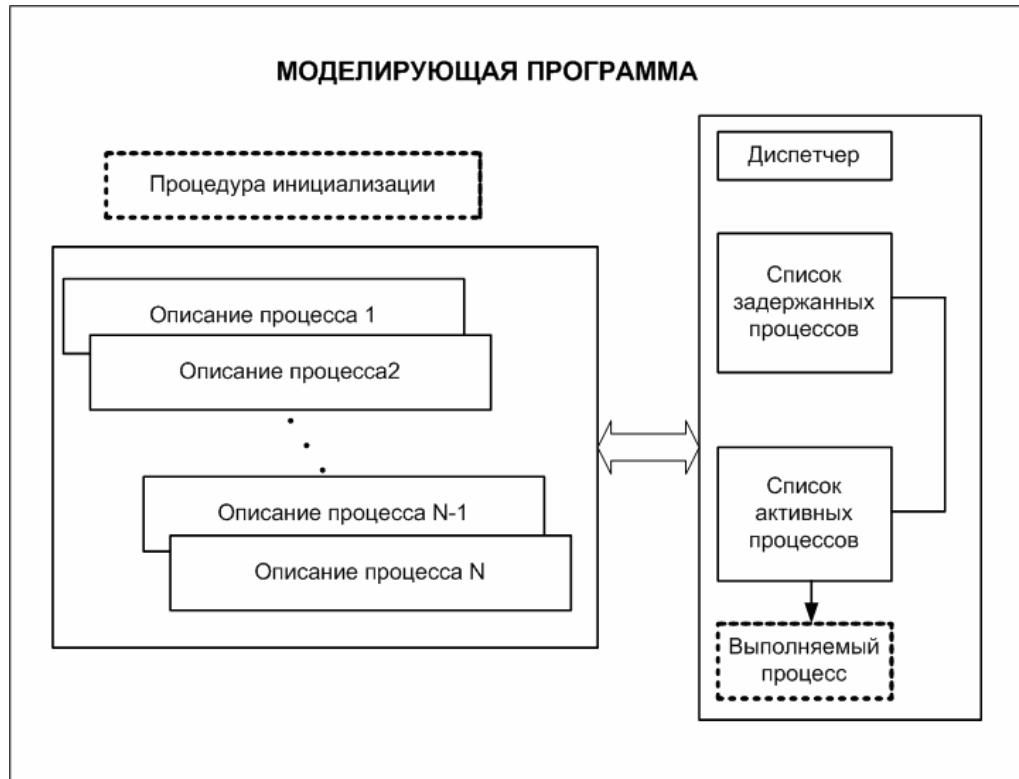


Рис. 13.3

В принципе, система имитации модели во времени, обеспечивающая квазипараллельное исполнение фрагментов модели, может быть интегрирована в моделирующую программу, составляя с ней единое целое. Именно так построен алгоритм имитации одноканальной МО с ограниченной очередью (см. раздел 13.2), представляющий пример событийного подхода к формированию имитационной модели.

В практике имитационного моделирования принято разделять собственно описание системы — моделирующую программу и программу-ДИСПЕТЧЕР, реализующую упомянутый выше квазипараллелизм. ДИСПЕТЧЕР выполняет планирование событий (event scheduling) в моделируемой системе, оперируя списковыми структурами типа списка событий, списка процессов и т. п., определяя текущее активное событие или точку возобновления активного процесса.

Любую систему можно представить одним из рассмотренных способов, отчетливо представляя, что построенные на их основе имитационные модели могут различаться размерами и количеством ресурсов, затрачиваемых на их создание, испытание и использование.

Далее в пособии более подробно рассматривается событийный подход к построению имитационных моделей.

13.2. Событийный подход к построению моделирующих алгоритмов

При построении алгоритма имитации в соответствии с данным подходом функционирование системы, формализованное в математической модели, рассматривается как совокупность параллельно протекающих процессов, причем каждый процесс есть некоторая последовательность однородных событий, с каждым событием связано изменение состояния системы.

Событие, возникающее в системе, определяется как особое состояние. Процессы, в общем случае, не являются независимыми, а взаимодействуют между собой. Для иллюстрации данных понятий рассмотрим пример.

Пример 13.1

Пусть в систему массового обслуживания, состоящую из одного прибора обслуживания, поступает поток требований. Если требование, приходя в систему, застает прибор свободным, то оно занимает прибор и обслуживается в нем в течение некоторого времени, после чего покидает систему. В противном случае требование поступает в очередь к прибору, где будет ожидать конца обслуживания всех ранее поступивших требований. В такой системе возможны два процесса:

1) процесс поступления требований;

2) процесс освобождения обслуживающего прибора, причем процесс поступления требований инициирует второй процесс, если последний был приостановлен по причине отсутствия требований.

Под состоянием системы будем понимать, как и ранее, число требований, находящихся в системе.

Из алгоритма функционирования видно, что состояние системы изменяется либо под влиянием событий процесса N_1 (приход требований), либо под влиянием событий процесса N_2 (освобождение прибора). При построении алгоритма моделирования ДИСПЕТЧЕР выявляет наличие одной из двух ситуаций:

- поступление требования;
- освобождение прибора.

Эти однородные события и являются элементами двух разных процессов. События процессов «поступление требования» и «освобождение обслуживающего прибора» могут совпадать.

С целью формализации событийного подхода для каждого выделен-

ногого i -го процесса ($i = \overline{1, n}$) определим момент T_i наступления очередного события. Затем найдем наиболее ранний момент наступления особого состояния в соответствии с операцией:

$$T_r = \min_{(i)} \{ T_i \},$$

где r — номер процесса (потока событий), в котором наступило ближайшее событие.

Момент T_i называется моментом системного времени (МВ), в отличие от реального времени, в котором работает моделирующая ЭВМ.

Структурная схема моделирующего алгоритма представлена на рис. 13.4.

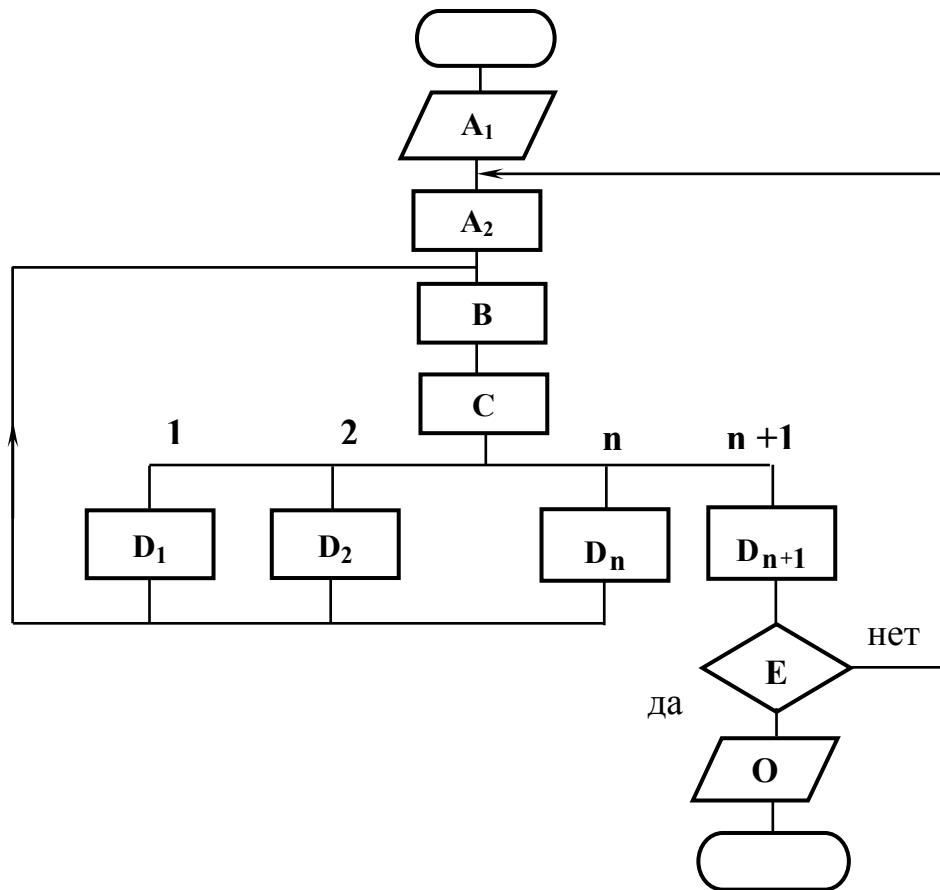


Рис. 13.4

Раскроем назначение основных операторов.

A_1 — оператор задания начальных условий для моделируемого варианта.

A_2 — оператор задания начальных условий для одной реализации.

B — оператор определения очередного момента изменения состоя-

ния и самого состояния в соответствии с соотношением

$$T_r = \min_{(i)} \{ T_i \}. \quad (13.1)$$

C — оператор выбора блока реакции D_r , на текущее событие в r -м потоке, где r определяется оператором B .

D_r — оператор реакции, реализующий следующие функции:

- выполнение всех необходимых операций, предусмотренных алгоритмом функционирования системы, как реакцию на данное событие;
- вычисление и накопление статистики по интересующим показателям качества, если она формируется по событиям данного процесса;
- определение следующего момента поступления события в данном процессе и состояния системы и занесения их в соответствующие структуры данных для оператора B ;
- вычисление и занесение в данные оператора B моментов поступления событий и состояний в других процессах, если последние были приостановлены, а активизация их возможна только событием данного процесса; в результате эти заторможенные процессы становятся активными. Следует заметить, что, если процесс переходит в приостановленное состояние, то время поступления очередного события определить нельзя. В этом случае обычно принимают $T_j = \infty$ (где j — номер такого процесса) для того, чтобы исключить этот процесс из массива $\{T_i\}$ в силу указанного выше соотношения (13.1).

D_{n+1} — оператор, который вводится в тех случаях, когда имитация производится на ограниченном интервале длины T_0 . T_{n+1} принимается равным T_0 , длительности одного имитационного эксперимента. Когда T_0 окажется очередным моментом изменения состояния, управление передается оператору D_{n+1} , выполняющему все необходимые действия по завершению одного имитационного эксперимента.

E — оператор проверки достаточности проведенных экспериментов для достижения требуемой статистической точности;

O — оператор обработки и выдачи результатов имитационных экспериментов на печать.

Для конкретизации и более детального знакомства с данным подходом рассмотрим пример.

Пример 13.2

Простейший пример моделирования одноканальной СМО с потерями. В систему поступает однородный простейший поток требований. Дли-

тельность обслуживания — случайная величина с экспоненциальным законом распределения. На входе системы установлен накопитель. Если требование застает прибор занятым и накопитель заполненным, то оно теряется, получает отказ. В результате моделирования необходимо оценить вероятность отказа в обслуживании требований. Структурная схема алгоритма приведена на рис. 13.5.

Под реализацией будем понимать процесс обработки одного требования.

В системе приняты следующие обозначения:

ТПОСТ — момент времени поступления требования в систему,

ТОСВ — момент времени освобождения обслуживающего прибора,

T — момент системного времени,

T — длительность интервала между моментами поступления соседних требований,

K — количество требований в системе, находящихся на обслуживании и в очереди,

ОЧЕР — количество требований в очереди,

KOTK — количество требований, получивших отказ в обслуживании,

КОЛ — количество поступивших требований (текущее число реализаций),

H — число заданных реализаций,

M — число мест для ожидания в очереди (в накопителе).

Оператор **1** осуществляет ввод заданных значений **H**, **M** и присваивает нулевые значения счетчикам числа требований, поступивших в систему (**КОЛ**), находящихся в системе (**K**) и в очереди (**ОЧЕР**), получивших отказ в обслуживании (**KOTK**), а также переменным, имеющим значения текущего времени поступления требований (**ТПОСТ**) и системного времени (**T**). Переменной, имеющей значение текущего времени освобождения прибора (**ТОСВ**), присваивается значение, равное машинной бесконечности (∞).

Оператор **2** формирует случайный интервал между соседними поступающими требованиями в соответствии с моделью простейшего потока.

Оператор **3** формирует момент поступления первого требования в систему.

Операторы **4, 5, 19** совместно определяют очередной момент изменения состояния системы.

Операторы **6 – 15** составляют оператор реакции **D₁** алгоритма на приход требования в систему.

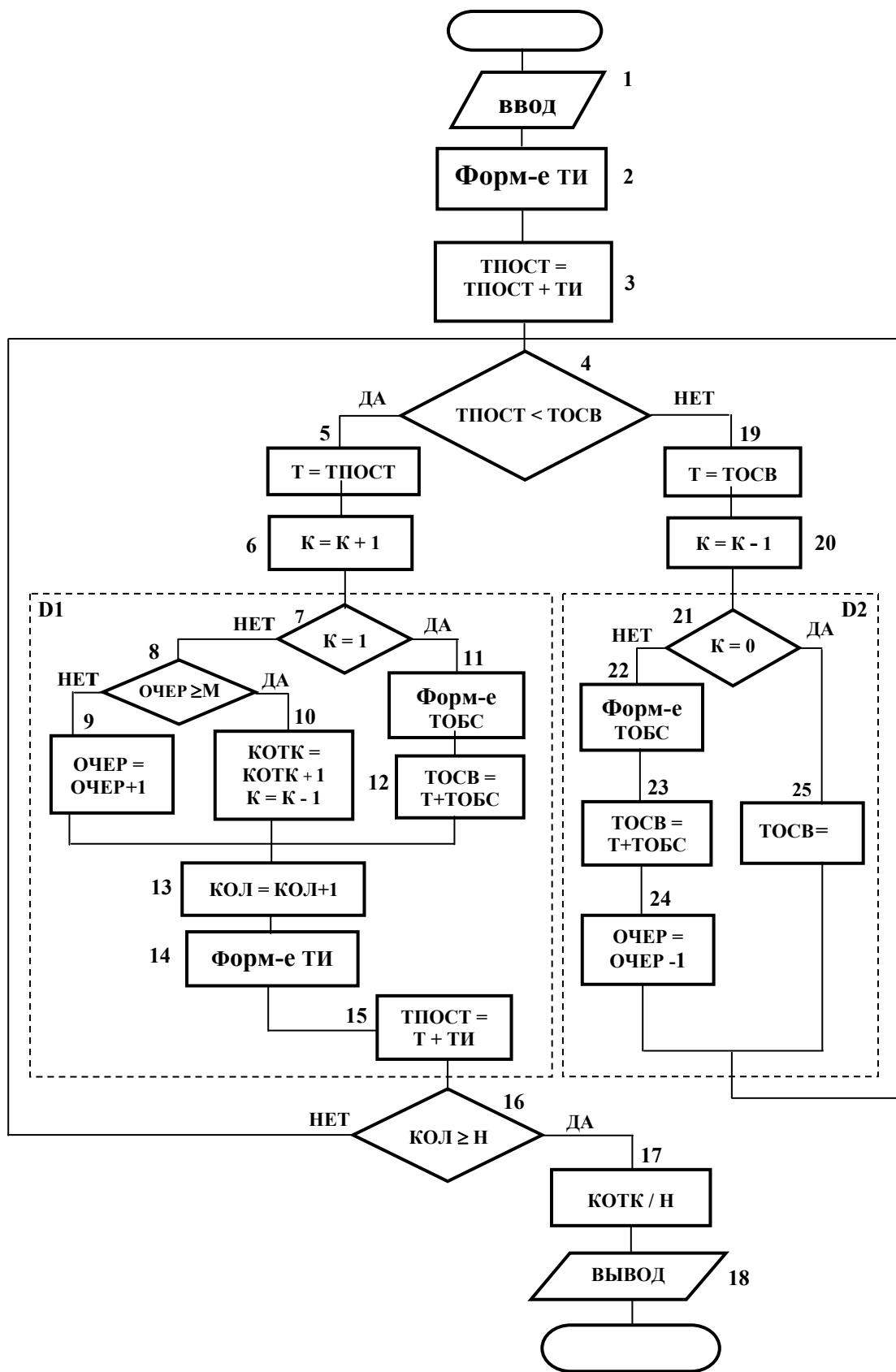


Рис. 13.5

Операторы **20 – 25** составляют оператор реакции **D₂** на событие, связанное с освобождением обслуживающего прибора.

В обобщенных операторах **D₁**, **D₂** операторы **11** и **22** моделируют имитацию случайной длительности обслуживания в соответствии с экспоненциальным законом: $\text{TOБС} = \left(-\frac{1}{\mu}\right) \cdot \ln y_i$, где μ — интенсивность обслуживания, а $y_i \in R(0,1)$. В обобщенном операторе **D₁** оператор **14** также, как и оператор **2**, воспроизводит модель простейшего потока: $\text{ТИ} = \left(-\frac{1}{\lambda}\right) \cdot \ln y_i$, где λ — параметр потока, а $y_i \in R(0,1)$.

Условный оператор **16** производит сравнение текущего числа требований (**КОЛ**), поступивших в систему с заданным числом (**H**). В случае, если условие не выполняется, управление передается оператору **4**, в противном случае управление передается **17**, осуществляющему обработку результатов. В данном случае, оператор **17** вычисляет оценку вероятности отказа в обслуживании, равную **КОТК/H**, и выводит на устройство печати результаты моделирования.

Хотя данный пример достаточно прост, он позволяет методически почувствовать основные особенности имитации случайных потоков и законов обслуживания при моделировании СМО.

Рассмотрим далее более подробно основные принципы моделирования случайных факторов.

13.3. Общие принципы имитации случайных факторов

Базовой последовательностью случайных чисел, используемой для формирования в ЭВМ случайных элементов различной природы с различными законами распределения, является совокупность случайных чисел с равномерным законом распределения:

$$f(x) = \begin{cases} \frac{1}{b-a}, & x \in [a,b] \\ 0, & x < a, \quad x > b \end{cases}, \quad (13.2)$$

где $f(x)$ — дифференциальный закон распределения равномерно распределенных чисел x в интервале $[a, b]$.

Строго говоря, на ЭВМ получить последовательность непрерывных случайных величин с равномерным законом распределения не представляется возможным. Поэтому, если считать, что число разрядов ЭВМ равно k , а случайное число сформировано согласно формуле:

$$\alpha = \sum_{j=0}^{k-1} 2^j , \quad (13.3)$$

где $\alpha_j = \begin{cases} 0, P_j = 1/2 \\ 1, P_j = 1/2 \end{cases}$, то $x = \alpha / (2^k - 1)$ принимает значение $i / (2^k - 1)$, $i = 0, 1, 2, \dots, 2^k - 1$ с вероятностью $P_i = 1/2^k$. Такое распределение чисел называется квазиравномерным в интервале $[0,1]$, причем математическое ожидание и дисперсия определяются следующими соотношениями:

$$M[x] = \sum_{i=0}^{2^k-1} \frac{i}{2^k-1} \frac{1}{2^k} = \frac{1}{2},$$

$$D[x] = \sum_{i=0}^{2^k-1} \left(\frac{i}{2^k-1} - \frac{1}{2} \right)^2 \frac{1}{2^k} = \frac{1}{12} \frac{2^k+1}{2^k-1}.$$
(13.4)

Из формул (13.4) видно, что математическое ожидание $M[x]$ точно совпадает с генеральным средним для равномерного распределения в интервале $[0,1]$; а дисперсия при $k \rightarrow \infty$ асимптотически стремится к дисперсии для равномерного распределения при $a = 0, b = 1$, равной $1/12$. Практически при $k > 15$ достигается необходимая точность моделирования. Поэтому в дальнейшем будем говорить о равномерном законе, хотя в действительности при программном моделировании имеем дело с квазиравномерным.

При выводе выражений (13.4) предполагалось, что число α формируется на основе случайных чисел α_j , принимающих значения $(0,1)$ с вероятностью $P_j = 1/2$, для чего в ЭВМ должен существовать генератор, дающий строго случайные последовательности чисел α_j с соответствующим распределением. Так как в ЭВМ такого генератора нет, случайные числарабатываются программным путем, в силу чего они, строго говоря, не являются случайными, так как формируются на основе вполне детерминированных преобразований. Поэтому их называют псевдослучайными. Существует достаточно большое число способов генерации последовательности квазиравномерных псевдослучайных чисел. Правомерность же применения того или иного способа получения случайных чисел (СЧ) определяется только результатом статистической проверки.

Для проверки качества генерируемой последовательности СЧ используются различные системы проверочных тестов. Укажем наиболее часто применяемые тесты.

Тест частот

Отрезок $[0,1]$ разбивают на m (обычно $10 - 20$) равных интервалов. Полученные эмпирические частоты n_i / N попадания случайного числа в i -й интервал

$$n_i / N, \quad i = \overline{1, m}; \quad \sum_{i=1}^m n_i = N,$$

где N — объем выборки, сравнивается с теоретическими вероятностями $P_i = 1/m$.

Согласие проверяется по критерию χ^2 , так как статистическая функция

$$\chi^2 = \frac{1}{NP_i} \sum_{i=1}^m (n_i - NP_i)^2 \quad (13.5)$$

подчиняется распределению χ^2 с $(m - 1)$ степенями свободы [63].

По выбранному уровню значимости $q\%$ для числа степеней свободы $(m - 1)$ в соответствии с таблицами χ^2 распределения выбирают порог χ_q^2 .

Если значение χ^2 , вычисленное в соответствии с (13.5), меньше χ_q^2 , то гипотеза о соответствии выборочного распределения теоретическому, т. е. равномерному, принимается. В противном случае гипотеза отвергается.

Тест пар

Рассматриваются пары СЧ при делении интервала $[0,1]$ на m частей. Каждая пара случайно попадает в одно из m^2 делений квадратной таблицы $m \times m$. При этом в зависимости от метода образования пар изменяется число степеней свободы. Пусть дана серия чисел x_1, x_2, \dots, x_N . Если пары образовать в виде $(x_1, x_2), (x_3, x_4), \dots$, то пары взаимно независимы. Эмпирические частоты сравниваются с теоретическими вероятностями $1/m^2$ равномерного распределения. Функция

$$\chi^2 = \left(\frac{2m^2}{N} \right) \sum_{i=1, j=1}^m \left(n_{ij} - \frac{N}{2m^2} \right)^2 \quad (13.6)$$

распределена по закону χ^2 с $(m^2 - m)$ степенями свободы, где n_{ij} — число попаданий в (i, j) -ю клетку таблицы размером $m \times m$, $N/2$ — число пар СЧ, составляющих выборку.

В отличие от теста частот, направленного на проверку близости рас-

пределения полученной последовательности к равномерной, тест пар направлен на проверку независимости, а точнее отсутствия корреляции в последовательности СЧ.

Тест на периодичность

Если среди множества программно получаемых СЧ x_1, x_2, \dots, x_{r-1} нет одинаковых, а следующее случайное число x_r совпадает с одним из полученных ранее чисел, то r называется отрезком аperiодичности. Очевидно, что $r \leq 2^k$, где k — разрядность ЭВМ.

При исследовании программного генератора СЧ всегда необходимо установить длину отрезка аperiодичности. Если число необходимых для экспериментов СЧ меньше отрезка аperiодичности r , то данный генератор может быть использован для моделирования. В противном случае необходимо, строго говоря, использовать другой генератор случайных последовательностей с проверенными статистическими свойствами.

Рассмотрим далее способы моделирования различных случайных факторов на ЭВМ на базе последовательности псевдослучайных квазиравномерных чисел.

13.3.1. Моделирование дискретных случайных величин

Пусть случайная величина X принимает n исходов:

$$X = x_i, \quad i = \overline{1, n} \text{ с вероятностями } P(X = x_i) = p_i, \quad \sum_{i=1}^n p_i = 1 \quad \sum_{i=1}^n p_i = 1.$$

Общий метод моделирования дискретной случайной величины основан на следующем очевидном соотношении:

$$P\left(\sum_{j=1}^{i-1} p_j \leq y \leq \sum_{j=1}^i p_j\right) = p_i, \quad (13.7)$$

где $p_i = P(X = x_i), i = \overline{1, n}$, а y — равномерно распределенная случайная величина в интервале $[0,1]$.

В дальнейшем используется запись $y \in R(0,1)$. Для частного случая $n = 2$ при выполнении условия $y < p_1$, $X = x_1$, в противном случае $X = x_2$, $p_1 + p_2 = 1$.

Наиболее важными дискретными случайными величинами являются целочисленные с распределением

$$p_i = P(X = i), \quad i = 1, 2, \dots,$$

которое можно представить простыми рекуррентными формулами:

$$p_{i+1} = p_i r(i).$$

Для таких случайных величин значение p_i и x_i можно не записывать в памяти ЭВМ, а моделирование осуществлять по схеме, изображенной на рис. 13.6. [12].

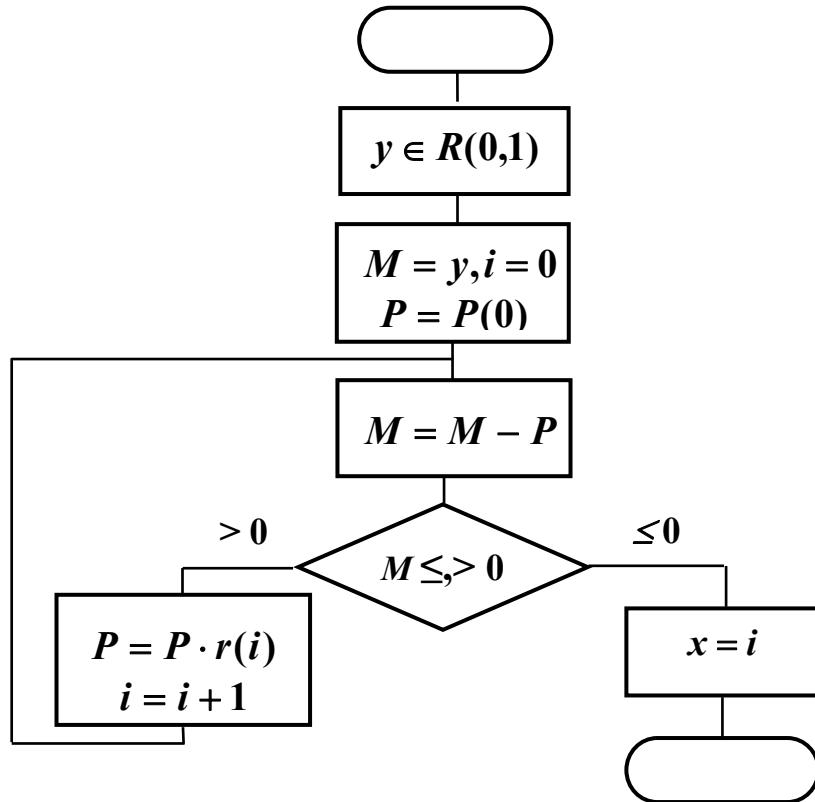


Рис. 13.6

Получим значение $r(i)$ для наиболее характерных распределений дискретных случайных величин:

1. Биномиальное распределение:

$$p_i = P(x = i) = C_n^i p^i (1 - p)^{n-i},$$

$$r(i) = \frac{p_{i+1}}{p_i} = \frac{(n-i)p}{(i+1)(1-p)}, \quad i = 0, 1, \dots, n.$$

2. Распределение Пуассона с параметром λ :

$$p_i = \frac{\lambda^i}{i!} e^{-\lambda}, \quad r(i) = \frac{\lambda}{i+1}, \quad i = 0, 1, 2, \dots$$

3. Геометрическое распределение с параметром p :

$$p_i = p(1-p)^{i-1}, \quad r(i) = 1 - p, \quad i = 0, 1, 2, \dots$$

Численные характеристики случайных величин с рассматриваемыми законами распределения приведены в табл. 13.1.

Таблица 13.1

Название закона распределения		Вид $P(k) / f(x)$	Среднее	Дисперсия
Дискретные	Биномиальный	$C_n^k p^k (1-p)^{n-k}, \quad k = 0, 1, 2, \dots, n$	np	$np(1-p)$
	Пуассона	$\frac{\lambda^k}{k!} e^{-\lambda}$	λ	λ
	Равномерный	$1/n, \quad k = 0, 1, 2, \dots, n$	$(n+1)/2$	$(n^2 - 1)/12$
Непрерывные	Равномерный	$1/(b-a)$	$(a+b)/2$	$(b-a)^2/12$
	Нормальный	$\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$	μ	σ^2
	Экспоненциальный	$\lambda e^{-\lambda x}$	$1/\lambda$	$(1/\lambda)^2$

13.3.2. Моделирование непрерывных скалярных случайных величин

Рассмотрим сначала общие приемы получения случайных чисел с заданным законом распределения из равномерно распределенных случайных чисел.

Метод обратного преобразования

Пусть y — случайная величина, равномерно распределенная на отрезке $[0,1]$, тогда случайная величина x , являющаяся решением уравнения

$$y = \int_{-\infty}^x f(z) dz \quad (13.8)$$

имеет плотность распределения $f(x)$.

Обоснование метода. Если $f(x)$ — плотность распределения, то $y = F(x)$, где $F(x)$ — интегральная функция распределения, монотонно возрастающая и непрерывная функция. Тогда в соответствии с [36]:

$$P[y < F^{-1}(y)] = F[F^{-1}(y)] = y, \quad (13.9)$$

что свидетельствует о равномерном законе распределения y :

$$P[y_1 < y < y_2] = y_2 - y_1,$$

т. е. вероятность попадания случайной величины y в интервал $[y_1, y_2]$ равна длине интервала.

Рассмотрим примеры.

Пример 13.3

Необходимо сгенерировать последовательность случайных чисел, равномерно распределенных на интервале $[a,b]$. Используя (13.8), получим:

$$y_i = \int_a^{x_i} f(z) dz = \int_a^{x_i} \frac{1}{b-a} dz = \frac{x_i - a}{b-a}, \quad (13.10)$$

$$x_i = a + y_i(b - a).$$

Пример 13.4

Необходимо получить последовательность случайных чисел, имеющих экспоненциальный закон распределения:

$$f(x) = \begin{cases} \lambda e^{-\lambda x}, & 0 \leq x < \infty \\ 0, & x \geq 0 \end{cases}. \quad (13.11)$$

В соответствии с (13.8) имеем:

$$y_i = \int_0^{x_i} \lambda e^{-\lambda z} dz = 1 - e^{-\lambda x_i}, \text{ откуда}$$

$$x_i = -\frac{1}{\lambda} \cdot \ln(1 - y_i). \quad (13.12)$$

Так как величина $(1 - y_i)$ так же как и y_i имеет равномерное распределение на интервале $[0,1]$, то формула (13.12) может быть записана другим способом: $x_i = -\frac{1}{\lambda} \cdot \ln y_i$.

Пример 13.5

Смоделировать последовательность случайных чисел, имеющих закон распределения Вейбула:

$$f(x) = \lambda \alpha x^{\alpha-1} e^{-\lambda x^\alpha}, \quad \lambda > 0, \alpha > 0, \quad 0 < x < \infty.$$

В соответствии с (13.8) имеем:

$$y_i = \int_0^{x_i} \lambda \alpha z^{\alpha-1} e^{-\lambda z^\alpha} dz = 1 - e^{-\lambda x_i^\alpha}, \text{ откуда}$$

$$x_i = \left[-\frac{1}{\lambda} \ln(1 - y_i) \right]^{\frac{1}{\alpha}}. \quad (13.13)$$

При $\alpha = 1$ получаем соотношение (13.12).

Пример 13.6

Смоделировать случайную величину с законом распределения Релея с плотностью распределения:

$$f(x) = \frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}, \quad x \geq 0. \quad (13.14)$$

Используя (13.8) получим:

$$y_i = \int_0^{x_i} \frac{z}{\sigma^2} \cdot e^{-\frac{z^2}{2\sigma^2}} dz = 1 - e^{-\frac{x_i^2}{2\sigma^2}},$$

откуда имеем:

$$x_i = \sigma \sqrt{-2 \ln(1 - y_i)} = \sigma \sqrt{-2 \ln y_i}. \quad (13.15)$$

Общий принцип генерации случайных чисел в соответствии с задан-

ным методом включает:

- выборку числа $y_i \in R(0,1)$;
- преобразование в соответствии с соотношением (13.8).

К сожалению, не всегда существуют элементарные преобразования равномерно распределенных чисел для получения случайных величин с заданным законом распределения. В частности, у случайных величин с нормальным распределением функция $F^{-1}(y_i)$ не выражается в явном виде через элементарные функции. В этих случаях для формирования случайных величин с заданным распределением используются различные аппроксимации $F^{-1}(y_i)$.

Использование предельных теорем

Для имитации определенных законов распределения используют предельные теоремы теории вероятностей. Так, например, для получения нормального закона распределения используется свойство асимптотической сходимости сумм независимых случайных величин к нормальному закону распределения.

Если взять n значений чисел из совокупности, равномерно распределенной на интервале $[0,1]$, то при $n = 12$ величину

$$x = \sum_{i=1}^{12} y_i - 6 \quad (13.16)$$

можно считать распределенной поциальному закону с параметрами

$$M[x] = 0, \quad D[x] = 1, \quad \text{т. е. } x = N(0,1), \quad \text{а } f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}.$$

Для получения нормального закона распределения $N(\mu, \sigma)$ с плотностью

$$f(x') = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2} \frac{(x'-\mu)^2}{\sigma^2}}$$

используют линейное преобразование:

$$x' = \mu + \sigma \cdot x.$$

Рассмотрим возможные подходы для генерирования нормального распределения [39].

Некоторые специальные методы моделирования случайных величин

Метод Бокса и Маллера

Согласно этому методу генерируется пара нормированных нормальных чисел x_1 и x_2 , имеющих закон распределения $N(0,1)$ из двух стандартных случайных равномерно распределенных на интервале $[0,1]$ чисел y_1 , y_2 с помощью следующих формул:

$$\begin{aligned} x_1 &= \sqrt{-2 \ln y_1} \sin 2\pi y_2, \\ x_2 &= \sqrt{-2 \ln y_1} \cos 2\pi y_2. \end{aligned} \quad (13.17)$$

Используя преобразование (13.17) в виде:

$$\begin{aligned} y_1 &= \exp \left[-\frac{1}{2} (x_1^2 + x_2^2) \right] \\ y_2 &= \frac{1}{2\pi} \operatorname{arctg} \frac{x_1}{x_2}, \end{aligned} \quad (13.18)$$

а также известные соотношения для получения законов распределения от функций случайных величин [39]:

$$f(x_1, x_2) = f(y_1, y_2) \frac{d(y_1, y_2)}{d(x_1, x_2)}, \quad (13.19)$$

где $\frac{d(y_1, y_2)}{d(x_1, x_2)}$ — якобиан преобразования,

$$f(y_1, y_2) = f(y_1)f(y_2) = 1, \quad y_1, y_2 \in R(0,1),$$

получим, что:

$$f(x_1, x_2) = \frac{1}{2\pi} e^{-\frac{1}{2}(x_1^2 + x_2^2)}. \quad (13.20)$$

Соотношение (13.20) свидетельствует о том, что числа x_1 и x_2 являются независимыми, имеющими закон $N(0,1)$.

Метод Марсальи и Брея

Ими предлагается модификация метода Бокса и Маллера, столь же легкая для программирования и дающая точные результаты, но дающая их быстрее. По этому методу осуществляются следующие действия.

1. Генерируются два случайных числа y_1, y_2 .

2. Проводится их центрирование, т. е. приведение к нулевому математическому ожиданию:

$$v_1 = -1 + 2y_1; \quad v_2 = -1 + 2y_2.$$

3. Вычисляется $s = v_1^2 + v_2^2$, при $s > 1$ цикл начинается снова, т. е. производится возврат к п. 1, в случае $s \leq 1$ получаем пару нормально распределенных чисел:

$$\begin{aligned} x_1 &= v_1 \sqrt{\frac{-2 \ln s}{s}}, \\ x_2 &= v_2 \sqrt{\frac{-2 \ln s}{s}}. \end{aligned} \tag{13.21}$$

Модификация Марсельи-Брея позволяет исключить необходимость вычисления синусов и косинусов, хотя необходимость расчета квадратных корней и натуральных логарифмов сохраняется.

В отличие от соотношения (13.16) соотношения (13.17) и (13.21) являются точными, т. е. дают строго нормальный закон распределения по выбранным равномерно распределенным числам. Кроме того, генератор, построенный в соответствии с соотношением (13.16), имеет ряд недостатков.

1. Для вычисления каждой нормально распределенной величины ему необходимо 12 равномерно распределенных случайных чисел. В силу наличия периода у любого генератора **R(0,1)** при необходимости получения большого количества случайных переменных, имеющихся в распоряжении равномерно распределенных чисел может не хватить.

2. При генерировании “хвостов” нормального распределения этот метод работает плохо. Как следует из [39], за пределами порогов 2σ имеет место сильное расхождение характеристик этого генератора с желаемыми.

Метод Неймана (метод исключения)

Данный метод является достаточно универсальным для моделирования случайных величин, возможные значения которых не выходят за пре-

делы некоторого ограниченного интервала $[a, b]$ (речь идет об усеченных законах распределения). Общий принцип генерации числа с плотностью $f(x)$ распределения вероятностей следующий.

1. Из генератора $R(0,1)$ выбираются два числа y_1, y_2 .

2. Проводятся преобразования

$$u_1 = a + (b - a)y_1, \quad u_2 = f_M \cdot y_2,$$

где $f_M = \max f(x), x \in [a, b]$.

3. В качестве реализации случайной величины x берется число u_1 , удовлетворяющее условию $u_2 < f(u_1)$. Пары чисел y_1, y_2 , не удовлетворяющие данному неравенству, отбрасываются.

Метод суперпозиции

При известных условной плотности распределения $f(x/z)$ и плотности $f(z)$ параметра z плотность распределения $f(x)$ выражается следующим образом:

$$f(x) = \int_{-\infty}^{+\infty} f(x/z) f(z) dz. \quad (13.22)$$

Если параметр z принимает только целые значения

$z_i, i = 1, 2, \dots, n$ с вероятностями p_i , $\sum_{i=1}^n p_i = 1$, то

$$f(x) = \sum_{i=1}^n p_i f_i(x), \quad (13.23)$$

где $p_i = P(z = z_i)$, а $f_i(x) = f(x/z_i)$.

В случае $n = 2$ и нормального закона распределения соотношение (13.23) выглядит следующим образом:

$$f(x) = p_1 \frac{1}{\sigma_1 \sqrt{2\pi}} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} + p_2 \frac{1}{\sigma_2 \sqrt{2\pi}} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}}, \quad (13.24)$$

$$p_1 + p_2 = 1.$$

При генерации распределения (13.24) сначала реализуется выбор дискретной случайной величины в соответствии с соотношением (13.7), а далее с учетом полученного значения z генерируется число с нормальным

законом распределения и параметрами $\mu_i, \sigma_i, i = 1, 2$ согласно соотношениям (13.16), (13.17), (13.21).

Метод кусочной аппроксимации

Среди различных приближенных приемов моделирования случайных величин наиболее простым и достаточно универсальным является метод кусочной аппроксимации, предложенный Н.П. Бусленко [41].

Сущность этого метода состоит в следующем. Пусть требуется получить случайную величину x с функцией плотности $f(x)$. Предположим, что область возможных значений величины x ограничена интервалом $[a, b]$. Разобьем интервал $[a, b]$ на несколько достаточно малых интервалов $[a_i, a_{i+1}]$, $i = \overline{0, n-1}$, $a_0 = a$, $a_n = b$ так, чтобы распределение заданной случайной величины x в пределах этих интервалов можно было достаточно точно аппроксимировать каким-нибудь простым распределением, например, равномерным.

Пусть p_i — вероятность попадания случайной величины в каждый из интервалов $[a_i, a_{i+1}]$. Обычно принимают, $p_i = 1/n$, т. е.

$$p_i = \int_{a_i}^{a_{i+1}} f(x) dx = \frac{1}{n}. \quad (13.25)$$

Исходя из соотношения (13.25), предварительно для заданной $f(x)$ рассчитывают границы интервалов $a_i, i = \overline{0, n-1}$.

В соответствии с выбранной аппроксимацией приходят к следующему способу преобразования равномерно распределенных чисел в случайные числа с заданным законом распределения $f(x)$.

1. Из генератора $R(0,1)$ выбирается пара случайных чисел y_1, y_2 .

2. По первому числу y_1 в соответствии с соотношением (13.7), где значения $p_i = 1/n$, выбирается интервал $[a_i, a_{i+1}]$.

3. По второму числу y_2 рассчитывается величина x :

$$x = a_i + y_2(a_{i+1} - a_i), \text{ имеющая заданную плотность } f(x).$$

Такой алгоритм является довольно экономичным по количеству требуемых операций, которое не зависит от числа n , т. е. не зависит от точности аппроксимации. Однако с увеличением точности аппроксимации возрастает количество ячеек памяти, требуемых для хранения величин a_i ,

$i = \overline{0, n}$, что является недостатком рассмотренного метода, в особенности при больших n .

Перейдем к рассмотрению принципов моделирования векторных случайных величин.

13.3.3. Моделирование случайных векторов по заданным многомерным распределениям

Рассмотрим моделирование непрерывной векторной случайной величины $\mathbf{x}^T = (x_1, x_2, \dots, x_n)$. Ее полное описание задается совместной плотностью распределения $f(\mathbf{x}) = f(x_1, x_2, \dots, x_n)$, где $\mathbf{x} \in R^n$.

Стандартный метод моделирования основан на представлении $f(\mathbf{x})$ в виде произведения:

$$f(\mathbf{x}) = f_1(x_1) f_2(x_2 / x_1) \cdots f_n(x_n / x_1, x_2, \dots, x_{n-1}) \quad (13.26)$$

частной плотности распределения величины x_1 и условных плотностей распределения x_k при известных значениях x_1, x_2, \dots, x_{k-1} ; $k = \overline{2, n}$.

Из формулы (13.26) следует, что вектор \mathbf{x} можно моделировать по-компонентно, используя для моделирования каждого одномерного распределения приемы и методы, рассмотренные в разделе 13.3.2.

Данный подход требует определенной вычислительной работы, связанной с нахождением условных и частных плотностей распределения компонент вектора \mathbf{x} .

С практической точки зрения более приемлемым оказывается получение составляющих случайного вектора в рамках корреляционной теории. Обычно с этих позиций моделируется вектор, имеющий многомерное нормальное распределение $f(\mathbf{x})$:

$$f(\mathbf{x}) = (2\pi)^{-n/2} (\det \mathbf{R})^{-1/2} \exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \mathbf{R}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right], \quad (13.27)$$

где $\boldsymbol{\mu}^T = (\mu_1, \mu_2, \dots, \mu_n)$ — математическое ожидание \mathbf{x} , $\boldsymbol{\mu} = M[\mathbf{x}]$,

$\mathbf{R} = [r_{ij}] = M[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T]$ — заданная ковариационная матрица вектора \mathbf{x} , являющаяся симметрической и положительно определенной матрицей.

Распределение (13.27) задается двумя параметрами: вектором $\boldsymbol{\mu}$ и матрицей \mathbf{R} . Далее используется краткое обозначение: $\mathbf{x} = N(\boldsymbol{\mu}, \mathbf{R})$. Если математическое ожидание является нулевым вектором, а ковариационная матрица $\mathbf{R} = \mathbf{E}$, где \mathbf{E} — единичная матрица, то $\mathbf{x} = N(\mathbf{O}, \mathbf{E})$, т. е. имеем вектор \mathbf{x} с независимыми компонентами. Такое распределение легко мо-

делируется на основе приемов и методов, представленных в разделе 13.3.2, если все компоненты $x_i, i = \overline{1, n}$ положить равными независимым реализациям случайной величины, имеющей закон $N(\theta, 1)$.

В общем случае при $R \neq E$ распределение (13.27) моделируется с помощью линейного преобразования:

$$x = Ay + \mu, \quad y = N(O, E) \quad (13.28)$$

Матрица $A = [a_{ij}]$ n определяется условием:

$$R = AA^T. \quad (13.29)$$

Обычно уравнение (13.29) решается относительно A численно. Наиболее простое решение получается, если считать матрицу A нижней треугольной:

$$A = \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}.$$

В этом случае явный вид коэффициентов a_{ij} определяют следующие уравнения:

$$a_{11} = \sqrt{r_{11}}, \quad a_{k1} = r_{k1} / a_{11}, \quad k = \overline{2, n}; \quad (13.30)$$

$$a_{kl} = \frac{r_{kl} - \sum_{j=1}^{l-1} a_{kj} a_{lj}}{a_{ll}}, \quad l = 2, \dots, (k-1); \quad (13.31)$$

$$a_{kk} = \sqrt{r_{kk} - \sum_{j=1}^{k-1} a_{kj}^2}, \quad k = \overline{1, n}. \quad (13.32)$$

После определения a_{11} вычисление элементов A осуществляется по строкам: сначала по формуле (13.30) вычисляется первый элемент k -й строки a_{k1} , далее по формуле (13.31) находятся последующие элементы $a_{k2}, \dots, a_{k,k-1}$. Диагональный элемент вычисляется по уравнению (13.32). После вычисления диагонального элемента осуществляется переход на следующую строку.

Рассмотрим случай $n = 2$ при $\mu = O$. Соотношение (13.28) выглядит следующим образом:

$$\begin{aligned}x_1 &= a_{11}y_1 \\x_2 &= a_{21}y_1 + a_{22}y_2\end{aligned}\tag{13.33}$$

$$\begin{aligned}M[x_1^2] &= a_{11}^2 = r_{11}; \\M[x_1x_2] &= a_{11}a_{21} = r_{12}; \\M[x_2^2] &= a_{21}^2 + a_{22}^2 = r_{22}. \\a_{11} &= \sqrt{r_{11}}, \quad a_{21} = \frac{r_{12}}{\sqrt{r_{11}}}, \quad a_{22} = \sqrt{r_{22} - r_{12}^2 / r_{11}},\end{aligned}$$

При этом, естественно, предполагается, что величины y_1, y_2 имеют закон $N(0,1)$ и являются некоррелированными.

Общий принцип моделирования в соответствии с распределением (13.27) сводится к следующему.

1. Предварительно в соответствии с соотношением (13.29) вычисляется матрица A по заданной матрице R .
2. Генерируются n чисел y_1, y_2, \dots, y_n , имеющих закон распределения $N(0,1)$.
3. В соответствии с линейным преобразованием (13.28) получаем по вектору y вектор x , имеющий закон распределения (13.27).

При решении задач оптимизации очень часто представляется необходимым выбрать случайное направление в пространстве R^n .

При этом обычно действуют так.

1. Переходят к полярным координатам:

$$\rho, \gamma_1, \gamma_2, \dots, \gamma_{n-1}, \text{ где } \rho = 1, \gamma_i \in [0, 2\pi], i = \overline{0, n-1}.$$

2. Генерируют $n-1$ чисел y_1, y_2, \dots, y_{n-1} , имеющих закон распределения $R(0,1)$.

3. Проводят линейное преобразование

$$\gamma_i = 2\pi y_i, i = \overline{1, n-1}.$$

4. Далее, если это необходимо, переходят к декартовым координатам в соответствии с известными преобразованиями:

$$\begin{aligned} x_1 &= \rho \cos \gamma_1, \\ x_2 &= \rho \sin \gamma_1 \cos \gamma_2, \\ &\dots \\ x_{n-1} &= \rho \sin \gamma_1 \sin \gamma_2 \cdots \cos \gamma_{n-1}, \\ x_n &= \rho \sin \gamma_1 \sin \gamma_2 \cdots \sin \gamma_{n-1} \end{aligned} \tag{13.34}$$

В случае, если $n = 3$, в [39] предлагается оригинальный способ формирования случайных чисел, имеющих равномерное распределение на поверхности сферы единичного радиуса.

Пусть u_i распределены равномерно в интервале $[-1, 1]$, т. е. $u_i = 2y_i - 1$, где $y_i \in R(0,1)$. Выбираем четыре последовательных значения y_1, y_2, y_3, y_4 и проверяем справедливость условия

$$\sum_{i=1}^4 u_i^2 < 1. \quad (13.35)$$

Если это условие выполнено, то случайные числа с координатами:

$$\begin{aligned} x_1 &= \frac{2(u_2 u_4 + u_1 u_3)}{u_1^2 + u_2^2 + u_3^2 + u_4^2} \\ x_2 &= \frac{2(u_3 u_4 - u_1 u_2)}{u_1^2 + u_2^2 + u_3^2 + u_4^2}, \\ x_3 &= \frac{u_1^2 - u_2^2 - u_3^2 + u_4^2}{u_1^2 + u_2^2 + u_3^2 + u_4^2} \end{aligned} \quad (13.36)$$

отображают точки, которые распределены равномерно на поверхности сферы единичного радиуса.

Изложенные выше принципы и приемы моделирования скалярных и векторных случайных величин являются основой моделирования случайных процессов.

13.3.4. Моделирование марковских процессов

Класс марковских случайных процессов является простейшим и наиболее изученным в теории случайных процессов. В настоящее время он весьма широко используется для описания поведения систем управления, вычислительных систем, систем передачи данных при исследовании их производительности, надежности и других свойств систем. В данном разделе рассматриваются только процессы с конечным или счетным множеством состояний.

вом состояний z (значений $X(t)$).

Пусть имеется дискретный случайный процесс с множеством состояний: $z_0, z_1, z_2, \dots, z_i, \dots, z_j, \dots$. Обозначим условную вероятность того, что в момент $t = t_0 + \tau$ процесс будет находиться в состоянии z_j , если в момент t_0 он был в состоянии z_i через $p_{ij}(t_0, \tau)$.

Дискретный случайный процесс называется марковским, если вероятность $p_{ij}(t_0, \tau)$ зависит только от указанных в обозначении параметров i, j, t_0, τ , т. е. от того, в каком состоянии он был в момент t_0 и в какое состояние он должен перейти через время τ .

Далее рассматриваются два основных вида марковских процессов: с дискретным временем (дискретные марковские цепи), с непрерывным временем перехода (непрерывные марковские цепи). В первом случае переходы из одного состояния в другое возможны только в строго определенные моменты времени, во втором случае — в любой момент времени.

Для моделирования марковского процесса должны быть определены:

- начальное состояние процесса или вектор начального состояния;
- граф переходов;
- матрица вероятностей переходов $P(t) = \{p_{ij}(t)\}$ для дискретного времени;
- матрица интенсивностей переходов $\Lambda(t) = \{\lambda_{ij}(t)\}$ для непрерывного времени,

$$\lambda_{ij}(t) = \lim_{\Delta t \rightarrow 0} \frac{p_{ij}(t, t + \Delta t)}{\Delta t}.$$

Далее рассматриваются только однородные марковские процессы $P_{ij}(t) = p_{ij}$ и $\lambda_{ij}(t) = \lambda_{ij}$, т. е. отсутствует зависимость вероятностей и интенсивностей переходов от времени.

Моделирование дискретной марковской последовательности осуществляется по схеме моделирования дискретной случайной величины в соответствии соотношением (13.7).

1. Сначала выбирается случайное число $y_0 \in R(0,1)$ и производится выбор начального состояния z_{m0} , для которого оказывается справедливым условие

$$l_{m0-1}^{(0)} = \sum_{i=1}^{m0-1} p_i(0) \leq y_0 \leq \sum_{i=1}^{m0} p_i(0) = l_{m0}^{(0)},$$

где $p(0)$ — вектор распределения вероятностей начального состояния, $m0$ — номер начального состояния.

2. Затем выбирается следующее случайное число $y_1 \in R(0,1)$, однако в качестве вероятности p_i для определения границ $l_r^{(1)}$ используются элементы строки матрицы перехода $P = [p_{ij}]$ с номером $m0$: $p_{m0,1}, p_{m0,2}, \dots, p_{m0,n}$, где n — размерность вектора состояний процесса. В заключение путем сравнения устанавливается номер $m1$, для которого справедливо условие:

$$l_{m1-1}^{(1)} = \sum_{j=1}^{m1-1} p_{m0,j} \leq y_1 \leq \sum_{j=1}^{m1} p_{m0,j} = l_{m1}^{(1)}.$$

Следовательно, следующим состоянием в данной реализации будет состояние z_{m1} . Таким же образом процесс моделирования повторяется далее.

Рассмотрим, как будет выглядеть процесс моделирования, если переход из состояния в состояние может происходить в любой момент времени.

Один из возможных способов моделирования такого марковского процесса сводится к следующему.

Пусть марковский процесс $z(t)$ удовлетворяет следующему условию:

$$z(t) = z_k, t_k < t < t_{k+1},$$

где $t_0 < t_1 < t_2 < \dots < t_k < t_{k+1} \dots$ — последовательность моментов изменения состояний, начиная от t_0 .

Для определения следующего состояния z_{k+1} и момента t_{k+1} перехода в следующее состояние генерируется ряд независимых случайных величин τ_{z_k, z_j} по экспоненциальному закону распределения с параметром, λ_{z_k, z_j} , где $z_j \in Z(z_k \rightarrow z_j)$:

$$f_{z_k, z_j}(x) = \lambda_{z_k, z_j} \exp(-\lambda_{z_k, z_j} x) \quad (13.37)$$

и определяется $\tau_k = \min_{z_j} \tau_{z_k, z_j}$. Тогда следующий момент изменения состояния: $t_{k+1} = t_k + \tau_k$, а состояние, в которое переходит процесс в данный момент:

$$z(t) = z_{k+1}, \quad t_{k+1} < t < t_{k+2},$$

где z_{k+1} равно тому значению z_j , при котором τ_{z_k, z_j} является минималь-

ным. Для моделирования (13.37) используется (13.12).

Такой способ моделирования марковского процесса называется развернутой рекуррентной имитацией. Другой способ моделирования определяет следующую последовательность действий.

1. Для выделенного состояния z_k по матрице $\Lambda(t) = \{\lambda_{ij}(t)\}$ вычисляется сумма интенсивностей

$$\Lambda = \sum_{z_j \in U(z_k)} \lambda_{z_k, z_j}, \quad (13.38)$$

где $U(z_k)$ — множество индексов состояний, в которые возможен переход из состояния z_k .

2. По выбранному числу $y \in R(0,1)$ генерируется в соответствии с (13.12) число x , где параметр $\Lambda = \sum_{z_j \in U(z_k)} \lambda_{z_k, z_j}$ и вычисляется момент изменения состояния процесса $t_{k+1} = t_k + x$.

3. Состояние, в которое переходит процесс, определяется в результате розыгрыша в соответствии с соотношением (13.7), где

$$P_{z_k, z_j} = \frac{\lambda_{z_k, z_j}}{\sum_{z_j \in U(z_k)} \lambda_{z_k, z_j}}, \quad (13.39)$$

$$\sum_{z_j \in U(z_k)} P_{z_k, z_j} = 1.$$

Указанная процедура, включающая п.п. 1, 2, 3, повторяется для каждого нового состояния процесса.

Этот способ моделирования может быть легко перенесен на класс полумарковских процессов, который задается следующим образом.

1. Для каждого i -го состояния процесса известна функция распределения $F_i(t)$ времени пребывания в данном состоянии.

2. Одновременно определена матрица переходов $P = \{p_{ij}\}$ на выделенном множестве состояний процесса.

Из этого следует, что в общем алгоритме моделирования полумарковского процесса присутствуют два алгоритма:

— первый отвечает за имитацию времени пребывания процесса в i -м

состоянии в соответствии с функцией распределения $F_i(t)$,

– второй — за выбор следующего состояния процесса в соответствии с заданной матрицей P .

При моделировании СМО центральным вопросом является моделирование потоков и законов распределения длительности обслуживания.

13.3.5. Моделирование потоков

Простейший поток

Простейший поток является самой распространенной моделью потоков требований среди однородных стационарных потоков без последействия.

Плотность распределения $f(\tau)$ для интервалов времени между требованиями равна

$$f(\tau) = \lambda e^{-\lambda\tau}, \tau \geq 0, \quad (13.40)$$

где $\lambda = 1/\tau$, τ — средняя длительность интервала между требованиями.

При формировании такого потока последовательность чисел $y \in R(0,1)$ преобразуется в соответствии с соотношением (13.12) в последовательность интервалов $\tau_1, \tau_2, \dots, \tau_i$, имеющую закон распределения (13.11).

Простейший поток имеет пуассоновский закон распределения числа требований на интервале T :

$$P_k(T) = \frac{(\lambda T)^k}{k!} e^{-\lambda T}, \quad k = 0, 1, 2, \dots, \quad (13.41)$$

Поток Эрланга

Потоком Эрланга порядка r называется ординарный стационарный поток с ограниченным последействием, для которого функция плотности распределения длительности интервала между требованиями имеет вид:

$$f_r(\tau) = \frac{\lambda(\lambda\tau)^{r-1}}{(r-1)!} e^{-\lambda\tau}. \quad (13.42)$$

Очевидно, что при $r = 1$ получаем закон распределения (13.11).

При моделировании значения τ с законом распределения (13.42) необходимо генерировать x_1, x_2, \dots, x_r с законом распределения (13.40).

Величина $\tau = \sum_{i=1}^r x_i$ имеет закон распределения (13.42) в соответствии с основными теоремами теории вероятностей: если $z = x + y$ и если плотности распределения величин x и y равны соответственно $f_1(x)$ и $f_2(y)$, то $f(z) = \int_{-\infty}^{+\infty} f_1(x)f_2(z-x) dx$.

Неординарные потоки

Если количество требований, поступающих в момент времени t_i , является случайной величиной, независимой от t_i , то можно предложить следующую процедуру получения возможных значений количества требований. Пусть вероятность того, что в момент времени t_i поступает k требований, равна p_k , $\sum_{(k)} p_k = 1$. Общая схема моделирования такого неординарного потока выглядит так.

1. Генерируется $y_i \in R(0,1)$.
2. Формируется $\tau_i = (-1/\lambda) \ln y_i$.
3. Выбирается число требований $k(\tau_i)$ в соответствии с соотношением (13.7) для заданного распределения k :

$$p_1, p_2, \dots, p_k, \dots, p_n; \quad \sum_{k=1}^n p_k = 1.$$

13.4. Построение оценок и доверительных интервалов характеристик случайных величин

13.4.1. Оценки характеристик случайных величин

В дальнейшем считаем, что при проведении некоторой серии моделирующих экспериментов на ЭВМ их результаты представлены в виде совокупности наблюдений над некоторой случайной величиной, имеющей закон распределения. Будем называть вектор $x = (x_1, x_2, \dots, x_N)$ выборкой объема N из генеральной совокупности с распределением $F(x)$.

Выборочными характеристиками скалярной случайной величины называют обычно функции от выборки, наиболее простые из них — выборочные моменты, являющиеся оценками начальных и центральных моментов. В дальнейшем речь пойдет только об оценках двух основных моментов: математического ожидания и дисперсии случайной величины.

Несмешенные и эффективные оценки указанных характеристик представлены следующими соотношениями:

$$\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i, \quad (13.43)$$

$$S^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2. \quad (13.44)$$

При проведении экспериментов с целью экономии основной памяти ЭВМ, на которой проводится моделирование, оценки \bar{x} и S^2 вычисляют в ходе моделирования путем корректировки результатов оценивания при каждом очередном измерении случайной величины по следующим рекуррентным формулам:

$$\bar{x}_N = \bar{x}_{N-1} \frac{N-1}{N} + \frac{x_N}{N}, \quad (13.45)$$

$$S_N^2 = S_{N-1}^2 \frac{N-2}{N-1} + \frac{1}{N} (x_N - \bar{x}_{N-1})^2. \quad (13.46)$$

Оценку (13.43) обычно характеризуют как меру положения, а оценку (13.44) — как меру рассеяния.

Наряду с этими оценками в качестве меры положения можно использовать оценки медианы и моды, а в качестве меры рассеяния.

1. Среднеквадратическое отклонение S

$$S = \left[\frac{1}{N-1} \sum_{i=1}^N (x_i - \bar{x})^2 \right]^{1/2}. \quad (13.47)$$

2. Среднее отклонение d

$$d = \frac{1}{N} \sum_{i=1}^N |x_i - \bar{x}| \quad (13.48)$$

3. Размах — разность между наибольшим и наименьшим значениями x в выборке:

$$R = x_{\max} - x_{\min} \quad (13.49)$$

Совершенно аналогичным образом выглядит построение выборочных характеристик в многомерном случае, когда наблюдаемая случайная величина x , а вместе с ней и выборочные значения $x_1, x_2, \dots, x_j, \dots, x_n$

есть векторы размерности $\mathbf{n}: \mathbf{x}_k^T = (x_{k1}, x_{k2}, \dots, x_{kj}, \dots, x_{kn})$.

В многомерном случае, как и в одномерном, они представляют собой различные функции от выборки.

Выборочные моменты первого порядка

$$\hat{\mathbf{m}}_{1j} = \frac{1}{N} \sum_{k=1}^N x_{kj}, \quad j = \overline{1, n}. \quad (13.50)$$

Выборочные моменты второго порядка (обычные и центральные)

$$\hat{\mathbf{m}}_{2ij} = \frac{1}{N} \sum_{k=1}^N x_{ki} x_{kj}, \quad i, j = \overline{1, n}, \quad (13.51)$$

$$\hat{\mathbf{M}}_{2ij} = \frac{1}{N} \sum_{k=1}^N (x_{ki} - \mathbf{m}_{1i})(x_{kj} - \mathbf{m}_{1j}) \quad (13.52)$$

Как и в одномерном случае, легко убедиться с помощью закона больших чисел, что эти характеристики с вероятностью 1 сходятся к соответствующим теоретическим моментам, в частности $\hat{\mathbf{M}}_{2ij}$ сходится к

$$\hat{\mathbf{M}}[(x_i - \mathbf{m}_{1i})(x_j - \mathbf{m}_{1j})] = r_{ij}.$$

Нетрудно убедиться, что этим же свойством обладают и выборочные коэффициенты корреляции:

$$\hat{\rho}_{ij} = \frac{r_{ij}}{\sqrt{r_{ii}} \sqrt{r_{jj}}} \rightarrow \rho_{ij}. \quad (13.53)$$

Несмешенные оценки \hat{r}_{ij} , аналогичные скалярным оценкам (13.43) и (13.44), в случае векторной случайной величины рассчитываются следующим образом:

$$\hat{r}_{ij} = \frac{1}{N-1} \sum_{k=1}^N (x_{ki} - \mathbf{m}_{1i})(x_{kj} - \mathbf{m}_{1j}). \quad (13.54)$$

$$\hat{r}_{ij,N} = \frac{N-2}{N-1} \hat{r}_{ij,N-1} + \frac{1}{N} (x_{j,N} - \mathbf{m}_{1i,N-1})(x_{i,N} - \mathbf{m}_{1j,N-1}), \quad (13.55)$$

где $\mathbf{m}_{1i,N-1}$, $\mathbf{m}_{1j,N-1}$ вычисляются в соответствии с соотношениями (13.43) и (13.45).

Перечисленные выше скалярные и векторные выборочные характеристики $\bar{\mathbf{x}}, S^2, \hat{r}_{ij}$ являются также случайными величинами. Относитель-

но рассматриваемых скалярных выборок в дальнейшем предполагается, что они производятся из генеральной совокупности со средним значением μ и среднеквадратическим отклонением σ .

Рассмотрим, как выглядят математическое ожидание и дисперсия для введенных ранее оценок.

Среднее

Каково бы ни было распределение случайной величины x , имеем:

$$M[\bar{x}] = \mu, \quad D[\bar{x}] = M[(\bar{x} - \mu)^2] = \sigma^2 / N. \quad (13.56)$$

Согласно центральной предельной теореме, если N велико ($N > 30$) и x_i некоррелированы, то \bar{x} подчиняется нормальному закону распределения со средним значением μ и среднеквадратическим отклонением σ/\sqrt{N} . Этот результат справедлив для любого распределения x в предположении, что оно обладает конечным средним значением μ и дисперсией σ^2 .

Если x распределен поциальному закону, то высказанное утверждение является строгим при любом N . На практике обычно μ и σ неизвестны и в качестве их оценок используются \bar{x} и S . В этом случае среднеквадратическое отклонение приближенно считают равным S/\sqrt{N} .

Дисперсия

Выборочная дисперсия, вычисляемая по формуле (13.44), является несмещенной оценкой, т. е.

$$M[S^2] = \sigma^2 \quad (13.57)$$

Если выборка $[x_i, i = \overline{1, N}]$ принадлежит нормальному генеральной совокупности $N(\mu, \sigma)$, то дисперсия $D[S^2]$ оценки S^2 равна:

$$D[S^2] = 2\sigma^4 / (N - 1) \quad (13.58)$$

Среднее отклонение

Значение d вычисляется по формуле (13.48). Если значения x_i подчиняются нормальному распределению, то в соответствии с [58]: 

$$M[d] = \sigma \left[\frac{2(N-1)}{\pi N} \right]^{1/2}, \quad (13.59)$$

$$S[d] = \left(1 - \frac{2}{\pi} \right)^{1/2} \frac{\sigma}{\sqrt{N}}. \quad (13.60)$$

Коэффициент корреляции

Коэффициент корреляции вычисляется по формуле (13.53). В случае нормального закона распределения величин (x_1, x_2, \dots, x_n) используется следующая оценка среднего квадратического отклонения коэффициента корреляции:

$$S(\rho_{ij}) \approx (1 - \rho_j^2) \frac{1}{\sqrt{N}}. \quad (13.61)$$

13.4.2. Вычисление доверительных интервалов

В разделе 13.4.1. рассмотрены способы получения точечных оценок различных выборочных характеристик. Точечная оценка без указания степени точности и надежности еще мало определена, так как является лишь частным значением случайной величины. Для того чтобы получить представление о точности и надежности оценки $\hat{\Theta}$ некоторого параметра Θ , мы можем для каждого малого $\alpha > 0$ указать такое ε , что

$$P\left(\left|\Theta - \hat{\Theta}\right| \leq \varepsilon\right) = P\left(\hat{\Theta} - \varepsilon \leq \Theta \leq \hat{\Theta} + \varepsilon\right) = 1 - \alpha \quad (13.62)$$

Чем меньше для данного α будет ε , тем точнее оценка $\hat{\Theta}$.

Соотношение (13.62) говорит о том, что вероятность того, что интервал $[\hat{\Theta} - \varepsilon, \hat{\Theta} + \varepsilon]$ со случайными границами покроет неизвестный па-

раметр Θ , равна $1 - \alpha$. Такой интервал называется доверительным интервалом, $1 - \alpha$ — доверительной вероятностью, α — уровнем значимости.

Рассмотрим теперь построение доверительных интервалов для некоторых рассмотренных выше законов.

Нормальный закон распределения $N(\mu, \sigma)$

В качестве оценки параметра μ выступает \bar{x} . При этом параметр σ считается известным. Тогда для всякого $1 - \alpha$ можно найти такое $t_{\frac{1-\alpha}{2}}$, что

$$P\left(\left|\bar{x} - \mu\right| < t_{\frac{1-\alpha}{2}} \frac{\sigma}{\sqrt{N}}\right) = 1 - \alpha, \quad (13.63)$$

что приводит к следующим доверительным интервалам для параметра μ :

$$\bar{x} - t_{\frac{1-\alpha}{2}} \frac{\sigma}{\sqrt{N}} < \mu < \bar{x} + t_{\frac{1-\alpha}{2}} \frac{\sigma}{\sqrt{N}}, \quad (13.64)$$

где N — объем выборки, а квантиль $t_{\frac{1-\alpha}{2}}$ соответствует условию

$$\frac{1}{\sqrt{2\pi}} \int_0^{t_{1-\alpha}} e^{-\frac{1}{2}z^2} dz = \frac{1-\alpha}{2}. \quad (13.65)$$

и находится по таблицам нормального распределения.

Если значение σ неизвестно, то при больших N с высокой степенью надежности можно заменить его на оценку S и считать интервал

$$\left[\bar{x} - t_{\frac{1-\alpha}{2}} \frac{S}{\sqrt{N}}, \bar{x} + t_{\frac{1-\alpha}{2}} \frac{S}{\sqrt{N}} \right]$$

приближенно доверительным интервалом для μ . Для малых N ($N < 10$) обязательно используют распределение Стьюдента.

Рассматриваемый подход можно применять и для других (ненормальных) распределений случайной величины x , принимая условие асимптотически нормального распределения оценок в больших выборках.

Биномиальное распределение

В соответствии с биномиальным распределением оценивается параметр p . В качестве оценки выступает частота m/N , где

$$m = \sum_{i=1}^N x_i, \quad (13.66)$$

где $x_i = 1$ с вероятностью p и $x_i = 0$ с вероятностью $1 - p$. Тогда с учетом (13.66) имеем:

$$\begin{aligned} M\left(\frac{m}{N}\right) &= p, \\ D\left(\frac{m}{N}\right) &= \frac{p(1-p)}{N}, \\ S &= \sqrt{\frac{p(1-p)}{N}}. \end{aligned} \quad (13.67)$$

В соответствии с (13.62) и (13.63), (13.64) получим следующий интервал¹:

$$\frac{m}{N} - t_{\frac{1-\alpha}{2}} \sqrt{\frac{p(1-p)}{N}} < p < \frac{m}{N} + t_{\frac{1-\alpha}{2}} \sqrt{\frac{p(1-p)}{N}}. \quad (13.68)$$

В случае неизвестного параметра p при больших выборках N параметр p заменяют оценкой m/N .

Такой подход при больших N ($N > 30$) можно использовать не только для оценки среднего и частоты, но и дисперсии и коэффициента корреляции.

Так, например, оценка коэффициента корреляции, вычисленная в соответствии с (13.53) и (13.61) приближенно распределена по закону

$$N \left(\rho_{ij}, \frac{(1 - \rho_{ij}^2)}{\sqrt{N}} \right).$$

Отсюда может быть получен доверительный интервал для коэффициента корреляции:

$$\rho_{ij} - t_{\frac{1-\alpha}{2}} \frac{(1 - \rho_{ij}^2)}{\sqrt{N}} < \rho_{ij} < \rho_{ij} + t_{\frac{1-\alpha}{2}} \frac{(1 - \rho_{ij}^2)}{\sqrt{N}}. \quad (13.69)$$

Формулы (13.64), (13.68), (13.69) могут быть использованы для получения оценки необходимого объема N выборки, гарантирующего требуемые размеры доверительного интервала.

¹ Формула (13.68) предполагает аппроксимацию биномиального распределения нормальным, допустимую при Np и $N(1-p)$, не меньших четырех [61].

1. Для оценки среднего:

$$N \geq \frac{t_{1-\alpha}^2 \sigma^2}{\varepsilon^2}. \quad (13.70)$$

2. Для оценки частоты:

$$N \geq \frac{t_{1-\alpha}^2 (1-p)p}{\varepsilon^2}. \quad (13.71)$$

3. Для оценки коэффициента корреляции:

$$N \geq \frac{t_{1-\alpha}^2 (1-\rho_{ij}^2)^2}{\varepsilon^2}. \quad (13.72)$$

В соотношениях (13.70), (13.71), (13.72) величина ε определяется как статистическая точность. Как видно из этих соотношений, для определения объема имитационных экспериментов необходимо знать значение либо σ , либо p , либо ρ_{ij} , а они очень часто бывают неизвестны. В этом случае либо проводят предварительную «пристрелку» для определения оценок σ , p или ρ_{ij} , либо используют последовательный алгоритм определения необходимого объема.

Например, для оценки среднего алгоритм выглядит следующим образом. Вначале выделяется некоторое число экспериментов ΔN и вычисляется оценка S^2 в соответствии с соотношением (13.44). Если оказывается, что

$$\frac{t_{1-\alpha}^2}{2} \cdot S^2 > \varepsilon,$$

то выделяется следующее дополнительное число прогонов, после которого производится уточнение оценки x . Таким образом, процесс повторяется далее до получения требуемой статистической точности при условии, что число экспериментов не превышает максимально допустимого N_{\max} .

Указанные выше подходы обеспечения статистической точности результатов рассматриваются при естественном предположении асимптотической сходимости оценок параметров к их истинным значениям при не-

ограниченном увеличении объема выборки. Практически же всегда объем выборки ограничен в связи с конечной периодичностью используемых программных генераторов случайных величин. Но даже если предположить наличие идеальных генераторов с бесконечной периодичностью, всегда остается погрешность результатов, обусловленная неполным соответствием имитационной модели моделируемому процессу.

13.5. Сравнительный анализ систем имитационного моделирования

Основные понятия и определения

Система моделирования — совокупность средств, обеспечивающих полный цикл технологического процесса моделирования от построения модели до получения и обработки результатов моделирования.

Основные составляющие системы моделирования

1. Мета-язык и поддерживающие его средства представления моделируемой системы.
2. Язык моделирования — язык построения исходного текста моделирующей программы.
2. Транслятор (интерпретатор или компилятор), обеспечивающий перевод исходного текста в исполняемый код.
3. Программа-диспетчер, обеспечивающая псевдопараллельное исполнение различных фрагментов моделирующей программы.
4. Средства сбора и обработки статистических данных имитационных экспериментов.
5. Средства автоматизации имитационных экспериментов, в том числе средства оптимизации параметров модели.

Некоторые из введенных понятий требуют определенных пояснений.

Следует различать язык моделирования и мета-язык описания объекта моделирования, реализованный в графическом интерфейсе инструментального средства. Многие системы моделирования обеспечивают пользователю возможности визуального представления исследуемого объекта с помощью набора примитивов-шаблонов (*templates*). При этом пользователь, как правило, избавлен от необходимости изучения языка моделирования, так как исходный текст исполняемой программы генерируется автоматически. Для эффективной работы, естественно, требуется определенная ориентация в интерфейсе, которая, впрочем, достаточно быстро приходит с опытом и не требует предварительных знаний.

Термин "псевдопараллелизм" указывает на то, что истинный парал-

лелизм, имеющий место в реальной жизни, воспроизводится в имитационной модели посредством выполнения линейной последовательности действий для данного момента системного времени.

Существует множество разнообразных систем моделирования, поэтому их спектр целесообразно ограничить с учетом следующих соображений.

Исследованию подлежит класс объектов, объединенных понятием "сложные системы". В этот класс входят не только вычислительные системы (сети, системы реального времени, операционные системы и т. п.), но и организационно-производственные системы различного вида и назначения.

Принимается системный (в общепринятой терминологии [50]) уровень детализации при построении модели объекта.

С учетом выбранного класса объектов и уровня детализации рассматриваются системы с дискретными состояниями, приводящие к дискретным событийным моделям (discrete-event models) [51].

Системы моделирования должны характеризоваться приемлемым уровнем доступности при градации от коммерческих до свободно распространяемых (freeware) систем.

Системы моделирования должны быть хорошо документированы, желательно (но не обязательно) на русском языке.

Системы моделирования должны обладать определенной репутацией, складывающейся из таких составляющих, как время существования, широкая распространенность и известность, наличие постоянного сопровождения и поддерживающих организационных структур, наличие новых версий, положительные отзывы экспертов.

Классификация систем имитационного моделирования

Эти требования к системам моделирования довольно сложно оценивать по бинарной шкале, поэтому степень их удовлетворения является существенным, но не единственным критерием их эффективности.

Классификация, как известно, есть выделение классификационных групп в соответствии с принятыми классификационными признаками. В данном случае наиболее существенными являются два признака:

- структурная организация исходного текста моделирующей программы;
- форма представления алгоритма функционирования моделируемого объекта.

Языки моделирования являются подклассом языков программирова-

ния и формируются по тем же правилам. Поэтому в полном соответствии с оформленвшимися к настоящему времени направлениями в программировании [52] системы моделирования можно разделить на системы с процедурными и с объектно-ориентированными языками моделирования. Наиболее важным представителем процедурных языков моделирования является язык GPSS, составляющий основу нескольких версий одноименной системы моделирования. Среди объектно-ориентированных языков и систем моделирования по ряду причин основное внимание в обзоре уделяется системе Anylogic.

По форме представления алгоритма функционирования объекта целесообразно выделить классы систем моделирования, основанных на использовании следующих подходов.

Процессно-ориентированное представление алгоритма функционирования объекта — GPSS.

Математические схемы описания объекта в виде расширений сетей Петри — CPN-Tools и T-Net.

Представление моделей в виде структурных диаграмм и расширенных автоматов в нотации Unified Modeling Language (UML) [53] — Anylogic.

Представление моделей в виде упорядоченной совокупности примитивов различного типа, выбранных из ряда наборов — системы Arena (Arena Professional, Arena Business Edition, BPSim).

Представление моделей в виде структурных диаграмм с использованием IDEF стандартов структурного анализа — All Fusion Process Modeler (BPWin).

Сведения о системах имитационного моделирования

GPSS

Язык GPSS и одноименная система моделирования (общечелевая система имитационного моделирования) разработаны фирмой IBM в конце 60-х годов. Постоянное развитие GPSS, при неизменности базисных принципов его построения, привело к появлению нескольких версий GPSS от двух производителей: Wolverine Software Corporation (GPSS/H) и Minuteman Software (GPSS/PC и GPSS /World). Результаты экспертного оценивания GPSS позволяют сделать вывод о его исключительной гибкости и выразительной мощности. GPSS является одним из наиболее популярных языков моделирования, изучается в большинстве учебных курсов, достаточно полно представлен в литературе [54, 55, 56].

Форма представления моделирующей программы на GPSS

GPSS является процессно-ориентированным языком, поэтому моделирующая программа представляется в форме совокупности сегментов — изолированных описаний параллельных процессов, задающих алгоритм функционирования моделируемого объекта.

GPSS принадлежит семейству языков моделирования транзактного типа. Транзакт исполняет роль динамического объекта (сообщения), продвигающегося по блокам моделирующей программы и инициирующего действия, обусловленные смысловым содержанием блоков. В состав системы моделирования входит интерпретатор, обеспечивающий правильную временную последовательность выполнения действий, связанных с продвижением транзактов. В состав системы GPSS включены программные генераторы случайных величин, средства обработки статистики и создания отчетов. Далее приводятся характерные особенности современных версий GPSS.

GPSS/PC

Разработка середины 80-х годов, предназначена для работы с DOS, требования к оперативной памяти и платформе весьма умеренные и соответствуют возможностям своего времени. Последние версии не ограничивают числа присутствующих в модели транзактов. Имеется пошаговый режим отладки программы, в том числе и с использованием простейшей анимации. Свободно распространяемая версия GPSS/PC доступна по адресу www.minutemansoftware.com.

GPSS World

Это наиболее совершенная и постоянно развивающаяся версия GPSS, созданная компанией Minuteman Software, и функционирующая в операционной системе WINDOWS. Требования к аппаратной части — процессор Pentium III, минимальный объем оперативной памяти 32 Mb.

GPSS World дополняет широко используемую в практике моделирования версию GPSS/PC следующими возможностями.

Размер программы может достигать 100 Mb (20000 строк исходного текста).

Исходный текст формируется с помощью любого редактора и, в отличие от GPSS/PC, не требует фиксированного формата.

Язык моделирования дополнен Programming Language Under Simulation (PLUS), фортраноподобным языком описания процедур, позволяющим составлять подпрограммы различного назначения.

Обеспечена визуализация процесса исполнения моделирующей программы с использованием 10 видов окон, в том числе, окном блоков (BLOCKS WINDOW), средством удобного пошагового мониторинга.

Дополнительная визуализация, реализованная в коммерческой версии, позволяет постоянно отслеживать текущее состояние моделируемого объекта (SNAPSHOTS) на шести разнотипных экранах.

Наличие обширной библиотеки процедур построения генераторов случайных величин, распределенных по разным законам, и развитых средств обработки статистики.

Управление потоками данных (Data Streams) с помощью команд READ, WRITE, SEEK, обеспечивающих удобный импорт/экспорт файлов.

Наличие средств проведения дисперсионного анализа (ANOVA — Analysis of Variation) по результатам имитационных экспериментов.

Возможность пользовательской и автоматической генерации оптимизирующих имитационных экспериментов.

Коммерческая версия полностью совместима с GPSS/PC, относительно свободно распространяемой студенческой версии справедлив вывод о практической, но не гарантированной совместимости с GPSS/PC.

Торговые марки GPSS World и GPSS/PC принадлежат компании Minuteman Software. По адресу www.minutemansoftware.com размещена свободная версия GPSS World Student Version 5.2.1, датированная 2007 г.

GPSS/H

Система моделирования GPSS/H выпущена фирмой Wolverine Software Corporation в 1996 году. От старых версий GPSS ее отличает множество новых положительных свойств и возможностей. Перечислим некоторые из них.

Отсутствие собственной оболочки и выполнение команд из командной строки соответствующей операционной среды (DOS, WINDOWS).

Возможность кодирования модели в свободном формате с комментариями на русском языке.

Наличие ввода и вывода данных с помощью внешних файлов.

Наличие отладчика программ, или дебаггера, что позволяет сократить и сделать более эффективным этап отладки программ.

Наличие фортраноподобных переменных (амперсанд-переменных), которые позволяют значительно упростить многие операции и сделать модель более информативной для наблюдателя и удобной в работе.

Возможность управления форматом и количеством информации в файле отчета, содержащем результаты моделирования.

Системы GPSS/H и GPSS World до недавнего времени можно было считать конкурентами. Достоинством GPSS World является работа в операционной среде WINDOWS, GPSS/H, управляемый из командной строки и лишенный удобного интерфейса, характеризуется вдвое большим быстродействием. Торговая марка GPSS/H принадлежит компании Wolverine Software Corporation. К настоящему времени проект GPSS/H завершил свое развитие.

Arena

Arena — графическая среда визуального имитационного моделирования. Торговая марка Arena принадлежит компании Rockwell Software Inc. Интерфейс системы моделирования позволяет строить модель в виде упорядоченной совокупности графических объектов из наборов шаблонов (*templates*). Графическая модель проходит трансляцию, в результате которой генерируется исполняемый код на языке SIMAN. Семантика и синтаксис языков SIMAN [58] и GPSS чрезвычайно похожи.

Выразительная мощность графического метаязыка описания объектов моделирования обусловлена наличием большого числа наборов шаблонов различной степени интеграции: начиная от объектов-подсистем и кончая объектами-блоками (исполняемыми операторами языка SIMAN). Наиболее развитая версия *Arena* — *Arena Professional* позволяет формировать собственные шаблоны, адаптируя систему моделирования к определенному классу объектов.

Отметим наличие предметно-ориентированных систем *Arena* — *Arena Business Edition* и *BPSim*, предназначенных для моделирования бизнес-процессов.

К числу достоинств *Arena* можно также отнести следующие.

Наличие *Input Analyzer*, средства создания файлов данных, распределенных по заданному закону.

Наличие *Output Analyzer*, средства анализа статистических данных, полученных в результате моделирования.

Наличие мощной системы визуализации, позволяющей организовать полноценную анимацию процесса функционирования исследуемого объекта.

Наличие оптимизатора *OptQuest*, встроенного в систему моделирования и позволяющего находить параметры модели, доставляющие условный экстремум выбранному критерию.

Наличие развитых средств отладки и документирования результатов моделирования.

Возможность экспорта специально подготовленных с использованием средства BPwin моделей, созданных в стандарте IDEF3 (начиная с версии Arena 5).

Arena является удобным средством моделирования как для специалистов в предметной области, неискушенных в технологии и языках моделирования, так и для тех, кто профессионально работает в этой сфере. Для полноценного использования всех возможностей Arena рекомендуется определенное знание языка SIMAN, полезное также и при построении собственных шаблонов.

Создатели Arena обеспечивают постоянное методическое сопровождение своей системы моделирования [57, 58].

T-net

Система моделирования является отечественной разработкой, созданной в Санкт-Петербургском государственном университете аэрокосмического приборостроения (ГУАП) для исследования мультипроцессорных вычислительных систем.

Исходное описание объекта задается в форме F-сети. F-сети специально сконструированы авторами T-net и являются расширением сетей Петри, во многом напоминающим известные E-сети. Система T-net, реализованная на русском языке (единственная из включенных в обзор), предоставляет пользователю следующие возможности.

1. Построение F-сети с помощью многофункционального графического редактора.
2. Связывание подсетей в единую F-сеть с помощью конструктора.
3. Возможность аналитического исследования F-сети: определение позиционных и переходных инвариантов.
4. Визуализация процесса исполнения сети.

Формирование отчета по результатам моделирования.

T-net содержит справочную систему, включающую текстовое описание примеров моделирования и учебные методические материалы по сетям Петри и F-сетям. Методика построения моделей объектов разного класса с использованием F-сетей в доступной форме изложена в учебном пособии [59]. Свободно распространяемая демо-версия системы с интерфейсом на английском языке, не позволяющая сохранять модели, но являющаяся практически полно-функциональной находится по адресу

<http://www.fi.ru/os/tnets.zip>.

CPN Tools

CPN Tools — свободно распространяемое средство моделирования, разработано под руководством Курта Енсена (Kurt Jensen) коллективом специалистов университета г. Ааруса (Дания). Курт Енсен — один из создателей формализма, известного как цветные сети Петри (Coloured Petri Nets (CPN)).

CPN Tools — система моделирования с использованием цветных сетей Петри, основанная создателями системы DESIGN/CPN и практически являющаяся ее развитием на новой основе и в другой операционной среде.

Требования к программному и аппаратному обеспечению состоят в следующем:

- операционные системы: WINDOWS 2000 и WINDOWS XP;
- процессор — Pentium 2 с тактовой частотой не ниже 400 МГц;
- рекомендуемый объем оперативной памяти не ниже 256 Мб и не менее 64 Mb свободного пространства на винчестере;
- обязательное наличие графического акселератора OpenGL.

Являясь преемницей системы DESIGN/CPN тех же авторов, новая система CPN Tools унаследовала не только ее достоинства, но и определенные недостатки, в частности, отсутствие средств моделирования случайных факторов. Пользователи отмечают своеобразный интерфейс, не-привычный пользователям WINDOWS. CPN Tools постоянно развивается, оставаясь так же, как его предшественник, свободно распространяемым программным продуктом. Страница разработчиков пакета размещена по адресу <http://www.daimi.au.dk/CPNTools/>. Система постоянно совершенствуется развиваясь от бэта-версий к версиям с расширяющимися и практически реализованными заявленными функциями.

Modsim III

Modsim III — объектно-ориентированный язык моделирования (ООМ) общего назначения. Синтаксис и структура Modsim III заимствованы из языка Modula-2. Структурные элементы допускают простое и множественное наследование, динамическое связывание объектов, полиморфизм, инкапсуляцию, абстрактные данные и сокрытие информации. Modsim III взаимодействует с аниматором SIMGRAPHICS II. Торговые марки Modsim III и SIMGRAPHICS II принадлежат CACI Products Company.

Modsim III реализует принципы объектно-ориентированного моде-

лирования, в соответствии с которыми модель представляет собой собрание взаимодействующих компонентов (объектов). Программный модуль объекта описывает его поведение или, в терминологии ООМ, методы. Характеристики объекта содержатся в его полях (или переменных). Все "знания" объекта сосредоточены в его переменных; все, что объект в состоянии "делать" описано в его методах. Взаимодействие объектов реализуется ими посредством обмена сообщениями (message-passing communication). В состав системы моделирования входят библиотечные модули и, в частности, средства сбора, обработки статистики и создания отчетов.

Anylogic

Система Anylogic основана на использовании трех концепций описания функционирования сложных систем: моделей системной динамики Форрестола, методов дискретно-событийного моделирования и методов агентного моделирования. Вначале появилась система COVERS, представлявшая собой интерактивную графическую среду ввода, редактирования, отладки и анализа моделей параллельных систем реального времени. Входным языком являлся язык взаимодействующих автоматов, включенный во множество описательных средств универсального языка моделирования (UML). Последняя версия, COVERS 3.1a, датирована 1998 годом. Торговые марки COVERS и Anylogic принадлежат компании XJ Technologies. Anylogic наследует ту же концепцию моделирования, что и COVERS.

AnyLogic включает набор примитивов и библиотечных объектов для эффективного моделирования систем широкого назначения. Предлагаемый AnyLogic объектно-ориентированный подход облегчает итеративное поэтапное построение больших моделей.

Процесс моделирования разбивается на три этапа:

- построение и редактирование модели,
- компиляция — генерация исполняемого Java-кода,
- пошаговое или автоматическое выполнение модели с возможностью анимации структурной диаграммы.

Исходная модель формируется в соответствии с принципами объектно-ориентированного программирования: ключевыми элементами модели являются активные объекты, допускающие инкапсуляцию, и обменивающиеся сообщениями (messages). Поведение объекта задается расширенным конечным автоматом (statechart).

Имеется возможность моделирования непрерывных и гибридных (дискретно-непрерывных) систем.

Встроенная библиотека обеспечивает выполнение функций общего назначения (работа со строками, файловый ввод/вывод и т. п.) и специаль-

ные (стандартные распределения, сбор статистики).

В зависимости от комплектации приобретенного пакета пользователь получает доступ к библиотекам, существенно упрощающим процесс построения моделей в различных предметных областях.

Практическое использование Anylogic для создания серьезных моделей требует определенной квалификации в области программирования на Java.

Информация о системе Anylogic размещена на сайте компании XJ Technologies по адресу <http://www.xjtek.com>.

BPWin

BPWin — средство описания и документирования процессов функционирования систем широкого назначения². BPWin — CASE-средство построения информационных систем и, строго говоря, не является системой моделирования. Тем не менее, это средство включено в обзор по следующим причинам.

Начало многоэтапного процесса имитационного моделирования предполагает построение концептуальной модели, в которой заданы основные элементы исследуемой системы и определены способы их взаимодействия.

Концептуальная модель формируется на начальном этапе формализации и представляет собой структурированное содержательное описание системы. Это описание еще остается доступным специалистам в предметной области, знающим объект исследования, и уже может служить информационной основой имитационной модели для специалистов в области моделирования.

Адекватность имитационной модели в значительной степени обусловлена правильным выбором концептуальной модели, поэтому инструментальные средства, позволяющие упорядочить и облегчить процесс ее построения, заслуживают соответствующего внимания. BPWin, как раз, и относится к таким средствам, занимая среди них лидирующее положение.

BPWin — средство описания процессов в стандартах IDEF0 и IDEF3, являющихся воплощением и развитием идей структурного анализа и проектирования (Structured Analysis and Design Technique (SADT)).

² Здесь используется широко известное старое название программного продукта, его новое название — All Fusion Process Modeler. Торговые марки BPWin и All Fusion Process Modeler принадлежат компании Computers Associates International, Inc.

BPWin обеспечивает автоматический перевод BPWin-моделей, созданных в стандарте IDEF3, в имитационные модели, упоминавшейся выше системы Arena. Последние версии системы BPWin обеспечивают экспорт моделей в формат XML, что существенно расширяет возможности по использованию сформированных в BPWin IDEF3-описаний для генерации кода имитационной программы в различных средах моделирования, например, GPSS World или Anylogic.

Современные тенденции развития средств моделирования

Имитационное моделирование сложных организационно-технических комплексов интегрируется в современную технологию построения информационных систем. Это объясняется сходством задач создания информационных систем и имитационных моделей. В обеих задачах присутствует этап формирования адекватных описаний объектов исследования. Поэтому имеет место взаимное проникновение методов и средств из одной смежной области в другую.

Поиски универсальных методов описания сложных систем привели к появлению Универсального Языка Моделирования (Unified Modeling Language (UML)) — средства формирования интегрированного описания систем [53]. UML создавался как высокоуровневый язык визуализации, специфирования, конструирования и документирования сложных программных продуктов.

В настоящее время UML активно используется в технологии программирования, приобретая репутацию незаменимого инструмента квалифицированного специалиста в этой области. Это обстоятельство можно считать решающим для начавшегося внедрения UML в технологию моделирования. Отражением этого факта является использование элементов нотации UML в исходных описаниях моделей, потребляемых современным системами моделирования COVERS и Anylogic.

Язык UML является идеологической основой мощного инструментального средства разработки и моделирования информационных систем ARIS³ (Architecture of Integrated Information Systems — Архитектура Интегрированных Информационных Систем). В ARIS используется несколько видов формализованных описаний систем (process models, function assignment diagrams, event diagrams, organizational charts), которые автоматически трансформируются в имитационную модель.

Отметим еще один пример из того же ряда — мощное инструмен-

³ ARIS – торговая марка, принадлежащая компании IDS Scheer AG.

тальное средство Popkin Software Architect⁴ создания совокупности формализованных представлений сложных объектов (в том числе имитационных моделей) в рамках задачи построения информационной системы.

Другое универсальное описание — совокупность стандартов IDEF — реализовано в упоминавшемся выше BPWin — средстве, согласованном с системой имитационного моделирования ARENA.

Таким образом, одной из явных тенденций развития современных систем моделирования сложных объектов является внедрение универсальных стандартов описания систем, таких, как UML и IDEF, сочетаемое с разработкой средств, обеспечивающих автоматический переход к исполняемой имитационной модели.

⁴ Popkin Software Architect – торговая марка, принадлежащая компании Popkin Software & Systems Inc.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Зайченко Ю.П. Исследование операций. – Киев: Высш. школа, 1979. – 392 с.
2. Банди Б. Основы линейного программирования Пер. с англ. – М.: Радио и связь, 1989. – 176 с.
3. Вентцель Е.С. Исследование операций. – М.: Сов. радио, 1972. – 552 с.
4. Ашманов С.А., Тимохов А.В. Теория оптимизации в задачах и упражнениях – М.: Наука, 1991. – 448 с.
5. Химмельблау Д. Прикладное нелинейное программирование / Пер. с англ. – М.: Мир, 1975. – 534 с.
6. Реклейтис Г., Ревиндрен А., Рэгсдел К. Оптимизация в технике. В 2-х кн. / Пер. с англ. – М.: Мир, 1986. – 320 с.
7. Кузнецов Ю.Н., Кузубов В.Н., Волощенко А.В. Математическое программирование: Учеб. пособие. – М.: Высш. школа, 1980. – 300 с.
8. Базара М., Шетти К. Нелинейное программирование. Теория и алгоритмы / Пер. с англ. – М.: Мир, 1982. – 583 с.
9. Козлов В.Н., Колесников Д.Н. Решение задач автоматики и вычислительной техники методами исследования операций: Учеб. пособие. – Л.: ЛПИ, 1982. – 84 с.
10. Карманов В.Г. Математическое программирование: Учеб. пособие. 3-е изд., перераб. и доп. – М.: Наука. 1986. – 288 с.
11. Моисеев Н.Н. Математические задачи системного анализа. – М.: Наука, 1981. – 487 с.
12. Волкова В.Н., Денисов А.А. Основы теории систем и системного анализа. – СПб.: СПбГТУ, 1997. – 509 с.
13. Дегтярев Ю.И. Исследование операций. – М.: Высшая школа, 1986. – 319 с.
14. Бондаренко Н.И. Методология системного подхода к решению проблем. – СПб.: Издательство Петербургского университета экономики и финансов. 1997. – 383 с.
15. Колесников Д.Н. Душутина Е.В., Пахомова В.И. Введение в MATLAB с примерами решения задач оптимизации и моделирования: Учеб. пособие. – СПб.: Издательство СПбГТУ, 1995. – 113 с.
16. Конев В.Ю., Мироновский А.А. Основные функции пакета MATLAB. СПб.: ГААП, 1992. – 74 с.
17. Потемкин В.Г. MATLAB: Справочное пособие. – М.: Далог–МИФИ, 1998. – 348 с.

18. Потемкин В.Г. MATLAB-5 для студентов. – М.: Далог–МИФИ, 1998. –314 с.
19. Глущенко В.В., Сахаров В.В. Сумеркин Ю.В. Моделирование динамических систем и электрических цепей в среде MATLAB: Учеб. пособие. – СПб.: Издательство СПбГУ водных коммуникаций. 1998. – 293 с.
20. Дьяконов В.П. Справочник по применению системы MATLAB. – М.: Наука, 1993. – 127 с.
21. Коробова Н.Л., Загошвили Ю.В. Комплекс автоматизированного проектирования. MATLAB-CTRL: Учеб. пособие. – СПб.: Издательство Балтийского ГТУ. 1993. – 67 с.
22. Афоничкин А.И. Принятие управленческих решений в экономических системах. Учебное пособие. – Саранск.: Изд-ство Мордовского университета. 1998. – 183 с.
23. Черноруцкий И.Г. Методы оптимизации и принятия решений. Учебное пособие. – СПб.: 2001 г. – 382 с.
24. Макаров И.М. и др. Теория выбора и принятия решений. – М. Наука. 1982 г. – 325 с.
25. Садовский Л.Е., Садовский А.Л. Математика и спорт. – М.: Наука. 1985. – 190 с.
26. Дехтяренко В.А., Своятыцкий Д.А. Методы многокритериальной оптимизации сложных систем при проектировании. – Киев.: техника. 1976.
27. Черчмен У., Акофф Р. Введение в исследование операций. – М.: Мир. 1969. – 486 с.
28. Нейман Дж., Моргенштерн О. Теория игр и экономического поведения. – М.: Наука. 1970.
29. Клейнрок. Л. Теория массового обслуживания. – М.: Машиностроение, 1979. – 431 с.
30. Гнedenko Б.В., Коваленко И.Н. Введение в теорию массового обслуживания. – М.: Наука, 1987. – 336 с.
31. Клейнрок Л. Вычислительные системы с очередями. – М.: Мир, 1979. – 598 с.
32. Кемени Дж., Снелл Дж. Конечные цепи Маркова. – М.: Наука, 1970. – 270 с.
33. Кузовлев В. И., Шкотов П.Н. Математические методы анализа производительности и надежности САПР. – М.: Высшая школа, 1990. – 143с.
34. Липаев В.В. Колин К.К., Серебровский Л.А. Математическое обеспечение управляющих УВП. – М.: Советское радио, 1972. –527с.
35. Основы теории вычислительных систем./Под ред. С.А. Майорова/ – М.: Высшая школа. 1978. – 407 с.

36. Алиев Т.И. Математические методы теории вычислительных систем. Учебное пособие. – Л.: ЛИТМО., 1979. – 91с.
37. Решение задач автоматики и вычислительной техники методами теории массового обслуживания. Учебное пособие. / Под ред. Д.Н. Колесникова./ – Л.: ЛПИ, 1987. – 77 с.
38. Варжапетян А.Г., Глушенко В.В. Системы управления. – М.: Вызовская книга, 2000. – 126 с.
39. Быков В.В. Цифровое управление в статистической радиотехнике. – М.: Советское радио. 1971. – 324 с.
40. Колесников Д.Н., Сиднев А.Г., Юрганов А.А. Моделирование случайных факторов в задачах автоматики и вычислительной техники. Учебное пособие. Санкт Петербург. – Л.: СПбГТУ. 1994.– 104 с.
41. Бусленко Н.П. Моделирование сложных систем. – М. Наука. 1977. – 204 с.
42. Худсон Д. Статистика для физиков. – М.: Мир. 1967. – 242 с.
43. Денисов А.А., Колесников Д.Н. Теория больших систем управления. – Л.: Энергоиздат. 1982. – 286 с.
44. Дурандин К.П., Ефремов В.Д., Колесников Д.Н. и др. Моделирование сложных систем с использованием сетей массового обслуживания. – Л.: ЛПИ. 1981 – 83 с.
45. Советов Б.Я., Яковлев С.А. Моделирование систем. – М.: Высшая школа. 1998. – 319 с.
46. Шенон Р. Имитационное моделирование систем. – Искусство и наука. – М.: Мир. 1978. – 418 с.
47. Жожикашвили В.А., Вишневский В.М. Сети массового обслуживания. Теория и применение к сетям ЭВМ. М.: Радио и связь. 1988. – 191 с.
48. Шварц М. Сети связи: протоколы, моделирование и анализ: в 2-х ч. Ч.1: Пер. с англ. – М.: Наука, 1992, – 336 с.
49. Тимоти Паркер. TCP/IP. Освой самостоятельно. Пер. с англ. – М.: БИНОМ, 1997, – 448 с.
50. Под ред. Брейера М. Автоматизация проектирования вычислительных систем. Языки моделирования и базы данных. – М.: Мир. 1979. – 464 с.
51. Jerry Banks, John S Carson Barry L. Nelson Discrete-Event System Simulation. 2nd ed. Prentice-Hall, 1996. – 548 р.
52. Скляров В.А. Язык C++ и объектно-ориентированное программирование. –Мн.: Высш. Шк., 1997. – 478 с.
53. Буч Г., Рамбо Д., Джекобсон А. Язык UML. Руководство пользователя: Пер. с англ. – М.: ДМК, 2000. – 432 с.

54. Шрайбер Т. Дж. Моделирование на GPSS. – М.: – Машиностроение, 1980.
55. Власов Л.В., Колесников Д.Н., Сорокин И.А. Имитационное моделирование систем массового обслуживания с использованием GPSS: Учебное пособие. – СПб.: ЛПИ, 1989. – 88 с.
56. Власов Л.В., Черносвитов Р. Исследование систем массового обслуживания. Язык моделирования GPSS: Учебное пособие. – СПб.: СпбГТУ, 1996. – 110 с.
57. Kelton W.D., Sadovski D.A. Simulation with Arena. – McGraw-Hill, 1998. – 547 p.
58. Pegden C.D., Shannon R.E., Sadovski R.P. Introduction to Simulation Using SIMAN. – 2nd ed., McGraw-Hill, 1995.
59. Гордеев А.В., Молчанов А.Ю. Применение сетей Петри для анализа вычислительных процессов и проектирования вычислительных систем.: Учеб. Пособие. СПб.: СПбГААП, 1993. – 75 с.
60. Маклаков С.В. BPwin и Erwin. CASE-средства разработки информационных систем. – М.: ДИАЛОГ-МИФИ, 1999. – 256 с.
61. Вентцель Е.С. Теория вероятностей. М.: Наука. 1969. – 576 с.

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ 1

УЧЕБНЫЕ ЗАДАНИЯ ПО ЛИНЕЙНОМУ ПРОГРАММИРОВАНИЮ

Содержанием задания является анализ и решение задачи линейного программирования, заданной в следующей форме:

$$\max\{C_1x_1 + C_2x_2\},$$

$$\begin{cases} a_{11}x_1 + a_{12}x_2 \leq b_1, \\ a_{21}x_1 + a_{22}x_2 \leq b_2, \\ x_1, x_2 \geq 0. \end{cases}$$

Значения параметров задачи в зависимости от номера варианта приведены в табл. П. 1.

Задание является комплексным и предполагает выполнение следующих разделов.

1. Геометрическая интерпретация задачи и ее графическое решение.
2. Обозначение опорных точек и соответствующих им наборов базисных переменных.
3. Решение задачи симплекс-методом в табличной форме.
4. Решение задачи симплекс-методом в матричной форме.
5. Введение дополнительного ограничения, отсекающего оптимальную точку. Решение новой задачи двойственным симплекс-методом в табличной форме.
6. Формулировка задачи, двойственной по отношению к исходной.
7. Определение координат сопряженных опорных точек прямой и двойственной задач. Нахождение оптимального решения двойственной задачи по оптимальному решению прямой задачи.

Выполненное задание должно содержать графическое изображение области допустимых решений и траекторию поиска в пространстве R^2 для прямой и двойственной задач (см. рис. 2.3, 2.4, 2.5), а также симплекс-таблицы для каждой опорной точки траектории. Примеры выполнения соответствующих пунктов задания приводятся в разделе 2.6.

Таблица П. 1

	C_1	C_2	a_{11}	a_{12}	a_{21}	a_{22}	b_1	b_2
1	1,0	2,0	1,0	1,0	-1,0	1,0	6,3	3,2
2	1,0	1,5	1,0	0,5	-1,0	0,5	10,1	5,3
3	2,0	1,0	1,0	1,0	1,0	-1,0	5,6	2,4
4	1,0	1,0	1,0	2,0	-1,0	2,0	5,3	2,8
5	1,0	3,0	0,5	1,0	-1,0	2,0	6,7	0,7
6	1,0	-3,0	1,0	1,0	-1,0	1,0	4,8	1,3
7	1,0	-2,0	1,0	0,3	-1,0	0,4	10,2	1,2
8	2,0	3,0	1,0	1,0	-3,0	-1,0	4,8	-3,5
9	2,0	3,0	1,0	0,8	-3,5	-1,0	6,3	-4,7
10	1,5	2,0	1,0	1,2	-2,7	-0,9	7,3	-1,5
11	1,0	2,0	1,0	1,4	-3,5	-1,0	4,8	-8,7
12	1,3	4,2	0,6	1,5	-2,0	-1,0	6,7	-8,3
13	2,0	-3,0	0,7	1,2	-2,1	-0,8	6,3	-9,7
14	1,0	-3,0	1,0	1,0	-2,5	-1,0	5,8	-12,3
15	1,0	2,0	1,0	1,0	1,0	-1,0	5,7	1,6
16	1,0	1,0	0,8	1,0	1,0	-0,7	4,9	2,1
17	1,0	1,5	1,2	1,0	1,3	0,8	7,3	0,6
18	1,0	3,0	2,0	1,0	1,0	-1,0	14,2	1,7
19	1,2	2,8	1,6	0,9	1,2	-1,1	13,2	1,6
20	1,5	4,5	3,8	1,7	2,4	-2,1	25,2	3,7
21	1,0	-2,0	1,0	1,0	-1,0	-4,0	4,6	-7,3
22	1,0	-2,0	1,0	0,5	-1,0	-5,0	4,9	-8,3
23	1,0	-1,0	0,8	0,6	-1,2	-3,8	3,7	-10,8
24	1,0	-1,5	1,2	1,7	-0,7	-3,2	5,8	-7,2

ПРИЛОЖЕНИЕ 2

УЧЕБНЫЕ ЗАДАНИЯ ПО НЕЛИНЕЙНОМУ ПРОГРАММИРОВАНИЮ

Задача нелинейного программирования задана в следующей форме:

$$\begin{aligned} \max \{f(X) = C_{11}x_1^2 + C_{22}x_2^2 + C_{12}x_1x_2 + C_1x_1 + C_2x_2\} \\ \left\{ \begin{array}{l} a_{11}x_1 + a_{12}x_2 \leq b_1, \\ a_{21}x_1 + a_{22}x_2 \leq b_2, \\ a_{31}x_1 + a_{32}x_2 \leq b_3, \\ a_{41}x_1 + a_{42}x_2 \leq b_4, \\ a_{51}x_1 + a_{52}x_2 \leq b_5, \\ d_{11}x_1^2 + d_{22}x_2^2 = b_6, \\ d_{33}x_1^2 + d_{44}x_2^2 \leq b_7. \end{array} \right. \end{aligned}$$

Значения параметров задачи в зависимости от номера варианта приведены в табл. П2.

Задание состоит из двух частей:

- решение задачи безусловной оптимизации (информация об ограничениях не используется);
- решение задачи условной оптимизации при известном безусловном экстремуме.

Безусловная оптимизация целевой функции

Содержанием задания является поиск безусловного максимума целевой функции следующими методами:

- наискорейшего подъема;
- Ньютона;
- сопряженных градиентов;
- релаксационный;
- переменной метрики Бройдена;
- переменной метрики Дэвидона-Флетчера-Пауэлла.

В одну из итераций каждого метода необходимо включить процедуру одномерного поиска длины шага.

Задания должны содержать графическое изображение линий равного уровня целевой функции и траекторию поиска в пространстве. Каждая итерация метода должна быть подробно описана с приведением промежуточных результатов. Примеры выполнения и оформления заданий приводятся в разделе 3.6.

Условная оптимизация целевой функции

Содержанием задания является поиск условного максимума целевой функции следующими методами:

- Лагранжа;
- Била;
- Баранкина-Дофмана;
- проекции градиента;
- возможных направлений;
- штрафных функций;
- барьерных функций.

Оформление второй части задания предполагает нанесение границ допустимой области на построенный ранее график линий равного уровня целевой функции и изображение траектории поиска ее условного максимума. Итерации метода должны быть описаны и прокомментированы подобно тому, как это сделано в примерах, приведенных в четвертом разделе.

Таблица П. 3 (начало)

$N_{\text{зад}}$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
C_{11}	-6	-16	-7	-13	-21	-31	-22	-39	-16	-50	-39	-32	-8	-14	-17
C_{22}	-9	-19	-13	-22	-24	-34	-28	-51	-34	-65	-66	-53	-17	-26	-23
C_{12}	4	4	8	12	4	4	8	16	24	20	36	28	12	16	8
C_1	20	80	10	30	110	286	172	294	6	380	174	152	28	84	182
C_2	60	140	80	40	180	388	296	532	288	680	612	496	154	252	266
a_{11}	-1	-1	-1	2	2	7	7	7	7	9	9	7	10	9	
a_{12}	1	1	1	3	1	12	12	12	12	4	4	12	8	7	
a_{21}	1	1	1	2	1	10	9	10	-1	0	1	1	-1	-1	-1
a_{22}	1	1	0	1	1	8	7	8	1	1	1	1	1	1	1
a_{31}	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
a_{32}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a_{41}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a_{42}	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
a_{51}	1	1	1	3	0	0	0	0	9	-1	2	1	0	0	0
a_{52}	0	0	1	2	1	1	1	1	7	1	1	0	1	1	1
b_1	4	4	4	6	6	84	84	84	84	84	36	36	84	84	63
b_2	6	5	2	4	5	80	63	80	2	6	5	6	4	4	4
b_3	0	0	0	0	0	0	0	0	0	0	2	3	0	0	0
b_4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
b_5	2	2	6	6	3	5	6	5	63	3	6	2	5	5	4
d_{11}	1	4	1	1	3	16	9	4	3	25	9	12	49	49	9
d_{22}	9	25	4	9	12	25	16	9	12	49	4	3	25	16	25
d_{33}	9	25	4	3	9	25	16	9	12	16	9	3	4	9	4
d_{44}	1	4	9	12	25	16	9	4	3	25	25	12	9	25	81
b_6	36	100	16	9	108	400	144	144	108	1225	144	108	1225	784	225
b_7	36	100	36	108	225	400	144	144	108	400	225	108	144	225	324

Таблица П. 3 (окончание)

$N_{зад}$	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
C_{11}	-3	-7	-4	-7	-9	-13	-10	-18	-10	-23	-21	-17	-5	-8	-8
C_{22}	-3	-7	-4	-7	-9	-13	-10	-18	-10	-23	-21	-17	-5	-8	-8
C_{12}	-2	2	4	6	2	2	4	8	12	10	18	14	6	8	4
C_1	10	34	8	18	46	118	76	132	28	170	102	86	28	56	84
C_2	18	50	20	38	66	146	100	176	60	226	162	134	28	56	84
a_{11}	-1	-1	9	0	-1	7	7	7	7	9	9	7	10	9	
a_{12}	1	1	4	1	1	12	12	12	12	12	4	4	12	8	7
a_{21}	2	2	1	2	1	10	9	10	-1	0	1	1	-1	-1	-1
a_{22}	1	1	1	1	1	8	7	8	1	1	1	1	1	1	1
a_{31}	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
a_{32}	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0
a_{41}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
a_{42}	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
a_{51}	0	0	1	-1	1	0	0	0	9	-1	2	1	0	0	0
a_{52}	1	1	1	1	0	1	1	1	7	1	1	0	1	1	1
b_1	3	2	36	3	4	84	84	84	84	84	36	36	84	80	63
b_2	6	6	6	6	6	80	63	80	2	6	5	6	4	4	4
b_3	0	0	0	0	0	0	0	0	0	0	2	3	0	0	0
b_4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
b_5	3	3	5	2	2	5	6	5	63	3	6	2	5	5	4
d_{11}	9	9	4	49	9	16	9	4	3	25	9	12	49	49	9
d_{22}	25	16	9	36	25	25	16	9	12	49	4	3	25	16	25
d_{33}	16	16	9	9	16	25	16	9	12	16	9	3	4	9	4
d_{44}	9	9	25	16	9	16	9	4	3	25	25	12	9	25	81
b_6	225	144	144	900	225	400	144	144	108	1225	144	108	1225	784	225
b_7	144	144	225	144	144	400	144	144	108	400	225	108	144	225	324

ПРИЛОЖЕНИЕ 3

ПОСТРОЕНИЕ МАТРИЦ A_i, B_i, C_i ДЛЯ РЕГУЛЯРНОГО МАРКОВСКОГО ГРАФА СОСТОЯНИЙ ТИПА «РЕШЕТКА»

Введем векторные состояния $P_i^T = (p_{i0}, p_{i1}, \dots, p_{in})$, $i = \overline{0, m}$, объединяющие скалярные состояния $(i0, i1, \dots, in)$ регулярного марковского графа типа «решетка», представленного на рис. П.3.1.

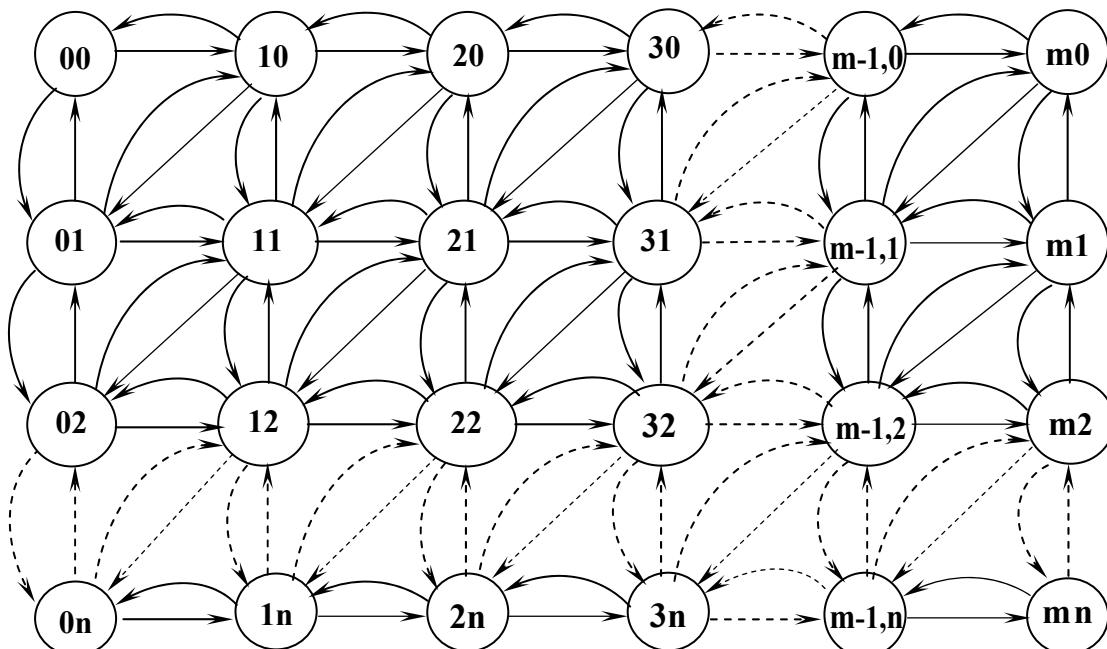


Рис. П. 3.1

Матрицы A_i, B_i, C_i объединяют соответствующие интенсивности переходов между состояниями марковского графа.

Матрицы A_i объединяют интенсивности ухода из векторного состояния i в соседние векторные состояния $(i-1)$ и $(i+1)$, а также интенсивности переходов в пределах скалярных состояний векторного состояния i .

Матрицы B_i объединяют интенсивности ухода из векторного состояния i в соседнее векторное состояние $(i+1)$.

Матрицы C_i объединяют интенсивности ухода из векторного состояния i в соседнее векторное состояние $(i-1)$.

Свойства матриц A_i, B_i, C_i поясняются рис. П.3.1, П.3.2, П.3.3.

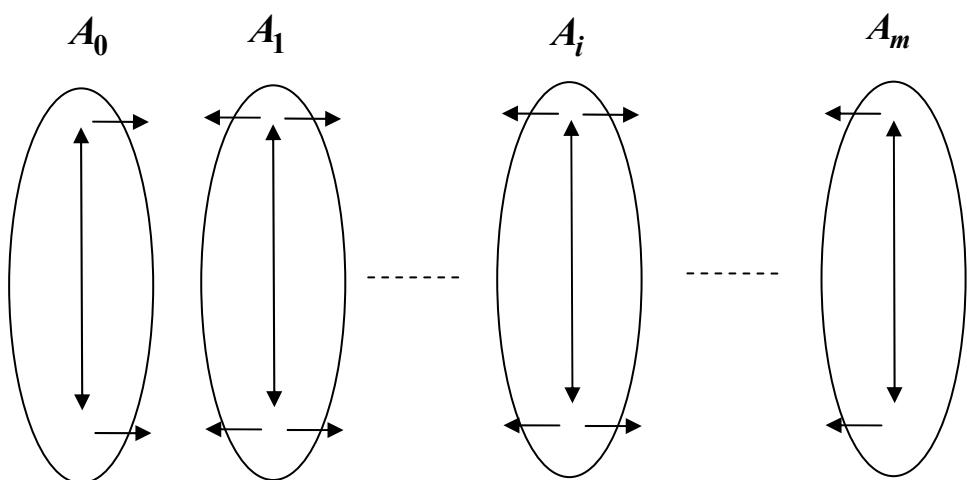


Рис. П.3.2

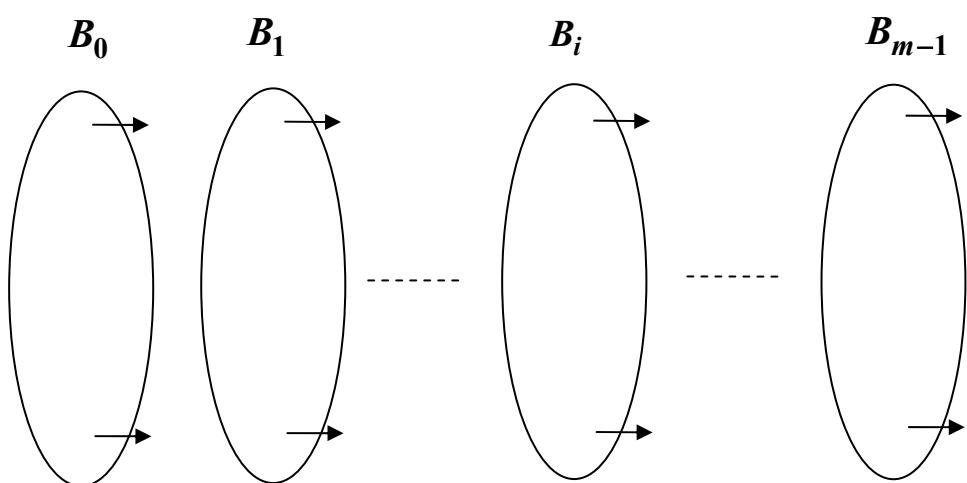


Рис. П.3.3

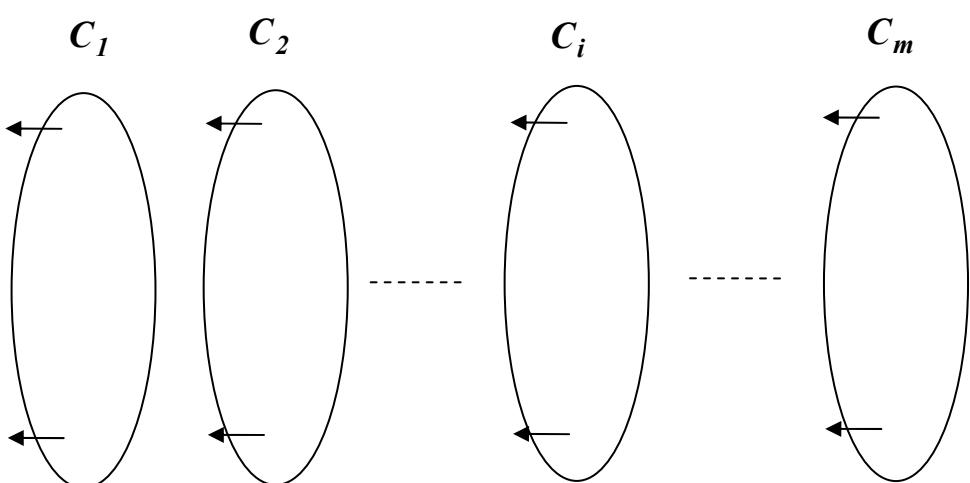


Рис. П.3.4

Рис. П.3.5 дает представление о способах формирования матриц A_i , B_i , C_i на примерах матриц A_0 , B_0 , C_1 .

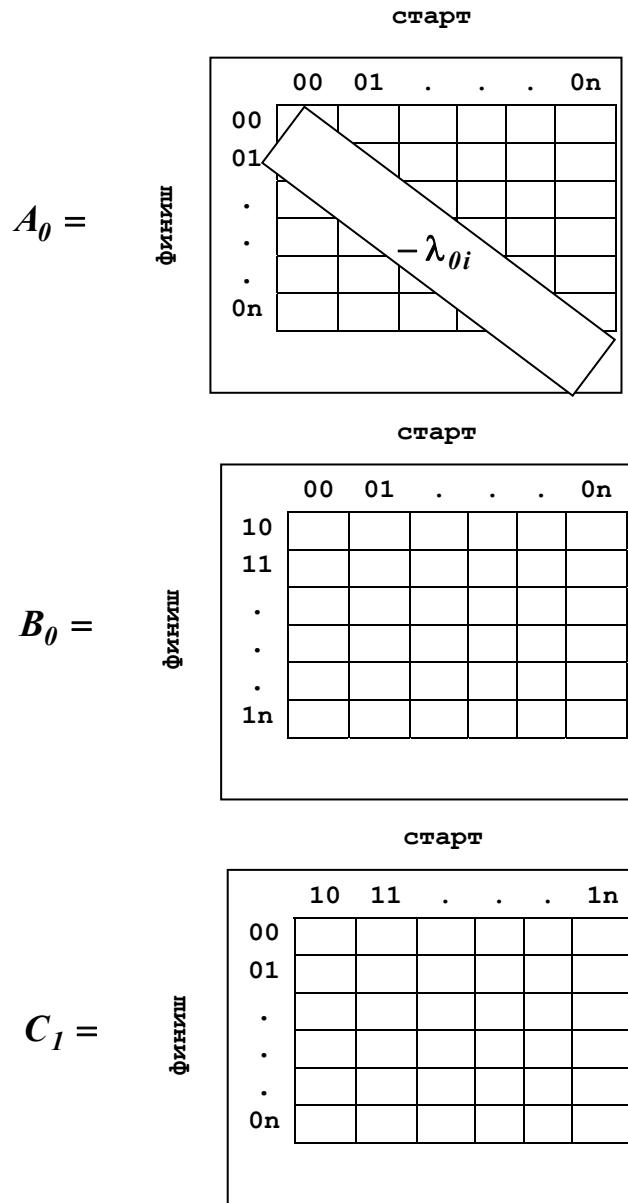


Рис. П.3.5

Диагональные элементы матрицы A_0 равны значениям интенсивностей ухода из соответствующих состояний, взятым со знаком « $-$ ». Отсутствующим на графе дугам соответствуют нулевые элементы матрицы. Остальные элементы матрицы A_0 равны интенсивностям перехода из состояния «старт» в состояние «финиш».

Элементы матриц B_0 и C_1 равны интенсивностям перехода из состояния «старт» в состояние «финиш». Отсутствующим дугам также соответствуют нулевые элементы матриц.

Для графа, представленного на рис. 9.4 и продублированного в данном приложении (рис. П.3.6), в соответствии с предложенной схемой получены следующие матрицы A_0, B_0, C_1 .

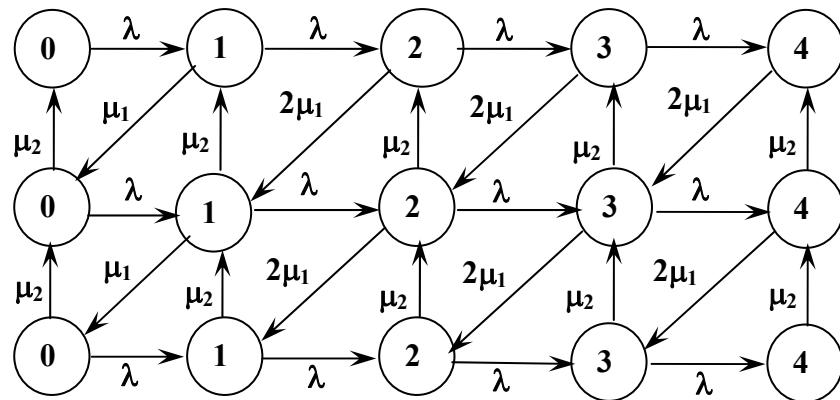


Рис. П.3.6

$$A_0 = \begin{Bmatrix} 00 & \overbrace{\begin{matrix} 00 \\ -\lambda \end{matrix}}^0 & \overbrace{\begin{matrix} 01 \\ \mu_2 \end{matrix}}^0 & \overbrace{\begin{matrix} 02 \\ 0 \end{matrix}}^0 \\ 01 & 0 & -(\lambda + \mu_2) & \mu_2 \\ 02 & 0 & 0 & -(\lambda + \mu_2) \end{Bmatrix},$$

$$B_0 = \begin{Bmatrix} 10 & \overbrace{\begin{matrix} 00 \\ \lambda \end{matrix}}^0 & \overbrace{\begin{matrix} 01 \\ 0 \end{matrix}}^0 & \overbrace{\begin{matrix} 02 \\ 0 \end{matrix}}^0 \\ 11 & 0 & \lambda & 0 \\ 12 & 0 & 0 & \lambda \end{Bmatrix}, \quad C_1 = \begin{Bmatrix} 00 & \overbrace{\begin{matrix} 10 \\ 0 \end{matrix}}^0 & \overbrace{\begin{matrix} 11 \\ 0 \end{matrix}}^0 & \overbrace{\begin{matrix} 12 \\ 0 \end{matrix}}^0 \\ 01 & \mu_1 & 0 & 0 \\ 02 & 0 & \mu_1 & 0 \end{Bmatrix}.$$