И. Г. Черноруцкий

$$\lim_{k\to\infty} u^k = u^*$$

МЕТОДЫ ПРИНЯТИЯ РЕШЕНИЙ

- Введение в теорию обоснованного выбора вариантов из множества допустимых
- Практические алгоритмы принятия решений в сложных ситуациях при неполной информации
- Методы рационального поведения руководителей организаций и системных аналитиков при принятии решений
- Алгоритмические методы скалярной оптимизации
- Проблема жесткости и плохой обусловленности в задачах компьютерной оптимизации
- Экспертные системы принятия решений

$$f_k(y) \to \max_{y \in Y}$$

$$x^r = x^c + \alpha(x^c - x^h),\ \alpha > 0$$

УЧЕБНОЕ ПОСОБИЕ

bhv®

И. Г. Черноруцкий

МЕТОДЫ ПРИНЯТИЯ РЕШЕНИЙ

Рекомендовано учебно-методическим объединением вузов по университетскому политехническому образованию в качестве учебного пособия для студентов высших учебных заведений, обучающихся по направлению подготовки бакалавров и магистров 553000 "Системный анализ и управление"

Санкт-Петербург
«БХВ-Петербург»
2005

УДК 681.3.06(075.8)
ББК 32.973я73
Ч-49

Черноруцкий И. Г.

Ч-49 Методы принятия решений. — СПб.: БХВ-Петербург, 2005. — 416 с.: ил.

ISBN 5-94157-481-9

В учебнике рассматриваются классические задачи принятия решений, формулируемые как задачи выбора вариантов из допустимого множества. В частности, рассматриваются задачи конечномерной оптимизации. Дается введение в экспертные системы принятия решений, что позволит разработать свою собственную экспертную систему. Основное внимание уделено прикладным и вычислительным аспектам принятия решений и оптимизации, связанным с разработкой компьютерных алгоритмов и вопросами их практического применения.

Для студентов и преподавателей вузов,
специалистов в области информационных технологий
и компьютерного моделирования

УДК 681.3.06(075.8)
ББК 32.973я73

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Людмила Еремеевская*
Зав. редакцией	*Григорий Добин*
Редактор	*Татьяна Темкина*
Компьютерная верстка	*Натальи Смирновой*
Корректор	*Наталия Першакова*
Дизайн обложки	*Игоря Цырульникова*
Зав. производством	*Николай Тверских*

Рецензенты:
Козлов В. Н., д. т. н., профессор, зав. кафедрой "Системный анализ и управление" Санкт-Петербургского политехнического университета,
Фомин Б. Ф., д. т. н., профессор Санкт-Петербургского электротехнического университета.

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 22.11.04.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 33,54.
Тираж 3000 экз. Заказ № 3702.
"БХВ-Петербург", 190005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар № 77.99.02.953.Д.001537.03.02 от 13.03.2002 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 5-94157-481-9

Содержание

Предисловие

В книге описываются методы теории принятия решений, составляющей важнейший раздел системного анализа. Термин "системный анализ" понимается здесь как совокупность методов, основанных на использовании компьютерных технологий и ориентированных на исследование сложных систем — технических, экономических, экологических, программных и т. д. Результатом этих исследований, как правило, является выбор определенной альтернативы: плана развития фирмы, параметров конструкции, стратегии управления проектом и т. п. Таким образом, системный анализ согласно принятой в данной книге интерпретации — это дисциплина, занимающаяся проблемами принятия решений в условиях, когда выбор альтернативы требует анализа сложной информации, характеризующей реальную ситуацию. С другой стороны, известный термин "исследование операций" часто трактуется как дисциплина, занимающаяся количественным обоснованием решений в различных областях целенаправленной человеческой деятельности. Поэтому можно сказать также, что наша книга посвящена исследованию операций. В некоторых публикациях делаются попытки провести грань между теорией принятия решений и исследованием операций, основываясь на концепции многокритериальной оптимизации. Именно: предлагается считать, что теория принятия решений начинается там, где возникает проблема выбора по многим критериям. Автор не является сторонником такого подхода и следует терминологии, используемой, например, в работе [8].

Основная направленность данного издания — пользовательский, прикладной аспект. Пользовательский аспект в данном случае понимается несколько шире, чем это обычно принято. Речь пойдет не только и не столько о пользовательском аспекте по отношению к каким-то программным продуктам, пакетам и системам поддержки принятия решений (СППР), но, в основном, о пользовательском аспекте в смысле внутреннего функционального наполнения подобных программных продуктов, о реализованных в них базовых принципах и применяемой в данной области терминологии. Приводимые сведения помогут будущему пользователю "вскрывать" используемые СППР, по крайней мере, в общих чертах и представлять, каких результатов следует ожидать от конкретной СППР и какие она дать не сможет.

Ссылки на литературу внутри текста ограничены, однако в конце книги приведен полный список использованной литературы. Автор не претен-

дует на новизну и оригинальность приводимого материала, хотя некоторые подходы и методы кажутся ему новыми. Книга имеет учебный и вводный характер и поэтому при заимствовании соответствующих разделов из ранее изданных книг и статей автор стремился по мере возможностей передать дух и пафос авторского изложения — разумеется, с необходимыми в подобных случаях согласованиями обозначений, сокращениями, дополнениями и комментариями учебного характера, а иногда и критическими замечаниями с позиций практической значимости предлагаемых алгоритмов.

Книга состоит из четырех частей.

В *части I* изложены основы теории выбора вариантов из заданного множества альтернатив при различных типах неопределенностей. Рассмотрены задачи выбора в условиях неопределенности "среды" (принятие решений в условиях риска, в условиях полной неопределенности, в игровых ситуациях выбора), а также задачи выбора решений в условиях неопределенности цели — многокритериальные задачи выбора. Изложение сопровождается модельными примерами, позволяющими легко оценивать ситуации на интуитивном уровне и облегчающими усвоение материала. Часто и сами базовые методы излагаются не в общем виде, а на примерах решения модельных задач с использованием соответствующих смысловых интерпретаций. Как показывает опыт обучения, а также опыт практической работы, после уяснения основных идей и алгоритмических конструкций у достаточно квалифицированного читателя уже не возникает проблем при представлении изученного материала "в общем виде". Все основные подходы и методы рассмотрены с алгоритмических позиций, позволяющих оценить в первую очередь практическую значимость обсуждаемого материала. Сложные математические модели в первой части почти не используются, что позволяет предъявлять минимальные требования к предварительной подготовке читателей.

В *части II* представлены методы поиска минимума (или максимума) вещественной функции от вещественных переменных. Такие задачи также связаны с проблемой выбора "наилучших" вариантов и поэтому, по мнению автора, могут изучаться в контексте теории принятия решений. Изложены вопросы взаимосвязи различных моделей выбора и, в частности, показано, что множество максимизаторов или минимизаторов целевого функционала в однокритериальных задачах оптимизации совпадает с ядром соответствующего бинарного отношения предпочтений лица, принимающего решение. Даны также элементы теории конечномерной оптимизации, играющей важнейшую роль при решении задач компьютерного моделирования. Представленные здесь методы и алгоритмы с позиций общей проблемы выбора или принятия решений трактуются как "тривиальные" задачи выбора, т. к. традиционные "неопределенности"

отсутствуют (имеем детерминистские однокритериальные задачи выбора). Необходимость в методах оптимизации возникает непосредственно, когда какого-то результата можно достичь различными способами, описываемыми конечными наборами вещественных чисел, и основная задача заключается в поиске наилучшего в каком-либо смысле варианта достижения цели. Кроме того, как показано в *части I*, общие задачи принятия решений при "раскрытии" соответствующих неопределенностей также сводятся в алгоритмическом плане к некоторой цепочке однокритериальных детерминистских задач. Таким образом, можно сказать, что теория конечномерной оптимизации является алгоритмическим фундаментом общей теории выбора.

Есть много публикаций, посвященных проблемам конечномерной оптимизации, а также программных разработок, пакетов и систем оптимизации. В частности, широкую и заслуженную известность получил пакет NAG (разрабатывается и поддерживается Numerical Analysis Group), где реализованы достаточно мощные и эффективные алгоритмы конечномерной оптимизации. Однако основную роль в развитии теории и методов оптимизации играли и играют в первую очередь математики. Поэтому в литературе, как отечественной, так и зарубежной, наиболее полно представлены лишь те вопросы теории (и практики!), где можно было получить математически завершенные результаты. Последнее заставляло исследователей ограничиваться наиболее удобными с математической точки зрения структурами и формализациями, связанными, в частности, с концепциями выпуклости, линейности и т. д. С другой стороны, практика реального компьютерного моделирования и численного эксперимента в различных областях часто выдвигает не обладающие указанными свойствами задачи, при решении которых традиционными методами возникают значительные вычислительные трудности.

Если условно назвать исследователей, занимающихся реальным компьютерным моделированием, инженерами, то уместно вспомнить известный тезис: "Математики делают то, что можно, так, как нужно, а инженеры — то, что нужно, так, как можно". Во второй части книги мы рассматриваем "то, что нужно" — вопросы, оказывающиеся существенными при проведении реальных компьютерных вычислений. Наряду с важнейшими из широко известных алгоритмических средств здесь представлены некоторые нетрадиционные вычислительные технологии, оказывающиеся, на наш взгляд, достаточно полезными во многих практических ситуациях.

Для чтения основного материала *части II* требуются знания в области численного анализа, линейной алгебры, теории матриц и отдельных разделов курса математики, читаемых в технических и экономических вузах. Читатели, уже знакомые со стандартными методами конечномерной оп-

тимизации, смогут найти некоторые нетрадиционные оценки этих методов и возможные альтернативные подходы к решению практических задач.

Часть III посвящена введению в экспертные системы принятия решений. В ней затронуты вопросы, связанные с выбором решений в трудно формализуемых областях, где стандартные математические технологии моделирования оказываются практически неприменимыми. Таким образом, в отличие от рассмотренных ранее технологий моделирования и принятия решений, экспертные системы имеют дело со слабо структурированными задачами, в которых, в частности, трудно ожидать наличия достоверных численных оценок различных вариантов. Вообще, под "экспертной системой" здесь понимается программная диалоговая система, в которую включены знания специалистов о некоторой конкретной проблемной области и которая в этой обычно достаточно узкой и специализированной области способна принимать обоснованные решения, заменяя высококвалифицированных экспертов.

Часть IV содержит пользовательское описание двух работающих систем поддержки принятия решений, разработанных под руководством автора и основанных на изложенных в книге концепциях. В первой системе реализована оригинальная технология многокритериального выбора вариантов на основе метода *t*-упорядочения. Вторая система является инструментальным средством для построения пользовательских экспертных систем и разработана с применением нейлоровской концепции построения диагностирующих байесовских экспертных систем. Указанные системы были разработаны, соответственно, А. В. Рассохиным и Ю. Г. Ефремовым в качестве магистерских диссертаций при их обучении на кафедре "Информационные и управляющие системы" факультета технической кибернетики Санкт-Петербургского государственного политехнического университета (бывший Политехнический институт им. Петра Великого). Более подробную информацию об этих системах и условиях доступа к ним можно найти на сайте университета (**www.spbstu.ru**).

Книга предназначена для различных категорий читателей. Во-первых, это студенты вузов и других учебных заведений, изучающие дисциплины, связанные с современными информационными технологиями и компьютерным моделированием. Во-вторых, это уже дипломированные специалисты, желающие оценить возможности компьютерной поддержки для решения внутренних проблем на своем рабочем месте. Наконец, это современные руководители, стремящиеся применить в своей работе достижения из указанной области. В частности, знание основных результатов и принципов теории принятия решений и оптимизации позволит не только лично руководствоваться ими, но и ставить обоснованные задачи своему системному аналитику или отделу системного анализа фирмы.

В настоящее время можно указать большое число предметных областей и практических ситуаций, когда выбор решения может и должен основываться на излагаемых в данной книге методах и технологиях. В частности, теория выбора и принятия решений, а также теория оптимизации могут быть использованы в таких областях, как:

- задачи выбора оптимальной номенклатуры товара в торговых и иных организациях;
- задачи выбора персонала в фирме (например, при приеме на работу);
- задачи рациональной организации разработки программного обеспечения для компьютерных систем;
- задачи, решаемые в риэлтерских фирмах, оказывающих услуги населению на рынке недвижимости (например, подбор квартир);
- задачи оптимального выбора параметров (числовых характеристик) какой-либо системы (или организации) — пректируемой или реально существующей;
- формирование оптимальных стратегий поведения на рынке ценных бумаг;
- задачи принятия решений на финансовом рынке в условиях риска и неопределенности;
- задачи реализации оптимальных стратегий замены оборудования;
- задачи максимизации доходов в условиях аукционных торгов и т. д.

Количество соответствующих примеров может быть существенно увеличено.

Введение

Как показывает практика (и далее это продемонстрировано на реальных примерах), широко распространенное мнение о том, что достаточно иметь хорошее программное обеспечение (ПО) из соответствующей области (а оно обычно есть), чтобы с успехом приступать к решению практических задач, оказывается принципиально неверным. В простейших случаях (например "проблемы", решаемые бухгалтерами) трудностей может и не быть, но в таких алгоритмически сложных областях, как принятие решений, управление, системное проектирование и т. д., ситуация совершенно иная.

Наличие хорошего ПО в соответствующей организации или фирме и хороших аппаратных средств — это лишь необходимое, но не достаточное условие. Кроме этого, совершенно обязательной является высокая профессиональная подготовка лица, принимающего решение (ЛПР). Это не обязательно глава фирмы, им может быть специальный человек (так называемый системный аналитик) или группа лиц — отдел системного анализа. Сказанное относится не только к области принятия решений, но и к другим областям компьютерного моделирования, требующим привлечения нетривиальных математических моделей, на которых основана любая современная информационная технология.

Вот характерный пример, иллюстрирующий справедливость сказанного. (Критикуется не самая свежая публикация, однако подобные казусы регулярно встречаются в практической работе как следствие пренебрежения продекларированными здесь простыми принципами.)

Приводимая задача и ее решение взяты из книги [35, оригинал вышел в 1971 году в США; авторы — профессора Коннектикутского и Иллинойского университетов соответственно; перевод под ред. Я. З. Цыпкина].

Рассматривается управляемая система второго порядка с одним управлением. Система описывается следующими разностными уравнениями состояния:

$$y_1(i+1) - y_1(i) - T_{i+1}(-y_1^2(i) + y_2(i) + u(i)) = 0;$$
$$y_2(i+1) - y_2(i) - T_{i+1}y_1(i) = 0, \quad i = 0, 1, ..., N-1,$$

где $y_j(i)$ — j-я компонента вектора состояния в дискретный момент t_i; $T_i = t_i - t_{i-1}$.

Задача заключается в выборе такой сеточной функции $u(i)$, чтобы перейти из заданного начального состояния

$$y(0) = (0,1)$$

в целевую область

$$[y_1(N) - 10]^2 - y_2^2(N) - 1 \le 0$$

при минимуме показателя качества

$$J = \sum_{i=1}^{N} \left[y_1^2(i) + y_2^2(i) + 0{,}1\, u^2(i-1) \right]$$

и выполнении ограничений:

$$0 \le u(i) \le 1,\ i = 0, 1, \ldots, N-1;$$

$$y_j(i) \ge 0,\ i = 1, \ldots, N;\ j = 1, 2.$$

При $N = 12$ (число периодов дискретизации) сформулированная задача естественным образом представляется как стандартная хорошо изученная задача нелинейного программирования с функционалом J, 48 переменными и 37 условиями. Здесь искомыми считаются все величины $y_j(i)$, $u(i)$, T_i. Для решения подобных задач нелинейного программирования разработаны "эффективные" методы и реализующие их программные системы оптимизации. Остается только воспользоваться ими, что авторы книги и проделали. Они использовали хорошо зарекомендовавшие себя метод и программу последовательной минимизации без ограничений, разработанные известными американскими авторами А. Фиакко (Anthony V. Fiacco), Г. Мак-Кормиком (Garth P. McCormick) (метод штрафных функций). Полученные результаты представлены в виде таблицы и графика, которые должны убедить читателя в эффективности применяемого подхода. Приведем фрагмент полученной таблицы (табл. В.1).

Таблица В.1

i	T_i(с)	t_i(с)	$u(i-1)$	$y_1(i)$	$y_2(i)$
1	0,00291	0,00291	0,0553	0,00265	0,964
2	0,00196	0,00487	0,5000	0,00375	0,930
3	0,00176	0,00663	0,5000	0,00434	0,898
4	0,00169	0,00832	0,0466	0,00467	0,867
5	0,00164	0,00996	0,0435	0,00479	0,839

Таблица В.1 (окончание)

i	T_i(с)	t_i(с)	$u(i-1)$	$y_1(i)$	$y_2(i)$
6	0,00162	0,01158	0,0419	0,00473	0,812
7	0,00159	0,01317	0,0406	0,00480	0,786
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
12	13,18224	13,20211	0,0339	9,26978	0,683

Далее авторы книги занимаются анализом и обсуждением результатов, совершенно не замечая (вместе с переводчиком и редактором перевода на русский язык), что задача фактически не решена и получен случайный набор чисел. Действительно, согласно разностным уравнениям системы, переменная $y_2(i)$ должна монотонно возрастать:

$$y_2(i+1) - y_2(i) = T_{i+1}y_1(i) > 0.$$

Однако в таблице $y_2(i)$ монотонно убывает, т. е. в полученных результатах не удалось отразить даже качественные характеристики решения. При этом использовались передовая по тем временам вычислительная техника (IBM-7094) и прекрасное программное обеспечение.

Книга содержит целый ряд других неверно решенных задач. Излишне говорить, что реализация на практике рекомендаций, полученных в результате подобных "исследований", может иметь крайне нежелательные, если не катастрофические последствия. Ведь рассматриваемые математические модели могут относиться к потенциально опасным реальным системам. Достаточно сказать, что в данной книге рассматриваются модели управления процессом отравления ксеноном в ядерных реакторах, управления ракетным ядерным реактором и т. п. Излишне также говорить, что примененные авторами обсуждаемой книги методы и технологии являются абсолютно корректными и с их помощью были успешно решены и решаются в настоящее время многие задачи компьютерного моделирования. Речь идет не о корректности применяемых методов, а о корректности их применения.

Таким образом, видимая тривиальность вычислительных задач моделирования вообще и задач принятия решений в частности, а также наличие хорошо развитого современного программного обеспечения не дают оснований отказываться от привлечения к соответствующей деятельности хорошо подготовленных и квалифицированных системных аналитиков.

Данная книга и предназначена в первую очередь для начинающих системных аналитиков в области систем оптимизации и принятия решений.

Чтобы наглядно очертить круг задач, с разной степенью подробности затрагиваемых в данной книге, рассмотрим несколько максимально упрощенных примеров из различных областей человеческой деятельности, которые можно трактовать как задачи принятия решений.

При этом под задачей *принятия решений* мы будем понимать задачу *выбора* наилучшего способа действия из некоторого множества допустимых вариантов. Сформулируем точнее.

Задано множество вариантов X (конечное или бесконечное). Выбор какого-либо из вариантов $x_i \in X$ приводит к некоторому исходу $y_j \in Y$, где Y — множество возможных исходов. Требуется выбрать такой вариант x_i, чтобы получить наиболее благоприятный в определенном смысле исход y_j. Множество вариантов X часто называют *множеством альтернатив*, хотя это определение противоречит канонам русского языка — альтернатив может быть только две. Мы также будем использовать термин "альтернатива" в указанном смысле.

Пример В.1. Чтобы попасть из пункта A (остановка автобуса) в пункт B (лодочная станция) (рис. В.1), человек должен пройти сначала по асфальтовой дороге (отрезок Ax), а затем по песчаному пляжу (отрезок xB). Известны скорости передвижения по асфальтовой дороге и по песку. Спрашивается, в каком месте нужно свернуть с асфальтовой дороги, чтобы затратить меньше времени на весь путь.

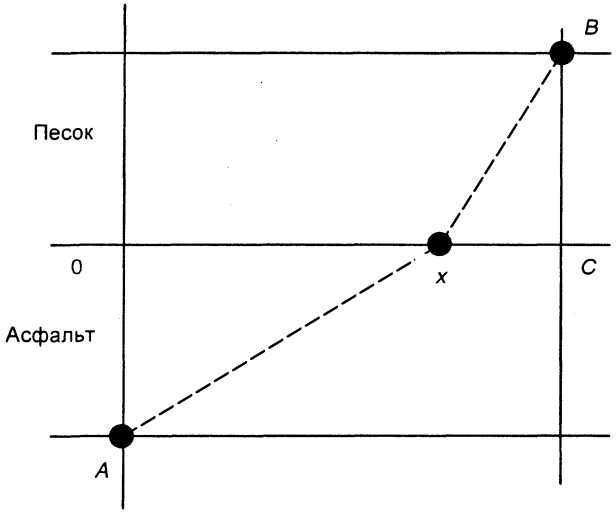


Рис. В.1. Выбор оптимального пути

Сформулированную задачу можно рассматривать как задачу принятия решения: множество альтернатив состоит из множества точек прямой OC, т. е. из множества вещественных чисел x. Каждому решению соответствует исход, или результат, — маршрут AxB. Таким образом, имеем задачу *принятия решения в условиях определенности*. Каждый исход (т. е. маршрут) оценивается числом — временем передвижения по маршруту.

Пример В.2. Предположим, что при разработке некоторой логической электронной схемы нас кроме функциональных требований интересуют два показателя: потребляемая схемой мощность (f_1) и время задержки распространения сигнала (f_2), причем мы хотим минимизировать оба эти показателя. Мы можем варьировать параметры (номиналы) части резистивных элементов схемы $R_1, \ldots, R_L$ в некоторых заданных границах. При этом каждому фиксированному набору $R = (R_1, \ldots, R_L)$ этих параметров соответствуют определенные значения f_1 (потребляемая мощность) и f_2 (время задержки). Таким образом, взяв за альтернативы наборы значений R, а затем в качестве исходов — соответствующие им пары чисел (f_1, f_2), приходим к задаче выбора решения в условиях определенности. Изобразив все возможные пары чисел (f_1, f_2) на плоскости, получим некоторую область F, каждая точка которой представляет собой один из возможных исходов (рис. В.2).

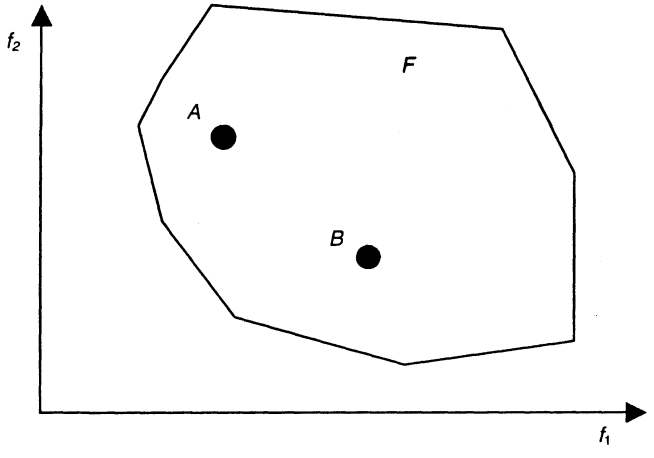


Рис. В.2. Оптимизация по двум критериям

Поскольку для принятия решений в условиях определенности выбор альтернативы равнозначен выбору исхода, принятие решения состоит здесь в выборе конкретной точки множества F. Какую точку надо взять в качестве оптимальной в данном случае?

По сравнению с примером В.1 это уже более трудная задача, т. к. при наличии не одного, а двух показателей, оценивающих исход, ответить на вопрос, какое решение является наилучшим, гораздо сложнее. Например, в точке A (см. рис. В.2) значение показателя f_1 лучше (меньше), чем в точке B, но зато в точке B лучше значение показателя f_2. Какую из них предпочесть? В данном случае речь идет не столько о том, как найти оптимальное решение, сколько о том, что следует понимать под оптимальным решением, т. е. здесь мы сталкиваемся с трудностями не технического, а концептуального характера.

Пример В.3. Студент факультета технической кибернетики, войдя в трамвай, решает, брать ли билет. Здесь исход определяется двумя обстоятельствами: решением студента и фактом появления контролера. Таким образом, студент выступает в качестве лица, принимающего решение, а факт появления контролера — в качестве среды. Имеются всего две альтернативы у принимающего решение и два состояния среды. Как численно оценить "полезности" исходов? Проще всего в качестве оценок взять выраженные в условных единицах денежные потери, как указано в табл. В.2.

Примечание

Здесь и далее "у. е." — некоторая условная единица, не обязательно совпадающая с конкретной денежной единицей.

Таблица В.2

Альтернатива	Состояние среды	
	Контролер появится	Контролер не появится
Брать билет	2 у. е.	2 у. е.
Не брать билета	8 у. е.	0 у. е.

Какое решение следует принять, если целью считать минимизацию потерь? Это пример задачи принятия решений в условиях неопределенности.

Методы решения подобных задач существенно зависят от наличия дополнительной информации, например, о том, можно ли каждому состоянию среды приписать вероятность его наступления или нет. Принципиальным также является вопрос о том, многократным или однократным является производимый выбор.

Пример В.4 (дилемма заключенного). Арестованы два подозреваемых в совершении серьезного преступления. У прокурора нет полного доказательства их вины, и результаты судебного разбирательства дела полностью зависят от стратегии поведения подозреваемых. У каждого из них есть две альтернативы — сознаться в совершении преступления или нет. Возможные исходы представлены в табл. В.3 (Н — непризнание, П — признание; 1, 2 — номера задержанных).

Таблица В.3

1	2	
	Н	**П**
Н	(1,1)	(10,0)
П	(0,10)	(7,7)

Таблица интерпретируется следующим образом. Если оба арестованных не сознаются, то им будет предъявлено обвинение в совершении относительно незначительного преступления (например, связанного с незаконным владением оружием), и оба они получат по 1 году лишения свободы. Если один сознается, а второй — нет, то первый за выдачу сообщника и помощь в расследовании дела будет полностью освобожден от ответственности, а второй получит полный срок — 10 лет лишения свободы. Если же оба сознаются, то оба понесут наказание, но за чистосердечное раскаяние срок заключения будет уменьшен до 7 лет. Какое решение следует принять каждому из заключенных, чтобы минимизировать наказание?

Здесь, как мы видим, тоже есть неопределенность, но в отличие от предыдущего примера, где присутствовала так называемая "природная" неопределенность, или неопределенность среды, в данном случае мы имеем неопределенность типа "активный партнер". Эффективность решения в такой задаче существенно зависит от стратегии поведения второго лица, а также от информированности обоих субъектов о намерениях другой стороны. Конфликтные ситуации выбора подобного типа рассматриваются в *разд. 3.5*.

Пример В.5. Во многих случаях лицо, принимающее решение, может указать лишь множество всех тех пар исходов, для которых первый исход в паре предпочтительнее второго. При этом какие-либо численные оценки исходов в принципе отсутствуют. Приведем конкретный пример. Моло-

дой ученый выбирает место своей будущей работы, исходя из следующего множества альтернатив:

1. x_1: ассистент в очень известном университете с окладом 250 у. е.
2. x_2: доцент в электротехническом институте с окладом 350 у. е.
3. x_3: профессор в малоизвестном периферийном институте с окладом 450 у. е.

Легко представить себе ситуацию, когда ученый предпочтет x_1 по сравнению с x_2, рассудив, что престиж известного университета и контакты с ведущими специалистами в данной области науки стоят 100 у. е. разницы в окладе. Данное предпочтение можно обозначить (x_1, x_2) или $x_1 \succ x_2$ (x_1 лучше x_2). Точно так же можно предположить, что $x_2 \succ x_3$. И в то же время, сравнивая x_1 и x_3, можно понять и выбор x_3 по сравнению с x_1 (слишком велика разница в окладе). Таким образом, система предпочтений задается множеством пар: (x_1, x_2), (x_2, x_3), (x_3, x_1). Следовательно, здесь нет самой предпочтительной альтернативы. Какими принципами следует руководствоваться для принятия решений в подобных ситуациях?

Пример В.6. Большой класс практических задач составляют *трудно формализуемые* задачи принятия решений, не имеющие адекватного традиционного математического описания. В качестве примера можно привести задачи медицинской диагностики, в которых по известной исходной информации (результаты анализов, внешние проявления болезни) требуется принять решение о типе заболевания. Такие задачи могут решаться на основе использования специальных программных комплексов — экспертных систем. Понятно, что здесь все традиционные методы математического анализа (как дисциплины) оказываются неприменимыми непосредственно и требуется особый подход. Важнейшее значение в таких системах принятия решений приобретают проблемы построения исходной базы знаний для конкретной (обычно достаточно узкой) предметной области и процедур логического вывода (правил), позволяющих делать разумные заключения из исходных фактов или утверждений. Характерным примером таких правил могут служить выражения типа "*ЕСЛИ* (условие), *ТО* (действие)", например:

> *ЕСЛИ* *х* за рыночную экономику и радикальные экономические реформы, *ТО* *х* будет голосовать за И. И. Иванова.

Указанный формат записи знаний характерен для важнейшего класса экспертных систем — *продукционных экспертных систем*, рассматриваемых в данной книге.

Пример В.7. Существуют проблемы так называемого группового выбора решений, когда основная задача состоит в том, чтобы указать "справедли-

вые" принципы учета индивидуальных выборов, приводящие к разумному общественному (или групповому) решению. В качестве содержательного примера можно привести заседание военного совета, когда каждый участник заседания высказывает свое мнение относительно плана проведения будущей операции, а в конечном итоге должен быть выбран один, оптимальный вариант. Как это сделать? Какой результат выбора считать "хорошим", каким свойством он должен обладать? Здесь у нас, как и в примере В.2, в первую очередь возникают концептуальные трудности, т. е. сначала нужно определить, какими показателями должен обладать разумный результат согласований индивидуальных предпочтений.

Простая модель задачи группового выбора формулируется следующим образом. Пусть множество вариантов решений X конечно: $X = \{x_1, x_2, \ldots, x_m\}$. Имеется группа из n членов, принимающих (выбирающих) решение. Каждый член группы с номером $i = 1, \ldots, n$ имеет свою систему предпочтений на множестве X, задаваемую с помощью бинарного отношения $R_i \subset X * X$,

$$R_i = \{(x_j, x_k), \ldots, (x_p, x_m)\}.$$

Здесь R_i — множество упорядоченных пар элементов из X, причем включение некоторой пары (x_s, x_t) в множество R_i означает, что с позиций i-го члена группы вариант x_s предпочтительнее варианта x_t: $x_s \succ x_t$. Требуется по заданной системе $R_1, \ldots, R_n$ индивидуальных предпочтений построить групповую (коллективную) систему предпочтений $R = f(R_1, \ldots, R_n)$, где f — некоторая функция, реализующая принятый принцип согласования индивидуальных предпочтений. Казалось бы, достаточно использовать логически очевидное правило большинства (что обычно и происходит на практике при коллективном решении проблем). Однако есть определенные трудности, связанные с естественными принципами согласования, типа правила большинства или оценивания по среднему баллу. В частности, хорошо известны парадоксы голосования, которые мы продемонстрируем на следующих задачах.

Принятие законопроекта в парламенте. Пусть три парламентские группы, обладающие приблизительно одинаковым числом голосов, обсуждают три варианта некоторого законопроекта a, b, c с целью утверждения одного "наилучшего" варианта. Пусть системы предпочтений групп имеют соответственно следующий вид:

1. $a \succ b \succ c$, $R_1 = \{(a, b), (b, c), (a, c)\}$.

2. $b \succ c \succ a$, $R_2 = \{(b, c), (c, a), (b, a)\}$.

3. $c \succ a \succ b$, $R_3 = \{(c, a), (a, b), (c, b)\}$.

Решено действовать по правилу простого большинства. Тогда в результате голосования получим $a \succ b$, потому что пара (a, b) присутствует в R_1 и R_3, а пара (b, a) — только в R_2. Аналогично устанавливаем, что $b \succ c$ и $c \succ a$, т. е. $a \succ b \succ c \succ a$.

Получаем "порочный круг" и потерю свойства транзитивности в групповом предпочтении. По результатам данного голосования по-прежнему нельзя выбрать наилучший законопроект. Более того, очевидно, что при умелом ведении заседания парламента председатель может обеспечить утверждение большинством голосов любого из трех вариантов. Действительно, председатель может предложить обсудить сначала какие-то два варианта, проголосовать и худший отсеять. Далее для обсуждения снова останутся два варианта — оставленный при первом рассмотрении и еще не рассматривавшийся. Тогда, очевидно, если на первое обсуждение выносятся варианты a, b, то оказывается $a \succ b$ и вариант b отбрасывается. Далее конкурируют a и c. В результате по принципу большинства имеем $c \succ a$ и в качестве окончательного варианта парламент выбирает вариант c. Если же, напротив, на первое обсуждение вынесем варианты d, c, то в итоге наилучшим окажется a. Точно так же можно обеспечить выбор b в качестве наилучшего. Невинное на первый взгляд предложение о порядке рассмотрения оказывает решающее влияние на результат!

Выборы президента (парадокс многоступенчатого голосования). Допустим, что на выборах президента некоторой компании (или государства) борются две партии, стремящиеся сделать победителем своего представителя. Далее показано, что при умелом ведении дела меньшинство может навязать свое мнение большинству, хотя голосование всегда будет проводиться по правилу большинства. Чтобы понять идею, достаточно изучить рис. В.3.

Из рис. В.3 видно, что группа, владеющая восемью голосами, в итоге навязала свое мнение группе из девятнадцати выборщиков. Все дело, конечно, заключается в умелом группировании сил. Но с помощью современных избирательных технологий это можно реализовать, и это делается повсеместно с помощью целенаправленного вложения средств, организации агитационных поездок в нужные регионы и т. д. Как показал анализ, несколько президентов США в указанном смысле действительно представляли меньшинство в результате реализации системы многоуровневого голосования. При этом чем больше ступеней, тем ярче проявляется указанный эффект.

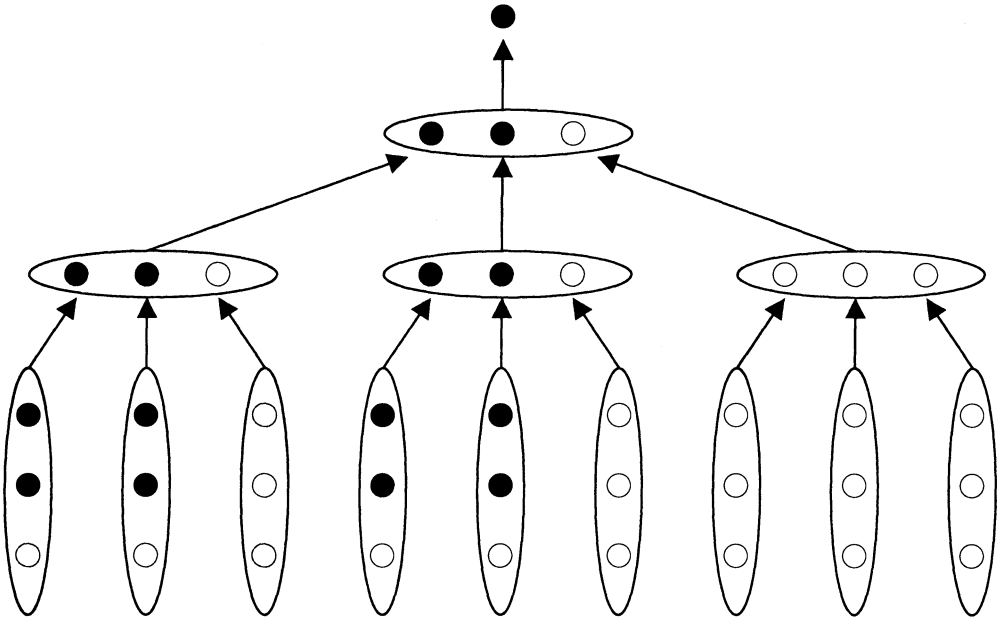


Рис. В.3. Система многоступенчатого голосования

И, наконец, последний пример.

Задача распределения ресурсов. Пусть некоторый ресурс (например, денежный) распределен между n членами некоторого сообщества. При этом состоянием сообщества (системы) будем называть вектор $(a_1, a_2, \ldots, a_n)$, где a_i — объем ресурса, которым владеет i-ый член сообщества. Общий объем ресурса постоянен и равен:

$$a = \sum_{i=1}^{n} a_i.$$

Рассмотрим другое состояние той же системы $b = (b_1, b_2, \ldots, b_n)$. Очевидно, состояние b не хуже состояния a для i-го субъекта, если $b_i \geq a_i$. Будем теперь производить перераспределение ресурсов на основе очень сильного большинства: переход системы из некоторого состояния a в состояние b разрешен, если новое состояние будет не хуже старого для всех членов сообщества кроме, может быть, одного (тотально-мажоритарное правило). Последовательность состояний $a_1, a_2, \ldots, a_m$ будем называть *тотально-мажоритарным путем из* a_1 *в* a_m, если каждый промежуточный переход из a_i в a_{i+1} был осуществлен на основе тотально-мажоритарного правила. Достаточно неожиданным является утверждение, что тотально-мажоритарный путь может связывать любые два состояния системы! Та-

ким образом, опираясь на мнение "всего общества" можно производить любые перераспределения ресурса, в том числе и представленные на рис. В.4.

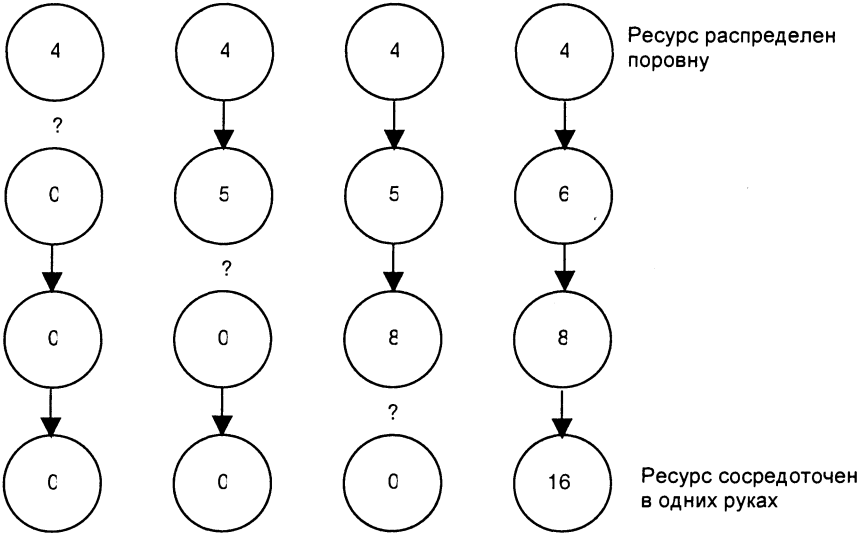
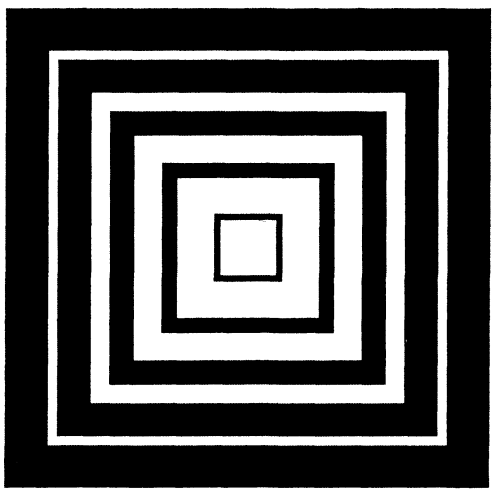


Рис. В.4. Распределение ресурса по принципу большинства

Приведенные примеры не исчерпывают всех типов задач теории принятия решений, хотя и охватывают значительное множество реальных ситуаций. В данной книге мы в основном ограничимся только теми задачами принятия решений, которые схематически проиллюстрированы в примерах В.1—В.6. Наиболее подробно будут изучены задачи однокритериального и многокритериального выбора в условиях определенности, а также задачи принятия решений в условиях "природных" неопределенностей и неопределенностей типа "активный партнер". Во всех случаях подразумевается наличие численных оценок исходов. Глава 12 посвящена введению в экспертные системы.

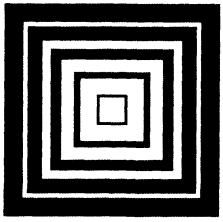
Цель книги — не только ознакомить читателей с некоторыми общими алгоритмическими принципами функционирования современных автоматизированных систем оптимизации и принятия решений, но и дать средства для разработки собственного программного продукта, реализующего те или иные принципы принятия решений в конкретной (может быть уникальной) практической ситуации.



ЧАСТЬ I

МЕТОДЫ ПРИНЯТИЯ РЕШЕНИЙ

Глава 1



Задача принятия решений

1.1. Постановка задачи принятия решений. Критериальный язык описания выбора

Задача принятия решений (ПР) возникает, когда присутствует несколько вариантов действий (альтернатив) для достижения заданного или желаемого результата. При этом требуется выбрать наилучшую в определенном смысле альтернативу.

Общую постановку задачи принятия решений, понимаемой нами как задача выбора из некоторого множества, можно сформулировать следующим образом.

Пусть X — множество альтернатив, Y — множество возможных последствий (исходов, результатов). (X и Y, вообще говоря, — произвольные абстрактные множества.) Предполагается существование причинной связи между выбором некоторой альтернативы $x_i \in X$ и наступлением соответствующего исхода $y_i \in Y$. Кроме того, предполагается наличие механизма оценки качества такого выбора — обычно оценивается качество исхода. В некоторых случаях целесообразно полагать, что мы имеем возможность непосредственно оценивать качество альтернативы x_i, и множество исходов по существу выпадает из рассмотрения. Требуется выбрать наилучшую альтернативу, для которой соответствующий исход имеет наилучшую оценку качества.

Задачу ПР можно проиллюстрировать с помощью рис. 1.1.

Перейдем к анализу сформулированной задачи ПР.

Первый важный момент заключается в определении характера связи альтернатив с исходами. Как мы видели из примеров, эта связь может быть детерминистской (или, как часто говорят, детерминированной).

В этом случае существует однозначное отображение

$$X \xrightarrow{\varphi} Y, \tag{1.1}$$

т. е. реализуется функция $y = \varphi(x)$, $x \in X$, $y \in Y$ (рис. 1.2).

Рис. 1.1. Задача принятия решений

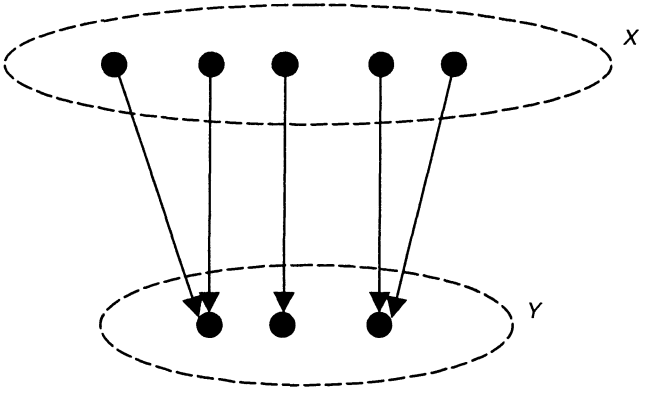


Рис. 1.2. Детерминированная связь

Эта же связь может иметь вероятностный характер, когда выбор x определяет некоторую плотность распределения вероятностей на множестве Y (иногда говорят, что с каждым x связана некоторая лотерея). В этом случае выбор x_i уже не гарантирует наступление определенного исхода y_i, а сама задача ПР называется задачей ПР в условиях *риска* (рис. 1.3).

Графы, представленные на рис. 1.2 и 1.3, называются *графами связей альтернатив с исходами*. Граф, представленный на рис. 1.3, является "взве-

шенным": каждая стрелка характеризуется весом, т. е. числом P_{ij} — вероятностью наступления исхода y_j при выборе альтернативы x_i. (В общем случае, как было сказано, задается соответствующая плотность распределения.) Очевидно,

$$\forall i : \sum_j P_{i,j} = 1. \tag{1.2}$$

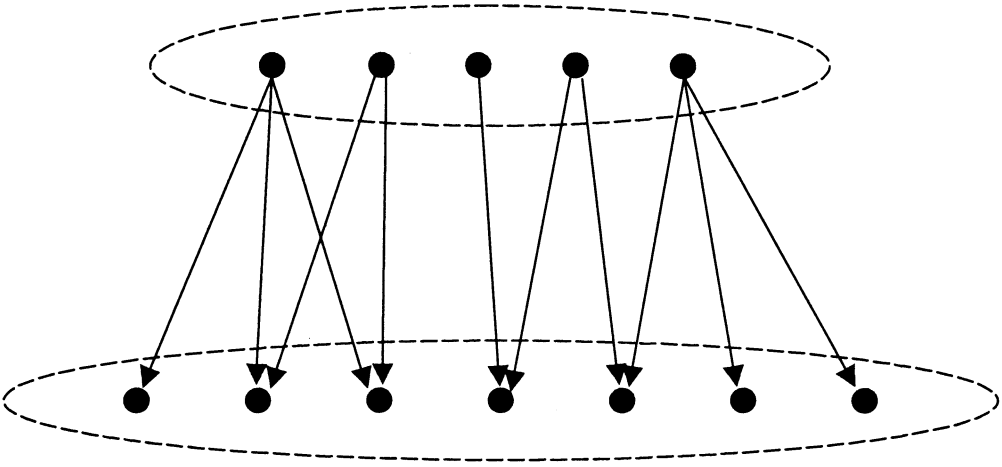


Рис. 1.3. Вероятностная связь

Тот же рис. 1.3 иллюстрирует третий вид связи альтернатив с исходами, реализуемый в задачах ПР в условиях *полной неопределенности*. При этом предполагается, что информация вероятностного характера отсутствует (стрелки на графе не имеют весов).

Как мы видели, неопределенность при выборе и при реализации связи альтернатив с исходами может иметь и другой, возможно более сложный характер (см. "дилемму заключенного" во *Введении*), но мы пока ограничимся указанными тремя случаями, которые могут быть проиллюстрированы также рис. 1.4.

При этом случаю 1 на рис. 1.4 соответствует ПР в условиях определенности; точками на оси y обозначены исходы, соответствующие выбору альтернатив x_1, x_2, x_3 (три альтернативы и три определенных исхода). Случай 2 характеризует задачу ПР в условиях неопределенности: после выбора любой из альтернатив x_1, x_2 или x_3 может быть указан лишь интервал расположения соответствующего исхода y. Случай 3 отражает ситуацию выбора в условиях риска. Показаны графики соответствующих

плотностей распределения вероятностей наступления события y в зависимости от выбора альтернативы x_1, x_2, или x_3.

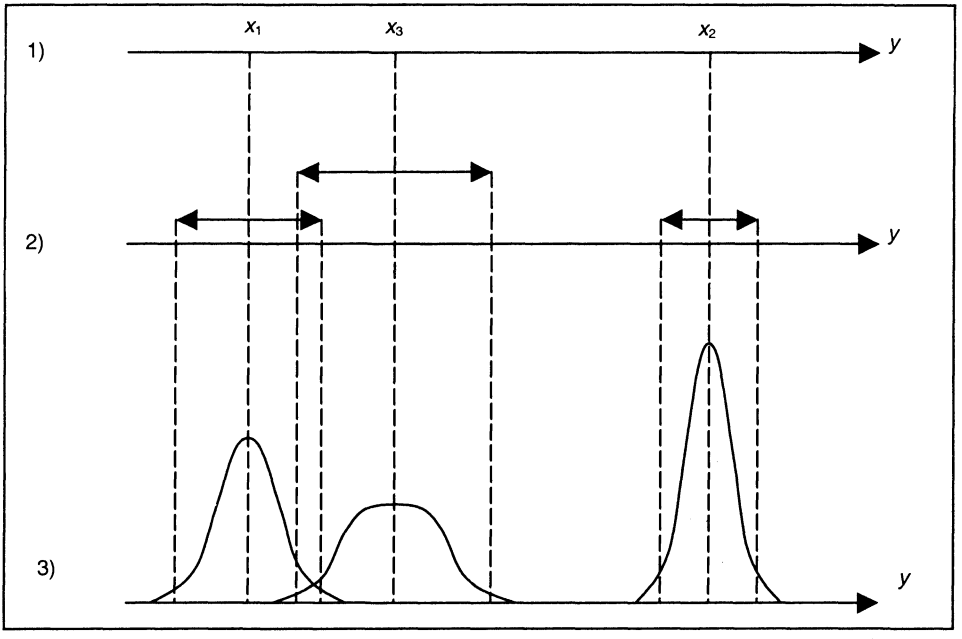


Рис. 1.4. Связь альтернатив с исходами при разных типах неопределенности

Заметим, что в каждом из рассмотренных случаев может дополнительно присутствовать свой механизм оценки качества исхода, не связанный непосредственно с механизмом появления y по заданному x. (Здесь предполагается, что числами y закодированы соответствующие исходы, которые могут оцениваться различным образом, например, по нескольким числовым критериям, см. далее).

Второй важный момент в общей задаче ПР состоит в изучении (задании) *системы предпочтений* лица, принимающего решение (ЛПР). Существенно, что второй момент, по сути, никак не связан с первым, и различные способы задания системы предпочтений могут быть реализованы для каждого вида связи альтернатив с исходами.

В некотором смысле простейшая ситуация возникает, когда каждый исход y можно оценить конкретным вещественным числом в соответствии с некоторым заданным отображением

$$f: Y \to R. \tag{1.3}$$

В этом случае сравнение исходов сводится к сравнению соответствующих им чисел, например, исход y_i может считаться более предпочтительным, чем y_j, если $f(y_i) > f(y_j)$ (задача максимизации). Исходы эквивалентны, если $f(y_i) = f(y_j)$. Для сравнения самих исходов употребляются выражения

$$y_i \succ y_j,\ y_i \sim y_j.$$

Такая функция f называется *целевой функцией*, *критериальной функцией*, *функцией критерия оптимальности* или даже просто *критерием оптимальности*. Последнее название не вполне корректно, ибо критерий оптимальности — это, вообще говоря, некоторое правило, позволяющее отличать "оптимальные" решения (исходы) от "неоптимальных" и сравнивать исходы между собой. В данном случае это правило связано с заданием целевой функции f. Как известно из математики, однозначное отображение произвольного множества на множество вещественных чисел называется *функционалом*. Поэтому целевые функции мы часто будем называть *целевыми функционалами*.

Если предположить, что связь между множеством альтернатив и множеством исходов детерминистская:

$$y = \varphi(x),$$

то функция f, заданная на множестве Y, трансформируется в некоторую функцию J, заданную на X и являющуюся суперпозицией φ и f:

$$J : X \to R,\ J = f \cdot \varphi.$$

В этом случае задача выбора оптимального исхода сводится к задаче выбора оптимальной альтернативы на множестве X и решается непосредственно методами теории оптимизации.

Более реалистичной часто оказывается ситуация, когда в отличие от предыдущего случая "качество" или "полезность" исхода y оценивается не одним числом $f(y)$, а несколькими. Иначе говоря, предполагается, что существует несколько показателей качества решения (критериев), описываемых функциями

$$f_k : Y \to R,\ k = 1, 2, \ldots, m,$$

причем каждую из *частных целевых функций* f_i требуется максимизировать (см. *Введение*, пример В.2). Понятно, что в случае многокритериальных оценок исходов возникают существенно более сложные математические модели ситуации выбора, чем в однокритериальном случае. Критерии обычно противоречивы и, как правило, достигают максимумов в различных точках $y \in Y$. Следовательно, возникают не только алгоритмические трудности по решению соответствующих оптимизацион-

ных задач, но и чисто концептуальные трудности: что понимать под оптимальным решением в этом случае? Кроме того, здесь уже появляются и несравнимые по векторному критерию $f = (f_1, \ldots, f_m)$ варианты y_i, y_j. Более подробно многокритериальные модели принятия решений будут рассмотрены далее.

Ограничиваясь пока указанными выше тремя способами связи альтернатив с исходами и двумя способами описания предпочтений ЛПР на критериальном языке, получим таблицу основных задач выбора (рис. 1.5).

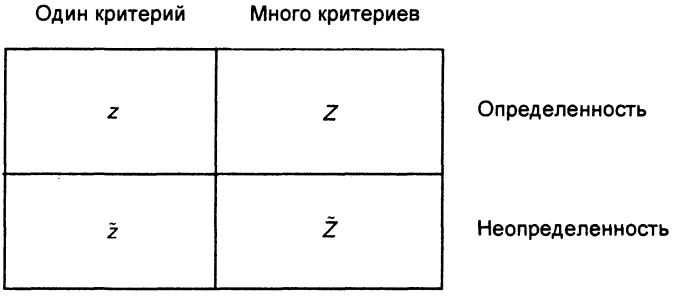


Рис. 1.5. Основные задачи выбора

На рис. 1.5:

$$z = f(y), f: Y \to R;$$

$$Z = f(y), f = (f_1, \ldots, f_m), f_k: Y \to R, k = 1, 2, \ldots, m.$$

Волна сверху означает наличие неопределенности в задаче ПР.

Необходимо отметить, что в настоящее время в приложениях часто применяется именно критериальный язык описания предпочтений, поэтому следующая важнейшая группа проблем — это формирование критериев и целевых функций (функционалов). Эти проблемы, как будет показано, решаются в тесной связи с методами преодоления различных видов неопределенностей на основе тех или иных гипотез.

1.2. Описание выбора на языке бинарных отношений. Формальные модели задачи принятия решений

Язык бинарных отношений — второй, более общий, чем критериальный, язык описания системы предпочтений ЛПР.

Предполагается, что:

- отдельный исход сам по себе не оценивается и критериальные функции не вводятся;
- каждая пара исходов y_i, y_j может находиться в одном из следующих отношений:
  - y_i предпочтительнее (строго доминирует) y_j;
  - y_j предпочтительнее y_i;
  - y_i не менее предпочтителен, чем (не строго доминирует) y_j;
  - y_j не менее предпочтителен, чем y_i;
  - y_i эквивалентен y_j;
  - y_i и y_j несравнимы между собой.

Будем далее предполагать, что свои предпочтения пользователь устанавливает в некотором множестве A. В стандартном случае — это множество исходов: $A = Y$. Однако при детерминистской связи X с Y возможно $A = X$, или при многокритериальной оценке исходов $A = f(Y), f = f_1, \ldots, f_m$. В последнем случае предполагается, что система предпочтений ЛПР задается непосредственно в пространстве векторных оценок исходов. При необходимости можно полагать, что это пространство и есть пространство исходов. В рассматриваемом случае система предпочтений пользователя задается с помощью соответствующего бинарного отношения R на A. Напомним, что бинарным отношением на множестве A называется произвольное подмножество R множества A^2, где A^2 — множество всех упорядоченных пар вида (a_i, a_j), где $a_i, a_j \in A$. Имеем, следовательно, $R \subseteq A^2$, в том числе $A^2 \subseteq A^2$. Основные свойства бинарных отношений (рефлексивность, симметричность, транзитивность, антирефлексивность и т. д.) предполагаются известными.

Существует наглядный способ задания бинарных отношений на конечных множествах, который мы используем в данной книге. Изобразим элементы конечного множества A точками на плоскости. Если задано отношение $R \subseteq A^2$ и $(a_i, a_j) \in R$, где $a_i, a_j \in A$, то проведем стрелку от a_i к a_j. Если $(a_i, a_i) \in R$, то у точки a_i нарисуем петлю-стрелку, выходящую из a_i и входящую в ту же точку. Получившаяся фигура называется *ориентированным графом*, а сами точки — *вершинами графа*. Заметим здесь же, что вместо $(a_i, a_j) \in R$ можно писать $a_i R a_j$.

Основной вопрос заключается в следующем. Пусть на множестве A задана система предпочтений ЛПР в виде бинарного отношения R (часто это

отношение строгого доминирования). Что тогда следует понимать под решением задачи выбора? Этот основной вопрос в разных случаях (системах оптимизации и принятия решений, пакетах прикладных программ) решается неоднозначно, и пользователям необходимо ясно осознавать, что же конкретно имеется в виду в каждом отдельном случае. Перейдем к точным формулировкам.

Пусть A — заданное множество и R — произвольное бинарное отношение на A. Тогда пара $<A, R>$ называется *моделью выбора.*

Определение 1.1

Пусть задана модель $<A, R>$. Элемент $a^* \in A$ называется наилучшим по R в A, если $(a^*, a) \in R$ при $\forall a \in A \setminus a^*$.

На рис. 1.6, *а* наилучшими элементами являются a_1, a_2. На графе рис. 1.6, *б* наилучших элементов нет.

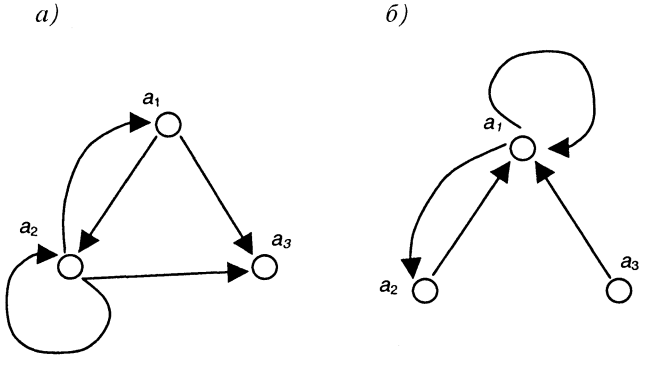


Рис. 1.6. Понятие наилучшего элемента

На языке графов понятие наилучшего элемента соответствует наличию вершины, соединенной исходящими из нее стрелками со всеми остальными вершинами графа. При этом могут присутствовать и любые другие дополнительные соединения.

Если предположить, что бинарное отношение, представленное на рис. 1.6, *б*, является отношением предпочтения, и стрелки означают некоторый вариант доминирования, то, по-видимому, целесообразно както исключить a_1 из дальнейшего рассмотрения, ибо этот элемент доминируется элементом a_3. С помощью понятия наилучшего элемента это сделать невозможно, хотя в случае, представленном на рис. 1.6, *а*, мы исключили элемент a_3, как не являющийся наилучшим.

Введем вместо наилучшего элемента более слабое понятие максимального элемента.

Определение 1.2

Элемент $a^0 \in A$ называется максимальным в модели $<A, R>$ или максимальным по R в A, если $\forall a \in A : (a, a^0) \in R \rightarrow (a^0, a) \in R$.

Множество всех максимальных в $<A, R>$ элементов обозначим через $\mathrm{Max}_R A$.

Очевидно, граф отношения, имеющего максимальные элементы, должен содержать вершины, в которых каждой входящей в нее стрелке (если таковые имеются) соответствует "компенсирующая", выходящая стрелка, направленная в вершину, из которой исходит указанная входящая стрелка.

В примерах, приведенных на рис. 1.6, максимальными по R элементами будут:
а) a_1, a_2; б) a_2, a_3.

Легко доказать, что наилучший по R в A элемент является и максимальным. Обратное верно только, если отношение R обладает специальным свойством слабой связности:

$$\forall a_1, a_2 : (a_1 \neq a_2) \rightarrow ((a_1, a_2) \in R) \vee ((a_2, a_1) \in R).$$

На рис. 1.6, *б* отношение R не является слабо связным.

Иногда используется понятие R-оптимального элемента.

Определение 1.3

Элемент $a^0 \in A$ называется R-оптимальным на A, если $\forall a \in A$, $a \neq a^0 \rightarrow$ $\rightarrow (a, a^0) \notin R$.

Иначе говоря, здесь требуется существование вершины, в которую не входит ни одна стрелка.

На рис. 1.6, *а* R-оптимальных элементов нет, на рис. 1.6, *б* элемент a_3 будет R-оптимальным.

Основным понятием для нас далее будет понятие максимального элемента.

Определение 1.4

Множество $\mathrm{Max}_R A$ называется внешне устойчивым, если для любого элемента $a \in A \setminus \mathrm{Max}_R A$ найдется такой $a^0 \in \mathrm{Max}_R A$, что справедливо $(a^0, a) \in$ $\in R$.

Понятие внешней устойчивости оказывается существенным для проблемы ПР. Действительно, если множество $\mathrm{Max}_R A$ внешне устойчиво, то последующий выбор оптимального элемента (на основе, например, привлечения дополнительной информации) может производиться только в пределах множества $\mathrm{Max}_R A$. В противном случае, когда устойчивости нет, такой вывод уже не будет иметь разумного обоснования.

Внешне устойчивое множество $\mathrm{Max}_R A$ называется *ядром отношения* R в A. Иногда термин "ядро" применительно к множеству $\mathrm{Max}_R A$ используется и без требования внешней устойчивости.

В примерах на рис. 1.6 множества $\mathrm{Max}_R A$ являются внешне устойчивыми. На рис. 1.7 представлен случай, когда множество

$$\mathrm{Max}_R A = \{a_1, a_2\}$$

не является внешне устойчивым.

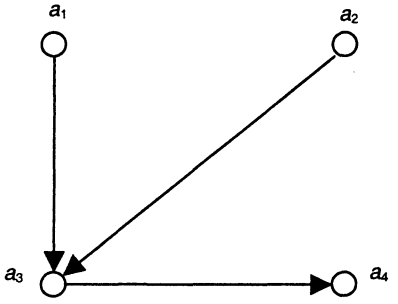


Рис. 1.7. Отсутствие внешней устойчивости

Далее под задачей ПР, сформулированной на языке бинарных отношений, понимается задача выделения ядра — множества максимальных элементов из A по некоторому бинарному отношению R: $A^* = \mathrm{Max}_R A$. Специфика конкретных задач ПР находит отражение в задании соответствующего множества вариантов A, а также в формировании бинарного отношения R, характеризующего вполне определенные цели принятия решений в той или иной практической ситуации. Весьма важным с практической точки зрения являются следующие специальные виды бинарных отношений:

- квазипорядок (R — рефлексивно и транзитивно);
- строгий порядок (R — антирефлексивно и транзитивно);
- эквивалентность (R — рефлексивно, симметрично и транзитивно).

1.3. Связь различных способов описания выбора. Однокритериальный и многокритериальный выбор

В данном разделе мы рассмотрим связь критериального языка описания выбора и языка бинарных отношений.

Однокритериальный выбор. Пусть

$$f: Y \to R$$

есть целевая функция, которую требуется максимизировать. Тогда с помощью этой функции на множестве Y индуцируются два бинарных отношения R_1 и R_2:

$$(y_1, y_2) \in R_1 \leftrightarrow f(y_1) \geq f(y_2);$$

$$(y_1, y_2) \in R_2 \leftrightarrow f(y_1) > f(y_2).$$

Отношение R_1 является, очевидно, рефлексивным и транзитивным и, следовательно, определяет квазипорядок на Y. Отношение R_2 обладает свойством антирефлексивности ($\forall y$ неверно, что $f(y) > f(y)$) и транзитивности, являясь строгим порядком. В обоих случаях справедливо равенство:

$$\mathrm{Max}_{R_i} Y = \mathrm{Arg\,max}\, f(y), \quad i = 1, 2.$$

Множество максимизаторов функции f является внешне устойчивым в Y. Таким образом, задача максимизации целевой функции f на множестве Y эквивалентна задаче построения ядра одного из бинарных отношений R_1, R_2, совпадающего с множеством максимизаторов f на Y.

Многокритериальный выбор. Предположим теперь, что "качество", или "полезность", исхода оценивается не одним числом, а несколькими. Иначе говоря, предполагается, что существует несколько показателей качества решения, описываемых частными целевыми функциями

$$f_k: Y \to R, \quad k = 1, 2, \ldots, m,$$

которые требуется максимизировать.

В теории многокритериальных задач обычно используются следующие отношения доминирования:

$$(y_i, y_j) \in R_p \leftrightarrow \forall k: [f_k(y_i) \geq f_k(y_j)] \wedge [f(y_i) \neq f(y_j)];$$

$$(y_i, y_j) \in R_s \leftrightarrow \forall k: [f_k(y_i) > f_k(y_j)].$$

Здесь $f = (f_1, f_2, \ldots, f_m)$. Отношение доминирования R_p называется *отношением Парето*, а R_S — *отношением Слейтера*. Употребляется также запись

$$(y_i, y_j) \in R_t \leftrightarrow y_i \overset{t}{\succ} y_j, \quad t = P, S.$$

Определение 1.5

Если для некоторой точки $y^0 \in Y$ не существует более предпочтительной по Парето точки, т. е. такой точки y, что $(y, y^0) \in R_p$, то тогда точка y^0 называется *эффективным или Парето-оптимальным решением многокритериальной задачи* $f_k(y) \to \max$, $k = 1, 2, \ldots, m$; $y \in Y$.

Множество, включающее в себя все эффективные элементы множества Y, обозначается $P_f(Y)$ или просто $P(Y)$ (если ясно, о каком векторном критерии f идет речь) и называется *множеством Парето для векторного отношения*

$$f: Y \to R^m, f = (f_1, \ldots, f_m).$$

Очевидно, $P(Y) \subseteq Y$. Образ множества $P(Y)$ в *пространстве критериев* R^m обозначается $P(f)$. Множество $P(f) = f(P(Y))$ называется *множеством эффективных оценок*. Множество эффективных оценок называется также *множеством Парето в пространстве критериев*.

Смысл введенного понятия эффективного решения состоит в том, что оптимальный исход следует искать только среди элементов множества недоминируемых элементов $P(Y)$ (принцип Парето). В противном случае всегда найдется точка $y \in Y$, оказывающаяся более предпочтительной с учетом всех частных целевых функций $f_i(y)$.

Ясно, что бинарное отношение R_p является антирефлексивным, т. к. $\forall y \in Y : (y, y) \notin R_p$. Кроме того, легко установить, что

$$((y_i, y_j) \in R_p) \wedge ((y_j, y_k) \in R_p) \to (y_i, y_k) \in R_p.$$

Таким образом, отношение R_p транзитивно. Отсюда следует, что R_p — частичный строгий порядок на Y.

Обычно цель решения многокритериальной задачи

$$f_k(y) \to \max_{y \in Y}$$

состоит в выделении множества Парето $P(Y)$. При отсутствии дополнительной информации о системе предпочтений пользователя большего сделать нельзя.

Обратимся теперь к отношению R_S.

Определение 1.6

Точка $y' \in Y$ называется *слабо эффективным решением многокритериальной задачи, или решением, оптимальным по Слейтеру*, если не существует более предпочтительной по Слейтеру точки, т. е. такой точки y, что $(y, y') \in R_S$.

Иначе говоря, исход "y" называется слабо эффективным, если он не может быть улучшен сразу по всем m критериям, задаваемым с помощью частных целевых функций $f_i(y)$, $i = 1, 2, ..., m$. Множество слабо эффективных элементов обозначается через $S_f(Y)$ или просто $S(Y)$. Очевидно, $S(Y) \subseteq Y$, $P(Y) \subseteq S(Y)$. Аналогично предыдущему случаю вводим обозначение $S(f) = f(S(Y))$.

Введение понятия слабо эффективного решения вызвано, в частности, тем, что в результате многокритериальной оптимизации часто получаются именно эти решения, представляющие, вообще говоря, меньший интерес, чем эффективные решения.

Точно так же, как и в однокритериальных задачах выбора, цель решения многокритериальной задачи может быть сформулирована как задача построения ядра отношения доминирования R_p (отношения Парето). Легко доказать непосредственно, что в этом случае

$$P(Y) = \mathrm{Max}_{R_p} Y$$

с выполнением свойства внешней устойчивости множества Парето.

Таким образом, мы видим, что задание целевых функций для оценки качества исходов, как в однокритериальном, так и многокритериальном случае, может порождать различные системы предпочтений, выраженные на языке бинарных отношений. При этом задача построения ядра оказывается эквивалентной либо задаче построения множества максимизаторов скалярной целевой функции, либо задаче построения множества Парето для векторной целевой функции.

Пример 1.1. Пусть задана векторная целевая функция $f = (f_1, f_2)$ на множестве $Y = \{y_1, ..., y_5\}$, причем частные целевые функции f_i требуется максимизировать. Образы точек y_i в *пространстве критериев* (f_1, f_2) обозначим соответствующими числами (рис. 1.8).

Используя определение доминирования по Парето, можно для этой задачи построить само отношение R_p и его граф (рис. 1.9).

Используя определение ядра, с помощью непосредственного рассмотрения графа получаем:

$$\mathrm{Max}_{R_p} Y = \{y_2, y_3\}.$$

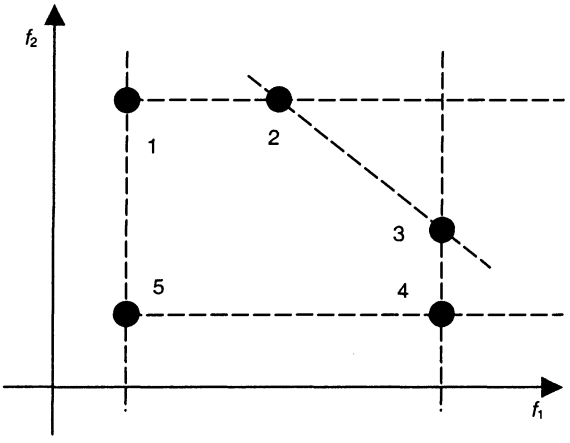


Рис. 1.8. Двухкритериальная задача

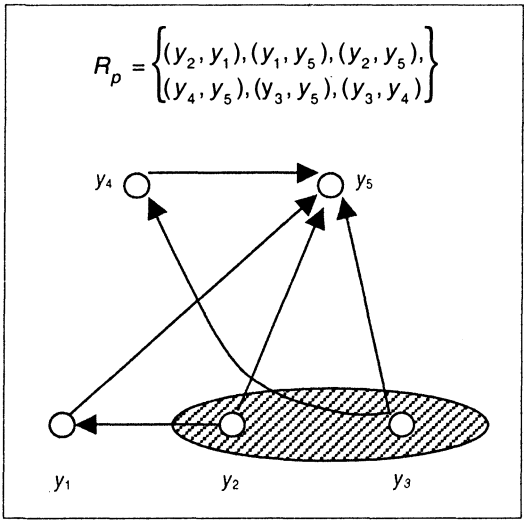


Рис. 1.9. Отношение Парето и его граф

На рис. 1.9 ядро выделено штриховкой. Можно заметить, что наилучшие элементы (см. определение 1.1) в данном случае отсутствуют, а понятие максимального элемента позволяет в полной мере формализовать многокритериальную задачу выбора как задачу построения множества недоминируемых по Парето элементов.

С помощью аналогичных рассмотрений устанавливаем, что отношение Слейтера R_S (так же, как и отношение R_p) является строгим порядком и

может быть представлено графически (рис. 1.10). Ядро выделено штриховкой.

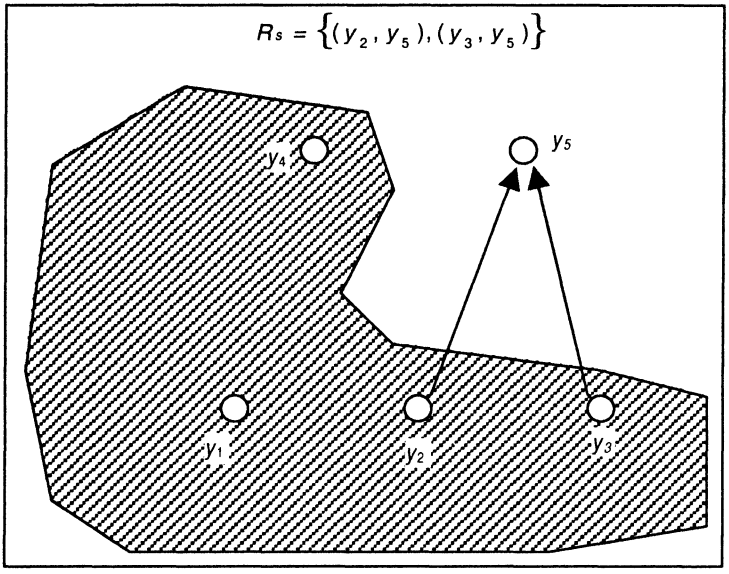


Рис. 1.10. Отношение Слейтера

Видно, что, во-первых, $R_S \subset R_p$, а во-вторых,

$$\mathrm{Max}_{R_p} Y \subset \mathrm{Max}_{R_S} Y = \{y_1, y_2, y_3, y_4\}.$$

Эти включения выполняются и в общем случае.

Замечание 1.1

Отношение Парето R_p порождает соответствующее отношение несравнимости R_H:

$$(y_1, y_2) \in R_H \leftrightarrow [(y_1, y_2) \notin R_p] \wedge [(y_2, y_1) \notin R_p].$$

Для последнего примера имеем, в частности, $R_H = \{(y_4, y_2), (y_2, y_3), (y_1, y_4), (y_4, y_1), \ldots\}$.

Важно отметить, что, например, $(y_4, y_2) \in R_H$, $(y_2, y_3) \in R_H$, но $(y_4, y_3) \notin R_H$.

Таким образом, отношение несравнимости в многокритериальных задачах, являясь симметричным ($(y_i, y_j) \in R_H \rightarrow (y_j, y_i) \in R_H$), не является транзитивным.

1.4. Функции выбора

Наряду с критериальными языками описания выбора и языком бинарных отношений существует еще более общий подход. Он основан на понятии *функции выбора*. Основная идея заключается не в оценке каждой альтернативы с помощью одного или нескольких числовых критериев и не в попарном сравнении альтернатив по предпочтительности, а в выделении из некоторого множества альтернатив подмножества "лучших" вариантов.

Перейдем к более точным определениям.

Пусть X — множество (может быть и бесконечное) всех возможных альтернатив. Тогда через 2^X обозначается множество всех подмножеств X. Среди всех подмножеств X выделяется класс XD допустимых предъявлений [4]:

$$XD \subseteq 2^X.$$

Введем следующее определение.

Определение 1.7

Функцией выбора на классе допустимых предъявлений XD называется функция

$$C : XD \to 2^X$$

такая, что для любого множества $A \in XD$

$$C(A) \subseteq A.$$

Таким образом, функция выбора ставит в соответствие каждому множеству альтернатив (из класса допустимых предъявлений) некоторое его подмножество. В результате происходит сужение предъявляемого множества альтернатив, что и моделирует процесс выбора нужных ("лучших") вариантов.

С помощью понятия функции выбора можно описывать и ранее рассмотренные варианты выбора, сформулированные на критериальном языке или языке бинарных отношений. Однако основное достоинство нового языка состоит в возможности моделирования более сложных принципов выбора. Например, может ставиться задача выбора из заданного множества альтернатив "среднего" или "типичного" варианта. Возможность описания такого выбора, скажем, на языке бинарных отношений представляется весьма сомнительной, если вообще не лишенной смысла.

Приведем пример функции выбора, осуществляющей выбор эффективных (Парето-оптимальных) точек в многокритериальной задаче

$$f_k(a) \to \max_{a \in A}, \quad A \subset R^n.$$

Такая функция выбора может быть задана следующим образом:

$$C_p(A) \stackrel{\Delta}{=} \{a \in A \mid \forall y \in A,\ y \neq a : \exists i,\ y_i < a_i\}.$$

Здесь точка a из A выбирается тогда и только тогда, когда любая другая точка из A будет "хуже" (в данном случае — меньше) хотя бы по одному из частных критериев f_k.

Типичные ситуации выбора описываются функциями выбора, удовлетворяющими некоторым специальным ограничениям, что позволяет строить и изучать различные классы функций выбора. Наиболее часто на функции выбора накладываются ограничения, сводящиеся к выполнению свойств (аксиом), представленных далее.

- **Наследование (Н-свойство).** Это свойство подразумевает, что вариант, выбираемый из некоторого множества, будет также выбран, если предъявить для выбора любое подмножество, содержащее этот вариант:

$$\forall Y' \subseteq Y \in YD \to C(Y) \cap Y' \subseteq C(Y').$$

  Смысл этой записи состоит в следующем. $C(Y)$ — это выбранные элементы из Y. Если какие-то из них будут предъявлены в составе подмножества Y', т. е. пересечение $C(Y) \cap Y'$ не пусто, то они также должны войти в выбор $C(Y')$.

  Легко видеть, что аксиома наследования автоматически не выполняется для любого "разумного" выбора. Действительно, если снова вернуться к примеру выбора альтернативы со "средними" характеристиками, то среднее во множестве Y может не совпадать со средним в более узком множестве Y', $Y' \subset Y$.

- **Отбрасывание (О-свойство).** Это свойство называется также *условием независимости от отвергнутых альтернатив*. Оно означает, что если удалить из предъявляемого множества какие-то (вообще говоря, не все) невыбранные альтернативы, то выбор на оставшемся множестве не изменится:

$$\forall Y' \subseteq Y \in YD, C(Y) \subseteq Y' \to C(Y') = C(Y).$$

  Иначе говоря, если подмножество Y' включает в себя все выбранные из Y варианты: $C(Y) \subseteq Y'$, то выбор на Y' совпадает с выбором на Y.

- **Согласованность (С-свойство).** Данное требование означает, что если вариант выбирается в каждом из двух множеств (предъявлений), то он будет выбран и в объединении этих множеств:

$$\forall Y', Y'' \in YD, Y' \cup Y'' \in YD : C(Y') \cap C(Y'') \subseteq C(Y' \cup Y'').$$

Упражнение 1.1. Докажите, что "паретовская" функция C_p обладает всеми тремя свойствами.

В теории функций выбора рассматриваются и другие аксиомы [1, 4, 26].

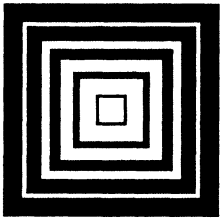
Поскольку язык функций выбора оперирует понятиями теории множеств, с помощью операций над множествами можно строить новые функции выбора из уже имеющихся. Так, например, если заданы функции выбора C_1, C_2 то имеют смысл и функции выбора вида $C_1 \cup C_2, C_1 \cap C_2$ и т. д.

Упражнение 1.2. Докажите, что:

- если C_1, C_2 обладают H-свойством, то это же свойство есть у функции $C_1 \cup C_2$;
- если C_1, C_2 имеют C-свойство, то оно же имеется у функции $C_1 \cap C_2$, а для функции $C_1 \cup C_2$ это неверно.

В настоящее время аппарат теории функций выбора имеет большое теоретическое значение, позволяя моделировать различные практически значимые ситуации выбора и анализировать с общих позиций, например, процедуры многокритериального выбора вариантов [4].

В заключение рассмотрения функций выбора отметим, что введение механизма предъявления множеств является принципиальным для практических применений [1, 4]. Часто полагают, что класс допустимых предъявлений совпадает с множеством всех подмножеств 2^X, где X — исходное множество альтернатив. Однако простейшие примеры показывают, что в действительности нам оказывается доступным лишь некоторое подмножество $XD \subseteq 2^X$. Например, пусть множество X означает множество всех возможных видов данного товара. Мы должны выбрать один или несколько видов товара. Обычно мы ограничены в своем выборе теми магазинами, которые мы в состоянии посетить. Реально мы будем иметь дело с некоторым множеством XD, определяемым фактом наличия различных видов товара в каждом из доступных магазинов.



Глава 2

Многокритериальные модели принятия решений в условиях определенности

Рассматривается следующая модель задачи ПР:

- X — множество альтернатив;
- Y — множество исходов;
- $f_i : Y \to R, i = 1, ..., m$ — множество показателей качества (критериев);
- $\varphi : X \to Y$ — детерминистская функция, отображающая множество альтернатив во множество исходов. (Здесь R — множество вещественных чисел.)

Таким образом, мы здесь предполагаем, что каждому решению $x \in X$ соответствует единственный элемент $y \in Y$, где $y = \varphi(x)$. "Качество" или "полезность" исхода y, а тем самым и соответствующего решения x оценивается несколькими (m) числами в соответствии с зависимостями f_i. По-прежнему предполагаем, что каждую из функций f_i требуется максимизировать.

С помощью суперпозиции

$$J_i(x) = f_i(\varphi(x)), i = 1, ..., m$$

мы имеем возможность непосредственно оценивать качество самого решения x и работать с векторным отображением

$$J : x \to R^m, J = (J_1, ..., J_m), J(X) = F \subset R^m.$$

Более того, задание бинарного отношения предпочтения R'; на множестве исходов Y индуцирует соответствующее бинарное отношение R'' на множестве X. Именно:

$$(x_1, x_2) \in R'' \leftrightarrow (\varphi(x_1), \varphi(x_2)) \in R'.$$

Соответственно возникает бинарное отношение R''' во множестве оценок $F \subset R^m$:

$$\forall z_1, z_2 \in F : (z_1, z_2) \in R''' \leftrightarrow (y_1, y_2) \in R',$$

где $z_1 = f(y_1)$, $z_2 = f(y_2)$. Поэтому в детерминистском случае (в условиях определенности) отношения предпочтения могут задаваться в любом из указанных трех множеств: X, Y, F. Далее в качестве основного отображения будет рассматриваться отображение

$$J : X \to F \subset R^m,$$

и соответственно системы предпочтений будут задаваться во множествах X, F.

В практических задачах часто непосредственно задается отображение J и, по сути, $Y = F$, т. е. в качестве исходов выступают сами оценки J_i.

В результате мы приходим к очень распространенной в приложениях многокритериальной модели принятия решений, или задаче многокритериальной оптимизации вида

$$J_i(x) \to \max_{x \in X}, \; i = 1, ..., m, \; X \subset R^n.$$

Мы здесь сделали еще одно уточнение: $X \subset R^n$. То есть мы предполагаем, что все альтернативы или решения параметризованы и каждому из решений соответствует точка $x \in R^n$, $x = (x_1, x_2, ..., x_n)$. И, наконец, вместо обозначений $J_i(x)$ мы снова вернемся к обозначениям $f_i(x)$. Множество X называется *множеством допустимых значений* и в разделах, посвященных многокритериальным задачам, будет обозначаться через D.

2.1. Методы многокритериальной оптимизации

Рассматривается задача многокритериальной оптимизации вида

$$f_i(x) \to \max_{x \in D}, \; f_i : D \to R, \; i = 1, ..., m; \; D \subseteq R^m. \qquad (2.1)$$

Таким образом, задано m функций или функционалов f_i, отображающих множество D n-мерных векторов $x = (x_1, ..., x_n)$ во множество вещественных чисел R. Здесь предполагается, что выбор оптимальных значений x производится не во всем n-мерном пространстве R^n, а лишь в пределах некоторого его подмножества D. Например, можно интерпретировать задачу (2.1) как задачу оптимального выбора параметров $x_1, ..., x_n$ некоторой системы (например, некоторого программного комплекса или перспективного плана развития фирмы), качество функционирования которой оценивается показателями $f_1, ..., f_m$ (см. *Введение*, пример В.2). В этом случае ограничение $x \in D$ отражает наши технологические и иные возможности реализации тех или иных значений x_i. Кроме того, часть ограничений может формироваться на основе имеющейся априорной информации, позволяющей исключить из рассмотрения заведомо неудачные варианты x.

Важнейшее значение при исследовании задач (2.1) имеет принцип Парето и связанные с ним понятия эффективного (Парето-оптимального) и слабо эффективного решения. Однако прежде чем перейти к рассмотрению численных методов построения множества Парето, обратимся к традиционным "инженерным" методам многокритериальной оптимизации, сводящим задачу (2.1) к некоторой ее однокритериальной версии.

Метод главного критерия. В методе главного критерия в качестве целевой функции выбирается один из функционалов f_i, например f_1, наиболее полно с точки зрения исследователя отражающий цель ПР. Остальные требования к результату, описываемые функционалами $f_2, ..., f_m$, учитываются с помощью введения необходимых дополнительных ограничений. Таким образом, вместо задачи (2.1) решается другая, уже однокритериальная задача вида

$$\begin{aligned} &f_1(x) \to \max_{x \in D'};\ \ D' \subseteq D \subseteq R^n; \\ &D' = \{x \in D \ / \ f_i(x) \geq t_i,\ i = 2, ..., m\}. \end{aligned} \tag{2.2}$$

Формально получили более простую задачу поиска максимума функционала f_1 на новом допустимом множестве D'. Добавились ограничения вида $f_i(x) \geq t_i$, показывающие, что мы согласны не добиваться максимальных значений для функционалов $f_2, ..., f_m$, сохраняя требование их ограниченности снизу на приемлемых уровнях. Важно понимать, что переход от задачи (2.1) к задаче (2.2) вовсе не есть переход от одной эквивалентной задачи к другой. Произошло существенное изменение исходной постановки задачи, которое в каждой конкретной ситуации требует отдельного обоснования. Мы еще вернемся далее к методу главного кри-

терия и его анализу с позиций оптимальности по Парето. Здесь же отметим, что применение этого метода на интуитивном уровне обычно наталкивается на трудности, связанные с возможным наличием нескольких "главных" критериев, находящихся в противоречии друг с другом. Кроме того, не всегда ясен алгоритм выбора нижних границ t_i. Их необоснованное задание может привести, в частности, к пустому множеству D'.

Метод линейной свертки. Это наиболее часто применяемый метод "скаляризации" (свертки) задачи (2.1), позволяющий заменить векторный критерий оптимальности $f = (f_1, ..., f_m)$ на скалярный $J : D \to R$. Он основан на линейном объединении всех частных целевых функционалов $f_1, ..., f_m$ в один:

$$J(x) = \sum_{i=1}^{m} \alpha_i f_i(x) \to \max_{x \in D}; \ \alpha_i > 0, \ \sum_{i=1}^{m} \alpha_i = 1. \qquad (2.3)$$

Весовые коэффициенты α_i могут при этом рассматриваться как показатели относительной значимости отдельных критериальных функционалов f_i. Чем большее значение мы придаем критерию f_j, тем больший вклад в сумму (2.3) он должен давать и, следовательно, тем большее значение α_j должно быть выбрано. При наличии существенно разнохарактерных частных критериев обычно бывает достаточно сложно указать окончательный набор коэффициентов α_i, исходя из неформальных соображений, связанных, как правило, с результатами экспертного анализа. Позже мы покажем, что, вообще говоря, априори неясно, в каком отношении должны находиться весовые коэффициенты α_i и α_j, если известно желательное соотношение между f_i и f_j в оптимальной точке (например, мы можем требовать, чтобы $f_i = 0, 1 f_j$).

Метод максиминной свертки. Обычно применяется в форме

$$J(x) = \min_i f_i(x) \to \max_{x \in D}.$$

Здесь, в отличие от метода линейной свертки, на целевой функционал $J(x)$ оказывает влияние только тот частный критерий оптимальности, которому в данной точке x соответствует наименьшее значение соответствующей функции $f_i(x)$. И если в случае (2.3), вообще говоря, возможны "плохие" значения некоторых f_i за счет достаточно "хороших" значений остальных целевых функционалов, то в случае максиминного критерия производится расчет "на наихудший случай", и мы по значению $J(x)$ можем определить гарантированную нижнюю оценку для всех функционалов $f_i(x)$. Этот факт расценивается как преимущество максиминного критерия перед методом линейной свертки.

При необходимости нормировки отдельных частных целевых функционалов, т. е. приведения во взаимное соответствие масштабов измерения значений отдельных $f_i(x)$, используется "взвешенная" форма максиминного критерия:

$$J(x) = \min_i \alpha_i f_i(x) \to \max_{x \in D}, \tag{2.4}$$

где весовые коэффициенты α_i удовлетворяют требованиям (2.3).

Подбирая различные значения α_i, можно определенным образом воздействовать на процесс оптимизации, используя имеющуюся априорную информацию. Приведем характерный пример.

Пример 2.1 (решение системы неравенств). Весьма часто в задачах оптимального выбора параметров реальных систем (в так называемых *задачах оптимального проектирования*) технические, экономические и другие требования к проектируемой системе выражаются в виде "условий работоспособности", имеющих форму неравенств вида

$$y_i(x) \le t_i, \ i = 1, ..., m. \tag{2.5}$$

Здесь функции $y_i(x)$ интерпретируются как частные показатели качества функционирования системы; $x = (x_1, ..., x_n)$ — вектор параметров, подлежащих выбору; t_i — допустимые верхние границы для заданных показателей качества (так называемые *контрольные показатели*). К форме (2.5) очевидным образом приводятся и обратные неравенства $z_k(x) \ge s_k$. Для этого достаточно положить $y_k = -z_k$, $t_k = -s_k$.

Для решения системы неравенств (2.5) методами теории оптимизации поступают следующим образом. Вводят так называемые *запасы* f_i, отражающие степень выполнения каждого из неравенств (2.5). Простейшая форма запаса имеет вид

$$f_i(x) = t_i - y_i(x). \tag{2.6}$$

Имеем, следовательно, многокритериальную задачу максимизации всех запасов:

$$f_i(x) \to \max_{x \in D}, \ i = 1, ..., m.$$

Максиминная свертка (максимизируется минимальный из запасов) приводит к следующей однокритериальной задаче:

$$J(x) = \min(t_i - y_i(x)) \to \max, \ x \in D.$$

При наличии весовых коэффициентов имеем задачу

$$J(x) = \min_i \alpha_i (t_i - y_i(x)) \to \max_{x \in D}. \tag{2.7}$$

Весовые коэффициенты α_i в функционале (2.7) выполняют функцию нормирования частных критериев по значению. Это можно реализовать, например, следующим образом. Для каждого из ограничений (2.5) задаются характерные значения $\delta_i > 0$, определяющие эквивалентные (с точки зрения лица, принимающего решения) приращения критериев f_i. Иначе говоря, утверждается, что увеличение критерия f_i на δ_i так же "хорошо", как и увеличение f_j на δ_j. В результате вместо задачи (2.7) будем иметь

$$J(x) = \min_i \left(\frac{t_i - y_i(x)}{\delta_i} \right) \to \max_{x \in D}. \tag{2.8}$$

Таким образом, каждая разность $t_i - y_i$ "измеряется" в специальных единицах, определяемых $\delta_i > 0$. В качестве δ_i для нормировки иногда используются значения $f_i(x^0)$ в заданной начальной точке x^0, какие-либо иные "характерные" значения $f_i(x)$ или сами значения t_i, если они не равны нулю. (Подобные соображения могут быть использованы и при выборе весовых коэффициентов в методе линейной свертки.)

Достаточно типичным для задач параметрической оптимизации, сформулированных в форме (2.5), можно считать случай, когда по условию задачи нежелательно делать некоторые из показателей, например $y_1(x)$, намного меньше, чем t_1. Требуется выполнение соответствующего неравенства (2.5), но с небольшим запасом. (Например, в транзисторных устройствах для правильного функционирования схемы может требоваться выполнение заданной степени насыщения транзистора, но значительное ее превышение нежелательно из-за увеличения времени переключения). В таких случаях можно воспользоваться регулирующими свойствами весовых коэффициентов. Именно, вместо задачи (2.8) решаем задачу

$$J(x) = \min_i \alpha_i' \left(\frac{t_i - y_i(x)}{\delta_i} \right) \to \max_{x \in D}, \tag{2.9}$$

причем выбираем α_1' много большим, чем α_i', $i = 2, \ldots, m$. Выбор достаточно большого весового коэффициента α_1' приводит к тому, что, с одной стороны, при нарушении первого неравенства

$$y_1(x) \le t_1 \tag{2.10}$$

мы имеем существенное ухудшение целевого функционала (2.9), т. к. разность $t_1 - y_1(x) < 0$, будучи умноженной на α_1', дает большое по абсолютной величине отрицательное число, определяющее значение

$$J(x) = \alpha_1' \frac{t_1 - y_1(x)}{\delta_1}.$$

С другой стороны, уже при незначительных положительных значениях запаса $f_1 = t_1 - y_1(x)$ он будет сравним с запасами работоспособности по остальным показателям качества. Следовательно, увеличение α_1' вносит некоторый стабилизирующий фактор. В результате соответствующее условие работоспособности с высокой вероятностью будет выполнено, с наличием в то же время небольшого положительного запаса в оптимальной точке.

2.2. Максиминные стратегии

Вернемся к введенным в *разд. 1.3* понятиям эффективного и слабо эффективного решения многокритериальной задачи

$$f_i(x) \to \max_{x \in D}, \quad i = 1, ..., m; \quad D \subseteq R^n.$$

Напомним, что в словесной формулировке эффективность решения $x^0 \in D$ означает, что оно не может быть улучшено по какому-либо показателю f_i без ухудшения ситуации по оставшимся показателям. Следовательно, если x^0 — эффективно (Парето-оптимально), то не существует других решений $x' \in D$, для которых справедливо:

$$f_i(x') \geq f_i(x^0), \; i = 1, \ldots, m, \tag{2.11}$$

где хотя бы одно из неравенств (2.11) — строгое. И аналогично под слабо эффективным решением мы понимаем решение, которое не может быть улучшено одновременно по всем показателям. На рис. 2.1 слабо эффективным решениям соответствуют "северная, северо-восточная и восточная части границы" множества достижимости $f(D)$ ($f(D)$ — это образ множества D для векторного отображения $f = (f_1, ..., f_m)$).

Иначе говоря, в данном примере множество слабо эффективных оценок $S(f)$ совпадает с множеством $[a, b] \cup [b, c] \cup [c, d]$. Множество $P(f)$ эффективных оценок равно $[b, c]$ и совпадает с "северо-восточной границей" множества $f(D)$.

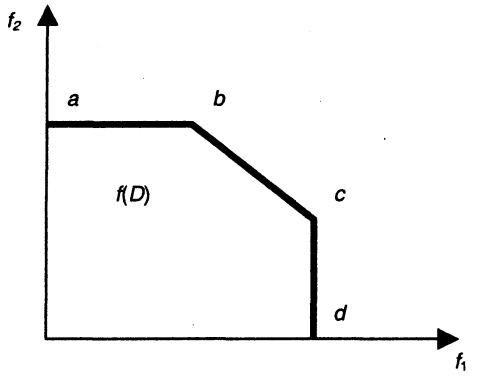


Рис. 2.1. Множество достижимости многокритериальной задачи

Основная задача данного и следующего разделов заключается в выяснении тех вычислительных средств, которые можно было бы использовать для построения аппроксимации множеств эффективных и слабо эффективных решений и оценок.

Справедлива следующая теорема.

Теорема 2.1. Пусть заданы произвольные числа $\alpha_i > 0$, $i = 1, ..., m$. Тогда решение задачи

$$\min_i \alpha_i (f_i - t_i) \to \max_{x \in D} \tag{2.12}$$

при любых фиксированных t_i есть слабо эффективный вектор. Наоборот, любой слабо эффективный вектор x^0 может быть получен как решение задачи (2.12) при некоторых $\alpha_i > 0$ и $t_i < f_i(x^0)$, $i = 1, ..., m$.

Доказательство. Прямое утверждение теоремы докажем от противного. Пусть x^0 есть решение задачи (2.12) и существует вектор $x' \in D$, для которого

$$f_i(x') > f_i(x^0), \ i = 1, ..., m,$$

что эквивалентно предположению о том, что вектор x^0 не является слабо эффективным. Тогда для любых наборов $\{\alpha_i > 0\}$, $\{t_i\}$ будем иметь (докажите это!)

$$\alpha_i(f_i(x') - t_i) > \alpha_i(f_i(x^0) - t_i), \ i = 1, ..., m$$

и, следовательно,

$$\min_i \alpha_i (f_i(x') - t_i) > \min_i \alpha_i (f_i(x^0) - t_i),$$

а это противоречит предположению о том, что x^0 есть решение задачи (2.12). Прямое утверждение теоремы доказано.

Докажем обратное утверждение. Пусть x^0 — слабо эффективный вектор: $x^0 \in S(D)$. Это означает, что не существует другого вектора x', для которого

$$\forall i : f_i(x') > f_i(x^0) \tag{2.13}$$

($\forall i$ — для всех номеров $i = 1, ..., m$).

По условию теоремы заданы такие $\{t_i\}$, что $\forall i : f_i(x^0) - t_i > 0$. Введем числа

$$\alpha_i' = \frac{1}{f_i(x^0) - t_i} > 0$$

и покажем, что

$$\max_{x \in D} \min_i \alpha_i'(f_i(x) - t_i) = \min_i \alpha_i'(f_i(x^0) - t_i) = 1, \tag{2.14}$$

т. е. что при выбранных коэффициентах α'_i максимум реализуется на векторе x^0. Этим самым теорема будет доказана.

Из (2.13) следует, что для любого вектора x', отличного от x^0, будет существовать такой номер $i = i_0$, для которого

$$f_{i_0}(x') \le f_{i_0}(x^0) \tag{2.15}$$

(это прямое следствие слабой эффективности вектора x^0). Из (2.15) получаем:

$$\alpha_{i_0}'(f_{i_0}(x') - t_{i_0}) \le \alpha_{i_0}'(f_{i_0}(x^0) - t_{i_0}) = 1$$

(напоминаем, что неравенства можно умножать на положительные числа α_{i_0}'). Но тогда и

$$\min_i \alpha_i'(f_i(x') - t_i) \le 1.$$

Таким образом, доказано, что для любого x', отличного от x^0,

$$\min_i \alpha_i'(f_i(x') - t_i) \le \min_i \alpha_i'(f_i(x^0) - t_i) = 1,$$

а значит, и максимум по x левой части последнего неравенства также не будет превышать единицы. Соотношение (2.14), а вместе с ним и теорема, доказаны.

Замечание 2.1

Если слабо эффективное решение x^0 получено как решение задачи (2.12) при каком-то наборе коэффициентов $\alpha_1, ..., \alpha_m$: $\alpha_i > 0$, то, очевидно, это же решение будет достигаться при любом наборе $k\alpha_1, ..., k\alpha_m$, где k — произвольное положительное число. Поэтому можно считать, что всегда выполнено условие нормировки:

$$\sum_{i=1}^{m} \alpha_i = 1. \tag{2.16}$$

В противном случае вместо коэффициентов α_i мы будем рассматривать другие коэффициенты:

$$\overline{\alpha}_i = \frac{\alpha_i}{\sum_{i=1}^{m} \alpha_i}.$$

В силу приведенного замечания будем далее предполагать, что условие (2.16) всегда выполнено.

Из доказанной теоремы следуют важные выводы. Будем для простоты считать, что все функционалы $f_1, ..., f_m$ исходно положительны, т. е. принимают во всех точках допустимого множества D строго положительные значения: $\forall x \in D: f_i(x) > 0$, $i = 1, ..., m$. Тогда для любого $x^0 \in S(D)$ будет выполнено условие $f_i > t_i$ при $t_i = 0$. Поэтому далее вместо задачи (2.12) будем рассматривать задачу

$$\begin{gathered} F(x,\alpha) \triangleq \min_i \alpha_i f_i(x) \to \max_{x \in D}; \\ \alpha = (\alpha_1, ..., \alpha_m), \ x = (x_1, ..., x_n). \end{gathered} \tag{2.17}$$

Обозначим множество решений задачи (2.17) при фиксированном наборе коэффициентов α через

$$X(\alpha) = \operatorname{Arg}\max_{x \in D} F(x, \alpha).$$

Согласно доказанной теореме, множество

$$\bigcup_{\alpha \in A} X(\alpha),$$

$$A = \left\{ \alpha = (\alpha_1, ..., \alpha_m) \middle| \alpha_i > 0, \ \sum_{i=1}^{m} \alpha_i = 1 \right\}$$

совпадает с множеством слабо эффективных решений:

$$\bigcup_{\alpha \in A} X(\alpha) = S(D).$$

Дадим геометрическую иллюстрацию доказанному утверждению для случая двух целевых функционалов f_1, f_2. Имеем

$$F(x, \alpha) = \min\{\alpha_1 f_1, \alpha_2 f_2\}.$$

Если рассматривать указанную зависимость в пространстве критериев, то получим функцию

$$\Phi(f_1, f_2) = \min\{\alpha_1 f_1, \alpha_2 f_2\}.$$

Построим линии уровня (линии постоянного значения) функции Ф на плоскости (f_1, f_2). Для этого рассмотрим прямую L, заданную уравнением

$$\alpha_1 f_1 = \alpha_2 f_2$$

при некотором фиксированном наборе $\{\alpha_1, \alpha_2\}$. График прямой $f_2 = \dfrac{\alpha_1}{\alpha_2} f_1$ показан на рис. 2.2.

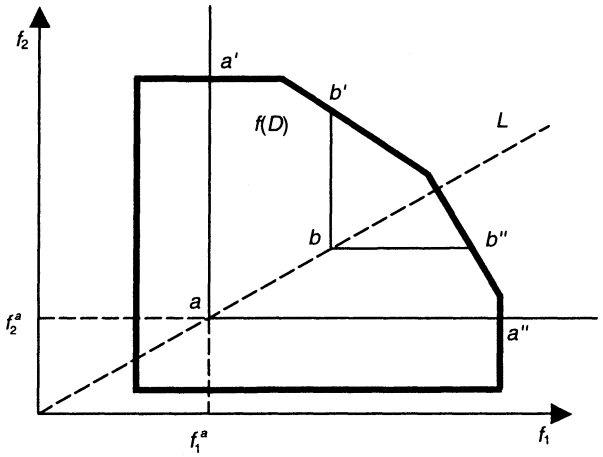


Рис. 2.2. Линии уровня функции минимума

В любой точке этой прямой, например, в точке $a = (f_1^a, f_2^a)$, будем иметь $\alpha_1 f_1^a = \alpha_2 f_2^a$. При смещении из точки a вправо параллельно оси f_1 получим $\alpha_1 f_1 > \alpha_2 f_2^a$. Аналогичная ситуация наблюдается и при перемещении вверх из точки a параллельно оси ординат: будем иметь $\alpha_2 f_2 > \alpha_1 f_1^a$. Поэтому, согласно определению функции Ф, ее линия уровня, соответствующая значению $\Phi = \alpha_1 f_1^a = \alpha_2 f_2^a$, будет совпадать с "уголком" ($a'\,a\,a''$) с верши-

ной в точке a, показанным на рис. 2.2 (естественно, данную линию уровня целесообразно рассматривать только в пределах множества достижимости $f(D)$). Следовательно, во всех точках отрезков $[a', a]$ и $[a, a'']$ функция Ф будет иметь одно и то же значение, совпадающее с ее значением в вершине "уголка", равным по построению $\alpha_1 f_1^a = \alpha_2 f_2^a$.

Легко видеть, что любой "уголок" подобного типа с вершиной, расположенной на прямой L, также будет линией уровня, соответствующей своему значению функции Ф. Причем при удалении вдоль прямой L от начала координат на "северо-восток" мы будем получать линии уровня, отвечающие бóльшим значениям Ф. Например, на рис. 2.2 показана линия уровня $(b' b b'')$, где $\Phi(b) > \Phi(a)$.

Таким образом, для каждого фиксированного набора весовых коэффициентов $\{\alpha_1, \alpha_2\}$ мы получаем целое семейство "уголковых" линий уровня функции Ф.

Ясно, что решению основной задачи (2.17) будет соответствовать наиболее удаленное от начала координат положение "уголка" (в пределах множества достижимости $f(D)$), которому соответствует максимальное возможное значение функции Ф, а значит, и F. На рис. 2.3 показано множество слабо эффективных оценок (отрезок $[c, d]$), полученных в результате решения задачи (2.17). На этом же рисунке показано решение $[c', d']$, полученное при другом наборе весовых коэффициентов, соответствующих прямой L'.

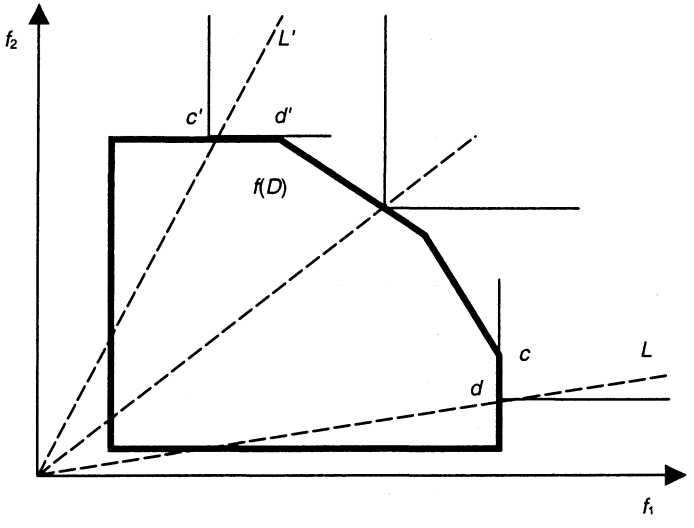


Рис. 2.3. Решения для разных наборов весов

Продолжая такие построения, легко убедиться, что, перебирая всевозможные $\alpha \in A$, можно получить "северную", "северо-восточную" и "восточную" части границы множества достижимости $f(D)$:

$$S(f) = [a, d'] \cup [d', c] \cup [c, b].$$

Это и отвечает основному содержанию сформулированной теоремы.

Здесь важно отметить, что задачи оптимизации типа (2.17) могут иметь не единственное решение. Так, для значения α, отвечающего прямой L, мы в качестве решения получим целое множество $[c, d]$ слабо эффективных оценок и соответствующих им слабо эффективных решений исходной многокритериальной задачи. Каждое из этих решений, вообще говоря, должно быть найдено.

Построенные на основе максиминной свертки вычислительные процедуры обычно подразумевают задание некоторой сетки в пространстве весовых коэффициентов A. Далее для полученного конечного множества наборов весовых коэффициентов

$$\alpha^1 = (\alpha_1^1, ..., \alpha_m^1);$$

$$.....$$

$$\alpha^N = (\alpha_1^N, ..., \alpha_m^N)$$

решается множество соответствующих однокритериальных задач (2.17) или (2.12). В результате приходим к построению требуемой аппроксимации множеств $S(D)$ и $S(f)$. Пользователь соответствующей программной системы обычно имеет возможность влиять на указанный процесс, управляя в той или иной мере выбором весовых коэффициентов. Последнее позволяет, в частности, получать более точные аппроксимации отдельных участков границ, представляющих наибольший интерес.

2.3. Метод линейной свертки и главного критерия. Лексикографическая оптимизация

Метод линейной свертки мы уже рассматривали на эвристическом уровне (так же, как и метод максиминной свертки). Здесь мы дадим более точные утверждения, касающиеся свойств получаемых решений.

Теорема 2.2. Пусть $\alpha \in A$. Тогда решение задачи

$$F_1(x,\alpha) \triangleq \sum_{i=1}^{m} \alpha_i f_i(x) \to \max_{x \in D} \tag{2.18}$$

есть эффективный вектор.

Доказательство. Пусть $x^0 \in D$ есть решение задачи (2.18) и существует такой $x' \in D$, что $f_i(x') \geq f_i(x^0)$, а для $i = i_0$ имеем $f_{i_0}(x') > f_{i_0}(x^0)$. Тогда

$$\sum_{i=1}^{m} \alpha_i f_i(x') > \sum_{i=1}^{m} \alpha_i f_i(x^0)$$

и, следовательно, x^0 не максимизирует функционал F_1. Полученное противоречие доказывает, что точки x' со сформулированными выше свойствами не существуют и поэтому x^0 — эффективный вектор. Теорема доказана.

Замечание 2.2

Обратное утверждение без дополнительных предположений неверно. Существуют эффективные векторы, не являющиеся решениями задачи (2.18). Для доказательства этого утверждения достаточно привести хотя бы один опровергающий пример, что и будет сделано далее.

Таким образом, согласно доказанной теореме

$$\bigcup_{\alpha \in A} X(\alpha) \subseteq P(D).$$

Здесь

$$X(\alpha) \triangleq \operatorname{Arg}\max_{x \in D} F_1(x, \alpha).$$

Обратимся снова к геометрическим иллюстрациям для $m = 2$. В этом случае

$$F_1(x, \alpha) = \alpha_1 f_1(x) + \alpha_2 f_2(x) = \Phi_1(f_1, f_2),$$

где функция Φ_1 считается определенной в пространстве критериев (f_1, f_2). Построим линии уровня функции Φ_1:

$$\alpha_1 f_1 + \alpha_2 f_2 = \text{const}. \tag{2.19}$$

Набор коэффициентов $\alpha = (\alpha_1, \alpha_2)$ считается фиксированным (неизменным в процессе всего рассмотрения). Графики прямых (2.19) для различных констант в правой части и различных весовых коэффициентов показаны на рис. 2.4.

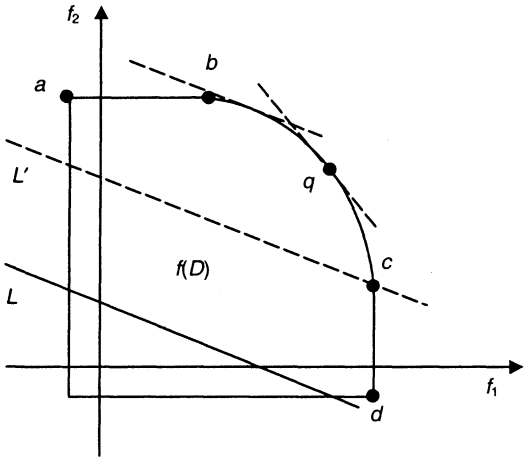


Рис. 2.4. Линии уровня функции Φ_1

Угловой коэффициент наклона прямой L определяется числами α_1, α_2 и равен $(-\alpha_1 / \alpha_2)$. При увеличении константы в правой части уравнения (2.19) прямая перемещается вверх параллельно L (занимая положение L'). Таким образом, мы имеем целое семейство линий уровня, и максимум функции Φ_1, а вместе с ней и F_1, достигается в точках плоскости (f_1, f_2), соответствующих точкам касания (но не пересечения) самой "верхней" линии уровня и множества достижимости $f(D)$. На рис. 2.4 точка b с координатами (f_1^b, f_2^b) реализует найденную рассмотренным методом эффективную оценку. Легко видеть, что ни одна из точек интервалов $[a, b)$, $(c, d]$, соответствующих слабо эффективным, но не эффективным решениям, не может являться точкой касания $f(D)$ и какой-либо линии уровня функции Φ_1 (угловой коэффициент $(-\alpha_1 / \alpha_2)$ не может равняться нулю или бесконечности, т. к. все $\alpha_i > 0$ и их величина ограничена сверху единицей).

На рис. 2.4 показана точка q, являющаяся решением задачи (2.18) при другом наборе α весовых коэффициентов. Перебирая различные α, можно (как и в случае максиминной свертки) получить достаточно точную аппроксимацию множеств $P(f)$ и $P(D)$.

Ситуация, связанная с существованием эффективных решений, не являющихся решениями задачи (2.18) ни при каких α, проиллюстрирована на рис. 2.5.

Все точки дуги a, b являются эффективными оценками, но ни одна из них (кроме самих точек a и b) не может являться точкой касания линий уровня функции Φ_1 к множеству $f(D)$ ни при каком наборе коэффициентов α. Таким образом, ясно, что отсутствие выпуклости множества $f(D)$ приво-

дит к принципиальным затруднениям при применении метода линейной свертки. Аналогично, наличие строго прямолинейных участков "северо-восточной" границы множества $f(D)$ может приводить к похожим трудностям из-за приближенного характера вычислений (точки внутри таких прямолинейных участков оказываются "неустойчивыми" решениями задачи (2.18)).

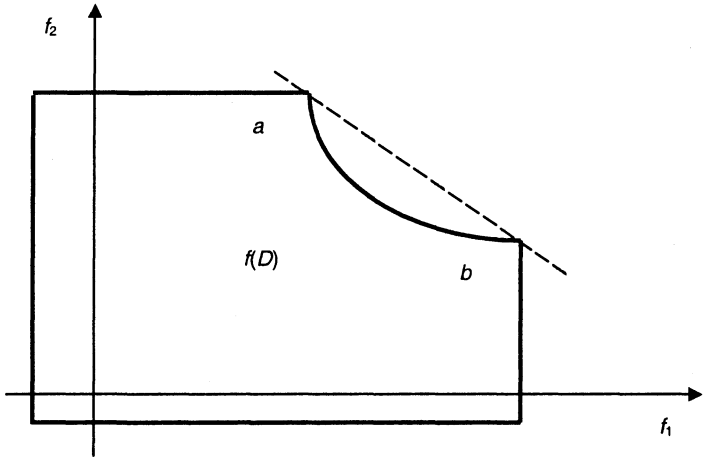


Рис. 2.5. Случай невыпуклой границы множества достижимости

Замечание 2.3

Весьма часто при эвристическом выборе весовых коэффициентов в методе линейной свертки пытаются сразу определить желаемую эффективную точку, исходя из заданных оценок критериев $f_1, ..., f_m$, по "важности". Так, например, полагая, что критерий f_2 существенно "важнее" чем критерий f_1, мы бы желали в качестве единственного решения многокритериальной задачи получить эффективную точку a на рис. 2.6. Однако мы не знаем при этом, на сколько коэффициент α_2 должен превышать значение α_1, чтобы была получена именно искомая точка. На рис. 2.6 показана ситуация, когда $\alpha_2 > \alpha_1$ и в то же время найденной точке b соответствуют значения $f_1^b > f_2^b$!

Мы здесь везде надеемся, что читатель понимает иллюстративный характер приводимых рисунков и графиков. На самом деле, при решении многокритериальных задач графическая информация полностью отсутствует, и мы имеем дело с чисто аналитическими постановками соответствующих оптимизационных задач.

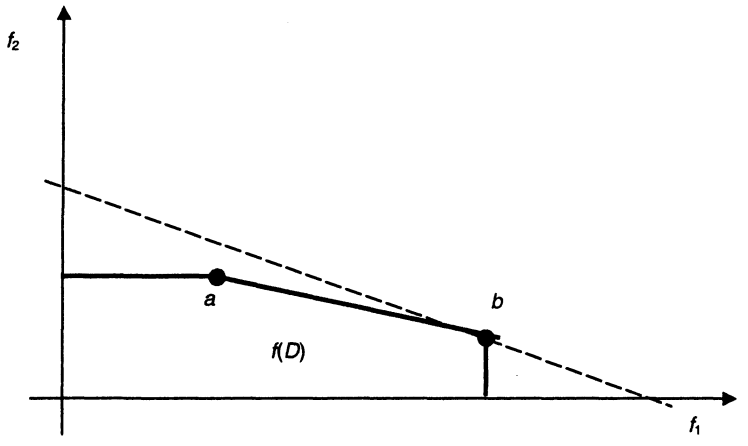


Рис. 2.6. Эвристический выбор весовых коэффициентов

Метод главного критерия также можно проинтерпретировать с помощью понятия слабо эффективного решения.

Теорема 2.3. Решение задачи

$$f_1(x) \to \max,\ x \in D', \tag{2.20}$$

где множество

$$D' = \{x \in D \mid f_i(x) \geq t_i,\ i = 2, ..., m\}$$

не пустое, есть слабо эффективный вектор.

Доказательство. Пусть x^0 есть решение задачи (2.20) и существует $x' \in D$, такой, что

$$f_i(x') > f_i(x^0),\ i = 1, ..., m. \tag{2.21}$$

Тогда $x' \notin D'$, т. к. в противном случае это противоречило бы свойству $f_1(x^0) \geq f_1(x)$ для $\forall x \in D'$. Следовательно, $x' \notin D'$ и поэтому существует такой номер $i = i_0$, для которого $f_{i_0}(x') < t_{i_0}$, что противоречит предположению (2.21). Теорема доказана.

Теорема 2.4. Любой эффективный вектор может быть получен как решение задачи (2.20) при некоторых t_i, $i = 2, ..., m$.

Доказательство. Пусть $x^0 \in P(D)$; примем $t_i = f_i(x^0)$, $i = 2, 3, ..., m$, и покажем, что

$$f_1(x^0) = \max f_1(x),\ x \in D'. \tag{2.22}$$

Выберем произвольный $x' \in D'$; тогда

$$f_i(x') \geq f_i(x^0),\ i = 2, \ldots, m.$$

Если предположить, что $f_1(x') > f_1(x^0)$, то это будет противоречить свойству эффективности вектора x^0, следовательно, $f_1(x') \leq f_1(x^0)$, что эквивалентно (2.22). Теорема доказана.

Из доказанных теорем следует, что в качестве "главного" критерия может быть выбран любой из частных критериев. Независимо от этого выбора произвольное эффективное решение может быть получено как решение задачи (2.19) при соответствующем задании t_i.

Метод главного критерия допускает простую графическую иллюстрацию.

Рис. 2.7 отражает предположение, что в качестве "главного" выбран критерий f_1, а на значения функционала f_2 наложено ограничение $f_2 \geq t_2$. Образ $f(D')$ множества D' точек из D, удовлетворяющих указанному дополнительному ограничению, соответствует незаштрихованной части множества $f(D)$.

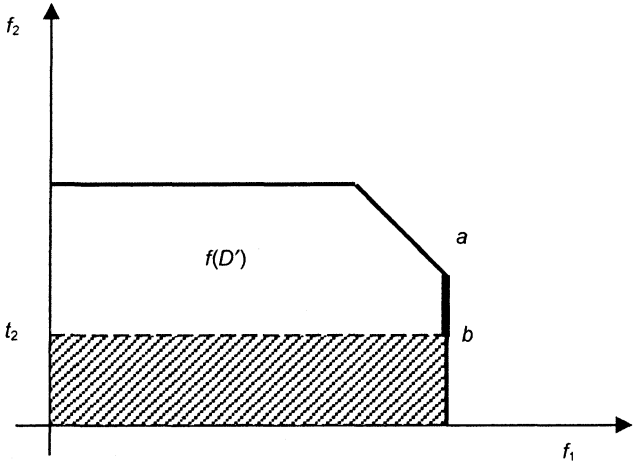


Рис. 2.7. Метод главного критерия

Максимизация критерия f_1 на множестве D', очевидно, приводит к построению отрезка $[a, b]$ на рис. 2.7. Как видно из рис. 2.7, задавая различные t_2, можно получить аппроксимацию "северо-восточной" и "восточной" частей границы множества $f(D)$, куда входят все эффективные решения задачи.

При выборе в качестве "главного" критерия f_2 мы аналогично построим "северную" и "северо-восточную" части границы.

Пусть известен диапазон изменения функционала f_i:

$$f_i^H \le f_i \le f_i^B, \quad i = 2, ..., m.$$

Тогда из доказательства последней теоремы следует, что соответствующие t_i (при работе с критерием f_1 как с "главным") должны меняться в тех же пределах, пробегая выбранную сетку значений (аналогично построению аппроксимаций множеств эффективных и слабо эффективных решений в методах максиминной и линейной сверток).

Лексикографическая оптимизация. Как мы видели, методом максиминной свертки могут быть получены все слабо эффективные решения, если вектор $\alpha = (\alpha_1, ..., \alpha_m)$ пробегает все множество векторов A. Напротив, по методу линейной свертки можно получить только эффективные решения, но не все. А. Джоффриону (A. M. Geoffrion) принадлежит идея выделения эффективных точек из множества $S(D)$, построенного методом максиминной свертки, с помощью процедуры максимизации линейной свертки. В результате будут построены все эффективные и только эффективные решения.

Теорема 2.5. Пусть $\alpha_i > 0$, $i = 1, ..., m$. Тогда решение задачи

$$\begin{aligned} &\sum_{i=1}^{m} f_i(x) \to \max_{x \in T_\alpha}; \\ &T_\alpha = \operatorname{Arg} \max_{x \in D} \min_i \alpha_i (f_i - t_i) \end{aligned} \tag{2.23}$$

есть эффективный вектор. Наоборот, любой эффективный вектор x^0 может быть получен как решение задачи (2.23) при некоторых $\alpha_i > 0$ и $t_i < f_i(x^0)$, $i = 1, ..., m$.

Доказывать теорему не будем. Обсудим ее смысл и дадим геометрическую иллюстрацию.

Множество T_α при фиксированном наборе коэффициентов α получают методом максиминной свертки. Можно доказать, что, если T_α — непустое множество, то оно обязательно наряду со слабо эффективными решениями содержит хотя бы один эффективный вектор.

Упражнение 2.1. Докажите последнее утверждение.

Далее, максимизируя линейную свертку частных критериев (2.23) на множестве T_α, мы получаем в результате эффективное решение. Совершенно очевидно, что если T_α состоит из одного элемента, то он же и будет эффективным решением и необходимость в максимизации (2.23), по существу, отпадает.

На рис. 2.8 показано множество T_α, найденное при определенном наборе коэффициентов α, и выделенная из него эффективная оценка e.

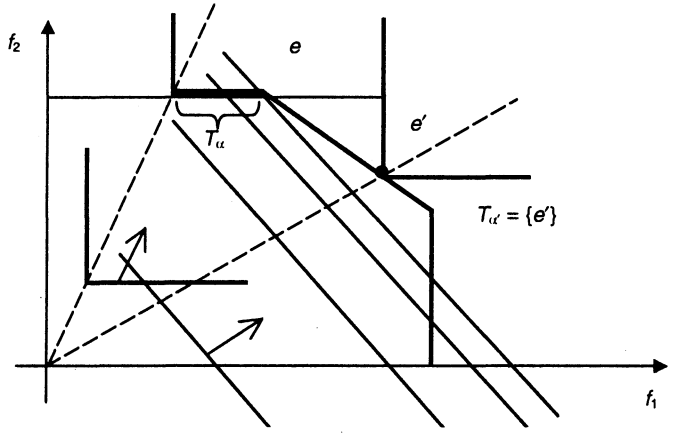


Рис. 2.8. Лексикографическая оптимизация

При другом наборе коэффициентов α' множество $T_{\alpha'}$ (см. рис. 2.8) является одноэлементным: $T_{\alpha'} = \{e'\}$. Точка e' также оказывается эффективной оценкой.

Термин "лексикографическая оптимизация" здесь использован в следующем смысле. Речь идет о так называемом лексикографическом упорядочении двух критериев — максиминного и линейного: вначале "срабатывает" максиминная свертка, и если в результате получен неоднозначный результат (множество T_α содержит более одного элемента), то выбираются тот или те из полученных элементов, которые максимизируют линейную свертку (2.23), т. е. только тогда "срабатывает" второй критерий. Данный процесс аналогичен поиску слова в словаре: сначала мы работаем с первой буквой, затем — со второй и т. д. (этим и определяется название).

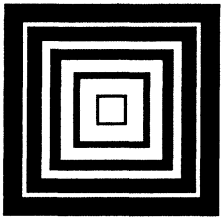
Задача многокритериальной оптимизации (2.23) часто записывается в следующем компактном виде:

$$\left\{\min_i \alpha_i (f_i - t_i),\ \sum_{i=1}^{m} f_i(x)\right\} \to \max_{x \in D}.$$

Выводы. Ни один из методов, представленных выше, не позволяет выделить единственное оптимальное решение. Решения, соответствующие различным наборам весовых коэффициентов, являются равноправными элементами множеств эффективных и слабо эффективных решений, ко-

торые согласно общей постановке задачи ПР реализуют ядра соответствующих бинарных отношений (отношений Парето и отношений Слейтера), т. е. и являются искомыми решениями. Однако с практической точки зрения, например, в задачах выбора вариантов (при покупке или заказе товаров, при выборе партнеров по бизнесу, при выборе вариантов программных средств и т. д.), а также в системах автоматизированного проектирования часто требуется выбрать единственное решение (проект). Для этого должна привлекаться некоторая дополнительная информация о предпочтениях лица, принимающего решения. Принцип Парето в этом смысле позволяет лишь сузить класс возможных претендентов на решение и исключить из рассмотрения заведомо неконкурентоспособные варианты.

Методы выбора единственного решения многокритериальной задачи существуют и связаны с использованием моделей и процедур, предназначенных для структуризации и количественного описания субъективного мнения лица, принимающего решения. В *главе 5* эти вопросы будут рассмотрены более подробно.



Глава 3

Принятие решений в условиях неопределенности

Мы ограничимся здесь обсуждением нескольких стандартных ситуаций неопределенности обстановки в процессе принятия решений.

По-прежнему считаем, что задано множество альтернатив X и множество возможных исходов Y. При рассмотрении задачи ПР в условиях определенности, когда каждому решению $x \in X$ соответствовал единственный исход $y \in Y$, было безразлично, на каком множестве X или Y задавать бинарное отношение предпочтения R. Ранее мы обычно задавали это отношение на множестве решений X. При этом на самом деле неявно задавалось и соответствующее отношение на множестве исходов Y. Действительно, пусть существует однозначная зависимость $y = \varphi(x)$, позволяющая по решению x определить единственный исход y. Альтернативы x' и x'', по существу, сравнивались по значениям оценок соответствующих исходов y' и y''. Иначе говоря, имели

$$x' \succ x'' \leftrightarrow y' \succ y'',$$

где использованы обозначения

$$x' \succ x'' \leftrightarrow (x', x'') \in R_X,$$

$$y' \succ y'' \leftrightarrow (y', y'') \in R_Y.$$

Таким образом, общая модель принятия решений $<A, R>$ могла быть сформулирована в виде $<X, R_X>$ либо $<Y, R_Y>$. Суть дела от этого не менялась. В таком случае говорят, что отображение $\varphi : X \to Y$ является *гомоморфизмом* модели $<X, R_X>$ в модель $<Y, R_Y>$, т. е. для всяких x', $x'' \in X$ из $x' R_X x''$ следует $\varphi(x') R_Y \varphi(x'')$.

При наличии неопределенных факторов ситуация усложняется. Теперь мы уже не можем гарантировать наступление определенного исхода y при выборе решения x.

Будем считать, что наша система предпочтений связана с оценкой "полезностей" исходов y. Выбор x осуществляется с единственной целью — получить "хороший" исход y, принадлежащий ядру — множеству максимальных элементов из Y по заданному отношению R_Y. В данном случае модель ПР имеет вид $<Y, R_Y>$. В частности, отношение R_Y может быть задано, хотя это не единственный способ, как и раньше, с помощью однокритериальной или многокритериальной системы оценок исходов, т. е. на критериальном языке описания выбора.

3.1. Основные понятия

В случае, когда множества альтернатив X и исходов Y конечны, ситуацию выбора альтернативы в условиях неопределенности можно представить с помощью матрицы, называемой *матрицей решений* (табл. 3.1).

Таблица 3.1. *Матрица решений*

X	Z				
	z_1	...	z_j	...	z_m
x_1	y_{11}	...	y_{1j}	...	y_{1m}
...	...	...	...	...	...
x_j	y_{j1}	...	y_{jj}	...	y_{jm}
...	...	...	...	...	...
x_n	y_{n1}	...	y_{nj}	...	y_{nm}

Здесь $X = \{x_1, ..., x_n\}$, $Y = \{y_{11}, ..., y_{nm}\}$. Вектор $Z = \{z_1, ..., z_m\}$ описывает неопределенность обстановки и также предполагается конечным. По существу, имеется функция двух аргументов: $y = F(x, z)$, $F : X \times Z \to Y$.

Заданная матрица интерпретируется следующим образом. Если мы выбрали решение x_j, то могут реализоваться различные исходы из соответствующей строки матрицы: y_{j1}, ..., y_{jm}. Какой именно исход реализуется, зависит от значения *параметра неопределенности* z, который может иметь различный содержательный смысл.

Будем различать две основные ситуации:

1. Вектор Z отражает так называемые "природные" неопределенности, т. е. неопределенность "состояния природы" в момент принятия решения.
2. Множество $Z = \{z_1, ..., z_m\}$ есть множество альтернатив, на котором (одновременно с нами) осуществляет выбор решения второй субъект, руководствуясь своим отношением предпочтения R_y (неопределенность типа "активный партнер"). При этом выбираемое нами решение x, в свою очередь, характеризует неопределенность обстановки для второго субъекта.

Далее эти вопросы рассматриваются более подробно. Заметим здесь же, что представление задачи ПР с помощью табл. 3.1 имеет достаточно общий характер, в частности, включает в себя случай полной определенности. При этом таблица будет состоять из одного столбца (что эквивалентно наличию одного состояния среды).

В общем случае мы будем предполагать существование функции

$$y = F(x, z),$$

где $x \in X$, $z \in Z$, $y \in Y$; X, Z, Y — множества (вообще говоря, абстрактные) альтернатив (решений), состояний среды и исходов соответственно. Особенностью рассматриваемых ниже задач ПР является предположение о неизвестном в момент принятия решения значении параметра z. Саму функцию F будем называть *функцией реализации*. Таким образом, функция реализации ставит в соответствие каждой паре вида (x, z), где x — альтернатива, а z — состояние фактора неопределенности, исход y.

При этом, как указывалось, характер неопределенности, описываемый переменной z, может быть различным. Вначале мы будем предполагать, что z описывает некоторую неопределенность среды (неопределенность типа "активный партнер" рассмотрена в *разд. 3.5*). Кроме того, мы будем предполагать, что каждый исход y оценивается вещественным числом e (отражающим, если угодно, "полезность" исхода), и требуется максимизировать эту оценку. (Имеем, следовательно, однокритериальную оценку исхода.) Для упрощения обозначений мы в таких случаях будем отождествлять y и e, считая, что исход y уже есть некоторая числовая оценка принятого решения. При этом функция реализации преобразуется в соответствующую вещественную функцию (целевую функцию) $J(x, z)$, которую следует максимизировать или минимизировать по x в зависимости от смысла решаемой задачи.

В указанной ситуации вполне естественно ввести следующее бинарное отношение доминирования R_1 на множестве X:

$$x_1 \overset{R_1}{\succ} x_2 \leftrightarrow \forall z \in Z : J(x_1, z) \geq J(x_2, z),$$

причем хотя бы для одного z имеем строгое неравенство. (Здесь $\succ$ — знак строгого доминирования). Отношение эквивалентности может быть введено с помощью соотношения:

$$x_1 \sim x_2 \leftrightarrow \forall z \in Z : J(x_1, z) = J(x_2, z)$$

($\sim$ — знак эквивалентности).

Пример 3.1. Рассмотрим числовую матрицу решений, где $Z = \{z_1, z_2\}$, $X = \{x_1, ..., x_5\}$. В этом случае каждому решению $x_i \in X$ соответствуют две числовые оценки полезности

$$y_{1i} = J(x_i, z_1);$$

$$y_{2i} = J(x_i, z_2),$$

отвечающие двум возможным состояниям среды. Таким образом, можно рассматривать значения y_{1i}, y_{2i} как координаты точки x_i в пространстве y_1, y_2. Пяти возможным решениям будут соответствовать пять точек в плоскости (y_1, y_2). Абсцисса каждой точки есть результат соответствующего решения при состоянии среды $z = z_1$, а ордината — при $z = z_2$ (рис. 3.1).

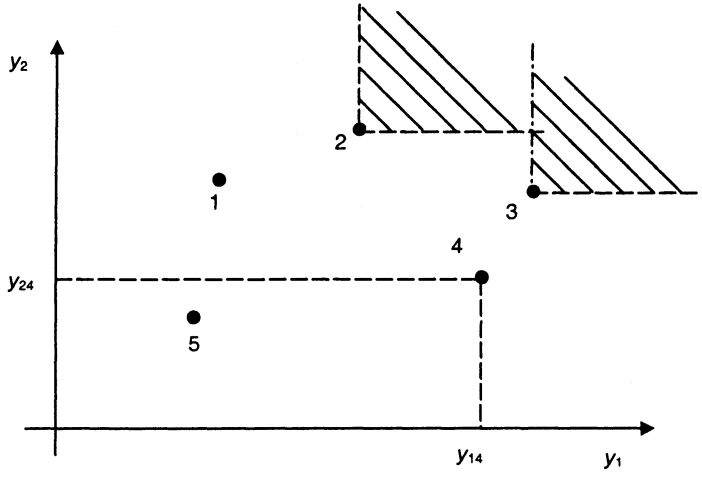


Рис. 3.1. Принцип доминирования строк в матрице решений

Совершенно очевидно, что здесь, так же как и в многокритериальных задачах, действует принцип Парето, и нас должны интересовать только недоминируемые в смысле отношения R_1 решения x_i. На рис. 3.1 это бу-

дут решения x_2, x_3. Остальные решения можно отбросить и в дальнейшем не учитывать.

Рассмотренный пример показывает, что все основные принципы и методы паретовского анализа многокритериальных задач переносятся на однокритериальные задачи принятия решений в условиях неопределенности. В частности, на основе принципа Парето исходное множество альтернатив X должно быть сокращено с целью удаления доминируемых по Парето вариантов. В общем случае это можно выполнить с помощью уже рассмотренных ранее численных методов выделения Парето-оптимальных решений.

Далее мы обычно будем предполагать конечность множеств X, Y, Z и задавать функцию реализации с помощью соответствующей матрицы решений. Однако большинство рассматриваемых методов допускают обобщение на бесконечномерный случай.

При рассмотрении методов принятия решений в условиях неопределенности используется понятие *оценочной функции*. Очевидно, что если принятие решений происходит в условиях определенности, то матрица решений будет содержать только один столбец. Принятие решений в условиях неопределенности состоит, по существу, тоже в формировании одностолбцовой матрицы решений и сведении задачи к случаю полной определенности. Эта процедура выполняется неоднозначно с помощью применения различных оценочных функций.

Пусть задана $(n \times m)$-матрица решений $\{y_{ij}\}$. *Оценочной функцией* называется вектор-функция Ψ, преобразующая эту матрицу в одностолбцовую матрицу $\{y_i\}$:

$$y_i = \Psi_i(y_{11}, y_{12}, \ldots, y_{nm}),$$

т. е. y_i зависит от всех элементов исходной матрицы. Многие методы принятия решений имеют оценочные функции вида

$$y_i = \Psi(y_{i1}, \ldots, y_{im}),$$

когда i-й элемент одностолбцовой матрицы зависит только от элементов i-ой строки исходной матрицы решений.

Например, оценочная функция Ψ может быть задана с помощью соотношения

$$y_i = \min_j y_{ij} + \max_j y_{ij}.$$

После построения оценочной функции выбор наилучшей альтернативы x^* производится из условия максимума или минимума значения оценочной функции (в зависимости от интерпретации элементов матрицы решений — "доходы" или "потери"), например,

$$i^* = \arg\min_i y_i, x^* = x_{i^*}.$$

Без существенного ограничения общности можно полагать, что всякое решение в условиях неполной информации — сознательно или неосознанно — принимается в соответствии с некоторой оценочной функцией. Выбор самой оценочной функции — это неформальный акт, и этот выбор всегда должен осуществляться с учетом качественных характеристик ситуации, в которой принимается решение.

Далее мы рассмотрим классические критерии принятия решений на основе различных оценочных функций.

3.2. Принятие решений в условиях риска

Напомним, что, говоря о принятии решений в условиях риска, обычно предполагают, что каждой альтернативе соответствует свое распределение вероятностей на множестве исходов. Если множества альтернатив и исходов конечны, то считаются известными вероятности всех исходов, возможных при выборе данной альтернативы.

Типичную постановку задачи о принятии решений в условиях риска поясним с помощью конкретного примера.

Пример 3.2 (задача о замене вратаря). На последней минуте хоккейного матча при ничейном счете тренер команды должен принять решение о замене вратаря шестым полевым игроком. Статистика, имеющаяся у тренера, показывает, что в аналогичных условиях в предыдущих встречах замена вратаря в одной шестой части случаев привела к выигрышу, в половине случаев — к ничьей и в одной трети случаев — к поражению. Если же вратарь не заменялся, то в 7/8 случаев встреча заканчивалась вничью, а в 1/8 части случаев команда проигрывала.

Построим для этой задачи граф связей альтернатив и исходов. Здесь имеются две альтернативы: x_1 — заменить вратаря, x_2 — не делать замены. В любом случае возможны три исхода: выигрыш (В), ничья (Н) и поражение (П). Принимая за вероятность каждого исхода частоту его появления в предыдущих матчах, получим граф, представленный на рис. 3.2. Решение задачи будет дано несколько позже.

Сформулированная задача ПР в условиях риска и приведенный пример не позволяют пока понять, где же здесь состояние среды? Какой характер имеет функция реализации $F(x, z)$ и возможно ли вообще ее построение? Оказывается, что язык функций реализации является достаточно общим и позволяет описывать различные ситуации неопределенности, в том числе и рассмотренную выше.

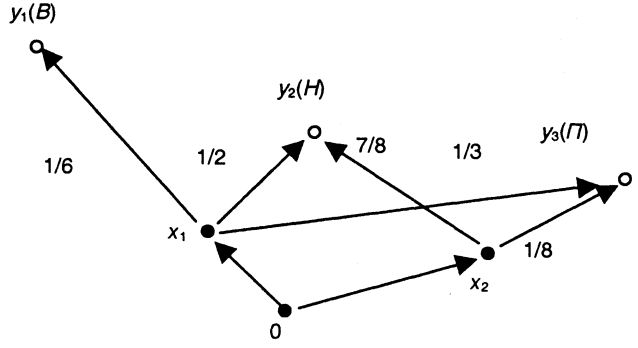


Рис. 3.2. Задача о замене вратаря

Обратимся снова к задаче о замене вратаря. Задание функции реализации означает, что при известном z мы по каждому x уже однозначно определяем исход y. Таким образом, зная состояние среды z, мы должны точно знать, что будет, если мы выберем альтернативу x_1, и каков будет исход при выборе x_2. Введем следующие шесть искусственных "состояний среды":

z_1: $x_1 \to$ В, $x_2 \to$ Н $\quad p(z_1) = 1/6 \times 7/8 = 7/48$

z_2: $x_1 \to$ Н, $x_2 \to$ Н $\quad p(z_2) = 1/2 \times 7/8 = 7/16$

z_3: $x_1 \to$ П, $x_2 \to$ Н $\quad p(z_3) = 1/3 \times 7/8 = 7/24$

z_4: $x_1 \to$ В, $x_2 \to$ П $\quad p(z_4) = 1/6 \times 1/8 = 1/48$

z_5: $x_1 \to$ Н, $x_2 \to$ П $\quad p(z_5) = 1/2 \times 1/8 = 1/16$

z_6: $x_1 \to$ П, $x_2 \to$ П $\quad p(z_6) = 1/3 \times 1/8 = 1/24$

В крайнем правом столбце указаны вероятности соответствующих событий.

Теперь функция реализации может быть задана в виде табл. 3.2. Около каждого состояния среды указана вероятность его появления.

Таблица 3.2. *Матрица решений с искусственными состояниями среды*

X	Z					
	z_1 (7/48)	z_2 (7/16)	z_3 (7/24)	z_4 (1/48)	z_5 (1/16)	z_6 (1/24)
x_1	В	Н	П	В	Н	П
x_2	Н	Н	Н	П	П	П

Рассмотрим теперь задачу ПР в более общем случае, когда имеется n альтернатив $x_1, ..., x_n$ и L исходов $y_1, ..., y_L$. В качестве "состояния среды" возьмем множество возможных согласно графу связей альтернатив и исходов отображений $z_j : X \to Y, j = 1, ..., S$. В случае конечных множеств X и Y будем иметь $S = \prod_{j=1}^{n} S_j$, где S_j — количество стрелок, исходящих из альтернативы x_j, на графе связей альтернатив и исходов (в нашем примере $S_1 = 3$, $S_2 = 2$, $S = 6$). Таким образом, каждое "состояние среды" z_j соответствует такому подграфу графа связей альтернатив и исходов (будем называть его *подграфом состояния*), в котором из каждой альтернативы x_i исходит только одна стрелка, указывающая, какой исход будет реализован при выборе альтернативы x_i (S — максимальное возможное число таких подграфов). Следовательно, выбор "состояния среды" z_j и альтернативы x_i полностью определяет исход — обозначим его через $y_j(x_i)$. Далее, каждому состоянию среды z_j соответствует вероятность его наступления (вероятность реализации соответствующего подграфа состояния):

$$p(z_j) = \prod_{i=1}^{n} p_i(y_j(x_i)), \quad j = 1, ..., S,$$

где $p_i(y_j(x_i))$ — заданная вероятность наступления исхода y_j при выборе альтернативы x_i. Таким образом, для вычисления $p(z_j)$ достаточно перемножить числа, стоящие около стрелок, составляющих подграф состояния z_j. Теперь таблица, представляющая функцию реализации, уже может быть построена.

Установленная выше возможность представления задачи ПР в условиях риска в форме функции реализации означает, что статистическую неопределенность, проявляющуюся в неоднозначной (вероятностной) связи между средством и результатом, всегда можно интерпретировать как существование некоторой среды, оказывающей влияние на результат. Методологическое значение этого факта состоит в том, что достаточно широкий класс задач ПР может быть приведен к указанной стандартной форме — функции реализации. Отметим также, что многие практические задачи ПР непосредственно формулируются в форме функции реализации. Это, прежде всего, такие задачи, где реально существует среда, влияющая на результат принятия того или иного решения. В качестве примера могут быть указаны задачи принятия оптимальных проектных решений в условиях технологического разброса параметров изделия.

Итак, пусть задана функция реализации $y = F(x, z)$, где множества X, Y, Z уже не будем предполагать конечными. В условиях полной определенности, как мы видели, задана однозначная связь $y = \varphi(x)$, которая, очевидно, и является соответствующей функцией реализации ("состояние сре-

ды" z задано и фиксировано). Основная задача ПР состоит в поиске ядра бинарного отношения R_Y в множестве исходов Y.

Будем считать, что задана функция $f: Y \to R$, отображающая множество исходов Y на множество вещественных чисел R. Бинарное отношение R_Y задается условием

$$(y', y'') \in R_Y \text{"} f(y') > f(y'').$$

Тогда существует функционал $J: X \times Z \to R$ и задача ПР эквивалентна задаче оптимизации

$$J(x, z) \to \max_{x \in X}. \tag{3.1}$$

В данном случае у функционала J появился новый аргумент z, т. к. вместо $y = \varphi(x)$ имеем в условиях риска в качестве функции реализации зависимость $y = F(x, z)$.

Таким образом, мы использовали здесь критериальный язык для задания бинарного отношения предпочтения на множестве исходов Y. Более того, исходы y оцениваются в данном случае по однокритериальной схеме, т. к. задана одна функция $f(y)$ (целевая функция), характеризующая "полезность" исходов.

Таким образом, говоря о задаче ПР, сформулированной в виде (3.1), мы имеем в виду выбор решения (альтернативы) x в условиях, когда целевая функция задана, но задана не совсем точно — она содержит неопределенный параметр z. Решая задачу (3.1), мы можем определить x лишь как некоторую функцию параметра z: $x = x(z)$. Если никакой информацией о факторе неопределенности z мы не располагаем, то и результат максимизации J произволен. При наличии статистической неопределенности мы предполагаем, что z — случайная величина, закон распределения которой известен.

Методологически важно различать две основные ситуации:

1. Исход $y \in Y$, соответствующий принятому решению x, реализуется многократно.
2. Исход y реализуется однократно.

Например, выбор конструктивных параметров x изделия, выпускаемого серийно, дает пример многократной реализации исхода одного и того же выбора. Напротив, оптимальный выбор параметров уникального изделия — пример второй ситуации.

Обратимся к методам ПР при наличии многократно реализованного исхода. В этих случаях задачу (3.1) естественно заменить некоторой вероятностной задачей. Вполне разумным представляется выбор такой

альтернативы x, которая максимизирует математическое ожидание критерия, т. е. является решением задачи

$$J_1(x) = \overline{J(x,z)} \to \max_{x \in X}, \tag{3.2}$$

где черта сверху означает математическое ожидание случайной величины $J(x, z)$. Правило выбора оптимальной альтернативы на основе решения задачи оптимизации (3.2) называется *критерием математического ожидания (или критерием Байеса—Лапласа)*. Если предположить, что функционал J характеризует "полезность" или "доход", полученный от решения x и реализовавшегося исхода y, то математическое ожидание можно рассматривать как "средний доход", и, решая задачу (3.2), мы фактически максимизируем "средний доход".

Пример 3.3. Вернемся к ситуации, описанной в примере В.3 из *Введения*. Обозначим через p вероятность появления контролера (вероятность его непоявления равна, следовательно, $1 - p$). Функция $J(x, z)$ может быть представлена в виде матрицы доходов (табл. 3.3). (Перед потерями поставлен знак минус.)

Таблица 3.3. *Матрица доходов для задачи о контролере*

X	Z	
	$z_1(p)$	$z_2(1-p)$
x_1	–2	–2
x_2	–8	0

В табл. 3.3 $J(x_1, z_1) = -2$ и т. д. Имеем теперь:

$$\overline{J(x_1,z)} = p(-2) + (1-p)(-2) = -2;$$
$$\overline{J(x_2,z)} = p(-8) + (1-p)\cdot 0 = -8p.$$

Следовательно, согласно критерию (3.2), надо предпочесть первую альтернативу x_1 ("брать билет") второй, если $-2 > -8p$, т. е. $p > 1/4$. В противном случае более предпочтительной следует признать альтернативу x_2. Если считать, что каждый вагон имеет одинаковые шансы посещения контролером, число вагонов равно k, а число контролеров равно r (предполагается, что $r \le k$), то можно положить, что $p \cong r / k$. Таким образом, если на 4 трамвайных вагона приходится более 1 контролера, выгоднее брать билет!

Пример 3.4 (продолжение примера с задачей о замене вратаря). Будем численно оценивать исходы игры по получаемым очкам: В — 2 очка, Н — 1 очко, П — 0 очков. Тогда таблица, задающая функционал $J(x, z)$, получается непосредственно из табл. 3.2 и имеет вид табл. 3.4.

Таблица 3.4. *Матрица доходов для задачи о замене вратаря*

X	Z					
	z_1(7/48)	z_2(7/16)	z_3(7/24)	z_4(1/48)	z_5(1/16)	z_6(1/24)
x_1	2	1	0	2	1	0
x_2	1	1	1	0	0	0

Аналогично предыдущему примеру вычисляем:

$$\overline{J(x_1, z)} = 2(7/48) + 1(7/16) + 2(1/48) + 1(1/16) = 5/6;$$

$$\overline{J(x_2, z)} = 1(7/48) + 1(7/16) + 1(7/24) = 7/8.$$

Имеем $\overline{J(x_2, z)} > \overline{J(x_1, z)}$ и поэтому, руководствуясь критерием числа ожидаемых очков, принимаем решение, что в подобных ситуациях нецелесообразно заменять вратаря. "В среднем" такая стратегия приведет к успеху, хотя в каждой конкретной игре, конечно, может реализоваться любой возможный исход.

Упражнение. Указанный в последнем примере критерий (число очков) может быть неадекватен цели лица, принимающего решение. Легко представить себе ситуацию, когда выигрыш оценивают числом t, показывающим, во сколько раз выигрыш важнее ничьей (при этом может быть, что $t > 2$). Определите, при каком t выгоднее предпочесть альтернативу x_1 (заменить вратаря). (Ответ: $t \geq 2, 25$.)

Замена задачи $J(x, z) \to \max$ задачей $J_1 = M(J(x, z)) \to \max$, где $M(\ldots)$ — знак математического ожидания, — не единственный способ перехода к статистической постановке. Можно поступить и иначе. Например, определенную роль может играть дисперсия критериальной функции J. И, может быть, имеет смысл иногда поступиться немного значением математического ожидания для уменьшения возможного разброса результатов, т. е. уменьшения значения дисперсии:

$$J_2(x) = \overline{J(x, z)} - k\overline{[J(x, z) - \overline{J(x, z)}]^2} \to \max_x. \qquad (3.3)$$

Здесь $\overline{[J(x,z)-\overline{J(x,z)}]^2}$ — дисперсия случайной величины $J(x, z)$; k — заданная постоянная. Эту постоянную целесообразно интерпретировать как *степень несклонности к риску*. Действительно, k определяет "степень важности" дисперсии по отношению к математическому ожиданию случайной величины J. Увеличение значения k приводит, вообще говоря, к уменьшению "среднего дохода" $\overline{J(x,z)}$, но зато уменьшается и вероятность отклонения от "среднего дохода" (в том числе в сторону его уменьшения). Таким образом, чем больше k, тем менее склонно к риску лицо, принимающее решение. Критерий (3.3) обычно называется *критерием ожидаемого значения-дисперсии.*

Трудности решения задач (3.2), (3.3) связаны с высокой трудоемкостью процедуры вычисления математического ожидания. Мы должны сначала задать значения компонент вектора x и лишь затем провести усреднение — операцию, связанную с вычислением многомерных интегралов и поэтому требующую значительных затрат машинного времени. Иначе говоря, в отличие от детерминистской постановки задачи оптимизации, функционалы J_1, J_2 не заданы в явном виде как функции x. Все это часто заставляет заменять эти задачи на другие. Если решение задачи

$$J(x,z) \to \max_x$$

при фиксированном значении случайного параметра z находится сравнительно просто, то вместо критерия математического ожидания применяется следующий критерий:

$$J_3(x) = J(x,\overline{z}) \to \max_x,$$

где $\overline{z}$ — математическое ожидание случайной величины z. Здесь важно понимать, что задача максимизации функции J_3 вовсе не эквивалентна такой же задаче для J_1 из (3.2). Переход от J_1 к J_3 носит неформальный характер и требует каждый раз дополнительного обоснования и объяснения.

Таким образом, в случае многократной реализации исхода принятого решения проблема выбора мало чем отличается от ситуаций, в которых случайные факторы отсутствуют. Дополнительные сложности здесь носят чисто вычислительный характер и связаны с необходимостью выполнения операций усреднения.

Рассмотрим еще один часто упоминаемый критерий — *критерий (принцип) недостаточного основания Бернулли.* Этот критерий, по существу, применяется в условиях полной неопределенности, когда информация о вероятностях состояния среды z_i отсутствует.

Сам Яков Бернулли (1654—1705) формулировал его следующим образом: если нет данных к тому, чтобы считать одно событие из полной системы несовместимых событий более вероятным, чем другие, то все события нужно считать равновероятными.

В контексте нашей задачи при конечности рассматриваемых множеств этот принцип приводит к оценочной функции

$$y_i = \frac{1}{m}\sum_{j=1}^{m} y_{ij} \to \max_i.$$

При этом, очевидно, множитель $1/m$ может быть опущен без изменения вводимого упорядочения альтернатив, и мы приходим просто к операции суммирования "доходов" по строкам матрицы решений.

При практическом использовании рассмотренных статистических критериев могут возникать, однако, значительные трудности, например, связанные с построением набора $\{z_i\}$ состояний среды (в дискретном случае). Обычно вполне справедливо указывается, что состояния z_i должны быть несовместны, а сам набор $\{z_i\}$ обладать свойством полноты — в обычном статистическом смысле. К сожалению, соблюдение этих важных требований часто тоже не спасает ситуации.

Пример 3.5. В определенных условиях следующие два набора состояний среды удовлетворяют вышеприведенным требованиям:

- набор 1:
  - z_1 — цель неподвижна;
  - z_2 — цель перемещается;
- набор 2:
  - z_1 — цель неподвижна;
  - z_2 — цель перемещается влево;
  - z_3 — цель перемещается вправо.

Однако если соответствующие две задачи решать, например, по критерию недостаточного основания Бернулли, то получим различные результаты. Другими словами, проблема формирования наборов $\{z_i\}$ для сложных ситуаций принятия решения — отдельная и далеко не тривиальная задача.

Ситуация становится еще более сложной, если исход принятого решения реализуется однократно ("одноразовое использование решения"). Такой случай характерен, в частности, при решении задач оптимального выбора параметров уникальных изделий, например, мостов, ирригационных

сооружений, финансовых проектов, программных продуктов и т. п. При этом информация о статистических характеристиках факторов неопределенности, даже если она и имеется, не имеет никакого смысла. Какова бы ни была вероятность того, что значение некоторого числового параметра неопределенности z будет равно 10^{10} или 10^{-10}, мы ничего не сможем сказать о значении функционала $J(x, z)$, которое реализуется в действительности при конкретном выборе x. Здесь мы в силу уникальности ситуации уже не можем "рассчитывать на средний случай". Подобные задачи принятия решений необходимо решать особыми нестатистическими методами, либо (опять же, допуская определенный риск) переходить к другой философии и, по существу, заменять вероятности некоторыми "коэффициентами уверенности" и т. п.

3.3. Критерии принятия решений в условиях полной неопределенности

Как уже указывалось, применение методов теории вероятностей при однократной реализации исхода принятого решения, вообще говоря, неправомерно. Методологически близкая ситуация возникает и в случаях многократной реализации исхода, но при дополнительном предположении, что либо распределение вероятностей параметра z неизвестно, либо параметр неопределенности z изменяется неизвестным образом, но не является случайным (не обладает свойством статистической устойчивости).

В указанных ситуациях информация о факторе неопределенности z обычно имеет вид

$$z \in Z, \tag{3.4}$$

где Z — некоторое множество. Но подобной информации также недостаточно для однозначного решения задачи выбора альтернативы x. Напомню, что мы по-прежнему рассматриваем задачу оптимизации $J(x, z) \to \max\limits_{x}$. Из решения этой задачи мы можем определить вектор x как функцию z:

$$x = x\,(z). \tag{3.5}$$

Формула (3.5) позволяет лишь отобразить множество неопределенности природных факторов Z на множество $G_x \subset X$, которое естественно назвать *множеством неопределенности решений* x. Выбор конкретного элемента из множества G_x может основываться на введении различных разумных гипотез о поведении среды. Одна из важнейших гипотез такого

типа называется *гипотезой антагонизма*. Она состоит в предположении, что среда ведет себя "наихудшим" (для лица, принимающего решение) образом. В итоге в качестве оптимальной альтернативы выбирается решение следующей задачи оптимизации:

$$J_4(x) = \min_{z \in Z} J(x, z) \to \max_{x \in X}. \tag{3.6}$$

Из последнего соотношения видно, что для вычисления значения функционала $J_4(x)$ при фиксированном значении x решается задача минимизации $J(x, z)$ по z, т. е. подбирается "наихудший" возможный вариант z. Соответствующее значение J и берется в качестве значения функционала J_4, соответствующего заданному значению x. Принцип выбора оптимальной альтернативы x^* на основе решения задачи (3.6) называется также *принципом гарантированного результата* или *принципом максимина* (используется также название *критерий Вальда*). Число $J_4(x^*) = J^*$ называется *гарантированной оценкой*, а сам элемент x^* — *гарантирующим решением*.

Замечание 3.1

Смысл введенных названий в том, что, каково бы ни было значение параметра неопределенности z, выбор $x = x^*$ согласно формуле (3.6) гарантирует, что при любом z значение целевого функционала $J(x, z)$ будет не меньше, чем J^* (докажите это).

Оценочная функция данного критерия для дискретного случая имеет вид:

$$y_i = \min_j y_{ij}.$$

Очевидно, что если значение функционала $J(x, z)$ отражает не "полезность" альтернативы x, не "доход", а, напротив, — "потери", то исходная задача состоит в минимизации функции $J(x, z)$, а максиминный критерий превращается в *минимаксный*:

$$J_5(x) = \max_{z \in Z} J(x, z) \to \min_{x \in X}. \tag{3.7}$$

Максиминные и минимаксные критерии являются крайне осторожными, "пессимистичными", что может иногда приводить к нелогичным выводам, противоречащим здравому смыслу.

Пример 3.6. Пусть функция $J(x, z)$ задана с помощью табл. 3.5, где элементы матрицы решений имеют смысл "потерь" (заданных в некоторых условных единицах, у. е.), которые следует минимизировать.

Таблица 3.5. *Матрица потерь*

X	Z	
	z_1	z_2
x_1	10 100 у. е.	100 у. е.
x_2	10 000 у. е.	10 000 у. е.

При выборе решения x_1 или x_2 мы по-прежнему не знаем, какое значение z_1 или z_2 примет фактор неопределенности z. Применение минимаксного критерия приводит к выбору x_2. Но интуитивно мы склонны выбрать x_1, поскольку совсем не исключено, что реализуется "состояние природы" z_2 и наш проигрыш будет существенно уменьшен (равен 100 у. е.). В то же время при выборе x_2 мы гарантированно получим потери в 10 000 у. е. при любом значении z.

Предположим теперь, что задана табл. 3.6, представляющая функционал $J(x, z)$.

Таблица 3.6. *Исходная матрица решений*

X	Z				
	z_1	...	z_j	...	z_s
x_1	y_{11}	...	y_{1j}	...	y_{1s}
...	...	...	...	...	...
x_i	y_{i1}	...	y_{ij}	...	y_{is}
...	...	...	...	...	...
x_n	y_{n1}	...	y_{nj}	...	y_{ns}

Здесь введены обозначения $y_{ij} = J(x_i, z_j)$. Здесь предполагается конечность множеств X, Z. Поясним на этом примере, как можно исправить положение с излишней "осторожностью" максиминного (или минимаксного) критерия. Введем новую матрицу вместо $\{y_{ij}\}$ следующим образом:

$$r_{ij} = \max_{k=1,\ldots,n} y_{kj} - y_{ij},$$

если y — "доход";

$$r_{ij} = y_{ij} - \min_{k=1,\dots,n} y_{ki},$$

если y — "потери".

Таким образом, r_{ij} есть разность между наилучшим значением в столбце j и значением y_{ij} при том же j. Следовательно, обработка матрицы $\{y_{ij}\}$ идет "по столбцам".

Построенная таким способом матрица $\{r_{ij}\}$ называется *матрицей сожалений*, т. к. по существу каждое число r_{ij} выражает "сожаление" лица, принимающего решение, по поводу того, что он не выбрал наилучшего решения относительно состояния z_j.

Критерий минимального сожаления, предложенный Сэвиджем, состоит в применении минимаксного критерия (независимо от того, какой характер имели элементы y_{ij} — "доходы" или "потери") к матрице сожалений $\{r_{ij}\}$:

$$J_6(x) = \max_{j=1,\dots,S} r_{ij} \to \min_{i=1,\dots,n} \Rightarrow i_*, \quad x^* = x_{i_*},$$

т. е. числа r_{ij} всегда носят характер "потерь" и их необходимо минимизировать. Соответствующая оценочная функция (при условии, что исходная матрица является матрицей доходов) имеет вид:

$$y_i = \max_j (\max_i y_{ij} - y_{ij}) \to \min_i.$$

Обратимся снова к последнему примеру. Матрица сожалений будет иметь вид табл. 3.7.

Таблица 3.7. *Матрица сожалений*

X	Z	
	z_1	z_2
x_1	100	0
x_2	0	9900

В этом случае имеем:

$$i = 1: \max_{j=1,2} y_{ij} = \max\{100, 0\} = 100;$$

$$i = 2: \max_{j=1,2} y_{ij} = \max\{0, 9900\} = 9900.$$

В результате, согласно критерию Сэвиджа, выбираем первую альтернативу x_1, к чему мы и стремились интуитивно.

Следующий критерий оптимальности принимаемого решения называется *критерием Гурвица*. Этот критерий охватывает ряд различных подходов к принятию решений — от наиболее оптимистичного до наиболее пессимистичного.

Наиболее оптимистичный подход (в предположении, что y_{ij} означает "выигрыш" или "доход") состоит в выборе x^* из условия

$$\max_i \max_j y_{ij} \to i_*; \quad x^* = x_{i_*}. \tag{3.8}$$

Аналогично при наиболее пессимистичных предположениях выбираемое решение соответствует

$$\max_i \min_j y_{ij}. \tag{3.9}$$

Критерий Гурвица, называемый также *критерием пессимизма–оптимизма*, сводится к взвешенной комбинации обоих способов, устанавливая баланс между случаями предельного оптимизма и крайнего пессимизма. Если y_{ij} означает "прибыль" (т. е. соответствующие величины необходимо максимизировать), то выбирается решение из условия

$$\max_i \{\alpha \max_j y_{ij} + (1-\alpha) \min_j y_{ij}\}, \; 0 \le \alpha \le 1.$$

Оценочная функция для случая "доходов" имеет, следовательно, вид:

$$y_i = \alpha \max_j y_{ij} + (1-\alpha) \min_j y_{ij} \to \max_i.$$

В том случае, когда y_{ij} представляет "затраты", оптимальное решение удовлетворяет аналогичному соотношению:

$$\min_i \{\alpha \min_j y_{ij} + (1-\alpha) \max_j y_{ij}\}.$$

При $\alpha = 1$ имеем случай предельного оптимизма (3.8); при $\alpha = 0$ — случай крайнего пессимизма (3.9). Промежуточные значения *показателя пессимизма—оптимизма* α характеризуют ту или иную склонность лица, принимающего решение, к пессимизму или оптимизму. При отсутствии явно выраженной склонности целесообразно полагать $\alpha = 1/2$.

В непрерывном случае, когда аргументы функционала $J(x, z)$ не обязаны принадлежать конечным множествам, имеем:

$$\max_{x \in X} \{\alpha \max_{z \in Z} J(x,z) + (1-\alpha) \min_{z \in Z} J(x,z)\} \Rightarrow x^*$$

или

$$J_7(x) = \alpha \max_{z \in Z} J(x,z) + (1-\alpha) \min_{z \in Z} J(x,z) \to \max_{x \in X}.$$

Аналогично для критерия Сэвиджа:

$$J_8(x) = \max_{z \in Z} r(x,z) \to \min_{x \in X},$$

где (в предположении, что функционал J требуется максимизировать)

$$r(x,z) = \max_{x \in X} J(x,z) - J(x,z).$$

Пример 3.7 [36]. Одно из предприятий, занимающихся обслуживанием населения, должно определить уровень предложения услуг так, чтобы удовлетворить потребности клиентов в течение предстоящих праздников. Точное число клиентов неизвестно, но ожидается, что оно может быть равным одному из четырех значений: $z_1 = 200$, $z_2 = 250$, $z_3 = 300$, $z_4 = 350$. Для каждого из этих возможных значений z_i существует наилучший уровень предложения (с точки зрения возможных затрат). Отклонения от этих уровней приводят к дополнительным затратам либо из-за превышения предложения над спросом, либо из-за неполного удовлетворения спроса.

Затраты J (в у. е.) приведены в табл. 3.8, где x_i означают варианты уровней предложения, среди которых надлежит найти оптимальный.

***Таблица 3.8.** Матрица затрат*

X	Z			
	z_1	z_2	z_3	z_4
x_1	5	10	18	25
x_2	8	7	8	23
x_3	21	18	12	21
x_4	30	22	19	15

Заметим, что все отраженные в табл. 3.8 уровни предложения оказываются наилучшими для соответствующих значений z_i. Так, x_1 оказывается наилучшим при $z = z_1$, x_2 — при $z = z_2$, x_3 — при $z = z_3$ и x_4 — при $z = z_4$. Таким образом, "лишних" x_i табл. 3.8 не содержит. Применение минимаксного критерия к выбору решения позволяет получить гарантированное значение $J^* = 21$ и $x^* = x_3$.

Критерий Сэвиджа приводит к матрице сожалений:

$$\{r_{ij}\} = \begin{bmatrix} 0 & 3 & 10 & 10 \\ 3 & 0 & 0 & 8 \\ 16 & 11 & 4 & 6 \\ 25 & 15 & 11 & 0 \end{bmatrix}.$$

В результате минимаксной обработки матрицы $\{r_{ij}\}$ получаем $x^* = x_2$, что соответствует "сожалению", равному 8.

Критерий Гурвица при $\alpha = 1/2$ приводит к выбору решения $x^* = x_1$ или $x^* = x_2$. Необходимые промежуточные результаты представлены в табл. 3.9.

Таблица 3.9. *Критерий Гурвица*

X	$\min\limits_j y_{ij}$	$\max\limits_j y_{ij}$	$\alpha \min\limits_j y_{ij} + (1-\alpha) \max\limits_j y_{ij}$
x_1	5	25	15
x_2	7	23	15
x_3	12	21	16, 5
x_4	15	30	22, 5

Упражнения

1. Примените минимаксный критерий в приведенном примере, если четвертое значение возможного числа клиентов z_4 исключено.

   Ответ: минимаксное значение равно 8 и соответствует решению x_2.

2. Примените критерий Сэвиджа, предполагая, что решение x_2 исключено.

   Ответ: минимаксное значение $r_{ij} = 10$ и соответствует выбору x_1.

3. Решите этот пример с помощью критерия Гурвица при $\alpha = 0, 75$.

   Ответ: следует выбрать x_1 со значением целевой функции 10.

3.4. Некоторые трудности

Рассмотренные в предыдущем разделе критерии обладают целым рядом недостатков и логических противоречий. Это довольно тонкие вопросы, и здесь невозможно достаточно полно изложить эти проблемы. Рассмотрим только несколько примеров и наводящих соображений, позволяющих уяснить характер возможных затруднений.

Пример 3.8. Рассмотрим случай, когда ЛПР (лицо, принимающее решение) не может остановиться ни на одном из предложенных критериев:

1) максиминном, 2) Гурвица ($\alpha = 3/4$) и 3) недостаточного основания Бернулли при выборе альтернативы согласно заданной матрице "доходов" (табл. 3.10).

Таблица 3.10. *Матрица доходов*

X	Z		
	z_1	z_2	z_3
x_1	2	12	–3
x_2	5	5	–1
x_3	0	10	–2

Поэтому ЛПР решает считать альтернативу x_i предпочтительнее, чем x_j, в том и только в том случае, если на это указывает большинство из трех рассмотренных критериев.

Легко проверить, что порядок полученных предпочтений имеет вид

1. $x_2 \succ x_3 \succ x_1$.

2. $x_3 \succ x_1 \succ x_2$.

3. $x_1 \succ x_2 \succ x_3$.

Таким образом, большинство критериев указывает на то, что $x_1 \succ x_2$ (критерии 2 и 3), $x_2 \succ x_3$, $x_3 \succ x_1$: $x_1 \succ x_2 \succ x_3 \succ x_1$.

Получили так называемый "порочный круг" (нарушение транзитивности) — уже знакомый нам парадокс группового выбора на основе принципа большинства.

Ранее мы уже вводили упорядочение вида:

$$x_1 \overset{P}{\succ} x_2 \leftrightarrow \forall z \in Z : F(x_1, z) \geq F(x_2, z),$$

где хотя бы для одного z неравенство строгое.

Это свойство строгого доминирования одной строки матрицы решений (для дискретного случая) над другой. Интуитивно представляется естественным, чтобы "хороший" критерий удовлетворял следующему требованию.

Аксиома 3.1

Если $x' \overset{P}{\succ} x''$, то x'' не может быть оптимальным.

В то же время применение максиминного (минимаксного) критерия и критерия Гурвица к матрице доходов (табл. 3.11) приводит к оптимальности как x_1, так и x_2. А это противоречит казалось бы очевидной аксиоме 3.1, т. к. в данном случае $x_1 \overset{P}{\succ} x_2$ и x_2 не должно быть оптимальным.

Таблица 3.11. *Матрица доходов*

X	Z		
	z_1	z_2	z_3
x_1	0	1	3/4
x_2	0	1	1/2

Рассмотрим еще одно "очевидное" требование к "хорошему" критерию.

Аксиома 3.2

Добавление к матрице решений новой строки, которая доминируется одной из уже имеющихся строк, не влияет на оптимальность прежних решений.

Однако следующая жизненная ситуация показывает и правомерность подхода, связанного с нарушением аксиомы 3.2.

Пример 3.9 (Кини, Райфа). Человек, блуждая в чужом городе в обеденное время, случайно набрел на скромный ресторан и нерешительно входит в него. Официант сообщает ему, что меню нет и что посетитель может получить либо отварную лососину за $2, 5, либо бифштекс за $4. В первоклассном ресторане он выбрал бы бифштекс, но принимая во внимание

то, что ему неизвестна обстановка, и учитывая разницу цен, он выбирает лососину. Вскоре официант возвращается из кухни и многословно извиняется, упрекая неразговорчивого шеф-повара, не сказавшего, что в меню есть также жареные улитки и лягушачьи лапки, то и другое по $4, 5. Оказывается, что наш герой питает отвращение к тому и другому и предпочел бы им лососину, тем не менее, он отвечает: "Прекрасно, я переменю свой заказ на бифштекс."

Очевидно, это является нарушением вроде бы обоснованной аксиомы 3.2, ибо предполагается, что каждая из вновь добавленных альтернатив (улитки и лягушачьи лапки) в любом случае проигрывает уже имеющейся альтернативе — лососине, т. е. является доминируемой. Но можем ли мы сказать, что посетитель действует неразумно? Он, подобно многим другим, заключил, что лишь в "хороших" ресторанах подаются улитки и лягушачьи лапки, и поэтому риск получить плохой бифштекс, по его мнению, уменьшается.

Данные рассуждения, конечно, поддаются критике, но мы теперь уже не так уверены в бесспорности аксиомы 3.2.

Пример 3.10. Еще одно возможное возражение против весьма распространенного критерия Гурвица состоит в том, что для следующей задачи выбора он дает решение, противоречащее здравому смыслу. Действительно, рассмотрим матрицу доходов (табл. 3.12).

***Таблица 3.12.** Матрица доходов*

X	Z				
	z_1	z_2	z_3	...	z_{100}
x_1	0	1	1		1
x_2	1	0	0	...	0

Согласно критерию Гурвица оба решения x_1 и x_2 равноценны и их оценки для любого α равны $1 - \alpha$. Однако если истинное состояние среды z_i совершенно неизвестно, то интуитивно мы бы предпочли x_1, подразумевая, что реальным состоянием "вероятнее" окажется одно из состояний z_2, ..., z_{100}, а не z_1.

Здесь, по существу, возникают почти философские проблемы, связанные с понятием "полного незнания" и т. д. Ведь интуитивно мы все равно предположили, что вероятность реализации одного из состояний z_2, ..., z_{100} выше, чем у z_1.

Основной вывод заключается в том, что сложность проблемы принятия решений в значительной степени определяется самим процессом формализации задачи, например, в виде соответствующей матрицы решений. Это далеко не формальный акт и он должен выполняться опытным системным аналитиком, специалистом в конкретной предметной области.

3.5. Принятие решений в условиях конфликта (элементы теории игр)

Снова будем считать, что задача принятия решений сформулирована в виде задачи оптимизации

$$J(x,z) \to \max_{x \in X}, z \in Z. \tag{3.10}$$

В отличие от предыдущих случаев, когда параметром z управляла "природа", здесь мы предполагаем, что параметр z управляется "разумным" противником, преследующим собственные цели. Эти цели выражаются с помощью задачи ПР, аналогичной (3.10):

$$I(x,z) \to \max_{z \in Z}, x \in X. \tag{3.11}$$

Подобные конфликтные задачи ПР первоначально были формализованы как задачи анализа салонных игр, что придало всей терминологии несколько легкомысленное звучание. Так, обе противоборствующие стороны называются *игроками*, выбираемые ими альтернативы (соответственно x и z) — *ходами*, правила выбора решений — *стратегиями*, значения функционалов J и I — *выигрышами*, а вся теория ПР с неопределенностью типа "активный партнер" — *теорией игр*. Иногда задачи ПР в условиях "природных" неопределенностей, которые мы рассматривали в предыдущем разделе, называют *играми против природы*.

Итак, пусть два субъекта А и Б, располагающие возможностью выбора, соответственно, элементов $x \in X$ и $z \in Z$, стремятся к достижению своих целей, представленных в виде (3.10), (3.11).

Расхождение между функционалами I и J определяет степень антагонизма игроков. В частном случае может оказаться, что $J = -I$ при любых x и z; такую ситуацию, возникающую в игре двух субъектов, называют *антагонистической*, *строго конкурентной* или *игрой с нулевой суммой* ($J + I = 0$). Однако чисто антагонистическая ситуация является в известном смысле вырожденной. Наиболее типичен конфликт, в котором интересы игроков не совпадают, но и не строго противоположны.

Легко представить себе ситуацию, когда не два, а k игроков максимизируют свои выигрыши $p_i(x^1, x^2, ..., x^k)$, $i = 1, ..., k$. В этом случае, например, для первого игрока, выбирающего решение x^1, остальные x^i будут составлять фактор неопределенности z: $p_1(x_1, z) \to \max\limits_{x^1 \in X}$, $z = (x^2, ..., x^k)$. Если $\sum\limits_{i=1}^{k} p_i = 0$, то мы по-прежнему говорим об игре с нулевой суммой, хотя термин "антагонистическая игра" здесь уже неприменим. Далее мы будем рассматривать только игры двух лиц.

Итак, пусть две стороны А и Б стремятся к достижению своих целей:

$$\text{А}: J(x,z) \to \max_{x \in X}, \quad z \in Z;$$

$$\text{Б}: I(x,z) \to \max_{z \in Z}, \quad x \in X.$$

Оба лица, принимающие решения (ЛПР), или оба "игрока", располагают возможностью выбора x и z соответственно. Далее для определенности будем полагать, что $X \subset R^n$, $Z \subset R^m$, т. е. x и z — числовые векторы соответствующих размерностей.

Пример такой постановки задачи уже приводился — это пример В.4 из *Введения* ("дилемма заключенного"). Проводя рассуждения со стороны первого игрока (игрока А), легко установить, что оба функционала J и I задаются табл. 3.13.

Таблица 3.13. *Игра двух лиц*

X	Z	
	z_1 = **Н**	z_2 = **П**
x_1 = **Н**	(1, 1)	(10, 0)
x_2 = **П**	(0, 10)	(7, 7)

На пересечении строки i и столбца j в табл. 3.13 стоит пара чисел (p, q), где $p = J(x_i, z_j)$, $q = I(x_i, z_j)$. В данном примере, очевидно, требуется минимизировать функционалы J и I, а не максимизировать.

Далее мы везде будем считать себя игроком А и проводить рассуждения с позиций его интересов.

В связи с тем, что исход нашего выбора решения зависит от выбора игрока Б, необходимо сделать какие-то предположения о его возможном

поведении в процессе решения задачи. Правомерность подобных предположений (гипотез) впрямую зависит от характера информированности сторон о поведении другой стороны.

При принятии решений в условиях риска (а подобные задачи, как уже говорилось, могут также относиться к теории игр — игр против природы) мы, по существу, предполагали, что сторона Б ("природа") действует не целенаправленно. Мы предполагали, что каждый выбор $z = z_i$ (при дискретном множестве z) характеризуется своей вероятностью, т. е. мы могли оценить частоту появления тех или иных z_i и в соответствии с этим строили свою стратегию поведения. Это одна из возможных гипотез. При игре с "думающим" противником, который преследует в процессе принятия своих решений вполне определенные цели, разумно прибегать к иным гипотезам, лучше отражающим существо такой задачи. По сути дела особым характером вводимых гипотез данный раздел теории ПР и выделяется в отдельную теорию — теорию игр.

Будем различать следующие основные гипотезы (случаи).

Гипотеза 1. Каждый из субъектов А и Б не имеет информации о выборе, сделанном второй стороной. Дополнительные гипотезы о характере поведения второго игрока отсутствуют. В этом случае можно поступать аналогично решению задачи в условиях полной неопределенности. Это, по существу, в точности тот же случай, и мы можем воспользоваться известным принципом наилучшего гарантированного результата. Для субъекта А гарантированная оценка будет равна

$$J^* = \max_{x \in X} \min_{z \in Z} J(x, z), \tag{3.12}$$

а для субъекта Б

$$I^* = \max_{z \in Z} \min_{x \in X} I(x, z). \tag{3.13}$$

Решая задачи максимизации (3.12), (3.13), мы находим и векторы x^*, z^*, реализующие соответствующие гарантированные оценки.

Пример 3.11. Дадим графическую иллюстрацию применения принципа гарантированного результата. Пусть $J(x, z) = x^2 - z^2$, $I(x, z) = -J(x, z)$ и требуется минимизировать J и I. В результате мы имеем антагонистическую игру:

$$J(x,z) = x^2 - z^2 \to \min_{x \in X}, \quad z \in Z;$$

$$J(x,z) = -I(x,z) = x^2 - z^2 \to \max_{z \in Z}, \quad x \in X.$$

Будем считать, что $X = Z = R$ есть множества всех вещественных чисел (здесь мы использовали то очевидное обстоятельство, что вместо поиска

минимума функции I можно искать максимум функции $-I = J$). Для данного примера гарантированная оценка находится из условия

$$J^* = \min_x \max_z J(x, z).$$

Обозначим

$$\varphi(x) = \max_z J(x, z). \tag{3.14}$$

Таким образом, для вычисления одного значения функции φ при фиксированном x необходимо решить задачу оптимизации (3.14). Получим

$$\varphi(x) = \max_z (x^2 - z^2) = x^2,$$

т. к. любой $z \neq 0$ приводит к уменьшению функции J. Теперь находим

$$\min_x \varphi(x) = \min_x x^2 = 0,$$

что достигается при $x = 0$. Таким образом, мы получим $J^* = 0$ и при этом $x^* = 0$. Это гарантированный результат, ибо при любом z мы будем иметь значение J не хуже (т. е. не больше), чем ноль, т. е. при любом z

$$J(x^*, z) = J(0, z) = -z^2 \leq J^0 = 0.$$

На рис. 3.3, *а* представлены линии постоянного уровня функционала $J(x, z)$ на плоскости (x, z).

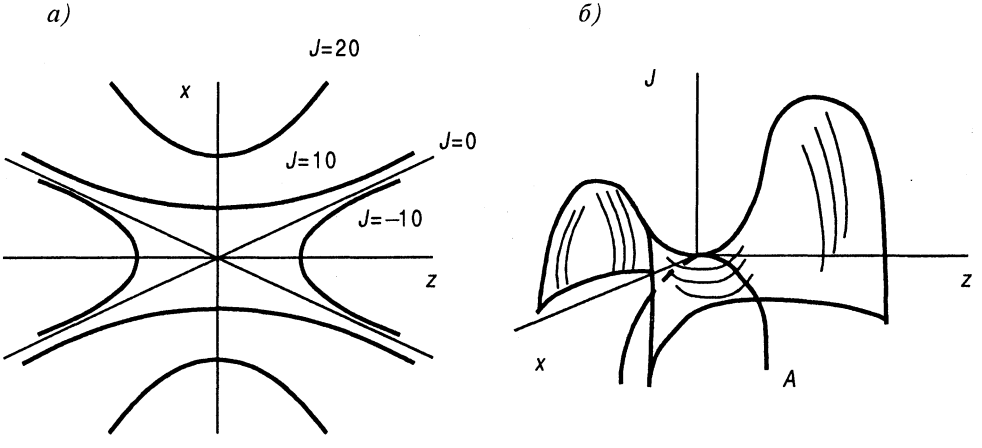


Рис. 3.3. Принцип гарантированного результата

Вспомним, что *линией уровня* называется геометрическое место точек на плоскости, где $J = C = \text{const}$. Меняя постоянную C, мы будем получать различные линии уровня. Если функция зависит более чем от двух пере-

менных, то следует говорить не о линиях уровня, а о *поверхностях уровня*. На рис. 3.3, *б* изображена зависимость $J(x, z)$ в трехмерном пространстве, имеющая характерный вид "седла". Можно считать, что соответствующая поверхность "склеена" из двух видов парабол:

$$y_1 = x^2, y_2 = -z^2.$$

При выборе гарантирующего решения $x^* = 0$ мы при различных z будем всегда находиться на параболе A (рис. 3.3, *б*), обеспечивая выполнение неравенства $J \le J^* = 0$.

Гипотеза 2. Предполагаем, что субъект Б следует принципу максимина и выбирает z^* из условия (3.13):

$$I^* = \max_{z \in Z} \min_{x \in X} I(x, z).$$

Тогда мы можем выбирать x согласно правилу

$$J(x, z^*) \to \max_{x \in X}, \tag{3.15}$$

где z^* — гарантирующее решение второго игрока. Обозначим решение задачи (3.15) через x^{**}. При этом оказывается, что

$$J^* = J(x^{**}, z^*) \ge J^*, \tag{3.16}$$

где J^* — наша гарантированная оценка, получаемая по принципу максимина.

Упражнение. Доказать неравенство (3.16).

На примерах легко убедиться, что неравенство (3.16) может быть строгим, и, следовательно, следуя гипотезе 2, мы в случае ее правомерности можем получить реальный выигрыш, выбирая решение x^{**}, а не x^*.

Пример 3.12. Игра с нулевой суммой задана с помощью табл. 3.14, где числа на пересечении строк и столбцов означают наш выигрыш, т. е. проигрыш игрока Б — нашего соперника.

Таблица 3.14. *Игра с нулевой суммой*

X	Z			
	z_1	z_2	z_3	z_4
x_1	5	–10	9	0
x_2	6	7	8	1
x_3	8	7	15	2
x_4	3	4	–1	4

Наша гарантирующая стратегия $x^* = x_3$, а гарантированная оценка $J^* = 2$. (Если бы мы выбрали другое решение, отличное от x_3, то могли бы в зависимости от действий игрока Б получить и меньшее значение выигрыша, чем 2.) Аналогично для игрока Б (он в отличие от нас стремится минимизировать наш выигрыш, а тем самым и свой проигрыш) имеем $z^* = z_4$, $I^* = 4$. Действительно, игрок Б выбирает тот столбец, в котором максимальное число было бы наименьшим. В первом столбце максимальное число равно 8, во втором — 7, в третьем — 15 и в четвертом — 4. Следовательно выбирая $z^* = z_4$, игрок Б никогда не проиграет больше четырех условных единиц.

Если выбирать x^{**} из условия

$$J(x, z^*) \to \max_{x \in X},$$

то мы получаем $x^{**} = x_4$, $J(x^{**}, z^*) = J^* = 4 > J^* = 2$. Таким образом, следуя гипотезе 2, можно, вообще говоря, получить лучший результат по сравнению с принятием решений на основе принципа гарантированного результата.

Гипотеза 3. Мы теперь можем допустить, что субъект рассуждает точно так же, как и в предыдущем случае, т. е. использует не стратегию z^*, а аналогичную стратегию z^{**}. Поэтому мы можем это учесть и выбирать оптимальное решение с учетом уже этой гипотезы:

$$J(x, z^{**}) \to \max_{x \in X} \Rightarrow x^{***}, J^{***}.$$

Гипотеза 4. Возможен другой сорт гипотез: мы по условиям игры знаем первый ход субъекта Б (он обязан сообщить его нам). Тогда наше поведение будет определяться стратегией в виде функции $x = x(z)$. Мы можем ее определить в результате решения задачи оптимизации

$$J(x, z) \to \max_{x \in X}. \tag{3.17}$$

Условие (3.17) позволяет для каждого фиксированного z определить искомое значение x, т. е. задать функцию $x(z)$.

Для этого случая мы также можем определить гарантированный результат $\hat{J}$

$$\hat{J} = \min_{z \in Z} \max_{x \in X} J(x, z) = \min_{z \in Z} J(x(z), z).$$

Результат $\hat{J}$ будет отличаться от значения J^*, найденного согласно гипотезе 1. Именно, во всех случаях будем иметь

$$\hat{J} \geq J^*. \tag{3.18}$$

Таким образом, принятие гипотезы 4 вновь позволяет улучшить результат, полученный по принципу максиминного гарантированного результата.

Докажем неравенство (3.18), которое имеет вид

$$\min_{z\in Z}\max_{x\in X} J(x,z) \geq \max_{x\in X}\min_{z\in Z} J(x,z).$$

Для любых фиксированных x', z', очевидно, справедливо неравенство

$$\varphi_1(z') \geq \varphi_2(x'), \tag{3.19}$$

где

$$\varphi_1(z') = \max_x J(x,z'); \quad \varphi_2(x') = \min_z J(x',z).$$

Действительно, пусть

$$\max_x J(x,z') = J(x'',z');$$
$$\min_z J(x',z) = J(x',z'').$$

Отсюда имеем следующую цепочку неравенств:

$$\varphi_1(z') = J(x'', z') \geq J(x', z') \geq J(x', z'') = \varphi_2(x').$$

Таким образом, (3.19) доказано. Далее, поскольку x', z' могут быть любыми, их можно выбрать следующим образом:

$$x' = \arg\max_{x\in X} \varphi_2(x); \quad z' = \arg\min_{z\in Z} \varphi_2(z).$$

Подставляя эти значения в (3.19), приходим к (3.18).

В качестве примера рассмотрим игру, представленную в табл. 3.14. Для этой игры, как мы видели,

$$J^* = \max_x \min_z J(x,z) = 2.$$

Для $\hat{J}$ имеем:

$$\hat{J} = \min_z \max_x J(x,z) = 4.$$

Упражнение. Проверьте приведенные числа.

Гипотеза 5. Пусть Б знает наш первый ход. В этом случае естественно предположить, что он будет придерживаться стратегии $z = z(x)$, которая строится в результате решения оптимизационной задачи

$$I(x,z) \to \max_{z\in Z} \Rightarrow z = z(x) \tag{3.20}$$

(именно так мы поступали при принятии гипотезы 4). Принятие этих допущений, т. е. допущения о том, что мы сообщили свой ход субъекту Б, а также допущения об использовании Б стратегии $z(x)$, позволяет нам так воздействовать на выбор субъекта Б, чтобы он в максимальной степени соответствовал нашим целям. Именно, мы можем выбирать x из условия

$$J(x, z(x)) \to \max_{x \in X} \Rightarrow \tilde{x}.$$

Если максимум в соотношении (3.20) достигается не в одной точке z, а на некотором множестве $M(x)$, то наш гарантированный результат $\tilde{J}$ определяется из условия:

$$\tilde{J} = \max_{x \in X} \min_{z \in M(x)} J(x, z).$$

Общим для всех рассмотренных случаев является предположение, что обе стороны, участвующие в игре, не только точно знают свои цели, но и полностью информированы о целевых функциях "противника" или партнера по игре. Для реальных конфликтных ситуаций это не всегда выполняется. Гораздо чаще мы не знаем точно целей наших партнеров, которые, в свою очередь, имеют ограниченную информацию о наших намерениях. Кроме того, необходимо учитывать и возможную сознательную дезинформацию, "блеф" со стороны каждого из игроков. Да и игроков может быть не два, а больше. Формальные модели указанных, а также других игровых ситуаций могут быть построены, но соответствующий материал выходит за рамки этой книги.

До сих пор мы рассматривали проблемы принятия решений в играх двух лиц с позиций одного из игроков. На ту же проблему можно взглянуть со стороны некоторого третьего "нейтрального" лица. Нас здесь будут интересовать некоторые характеристики решения в целом с учетом целевых функций всех игроков. Наиболее важными характеристиками являются, во-первых, *свойства эффективности* принимаемых решений по получаемым игроками "выигрышам", а во-вторых, *свойства устойчивости* решений. С позиций третьего лица — "арбитра" — игра двух лиц с целевыми функциями

$$\begin{aligned} J(x, z) &\to \max_{x \in X}, \\ I(x, z) &\to \max_{z \in Z} \end{aligned} \tag{3.21}$$

может трактоваться как многокритериальная (в данном случае — двухкритериальная) задача оптимизации на множестве $L = X \times Z$. Аргумен-

том при этом является вектор $\eta \in L$, $\eta = (x, z)$, а задача (3.21) принимает обычный вид многокритериальной задачи:

$$\begin{aligned} J(\eta) &\to \max_{\eta \in L}; \\ I(\eta) &\to \max_{\eta \in L}. \end{aligned} \tag{3.22}$$

При анализе эффективности решения задачи (3.22) можно снова воспользоваться уже знакомым принципом Парето — важнейшим из принципов отбора рациональных решений. Этот принцип позволяет отбросить все те решения (альтернативы выбора), которые могут быть заменены другими, обеспечивающими лучшие (в данном случае — бóльшие) значения целевых функций всех игроков одновременно или части игроков, но без уменьшения значений целевых функций остальных субъектов, участвующих в игре. Решения, которые не могут быть указанным образом улучшены, мы и называем *эффективными* или *Парето-оптимальными.* Такие эффективные решения обладают тем свойством, что улучшать значение целевой функции одного из игроков можно только за счет других субъектов. Казалось бы, задача выбора рациональных компромиссных решений и должна решаться только в пределах множества Парето (которое в теории игр называется также "переговорным множеством"). Ведь совершенно ясно, что любое решение, находящееся вне этого множества, может быть улучшено сразу для всех игроков. Однако реальная ситуация часто оказывается значительно сложнее. Основной вопрос заключается в том, что на самом деле выбор η осуществляется не одним лицом, а несколькими. На самом деле здесь мы имеем игру, а не обычную многокритериальную задачу. Важнейшее значение приобретает другой принцип принятия решений, связанный с понятием устойчивости.

Определение 3.1

Будем называть точку $\hat{\eta} = (\hat{x}, \hat{z})$ *устойчивым решением* или *точкой равновесия* игры (3.21), если

$$\max_{x \in X} J(x, \hat{z}) = J(\hat{x}, \hat{z});$$

$$\max_{z \in Z} I(\hat{x}, z) = I(\hat{x}, \hat{z}).$$

При выборе устойчивого решения $\hat{\eta}$ говорят также, что достигнута *ситуация равновесия.*

Из приведенного определения непосредственно следует, что неустойчивость какой-либо ситуации проявляется в том, что в случае ее возникновения ей грозит распад, обусловленный возможностями одного из игро-

ков путем изменения только своей стратегии улучшить свое положение за счет других. На этом основании возник так называемый *принцип устойчивости Нэша* (по имени автора — американского математика Джона Нэша (J. F. Nash)). Он гласит, что выбор рациональной стратегии η должен производиться среди множества точек равновесия. Равновесные решения называются также *оптимальными по Нэшу*. Данный принцип отражает очень важное свойство коллективного решения. Именно, если оба субъекта А и Б смогли договориться о том, чтобы придерживаться выбора $x = \hat{x}$, $z = \hat{z}$, то тот субъект, который нарушает договоренность, прежде всего и пострадает: свойство устойчивости решения дает известную гарантию против нарушения договоренности.

Замечание 3.2

Все сказанное справедливо и для случая N игроков, где $N > 2$.

Рассмотрим теперь связь двух сформулированных принципов выбора решений — принципа Парето и принципа Нэша. Возникает вопрос, насколько хороши устойчивые решения в отношении их эффективности, т. е. в отношении выигрышей, получаемых игроками в разновесных точках. Ведь каждый игрок может рассматривать выбор своего решения как принятие решения в условиях неопределенности (другой игрок выступает в качестве "природной" неопределенности или неопределенности среды). При этом можно воспользоваться принципом наилучшего гарантированного результата и выбрать соответствующую максиминную стратегию, гарантирующую ему независимо от действий другого игрока некоторый минимальный результат. Не получит ли он в таком случае больший выигрыш, чем в ситуации равновесия? Тогда все рассуждения о равновесии вообще не нужны. Справедливо, однако, следующее утверждение: в ситуации равновесия каждый из игроков получает выигрыш, не меньший, чем соответствующий гарантированный максиминный результат.

Докажем данное утверждение. Пусть $(\hat{x}, \hat{z})$ — точка равновесия игры (3.21) и x^* — максиминная стратегия игрока А. Тогда

$$\max_{x \in X} \min_{z \in Z} J(x, z) = \min_{z \in Z} J(x^*, z).$$

Отсюда, используя определение устойчивого решения, имеем

$$\hat{J} = J(\hat{x}, \hat{z}) \geq J(x^*, \hat{z}) \geq \min_{z \in Z} J(x^*, z) = \max_{x \in X} \min_{z \in Z} J(x, z) = J^*.$$

Таким образом, доказано, что $\hat{J} \geq J^*$.

Следовательно, "максиминное возражение" не проходит и анализ равновесия принимаемых решений имеет под собой реальную основу. Вместе с тем оказывается (и это принципиально), что устойчивое решение может не принадлежать к числу эффективных решений, т. е. к множеству Парето. А это уже серьезное замечание (которое ниже будет доказано с помощью соответствующего опровергающего примера). В итоге мы имеем явное противоречие между эффективностью принимаемых решений и их защищенностью от "несанкционированных" действий других игроков. Таким образом, противоречие между оптимальностью по Парето и оптимальностью по Нэшу есть противоречие между выгодностью и устойчивостью.

Только в случаях, когда устойчивые решения являются одновременно паретовскими, можно эффективно использовать принцип Нэша для решения реальных задач. В противном случае мы всегда будем выбирать между эффективностью и надежностью принимаемых решений. В этом, по-видимому, состоит одна из важных первопричин многих конфликтов и неудачных решений в человеческом обществе. Поэтому одним из важных направлений теории ПР и системного анализа является изучение систем, в которых устойчивые точки принадлежат множеству Парето.

Пример 3.13. Пусть к нерегулируемому перекрестку едут на высокой скорости под прямым углом друг к другу два автомобиля. У каждого из водителей есть две стратегии:

- снизить скорость до безопасной (безопасная стратегия — стратегия Б);
- продолжать ехать на высокой скорости (рискованная стратегия — стратегия Р).

Если оба водителя будут придерживаться стратегии Б, то это приведет к благополучному исходу, оцениваемому для каждого водителя числом 1. Если оба водителя следуют стратегии Р, то происходит авария и потери каждого отражаются отрицательным числом (–9). При других комбинациях (Б, Р) или (Р, Б) исход оценивается числом 0 для снизившего скорость (за потерю времени) и числом 3 для двигающегося на высокой скорости (за экономию времени). В итоге имеем игру, представленную в табл. 3.15.

Таблица 3.15. *Устойчивые и неустойчивые ситуации*

1	2	
	Б	**Р**
Б	(1, 1)	(0, 3)
Р	(3, 0)	(–9, –9)

Числа в таблице представляют соответствующие доходы.

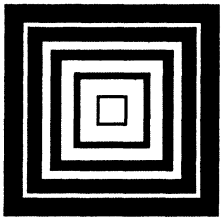
В данном случае ситуации (Б, Б), (Р, Р) являются, очевидно, неустойчивыми, т. к. каждый из водителей может получить лучший результат за счет одностороннего изменения своего решения. Например, водитель 1 может в ситуации (Б, Б) получить лучший для себя результат, изменив стратегию на стратегию Р. В этом случае он получит выигрыш 3 вместо 1. То же справедливо в ситуации (Б, Б) и для игрока 2. Аналогично проверяется и неустойчивость ситуации (Р, Р).

Ситуации (Б, Р) и (Р, Б), напротив, обе являются устойчивыми, т. к. если они возникли, ни у одного из игроков нет оснований для одностороннего изменения стратегии своего поведения.

Упражнения

1. Покажите, что в примере 3.13 оптимальной по Парето не является только ситуация (Р, Р). Таким образом, в данной задаче существуют ситуации — (Б, Р), (Р, Б), которые оптимальны одновременно и по Парето, и по Нэшу.
2. Рассмотрите снова пример В.4 из *Введения* ("дилемма заключенного"). Определите, какие стратегии поведения заключенных являются оптимальными по Парето, а какие — по Нэшу. Установите, что решение (П, П) будет устойчивым по Нэшу, но не оптимальным по Парето.

Глава 4



Многостадийные задачи принятия решений

4.1. Постановка задачи

Понятие многостадийной (многоэтапной, многошаговой) задачи принятия решений весьма многогранно. Поэтому могут рассматриваться совершенно различные модели многостадийности от простых до достаточно сложных. Мы здесь остановимся на обсуждении некоторых традиционных подходов к проблеме, позволяющих уяснить главные черты и особенности многостадийных задач принятия решений в условиях неопределенности. В частности, будем предполагать, что решаемая проблема является одноцелевой. Например, весьма часто цель всей операции заключается в максимизации "доходов" (прибыли, полезности) или минимизации "затрат". Предполагается, что получение "доходов" реализуется на каждом этапе процесса принятия решений, а затем эти "доходы" суммируются (принцип аддитивности).

Рассматриваемая далее модель многостадийной задачи принятия решений предполагает наличие некоторого графа, называемого *деревом решений* и, по существу, описывающего то, как можно попадать из заданного множества его начальных вершин в заданное множество его конечных вершин. При этом с каждой вершиной графа ассоциируется некоторое состояние S_i, в котором находится объект принятия решений, а дуги, выходящие из вершины, соответствуют возможным переходам из одного состояния в другое в зависимости от принимаемых решений.

На рис. 4.1 дан пример так называемого детерминистского дерева решений.

Здесь и далее предполагается, что процесс разворачивается во времени и движение по графу осуществляется слева направо. Допустимые начальные и конечные вершины заштрихованы. Считается, что каждая ветвь графа имеет свой *вес* — вещественное число, означающее соответствующие локальные "затраты" на переход в другое состояние. Основная зада-

ча состоит в оптимальном выборе начальной вершины (из множества допустимых) и пути из нее в любую из допустимых конечных вершин. Оптимальность понимается в смысле построения допустимого пути, реализующего минимальные суммарные затраты (задача выбора минимального пути на графе). В частном случае множества допустимых начальных и конечных вершин могут быть одноэлементными.

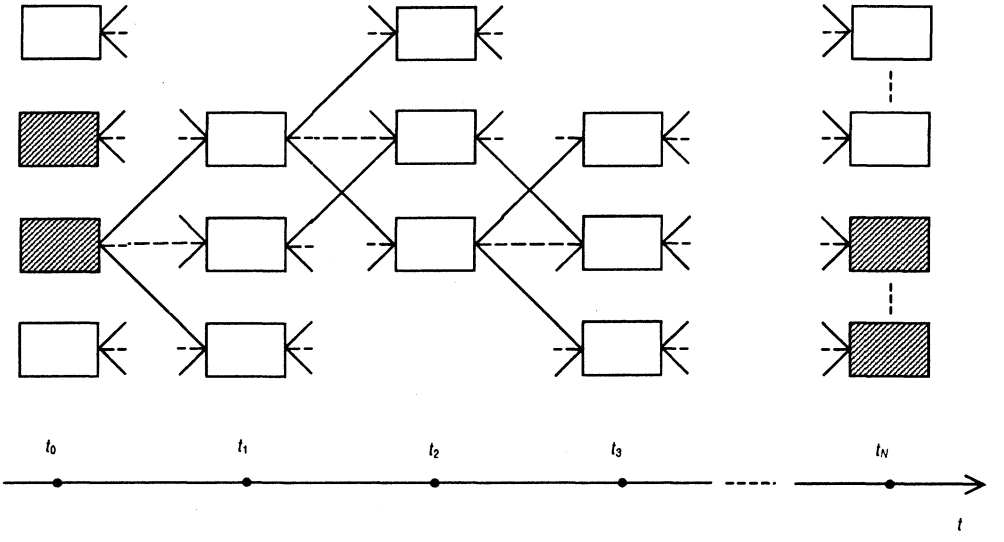


Рис. 4.1. Дерево решений в условиях определенности

В приведенном примере граф содержит только так называемые *основные*, или *"решающие"* вершины (рис. 4.2).

В каждую такую вершину можно попасть различными способами, что показывается наличием нескольких дуг, входящих в вершину. При этом считается, что система (объект принятия решений) находится в определенном фазовом состоянии S_i, а число состояний конечно. Из вершины S_i исходит несколько дуг графа, соответствующих различным решениям, которые могут быть приняты в данном состоянии. Выбор конкретной альтернативы d_i приводит к переходу системы в новую "решающую" вершину (новое состояние).

Более сложная ситуация возникает, когда выбор конкретного решения d_i определяет не новое состояние системы, а задает некоторую лотерею на множестве возможных новых состояний (плотность распределения вероятности) (рис. 4.3).

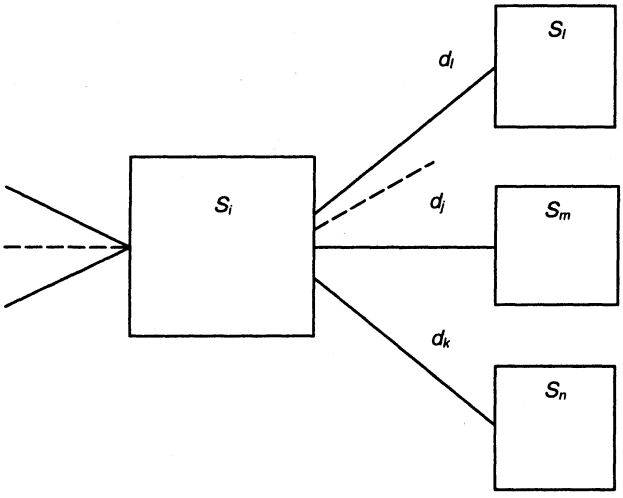


Рис. 4.2. "Решающие" вершины

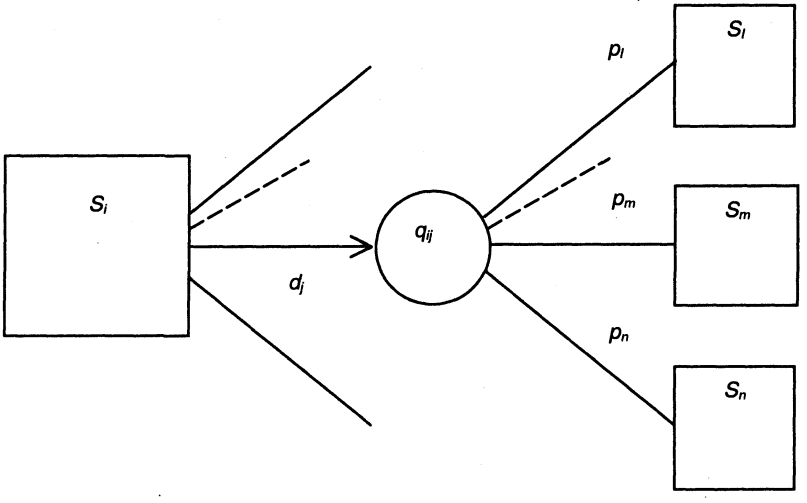


Рис. 4.3. Вероятностная связь вершин

Фактически в конечномерном случае (который и рассматривается) это означает, что после выбора d_j мы попадаем в некоторую "случайную" вспомогательную вершину q_{ij} и далее переходим в одно из возможных для данного этапа состояний S_l, ..., S_m, S_n в соответствии с заданными вероятностями p_l, ..., p_m, p_n (рис. 4.3), где

$$\sum_{i=l,\ldots,m,n} p_i = 1..$$

Это случай так называемой вероятностной неопределенности. В случае полной неопределенности структура рис. 4.3 сохраняется, но стрелки, исходящие из вершины q_{ij}, уже не будут иметь весов (соответствующие вероятности отсутствуют).

В пределах одного и того же графа (дерева решений), описывающего конкретную ситуацию, могут реализоваться все возможные виды переходов.

Перейдем теперь к методологии решения сформулированных многоэтапных задач принятия решений.

4.2. Детерминистский случай. Метод Беллмана

Основная схема и главная идея *метода динамического программирования* (метода Беллмана) применительно к рассматриваемой проблематике достаточно проста и естественна. Рассмотрим конкретный пример детерминистского дерева решений (рис. 4.4).

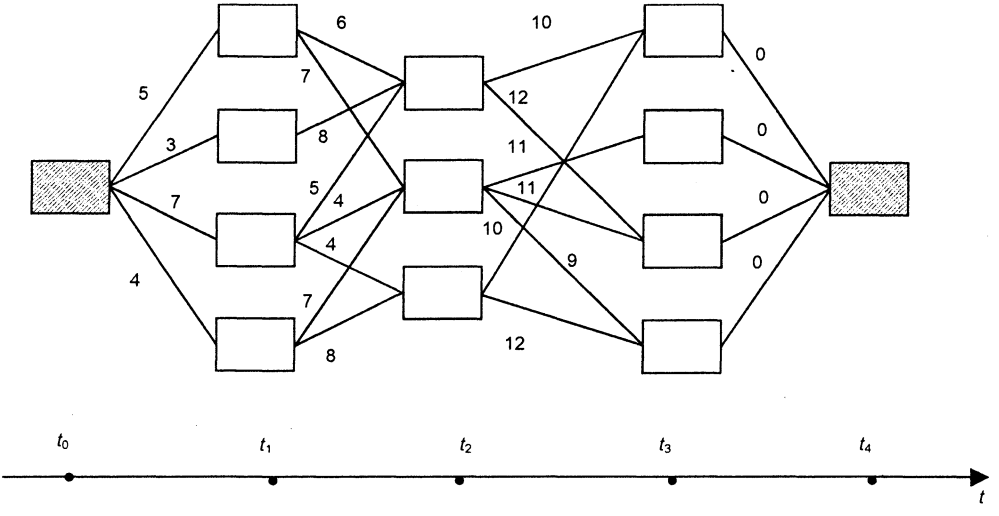


Рис. 4.4. Дерево решений с заданными локальными затратами

На рис. 4.4 одна начальная и одна конечная вершина. Легко видеть, что, по существу, конечными являются все четыре вершины, соответствующие моменту времени t_3. Однако с помощью введения фиктивной вершины для t_4 удалось свести задачу к графу с одной конечной вершиной. Точно так же можно поступать и с начальными вершинами, если их не-

сколько. Числа у дуг графа означают трудоемкости (затраты), обеспечивающие переход от одной "решающей" вершины к другой. Дуги на промежутке $[t_3,t_4]$ имеют нулевые веса, что и означает, что все вершины, отвечающие t_3, являются, по существу, конечными. Главная задача заключается в выборе оптимального в смысле суммарных затрат пути, соединяющего начальную и конечную вершины.

Основная вычислительная идея метода Беллмана состоит из двух моментов:

1. Задача поиска оптимального пути начинает решаться с конца.
2. Исходная задача погружается в множество аналогичных задач с различными начальными вершинами и одной и той же конечной вершиной. При этом предполагается, что в качестве начальной вершины последовательно выступают все без исключения вершины графа.

Реализуем метод Беллмана для приведенного примера. Будем продвигаться по графу справа налево, выставляя определенные числа внутри каждой из вершин (помечая вершины) и указывая оптимальные направления движения из каждой вершины с помощью одной или нескольких стрелок. При этом числа внутри квадратиков-вершин будут означать всегда одно и то же — это суммарные затраты, которые получаются при движении из данной вершины, выбранной в качестве начальной, по оптимальному пути (т. е. это наименьшие из возможных затрат). Например, если мы имеем ситуацию, изображенную на рис. 4.5, *а*, где вершины 2, 3, 4 уже помечены, и надо пометить вершину 1, то будем рассуждать следующим образом (все вершины на рис. 4.5 для удобства объяснения пронумерованы в правом верхнем углу).

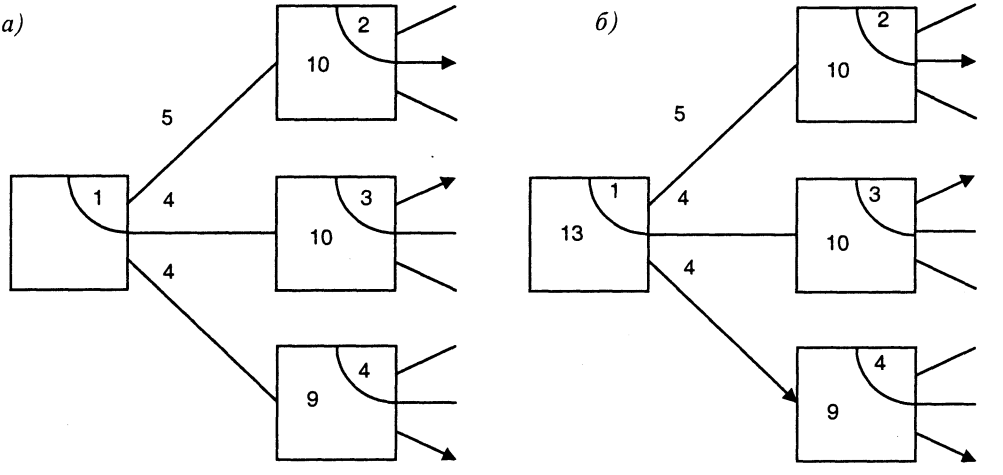


Рис. 4.5. Метод динамического программирования Беллмана

Если в качестве начальной взять вершину 1, то поиск оптимального пути из нее сводится к сравнению трех чисел: 5 + 10 = 15, 4 + 10 = 14, 4 + 9 = 13. Наименьшее из этих чисел — 13, и, следовательно, именно число 13 мы запишем в вершине 1, а стрелочкой соединим вершины 1 и 4 (рис. 4.5, *б*). Действительно, по построению число 9 (полученное на предыдущем этапе) означает минимальные возможные потери при движении из вершины 4 как из начальной в фиксированную конечную вершину. Затраты в 4 единицы необходимы для перехода из вершины 1 в вершину 4. В итоге мы и получаем число 13. В двух других возможных случаях мы имеем большие затраты и, следовательно, оптимальный маршрут из вершины 1 лежит через вершину 4.

Продвигаясь справа налево, мы, обрабатывая последовательно вертикальные слои вершин, разметим весь граф (рис. 4.6).

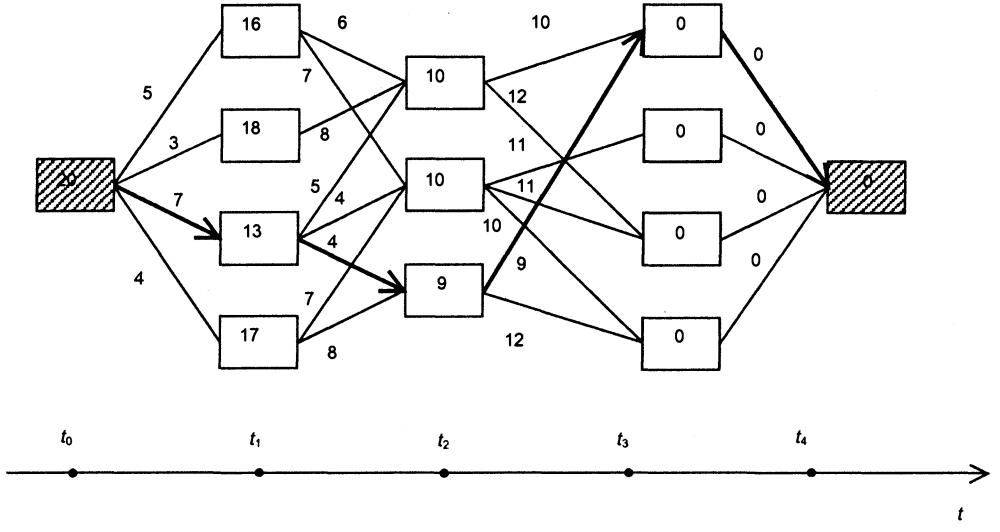


Рис. 4.6. Размеченный граф

Для восстановления искомого оптимального пути достаточно пройти теперь уже слева направо в направлении стрелок, начиная из уже помеченной начальной вершины (оптимальный путь на рис. 4.6 выделен). Число 20 означает минимальные возможные затраты и само оно получается уже на первом этапе разметки графа. Оптимальный путь, очевидно, может быть и не единственным.

Согласно методу Беллмана одновременно находятся все возможные оптимальные пути.

Можно видеть, что проведенная процедура позволяет находить абсолютный глобальный минимум. Предыдущие рассуждения легко преобразуются к строгому доказательству данного утверждения. В то же время важно понимать, что глобально оптимальная стратегия не есть суперпозиция локально оптимальных стратегий. Скажем, при попытке построения оптимального пути на графе рис. 4.6 при движении слева направо и выборе каждый раз стрелок с наименьшим весом мы, конечно, терпим неудачу и получаем результат:

$$3 + 8 + 10 = 21,$$

что больше 20.

Пример 4.1. Можно дать содержательную интерпретацию многоэтапной задачи принятия решений, представленной графом на рис. 4.4. Основная задача может быть сформулирована следующим образом. Некоторая фирма для реализации проекта должна осуществить постройку здания с привлечением субподрядчиков. На этапе $[t_0, t_1]$ необходимо выполнить нулевой цикл и возвести фундамент. Свои услуги для выполнения этого этапа предложили четыре фирмы, и стоимость работ составляет 5, 3, 7, 4 условные единицы соответственно. Возникает вопрос, какую фирму выбрать? На втором этапе $[t_1, t_2]$ на построенном фундаменте необходимо возвести стены и кровлю. Имеются три фирмы, согласные выполнить указанный объем работ. Однако при этом возникают определенные требования к качеству и конструкции фундамента. Каждая фирма может возводить свои стены из своего материала не на любом фундаменте. Поэтому на графе на промежутке $[t_1, t_2]$ каждый квадратик t_1 соединяется не с каждым квадратиком t_2. Даны только допустимые в указанном смысле соединения. Веса у дуг означают, как и прежде, стоимость работ.

Точно так же на этапе $[t_2, t_3]$ мы имеем четыре фирмы, осуществляющие внутренние и отделочные работы и завершающие строительство здания.

Основная задача состоит в экономии затрат по привлечению субподрядчиков. Выше эта задача была решена методом Беллмана.

4.3. Многостадийные задачи принятия решений в условиях неопределенности

Многостадийные задачи принятия решений в условиях неопределенности мы рассмотрим на конкретном примере. В основе по-прежнему лежит метод Беллмана, а также методы раскрытия неопределенностей, обсуждаемые в предыдущих разделах.

Пример 4.2. Торговая фирма должна выполнить оптовые закупки у внешнего производителя с последующей перепродажей товара в течение

года в своих торговых точках. Фирма должна принять решение о закупке крупной партии товара или небольшой партии. Впоследствии, если была закуплена небольшая партия, можно докупить товар у производителя (может быть, по новым оптовым ценам); при первоначальной покупке крупной партии есть опасность убытков из-за возможного невысокого спроса на этот товар на внутреннем рынке. Таким образом, решение, в основном, определяется будущим спросом, который заранее достоверно неизвестен. Кроме того, предполагается, что спрос со временем может измениться. По условиям контракта дополнительные закупки товара фирма сможет выполнить лишь через 4 месяца после начала календарного года при условии, что вначале была закуплена небольшая партия. Вопрос о дополнительных закупках встанет, если установится достаточно высокий спрос на товар. Необходимо обеспечить правильные решения как при первоначальной закупке, так и при возможной дополнительной закупке товара с целью обеспечения максимальной ожидаемой прибыли, получаемой в течение одного года.

Ежемесячная торговая прибыль, получаемая фирмой в каждой из возможных ситуаций, представлена в табл. 4.1.

Таблица 4.1. *Получаемая прибыль при различных уровнях спроса*

	z_1 (высокий спрос)	z_2 (низкий спрос)
x_1 (малая партия)	50	40
x_2 (крупная партия)	200	60

В данном модельном примере мы предположили, что спрос может быть либо "высоким", либо "низким". В принципе возможен более подробный подход с более точными градациями спроса. Кроме того, предполагается, что суммарный торговый ежемесячный доход от продажи докупленной через четыре месяца продукции будет несколько меньше, чем при первоначальной закупке крупной партии и составит 180 у. е. (условных единиц) в месяц при высоком спросе и 40 у. е. — при низком спросе. (Причины этого могут быть различными, в том числе связанными с условиями дополнительной аренды складских помещений, изменением закупочных оптовых цен и т. п.)

Затраты на закупку крупной и мелкой партий товара соответственно составляют 1000 и 200 у. е., а затраты на возможную дополнительную закупку товара (через четыре месяца) равны 840 у. е.

Будем считать, что проведенные маркетинговые исследования показали, что вероятность высокого спроса на данный товар составляет 0, 75, а низкого, соответственно, 0, 25.

Требуется рекомендовать руководству торговой компании такое решение проблемы, чтобы в итоге обеспечить максимальный ожидаемый объем прибыли через год. В данном случае, очевидно, что раз уровень спроса достоверно неизвестен, то речь идет о задаче принятия решений в условиях неопределенности. В частности, можно говорить о максимизации математического ожидания объема прибыли (ожидаемой прибыли). В действительности, в каждом отдельном случае расчет может не совпадать с реально полученными результатами, но "в среднем", при многократном повторении описанной ситуации выбора расчетные значения прибыли дадут хорошую оценку для фактически полученной средней прибыли. Если же описанная выше ситуация с торговой компанией является уникальной (единичной), то, по-видимому, целесообразно при выработке решения использовать не критерий математического ожидания, а, например, критерий Гурвица или Сэвиджа. Для полной гарантии расчеты должны быть основаны на максиминном критерии. Применение принципа гарантированного результата в любом случае полезно, по крайней мере, на первой стадии исследования, с целью предварительной оценки имеющихся потенциальных возможностей и выбора окончательного критерия.

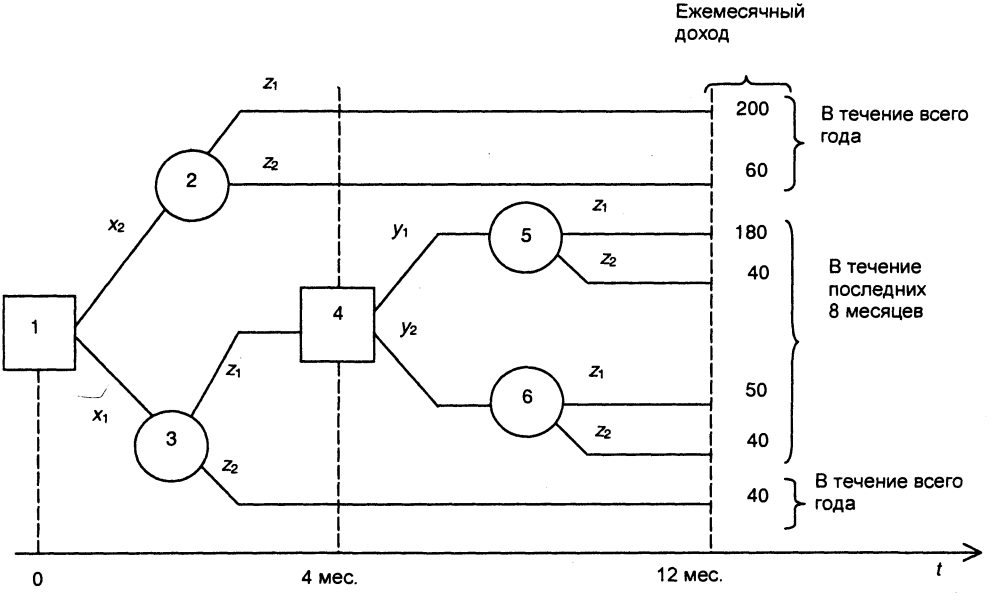


Рис. 4.7. Дерево решений

В силу модельного характера рассматриваемой проблемы мы далее не будем вдаваться в детали соответствующих экономических интерпретаций. Представим сформулированную задачу в виде дерева решений (рис. 4.7).

Здесь квадратиками изображены "решающие" вершины, а кружками — вспомогательные вершины, описывающие неопределенное состояние среды. Внутри указаны номера вершин графа. Обозначения x_1, x_2, z_1, z_2 соответствуют табл. 4.1, а переменные y_i означают следующее:

- y_1 — решение о дополнительных закупках товара через четыре месяца;
- y_2 — решение об отказе от дополнительных закупок.

Дадим решение задачи, воспользовавшись критерием математического ожидания для раскрытия неопределенностей.

Согласно общей рецептуре метода Беллмана решение задачи начинается движением по "решающим" вершинам графа справа налево. Таким образом, вначале обрабатывается "решающая" вершина 4 (рис. 4.8).

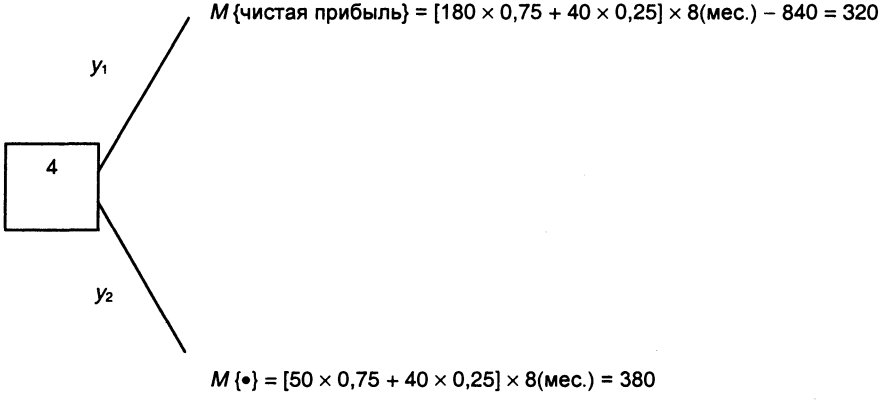


Рис. 4.8. Вычисление ожидаемой прибыли для вершины 4

Из полученных двух чисел 380 оказывается бóльшим (мы максимизируем "доходы"). Этим числом помечается вершина 4, а стрелка совпадает с направлением y_2 (рис. 4.9).

Далее переходим к вершине 1 (рис. 4.10).

Основной вывод состоит в том, что с позиций критерия математического ожидания выгоднее в вершине 1 идти по направлению x_2, т. е. сразу закупать крупную партию товара (при данных числовых характеристиках задачи). Ожидаемая прибыль составит при этом 980 у. е. в год.

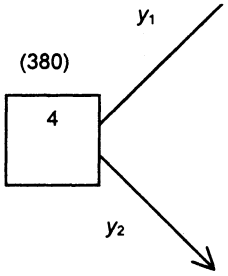


Рис. 4.9. Выбор направления движения из вершины 4

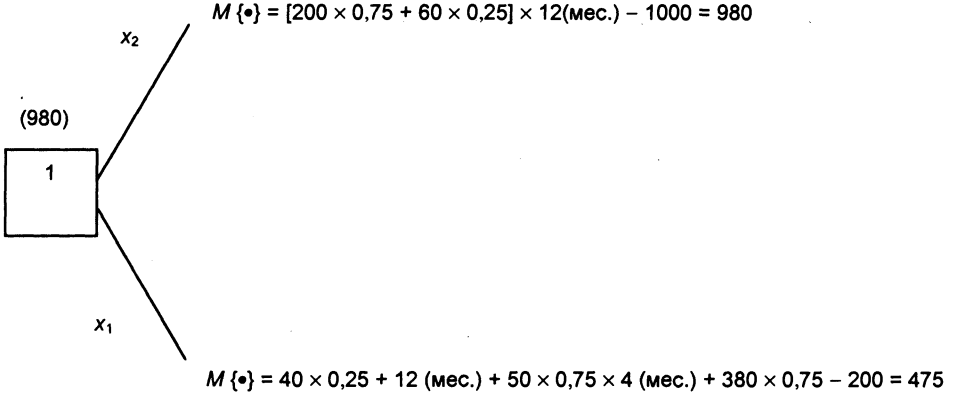


Рис. 4.10. Вычисление ожидаемой прибыли для вершины 1

При обработке вершины 4 мы фактически имели дело с матрицей решений, представленной в табл. 4.2, где:

- 600 = 180 × 8 – 840;
- –520 = 40 × 8 – 840;
- 400 = 50 × 8;
- 320 = 40 × 8.

Таблица 4.2. *Матрица решений для вершины 4*

Y	*Z*	
	z_1 **(0, 75)**	z_2 **(0, 25)**
y_1	600	–520
y_2	400	320

Сама матрица является матрицей доходов (в ней представлена суммарная прибыль за последние восемь месяцев).

В вершине 1 имеем матрицу решений ("доходов"), представленную в табл. 4.3, где:

- $380 = 50 \times 4 + 380 - 200$;
- $280 = 40 \times 12 - 200$;
- $1400 = 200 \times 12 - 1000$;
- $-280 = 60 \times 12 - 1000$.

Таблица 4.3. *Матрица решений для вершины 1*

X	Z	
	z_1 **(0, 75)**	z_2 **(0, 25)**
x_1	380	280
x_2	1400	–280

Решим теперь ту же самую задачу, используя принцип гарантированного результата. В вершине 4 по-прежнему имеем матрицу "доходов", представленную в табл. 4.2. Используя принцип гарантированного результата, заключаем, что оптимальным решением является решение y_2, т. к. наихудший возможный результат при этом равен 320 у. е. "дохода", а при выборе y_1 можем получить потери в объеме 520 у. е. С вершиной 4 теперь ассоциируется число 320, а стрелка пойдет в направлении y_2.

Далее переходим к вершине 1 с матрицей из табл. 4.4 (она уже отличается от матрицы из табл. 4.3). Здесь: $320 = 50 \times 4 + 320 - 200$.

Таблица 4.4. *Матрица решений для вершины 1 в случае гарантированного результата*

X	Z	
	z_1	z_2
x_1	320	280
x_2	1400	–280

По критерию гарантированного результата лучшей оказывается альтернатива x_1 с гарантированным "доходом" в 280 у. е. в год. Следовательно,

первоначально рекомендуется закупить небольшую партию товара. Если сразу установится высокий спрос на товар (и продержится четыре месяца), то мы окажемся в "решающей" вершине 4 и получим прибыль в $50 \times 4 + 320 - 200 = 320$ у. е. за год. Причем через четыре месяца согласно принципу гарантированного результата рекомендуется не делать дополнительных закупок товара (решение y_2 считается оптимальным в вершине 4).

Таким образом, при наличии ситуации неопределенности на различных этапах многошаговой проблемы принятия решений метод Беллмана позволяет указывать оптимальные стратегии поведения в любой "решающей" вершине (т. е. в любом состоянии, в котором может оказаться реальная система). Все эти стратегии представляют для пользователя несомненный интерес, ведь из-за наличия неопределенностей заранее невозможно достоверно установить, какая траектория развития системы реализуется в действительности.

Представленная методика решения многостадийных задач естественным образом обобщается на более сложные многоальтернативные деревья решений.

4.4. Марковские модели принятия решений

В данном разделе рассматриваются многостадийные задачи принятия решений с конечным числом состояний s_j ($j = 1, \ldots, m$) оптимизируемой системы S. Предполагается, что в дискретные моменты времени $t_1, t_2, \ldots$ система переходит в новое состояние в соответствии с некоторой матрицей переходных вероятностей:

$$P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & \cdots & p_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ p_{m1} & p_{m2} & \cdots & p_{mm} \end{bmatrix}.$$

Элемент p_{ij} матрицы означает вероятность перехода системы из состояния s_i в состояние s_j. Таким образом, строки матрицы соответствуют "старым", а столбцы — "новым" состояниям. Очевидно, сумма элементов любой строки матрицы равна 1.

Такой процесс поведения системы называется *марковским*, если вероятность перехода системы в любое возможное состояние в каждый момент

времени определяется только ее состоянием в предыдущий момент времени и не зависит от более ранней предыстории.

Многие практические ситуации могут быть описаны с помощью аппарата марковских моделей. Рассмотрим конкретный пример.

Пример 4.3. Некоторая фирма занимается промышленной разработкой программного обеспечения для компьютерных систем. В начале каждого года она решает задачу замены оборудования, включающего технические и программные средства, используемые в производственном процессе и обеспечивающие необходимую технологическую среду разработки. В зависимости от результатов экспертной оценки оборудования состояние фирмы (это система S) оценивается как "хорошее" (1), "удовлетворительное" (2) и "плохое" (3). Следовательно, система может находиться в одном из трех указанных состояний. Матрица переходных вероятностей может иметь вид:

$$P^1 = \begin{bmatrix} 0,2 & 0,5 & 0,3 \\ 0 & 0,5 & 0,5 \\ 0 & 0 & 1 \end{bmatrix}.$$

Здесь, например, число 0, 3 означает, что если система находилась в "хорошем" состоянии, то в следующий момент изменения состояния (следующий момент анализа состояния фирмы) она окажется в "плохом" состоянии с данной вероятностью. В действительности изменение (ухудшение) состояния связано с процессом износа и устаревания оборудования и технологических сред. Если матрица переходных вероятностей не меняется, то достаточно просто проанализировать весь жизненный цикл системы S.

Предположим, что в зависимости от состояний, в которых последовательно оказывается система, может быть вычислен доход, приносимый фирмой. Логично предположить, что доход за период t_{i+1}—t_i зависит от уровня оснащенности фирмы современным оборудованием и технологическим окружением, включая профессиональный уровень персонала, также требующий непрерывного повышения. В свою очередь, уровень оснащенности в значительной степени коррелирует с тем состоянием, в котором находилась фирма в начале рассматриваемого периода и в его конце. Если, например, в момент времени t_i система находилась в "хорошем" состоянии и в момент t_{i+1} это состояние сохранилось, то, по-видимому, доход будет максимальным (конечно, при выполнении прочих условий, связанных с наличием заказов, ситуации на рынке и т. д.).

Для моделирования этой ситуации можно матрице переходных вероятностей P^1 поставить в соответствие матрицу доходов R^1:

$$R^1 = \begin{bmatrix} 7 & 6 & 3 \\ 0 & 5 & 1 \\ 0 & 0 & -1 \end{bmatrix}.$$

Элемент r_{ij} матрицы означает доход, полученный за период t_{i+1}—t_i при переходе системы из состояния i в состояние j. Так, число 5 означает доход, выраженный в некоторых условных единицах, при сохранении системой "удовлетворительного" состояния. Отрицательные значения отражают потери.

Имея матрицы P^1 и R^1, можно достаточно просто прогнозировать результаты функционирования системы. Ясно, что со временем оборудование устаревает и нуждается в обновлении в соответствии с новыми международными стандартами и требованиями рынка. В результате при постоянных матрицах P^1 и R^1 система может деградировать, неизменно оставаясь в плохом состоянии и принося одни убытки.

В реально работающих фирмах по результатам экспертного анализа проводится периодическое обновление оборудования с изменением технологического окружения и обучением персонала. Данный процесс моделируется изменением матриц переходных вероятностей и доходов. В нашем примере они, например, могут измениться следующим образом:

$$P^2 = \begin{bmatrix} 0{,}3 & 0{,}6 & 0{,}1 \\ 0{,}1 & 0{,}6 & 0{,}3 \\ 0{,}05 & 0{,}4 & 0{,}55 \end{bmatrix}; \qquad R^2 = \begin{bmatrix} 6 & 5 & -1 \\ 7 & 4 & 0 \\ 6 & 3 & -2 \end{bmatrix}.$$

Здесь в матрице доходов мы учли затраты на реорганизацию и модификацию. Например, элемент r_{11} матрицы R^2 оказывается меньше соответствующего элемента матрицы R^1.

На каждом этапе мы можем принять решение не проводить модернизацию фирмы и иметь матрицы P^1 и R^1 или принять решение о необходимых изменениях и получить матрицы P^2, R^2. Возникает проблема выбора или принятия решений с целью максимизации приносимого фирмой ожидаемого (речь идет о вероятностях!) дохода. Это многоэтапная задача принятия решений, т. к. выбор осуществляется каждый раз в заданные дискретные моменты времени.

С привлечением уже рассмотренного примера обсудим основные моменты выбора оптимального решения. Предположим, что планирование

стратегии поведения фирмы осуществляется на конечный период времени. Покажем, что решение может быть основано на уже известном методе динамического программирования (метод Беллмана) в соответствии с общей концепцией анализа и оптимизации многошаговых задач *(см. разд. 4.3)*.

Пусть период t_{i+1}—t_i соответствует одному году, а планирование проводится на трехлетний период. Для наглядности соответствующее дерево решений можно представить графически (рис. 4.11).

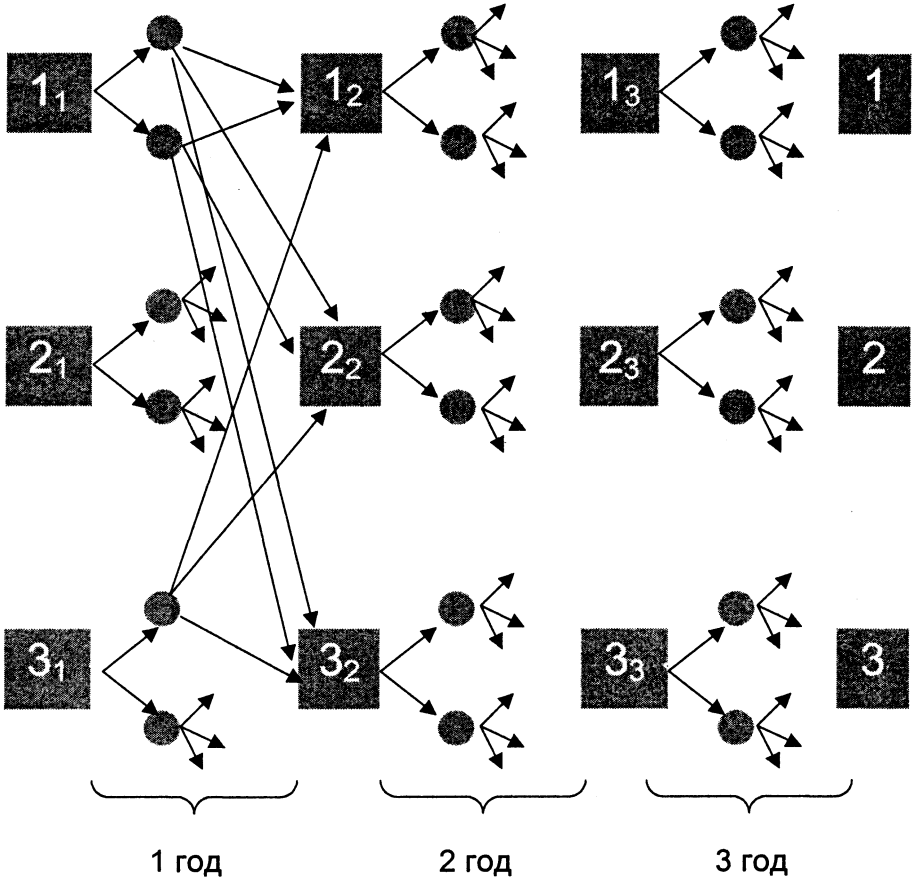


Рис. 4.11. Дерево решений

Как обычно, квадратики означают решающие вершины. Каждый квадратик соответствует определенному состоянию системы в определенный момент времени. Знак i_j внутри квадрата означает, что в момент време-

ни j, $j = 1, 2, 3$ (номер этапа) система находится в состоянии i, $i = 1, 2, 3$ (соответственно, "хорошее", "удовлетворительное" или "плохое" состояние). Две стрелки, исходящие из каждой "решающей" вершины, соответствуют двум альтернативам на каждом этапе: x_1 — проводить модернизацию (это верхняя стрелка, будем называть ее стрелкой или направлением 1) или x_2 — не проводить (это нижняя стрелка, будем называть ее стрелкой или направлением 2). Кружочки означают "случайные" вершины, переход из которых осуществляется в соответствии с выбранной матрицей переходных вероятностей.

Следуя общему алгоритму динамического программирования, решаем задачу с конца. Двигаемся справа налево по решающим вершинам. Начнем с вершины 1_3. Тогда при принятии решения x_1 (без модернизации) ожидаемый доход равен

$$d_1(1_3) = 0{,}2 \times 7 + 0{,}5 \times 6 + 0{,}3 \times 3 = 5{,}3.$$

При выборе x_2 (модернизация) имеем:

$$d_2(1_3) = 0{,}3 \times 6 + 0{,}6 \times 5 + 0{,}1 \times (-1) = 4{,}7.$$

Число 5,3 больше, чем 4,7, поэтому если мы окажемся в вершине 1_3, то пойдем по направлению 1, а сама вершина помечается числом 5,3. Стрелка 1 также выделяется.

Далее переходим к вершине 2_3. Получаем значения двух доходов в зависимости от принимаемых решений:

$$d_1(2_3) = 0{,}5 \times 5 + 0{,}5 \times 1 = 3{,}0;$$

$$d_2(2_3) = 0{,}1 \times 7 + 0{,}6 \times 4 + 0{,}3 \times 0 = 3{,}1.$$

Вершина 2_3 помечается числом 3,1 и выделяется направление 2.

Для вершины 3_3 получим:

$$d_1(3_3) = -1{,}0;$$

$$d_2(3_3) = 0{,}05 \times 6 + 0{,}4 \times 3 - 0{,}55 \times 2 = 0{,}4.$$

Вершина 3_3 помечается бóльшим числом 0,4 и выделяется стрелка 2.

Полученные числа 5,3, 3,1, 0,4 характеризуют один акт изменения состояния и получаемый при этом локальный доход. Далее эти вычисления уже не повторяются, а значения этих локальных доходов потребуются в дальнейших расчетах.

Переходим теперь к началу второго года. Начнем с вершины 1_2. При выборе направления (решения) 1 имеем:

$$d_1(1_2) = 0{,}2 \times 5{,}3 + 0{,}5 \times 3{,}1 + 0{,}3 \times 0{,}4 + 5{,}3 = 8{,}03.$$

Здесь число 5,3 отражает локальный доход этапа (рассчитанный ранее), а остальные слагаемые характеризуют наилучший ожидаемый доход, получаемый на оставшихся этапах. Для второго варианта решения для этой же вершины имеем:

$$d_2(1_2) = 0{,}3 \times 5{,}3 + 0{,}6 \times 3{,}1 + 0{,}1 \times 0{,}4 + 4{,}7 = 8{,}19.$$

Число 8,19 больше, чем 8,03, поэтому вершину 1_2 помечаем числом 8,19 и выделяем стрелку 2.

Для вершин 2_2 и 3_2 проводим аналогичные расчеты:

$$d_1(2_2) = 0 \times 5{,}3 + 0{,}5 \times 3{,}1 + 0{,}5 \times 0{,}4 + 3{,}0 = 4{,}75.$$

$$d_2(2_2) = 0{,}1 \times 5{,}3 + 0{,}6 \times 3{,}1 + 0{,}3 \times 0{,}4 + 3{,}1 = 5{,}61.$$

Выбираем число 5, 61 и выделяем стрелку 2. Далее имеем:

$$d_1(3_2) = 0 \times 5{,}3 + 0 \times 3{,}1 + 1 \times 0{,}4 + (-1) = -0{,}6.$$

$$d_2(3_2) = 0{,}05 \times 5{,}3 + 0{,}4 \times 3{,}1 + 0{,}55 \times 0{,}4 + 0{,}4 = 2{,}13.$$

Для первого этапа аналогично получаем:

$$d_1(1_1) = 10{,}38;$$

$$d_2(1_1) = 10{,}74;$$

$$d_1(2_1) = 6{,}87;$$

$$d_2(2_1) = 7{,}92;$$

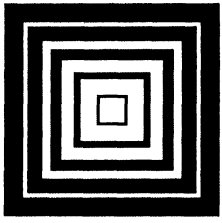
$$d_1(3_1) = 1{,}13;$$

$$d_2(3_1) = 4{,}23.$$

Теперь обратная процедура динамического программирования закончена и, двигаясь от начала дерева решений к концу, можно "прочитать" оптимальное решение. А именно: числа 10,74, 7,92, 4,23 означают оптимальный ожидаемый доход, если, соответственно, система находилась первоначально в состояниях 1, 2 и 3. Эти ожидаемые доходы достигаются, если мы всегда будем вести себя "оптимально", т. е. в соответствии с помеченными на дереве решений стрелками. В частности, в каком бы состоянии мы ни находились в начале первого года, целесообразно решение, связанное с модернизацией оборудования. То же относится к началу второго года (все выделенные стрелки направлены "вниз"). И только если в начале третьего года мы окажемся в состоянии 1, нам нецелесообразно проводить модернизацию оборудования фирмы.

Поставленная задача решена.

Отметим, что в теории марковских процессов принятия решений рассматриваются также модели с бесконечным числом этапов. Эти и другие вопросы рассматриваются в учебной литературе [36].



Глава 5

Методы многокритериального выбора на основе дополнительной информации

Решением многокритериальной задачи, сформулированной в *главе 2*, является соответствующее множество Парето — множество недоминируемых по Парето альтернатив. Это множество может оказаться достаточно обширным, а пользователя соответствующей системы принятия решений обычно интересует выбор какого-то одного "наилучшего" варианта или небольшого их числа. Если какая-либо дополнительная информация о задаче отсутствует, то множество Парето — это лучшее, что можно предложить. Однако при наличии дополнительной информации о системе предпочтений пользователя могут быть применены различные методы сужения исходного множества альтернатив — более сильные, чем методы, основанные на доминировании по Парето. Весьма часто исходной информацией для таких методов выступает само множество Парето и ставится задача его сужения с целью выбора одной или нескольких альтернатив в качестве окончательного результата. Некоторые возможные подходы к решению этой проблемы рассмотрены далее.

5.1. Адаптивные процедуры выбора

Эти методы основаны на гипотезе о существовании некоторой "функции потерь" $u(x)$, определенной на исходном множестве альтернатив X:

$$u : X \to R,$$

где R — множество вещественных чисел.

Основная задача состоит в выборе одного из элементов $x^* \in X$, такого, что

$$x^* = \arg\min_{x \in X} u(x).$$

Можно вместо "функции потерь" ввести аналогичную "функцию полезности" и ставить задачу ее максимизации. В ряде случаев мы так и будем поступать. Это стандартная задача нелинейного программирования о поиске минимизатора или максимизатора. Отличие рассматриваемой ситуации от типовой задачи нелинейного программирования заключается в следующем. Мы здесь предполагаем, что функция $u(x)$ описывает цель операции выбора, и предпочтения лица, принимающего решение (ЛПР), устроены очень просто: чем меньше значение "функции потерь", тем лучше. Главное предположение состоит в том, что функция $u(x)$ считается заранее неизвестной (в противном случае мы просто воспользовались бы методами конечномерной скалярной оптимизации для выбора наилучшей альтернативы).

Далее от ЛПР, решающего многокритериальную задачу выбора, мы будем требовать не оценки значения $u(x)$ для конкретного $x \in X$ (что достаточно сложно), а только способности сравнения двух альтернатив по их векторным оценкам. Как будет показано далее, этого оказывается достаточно, чтобы реализовать, например, такой метод поиска минимума нулевого порядка, как *метод Нелдера—Мида (см. разд. 5.1.1)*. При этом ЛПР выступает в качестве своеобразного измерительного устройства, но ему, как было отмечено, не требуется указывать значения $u(x)$ (что очень важно), а только фиксировать: "хуже", "лучше", "одинаково". Здесь, конечно, есть свои трудности, связанные, например, с возможной противоречивостью ответов ЛПР, но они преодолимы, и мы здесь не будем останавливаться на них.

Рассматриваемые адаптивные процедуры часто оказываются более эффективными и легко реализуемыми, чем так называемые *апостериорные процедуры*, связанные с явным восстановлением функции $u(x)$ (теория полезности). Действительно, построение $u(x)$ в явном виде позволяет, в частности, упорядочить все альтернативы, что не требуется для решения исходной задачи выбора. В результате проделывается "лишняя" работа, при этом ЛПР приходится отвечать на многочисленные дополнительные достаточно сложные и неестественные для него вопросы.

5.1.1. Метод Нелдера—Мида

Метод Нелдера—Мида *(НМ-метод, или метод деформируемого многогранника)* является стандартным методом нелинейного программирова-

ния и изучается в соответствующих курсах по теории конечномерной скалярной оптимизации. Однако здесь для полноты изложения кратко рассматриваются основные элементы этого метода, при необходимости этого может оказаться достаточно для написания соответствующей программы.

НМ-метод решает задачу поиска минимизатора x^* некоторой заданной функции u:

$$X \xrightarrow{u} R,\ X \subset R^n,$$

$$x^* = \arg\min_{x \in X} u(x).$$

Прообразом НМ-метода явился симплексный метод Спендли, Хекста и Химсворта, основная идея которого состоит в следующем.

В пространстве поиска R^n строится равносторонний многогранник (регулярный симплекс) с количеством вершин, равным $n + 1$ (для $n = 2$ это будет равносторонний треугольник). Далее выясняется, какая из вершин симплекса является наихудшей в смысле значения функции $u(x)$. Для этого можно вычислить $u(x)$ во всех вершинах (если функция $u(x)$ задана аналитически или алгоритмически). Но можно производить попарные сравнения вершин, пользуясь категориями "больше", "меньше", "равно", как указывалось ранее.

Найденная наихудшая вершина заменяется на новую вершину, которая является отражением наихудшей вершины относительно центра тяжести оставшихся вершин. Получается новый симплекс, где вся процедура повторяется. В результате симплекс передвигается по пространству поиска в сторону искомого минимизатора функции $u(x)$ (рис. 5.1).

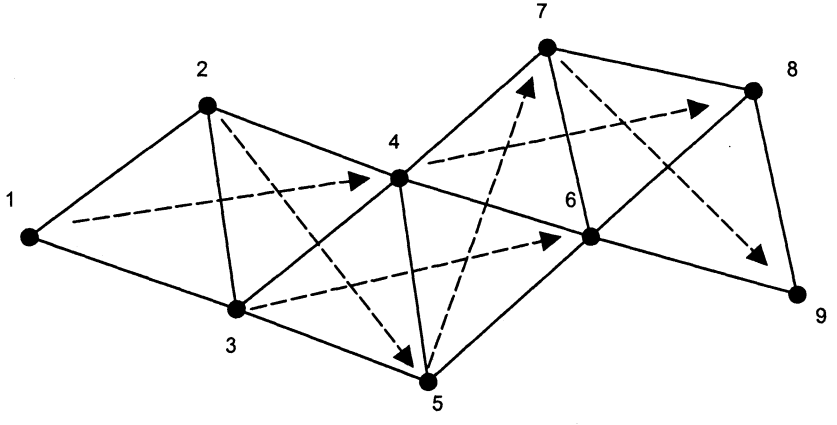


Рис. 5.1. Симплексный метод

На самом деле в таких процедурах принимаются специальные меры, предотвращающие циклы (циклические движения), а также используются правила уменьшения размера симплекса.

Описанный метод испытывает определенные трудности в связи с отсутствием механизма ускорения поиска в перспективных направлениях, а также в связи с продвижением вдоль искривленных оврагов и хребтов функции $u(x)$. В НМ-метод внесены соответствующие усовершенствования. Симплекс получает возможность изменять свою форму, вытягиваться, сжиматься и, таким образом, не будет оставаться симплексом. Поэтому для него используется более подходящее название — деформируемый многогранник, а сам метод часто называют *методом деформируемого многогранника.*

Основные операции НМ-метода:

1. *Отражение* — проектирование худшей вершины x^h через центр тяжести x^c оставшихся вершин (рис. 5.2):

$$x^r = x^c + \alpha(x^c - x^h), \quad \alpha > 0.$$

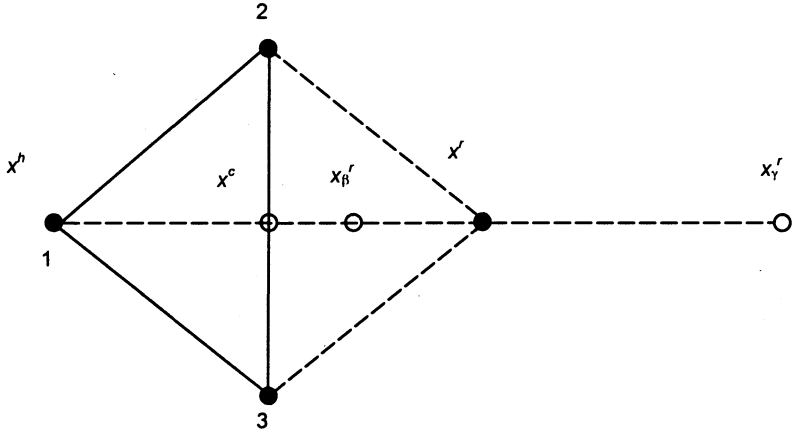


Рис. 5.2. Операция отражения

Здесь x^c — не вершина, а точка центра тяжести вершин 2, 3.

2. *Растяжение.* Если значение функции $u(x)$ в точке x^r оказывается лучше, чем в лучшей вершине из списка $\{1, 2, 3\}$, то выполняется растяжение в γ раз ($\gamma > 1$ — коэффициент растяжения), и точка x^r заменяется на точку x_γ^r.

3. *Сжатие.* Если в отраженной вершине x^r значение функции $u(x)$ хуже, чем во всех других вершинах (кроме x^h), то производится сжатие (с коэффициентом $0 < \beta < 1$) и точка x^r заменяется на точку x_β^r.

4. *Редукция.* Если в отраженной вершине значение функции $u(x)$ хуже, чем в точке x^h, то весь многогранник сжимается в 2 раза относительно лучшей вершины x^b (рис. 5.3).

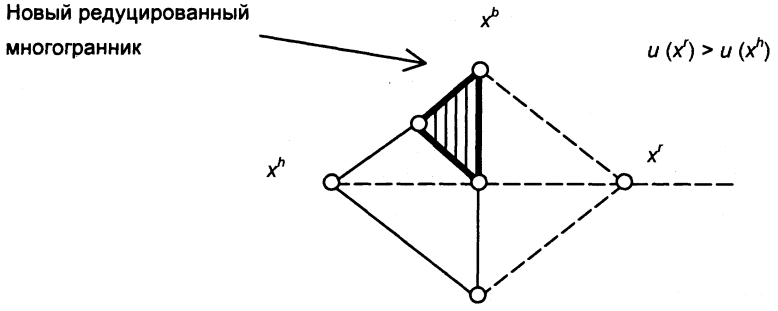


Рис. 5.3. Операция редукции

5. В остальных случаях операции 2, 3, 4 не производятся, а сам процесс продолжается для нового многогранника $\{2, 3, x^l\}$.
6. *Окончание процесса* производится, когда выполняется условие приблизительного равенства значений функции в вершинах текущего многогранника и в центре тяжести многогранника без учета худшей вершины. Могут быть использованы и другие условия окончания процесса.

5.1.2. Реализация адаптивной процедуры выбора на основе НМ-метода

Итак, решается следующая основная задача:

$$f_i(x) \to \min_{x \in X}, \; i = 1, \ldots, m, \; X \subset R^n,$$

где X — множество альтернатив; через $F = f(X)$ обозначим множество допустимых оценок.

Основная гипотеза: в многокритериальной задаче, в которой необходимо выбрать единственную альтернативу x^* из всех возможных (или небольшое число "наилучших" альтернатив), необходимо предположить, что существует единственная, хотя, возможно, и неизвестная ЛПР цель операции, описываемая скалярной функцией $u(x)$, причем

$$x^* = \arg\min_{x \in X} u(x).$$

Будем предполагать также, что эта функция $u(x)$ "согласована" с векторным отображением $f = (f_1, \ldots, f_m)$ в том смысле, что

$$\forall \arg\min_{x \in X} u(x) \in P(X),$$

где $P(X)$ — множество Парето заданной многокритериальной задачи.

Из теории многокритериальной оптимизации известно *(см. главу 2)*, что при определенных предположениях о выпуклости множества достижимости F найдутся такие весовые коэффициенты

$$\alpha_i^* \geq 0, \ i = 1, ..., m; \quad \sum_{i=1}^{m} \alpha_i^* = 1,$$

что линейная функция

$$J(x) = \sum_{i=1}^{m} \alpha_i^* f_i(x)$$

достигает своего минимума в точке x^*, т. к. последняя является точкой Парето. Поэтому нет никакой разницы, определять ли x^* или соответствующие весовые коэффициенты α^*.

Согласно этой основной идее процесс выбора будет протекать следующим образом.

Каждому $\alpha = (\alpha_1, ..., \alpha_m)$ будет соответствовать точка $x(\alpha) \in P(X)$, полученная как решение задачи минимизации линейной свертки

$$J(x) = \sum_{i=1}^{m} \alpha_i f_i(x) \to \min_{x \in X} \tag{5.1}$$

любым методом скалярной оптимизации. Векторная оценка $f(x(\alpha))$ будет одновременно оценкой α. А далее во множестве

$$A = \alpha = \left\{ (\alpha_1, ..., \alpha_m) \middle| \alpha_i > 0, \ \sum_{i=1}^{m} \alpha_i = 1 \right\}, \ A \subset R^m$$

реализуется НМ-метод, позволяющий последовательно находить

$$\alpha^* \to x^* = x(\alpha^*) \to f^* = f(x^*).$$

Таким образом, в отличие от возможного "прямого" подхода, когда процедура НМ-метода реализуется непосредственно во множестве $X \subset R^n$, мы здесь работаем в пространстве весовых коэффициентов R^m, причем, как правило, $m << n$. Правда, мы вынуждены решать вспомогательные оптимизационные задачи в пространстве R^n при минимизации функционала линейной свертки (5.1) для каждого пробного α. Но эти задачи ре-

шаются не в режиме диалога с пользователем и поэтому могут быть решены с меньшими временными затратами. Наиболее трудоемкая "диалоговая" часть процедуры выбора реализуется в пространстве R^m существенно меньшей размерности и в этом главный выигрыш построенной "косвенной" процедуры.

Важно отметить, что предлагаемый "косвенный" подход оказывается реализуемым и в случае, если X является множеством (конечным или бесконечным) объектов произвольной природы, т. е. требование $X \subset R^n$ является в данном случае, вообще говоря, непринципиальным. Важно лишь, чтобы для каждого из элементов $x \in X$ можно было вычислить соответствующую векторную оценку $f(x)$ согласно заданному отображению

$$X \xrightarrow{f} R^m.$$

Если абстрактные объекты из X являются непараметризованными (т. е. с ними не ассоциируются какие-либо числовые векторы), то обычно множество X оказывается конечным и вспомогательные задачи минимизации (5.1) решаются простым перебором вариантов. При этом диалоговый НМ-метод по-прежнему реализуется в числовом непрерывном пространстве весовых коэффициентов R^m, как это и было описано ранее.

Дополнительное преимущество рассмотренной косвенной реализации по сравнению с прямой реализацией НМ-метода во множестве $X \subset R^n$ заключается в том, что здесь мы гарантированно осуществляем выбор строго в пределах множества Парето и поэтому гарантируется эффективность получаемых решений, независимо от системы предпочтений ЛПР.

Легко видеть также, что вместо линейной свертки (5.1) можно использовать более эффективную свертку Джоффриона, реализующую принцип лексикографического упорядочения, не требуя свойства выпуклости множества достижимости F.

5.2. Выбор на основе метода *t*-упорядочения

Пусть решается детерминистская многокритериальная задача

$$f_i(x) \to \max,\ f_i : X \to R,\ \ i = \overline{1,m}, \tag{5.2}$$

где X — произвольное абстрактное множество. В данном разделе будем предполагать, что все критериальные функции f_i отражают "полезность"

объекта с позиций различных критериев и являются соизмеримыми в том смысле, что значения каждой критериальной функции изменяются в одних и тех же пределах $[a, b]$:

$$\forall x \in X : 0 \le a \le f_i(x) \le b, \quad i = \overline{1, m}. \tag{5.3}$$

Таким образом, мы предполагаем, что оценочные шкалы критериев являются числовыми и одинаковыми.

Отметим, что предположение о соизмеримости критериев является существенным и требует для каждой решаемой задачи отдельного обоснования и исследования.

Требование, связанное с необходимостью приведения всех числовых шкал к единому промежутку, выглядит весьма невинно и формально достигается с помощью, например, следующих простых преобразований:

$$\overline{f}_i(x) = a + (b-a)\frac{f_i(x) - \min f_i}{\max f_i - \min f_i}. \tag{5.4}$$

Здесь $\max f_i$, $\min f_i$ — максимальные и минимальные значения f_i соответственно. Новые оценочные функции $\overline{f}_i$ будут изменяться уже в пределах заданного промежутка $[a, b]$. При этом наименее предпочтительный по любому из частных критериев вариант получит оценку a, а наиболее предпочтительный — оценку b. Часто полагают $a = 0$, $b = 1$. Могут использоваться и другие (может быть, нелинейные) формулы нормировки, аналогичные (5.4).

Многие полагают, что проблемы соизмеримости числовых критериев вовсе не существует. Достаточно воспользоваться любыми соотношениями типа (5.4). Однако это, к сожалению, не так. Рассмотрим конкретный пример, показывающий, что отношение доминирования, устанавливаемое в пространстве нормированных оценок, неинвариантно (из-за изменения нижних и верхних границ) относительно изменения рассматриваемой совокупности объектов X.

Пусть для трех объектов $X = \{x_1, x_2, x_3\}$ имеем следующие оценки по трем частным критериям $f = (f_1, f_2, f_3)$:

$$f(x_1) = (10, 10, 3); \; f(x_2) = (8, 8, 10); \; f(x_3) = (0, 0, 0).$$

Тогда для преобразованных оценок при $[a, b] = [0, 1]$ получим

$$\overline{f}(x_1) = (1, \; 1, \; 0{,}3); \; \overline{f}(x_2) = (0{,}8, \; 0{,}8, \; 1); \; \overline{f}(x_3) = (0, \; 0, \; 0).$$

Если предположить, что все три частные критерия равноправны и нас интересует средняя оценка для каждого объекта, то тогда

$$\overline{f}_1(x_1) + \overline{f}_2(x_1) + \overline{f}_3(x_1) = 2{,}3 < \overline{f}_1(x_2) + \overline{f}_2(x_2) + \overline{f}_3(x_2) = 2{,}6$$

и, следовательно, объект x_2 оказывается более предпочтительным, чем объект x_1 (по "средней полезности").

Но если исключить объект x_3, как наихудший по всем трем критериям, то получим обратное утверждение. Действительно, в этом случае

$$\overline{f}(x_1) = (1,1,0);\ \ \overline{f}(x_2) = (0,\ 0,\ 1)$$

и

$$\overline{f}_1(x_1) + \overline{f}_2(x_1) + \overline{f}_3(x_1) = 2 > \overline{f}_1(x_2) + \overline{f}_2(x_2) + \overline{f}_3(x_2) = 1.$$

Итак, мы получили обескураживающий результат. Если исходная совокупность объектов X содержит три объекта x_1, x_2, x_3, то оказывается, что объект x_2 лучше, чем объект x_1. Если же исключить из рассмотрения (худший по всем критериям) объект x_3, то получим, что объект x_1 оказывается лучше, чем объект x_2. Безобидная на первый взгляд процедура нормировки оказывается на деле весьма сложной, т. к. приводит к явно антиинтуитивным результатам. Можно пытаться исправить ситуацию и, например, фиксировать верхние и нижние границы критериальных функций раз и навсегда (для данной задачи) независимо от конкретного набора объектов и их оценок. При этом мы получим однозначность, но ситуация в целом вряд ли улучшится, т. к. установленные границы, по существу, и будут определять соответствующее отношение доминирования, и в этом смысле произвол сохраняется.

Итак, пусть критериальные функции (или просто — критерии) соизмеримы и удовлетворяют условиям (5.3). В качестве примера изначально соизмеримых критериев можно привести систему школьных оценок по нескольким предметам f_i.

Определение 5.1

Нормированные критерии f_i и f_j называются равноценными (что записывается в виде $f_i = f_j$), если всякие две векторные оценки Z, W, где

$$\begin{aligned} &Z = (z_1; ...; z_i; ...; z_j; ...; z_m),\ \ Z = f(x),\ \ x \in X \\ &W = (z_1; ...; z_i + \delta; ...; z_j - \delta; ...; z_m) \end{aligned} \qquad (5.5)$$

одинаковы по предпочтительности при любом δ (большем или меньшем нуля), удовлетворяющем неравенствам:

$$a \le z_i + \delta \le b,\ \ a \le z_i - \delta \le b.$$

Легко видеть, что суммы частных оценок в позициях i, j у векторных оценок Z, W совпадают.

Таким образом, если, например, два школьника оцениваются по четырем предметам и имеют оценки (которые необходимо максимизировать)

$$Z = (5, 4, 4, 3),\ W = (5, 5, 3, 3), \qquad (5.6)$$

то при условии равноценности критериев f_2, f_3 приведенные векторные оценки будут одинаковы по предпочтительности, т. к. 4 + 4 = 5 + 3, а остальные оценки совпадают.

Следовательно, если критерии f_i, f_j равноценны, то можно "забрать" δ единиц у частной оценки z_j в (5.5) и "передать" их частной оценке z_i. При этом получим векторную оценку, одинаковую с исходной по предпочтительности.

Если в приведенном примере (5.6) считать, что оценка Z предпочтительнее, чем W, то естественно предположить, что критерий f_3 важнее критерия f_2. Дадим соответствующее определение.

Определение 5.2

Критерий f_i более важен, чем критерий f_j (что записывается в виде $f_i > f_j$), если векторная оценка

$$Z = (z_1; \ldots; z_i; \ldots, z_j; \ldots; z_m)$$

менее предпочтительна, чем оценка

$$W = (z_1; \ldots ; z_i + \delta; \ldots; z_j - \delta; \ldots; z_m),$$

где

$$\delta \in \{\delta > 0 \mid z_i + \delta \le b,\ a \le z_j - \delta\}.$$

Таким образом, перенос δ единиц ($\delta > 0$) с частной оценки z_j на частную оценку z_i приводит к улучшению ситуации, если $f_i > f_j$.

Определения 5.1, 5.2 показывают, как может интерпретироваться дополнительная ординальная (порядковая) информация пользователя (ЛПР) об относительной важности частных критериев, на основе которой и происходит сокращение множества Парето решаемой многокритериальной задачи. Важно понимать, что одна и та же ординальная информация, задаваемая в виде цепочки равенств и неравенств, например, вида

$$f_i > f_j = f_k > f_l,$$

в других процедурах выбора может получать совершенно иную интерпретацию, что, в свою очередь, может приводить и к другим системам предпочтений, и к другим результатам выбора. Соответствующие вопросы рассмотрены в этом разделе далее.

Приведенные в определениях 5.1, 5.2 интерпретации ординальной информации ЛПР позволяют строить отношения доминирования более сильные, чем отношение Парето (что и приводит к сужению последнего, а это наша основная задача). Рассмотрим пример.

Пусть

$$Z = (1, 0{,}5, 0{,}1, 0{,}2);$$

$$W = (0{,}4, 0{,}9, 0{,}1, 0{,}2)$$

и пусть утверждается, что критерий f_1 важнее, чем $f_2 : f_1 > f_2$.

Эти векторные оценки, очевидно, несравнимы по Парето. Рассмотрим оценку

$$W' = (0{,}8, 0{,}5, 0{,}1, 0{,}2)$$

полученную из W с помощью "переноса" числа 0,4 со второй позиции в первую. Имеем, согласно определению 5.2,

$$W' \succ W,$$

т. к. более важному критерию f_1 в оценке W' соответствует большая оценка (0,8 вместо 0,4). И, поскольку имеем

$$Z \overset{p}{\succ} W'$$

(доминирование по Парето), то естественно считать

$$Z \succ W' \succ W$$

и в результате

$$Z \succ W.$$

Таким образом, оценки Z, W оказываются уже сравнимыми, и оценка W может быть отброшена. Здесь везде предполагается, что рассматриваемые отношения доминирования являются транзитивными.

Методы упорядочения альтернатив, основанные на рассмотренной процедуре "переноса" (transfer) с учетом ординальной информации пользователя, будем называть *методами t-упорядочения.*

Рассмотрим укрупненный алгоритм, реализующий метод t-упорядочения.

В качестве исходной информации для алгоритма t-упорядочения принимается множество S высказываний пользователя (ЛПР) об относительной важности частных критериев вида:

$$S = \{f_i = f_j; \ldots; f_q > f_p\}.$$

Мы будем предполагать, что множество S скорректировано следующим образом. Во-первых, необходимо проверить и при необходимости обеспечить непротиворечивость высказываний из S, может быть, путем проведения дополнительного диалога с пользователем для уточнения его системы предпочтений. Во-вторых, необходимо расширить множество S за счет добавления новых высказываний, являющихся транзитивными следствиями уже имеющихся (выполнить операцию транзитивного замыкания). Именно, если мы, например, уже ввели два высказывания

$$f_i > f_j;\ f_j > f_k,$$

то естественно ввести новое высказывание

$$f_i > f_k$$

и т. п.

Пусть теперь $Z = (z_1, \ldots, z_m)$; $W = (w_1, \ldots, w_m)$ — две векторные оценки, которые необходимо сравнить с учетом дополнительной скорректированной информации S.

Если эти оценки сравнимы по Парето, то задача решена. В противном случае вектор Z фиксируется, а по вектору W формируются следующие два множества (может быть, бесконечные, даже если исходные множества оценок конечны).

1. WE — множество W-эквивалентных векторов (включающее сам вектор W), полученное из W с помощью всех возможных переносов δ между парами равноценных критериев. Следовательно, множество WE строится с учетом всех данных типа $f_i = f_j$ из S.
2. WI — множество W-улучшенных векторов, каждый из которых получен с помощью возможных переносов δ с учетом всех данных

$$f_i = f_j;\ f_q > f_p.$$

При этом предполагается, что переносы согласно информации $f_q > f_p$ производятся только с целью улучшения вновь полученного вектора и, по крайней мере, одна такая улучшающая передача выполнена для любого вектора из WI.

Далее новое отношение предпочтения $\overset{t}{\succ}$ строится следующим образом:

$$Z \overset{t}{\succ} W \leftrightarrow [\exists W' \in WE : Z \overset{p}{\succ} W'] \vee [\exists W'' \in WI : Z \overset{p}{\underset{\sim}{\succ}} W'']. \qquad (5.7)$$

Здесь через $\left(\begin{matrix} \overset{p}{\succ} \\ \sim \end{matrix}\right)$ обозначено нестрогое предпочтение вида

$$Z \overset{p}{\underset{\sim}{\succ}} W \leftrightarrow \forall i \in [1:m] : z_i \geq w_i .$$

Определение (5.7) имеет весьма простой смысл. Если вектор Z строго лучше, чем некоторый вектор W', эквивалентный W, или нестрого лучше, чем некоторый вектор W'', который, в свою очередь, строго лучше, чем W, то полагаем, что Z строго лучше W.

Частным случаем изложенного метода t-упорядочения является метод, предложенный В. В. Подиновским (далее — метод П-упорядочения). Он основан на том, что мы имеем возможность формировать множества, аналогичные WE и WI, с помощью перестановок численных значений оценок между равноценными и неравноценными критериями. Например, пусть дана векторная оценка

$$Z = (5, 10, 6)$$

и известно, что $f_1 > f_2$. Тогда по Подиновскому оценка

$$Z' = (10, 5, 6),$$

полученная из Z перестановкой чисел 5, 10, будет признана лучшей, чем Z, т. к. на место более "важного" критерия f_1 пришло большее значение (10 вместо 5). Если бы критерии f_1 и f_2 были равноценными, то оценки Z, Z' считались бы эквивалентными.

Очевидно, в методе П-упорядочения множества WE, WI оказываются конечными и могут быть практически построены без особых вычислительных проблем.

Недостатком метода П-упорядочения является его недостаточная "мощность". Например, пусть ставится задача сравнения двух векторных оценок

$$W = (7, 9, 6),$$

$$Z = (5, 10, 6)$$

при наличии ординальной информации $f_1 > f_2$. Эти оценки, очевидно, несравнимы по Парето. Несравнимы они и по методу П-упорядочения (никакие перестановки численных значений оценок между f_1, f_2 не приводят к их сравнимости по Парето). В то же время легко видеть, что согласно методу t-упорядочения для

$$Z' = (6, 9, 6),$$

полученной из Z с помощью переноса $\delta = 1$ со второй позиции в первую, мы имеем

$$Z' \succ Z, \; W \succ Z' \succ Z$$

и, следовательно,

$$W \overset{t}{\succ} Z.$$

В то же время с позиций "физического смысла" метод t-упорядочения представляется столь же естественным, что и метод Подиновского.

5.3. Задачи с малым числом критериев и альтернатив

Решается многокритериальная задача

$$f_i(x) \to \max, \ i = 1, ..., m, \tag{5.8}$$

где $x \in X = \{x_1, x_2, ..., x_n\}$.

Таким образом, X — конечное множество, содержащее n элементов x_i. Предполагается, что числа m, n относительно невелики, т. к. именно они в рассматриваемых ниже методах будут определять трудоемкость диалога с пользователем в реальном времени по извлечению дополнительной информации о задаче.

Далее будут рассмотрены метод Саати и метод Коггера и Ю.

5.3.1. Проблема ранжирования объектов по "важности". Матрица попарных сравнений

Пусть задано конечное множество объектов

$$P = \{p_1, p_2, ..., p_m\}$$

и нам необходимо построить вектор

$$\alpha = (\alpha_1, \alpha_2, ..., \alpha_m)$$

с неотрицательными вещественными компонентами α_i, такими, что

$$\sum_{i=1}^{m} \alpha_i = 1..$$

Числа α_i будем интерпретировать как весовые коэффициенты, определяющие относительную "важность" или "полезность" объектов p_i. Чем большее значение α_i соотносится с объектом p_i, тем выше "полезность" этого объекта. Например, в исходной многокритериальной задаче (5.8) ранжироваться могут частные критерии f_i, а также непосредственно альтернативы x_i из X.

Основным объектом в рассматриваемых методах является треугольная матрица S, называемая здесь *матрицей попарных сравнений*:

$$S = \begin{bmatrix} 1 & \alpha_{12} & \alpha_{13} & \dots & \alpha_{1m} \\ 0 & 1 & \alpha_{23} & \dots & \alpha_{2m} \\ 0 & 0 & 1 & \dots & \alpha_{3m} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 1 & \alpha_{m-1,m} \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Элемент $\alpha_{ij} = \alpha_i / \alpha_j$ матрицы интерпретируется как коэффициент превосходства i-го объекта p_i над j-м объектом p_j из множества P. Если $\alpha_i / \alpha_j > 1$, то объект p_i "важнее" объекта p_j и т. д.

Предполагается, что пользователь или ЛПР имеет возможность отвечать на вопросы типа: "Во сколько раз объект p_i превосходит объект p_j по важности?"

Коэффициенты α_{ij} могут выбираться пользователем из фиксированной балльной шкалы, например (Саати):

$$\left\{\frac{1}{9}, \frac{1}{8}, \frac{1}{7}, \frac{1}{6}, \frac{1}{5}, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, 1, 2, 3, 4, 5, 6, 7, 8, 9\right\}.$$

Для облегчения работы пользователя целые числа шкалы могут получать соответствующую смысловую интерпретацию, например:

- 1 — равная важность;
- 3 — слабое превосходство;
- 5 — сильное превосходство;
- 7 — очень сильное превосходство;
- 9 — абсолютное превосходство;
- 2, 4, 6, 8 — промежуточные случаи.

Очевидно, что шкала должна содержать (и содержит) соответствующие обратные значения.

5.3.2. Метод Саати. Метод Коггера и Ю

Задача состоит в отыскании вектора весовых коэффициентов

$$\alpha = (\alpha_1, \alpha_2, \dots, \alpha_m)$$

по известной матрице попарных сравнений S.

Согласно методу Саати по треугольной матрице S строится следующая полнозаполненная матрица $\overline{S}$:

$$\overline{S} = \begin{bmatrix} 1 & \alpha_{12} & \alpha_{13} & \dots & \alpha_{1m} \\ \alpha_{21} & 1 & \alpha_{23} & \dots & \alpha_{2m} \\ \dots & \dots & \dots & \dots & \dots \\ \alpha_{m1} & \alpha_{m2} & \alpha_{m3} & \dots & 1 \end{bmatrix},$$

где элементы нижней треугольной части α_{ij} ($i > j$) матрицы удовлетворяют соотношениям

$$\alpha_{ij} = 1 / \alpha_{ji}.$$

Легко доказать, что искомый вектор

$$\alpha = (\alpha_1, \alpha_2, \dots, \alpha_m)$$

является собственным вектором матрицы $\overline{S}$, соответствующим максимальному собственному числу матрицы $\lambda = m$ и может быть найден как решение системы уравнений:

$$\overline{S}\alpha = \lambda_{\max}\alpha. \qquad (5.9)$$

Существует единственное решение данной системы линейных алгебраических уравнений, удовлетворяющее условию

$$\sum_{i=1}^{m} \alpha_i = 1.$$

Если матрица системы (5.9) задана неточно, то предлагается численно определить ее максимальное собственное число и соответствующий собственный вектор.

Собственно метод Саати сводится к следующему.

Предполагается, что частные критерии f_i не обязательно являются числовыми функциями и могут иметь качественный неформальный характер. В этом случае для каждого частного критерия f_i ставится задача ранжирования объектов $\{x_1, x_2, \dots, x_n\}$ с построением на основе диалога с пользователем соответствующей матрицы попарных сравнений и определением вектора весов

$$\alpha^i = (\alpha^i_1, \dots, \alpha^i_n).$$

Полученные числа α^i_j интерпретируются как значения $f_i(x_j)$, $j = \overline{1, n}$. Таким образом, каждая альтернатива получает уже числовую оценку по каждому из частных критериев.

Далее осуществляется аналогичная операция по ранжированию самих частных критериев по важности с построением вектора весов

$$\beta = (\beta_1, ..., \beta_m).$$

В качестве оптимальной альтернативы (их может быть несколько) выбираем

$$x^* = \arg\max_i J(x_i),$$

где

$$J(x_i) = \sum_{k=1}^{m} \beta_k f_k(x_i), \quad i = \overline{1,n}; \quad f_k(x_i) = \alpha_i^k.$$

Метод Коггера и Ю отличается от метода Саати тем, что для нахождения вектора весовых коэффициентов ранжируемых объектов используется не система уравнений (5.9), а система вида

$$TS\alpha = \alpha, \tag{5.10}$$

где S — треугольная матрица попарных сравнений, а

$$T = \mathrm{diag}[1/m, 1/m-1, ..., 1].$$

В данном случае матрица системы — треугольная, что облегчает решение задачи.

5.3.3. Обсуждение

При практическом использовании рассмотренных методов возникает целый ряд вопросов.

В стандартных вариантах алгоритмов предполагается, что все элементы матрицы попарных сравнений S определяются независимо в результате диалога, что может приводить к противоречивости ответов пользователя в смысле нарушений очевидных равенств вида

$$\alpha_{ik}\alpha_{kj} = \alpha_{ij}; \quad \left(\frac{\alpha_i}{\alpha_k} \times \frac{\alpha_k}{\alpha_j} = \frac{\alpha_i}{\alpha_j} \right).$$

Если, как это и рекомендует автор, продолжать решать системы (5.9), (5.10), то непонятен смысл механизма обработки заведомо противоречивой информации.

Таким образом, процедура независимой оценки элементов α_{ij} вызывает определенные сомнения и трудности.

С другой стороны, если полагать, что мы должны в конечном итоге получить непротиворечивую информацию от пользователя — может быть, с помощью дополнительных уточняющих вопросов, — то тогда можно поступать значительно проще.

В этом случае можно вообще отказаться от решения каких-либо систем уравнений. Действительно, для непосредственного определения

$$\alpha = (\alpha_1, \alpha_2, ..., \alpha_m)$$

достаточно, например, получить от пользователя цепочку коэффициентов превосходства вида

$$\alpha_{12}, \alpha_{13}, ..., \alpha_{1m}$$

или

$$\alpha_{12}, \alpha_{23}, ..., \alpha_{m-1,m}. \qquad (5.11)$$

При этом существенно сокращается трудоемкость диалога и достигается непротиворечивость информации.

Могут быть предложены процедуры получения от пользователя избыточной информации для уточнения его оценок на основе определенных механизмов усреднения, но это — предмет отдельного рассмотрения.

Продолжим обсуждение. Ясно, что применение фиксированной балльной шкалы Саати со смысловой интерпретацией в определенной степени провоцирует пользователя на дачу противоречивых, в указанном ранее смысле, ответов. Действительно, пусть, например, объект p_1 абсолютно превосходит объект p_2 и $\alpha_{12} = 9$ согласно шкале. Пусть также объект p_2 абсолютно превосходит объект p_3 и, следовательно, $\alpha_{23} = 9$. Спрашивается, какое число назовет пользователь в качестве α_{13} — тоже 9 или 81? Последнее число в шкале отсутствует.

По-видимому, в подобных процедурах целесообразно использовать так называемые "транзитивные" шкалы типа:

$$\begin{gathered} \text{слабое превосходство} = a, \\ \text{сильное превосходство} = aa, \\ \text{очень сильное превосходство} = aaa, \\ \text{абсолютное превосходство} = aaaa \text{ и более.} \end{gathered} \qquad (5.12)$$

В этом случае мы в значительной степени будем избавлены от весьма неожиданных согласно шкале Саати утверждений типа: если объект p_1 слабо превосходит объект p_2 ($\alpha_{12} = 3$), а объект p_2, в свою очередь, слабо превосходит объект p_3 ($\alpha_{23} = 3$), то объект p_1 абсолютно превосходит объект p_3 ($\alpha_{13} = 9$)!

Согласно шкале (5.12) комбинация двух слабых превосходств дает сильное превосходство, два сильных превосходства дают абсолютное превосходство и т. д. Получаемые согласно этой шкале результаты имеют весьма понятную смысловую интерпретацию. Выбор конкретного значения a для базы шкалы — это отдельный вопрос. Можно, например, положить $a = 1,5$ или $a = 2$ с получением шкал, представленных в табл. 5.1.

Таблица 5.1. *Коэффициенты превосходства*

Смысловая интерпретация	$a = 1,5$	$a = 2$
слабое превосходство	1,5	2
сильное превосходство	2,25	4
очень сильное превосходство	3,38	8
абсолютное превосходство	5,06 и более	16 и более

5.3.4. Простой алгоритм выбора

В данном разделе предлагается простой алгоритм выбора на основе информации (5.11). Поясним принципиальную схему метода на конкретном примере.

Рассмотрим проблему выбора научного руководителя студентом старшего курса университета. Глобальный показатель "качества", характеризующий правильность выбора, будем связывать с *общим удовлетворением работой* в конкретной научной группе. Этот показатель является достаточно расплывчатым и неопределенным, поэтому используются соответствующие критерии-заместители и задача трансформируется к некоторой многокритериальной задаче. Будем рассматривать следующие частные критерии оптимальности, характеризующие в совокупности исходный глобальный показатель:

1. Перспективность проводимых в группе исследований с позиций последующего трудоустройства (f_1).
2. Личный интерес студента к проводимым исследованиям (f_2).
3. Возможность получения дополнительной заработной платы в процессе обучения (f_3).
4. Связь научной группы с конкретными фирмами, принимающими на работу молодых специалистов (f_4).

5. Профессионализм, характер и человеческие качества лично научного руководителя (f_5).

6. Состав научной группы и отношения в коллективе (f_6).

Предполагается, что все введенные частные показатели необходимо максимизировать, т. е. большему значению каждого показателя будет соответствовать более желаемое состояние для студента.

Рассмотрим ситуацию выбора, когда имеется три потенциальных научных руководителя, обозначенных буквами A, B, C.

Будем следовать предлагаемому простому алгоритму выбора с транзитивной шкалой и базой $a = 2$ (см. табл. 5.1). Построим вектор весов для сформулированных частных критериев. Необходимо задать пользователю пять вопросов и определить в результате вектор коэффициентов превосходства (5.11). Будем считать, что по результатам диалога были получены следующие данные:

$$\alpha_{12} = 2;\ \alpha_{23} = 4;\ \alpha_{34} = 1/4;\ \alpha_{45} = 1;\ \alpha_{56} = 4.$$

Здесь равенство $\alpha_{12} = 2$, например, означает, что частный критерий f_1 в два раза превосходит по "важности" критерий f_2 и т. д.

Воспользовавшись соотношением

$$\alpha_{ij} = \alpha_i / \alpha_j$$

и условием нормированности вектора α, непосредственно получаем:

$$\alpha_1 = 0{,}364;\ \alpha_2 = 0{,}182;\ \alpha_3 = 0{,}045;\ \alpha_4 = 0{,}182;\ \alpha_5 = 0{,}182;\ \alpha_6 = 0{,}045.$$

Далее переходим к процедуре вычисления значений частных критериев оптимальности, соответствующих трем вариантам A, B, C.

Вначале с помощью того же самого подхода ранжируем варианты A, B, C по критерию f_1 (перспективность исследований). Пусть пользователь указал следующие значения коэффициентов превосходства:

$$\alpha^1_{12} = 1;\quad \alpha^1_{23} = 1/2.$$

Соответствующий вектор весов α^1 имеет компоненты

$$0{,}250;\ 0{,}250;\ 0{,}500.$$

которые интерпретируются как значения функции f_1 для трех вариантов A, B, C:

$$f_1(A) = 0{,}250;\ f_1(B) = 0{,}250;\ f_1(C) = 0{,}500.$$

Аналогично определяем значения остальных частных критериев для вариантов A, B, C:

$$\alpha_{12}^2 = 2; \quad \alpha_{23}^2 = 2;$$

$$f_2(A) = 0{,}571; \; f_2(B) = 0{,}286; \; f_2(C) = 0{,}143.$$

$$\alpha_{12}^3 = 1; \; \alpha_{23}^3 = 1;$$

$$f_3(A) = 0{,}333; \; f_3(B) = 0{,}333; \; f_3(C) = 0{,}333.$$

$$\alpha_{12}^4 = 1/2; \; \alpha_{23}^4 = 1/4;$$

$$f_4(A) = 0{,}091; \; f_4(B) = 0{,}182; \; f_4(C) = 0{,}727.$$

$$\alpha_{12}^5 = 4; \; \alpha_{23}^5 = 2;$$

$$f_5(A) = 0{,}727; \; f_5(B) = 0{,}182; \; f_5(C) = 0{,}091.$$

$$\alpha_{12}^6 = 1/2; \; \alpha_{23}^6 = 2;$$

$$f_6(A) = 0{,}250; \; f_6(B) = 0{,}500; \; f_6(C) = 0{,}250.$$

Воспользовавшись методом линейной свертки,

$$J(x_i) = \sum_{k=1}^{6} \alpha_k f_k(x_i),$$

получим значения обобщенного критерия оптимальности для трех вариантов $x_1 = A$, $x_2 = B$, $x_3 = C$:

- $J(A) = 0{,}364 \times 0{,}250 + 0{,}182 \times 0{,}571 + 0{,}045 \times 0{,}333 + 0{,}182 \times 0{,}091 + 0{,}182 \times 0{,}727 + 0{,}045 \times 0{,}250 = 0{,}370$;
- $J(B) = 0{,}364 \times 0{,}250 + 0{,}182 \times 0{,}286 + 0{,}045 \times 0{,}333 + 0{,}182 \times 0{,}182 + 0{,}182 \times 0{,}182 + 0{,}045 \times 0{,}500 = 0{,}247$;
- $J(C) = 0{,}364 \times 0{,}500 + 0{,}182 \times 0{,}143 + 0{,}045 \times 0{,}333 + 0{,}182 \times 0{,}727 + 0{,}182 \times 0{,}091 + 0{,}045 \times 0{,}250 = 0{,}383$.

Следовательно, наиболее перспективным с позиций применяемого метода признается выбор руководителя C. Однако видно, что выбор A оказывается почти столь же хорошим.

Замечание 5.1

Ясно, что если нет оснований считать множество достижимости рассматриваемой многокритериальной задачи выпуклым, целесообразно вместо линейной свертки в качестве обобщенного критерия использовать свертку Джоффриона, основанную на комбинации линейной и максиминной сверток.

Легко подсчитать, что в предложенном методе общее число сравнений объектов по важности, выполняемых пользователем, равно

$$N_1 = m - 1 + m(n-1) = mn - 1,$$

где m — число частных критериев, n — количество альтернатив.

В стандартных процедурах Саати и Коггера и Ю число сравнений равно

$$N_2 = \frac{m^2 - m}{2} + m\frac{n^2 - n}{2},$$

что может быть существенно больше N_1. Так, для $m = 6$, $n = 10$ имеем:

$$N_1 = 59, N_2 = 285.$$

Кроме того, как уже указывалось, при использовании предлагаемого подхода нет необходимости решать какие бы то ни было линейные системы и искать собственные числа матриц.

5.4. Метод ограничений

Простым и часто применяемым методом сжатия множества Парето является метод ограничений.

Решается стандартная многокритериальная задача

$$\begin{aligned} &f_i(x) \to \max, \quad x \in D, \\ &f_i : D \to R, \quad i = 1, ..., m, \end{aligned} \tag{5.13}$$

где D — произвольное абстрактное множество.

На первом этапе одним из известных методов строится множество Парето $P(D)$. Далее метод ограничений реализуется в соответствии со следующей последовательностью шагов.

Шаг 1. Пользователю предлагается назначить нижние допустимые границы t_i для всех m критериальных функций:

$$f_i(x) \geq t_i, \ i = 1, ..., m. \tag{5.14}$$

Предполагается, что указанные значения критериальных функций дают удовлетворяющие пользователя варианты.

Шаг 2. Строится подмножество множества Парето, состоящее из точек, удовлетворяющих неравенствам (5.14):

$$D_t = \{x \in P(D) \mid f_i(x) \geq t_i, \ i = 1, ..., m\}, \ D_t \subset P(D).$$

Шаг 3. Если D_t — пустое множество, то пользователю предлагается ослабить требования с помощью уменьшения какого-то из чисел t_i. Далее переходим к шагу 2. Если D_t не пусто — переходим к шагу 4.

Шаг 4. Выбирается $\forall x \in D_t$ и предъявляется пользователю в качестве кандидата на "решение" задачи (5.13). Если решение удовлетворяет пользователя, то процесс завершается. В противном случае переходим к шагу 5.

Шаг 5. Пользователю предлагается назначить новую (увеличенную) нижнюю границу по одному из критериев и осуществляется переход к шагу 2.

Как правило, в процессе диалога пользователь получает дополнительную информацию о задаче в виде диапазонов изменения векторных оценок для элементов множества Парето или множеств D_t. Иногда целесообразно иметь информацию о так называемой "идеальной" точке, соответствующей лучшим возможным значениям по всем критериям (такая точка реально обычно не существует, но позволяет оценить множество допустимых верхних границ). Кроме того, в процессе решения "хорошая" СППР должна информировать пользователя о структуре множеств $P(D)$, D_t. Например, для случая конечного множества D пользователь на шаге 4, возможно, захочет уточнить свой выбор (даже если он его первоначально и удовлетворил) с помощью просмотра других точек из D_t, удовлетворяющих той же системе ограничений. Должна быть обеспечена также возможность возврата назад к прежним вариантам и возможность выбора новых начальных условий, новых критериев и т. д.

Метод ограничений целесообразно использовать на завершающей стадии процесса выбора, например, после применения метода t-упорядочения. Тогда в качестве исходного для метода ограничений множества будет использоваться не $P(D)$, а некоторое его подмножество, что сократит наиболее трудоемкую диалоговую часть процедуры выбора.

Непосредственно в методе ограничений какая-либо ординальная информация не используется, а сокращение исходного множества альтернатив производится в процессе поступления дополнительной информации от пользователя в виде последовательности наборов нижних границ $\{t_i\}$.

5.5. Рандомизированные стратегии принятия решений

В данном разделе мы рассмотрим еще один подход к интерпретации ординальной информации пользователя об относительной важности частных критериев оптимальности многокритериальной задачи (5.13).

Основная идея заключается в следующем. Рассматривается какая-либо скалярная свертка (обычно линейная) векторного критерия оптимальности $f = (f_1, ..., f_m)$ с весовыми коэффициентами $\{\alpha_i\}$. Например, вводится следующий "обобщенный", "глобальный", "сводный" и т. п. критерий:

$$J(x, \alpha) = \langle f(x), \alpha \rangle = \sum_{i=1}^{m} \alpha_i f_i(x), \qquad (5.15)$$

где $\alpha = (\alpha_1, ..., \alpha_m)$ — вектор неотрицательных весовых коэффициентов, удовлетворяющих условию

$$\Sigma \alpha_i = 1.$$

Далее, поступающая от пользователя ординальная информация вида

$$f_j \succ f_k \text{ или } f_l \sim f_p$$

интерпретируется с помощью соответствующих неравенств:

$$\alpha_j \geq \alpha_k \text{ или } \alpha_l = \alpha_p. \qquad (5.16)$$

В результате исходное множество весовых коэффициентов

$$A = \{\alpha = (\alpha_1, ..., \alpha_m) \mid \alpha_i \geq 0, \ \Sigma \alpha_i = 1\}$$

сужается до некоторого подмножества $\overline{A} \subset A$. Для любого $\alpha \in \overline{A}$ выполняются все неравенства (5.16). В частном случае может быть $\overline{A} = A$.

Считаем, что α — случайный вектор с равномерной плотностью распределения во множестве $\overline{A}$. Тогда значение $J(x, \alpha)$ для любого фиксированного x также будет случайной величиной. Оптимальное значение x^* предлагается выбирать из условия

$$x^* = \arg\max_x M(J(x, \alpha)), \qquad (5.17)$$

где $M(...)$ — знак математического ожидания.

Описание основной идеи закончено. Дадим некоторые комментарии.

Если используется линейная свертка (5.15), то

$$M(J(x, \alpha)) = \langle f(x), M(\alpha) \rangle$$

и задача сводится к определению математических ожиданий весовых коэффициентов, одних и тех же для любого x. Полученные математические ожидания $M(\alpha_i)$ будут играть роль весовых коэффициентов в стандартном методе линейной свертки *(см. разд. 2.3)* со всеми вытекающими особенностями. В частности, при невыпуклой структуре множества $f(D)$ заведомо (независимо от ординальной информации пользователя) выпадают из рассмотрения эффективные решения, лежащие на невыпуклых

участках границы. Последнее замечание определяет существенный и определяющий недостаток рассмотренного подхода. Однако если $f(D)$ — выпуклое множество, то отмеченных трудностей не возникает.

Пример 5.1. Рассмотрим многокритериальную задачу выбора (5.13) с $m = 2$. Пусть множество достижимости $f(D)$ невыпукло и представлено на рис. 5.4.

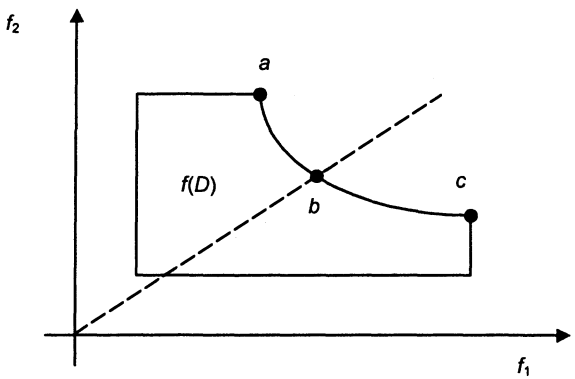


Рис. 5.4. Невыпуклое множество достижимости

Множество Парето в пространстве оценок, очевидно, совпадает с отрезком границы $[a, c]$. При условии "одинаковой важности" критериев f_1, f_2 пользователь в качестве окончательного решения, вполне возможно, выбрал бы точку b, где $f_1 = f_2$. Однако, как следует из рассмотрения в *разд. 2.3*, при любых наборах весовых коэффициентов α с помощью конструкции (5.15) могут быть получены только точки a и c. Точка b не будет решением ни при каких видах ординальной информации пользователя.

Основная проблема при численной реализации данного метода заключается в организации процедуры "вбрасывания" заданного количества m-мерных случайных точек в построенную, обычно достаточно сложную, область $\overline{A}$. В простейших случаях результат может быть получен аналитически.

Пример 5.2. Пусть решается многокритериальная задача (5.13) с $m = 2$ и пусть из ординальной информации, заданной пользователем, получим $\alpha_1 > \alpha_2$. Тогда множества A и $\overline{A}$ могут быть представлены с помощью рис. 5.5.

В данном случае, очевидно,

$$M(\alpha_1) = 0{,}75;\ M(\alpha_2) = 0{,}25.$$

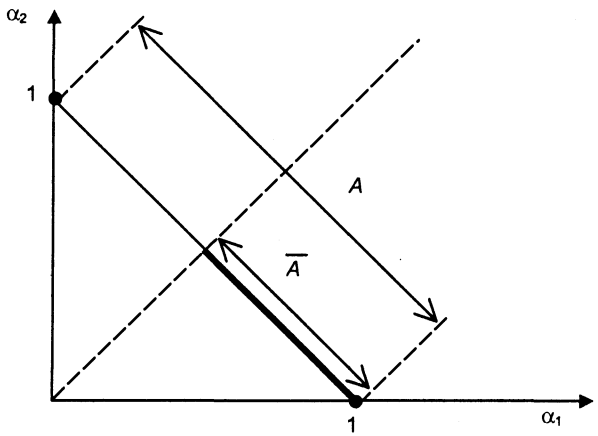


Рис. 5.5. Пространство весовых коэффициентов

(Это координаты середины отрезка $\overline{A}$.)

Таким образом, мы имеем более чем однозначную интерпретацию достаточно слабого утверждения пользователя о том, что $\alpha_1 > \alpha_2$. В этом также заключается определенная особенность метода: он, может быть, излишне категоричен.

В общем случае произвольной структуры множества $f(D)$, когда его выпуклость не может быть гарантирована, целесообразно в соответствии с рекомендациями *разд. 2.3* использовать метод лексикографической оптимизации:

$$J(x,\alpha) = \left\{ \min_i \alpha_i f_i(x), \sum_{i=1}^{m} f_i(x) \right\} \to \max_{x \in D} \qquad (5.18)$$

или

$$J(x,\alpha) = \sum_{i=1}^{m} f_i(x) \to \max_{x \in X(\alpha)},$$

$$X(\alpha) = Arg \max_{x \in D} \min_i \alpha_i f_i(x).$$

При этом оптимальная точка x^* также находится согласно выражению (5.17).

Простейшая прямая реализация изложенного метода на основе свертки (5.18) состоит в следующем. Пусть множество D — конечно. Для каждого $x \in D$ генерируется последовательность $\alpha^1, \ldots, \alpha^N$ векторов $\alpha \in \overline{A}$ в соответствии с равномерной плотностью распределения вероятностей. Для каждого α^k из этой последовательности вычисляется значение $J(x, \alpha^k)$ по формуле (5.18). Далее вычисляется среднее значение J как оценка для

$M(J(x, \alpha))$ при данном x. В соответствии с полученными значениями $M(J(x, \alpha))$ для всех x выбирается наилучшая точка x^* согласно представлению (5.17).

5.6. Многокритериальный выбор в условиях неопределенности

В случае наличия неопределенности среды выбор вариантов, оцениваемых по нескольким критериям, вообще говоря, усложняется. Для моделирования этой новой ситуации существуют различные подходы. Логически простейший подход, по-видимому, состоит в использовании некоторой свертки векторного критерия оптимальности с последующим обращением к стандартным методам однокритериального выбора, рассмотренным в *главе 3*.

При относительно небольшом числе критериев можно обратиться к методу ранжирования критериев "по важности" на основе процедуры построения матрицы попарных сравнений *(см. разд. 5.3)*. Используются также методы теории "многокритериальной полезности", изложенной, например, в [17].

Рассмотрим содержательный пример. Пусть задана "матрица решений", представленная в табл. 5.2.

Таблица 5.2. *Многомерная матрица решений*

X	Z	
	z_1	z_2
x_1	$f_1^{11}, f_2^{11}, f_3^{11},$ $f_4^{11}, f_5^{11}, f_6^{11}$	$f_1^{12}, f_2^{12}, f_3^{12},$ $f_4^{12}, f_5^{12}, f_6^{12}$
x_2	$f_1^{21}, f_2^{21}, f_3^{21},$ $f_4^{21}, f_5^{21}, f_6^{21}$	$f_1^{22}, f_2^{22}, f_3^{22},$ $f_4^{22}, f_5^{22}, f_6^{22}$

Здесь f_i^{jk} — вещественные числа. Строго говоря, мы имеем трехмерную матрицу $2 \times 2 \times 6$, т. к. каждая реализация (x_i, z_j) оценивается по шести числовым критериям $f_1, f_2, f_3, f_4, f_5, f_6$.

На первом этапе решения задачи проранжируем критерии f_i по важности так, как это описано в *разд. 5.3*.

Для построения коэффициентов превосходства

$$\alpha_{12}, \alpha_{23}, \alpha_{34}, \alpha_{45}, \alpha_{56}$$

зададим пользователю (ЛПР) пять вопросов в соответствии с таблицей смысловой интерпретации уровней превосходства. Пусть пользователь дал следующие ответы:

1. Критерий f_1 слабо превосходит по важности критерий f_2: $\alpha_{12} = 2$.
2. Критерий f_2 сильно превосходит f_3: $\alpha_{23} = 4$.
3. Критерий f_4 сильно превосходит f_3: $\alpha_{34} = 1/4$.
4. Критерии f_4, f_5 равны по важности: $\alpha_{45} = 1$.
5. Критерий f_5 сильно превосходит f_6: $\alpha_{56} = 4$.

Используя соотношение

$$\alpha_{ij} = \alpha_i / \alpha_j,$$

и условие нормированности вектора

$$\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_m),$$

находим вектор весов относительной важности критериев:

$$\alpha_1 = 0{,}364;\ \alpha_2 = 0{,}182;\ \alpha_3 = 0{,}045;\ \alpha_4 = 0{,}182;\ \alpha_5 = 0{,}182;\ \alpha_6 = 0{,}045.$$

Предполагаем, что числа f_i^{jk} нам заданы (табл. 5.3).

Таблица 5.3. *Численные оценки альтернатив по шести критериям*

X	Z	
	z_1	z_2
x_1	0,25; 0,5; 0,3; 0,1; 0,7; 0,25	0,5; 0,14; 0,3; 0,7; 0,1; 0,25
x_2	0,25; 0,3; 0,3; 0,18; 0,18; 0,5	0,5; 0,14; 0,3; 0,5; 0,18; 0,25

На основе построенных в результате диалога с пользователем весовых коэффициентов α_i получаем уже обычную матрицу решений (табл. 5.4), содержащую значения "обобщенного" критерия оптимальности J:

$$J^{jk} = \sum_{i=1}^{6} \alpha_i f_i^{jk}, \quad j, k = 1, 2.$$

Таблица 5.4. Матрица решений для обобщенного критерия оптимальности

X	Z	
	z_1	z_2
x_1	J^{11}	J^{12}
x_2	J^{21}	J^{22}

Здесь

$$J^{11} = 0{,}364 \times 0{,}25 + 0{,}182 \times 0{,}5 + 0{,}045 \times 0{,}3 + 0{,}182 \times 0{,}1 + 0{,}182 \times 0{,}7 + \\ + 0{,}045 \times 0{,}25 = 0{,}35235;$$

$$J^{12} = 0{,}364 \times 0{,}5 + 0{,}182 \times 0{,}14 + 0{,}045 \times 0{,}3 + 0{,}182 \times 0{,}7 + 0{,}182 \times 0{,}1 + \\ + 0{,}045 \times 0{,}25 = 0{,}37783;$$

$$J^{21} = 0{,}364 \times 0{,}25 + 0{,}182 \times 0{,}3 + 0{,}045 \times 0{,}3 + 0{,}182 \times 0{,}18 + \\ + 0{,}182 \times 0{,}18 + 0{,}045 \times 0{,}5 = 0{,}24712;$$

$$J^{22} = 0{,}364 \times 0{,}5 + 0{,}182 \times 0{,}14 + 0{,}045 \times 0{,}3 + 0{,}182 \times 0{,}5 + \\ + 0{,}182 \times 0{,}18 + 0{,}045 \times 0{,}25 = 0{,}35599.$$

Предположим, что мы имеем задачу принятия решений в условиях риска и вероятности состояний среды заданы: $p(z_1) = 0{,}4$; $p(z_2) = 0{,}6$. Применим критерий Байеса—Лапласа, максимизируя ожидаемое значение обобщенного критерия:

$$\overline{J(x_1)} = 0{,}4 \times 0{,}35235 + 0{,}6 \times 0{,}37783 = 0{,}367638;$$

$$\overline{J(x_2)} = 0{,}4 \times 0{,}24712 + 0{,}6 \times 0{,}35599 = 0{,}312442.$$

В соответствии с критерием Байеса—Лапласа выбирается альтернатива x_1. Задача решена.

В данном примере предполагалось, что все частные критерии приведены к какой-то одной шкале измерений (являются однородными) и все они максимизируются.

5.7. Функции полезности

Основная идея методов многокритериального выбора вариантов в условиях определенности на основе теории полезности состоит в построении некоторого функционала $u(y)$, определенного на множестве оценок альтернатив и позволяющего формально свести многокритериальную задачу

к однокритериальной. Такой функционал называется *функцией полезности*, а проблема заключается в выборе варианта с максимальной полезностью. Теория полезности строится исходя из следующих предпосылок.

1. ЛПР в состоянии сравнивать между собой любые две многокритериальные оценки — так, как это было описано в *разд. 5.1*. Именно, предполагается, что любые две оценки

$$y', y'' \in Y,$$

где Y — множество оценок, связаны одним из соотношений:

- $y' \succ y''$ (y' предпочтительнее y'');
- $y'' \succ y'$ (y'' предпочтительнее y');
- $y' \sim y''$ (y' и y'' одинаковы по предпочтительности).

Используется также символ $\succsim$ — "не менее предпочтительно чем". Запись $y' \succsim y''$ читается как "y' не менее предпочтительно, чем y''". Введенные знаки предпочтения связаны между собой. Если $y' \succsim y''$ неверно, то имеем $y'' \succ y'$. Если одновременно выполняются условия $y' \succsim y''$ и $y'' \succsim y'$, то получаем $y' \sim y''$. Таким образом, можно полагать, что основным знаком является $\succsim$, через который определяются знаки строгого доминирования $\succ$ и эквивалентности $\sim$.

2. Система предпочтений ЛПР удовлетворяет некоторым достаточно очевидным с позиций "здравого смысла" требованиям (аксиомам).
3. Выполняются некоторые требования (аксиомы) "независимости" для частных критериев оптимальности.
4. В зависимости от выполнения тех или иных требований по независимости критериев констатируется наличие определенных структурных особенностей функции полезности $u(y)$ (например, она может быть аддитивной), что облегчает ее практическое построение.

Замечание 5.2

Если отказаться от предположения 1 и считать, что пользователь по-прежнему не в состоянии сравнивать между собой, например, несравнимые по Парето варианты, то тогда никаких новых задач не возникает. Мы в этом случае имеем основную задачу многокритериальной оптимизации — задачу построения множества Парето.

Проверка условий 2, 3, а также фактическое построение функции полезности с заданной структурой 4 осуществляются на основе соответствующего диалога с пользователем (ЛПР).

Перейдем к более точному изложению. Пусть решается многокритериальная задача вида

$$f_i(x) \to \max, f_i : X \to R, \quad i = \overline{1, m},$$

где X — исходное множество допустимых альтернатив. С помощью частных целевых функционалов f_i каждому элементу $x \in X$ ставится в соответствие векторная оценка $y \in R^m$, $y = f(x)$, где R — множество вещественных чисел. Заметим, что природа элементов множества X здесь не оговаривается и X может быть множеством объектов произвольной природы, например, это может быть множество векторов из соответствующего векторного пространства.

Пусть на множестве R^m задано бинарное отношение предпочтения $\succ$, а также операция определения центра тяжести y' для любых двух оценок y^1, y^2

$$y'_{12} = \alpha y^1 + (1-\alpha) y^2, \quad 0 < \alpha < 1.$$

Сформулируем аксиомы, определяющие основные свойства отношения предпочтения $\succ$.

Аксиома 5.1 (связности).

Для любых $y^1, y^2 \in R^m$ справедливо одно из соотношений:

$$y^1 \succ y^2, \; y^2 \succ y^1, \; y^1 \sim y^2.$$

Аксиома 5.2 (транзитивности).

$$(y^1 \succ y^2, \; y^2 \succ y^3) \to y^1 \succ y^3.$$

Аксиома 5.3 (растворимости).

$$y^2 \succ y^3 \succ y^1 \to \exists \alpha \in (0,1) : y'_{12} \sim y^3.$$

Аксиома 5.4 (архимедова).

$$y^1 \succ y^3 \succ y^2 \to \exists \alpha \in (0,1) : y^3 \succ y'_{21}.$$

Справедливо следующее утверждение, доказанное фон Нейманом и Моргенштерном.

Теорема 5.1. Если система предпочтений ЛПР удовлетворяет аксиомам 5.1—5.4, то существует скалярная функция (функционал) $u(y)$, которая ставит в соответствие каждому $y \in Y$ такое вещественное число $u(y)$, что

$$\begin{aligned} y^1 \succsim y^2 &\rightarrow u(y^1) \geq u(y^2); \\ y^1 \sim y^2 &\rightarrow u(y^1) = u(y^2). \end{aligned} \tag{5.19}$$

Теорема 5.1, по существу, дает ответ на вопрос: когда система предпочтений пользователя, заданная на языке бинарных отношений, может быть сведена к задаче однокритериальной скалярной оптимизации?

Замечание 5.3

Поскольку в детерминированном случае каждому $x \in X$ соответствует единственная оценка $y = f(x)$, можно полагать, что во множестве альтернатив с помощью функции $u(y)$ индуцируется соответствующая функция $v(x)$:

$$u(y) = u(f(x)) = v(x).$$

Иногда предлагается называть функцию полезности $u(y)$ (или $v(x)$) *функцией ценности*, если рассматривается выбор в условиях определенности, как в данном случае. Термин *функция полезности* сохраняется только для задач многокритериального выбора в условиях риска или неопределенности. Мы, однако, будем использовать первый термин, т. к. рассмотрение второго типа задач выходит за рамки настоящей книги.

Легко видеть, что функция полезности $u(y)$ (или соответствующая функция $v(x)$), удовлетворяющая соотношениям (5.19), определяется неоднозначно. Любое монотонное преобразование, например, вида

$$\ln u(y),\ a + bu(y),\ b > 0,$$

преобразует функцию полезности в другую функцию полезности, также удовлетворяющую соотношениям (5.19).

Перейдем к анализу условий независимости частных критериев. Как уже указывалось, выполнение таких требований по независимости позволяет гарантировать существование функций полезности с определенными желательными свойствами, например, свойством аддитивности. В свою очередь, наличие таких свойств у функции полезности существенно облегчает задачу ее фактического построения.

Пусть $f_1, f_2, ..., f_m$ — критериальные функции многокритериальной задачи

$$f_i(x) \to \max, f_i : X \to R, \quad i = \overline{1,m}.$$

Тогда каждому элементу $x \in X$ ставится в соответствие точка

$$(f_1(x), f_2(x), ..., f_m(x))$$

в m-мерном пространстве оценок.

Разобьем множество критериев $F = \{f_1, ..., f_m\}$ на два подмножества F^1, F^2:

$$F^1 \cup F^2 = F, \quad F^1 \cap F^2 = \varnothing.$$

Например, при $m = 6$ можем иметь:

$$F^1 = \{f_1, f_2, f_4, f_6\}, \; F^2 = \{f_3, f_5\}.$$

Тогда различные элементы $x \in X$ могут оцениваться как по группе критериев F^1, так и по группе критериев F^2. Не ограничивая общности, можно полагать, что в множество F^1 входят какие-то первые s критериев f_i, $i = 1, ..., s$, а во множество F^2 — оставшиеся критерии f_i, $i = s + 1, ..., m$. Это достигается простой перенумерацией частных критериев. В результате любая оценка $y = f(x)$ может быть представлена в виде $y = (w, z)$, где

$$w \in F^1(X); z \in F^2(X).$$

Здесь через $F^1(\cdot)$ обозначена вектор-функция $(f_1, ..., f_s)$, а через $F^2(\cdot)$ — вектор-функция $(f_{s+1}, ..., f_m)$.

Определение 5.3 [17]

Множество критериев F^1 не зависит по предпочтению от дополняющего его подмножества F^2 тогда и только тогда, когда структура условного предпочтения в пространстве оценок w при фиксированном z' не зависит от z'. Более точно: F^1 не зависит по предпочтению от F^2 в том и только в том случае, если

$$\forall w', w'' : [\exists z' : (w', z') \succsim (w'', z')] \to [\forall z : (w', z) \succsim (w'', z)].$$

Требование независимости по предпочтению часто оказывается весьма естественным с практической точки зрения и отражает реальные структурные особенности решаемой задачи. Действительно, если группа критериев $f_1, ..., f_s$ характеризует некоторые обособленные стороны элемен-

та x, то эти характеристики могут и не зависеть от значений оценок $f_{s+1}, \ldots, f_m$.

Если можно полагать, что множество критериев F^1 не зависит по предпочтению от F^2, то структура предпочтений ЛПР в пространстве оценок w может моделироваться при каком-то одном фиксированном z. В результате строится функция полезности $u_1(w)$, удовлетворяющая условию:

$$(w', z) \succsim (w'', z) \leftrightarrow u_1(w') \geq u_1(w'').$$

Из независимости F^1 от F^2 не следует независимость F^2 от F^1. Однако если F^1 не зависит по предпочтению от F^2 и, в свою очередь, F^2 не зависит по предпочтению от F^1, то общая функция полезности структурируется следующим образом:

$$u(y) = u(w, z) = \varphi(u_1(w), u_2(z)),$$

где φ — некоторый функционал, объединяющий функции u_1, u_2.

Таким образом, в данном случае ЛПР имеет возможность независимо структурировать и уточнять свои системы предпочтений, как по w, так и по z.

Определение 5.4 [17]

Критерии $f_1, \ldots, f_m$ взаимонезависимы по предпочтению, если каждое подмножество F^1 этого множества критериев не зависит по предпочтению от своего дополнения F^2.

Теорема 5.2. Для критериев $f_1, \ldots, f_m$, $m \geq 3$, аддитивная функция полезности

$$u(y) = u(y_1, y_2, \ldots, y_m) = \sum_{i=1}^{m} u_i(y_i),$$

(где u_i — функция полезности по критерию f_i) существует тогда и только тогда, когда критерии взаимонезависимы по предпочтению.

Формальное доказательство этой теоремы было получено Дебре (Debreu) в 1960 году.

Аддитивная функция полезности является, по-видимому, самой простой структуризацией общего выражения для функции полезности. Однако непосредственное практическое применение сформулированной теоремы затруднено очень большим числом условий независимости по предпочтению, которые необходимо проверять. Реальное количество критериев,

которые еще возможно за разумное время проанализировать на независимость, вряд ли превышает 5—7 [5].

Для работы с большим числом критериев в теории полезности получены результаты, позволяющие существенно сократить число проверок критериев на независимость по предпочтению.

Например, доказано, что если каждая пара критериев не зависит по предпочтению от своего дополнения, то критерии взаимонезависимы по предпочтению. Доказаны и другие утверждения, позволяющие сокращать число условий независимости по предпочтению, проверка которых необходима для облегчения построения функции полезности [17]. В частности, удобным оказывается понятие *частично аддитивной* функции полезности. Частичная аддитивность может выполняться даже если взаимонезависимость критериев по предпочтению не имеет места, но построение функции полезности при этом существенно облегчается.

При практическом построении функций полезности необходимо обращаться к смысловой интерпретации используемых частных критериев оптимальности. Достаточно часто уже из "физических" соображений можно констатировать некоторые условия независимости без их непосредственной проверки. Один из наиболее рациональных подходов связан с выделением *естественных групп* критериев, не связанных по смыслу между собой. Например [17], в задаче выбора места для строительства какого-то промышленного объекта каждый вариант его размещения может характеризоваться разнородными показателями, определяющими набор частных критериев. Такие показатели могут характеризовать: денежные расходы (бюджет строительства), воздействие на экосистему региона, социально-политические аспекты. По-видимому, с достаточной долей уверенности можно утверждать, что перечисленные показатели являются взаимно независимыми. А это позволяет надеяться на существование аддитивной функции полезности для этих трех групп критериев. Мы здесь подразумеваем, что находимся на некотором верхнем уровне иерархической системы критериев, описывающих решаемую проблему. Установив независимость представленных показателей на данном уровне, мы на следующем уровне конкретизируем наши представления об оптимальности, вводя соответствующие группы частных критериев.

Из приведенного рассмотрения следует, что, вообще говоря, речь здесь идет не только об автоматизированных (компьютерных) технологиях выбора лучших вариантов. В книге [17], в частности, рассмотрены различные схемы диалоговых процедур для уточнения структуры функции полезности с позиций независимости критериев и групп критериев по предпочтению. Более подробное изучение теории полезности выходит за рамки настоящей книги; дополнительные источники информации для читателя указаны в списке литературы.

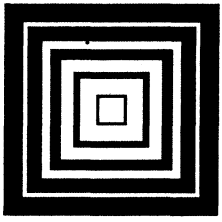
Несколько заключительных замечаний.

При практическом использовании теории полезности принятие решений имеет принципиально субъективный характер, т. к. источником дополнительной информации является ЛПР. Кроме того, как указывалось в *разд. 5.1*, при необходимости выбора в условиях многокритериальности единственного ("наилучшего") варианта или небольшого их числа мы должны предположить, что существует единственный "глобальный" критерий выбора, может быть, в явном виде и неизвестный самому ЛПР.

Как было показано ранее, для реализации процедур выбора на основе теории полезности необходимо построить функцию полезности по результатам диалога с пользователем. Этот диалог может оказаться весьма трудоемким, а число вопросов, требующих ответа ЛПР, — непомерно большим. Из общих соображений ясно, что в результате построения функции полезности получается явно избыточная информация. Действительно, в соответствии с этой функцией имеется возможность не только выбрать оптимальный вариант (или варианты), но и упорядочить все варианты по предпочтительности. Поэтому некоторые авторы полагают, что разумной альтернативой подходу, основанному на теории полезности, являются адаптивные процедуры, примеры которых были представлены в *разд. 5.1*.

Как уже говорилось, теория полезности при решении многокритериальных задач обобщается и на случай выбора в условиях риска и неопределенности. В данной книге этот материал не рассматривается.

Глава 6



Комментарий

В предыдущих главах изложены некоторые из важнейших проблем, входящих в теорию принятия решений. Основное содержание этой дисциплины схематично представляется в следующем виде.

1. Математическое описание — создание модели ситуации выбора.
2. Анализ неопределенностей, формализация понятия цели, формирование критериев и целевых функций.
3. Решение возникающих оптимизационных и других математических задач.

Приведенная последовательность действий достаточно условна, т. к. указанные разделы тесно переплетаются в процессе решения конкретной практической задачи. Однако основные моменты исследования здесь отражены.

Построение модели ситуации выбора, — а с него начинается любое исследование, — требует глубокого понимания специфики процесса. Мы более подробно рассмотрели два языка, на которых может быть сформулирована проблема ПР:

- язык бинарных отношений;
- критериальный язык описания выбора.

Язык бинарных отношений является более общим по сравнению с критериальным языком, поскольку не требует численной оценки качества каждой отдельно взятой альтернативы. Напротив, критериальный язык применяется, если сравнение альтернатив сводится к сравнению соответствующих им чисел. При этом мы допускали многокритериальность, т. е. возможность оценки альтернативы с помощью не одного числа, а нескольких. В теории полезности, в частности, устанавливаются условия, при выполнении которых система предпочтений, заданная на языке бинарных отношений, может быть представлена на критериальном языке.

Еще более общим языком *(см. разд. 1.4)* является язык функций выбора. Задачи, сформулированные на критериальном языке или языке бинарных отношений, описываются также с помощью функций выбора. Обратный вопрос о возможности введения бинарного отношения предпочтения по заданной функции выбора оказывается значительно более тонким.

Учитывая, что в настоящее время в приложениях наиболее часто применяется критериальный язык описания предпочтений, можно считать, что следующая важнейшая группа проблем — это формирование критериев и целевых функций (функционалов). Эти проблемы, как мы видели, решаются в тесной связи с методами преодоления различных видов неопределенностей на основе тех или иных гипотез.

Вот несколько общих, но важных для практики замечаний относительно применения оптимизационного подхода в теории ПР.

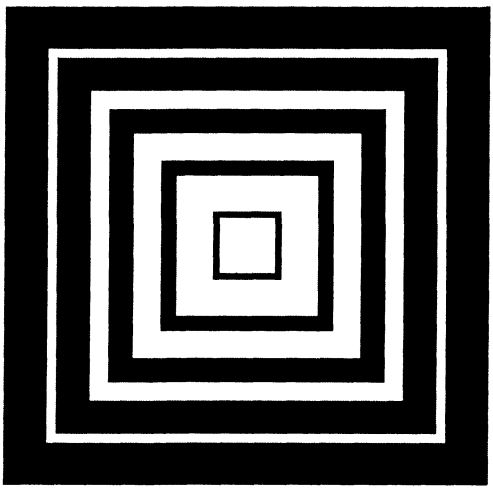
Критерий оптимальности — это, вообще говоря, некоторое правило, позволяющее отличать "оптимальные" решения от "неоптимальных". Простейший критерий связан, например, с заданием на множестве альтернатив некоторого функционала J: $A \to R$ и с утверждением, что *оптимальными* признаются такие альтернативы из множества A, которые доставляют максимум функционалу J на A.

Функционал J при этом называется *целевым функционалом* (или целевой функцией). Довольно часто допускается вольность речи и сам функционал J называется "критерием". При этом обычно говорят о задаче максимизации (или минимизации) "критерия J".

Важно также отличать цель принятия решений от отражающего эту цель критерия (или критериев). Приведем пример [17]. Общую цель службы скорой медицинской помощи можно сформулировать как "доставка больного в больницу в наилучшем состоянии, возможном при данных обстоятельствах". Эту цель достаточно трудно формализовать. Поэтому при анализе систем скорой помощи используется критерий "время реагирования" (в технических системах это "время отклика"). Этот критерий определяется как время, истекшее между получением вызова скорой помощи и прибытием машины к больному. Другой критерий, используемый при исследовании эффективности работы службы скорой помощи, — "время доставки" — время между получением вызова и прибытием больного в больницу. Смысл введения этих критериев состоит в предположении, что более краткие времена "реагирования" и "доставки" способствуют достижению общей цели системы скорой помощи, хотя полностью ее и не отражают (успех дела, в частности, зависит от квалификации персонала, его умения оказать эффективную немедленную помощь и т. д.).

Изложенная ситуация оказывается достаточно общей: критерии и отвечающие им целевые функции характеризуют цель лишь косвенно, иногда лучше, иногда хуже, но всегда приближенно. В этой связи обычно говорят о *критериях-заместителях*, т. е. таких критериях, которые лишь косвенно характеризуют степень достижения связанной с ними цели. В сущности, все критерии являются "заместителями", т. к. ни что не поддается абсолютно точному измерению.

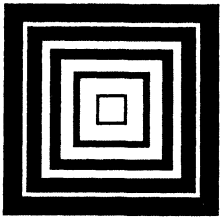
Все предыдущее изложение, по-видимому, в достаточной степени прояснило тот факт, что среди используемых в теории ПР математических методов особую роль играют методы решения оптимизационных задач. Они составляют фундамент системы математического обеспечения проблем принятия решений. Соответствующие вопросы, касающиеся проблемы конечномерной скалярной оптимизации, будут изложены во второй части книги.



ЧАСТЬ II

Алгоритмические методы скалярной оптимизации

Глава 7



Введение в проблему оптимизации

7.1. Постановка задачи оптимизации

Математически проблема оптимизации описывается следующим образом.

Рассматривается некоторое множество элементов (вообще говоря, произвольной природы) U, называемое *множеством допустимых элементов*. Пусть задана функция J, отображающая множество U во множество вещественных чисел (такие функции называются функционалами).

Задача 1. Требуется найти такой элемент u^* множества U, которому соответствует минимальное значение $J(u)$:

$$u^* \in U, \quad J(u^*) \le J(u), \quad \forall u \in U. \tag{7.1}$$

Если заменить J на $(-J)$, то задача минимизации трансформируется в задачу максимизации и обратно. Везде далее, если не оговорено противное, будем говорить о задачах минимизации. Элемент u^* (не обязательно единственный), удовлетворяющий соотношению (7.1), называется *точкой оптимума*, или *минимизатором* $J(u)$ на U. Часто используются следующие обозначения:

$$u^* = \arg\min_U J(u),$$

$$u^* \in \arg\min_{u \in U} J(u) = \left\{ u \in U \,\middle|\, u = \arg\min_U J(u) \right\}.$$

При формулировке задачи (7.1) предполагается ограниченность снизу функции J на U. Однако даже в этом случае задача (7.1) может не иметь решения, т. е. минимизаторы u^* могут отсутствовать среди элементов множества U. Например, функция $J(u) = e^u$, $U = R^1$ (вещественная прямая) ограничена снизу, но не достигает минимума ни в одной из точек на U.

Целесообразно поэтому рассматривать более общую задачу, которая для ограниченных снизу функций всегда имеет решение.

Задача 2. Требуется найти последовательность $\{u^k\}$ элементов (точек) множества U, удовлетворяющую предельному соотношению

$$\lim_{k\to\infty} J(u^k) = \inf_U J(u). \tag{7.2}$$

В обеих задачах функционал J называется целевым функционалом. Последовательности, для которых выполняется условие (7.2), называются *минимизирующими* для $J(u)$ на U.

Если существует минимизатор u^* для $J(u)$ на U и для минимизирующей последовательности $\{u^k\}$ справедливо соотношение

$$\lim_{k\to\infty} u^k = u^*, \tag{7.3}$$

то минимизирующая последовательность $\{u^k\}$ называется *сходящейся*.

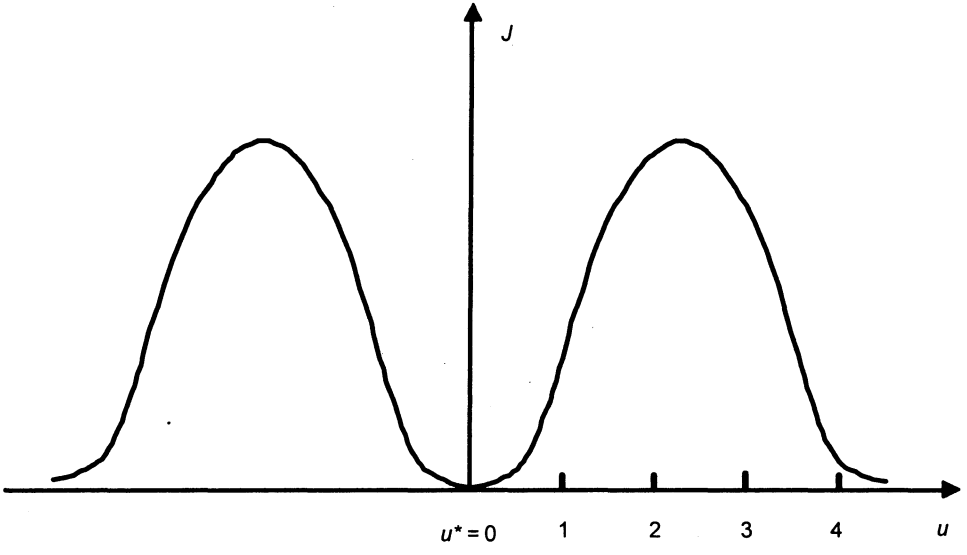


Рис. 7.1. Расходящаяся минимизирующая последовательность

На рис. 7.1 показан пример расходящейся минимизирующей последовательности $\{u^k\} = \{k\}$: $\lim_{k\to\infty} J(u_k) = J(u^*) = 0$. При выполнении последнего условия говорят, что имеет место сходимость по функционалу. Для схо-

дящихся минимизирующих последовательностей имеем еще и сходимость по аргументу.

Как правило, численные методы оптимизации позволяют строить минимизирующие последовательности и лишь в том случае, если они оказываются сходящимися в смысле (7.3), можно рассчитывать на получение достаточно хорошего приближения к минимизатору u^*.

При практических приложениях теории оптимизации принято различать "задачи аппроксимации" (аппроксимационные задачи оптимизации) и "задачи идентификации" (идентификационные задачи оптимизации). Приведем соответствующие примеры.

Пример 7.1 (задачи аппроксимации). Дана система обыкновенных дифференциальных уравнений

$$\frac{dx}{dt} = f(x, p, t), \tag{7.4}$$

где t — время; p — k-мерный вектор параметров. Предполагается, что система (7.4) адекватно описывает процессы в некоторой реальной системе. Задача заключается в поиске наилучшего по затратам памяти способа хранения информации о наборе из m экспериментальных кривых, характеризующих реальную систему

$$x^1(t), x^2(t), ..., x^m(t), \ t \in [0, T]. \tag{7.5}$$

Каждая кривая задана набором из N точек вида

$$x^i(t_j), \ j = 1, 2, ..., N; \ t_j \in [0, T].$$

Таким образом, всего требуется хранить $m \times l \times N$ чисел, где l — размерность вектора x (порядок системы (7.4)).

Предположим, что вектор p доставляет достаточно малое значение функционалу

$$J(p) = \sum_{j=1}^{N} \left\| x^i(t_j) - x^i(p, t_j) \right\|^2, \ i = 1, 2, ..., m, \tag{7.6}$$

где $x(p, t)$ — численное решение (7.3), полученное при текущем выборе p. Тогда для хранения заданной экспериментальной информации достаточно хранить m векторов p, что составит $m \times k$ вещественных чисел. При необходимости восстановления информации (7.5) в этом случае достаточно численно решить систему (7.4) при конкретном векторе p и соответствующих начальных условиях.

Если $m \times k << m \times l \times N$ или $k << l \times N$, то рассмотренный способ сжатия информации может оказаться оправданным. Совершенно ясно, что в данном случае нас не интересует сходимость по аргументу функционала $J(p)$. Важно лишь обеспечить достаточную точность аппроксимации, т. е. получение достаточно малых значений $J(p)$.

Многочисленные примеры оптимизационных задач, где требуется лишь сходимость по функционалу, дают не только задачи аппроксимации, но также теория и практика оптимального проектирования. В таких задачах целевой функционал J часто отражает качество проекта, а аргументом является некоторый вектор конструктивных параметров. При этом попрежнему часто оказывается неважным, с помощью какого вектора параметров (из допустимой области) удалось обеспечить заданное качество.

Пример 7.2 (задачи идентификации). Рассмотрим теперь несколько иную ситуацию. Пусть по-прежнему задана система (7.4), адекватно описывающая некоторую реальную систему, и набор экспериментальных кривых (7.5). Допустим теперь, что система (7.4) описывает некоторый химико-технологический процесс полимеризации, протекающий в лабораторных условиях, а вектор p является вектором скоростей элементарных реакций, принимающих при постоянной температуре конкретные, но неизвестные значения. Основная наша цель при решении задачи (7.6) минимизации $J(p)$ состоит в получении достаточно точных оценок этих скоростей, имеющих конкретный физический смысл. Далее полученный вектор p будет использован при моделировании и оптимизации в промышленных условиях режима управления каскадом реакторов по производству полимера (например, ударопрочного сополимера стирола с каучуком). Предполагается, что сам химико-технологический процесс производства полимера описывается другими математическими моделями, отличными от (7.4):

$$\frac{dz}{dt} = \varphi(z, p, q, ..., t).$$

Здесь p — найденный ранее (на этапе идентификации модели (7.4)) вектор оценок скоростей элементарных реакций; t — время; z — вектор фазовых переменных процесса; q — вектор управляющих (режимных) параметров, определяемый как решение некоторой дополнительной задачи оптимизации вида

$$F(q) = \sum_{k=1}^{s} \left\| z(t_k, q) - z(t_k) \right\|^2 \to \min_q, \tag{7.7}$$

где $z(t_k)$ — желаемые значения вектора фазовых переменных в заданный момент времени; $z(t_k, q)$ — соответствующие расчетные значения.

Ясно, что если при решении задачи (7.6) нас интересует сходимость по аргументу, то в случае (7.7) мы должны обеспечить лишь достаточно быструю сходимость по функционалу, как в примере 7.1.

7.2. Терминологические замечания. Классификация задач

В данном разделе мы рассмотрим некоторые частные случаи сформулированной в *разд. 7.1* общей задачи поиска минимизатора в абстрактном множестве U (задача 1).

Делая разумные предположения о свойствах функционала J и множества U (например, выпуклость), можно получать полезные теоретические результаты даже в такой общей постановке вопроса, как задача 1 из *разд. 7.1*. Это один уровень изучения проблемы оптимизации. Можно наделять указанные объекты J, U дополнительными свойствами, используя все более конкретные (частные) математические структуры. В результате мы приходим к целой иерархии теорий, изучающих проблему с различной степенью подробности. Если обратиться к множеству U, то можно наделить его, скажем, структурой банахова или даже гильбертова пространства и строить соответствующую теорию минимизации функционалов в банаховых или гильбертовых пространствах. В данной книге мы пойдем еще дальше и будем считать множество U некоторым подмножеством D конечномерного евклидова пространства R^n над множеством вещественных чисел. Такого типа *конечномерные задачи оптимизации* называются *задачами математического программирования (МП)*. Итак, МП-задачи — это задачи поиска минимума вещественной функции (функционала) от n вещественных переменных, определенной в некотором допустимом множестве $D \subset R^n$:

$$J : D \to R^1$$
$$J(x) \to \min_{x \in D}, x = (x_1, x_2, ..., x_n). \tag{7.8}$$

В данном случае термин "программирование" является неудачным переводом соответствующего английского аналога. Более точный перевод означал бы "планирование" или что-то подобное. Однако терминология в русскоязычной литературе устоялась и нам придется применять общепринятые названия.

Любая терминология вводится для каких-то целей. В данном случае с прикладной точки зрения МП-задачи позволяют оставить в стороне так

называемые бесконечномерные задачи оптимизации, изучаемые в основном в рамках теории оптимального управления.

Простейшая задача оптимального управления может быть сформулирована, например, следующим образом.

Управляемый процесс описывается системой обыкновенных дифференциальных уравнений:

$$\frac{dx}{dt} = f(x, u), \quad x(t_0) = x^0, \tag{7.9}$$

где $x(t)$ — n-мерный вектор состояний системы; x^0 — заданное начальное состояние; $u(t)$ — r-мерный вектор управлений: $u(t) = [u_1(t), ..., u_r(t)]$, где $u_i(t)$ — функции времени, принадлежащие некоторому заданному множеству функций Ф.

Допустим, что существуют управления $u \in \Phi$, переводящие управляемый процесс из заданного начального состояния x^0 в предписанное конечное состояние $x(t_1) = x^1$ (момент времени t_1 не фиксируется). Такие управления будут составлять некоторое множество $D \subset \Phi$ допустимых управлений.

Требуется отыскать такое управление $u \in D$, чтобы минимизировать функционал

$$J(u) = \int_{t_0}^{t_1} F[x(t), u(t), t]dt \to \min_{u \in D}, \tag{7.10}$$

где F — заданная функция своих аргументов.

В этом случае система (7.9) играет роль ограничений на управления $u(t)$ и фазовые переменные $x(t)$. Иногда здесь вводятся еще дополнительные ограничения на допустимые функции $u(t)$.

Сформулированная задача минимизации, очевидно, не является конечномерной, т. к. функции $u_i(t)$ не определяются, вообще говоря, конечным набором вещественных чисел. Для решения бесконечномерных задач развиты специальные методы и принципы исследования (такие как принцип оптимальности Беллмана, принцип максимума Понтрягина), однако в некоторых случаях удается свести бесконечномерную задачу к конечномерной. В качестве примера рассмотрим принцип параметризации. Он состоит в том, что оптимальные управления $u(t)$ ищутся в классе функций, представленных в виде

$$u(t) = \sum_{i=1}^{N} \alpha_i \varphi_i(t),$$

где $\{\varphi_1, \varphi_2, ..., \varphi_N\}$ — заданная система базисных функций.

В этом случае функционал (7.10) допускает представление

$$J(u) = J(\sum_{i=1}^{N} \alpha_i \varphi_i(t)) = J_1(\alpha),$$

т. е. задача минимизации $J(u)$ заменяется конечномерной задачей минимизации $J_1(\alpha)$ в N-мерном пространстве весовых коэффициентов $\alpha = (\alpha_1, ..., \alpha_N)$.

При реализации такого подхода основные трудности заключаются в выборе базисной системы функций, наиболее соответствующей решаемой задаче. В частности, при этом возникает проблема уменьшения размерности вектора α.

Далее мы будем рассматривать только МП-задачи, т. е. конечномерные задачи оптимизации.

Принято различать следующие виды задач математического программирования.

Нелинейное программирование. В задачах нелинейного программирования (НП) предполагается, что множество допустимых значений аргументов D минимизируемой функции (функционала) J задается с помощью систем равенств и неравенств. Таким образом, имеем следующее общее представление для НП-задач:

$$\begin{gathered} J(x) \to \min_{x \in D}; \\ D = \left\{ x \in R^n \mid g_i(x) \le 0,\ i = 1, ..., l;\ h_j(x) = 0,\ j = 1, ..., s \right\}, \end{gathered} \tag{7.11}$$

где J, g_i, h_j — функционалы, реализующие отображения $R^n \to R$.

Линейное программирование. Если целевой функционал и функционалы ограничений могут быть заданы с помощью линейных функций, то такие задачи НП называются *задачами линейного программирования (ЛП)*.

Так называемая *основная задача ЛП (ОЗЛП)* ставится следующим образом.

Требуется найти минимизатор линейной функции

$$J(x) = c_1 x_1 + c_2 x_2 + \cdots + c_n x_n = \sum_{i=1}^{n} c_i x_i = \langle c, x \rangle$$

в допустимом множестве D, элементы которого удовлетворяют ограничениям:

$Ax = b$ (ограничения-равенства); $x_i \ge 0$, $i = 1, ..., n$ (ограничения-неравенства), где A — $(m \times n)$-матрица вещественных чисел; b — m-вектор; $\langle \cdot, \cdot \rangle$ — знак скалярного произведения.

Кратко ОЗЛП может быть записана в виде $\min\{cx \mid x \geq 0, Ax = b\}$, где cx — краткая запись скалярного произведения вектора c на вектор x.

Существует несколько эквивалентных форм ЛП-задач. Например, все следующие виды ЛП-задач эквивалентны в смысле сводимости друг к другу:

- $\min\{cx \mid x \geq 0,\ Ax = b\}$ (ОЗЛП);
- $\min\{cx \mid x \geq 0,\ Ax \geq b\}$;
- $\min\{cx \mid Ax \geq b\}$;
- $\max\{cx \mid x \geq 0, Ax = b\}$;
- $\max\{cx \mid x \geq 0, Ax \leq b\}$;
- $\max\{cx \mid Ax \leq b\}$.

Таким образом, любой метод решения одной из представленных задач может быть преобразован для решения всех остальных. (Существуют и другие эквивалентные формулировки, основанные, в частности, на теореме двойственности.)

ЛП-задачи образуют важный класс НП-задач по крайней мере в двух отношениях. Во-первых, такие задачи характерны для многих практических (особенно экономических) приложений. Во-вторых, для решения ЛП-задач созданы специальные и достаточно эффективные методы, позволяющие в большом числе случаев получать решение за конечное число шагов.

Кроме того, существуют вычислительные технологии решения общих НП-задач на основе их последовательной аппроксимации соответствующими ЛП-задачами.

Выпуклое программирование. Задачи выпуклого программирования, как и ЛП-задачи, оказываются достаточно удобными для точного математического анализа основных ситуаций. Так называются задачи, в которых целевой функционал и допустимое множество D являются выпуклыми. Существуют свои специфические для данного класса задач методы минимизации, например, методы возможных направлений, линеаризации, двойственные и др. Мы не будем далее рассматривать эти методы, отметим только, что уже такая простая задача, как задача оценки скалярного параметра a скалярного дифференциального уравнения

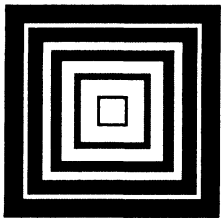
$$\frac{dx}{dt} = ax, \quad x(0) = x_0$$

по известному экспериментальному значению $x(t_1) = x_z$ из условия минимума функции $J(a) = (x(t_1, a) - x_z)^2 \to \min\limits_a$ оказывается невыпуклой. Здесь $x(t, a)$ — решение дифференциального уравнения при заданном произвольном значении параметра a.

Существует еще несколько "программирований", например, "квадратичное", "недифференцируемое" и т. д. Мы не будем здесь приводить соответствующие формулировки, имея в виду представленный выше общий случай НП-задач.

В заключение отметим важный класс задач нелинейного программирования, допускающих весьма специфические и принципиально важные методы решения. Это еще одно "программирование" — динамическое. Методы динамического программирования ориентированы на НП-задачи, в которых целевой функционал имеет специальную сепарабельную структуру, что позволяет трактовать задачу построения минимизатора как некоторую многоэтапную процедуру, в которой результирующее значение целевого функционала является суммой его значений на отдельных этапах. На методах динамического программирования основаны, в частности, многие сетевые методы, когда, например, требуется отыскать минимальный путь на графе и т. п. Некоторые аспекты применения метода динамического программирования в многошаговых задачах принятия решений были нами рассмотрены в *главе 4*. Общая теория динамического программирования имеет гораздо более широкий спектр приложений.

Глава 8



Основные математические модели оптимизации

8.1. Общая проблема оптимизации произвольной системы

Математические модели оптимизируемых систем будем характеризовать конечной совокупностью числовых параметров, которые условно можно разделить на три группы:

- внутренние;
- внешние;
- выходные.

Под *внутренними* параметрами понимаются параметры отдельных элементов, составляющих оптимизируемую систему. Так, при оптимальном проектировании электронной схемы в дискретном исполнении внутренними параметрами являются электрические параметры типа сопротивлений, емкостей, индуктивностей, токов и напряжений источников либо определяющие их другие величины. При расчете интегральных микросхем внутренними параметрами являются не только электрические, но также геометрические и физико-структурные параметры.

Внешние параметры характеризуют влияние внешней среды на оптимизируемый объект. Примерами внешних параметров могут служить параметры входных сигналов, температура окружающей среды, случайные факторы, определяющие шумовое воздействие среды на объекты, и т. д.

Важнейшее значение при описании объектов имеют *выходные* параметры, отражающие основные свойства и характеристики оптимизируемой системы. В качестве примера выходных параметров некоторой технической системы можно указать потребляемую мощность, быстродействие,

габариты, стоимость, оценки точности аппроксимации заданных характеристик, например амплитудно-частотных, и т. д.

Обозначим через $v = (v_1, v_2, ..., v_k)$, $w = (w_1, w_2, ..., w_l)$ и $y = (y_1, y_2, ..., y_m)$ векторы внутренних, внешних и выходных параметров соответственно. Компоненты векторов v, w являются независимыми переменными, определяющими значения зависимых выходных параметров. Таким образом, существует функциональная связь

$$y = \varphi(v, w), \tag{8.1}$$

описываемая некоторым набором алгоритмов, составляющих математическое описание, или математическую модель объекта оптимизации. Только в простейших случаях функция (8.1) может быть задана с помощью системы явных выражений. В большинстве же случаев связь y с v и w оказывается алгоритмической, что исключает применение аналитических методов исследования.

Не все внутренние параметры являются равноправными — обычно только часть из них может варьироваться в процессе оптимизации. Изменяемые внутренние параметры называются *управляемыми параметрами*, или *параметрами оптимизации*, и образуют вектор $x = (x_1, x_2, ..., x_n)$, являющийся подвектором вектора v.

При отсутствии факторов неопределенности все внешние и неизменяемые внутренние параметры принимают известные, заранее заданные значения. В результате от функции (8.1) приходим к основной для целей алгоритмической оптимизации зависимости

$$y = F(x), \tag{8.2}$$

осуществляющей связь между вектором параметров оптимизации и основными выходными характеристиками объекта. Выражения, реализующие зависимость (8.2), называются *детерминированной моделью объекта оптимизации*. Весьма часто, однако, оптимизация осуществляется при наличии различных типов неопределенностей. Довольно распространена в задачах оптимизации неопределенность ("неопределенность обстановки"), приводящая к усложнению зависимости (8.2) в результате введения в математическую модель некоторого, вообще говоря, неизвестного, вектора $z = (z_1, ..., z_s)$:

$$y = F(x, z). \tag{8.3}$$

При этом отдельные компоненты вектора z могут быть как случайными, так и неслучайными величинами.

Например, составляющие вектора z_i могут описывать влияние:

- случайных отклонений в технологическом процессе при массовом производстве каких-либо технических устройств (технологический разброс параметров);
- изменяющихся, как правило, неслучайным образом условий функционирования оптимизируемой системы, таких как температура, влажность, вибрация, уровень радиации (при оптимизации технических систем) или уровень спроса на продукцию, параметры фондового рынка (при оптимизации финансовых и экономических систем);
- старения или износа оборудования

и т. д.

Все перечисленные причины приводят к тому, что в действительности будем иметь систему, параметры и характеристики которой отличаются от расчетных. Методы учета неопределенности обстановки, применяемые в задачах оптимизации, призваны, с одной стороны, определить степень влияния z на основные характеристики системы, а с другой стороны, по возможности ослабить это влияние.

При случайном характере компонент вектора z модель (8.3) называется *вероятностной (стохастической)*, или *моделью оптимизации в условиях риска.*

Основные принципы однокритериальной оптимизации в условиях неопределенности обстановки были, по существу, изложены ранее в *главе 3* в контексте общей задачи принятия решений.

Непосредственно на основе математического описания оптимизируемой системы решается важнейшая задача оптимизации — задача анализа. Задача анализа заключается в вычислении вектора выходных параметров по заданным значениям всех остальных переменных при фиксированной функционально-структурной модели системы. Методы решения задач анализа определяются природой оператора F, задающего математическую модель объекта оптимизации (функцию реализации — в терминологии *главы 3*). Как уже указывалось, оператор F может быть задан в неявной форме и для его реализации необходимо решить одну или несколько стандартных для данного уровня моделирования задач численного анализа. Наиболее часто возникают задачи, связанные с необходимостью решения систем алгебраических и дифференциальных, а также смешанных уравнений. Соответствующие вопросы весьма полно представлены в литературе по численному анализу и теории математического моделирования.

Важная область применения теории и методов оптимизации связана с задачами оптимального проектирования объектов и систем. Под *проектированием* понимается процесс создания математического описания еще

не существующего объекта с целью его последующего изготовления или последующей реализации (воплощения). Используя ранее введенные обозначения, можно сказать, что задача оптимального проектирования по существу распадается на два этапа. На этапе *структурного синтеза* происходит формирование структуры, определяющей элементный (компонентный) состав будущей системы и связи между элементами. Иначе говоря, на этапе структурного синтеза конкретизируется вид функции реализации F в выражении (8.3). Задача выбора структуры в настоящее время решается, как правило, неформальными методами. И хотя в отдельных областях применения могут быть развиты регулярные методы структурного синтеза, в общем виде проблема, по-видимому, неразрешима. Обычно при проектировании систем исходные структуры формируются на основе имеющегося банка типовых структурных решений, обобщающих предшествующий опыт разработок в соответствующей области. Некоторые обнадеживающие результаты по формализации процесса структурного синтеза устройств с существенно неоднородным базисом проектирования получены при использовании "избыточных структур", для которых ищется частное структурное решение с помощью удаления "лишних" элементов.

Большой опыт проектирования, накопленный в различных областях деятельности, обычно позволяет существенно сузить множество (класс) возможных структурных решений для каждой реальной задачи и поэтому основные трудности возникают на этапе *параметрического синтеза.* Задача параметрического синтеза состоит в выборе такого вектора x параметров оптимизации (здесь — параметров проектирования), при которых все выходные характеристики проектируемой системы удовлетворяют требованиям технического задания (списку спецификаций). Эти требования обычно содержат ограничения на входные и выходные параметры; основные типы ограничений рассмотрены ниже.

Как правило, речь идет об оптимальном параметрическом синтезе, т. к. вектор x выбирается не только из условий правильности функционирования системы, но и из условий обеспечения оптимальности по принятым критериям качества. Задача оптимального параметрического синтеза часто называется *задачей параметрической оптимизации*, или *задачей оптимального проектирования.*

Задача оптимального параметрического синтеза в конечном счете основана на последовательном решении большого числа задач анализа при различных пробных значениях параметров элементов проектируемой системы. Число необходимых анализов в среднем оценивается как $(100 \div 200) \times n$, где n — число варьируемых параметров. Таким образом, уже при $n = 10$ необходимо выполнить анализ примерно 1000 вариантов проектируемой системы. Это вынуждает выдвигать достаточно жесткие ог-

раничения как на трудоемкость методов анализа, так и на размерность n вектора x. Существенные вычислительные трудности возникают, в частности, если алгоритм анализа содержит процедуру решения одной или нескольких жестких систем дифференциальных уравнений.

Решение задач конечномерной оптимизации вообще и оптимального проектирования в частности (в зависимости от интерпретации) производится с учетом имеющихся ограничений.

Наиболее простой вид имеют так называемые *прямые* ограничения на компоненты вектора управляемых параметров:

$$a_i \le x_i \le b_i, \tag{8.4}$$

где $[a_i, b_i]$ — заданный допустимый интервал изменения параметра x_i. В более общем случае границы интервалов могут быть функциями от других управляемых параметров, например

$$a_i(x_j) \le x_i \le b_i(x_i) \quad (i \ne j). \tag{8.5}$$

Такие ограничения тоже будем относить к классу прямых.

Ограничения (8.4), (8.5) часто бывают вызваны причинами, связанными с условиями физической и практической реализуемости, например, с требованиями неотрицательности емкостей, сопротивлений, индуктивностей при проектировании электронных схем или геометрических параметров. Прямые ограничения вытекают также из технологических возможностей производства, определяющих предельно допустимые значения управляемых параметров.

На выходные параметры накладывается два типа ограничений. *Функциональные* ограничения включают в себя условия работоспособности, имеющие принципиальное значение при оценке правильности функционирования оптимизируемой системы (реальной или проектируемой) исходя из целей оптимизации и выполнения системой своего функционального назначения. Эти ограничения обычно задаются в виде системы равенств и неравенств

$$y_i \le t_i;\ y_j \ge t_j;\ y_k = t_k, \tag{8.6}$$

где t_i, t_j, t_k — заданные числовые параметры. Если в качестве примера снова обратиться к задачам проектирования, то к функциональным ограничениям могут относиться следующие требования: рассеиваемая в элементах проектируемой электронной схемы мощность должна быть меньше предписанного порогового значения; полюсы передаточной функции фильтра должны лежать в левой полуплоскости; коэффициент обратной связи в схеме генератора должен быть больше критического значения; порог срабатывания ждущей релаксационной схемы должен находиться в заданных пределах и т. д.

Вторая группа ограничений накладывается на выходные параметры, имеющие смысл частных критериев оптимальности и характеризующие качество объекта оптимизации. Наличие подобных частных критериев, по существу, отражает ту неопределенность целей, которая присутствует при оптимизации любой сколько-нибудь сложной системы. Каждый из критериальных выходных параметров желательно максимизировать или минимизировать:

$$y_i \to \max_x; \quad y_j \to \min_x. \tag{8.7}$$

Однако в процессе оптимизации требования (8.7) могут быть изменены или дополнены с помощью следующих соотношений, называемых *критериальными ограничениями*:

$$y_i \geq t_i; \; y_j \leq t_j. \tag{8.8}$$

Примерами критериальных ограничений при оптимизации технических систем являются ограничения на стоимость изделия, помехоустойчивость, быстродействие, точность аппроксимации характеристик, нагрузочную способность, степень устойчивости, время срабатывания и т. д.

Соотношения (8.7), (8.8) не исключают друг друга. Напротив, как правило, оптимизационные задачи (8.7) решаются с учетом ограничений (8.8), отражающих требования к характеристикам качества, подлежащие безоговорочному выполнению.

Критериальные ограничения принципиально отличаются от функциональных. Выполнение критериальных ограничений отражает стремление получить оптимальный или близкий к оптимальному вариант системы среди систем, заведомо удовлетворяющих функциональным ограничениям, т. е. функционирующих правильно, в соответствии с предъявляемыми требованиями. В этом смысле можно сказать, что критериальные ограничения по своей сути являются менее жесткими, чем функциональные. Важно, однако, понимать, что грань между функциональными и критериальными ограничениями может быть весьма условной и зависеть от конкретных условий оптимизации.

На критериальные выходные параметры могут одновременно накладываться и функциональные ограничения. Например, в задаче максимизации запаса по устойчивости некоторой системы весьма естественным оказывается требование его неотрицательности.

Список прямых, функциональных и критериальных ограничений составляет основную часть требований при оптимизации системы. Кроме этого, указываются условия работы системы: характеристики возможных помех и факторов неопределенности среды, диапазоны температуры, давлений, влажности, диапазоны изменения параметров рынка (при оп-

тимизации экономических систем) и т. д. Эти условия формулируются как ограничения на допустимые диапазоны изменения компонент вектора внешних параметров.

Рассмотрим основные методы формальной постановки задачи оптимизации при различных предположениях об условиях и целях оптимизации.

Достаточно общая детерминированная задача оптимизации формулируется следующим образом:

$$f_i(x) \to \min_x;\ x \in D,\ i = 1, ..., k; \tag{8.9}$$

$$D = \left\{ x \in R^n \mid g_i(x) \le 0, i = \overline{1, m};\ g_i(x) = 0, i = \overline{m+1, s};\ a_j \le x_j \le b_j,\ j = \overline{1, q} \right\}.$$

Множество D называется *множеством допустимых значений.* В пределах этого множества выполняются прямые, функциональные и критериальные ограничения, представленные в виде общей системы неравенств и равенств. Прямые ограничения удобно указывать в явном виде, т. к. они учитываются обычно отдельно от других типов ограничений, имеющих более сложную структуру.

Задача (8.9) может рассматриваться как формальное представление основных требований к оптимизируемой системе. Предполагается, что каждый из критериальных выходных параметров f_i необходимо минимизировать. Это не ограничивает общности, т. к. максимизация функции $\varphi(x)$ эквивалентна минимизации $-\varphi(x)$. Кроме этого, нужно учитывать, что замена знака у левых частей неравенств $p(x) \ge 0$ меняет знаки неравенств на противоположные и приводит их к стандартному виду $g(x) \le 0$, где $g(x) = -p(x)$.

Задача (8.9) не является стандартной для базовых методов численного анализа из-за наличия векторного критерия оптимальности. Поэтому приобретают важное значение различные приемы ее сведения к однокритериальным задачам, допускающим эффективное численное решение обычными средствами. Такое сведение не является однозначным и обычно вызывает известные трудности. Вид и форма окончательной постановки задачи во многом определяются конкретными целями оптимизации, а также имеющимися в распоряжении алгоритмическими и программными средствами.

Существующие методы и технологии решения многокритериальных задач оптимизации были, по существу, рассмотрены в *главе 2.*

8.2. Методы преобразования и учета ограничений

Рассмотрим методы исключения и преобразования ограничений в общей задаче (8.9).

Наиболее просто снимаются прямые ограничения $a_i \le x_i \le b_i$ или $a_i(x_j) \le x_i \le b_i(x_j)$, $i \ne j$. Для этого достаточно заменить переменные по одной из формул, указанных в табл. 8.1, где z_i означают новые независимые переменные.

Таблица 8.1. *Метод замены переменных*

Ограничение	Преобразование
$x_i > a_i$	$x_i = a_i + \exp(z_i)$
$x_i \ge a_i$	$x_i = a_i + z_i^2$
$x_i \ge x_j,\ i \ne j$	$x_j = z_j,\ x_i = z_j + z_i^2$
$a_i \le x_i \le b_i$	$x_i = b_i + (a_i - b_i)\sin^2 z_i$ $x_i = 0{,}5(a_i + b_i) + 0{,}5(b_i - a_i)\sin z_i$
$a_i < x_i < b_i$	$x_i = b_i + (a_i - b_i)\frac{1}{\pi}\operatorname{arcctg} z_i$ $x_i = b_i + (a_i - b_i)\exp(z_i) / [1 + \exp(z_i)]$
$a \le x_i \le b$ $a \le x_j \le b$ $x_i \ge x_j,\ i \ne j$	$x_j = b + (a - b)\sin^2 z_j$ $x_i = b + (a - b)\sin^2 z_j \sin^2 z_i$
$a < x_i < b$ $a < x_j < b$ $x_i > x_j,\ i \ne j$	$x_j = b + (a - b)\frac{1}{\pi}\operatorname{arcctg} z_j$ $x_i = b + (a - b)\frac{1}{\pi^2}\operatorname{arcctg} z_i \operatorname{arcctg} z_j$
$a_i(x_j) \le x_i \le b_j(x_j)$ $i \ne j$	$x_j = z_j$ $x_i = b_i(z_j) + [a_i(z_j) - b_i(z_j)]\sin^2 z_i$
$a_i(x_k) \le x_i \le b_i(x_j)$ $i \ne j,\ i \ne k$	$x_j = z_j$ $x_k = z_k$ $x_i = b_i(z_j) + [a_i(z_k) - b_i(z_j)]\sin^2 z_i$

Далее будем предполагать, что прямые ограничения отсутствуют.

Существуют два класса методов оптимизации, ориентированных на решение однокритериальных задач с ограничениями. Первый класс составляют алгоритмы, реализующие методы проекции градиента, отсечения, а также различные варианты метода возможных направлений. Эти алгоритмы дают возможность на каждой итерации свести исходную задачу к формально более простой задаче с ограничениями, например к задаче линейного программирования.

Во вторую группу входят методы штрафных функций и модифицированных функций Лагранжа, основанные на учете ограничений непосредственно в конструкции критерия оптимальности с последующим использованием алгоритмов безусловной оптимизации. В сложных практических задачах оптимизации чаще применяется второй подход.

Основная идея метода штрафных функций состоит в следующем. Рассмотрим задачу нелинейной оптимизации вида

$$J(x) \to \min_{x \in D};\ D = \left\{ x \in R^n \mid h_i(x) = 0,\ i = \overline{1, q} \right\}, \tag{8.10}$$

не содержащую ограничений в виде неравенств. Тогда вместо (8.10) решается последовательность задач безусловной минимизации однопараметрического семейства функционалов $\{J_k\}$, где

$$J_k(x) = J(x) + \sigma_k \sum_{i=1}^{q} h_i^2(x),\ \sigma_k \to \infty,\ k \to \infty. \tag{8.11}$$

Второе слагаемое в (8.11) имеет смысл "штрафа" за нарушение ограничений, что и определяет название метода. Справедлива следующая теорема.

Теорема 8.1. Пусть задача (8.10) имеет единственное решение x^*; функции J, h_i непрерывны в R^n; для любого $k = 1,2, \ldots$ существует $x^k = \arg\min J_k(x) \in D \subset R^n$, где D — ограниченное замкнутое множество; $\sigma_k \to \infty$, $k \to \infty$. Тогда $\lim x^k = x^*$, $k \to \infty$.

Доказательства различных утверждений, аналогичных сформулированной теореме, содержатся во многих работах по математическому программированию.

Наиболее распространенный вариант метода штрафных функций для решения общей задачи математического программирования

$$J(x) \to \min;\ \ x \in D; \tag{8.12}$$

$$D = \left\{ x \in R^n \mid g_i(x) \le 0,\ i = \overline{1, m};\ g_i(x) = 0,\ i = \overline{m+1, s} \right\}$$

состоит в применении вспомогательных функционалов

$$J_k(x) = J(x) + \sigma_k \sum_{i=1}^{s} (g_i^+(x))^p, \tag{8.13}$$

где

$$g_i^+ = \begin{cases} \max\{g_i(x); 0\} & (i = \overline{1, m}); \\ |g_i(x)| & (i = \overline{m+1, s}). \end{cases}$$

Если функционалы J, g_i являются r раз непрерывно дифференцируемыми на некотором множестве D, то при любом $p > r$ этим же свойством будут обладать функционалы $J_k(x)$.

Основной недостаток метода штрафных функций заключается в ухудшении обусловленности вспомогательных задач при больших σ_k. Соответствующие вопросы рассмотрены далее.

Наиболее перспективным общим методом учета ограничений считается метод модифицированных функций Лагранжа. Применительно к задаче (8.10) он формулируется следующим образом.

Введем в качестве обобщенного критерия оптимальности функционал

$$M(x, \xi, \sigma) = J(x) + (\xi, h(x)) + 0{,}5\sigma \|h(x)\|^2, \tag{8.14}$$

где $\sigma > 0$ — параметр метода. Тогда алгоритм оптимизации сводится к итерационному процессу

$$x^{k+1} = \arg\min_x M(x, \xi^k, \sigma); \tag{8.15}$$

$$\xi^{k+1} = \xi^k + \sigma h(x^{k+1}); \quad h(x) = [h_1(x), \ldots, h_q(x)],$$

обобщающему методы штрафных функций и множителей Лагранжа. Основная особенность сформулированного алгоритма по сравнению с методом штрафных функций заключается в отсутствии неограниченно растущего штрафного коэффициента σ, который в данном случае влияет лишь на скорость сходимости, но не на сам факт сходимости последовательности $\{x^k\}$ к оптимуму x^*. При решении практических задач значение σ целесообразно подбирать в интерактивном режиме, т. к. надежные методы априорного задания σ в настоящее время отсутствуют.

Теоремы о сходимости методов модифицированных функций Лагранжа, а также вычислительные схемы решения общей задачи (8.12) мы не будем здесь рассматривать (см. список литературы).

Иногда при оптимизации возникает необходимость преобразования ограничений-равенств в неравенства и обратно. Неравенства могут быть сведены к равенствам в результате расширения списка управляемых па-

раметров. Например, ограничение $g(x) \le t$ эквивалентно двум ограничениям $h(\bar{x}) = t$, $x_{n+1} \ge 0$, где $h(\bar{x}) = g(x) + x_{n+1}$, $\bar{x} = (x_1, ..., x_{n+1})$. Дополнительное прямое ограничение $x_{n+1} \ge 0$ устраняется заменой переменных.

Ограничение-равенство $h(x) = 0$ эквивалентно двум неравенствам $g_1(x) = h(x) \le 0$, $g_2(x) = -h(x) \le 0$.

8.3. Оптимизация систем в условиях неопределенности

Приводимые далее сведения в известном смысле дополняют результаты из *главы 3*. Здесь больше внимания уделено алгоритмическим аспектам при решении общих "бесконечномерных" задач (множества изменения аргументов функции реализации могут быть бесконечными).

Рассмотрим математическую модель объекта оптимизации, заданную в виде функции реализации

$$y = F(x, z), \tag{8.16}$$

включающей неопределенный вектор z. В этих условиях, как уже указывалось, методологически целесообразно различать три основные ситуации:

1. z — случайный вектор с известным законом распределения; вектор выходных параметров y реализуется многократно для различных значений вектора z (оптимизация в условиях риска).
2. Вектор y реализуется однократно при заранее неизвестном векторе z.
3. Выходные параметры y реализуются многократно; вектор z изменяется неизвестным образом, но не является случайным, либо распределение вероятностей z оказывается неизвестным.

Первый случай, например, характерен для анализа технологического разброса параметров при массовом производстве каких-либо изделий. Проектирование уникальных изделий происходит в условиях неопределенности второго типа. Третий вариант возникает при моделировании влияния неконтролируемых параметров внешней среды, определяющих условия эксплуатации и функционирования объекта или системы. Приведенные примеры, разумеется, не исчерпывают все возможные случаи, когда оказывается оправданным следовать приведенным предположениям.

На практике возникают и более сложные ситуации. Например, часть компонент вектора z может иметь случайный характер с известными законами распределения, а некоторые компоненты могут меняться непред-

сказуемым образом, но не обладать свойством статистической устойчивости.

Обратимся к методам оптимизации при наличии неопределенности первого типа. Наиболее простой путь заключается в переходе от выражения (8.16) к зависимости

$$y = F(x, z_M), \tag{8.17}$$

где $z_M = M(z)$ — математическое ожидание случайного вектора z. В модели (8.16) неопределенность формально отсутствует и могут применяться методы исследования детерминированных моделей. Однако получаемые при этом результаты должны интерпретироваться с учетом вероятностной природы вектора z. При решении задачи (8.17) гарантируется лишь оптимальность "в среднем" для достаточно большого числа реализаций z.

Второй возможный подход основан на использовании представления

$$y = M(F(x, z)), \tag{8.18}$$

что соответствует критерию математического ожидания из *разд. 3.2*.

Понятно, что переход от модели (8.16) к (8.17) или (8.18) является неформальным актом и построение окончательной математической модели должно опираться на дополнительную информацию о задаче и на эвристические представления исследователя о действительных целях оптимизации.

При использовании модели (8.18) однокритериальная задача оптимизации может быть сформулирована следующим образом:

$$F_0(x) = M(J(x, z)) \to \min, \; x \in D, \tag{8.19}$$

$$D = \left\{ x \in R^n \mid F_i(x) = M(g_i(x, z)) \le 0, \; i = \overline{1, l} \right\}.$$

Предполагается, что исходная многокритериальная задача предварительно редуцирована к одной или нескольким задачам (8.19) со скалярным критерием качества.

Задача (8.19) является задачей стохастического программирования. В отличие от детерминированной постановки функционалы задачи (8.19) не заданы в явном виде и для их вычисления необходимо проводить усреднение по z, что, вообще говоря, как уже указывалось, связано с вычислением многомерных интегралов. Последнее приводит к большим вычислительным затратам в случае прямого применения детерминистских методов нелинейного программирования. Более эффективными в ряде случаев оказываются процедуры стохастического программирования, основанные на информации о конкретных реализациях функционалов $J(x, z)$, $g_i(x, z)$, отвечающих различным значениям вектора z.

Один из вариантов подобных методов сводит задачу (8.19) к последовательности детерминистских задач:

$$J(x,z^k) \to \min,\ x \in D_k \qquad (8.20)$$

$$D_k = \{x \in R^n \mid g_i(x, z^k) \le 0,\ i = 1, \ldots, l\}$$

при фиксированных значениях случайного вектора $z = z^k$, $k = 1, 2, \ldots$. Эти значения должны вырабатываться датчиком случайных чисел в соответствии с заданной плотностью распределения z. Решение задачи (8.20), отвечающее вектору z^k, обозначим через x^k. Тогда последовательность векторов $\{\tilde{x}^k\}$, сходящаяся к решению исходной задачи (8.19), строится следующим образом: $\tilde{x}^{k+1} = \tilde{x}^k + \alpha_k(x^{k+1} + \tilde{x}^k)$, $\alpha_k > 0$, где $\tilde{x}^2 = x^1 + \alpha_1(x^2 - x^1)$. Для сходимости процесса необходимо выполнение условий

$$\alpha_k \underset{k\to\infty}{\to} 0;\ \sum_{k=1}^{\infty}\alpha_k = \infty;\ \sum_{k=1}^{\infty}\alpha_k^2 < \infty.$$

Этим требованиям удовлетворяет, например, последовательность $\{\alpha_k = 1/k\}$. На практике, однако, возникают проблемы, связанные с эффективным выбором α_k для повышения скорости сходимости метода.

Рассмотрим второй и третий типы неопределенности. Информация о статистических свойствах вектора z, даже если она и имеется, здесь уже не может быть эффективно использована. В указанных условиях целесообразно производить расчет "на наихудший случай", используя принцип гарантированного результата *(см. разд. 3.3)* и дополнительную информацию вида $z \in G_z$, где G_z — некоторое ограниченное множество. Соответствующая задача оптимизации формулируется следующим образом:

$$F_0(x) = \max_{z\in G_z} J(x,z) \to \min_{x\in D} \qquad (8.21)$$

$$D = \{x \in R^n \mid F_i(x) = \max_{z\in G_z} g_i(x,z) \le 0 \quad (i = \overline{1,l})\}.$$

Из выражения (8.21) видно, что вычисление функционалов, задающих критерий и ограничения, сопряжено с решением вспомогательных задач оптимизации. В результате трудоемкость процедуры в целом оказывается достаточно высокой. Аналогичные замечания справедливы и для других критериев, применяемых в условиях данного типа неопределенностей, например, для критериев Гурвица *(см. разд. 3.3)*.

Как видно из изложенного, регулярный учет случайных факторов в процессе решения общей задачи оптимизации (когда множества допустимых

значений x и z оказываются бесконечными) алгоритмически достаточно сложен и в настоящее время ограничен лишь относительно простыми ситуациями. Поэтому в сложных случаях чаще вначале решается задача детерминистской оптимизации при фиксированных, например средних, значениях z. Далее в отношении наилучшего детерминистского результата проводится тот или иной вид статистического анализа. Более детально методы статистических расчетов изложены в специальных работах.

8.4. Декомпозиция задач оптимизации больших систем

Под методами *разделения*, или *декомпозиции*, понимаются способы сведения исходной "сложной" задачи к нескольким "простым", поддающимся решению стандартными методами математического программирования. Рассмотрим два характерных подхода.

Алгоритмически наиболее простым является метод введения макроописания объекта оптимизации. Он допускает следующее формальное представление.

Исходная задача формулируется в виде

$$J(x) \to \min_{x \in D}. \tag{8.22}$$

Далее вводятся агрегированные характеристики

$$z_i = \varphi_i(x_1, x_2, ..., x_n), \tag{8.23}$$

где вектор $z = (z_1, z_2, ..., z_s)$ должен иметь существенно меньшую по сравнению с x размерность. Функции φ_i строятся таким образом, чтобы:

1) критерий (8.22) был представим в виде суперпозиции отображений.

$$J(x) = F[\varphi(x)] = F(z);\ z \in D_z = \varphi(D); \tag{8.24}$$

2) существовали обратные отображения φ_i^{-1}, позволяющие для $\forall z \in D_z$ достаточно эффективно вычислять $x = \varphi^{-1}(z) \in D$.

Приведем одну из возможных "электронных" интерпретаций изложенной формальной конструкции, известную в теории оптимизации технических объектов как метод "аппроксимации и реализации".

Пусть, например, требуется спроектировать электронную схему, имеющую заданную амплитудно-частотную характеристику (АЧХ). Тогда на этапе аппроксимации строится дробно-рациональная передаточная функция $W(p, z)$ комплексного переменного p и вектора $z = (z_1, z_2, ..., z_s)$

коэффициентов полиномов в числителе и знаменателе. Вектор z вычисляется как решение задачи

$$F(z) \to \min_{z \in D_z}, \tag{8.25}$$

где $F(z)$ характеризует близость расчетной и желаемой АЧХ, а множество D_z задается условиями физической реализуемости функции $W(p, z)$. На этапе реализации по найденным z_i^* выбирается структурная схема устройства, конкретизирующая вид функций φ_i в соотношениях (8.23). А далее определяется одно из решений системы уравнений

$$\varphi_i(x_1, x_2, \ldots, x_n) = z_i^* \quad (i = \overline{1, s}). \tag{8.26}$$

Разрешимость системы (8.26) следует из выполненных на этапе аппроксимации условий физической реализуемости.

Такое разбиение процесса проектирования на два этапа обычно мотивируется соображениями удобства, позволяющими на этапе аппроксимации не рассматривать конкретные объекты и получать некоторые общие результаты. Однако не менее важная особенность такого подхода связана с идеей декомпозиции. Обратимся к предыдущему примеру. Пусть трудоемкость решения задачи минимизации функционала линейно зависит от размерности n вектора x и приближенно оценивается числом kn. Допустим также, что основная работа выполняется при вычислении "расстояния" между аппроксимируемой и аппроксимирующей АЧХ. Иначе говоря, коэффициенты z_i по заданным x_i рассчитываются относительно просто, и, напротив, реализация зависимостей $F(z)$, $J(x) = F[\varphi(x)]$ как функций z и x оказывается достаточно трудоемкой.

В этом случае трудоемкость прямого решения задачи $J(x) \to \min$ без разбиения ее на этапы аппроксимации и реализации оценивается числом kn, а трудоемкость этапа аппроксимации — числом ks. Пренебрегая вычислительными затратами на этапе реализации, связанными с получением значений z_i по формулам (8.23), получим выигрыш во времени оптимизации в результате декомпозиции приблизительно в n / s раз.

Второй известный принцип декомпозиции связан с сокращением множества D потенциально возможных решений. Это может быть выполнено с помощью введения вектора вспомогательных частных критериев $u(x) = [u_1(x), \ldots, u_m(x)]$. Предполагается, что критерий $J(x)$ удовлетворяет следующему условию монотонности: для любых двух точек $x', x'' \in D$ из системы неравенств $u_i(x') \geq u_i(x'')$, $i = \overline{1, m}$ следует $J(x') \geq J(x'')$. Таким образом, если решение x'' оказывается более предпочтительным, чем x' по векторному критерию $u(x)$, то оно будет более предпочтительным и с позиций скалярного критерия $J(x)$.

При выполнении условий монотонности исходная задача (8.22) может быть заменена следующей

$$J(x) \to \min_{x \in P_u(D)}, \tag{8.27}$$

где $P_u(D)$ — множество решений, эффективных (Парето-оптимальных) по векторному критерию u на множестве D. Очевидно, $P_u(D) \subset D$. Если достигнутое сужение множества D значительно, то задача (8.27) оказывается проще исходной и цель декомпозиции считается достигнутой.

Техническая сторона изложенного подхода заключается в следующем. Предполагается, что критерии $u_i(x)$ в отличие от $J(x)$ оказываются эффективно вычислимыми на компьютере с относительно малыми затратами машинного времени. Решая последовательность оптимизационных задач вида max

$$\max_i \alpha_i u_i(x) \to \min_{x \in D}; \quad \alpha = (\alpha_1, \alpha_2, ..., \alpha_m) \in \alpha^\varepsilon,$$

где α^ε — некоторая дискретная сетка в множестве

$$A = \left\{ \alpha = (\alpha_1, \ ..., \alpha_m) \middle| \alpha_i > 0; \sum_{i=0}^{m} \alpha_i = 1 \right\},$$

мы фактически реализуем некоторую функцию x(α), определяющую точки x, в которых необходимо вычислять исходный глобальный критерий $J[x(\alpha)]$. Число таких точек определяется числом узлов сетки α^ε. Существенно, что размерность пространства векторов α может оказаться значительно меньше (на несколько порядков) размерности исходного пространства векторов x.

Для построения α^ε-сетки может быть использован метод зондирования пространства векторов α, основанный на построении известных из теории математического моделирования $ЛП_\tau$-последовательностей, обладающих свойством равномерного заполнения заданной многомерной области.

Приведенные подходы далеко не исчерпывают все известные методы декомпозиции (см. список литературы).

8.5. Особенности оптимизационных задач

Возникающие на практике оптимизационные задачи обладают особенностями по сравнению с общей постановкой задачи нелинейного программирования, что необходимо учитывать и использовать при прове-

дении реальных вычислений. Основные характерные черты заключаются в следующем.

1. Алгоритмическое задание функционалов, задающих критерии и ограничения, существенно увеличивает трудоемкость их вычисления и вынуждает ограничиваться методами оптимизации, не использующими в явном виде выражения для производных.
2. Критерии оптимальности, применяемые в реальных задачах оптимизации, часто имеют характерную структуру, позволяющую, например, строить специальные методы оптимизации второго порядка, использующие упрощенные выражения для вторых производных.
3. Однократное вычисление функционала, задающего критерий оптимальности, обычно связано с достаточно сложным и трудоемким решением соответствующей задачи анализа. Наиболее эффективными целесообразно считать алгоритмы, которые в процессе оптимизации наименьшее число раз обращаются к вычислению значений минимизируемых функционалов и ограничений для получения решений с требуемой точностью.
4. Если оптимизируется сложная многопараметрическая система, то ее обычно можно представить как некоторую совокупность связанных подсистем меньшей размерности. Учет подобной структуры системы позволяет строить более рациональные методы оптимизации по сравнению с традиционными универсальными алгоритмами нелинейного программирования.
5. Невыпуклая структура минимизируемых функционалов существенно понижает эффективность обычных методов нелинейной оптимизации, особенно если такой структуре сопутствует описываемая ниже овражная ситуация.
6. Как свидетельствует практика оптимизационных расчетов, возникающие оптимизационные задачи являются, как правило, плохо обусловленными. Это определяет характерную овражную структуру поверхностей уровня минимизируемых функционалов и вызывает резкое замедление сходимости стандартных методов оптимизации.

Указанные характерные черты оптимизационных задач определяют конкретные требования к практическим методам оптимизации и использующим их программным системам оптимизации. С позиций существующего аппарата нелинейной оптимизации наиболее существенными оказываются особенности, отмеченные в п. п. 5, 6. Основная трудность состоит в том, что математически проблема невыпуклости минимизируемого функционала оказывается неразрешимой в силу сложности класса невыпуклых оптимизационных задач. Основной вывод заключается в том, что даже для гладких одноэкстремальных функционалов в

задачах с не очень малой размерностью пространства управляемых параметров скорость сходимости любого метода (равномерно по всем задачам) безнадежно мала и попытка построить общий метод, эффективный для всех задач с гладкими невыпуклыми целевыми функционалами, заранее обречена на неудачу.

Однако с позиций специалиста по реальным вычислениям представляет интерес задача построения методов оптимизации, вырабатывающих эффективные направления поиска в точках пространства, где стандартные процедуры оказываются неработоспособными. В последующих главах книги эта проблема решается на основе построения методов, которые как в выпуклой, так и в невыпуклой и одновременно овражной ситуации локально (для квадратичной модели) дают существенно более удовлетворительные по скорости убывания функционала результаты по сравнению с традиционными методами.

О правомерности развиваемого подхода и целесообразности использования соответствующих алгоритмов можно судить только по результатам решения реальных задач. Создание новых методов будет оправданно, если их применение окажется эффективным для заведомо непустого множества практических ситуаций, вызывающих трудности для известных поисковых процедур. В данном случае такое множество можно указать заранее — это множество задач с целевыми функционалами, близкими к кусочно-квадратичным, не обязательно выпуклым зависимостям. Типичность подобных функций подтверждается практикой решения реальных задач. Речь, следовательно, идет не о замене традиционных методов новыми, а о некотором существенном расширении уже имеющегося арсенала методов и алгоритмов оптимизации.

Подтверждаемая экспериментально достаточно высокая работоспособность рассматриваемых методов никак не противоречит тезису о безнадежной трудности невыпуклых задач. Дело, по-видимому, заключается в том, что на практике достаточно редко реализуются те специальные структуры невыпуклых задач, которые и приводят к пессимистическим теоретическим оценкам. Заметим, что именно так обстоит дело со знаменитым симплекс-методом линейного программирования: строгий теоретический анализ показывает (вопреки практике) его достаточно высокую трудоемкость.

8.6. Некоторые стандартные схемы оптимизации

На основе сделанных ранее замечаний в зависимости от реальной ситуации могут формироваться различные вычислительные схемы оптимиза-

ции. Ниже рассмотрены некоторые варианты таких схем, охватывающие значительное число практических задач.

Задачи аппроксимации. В большом числе случаев задача оптимизации некоторой системы состоит в реализации заданной зависимости некоторой величины W от непрерывной переменной s, например, частоты или времени. В качестве таких зависимостей при оптимизации реальных объектов могут выступать амплитудно-частотные, фазо-частотные, переходные и другие характеристики. Необходимо подобрать вектор управляемых параметров таким образом, чтобы "расстояние" между заданной и расчетной характеристиками было минимальным.

К задачам аппроксимации относятся также многочисленные задачи идентификации, т. е. задачи выбора параметров моделей реальных объектов по экспериментально полученным характеристикам.

Критерии оптимальности в этих случаях чаще всего формируются одним из следующих способов:

$$J_1(x) = \sum_{i=1}^{N} \alpha_i^2 [W(x, s_i) - W^*(s_i)]^2 \to \min_{x \in D}; \tag{8.28}$$

$$J_2(x) = \max_{i=\overline{1,N}} \alpha_i \left| W(x, s_i) - W^*(s_i) \right| \to \min_{x \in D}, \tag{8.29}$$

где D — множество допустимых значений x_i; α_i — весовые коэффициенты, определяющие необходимую точность аппроксимации в отдельных точках диапазона изменения независимой переменной s; s_i, $i = \overline{1, N}$ — дискретная сетка значений s, при которых происходит сравнение заданной $W^*(s)$ и расчетной $W(x, s)$ характеристик.

Каждый из приведенных критериев имеет свои особенности, существенные для организации вычислительного процесса. Функционал J_1 достаточно прост и обладает свойством "гладкости". Именно, если функция $W(x, s)$ является дважды непрерывно дифференцируемой функцией x, то этим же свойством будет обладать зависимость $J_1(x)$, что существенно облегчает последующую процедуру оптимизации. Основная характерная черта J_1 заключается в ограниченной точности аппроксимации отдельных слагаемых. Иначе говоря, плохая точность аппроксимации в некоторых точках при больших значениях N может компенсироваться хорошей точностью в других точках. Иногда эта ситуация не соответствует смыслу решаемой задачи. Эта особенность отсутствует в критерии (8.29), однако он не сохраняет характеристики гладкости функции $W(x, s)$, что требует привлечения специальных методов оптимизации.

Как показывает практика, достаточно простой и надежный способ решения задач аппроксимации заключается в использовании гладких среднестепенных аппроксимаций минимаксного критерия J_2. Согласно этому подходу вместо решения задачи (8.29) ищется минимум функционала со среднестепенной структурой:

$$J_3(x) = \sum_{i=1}^{N} \varphi_i^{\nu}(x) \to \min_{x \in D}, \nu = 2, 3, ..., \tag{8.30}$$

где $\varphi_i(x) = \alpha_i \left| W(x, s) - W^*(s_i) \right|$.

При достаточно больших значениях ν решения задач (8.29), (8.30) будут почти совпадать. Действительно, справедливо предельное соотношение

$$\left(\sum_{i=1}^{N} \varphi_i^{\nu} \right)^{1/\nu} \to \max_i \varphi_i, \nu \to \infty,$$

где $\varphi_i \geq 0$, $i = \overline{1, N}$ — произвольные числа. Кроме этого, можно показать, что операция извлечения корня ν-й степени не влияет на локализацию точки минимума.

Функционал J_3 совмещает в себе особенности функционалов J_1, J_2. Являясь гладким подобно J_1, J_3 не допускает значительных отклонений точности аппроксимации в отдельных точках. Иногда такой подход оказывается наиболее адекватным истинным целям моделирования.

При решении практических задач на основе критерия J_3 целесообразно пошаговое увеличение параметра ν, начиная с $\nu = 2$. Таким способом обычно удается избежать переполнения разрядной сетки компьютера при возведении первоначально больших значений локальных ошибок аппроксимации φ_i в высокую степень ν. Кроме этого, проводя в интерактивном режиме оценку получаемых в процессе увеличения ν решений, можно вовремя прервать процесс, если получены удовлетворяющие разработчика результаты. Заранее задать оптимальное значение ν обычно трудно. Как правило, при практических расчетах значение ν не превышает 10—15.

Рассмотренный подход очевидным образом распространяется на вектор-функцию W. При этом в качестве функций $\varphi_i(x)$ могут использоваться зависимости $\varphi_i(x) = \alpha_i \left\| W(x, s_i) - W^*(s_i) \right\|$.

Задачи решения систем неравенств. Задачи моделирования и оптимизации реальных или проектируемых систем часто могут быть представлены как задачи решения систем неравенств. Так, например, к системам неравенств приводят формализации задач проектирования, цель решения ко-

торых заключается в увеличении процента создаваемых при массовом производстве годных изделий в условиях статистического разброса параметров. Пусть технические требования (ТТ) к проектируемому устройству выражаются системой функциональных и критериальных ограничений

$$y_i(x) \le t_i,\ i = 1, ..., m. \tag{8.31}$$

Годным считается изделие, выходные параметры y_i которого удовлетворяют соотношениям (8.31).

Основная трудность заключается в том, что номинальные значения x_i^*, т. е. значения, на которые настроен соответствующий технологический процесс, могут быть выбраны так, что ТТ выполняются. Однако из-за случайных отклонений параметров в процессе производства некоторые из ТТ могут оказаться нарушенными. Ставится задача такого выбора вектора x^*, чтобы случайные отклонения в технологическом процессе, а также в условиях эксплуатации устройства, в наименьшем числе случаев приводили бы к нарушению ТТ.

Известен достаточно простой и эффективный подход к решению сформулированной задачи, которая может иметь и иную интерпретацию. Потребуем, чтобы ТТ выполнялись с некоторыми запасами

$$y_i(x) + \delta_i \le t_i;\ \delta_i > 0, \tag{8.32}$$

где δ_i характеризует рассеяние i-го выходного параметра в результате статистических вариаций внутренних и внешних параметров. Требование (8.32) эквивалентно неравенству

$$z_i(x) = \frac{t_i - y_i(x)}{\delta_i} - 1 \ge 0. \tag{8.33}$$

Величина z_i называется "запасом работоспособности" по i-му выходному параметру. В результате имеем многокритериальную задачу

$$z_i(x) \to \max_{x \in D'},\ i = 1, ..., m. \tag{8.34}$$

Здесь D' — множество, в котором выполняются прямые ограничения на управляемые параметры. Предполагается, что функциональные и критериальные ограничения учтены в результате расширения системы неравенств (8.31). С помощью соответствующей замены переменных или перевода прямых ограничений в ранг функциональных можно избавиться от ограничений и принять $D' = R^n$.

В практике оптимизации получила распространение максиминная свертка векторного критерия (8.34), приводящая к следующему обобщенному показателю качества:

$$J_4(x) = \min_{i=1,m} z_i(x) \to \max_{x \in R^n}. \tag{8.35}$$

Иногда в выражение (8.6.8) вводятся весовые коэффициенты α_i:

$$J_4(x) = \min \alpha_i z_i(x) \to \max.$$

Их роль заключается во введении некоторого стабилизирующего фактора. Действительно, увеличение некоторого весового коэффициента α_i усиливает влияние запаса z_i на результирующую целевую функцию. В результате уже незначительное нарушение соответствующего неравенства приводит к существенному ухудшению целевой функции. С другой стороны, уже при незначительных положительных значениях z_i имеем запас в выполнении i-го неравенства, сравнимый с запасами по остальным выходным параметрам.

Выбор параметров δ_i по существу определяет единицы измерения разностей $t_i - y_i(x)$. Этот выбор обычно облегчается конкретным физическим смыслом δ_i, которые, как правило, могут задаваться как характеристики рассеяния. Для их определения может проводиться статистический анализ в окрестности текущей точки x, что позволяет говорить о превращении показателя (8.35) в статистический критерий. Значения δ_i обычно имеют смысл трехсигмовых допусков, которые периодически уточняются в процессе оптимизации. Весьма часто величины δ_i задаются как исходные данные на основе априорной информации, что заметно сокращает трудоемкость процедуры оптимизации.

Функционал J_4, так же как и J_2, не является гладким, что существенно усложняет ситуацию и требует применения специальных оптимизирующих процедур. Ниже излагается альтернативный подход, основанный на процедуре сглаживания исходного функционала с последующим обращением к методам гладкой оптимизации.

Очевидно,

$$\arg\min_i z_i = \arg\max_i [\exp(-z_i)].$$

Поэтому задача (8.35) эквивалентна задаче

$$\max_i [\exp(-z_i)] \to \min_x. \tag{8.36}$$

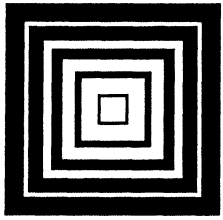
Для задачи (8.36) применима среднестепенная свертка (8.30), если принять $\varphi_i(x) = \exp[-z_i(x)]$. В результате приходим к следующему критерию оптимальности:

$$J_5(x) = \sum_{i=1}^{m} \exp[-\nu z_i(x)] \to \min_x, \quad \nu = 1, 2, \ldots . \tag{8.37}$$

Как показывает вычислительная практика, трудности оптимизации систем по критериям типа минимального запаса работоспособности (8.35) часто возникают из-за негладкости критериев, приводящей к преждевременной остановке поисковой процедуры. Целесообразно сразу обращаться к модифицированным критериям (8.37) с применением на первом этапе простейших алгоритмов оптимизации типа метода простого покоординатного спуска.

Использование среднестепенных критериев оптимальности в задачах оптимизации, где, по существу, необходим минимаксный подход, оправданно также с позиций рассмотренного явления плохой обусловленности. Развитая в настоящее время техника решения негладких оптимизационных задач достаточно сложна и в невыпуклой овражной ситуации многие алгоритмы теряют эффективность. В то же время излагаемые далее методы позволяют получать удовлетворительные результаты для невыпуклых овражных функционалов при условии их гладкости. При этом удается использовать структурные особенности функционалов (8.30), (8.37) для увеличения эффективности соответствующих вычислительных процедур.

Глава 9



Проблема плохой обусловленности

9.1. Явление овражности

В этом разделе анализируются часто возникающие на практике случаи получения неудовлетворительных результатов с помощью стандартных методов конечномерной оптимизации, применяемых в задачах моделирования, численного эксперимента и других задач системного анализа (см. пример об управляемой системе второго порядка во *Введении*). Как правило, это выражается в резком увеличении затрат машинного времени, а в некоторых случаях — в невозможности получения приемлемых результатов из-за полной остановки алгоритма задолго до достижения оптимальной точки.

Возникновение подобных трудностей связывается далее со специальной формой плохой обусловленности матрицы вторых производных минимизируемых целевых функционалов, приводящей к характерной овражной структуре поверхностей уровня критерия оптимальности.

Рассмотрим критерий оптимальности, зависящий от двух управляемых параметров x_1, x_2:

$$J(x_1, x_2) = g_0^2(x_1, x_2) + \sigma g_1^2(x_1, x_2) \to \min_x, \tag{9.1}$$

где σ — достаточно большое положительное число. Рассмотрим также уравнение

$$g_1(x_1, x_2) = 0, \tag{9.2}$$

определяющее в простейшем случае некоторую зависимость $x_2 = f(x_1)$. Тогда при стремлении параметра σ к бесконечности значение функционала J в каждой точке, где $g_1(x_1, x_2) \neq 0$, будет неограниченно возрастать по абсолютному значению, оставаясь ограниченным и равным $g_0^2(x_1, x_2)$ во всех точках на кривой $x_2 = f(x_1)$.

То же самое будет происходить с нормой вектора градиента $J'(x) = (\partial J/\partial x_1, \partial J/\partial x_2)$, где

$$\partial J/\partial x_1 = 2g_0(x_1, x_2)\, \partial g_0/\partial x_1 + 2\sigma g_1(x_1, x_2)\partial g_1/\partial x_1,$$

$$\partial J/\partial x_2 = 2g_0(x_1, x_2)\partial g_0/\partial x_2 + 2\sigma g_1(x_1, x_2)\partial g_1/\partial x_2.$$

Линии уровня $J(x) = \text{const}$ для достаточно большого σ представлены на рис. 9.1. Там же стрелками показано векторное поле антиградиентов, определяющее локальные направления наискорейшего убывания $J(x)$.

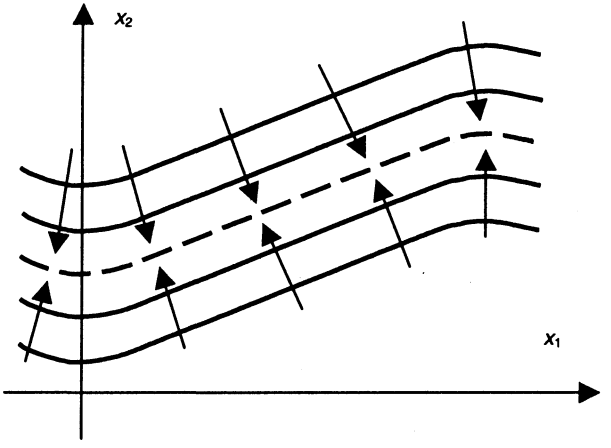


Рис. 9.1. Линии уровня минимизируемого функционала

Ясно, что минимальные значения $J(x)$ следует искать вдоль зависимости $x_2 = f(x_1)$, определяющей так называемое *дно оврага*. Из (9.1) следует, что изменение $J(x)$ вдоль дна задается выражением $g_0^2(x_1,x_2)$ и не зависит от значения параметра σ. Таким образом, задача минимизации $J(x)$ сводится к минимизации функционала $g_0^2(x_1,f(x_1))$ от одной переменной x_1. В общем случае уравнение $x_2 = f(x_1)$ обычно неизвестно.

Приведенный пример овражной структуры критерия оптимальности является достаточно простым, хотя и из него уже видны принципиальные трудности, связанные, например, с применением широко распространенных методов спуска по антиградиенту.

Действительно, из рис. 9.1 следует, что направления поиска, задаваемые антиградиентами, оказываются неэффективными.

Приводя достаточно быстро процесс поиска на дно оврага, эти направления в окрестности дна начинают осциллировать, оставаясь почти перпендикулярными направлению, указывающему в точку минимума $J(x)$.

Возможны различные усложнения и обобщения рассматриваемой ситуации. Например, уравнение (9.2) может, вообще говоря, определять не одно, а несколько решений $x_2 = f(x_1)$, каждое из которых определяет свой овраг. Более существенным обобщением является предположение о наличии многомерного дна оврага. Чтобы проиллюстрировать это явление, обратимся к следующему примеру:

$$J(x) = g_0^2(x) + \sigma \sum_{i=1}^{m} g_i^2(x) \to \min_x; \qquad x \in R^n;\ n > 2;\ m < n. \tag{9.3}$$

В этом случае дно оврага задается системой уравнений

$$g_i(x) = 0,\ i = 1, \ldots, m, \tag{9.4}$$

что в принципе позволяет выразить m параметров x_i через оставшиеся $n - m$ переменных. Предположим, не ограничивая общности, что уравнения (9.4) определяют следующие зависимости:

$$x_1 = f_1(x_{m+1}, \ldots, x_n);$$

$$\ldots$$

$$x_m = f_m(x_{m+1}, \ldots, x_n).$$

Аналогично предыдущему случаю устанавливаем, что задача минимизации (9.3) при достаточно больших σ эквивалентна минимизации функции $g_0^2(f_1, \ldots, f_m, x_{m+1}, \ldots, x_n)$ от $r = n - m$ переменных $x_{m+1}, \ldots, x_n$. Число r называется *размерностью дна оврага*. Легко представить, что, например, для $n = 3$ поверхности уровня одномерного оврага имеют характерный "сигарообразный" вид, а для двухмерного оврага они будут близки к деформированным дискам.

В любом случае для овражной ситуации определяющим фактором является специальная структура поверхностей уровня $J(x)$, сильно отличающаяся от сферической. Характерно наличие некоторой области притяжения (дно оврага) $Q \subset R^n$, содержащей оптимальную точку $x^* = \arg\min J(x)$. При этом норма вектора градиента $J'(x)$ для $x \in Q$, как правило, существенно меньше, чем в остальной части пространства.

Овражную структуру могут иметь не только функционалы вида (9.1), (9.3), явно содержащие большой параметр σ.

Можно привести следующий пример квадратичного функционала:

$$f(x_1, x_2) = 0{,}250025\, x_1^2 + 0{,}49995\, x_1 x_2 + 0{,}250025\, x_2^2 - x_1 - x_2. \qquad (9.5)$$

Линии уровня $f(x) = \text{const}$ функционала (9.5) представляют семейство подобных эллипсоидов с центром в точке (1,1). Длины полуосей эллипсоидов относятся при этом как 1:100. Указать малый параметр в выражении (9.5) нельзя, хотя овражная ситуация налицо и так же, как и в предыдущих случаях, явно выделяется дно оврага (прямая *ab* на рис. 9.2), описываемое уравнением $x_2 = -x_1 + 2$.

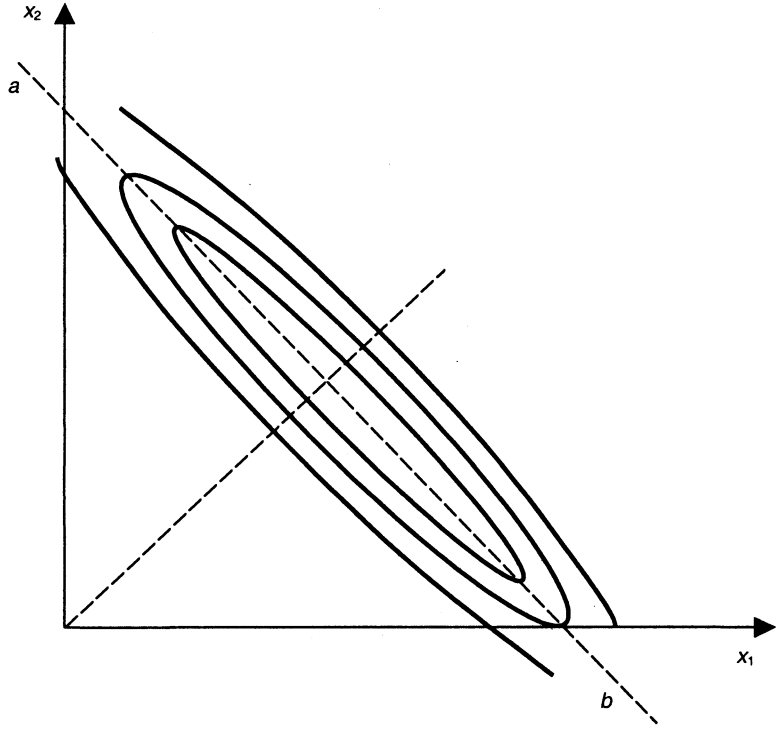


Рис. 9.2. Квадратичный функционал

Подставляя выражение для x_2 в (9.5), снова приходим к эквивалентной задаче меньшей размерности

$$f_1(x_1) = 10^{-4}(x_1 - 1)^2 - 1 \longrightarrow \min.$$

Необходимость выделения овражных оптимизационных задач в отдельный класс обусловлена, с одной стороны, значительными вычислитель-

ными трудностями при их решении стандартными для практики моделирования и вычислительного эксперимента методами, а с другой стороны, бесспорным фактом важности данного класса задач для большинства приложений и особенно для задач оптимального выбора параметров как реально существующих систем — технических, экономических, экологических, так и вновь создаваемых систем, находящихся в стадии проектирования.

Существуют различные методы, ориентированные на решение рассматриваемых оптимизационных задач, однако и в настоящее время проблема минимизации овражных функционалов является актуальной. Особенно важно решить вопрос минимизации овражных и одновременно невыпуклых функционалов, т. к. именно в этой ситуации "отказывает" большинство из известных методов конечномерной оптимизации.

9.2. Формальное определение. Критерии овражности целевого функционала

Пусть решается задача $J(x) \to \min;\ J \in C^2(D);\ x \in D \subset R^n$. Будем предполагать далее, что функционал $J(x)$ ограничен снизу на множестве D.

Траектория наискорейшего спуска (ТСН) $x(\tau)$ функционала $J(x)$ задается известным векторным дифференциальным уравнением

$$dx/d\tau = -J'(x);$$

$$J'(x) = (\partial J/\partial x_1, ..., \partial J/\partial x_n). \qquad (9.6)$$

Эти траектории обладают специфическими чертами. Например, для функционала (9.5) имеем

$$x(\tau) = \sum_{i=1}^{2} [\alpha_i^* + (\alpha_i^0 - \alpha_i^*)\exp(-\lambda_i \tau)] u_i, \qquad (9.7)$$

где

$$x(0) = \sum_{i=1}^{2} \alpha_i^0 u_i;\ x^* = (1;1) = \sum_{i=1}^{2} \alpha_i^* u_i;$$

$$\lambda_1 = 1;\ \lambda_2 = 10^{-4};\ u_1 = \frac{1}{\sqrt{2}}(1;1);\quad u_2 = \frac{1}{\sqrt{2}}(-1;1).$$

Из выражения (9.7) видно, что ввиду наличия быстро затухающей и медленно затухающей экспонент отчетливо выделяются два участка с существенно различным поведением решения. Первый, сравнительно непродолжительный, характеризуется большими значениями производных $dx_i(\tau)/d\tau$ и означает спуск на дно оврага. На дне выполняются условия типа (9.2) и норма вектора градиента, а с ней и производные $dx_i(\tau)/d\tau$ становятся относительно малыми. Поэтому для второго участка характерно относительно плавное изменение переменных x_i. Таким образом, прослеживается полная аналогия с поведением решений так называемых жестких систем обыкновенных дифференциальных уравнений. В связи с этим в [32] было предложено следующее общее определение.

Определение 9.1

Функционал $J(x)$ называется *овражным*, если отвечающая ему система дифференциальных уравнений (9.6) — жесткая.

Однако при решении задач оптимизации более конструктивным может оказаться приведенное ниже непосредственное определение овражного функционала. Это определение не содержит, в частности, таких неестественных для задач оптимизации требований, как необходимость задания промежутка интегрирования уравнения (9.6), что предполагается в теории жестких систем.

Определение 9.2

Функционал $J(x) \in C^2(D), D \subset R^n$ называется *овражным (жестким) во множестве* $Q \subset D$, если найдутся такие числа $\delta > 0$, $\sigma >> 1$, что

$$1)\ \forall x \in Q_\delta : \lambda_1[J''(x)] \geq \sigma \left|\lambda_n[J''(x)]\right|;$$

$$2)\ \forall x \in Q : \operatorname{Arg} \min_{x' \in X_\delta(x)} J(x') \subset X_\delta(x) \cap Q; \qquad (9.8)$$

$$3)\ \forall x \in Q : L[X_\delta(x) \cap Q] \leq \sigma^{-1} L[X_\delta(x)],$$

где $C^2(D)$ — множество дважды непрерывно дифференцируемых на D функционалов;

$$X_\delta(x) = \left\{x' \in R^n \,\middle|\, \|x' - x\| \leq \delta\right\}; \quad Q_\delta = \bigcup_{x \in Q} X_\delta(x);$$

$\lambda_i(A)$ — собственные числа матрицы вторых производных $A = J''(x)$, упорядоченные по убыванию: $\lambda_1(A) \geq \lambda_2(A) \geq \ldots \geq \lambda_n(A)$; $L(S)$ — минимальная константа Липшица в соотношении

$$\|J'(x') - J'(x)\| \leq L(S)\|x' - x\|; \ \forall x', x \in S \subset R^n.$$

Основным в определении 9.2 является первое условие, констатирующее резко несимметричное расположение спектра матрицы $J''(x)$ относительно начала координат: $\lambda_i \in [-m, M]$, $M >> m > 0$. Второе и третье условия необходимы для описания свойства устойчивости множества Q: можно показать, что все ТСН, начинающиеся в любой точке $x \in Q_\delta$, быстро попадают в достаточно малую окрестность Q_ε ($\varepsilon << \delta$) множества Q и остаются там до выхода из множества Q_δ.

Как правило, оказывается достаточной более грубая модель явления овражности (жесткости), когда предполагается, что собственные числа матрицы вторых производных можно отчетливо разделить на две группы, в одну из которых входят собственные числа, по модулю намного превосходящие элементы второй группы. Будет использоваться следующее определение.

Пусть в $D \subset R^n$ задана r-мерная поверхность (конфигурационное пространство)

$$Q = \left\{ x \in D \middle| g_i(x) = 0,\ i = \overline{1, n-r} \right\},\ g_i \in C^2(D).$$

Определение 9.3

Функционал $J(x) \in C^2(D)$, $D \subset R^n$ называется овражным (жестким) в множестве Q, если найдутся такие числа $\delta > 0$, $\sigma >> 1$, что

$$1)\ \forall x \in Q_\delta : \lambda_1[J''(x)] \geq \ldots \geq \lambda_{n-r}[J''(x)] \geq \sigma \left| \lambda_{n-r+1}[J''(x)] \right| \geq \ldots \geq \sigma \left| \lambda_n[J''(x)] \right|;$$

$$2)\ \forall x \in Q : \operatorname{Arg} \min_{x' \in X_\delta} J(x') \subset X_\delta(x) \cap Q; \qquad (9.9)$$

$$3)\ \forall x \in Q : L[X_\delta(x) \cap Q] \leq \sigma^{-1} L[X_\delta(x)].$$

Число r называется размерностью (дна) оврага Q.

Пример 9.1. Рассмотрим квадратичный функционал

$$f(x) = 1/2\langle Ax, x\rangle - \langle b, x\rangle + c;\ c = \text{const}. \qquad (9.10)$$

Пусть собственные числа λ_i матрицы $A = f''(x)$ удовлетворяют неравенствам (9.9), а u_i означают соответствующие собственные векторы. Предположим, что $\det A \neq 0$ и обозначим через x^* решение уравнения $Ax = b$.

Примем

$$Q=\left\{x\in R^n \mid \left\langle x-x^*, u_i\right\rangle = 0,\ i=\overline{1,n-r}\right\};$$
$$Q_\delta = R^n;\ \sigma \cong \frac{\lambda_1}{\left|\lambda_{n-r+1}\right|}. \qquad (9.11)$$

Можно доказать, что все три условия будут выполнены.

Таким образом, для квадратичных функционалов при сдвинутом в точку x^* начале координат дно оврага Q совпадает с линейной оболочкой (9.11) собственных векторов, отвечающих малым собственным числам. Это согласуется с интуитивными представлениями, развитыми в *разд. 9.1.*

На этом примере можно проиллюстрировать значение отдельных условий в определениях 9.2 и 9.3. Действительно, для квадратичного функционала (9.10) первое и второе условия могут выполняться для всего пространства $Q = R^n$ и любого $\delta > 0$. Необходимая линейная оболочка собственных векторов может быть выделена только при дополнительном требовании, эквивалентном третьему условию. В то же время требования 1, 3 также оказываются недостаточными. В этом случае сдвиг линейной оболочки Q, являющейся дном оврага, вдоль любого из не вошедших в оболочку собственных векторов не приведет к нарушению условий 1 и 3, а условие 2 при этом нарушится.

Рассмотренные выше модели явления овражности не являются исчерпывающими. Однако они описывают наиболее существенные стороны большинства практических ситуаций, связанных с задачами оптимизации реальных и проектируемых систем.

Определение 9.4

Пусть $\forall x \in Q$, $\det J''(x) \neq 0$. Наименьшее из чисел σ, удовлетворяющих определению 9.2, называется *степенью овражности* $J(x)$ в Q и обозначается $\eta(Q)$. Отношение

$$\eta(x) = \lambda_1(x) / \left|\min_i \lambda_i(x)\right|,\ x \in Q$$

называется *локальной степенью овражности* $J(x)$ *в точке* x. Для вырожденных матриц $J''(x)$ величина $\eta(x)$ принимается равной бесконечности.

Если $J''(x) > 0$, то

$$\eta(x) = \operatorname{cond}[J''(x)] = \max_i \lambda_i(x) / \min_i \lambda_i(x).$$

В общем случае справедливо неравенство $1 \le \eta \le \mathrm{cond}(J'')$. При наличии больших по модулю отрицательных собственных чисел $\lambda_i(x)$ (т. е. при отсутствии овражной ситуации) возможно неравенство $\eta(x) << \mathrm{cond}[J''(x)]$. Из высокой степени овражности $J(x)$ в точке x следует плохая обусловленность матрицы $J''(x)$; обратное неверно. Действительно, пусть спектр матрицы $J''(x)$ расположен в множестве $[-M, m] \cup [m, M]$, $M >> m > 0$, включая граничные значения. Тогда $\eta(x) = 1$, а $\mathrm{cond}[J''(x)] = M / m >> 1$. Данный функционал не будет относиться к классу овражных, что естественно, ибо трудностей при его минимизации, например методом наискорейшего спуска, не возникает. Отличие между двумя характеристиками $\eta(x)$ и $\mathrm{cond}[J''(x)]$ функционала $J(x)$ часто игнорируется и овражными называют функционалы с большим числом $\mathrm{cond}(J'')$, что не оправдывается с позиций основных вычислительных трудностей, возникающих при решении задач оптимизации. Однако, учитывая указанную выше связь между $\eta(x)$ и $\mathrm{cond}(J'')$, овражные задачи, т. е. задачи минимизации овражных или жестких функционалов, далее будут называться плохо обусловленными задачами оптимизации.

В каждом конкретном случае различные значения $\eta(x)$ следует считать большими. Здесь существует полная аналогия с понятием плохой обусловленности матрицы. В большинстве случаев все определяется точностью вычислений и типом применяемого алгоритма оптимизации. Традиционно принято классифицировать задачу как плохо обусловленную, если

$$\log_2 \eta > t, \tag{9.12}$$

где t — длина применяемой разрядной сетки компьютера. Однако и при меньших значениях η для многих алгоритмов могут возникать значительные вычислительные трудности, особенно если овражная структура сопровождается отсутствием выпуклости $J(x)$.

Дополнительным фактором, характеризующим степень сложности оптимизационной задачи и затрудняющим применение традиционных алгоритмов минимизации, является наличие многомерных оврагов с $r > 1$. В указанной ситуации целый ряд методов, специально ориентированных на решение плохо обусловленных задач, становится неэффективным.

Рассмотрим практические методы распознавания овражной ситуации, играющие роль критериев овражности. Наиболее существенной характеристикой оказывается значение показателя η в допустимой области изменения управляемых параметров.

Своеобразным индикатором может служить метод простого градиентного спуска (ПГС), реализуемый по схеме

$$x^{k+1} = x^k - hJ'(x^k) \tag{9.13}$$

с постоянным шагом $h \in R^1$.

Принадлежность $J(x)$ к классу овражных в этом случае проявляется в необходимости применения относительно малых значений h. Попытки увеличения h вызывают потерю свойства релаксационности (монотонного убывания) последовательности $\{J(x^k)\}$ и значения $J(x^k)$ начинают резко возрастать. Если для некоторого фиксированного h (наибольшего из возможных) удалось заставить процесс (9.13) протекать без полной остановки, то можно количественно оценить величину η. Для этого процесс (9.13) продолжается до тех пор, пока отношение $\|J'(x^{k+1})\| / \|J'(x^k)\|$ не установится около некоторого значения μ. Тогда справедливо равенство

$$\eta \cong 2 / |1 - \mu|. \tag{9.14}$$

Соотношение (9.14) является основным для грубой практической оценки степени овражности минимизируемого функционала в окрестности текущей точки. Доказательство соотношения (9.14) дано в *главе 11*, посвященной градиентным схемам оптимизации.

В силу изложенного можно рекомендовать процесс оптимизации начинать с помощью метода ПГС. Если задача простая и степень овражности функционала невелика, то уже этот стартовый метод достаточно быстро приведет в малую окрестность оптимума. В противном случае будет получено значение η, что позволит правильно оценить ситуацию и выбрать наиболее рациональный алгоритм.

Другой метод оценки η сводится к вычислению матрицы Гессе функционала и решению для нее полной проблемы собственных значений. Тогда на основе непосредственной проверки выполнения неравенства (9.9) для вычисленных собственных чисел делается вывод о значении η. При этом определяется также размерность дна оврага r. Главный недостаток такого подхода заключается в существенных вычислительных трудностях принципиального характера, возникающих при определении малых собственных значений. Известно, что абсолютная погрешность $|d\lambda_i|$ представления любого собственного значения матрицы A за счет относительного искажения δ ее элементов удовлетворяет неравенству

$$|d\lambda_i| \le n\delta|\lambda_1|,$$

где $|\lambda_1| = \max_i |\lambda_i|$. Принимая $\delta = \varepsilon_м = 2^{-t}$, где $\varepsilon_м$ — относительная машинная точность (машинное эпсилон), а t — длина разрядной сетки мантиссы числа, получим оценку для абсолютных искажений собственных чисел из-за ошибок округления:

$$|d\lambda_i| \le n\varepsilon_м |\lambda_1|. \qquad (9.15)$$

Параметр $\varepsilon_м$ известен для каждого компьютера. Из последнего неравенства можно сделать следующее заключение. Если все вычисленные собственные числа матрицы $A = J''(x)$ достаточно велики, т. е. $|\lambda_i| \ge n\varepsilon_м|\lambda_1|$, то параметр η может быть вычислен непосредственно. Если же некоторые из вычисленных собственных чисел удовлетворяют неравенству $|\lambda_i| < n\varepsilon_м|\lambda_1|$, то все они должны быть отнесены к блоку малых собственных чисел, а для η имеем границу снизу:

$$\eta \ge 1 / (n\varepsilon_м).$$

Качественным признаком плохой обусловленности оптимизационной задачи может служить существенное различие в результатах оптимизации, например, методом ПГС при спуске из различных начальных точек. Получаемые результирующие точки обычно расположены достаточно далеко друг от друга и не могут рассматриваться как приближения к единственному решению или конечной совокупности решений (при наличии локальных минимумов). Описанная ситуация, как правило, означает наличие оврага, а точки остановки применяемой поисковой процедуры трактуются как элементы дна оврага Q.

9.3. Основные причины возникновения овражных целевых функционалов

Несмотря на то, что типичность овражной ситуации может считаться установленным экспериментальным фактом, определенный интерес представляет выяснение основных причин появления оврагов в задачах моделирования и численной оптимизации.

Естественная причина появления овражной ситуации может быть связана с наличием некоторых неучтенных устойчивых связей между управляемыми параметрами, определяемых внутренними законами функционирования моделируемой системы. Если уравнения связей известны, то часть управляемых параметров может быть исключена из рассмотрения. В этом случае наличие овражной ситуации целесообразно трактовать как

следствие некоторой избыточности в математическом описании объекта. С позиций приведенных в предыдущем разделе определений эти неизвестные соотношения между управляемыми параметрами можно трактовать как уравнения дна оврага.

Таким образом, поверхности уровня отдельных критериальных характеристик системы могут иметь овражный характер в силу естественных, в известном смысле не зависящих от исследователя причин.

Фактор агрегированности аргументов минимизируемого функционала. Выбор множества управляемых параметров (аргументов функционала) обычно производится по результатам анализа их влияния на основные характеристики оптимизируемой системы, а также исходя из реальных возможностей изменения этих параметров в нужных пределах. Результатом такого вполне естественного подхода является ситуация, когда вектор выходных характеристик y оптимизируемой системы в действительности зависит от s агрегатов:

$$y = \Phi(z_1, z_2, ..., z_s), \tag{9.16}$$

где

$$z_i = \varphi_i(x);\ i = \overline{1,s};\ s < n.$$

Подтверждением сказанному служит широко применяемый на практике метод декомпозиции, связанный с выделением этапов аппроксимации и реализации в процессе оптимального выбора параметров x_i *(см. разд. 8.4)*. Прямое решение задачи $J(x) \to \min$ в пространстве параметров x в данном случае связано с наличием овражной ситуации. Действительно, значение $J(x)$ мало меняется на множестве, определяемом равенствами

$$\varphi_i(x) = z_i^*,\ i = \overline{1,s}.$$

Поэтому последние уравнения фактически являются уравнениями дна оврага. Следовательно, фактор агрегированности естественным образом указывает на овражную ситуацию.

Во многих случаях соотношения (9.16) неизвестны, хотя агрегированные переменные z_i, полностью определяющие выходные параметры объекта оптимизации, по-прежнему существуют. Поэтому применение изложенного в *главе 8* метода декомпозиции, в определенной степени исключающего проблему овражности, невозможно.

Отметим также возможную причину появления овражной ситуации в результате возникновения агрегатов при оптимизации динамических систем, описываемых жесткими системами обыкновенных дифференциальных уравнений.

Рассмотрим пример.

Пусть функционирование системы описывается вектор-функцией $u(t) = [u_1(t), u_2(t)]$, являющейся решением жесткой дифференциальной системы, где

$$u_1(t) = [x_1\exp(-At) + (x_1 + x_2)\exp(-at)]t \quad (A >> a);$$
$$u_2(t) = [x_2\exp(-Bt) + (x_1 + x_2)\exp(-bt)]t \quad (B >> b). \qquad (9.17)$$

Требуется получить оптимальные значения параметров x_1, x_2 из условия наилучшего совпадения $u(t)$ с заданной вектор-функцией $\bar{u}(t)$ для $t \in$ $\in [t_0, T]$. Если предположить, что $t_0 > \tau_{\text{п.с}}$, где $\tau_{\text{п.с}}$ — длина пограничного слоя, определяющего почти полное затухание экспонент $\exp(-At)$, $\exp(-Bt)$, то из соотношений (9.17) видно, что поведение решения $u(t)$ для $t \in [t_0, T]$ будет определяться агрегатом $z = x_1 + x_2$. Если продолжать считать x_1 и x_2 независимыми параметрами, то для оптимизационной задачи

$$J(x) = [u_1(t_1) - \bar{u}_1(t_1)]^2 + [u_2(t_1) - \bar{u}_2(t_1)]^2 \to \min_x,$$

где $t_1 = 1 \in [t_0, T]$, степень овражности равна $\eta \cong 1{,}8 \times 10^8$. Мы здесь предполагали, что заданная вектор-функция $\bar{u}(t)$ определяется соотношениями (9.17) при $a = 1$; $b = 1{,}5$; $A = 20$; $B = 15$. Степень овражности рассчитывалась по формуле:

$$\eta \cong \frac{\exp(-\min(a, b))}{\exp(-\max(A, B))}.$$

Таким образом, до тех пор, пока не разработаны регулярные методы выделения агрегатов для последующей декомпозиции исходной задачи оптимизации, необходимо работать в пространстве переменных x_i в условиях отчетливо выраженной овражной ситуации.

Методы учета ограничений. Овражная ситуация может быть внесена в задачу оптимизации при учете ограничений с помощью построения обобщенного критерия оптимальности. Рассмотрим эффект овражности, возникающий при использовании методов штрафных функций и модифицированных функций Лагранжа. На почти обязательное наличие овражной ситуации в подобных случаях указывается во многих работах.

Рассмотрим в качестве примера ограничения в виде равенств

$$g_j(x) = 0, \; j = \overline{1, p}.$$

Тогда, согласно методу штрафных функций, задача сводится к минимизации вспомогательных функционалов

$$J_0(x,\sigma) = J(x) + \sigma \sum_{j=1}^{p} g_j^2(x) \to \min_x$$

с достаточно большим положительным коэффициентом σ. При этом структура расширенного критерия J_0, содержащего большой параметр σ, как правило, оказывается овражной, даже если исходный функционал $J(x)$ этим свойством не обладает.

Пример 9.2. Пусть требуется найти x_1 и x_2, минимизирующие квадратичный функционал $f(x) = x_1^2 + x_2^2$ при условии $x_1 = 2$. Задача имеет очевидное решение $x_1^* = 2$, $x_2^* = 0$. Поступим формально и составим вспомогательный функционал согласно общей рецептуре метода штрафных функций:

$$J_0(x) = x_1^2 + x_2^2 + \sigma (x_1 - 2)^2 = [(x_1 - b_1)^2 / a_1^2] + [(x_2 - b_2)^2 / a_2^2] + d,$$

где $a_1 = \sqrt{1+\sigma}$; $a_2 = 1$; $b_1 = 2\sigma / (\sigma + 1)$; $b_2 = 0$; $d = 4\sigma / (\sigma + 1)$.

Уравнение линии уровня $J_0(x) = \text{const}$ является уравнением эллипса с центром в точке (b_1, b_2) и длинами полуосей, относящихся как $a_2 / a_1 = \sqrt{1+\sigma}$. При больших значениях σ, обеспечивающих относительно точное выполнение ограничения $x_1 = 2$, линии уровня оказываются сильно вытянутыми (рис. 9.3).

Чем точнее выполняются ограничения, тем ярче выражен эффект овражности. В данном случае степень овражности равна $\eta = 1 + \sigma$. Заметим, что линии уровня исходного функционала являются сферами и явление овражности отсутствует.

На практике метод штрафных функций широко используется на начальных этапах оптимизации, при этом применяются такие возможно большие значения σ, для которых удается достигнуть относительно быстрого убывания $J_0(x)$ при достаточно точном выполнении ограничений. Для последующего уточнения решения привлекаются более тонкие стратегии, которые, как правило, оказываются и существенно более трудоемкими. Кроме того, метод штрафных функций до сих пор не имеет разумных альтернатив в некоторых критических ситуациях, характерных для реальных задач оптимизации.

Например, при наличии вырожденного минимума оптимизационной задачи, когда нарушается условие линейной независимости градиентов

$g_j'(x)$, могут потерять работоспособность все методы учета ограничений, основанные на обычной и модифицированной функциях Лагранжа, а также на линеаризации ограничений. Метод же штрафных функций в указанной ситуации применим. Он оказывается наименее чувствительным ко всем формам вырождения.

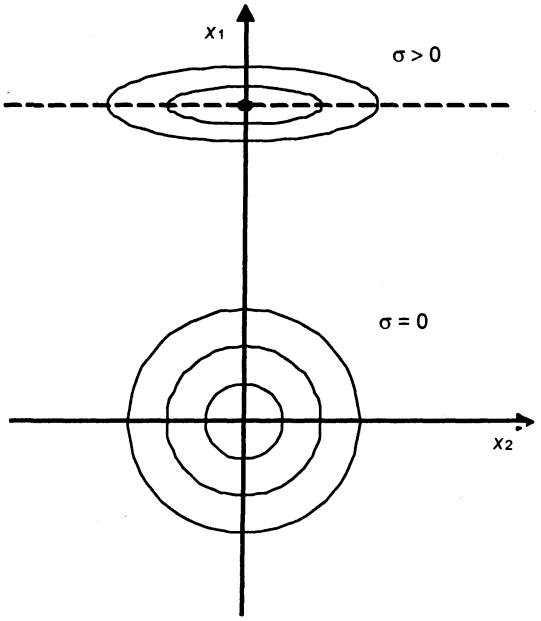


Рис. 9.3. Метод штрафных функций

Второй критической ситуацией, возникающей в практике оптимизации, является неоправданное завышение функциональных требований к объекту оптимизации, которое приводит к пустому множеству D допустимых значений управляемых параметров. В подобном случае наиболее целесообразно применять метод штрафных функций, позволяющий получить такое решение задачи $\|g(x)\|^2 \to \min$, для которого значение $J(x)$ минимально. Другие методы либо теряют смысл, либо заведомо не будут сходящимися.

В силу изложенного наличие алгоритмов оптимизации, сохраняющих работоспособность при достаточно высокой степени овражности минимизируемых функционалов, оказывается чрезвычайно желательным. Этот вывод подтверждается также тем фактом, что и при использовании

модифицированных функций Лагранжа мы сталкиваемся с овражной ситуацией, хотя и в ослабленной форме. Например, как известно, повышая скорость сходимости итераций (8.15) за счет выбора достаточно больших коэффициентов σ, мы снова приходим к плохо обусловленной задаче минимизации функционалов вида (8.14).

Объединение конфликтных выходных параметров. Учет противоречивых требований к многокритериальному объекту оптимизации с помощью единого критерия оптимальности является важнейшим фактором, обусловливающим возникновение овражной ситуации. Используемые методы построения Парето-оптимальных решений приводят к двум основным видам сверток: линейной и минимаксной (максиминной). Остановимся в качестве примера на минимаксной свертке двух частных критериев:

$$J(x) = \max\{\alpha_1 J_1(x), \alpha_2 J_2(x)\};\ \alpha_i > 0;\ \alpha_1 + \alpha_2 = 1.$$

При этом отдельные критериальные выходные параметры как функции от параметров оптимизации могут иметь монотонный, существенно неовражный характер. Однако их объединение почти неизбежно приводит к овражной ситуации. При этом крутые склоны оврага характеризуют доминирующее влияние на обобщенный критерий какого-то одного из частных критериев. Как следует из рис. 9.4, объединение критериев J_1 и J_2 приводит к образованию сложной "клювообразной" зависимости, порождающей в многомерном случае овраг с крутыми склонами.

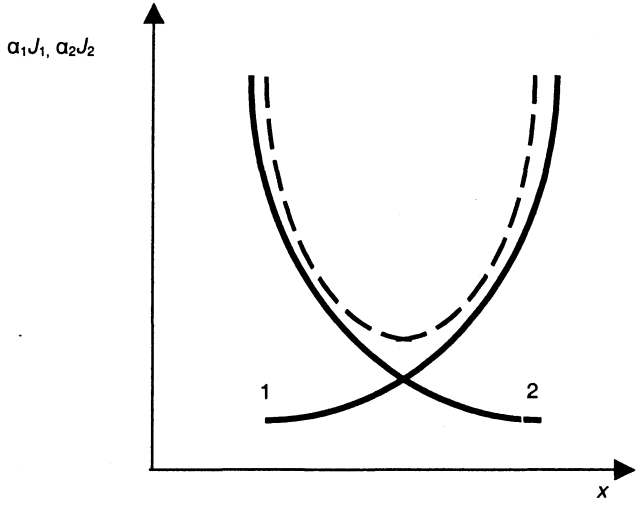


Рис. 9.4. Минимаксная свертка двух функций

Характерно, что движение по любой из поверхностей $\alpha_i J_i$ в отдельности с помощью любых методов оптимизации обычно не вызывает затруднений.

Широко применяемый на практике метод наименьших квадратов связан с минимизацией функционалов вида

$$J(x) = \sum_{i=1}^{n} \alpha_i \varphi_i^2(x) \to \min. \qquad (9.18)$$

Конструкция $J(x)$ в принципе может рассматриваться как линейная свертка многокритериальной задачи $\varphi_i^2(x) \to \min\limits_x$, $i = 1, ..., n$. На типичность овражной ситуации при решении задач (9.18) указано во многих как теоретических, так и экспериментальных исследованиях. Появление оврагов выражается, в частности, в известном факте плохой обусловленности соответствующей системы нормальных уравнений.

9.4. Некоторые стандартные схемы конечномерной оптимизации

Рассмотрим возможности некоторых традиционных методов с позиций их практической применимости.

Методы сопряженных градиентов (СГ). Методы СГ позволяют строить достаточно эффективные вычислительные процедуры и поэтому занимают важное место в арсенале средств конечномерной оптимизации. Главный недостаток методов СГ, существенно ограничивающий область их рационального использования, заключается в понижении скорости сходимости для плохо обусловленных задач оптимизации. Оценка скорости сходимости этих методов показывает, что стандартные схемы СГ сходятся по закону геометрической прогресии со знаменателем t, близким к единице:

$$t \cong 1 - 2/\sqrt{\eta},$$

где η — степень овражности минимизируемого функционала. В некоторых работах имеются указания на достаточно высокую скорость сходимости метода СГ по функционалу независимо от значения η:

$$J(x^k) - J(x^*) = L \, ||x^0 - x^*||^2 / [2(2k+1)^2] \quad (L = \text{const}). \qquad (9.19)$$

Однако оценки типа (9.19) получены в предположении строгой положительной определенности матрицы $J''(x^*)$. Кроме этого, согласно (9.19), достаточно эффективно получаются значения функционала порядка

$J(x^*)$, где x^* — точка минимума аппроксимирующего квадратичного функционала $f(x)$, что в общем случае не решает задачу.

Предположение о невыпуклости $J(x)$ вносит дополнительные трудности. Как известно, в этих условиях метод СГ эквивалентен классическому методу наискорейшего спуска со всеми вытекающими отсюда последствиями.

Эти выводы подтверждаются опытом практической работы. Кроме того, следует учесть, что основные преимущества методов СГ перед методами второго порядка ньютоновского типа в значительной степени теряются, если используются специальные экономичные методы вычисления вторых производных, обсуждаемые ниже.

Ньютоновские методы. Классическая формула метода Ньютона имеет вид

$$x^{k+1} = x^k - h_k[J''(x^k)]^{-1}J'(x^k) \quad (h_k \in R^1). \tag{9.20}$$

Предполагается, что все матрицы $J''(x^k)$ положительно определены, что гарантирует разрешимость задачи вычисления x^{k+1}. Известны различные модификации метода (9.20), построенные с целью его обобщения — на ситуации, когда матрица $J''(x^k)$ оказывается вырожденной или не обладает свойством положительной определенности. При этом вместо матрицы $J''(x^k)$ в схеме метода начинает фигурировать некоторая другая, положительно определенная, матрица G_k. Один из таких методов рассмотрен в *разд. 11.2* под названием метода Левенберга. Общий недостаток этих алгоритмов заключается в том, что изменения в схему (9.20) вносятся с единственной целью — сделать осмысленными все вычислительные операции независимо от определенности матрицы $J''(x^k)$. В то же время полезная информация о рельефе минимизируемого функционала, содержащаяся в матрице $J''(x^k)$, почти не используется.

Важный подкласс ньютоновских методов составляют методы Гаусса-Ньютона (ГН), аппроксимирующие метод Ньютона и его модификации на основе использования информации о структуре минимизируемого функционала. В классических вариантах методов ГН предполагается, что $J(x)$ имеет вид суммы квадратов. В результате вычисление матрицы $J''(x)$ с достаточной точностью сводится к вычислению только первых производных от составляющих $J(x)$ функций. Соответствующие вопросы рассмотрены в *разд. 10.3*. Обсуждавшиеся ранее недостатки ньютоновских методов сохраняются в методах ГН.

В *главах 10, 11* излагаются методы, также использующие аппроксимации матрицы $J''(x)$ на основе первых производных. В отличие от методов ГН их вычислительные схемы ориентированы на общую, невыпуклую ситуацию, что оказывается более рациональным при решении реальных задач оптимизации.

Квазиньютоновские методы (КН). КН-методы имеют структуру

$$x^{k+1} = x^k - h_k H_k J'(x^k) \quad (h_k \in R^1), \tag{9.21}$$

где H_k — $(n \times n)$-мерная матрица, пересчитываемая на каждом шаге с помощью одной из известных рекуррентных формул; h_k — длина шага, выбираемая, например, из условия минимума $J(x)$ в направлении $p^k = -H_k J'(x^k)$. Существуют и другие стратегии назначения величины h_k.

Обычно КН-методы (9.21) обладают свойством $H_n = [J''(x)]^{-1}$ при минимизации сильно выпуклых квадратичных функционалов. Начальная матрица H_0 выбирается симметричной и положительно определенной. Тогда этими же свойствами будут обладать последующие матрицы H_k. Поэтому направления p^k будут указывать в сторону убывания $J(x)$, независимо от выпуклости $J(x)$.

Таким образом, в КН-методах аппроксимация матрицы, обратной матрице Гессе, осуществляется с помощью первых производных. Это определяет высокую эффективность КН-методов при решении широкого класса задач.

Доказано, что большинство вариантов КН-методов при минимизации сильно выпуклых квадратичных функционалов приводит к одной и той же траектории поиска, вырождаясь в методы СГ. Поэтому для них характерно аналогичное замедление сходимости в овражной ситуации. Кроме этого, КН-методы построены и исследованы в расчете на выпуклые задачи; нарушение свойства выпуклости и особенно неудачное масштабирование могут приводить к вырождению матриц, а также к необходимости работы с бесконечно большими шагами, что существенно снижает эффективность этих методов.

Для решения "больших" задач оптимизации традиционные процедуры КН-методов в отличие от методов СГ не применяются, т. к. их вычислительные схемы предполагают хранение и пересчет заполненных $(n \times n)$-мерных матриц H_k, что требует больших затрат памяти[1].

Несмотря на отмеченные недостатки, КН-методы считаются достаточно эффективными и широко применяются в задачах практической оптимизации.

"Метод оврагов" Гельфанда—Цетлина. Недостатки этого эвристического метода, ориентированного на решение плохо обусловленных задач, достаточно полно изложены во многих работах. Основной недостаток заключается в полной непригодности метода для ситуации многомерно-

[1] В последнее время появились некоторые новые варианты КН-методов, ориентированные на задачи большой размерности.

го оврага. По существу это обстоятельство ограничивает возможное число аргументов минимизируемого функционала на уровне $n = 2$. Метод должен применяться для специальных классов задач, структура которых определяет наличие только одномерных оврагов.

Аналогичными недостатками обладает известный метод вращения осей Розенброка, кратко рассмотренный в *разд. 10.1.*

Методы поиска глобального оптимума. Все рассмотренные выше методы, за исключением "метода оврагов", являются локальными, т. к. с их помощью может быть найден один из локальных минимумов функционала $J(x)$. Представляет практический интерес поиск глобального минимума, т. е. такого локального минимума, где значение критерия оптимальности оказывается наименьшим.

Трудность вопроса заключается в том, что для произвольного функционала $J(x)$ задача глобальной оптимизации неразрешима с помощью вычислений $J(x)$ в любом сколь угодно большом, но конечном числе точек. Поэтому алгоритмы глобальной оптимизации должны развиваться для достаточно узких классов задач на основе имеющейся априорной информации.

В настоящее время известен один довольно общий класс критериев оптимальности, для которых обеспечивается возможность локализации глобального минимума за обозримое машинное время. Речь идет о классе функционалов, удовлетворяющих условиям Липшица:

$$|J(x') - J(x'')| \le L \, \| x' - x'' \|, \; L = \text{const}, \; L \ge 0.$$

Существующие для таких функционалов методы глобальной оптимизации достаточно подробно изложены в литературе. Недостатком этих методов является требование знания константы Липшица для всей области изменения x. Неправильное назначение L может резко уменьшить скорость сходимости метода либо привести к потере глобального минимума. В то же время задача поиска "глобального" значения константы Липшица при отсутствии дополнительной информации по трудности аналогична исходной задаче.

Реальная ситуация в области глобальной оптимизации в настоящее время расценивается как неблагоприятная. Существующие методы поиска глобального экстремума, особенно в овражной ситуации, не могут рассматриваться как исчерпывающие при решении задач достаточно высокой размерности.

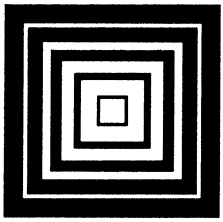
Наиболее распространенный и достаточно эффективный эвристический метод заключается в задании некоторой грубой сетки начальных точек в допустимом множестве с последующим применением методов локальной

оптимизации. Для построения таких сеток целесообразно применять широко известные в практике математического моделирования ЛПτ-последовательности, обладающие свойством равномерного заполнения многомерной области. При этом в качестве начальных точек для локальных процедур спуска могут использоваться только некоторые точки сетки, которым отвечают меньшие значения функционала.

Таким образом, в настоящее время основным инструментом практической оптимизации продолжают оставаться локальные методы.

В заключение отметим, что в некоторых случаях проблема многоэкстремальности возникает в результате определенного непонимания реальной ситуации. Регистрируемые на практике многочисленные локальные экстремумы в действительности оказываются точками остановки применяемых поисковых процедур *(см. разд. 10.1)*. Можно утверждать, что наличие многих локальных минимумов в практических задачах встречается значительно реже, чем об этом принято говорить (за исключением многочисленных специально сконструированных тестовых многоэкстремальных задач).

Глава 10



Покоординатные стратегии конечномерной оптимизации

Покоординатные стратегии оптимизации основаны на простой идее спуска (имеется в виду поиск минимума) вдоль координатных ортов. Однако в процессе оптимизации системы координат могут изменяться в зависимости от ситуации, что приводит к соответствующим изменениям в направлениях поиска. В данном разделе представлены основные алгоритмы, обычно используемые при практических вычислениях.

10.1. Методы покоординатного спуска

Пусть задан функционал $J(x) \in C^1(R^n)$. Решается задача построения минимизирующей последовательности $\{x^k\}$ для $J(x)$.

Часто используемый метод решения поставленной задачи состоит в применении покоординатной стратегии исследования пространства поиска.

Переход от вектора x^i к вектору x^{i+1} с помощью метода покоординатного спуска (ПС) происходит следующим образом: для $l = \overline{1, n}$ компонента x_l^{i+1} определяется из условия минимума:

$$J(x_1^{i+1}, x_2^{i+1}, \ldots, x_{l-1}^{i+1}, x_l^{i+1}, x_{l+1}^{i}, \ldots, x_n^{i}) = \\ = \min_{x \in R^1} J(x_1^{i+1}, x_2^{i+1}, \ldots, x_{l-1}^{i+1}, x, x_{l+1}^{i}, \ldots, x_n^{i}). \tag{10.1}$$

В овражной ситуации этот метод применим лишь в редких случаях ориентации оврагов вдоль координатных осей. Трудности применения подобных процедур проиллюстрированы на рис. 10.1.

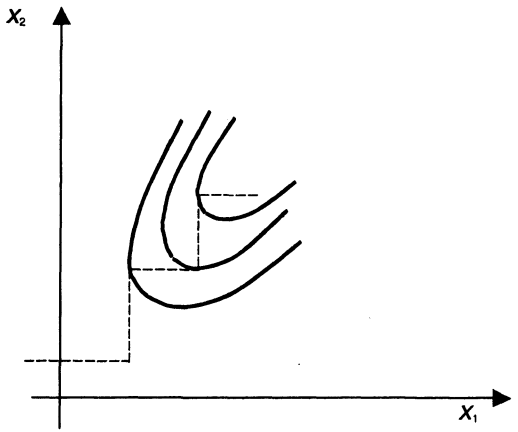


Рис. 10.1. Траектория метода покоординатного спуска

Продвижение к точке минимума становится замедленным при наличии сильно вытянутых поверхностей уровня. Обычно имеет место ситуация "заклинивания", вызываемая дискретным характером представления информации в вычислительной машине. Рассмотрим этот вопрос подробнее.

Числа, представимые в компьютере, расположены на вещественной оси дискретно (рис. 10.2).

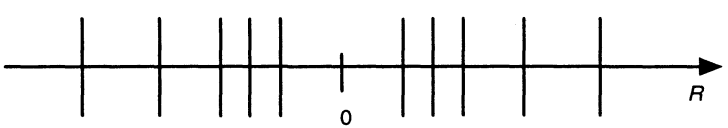


Рис. 10.2. Числа, представимые в компьютере

Аналогичным образом плоскость (x_1, x_2) в памяти компьютера аппроксимируется также конечным множеством точек, лежащих на пересечении соответствующих прямых (рис. 10.3).

В результате любая точка A плоскости R^2 может оказаться точкой "заклинивания" метода ПС, если в ее окрестности линии уровня $J(x)$ имеют овражную структуру (рис. 10.4).

Из рисунка видно, что если процесс попадает в точку A или в любую другую точку, расположенную на дне оврага, то ближайшие доступные точки B, B' и C, C' будут соответствовать большему значению функционала и убывания J не будет ни в одном из координатных направлений.

Подобная ситуация может возникнуть и на очень больших расстояниях от точки минимума. Вероятность заклинивания возрастает при использовании негладких функционалов типа максиминных критериев минимального запаса при решении систем неравенств, обсуждавшихся в *разд. 8.6*. Типичный случай представлен на рис. 10.5.

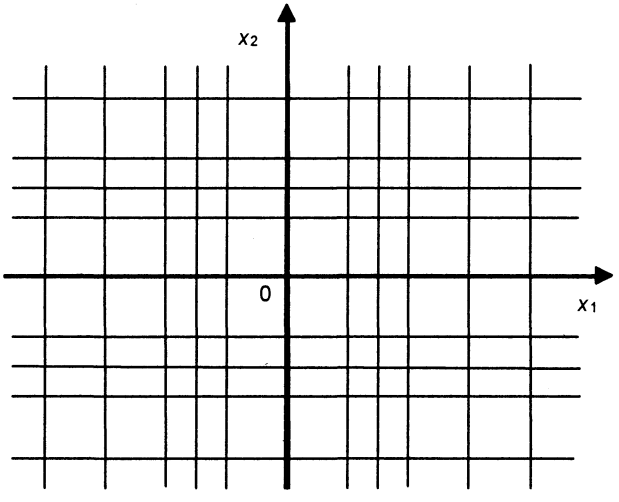


Рис. 10.3. Представление плоскости в памяти компьютера

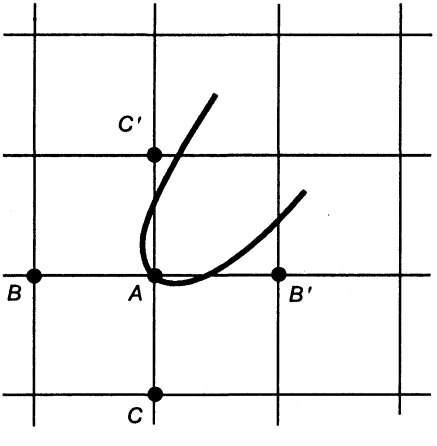


Рис. 10.4. Ситуация заклинивания

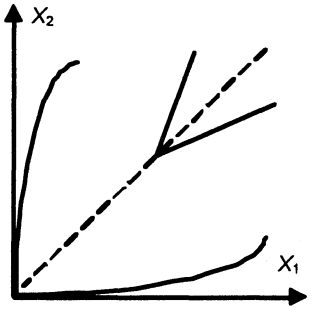


Рис. 10.5. Линии уровня негладкого функционала

Линии уровня имеют изломы, поэтому метод теряет работоспособность уже в относительно простых задачах. Такого типа ситуации, по существу, не являются следствием высокой степени овражности и могут быть устранены переходом к гладким аппроксимациям, построенным по методике, изложенной в *разд. 8.6*.

Несмотря на отмеченную низкую эффективность метода ПС в овражной ситуации, его включение в библиотеку алгоритмов целесообразно при решении любого класса оптимизационных задач, по крайней мере как стартового алгоритма, с целью получения разумного начального приближения для последующих процедур. Причина этого заключается в высокой надежности метода по отношению к различным сбойным ситуациям, а также в простоте процесса подготовки задачи к решению на компьютере. Метод имеет нулевой порядок, т. е. не требует включения в вычислительную схему информации о производных минимизируемого функционала. При реализации метода могут быть использованы стандартные способы одномерного поиска минимума типа золотого сечения, квадратичной аппроксимации и другие, известные из численного анализа. Однако это приводит к заметному и часто неоправданному усложнению алгоритма. Легко можно привести примеры, когда точный поиск минимума вдоль координатных направлений не только не обязателен, но даже вреден. Поэтому часто применяются более простые стратегии выбора шагов. Рассмотрим в качестве примера один простой вариант, оказывающийся вполне приемлемым для практических вычислений.

Задается вектор начальных шагов $h = (h_1, ..., h_n)$ продвижений из точки x в направлении ортов $e_1, e_2, ..., e_n$. Далее шаги h_i модифицируются от итерации к итерации. Если выполняется неравенство $J(x + h_i e_i) \le J(x)$, то текущая точка x заменяется на $x + h_i e_i$, а величина h_i утраивается: $h_i := 3h_i$. После этого осуществляется переход к следующему номеру i. Если

$J(x + h_i e_i) > J(x)$, то производится умножение h_i на $-0{,}5$ и также осуществляется переход к следующему координатному орту.

Таким образом, алгоритм адаптируется к конкретным условиям оптимизации в результате изменения значений и знаков шагов. Если начальные значения шагов были выбраны неудачно, то они быстро скорректируются до необходимых значений.

Указанный метод выбора координатных шагов реализован в алгоритме GZ1. Это, по-видимому, простейшая из возможных реализаций метода покоординатного спуска.

10.1.1. Алгоритм GZ1

- Шаг 1. Ввести начальную точку $x = (x_1, ..., x_n)$ и шаг s, принять $F := J(x)$.
- Шаг 2. Принять $h_i := s$, $i = \overline{1,n}$.
- Шаг 3. Принять $m := 1$.
- Шаг 4. Принять $x_m := x_m + h_m$; вычислить $F_1 = J(x)$.
- Шаг 5. Если $F_1 \le F$, принять $h_m := 3h_m$, $F := F_1$ и перейти к шагу 7; иначе — перейти к шагу 6.
- Шаг 6. Принять $x_m := x_m - h_m$, $h_m := -0{,}5h_m$.
- Шаг 7. Принять $m := m + 1$. Если $m \le n$, перейти к шагу 4; иначе — к шагу 3.

Выход из алгоритма осуществляется после достижения заданного числа N вычислений $J(x)$. Обычно программа составляется таким образом, чтобы обеспечить возможность продолжения работы с прерванного места после повторных входов в GZ1. Таким образом, в этом случае мы отказываемся от применения каких-либо внутренних критериев сходимости процесса оптимизации и обрываем его после заранее обусловленного числа шагов. При необходимости такой анализ сходимости может выполняться во внешней программе путем сравнения результатов, полученных при двух последовательных обращениях к GZ1.

При каждом новом входе в алгоритм счетчик числа вычислений функционала переводится в нулевое состояние, следовательно, разрешается еще N обращений к подпрограмме вычисления $J(x)$. Задавая различные значения N, можно регулировать частоту выходов из GZ1 во внешнюю программу для оценки получаемых результатов, например, для снятия выходных характеристик, соответствующих текущим значениям компо-

нент вектора настраиваемых параметров. Кроме этих вспомогательных действий во внешней программе должна быть организована требуемая процедура вывода окончательных результатов.

Из-за рассмотренного выше явления "заклинивания" Х. Розенброк (H. H. Rosenbrock) в 1960 г. был вынужден модифицировать процедуру ПС. Подробное описание полученного алгоритма содержится во многих работах. Основная идея заключается в организации процесса покоординатного спуска не вдоль фиксированных координатных ортов, а вдоль осей специальным образом выбираемой системы координат. При этом одна из осей должна составлять достаточно малый угол с образующей дна одномерного оврага. В результате смещения по этой оси совпадают с продвижением вдоль дна оврага в направлении точки минимума. Схема метода Розенброка сводится к трем основным этапам. Пусть x^{m-1} и x^m — две соседние точки в минимизирующей последовательности $\{x^k\}$, построенной рассматриваемым методом. Тогда переход к x^{m+1} осуществляется следующим образом:

1. Выбрать новую систему координат, первая ось которой направлена вдоль вектора x^m—x^{m-1}, а остальные дополняют ее до ортонормированного базиса ("поворот осей").

2. В новой системе координат для поиска x^{m+1} осуществить алгоритм GZ1 до выполнения условий поворота осей.

3. Возвратиться к старой системе координат и перейти к шагу 1.

Точка x^0 задается, а x^1 получается из x^0 с помощью алгоритма GZ1 в естественной системе координат.

Различные модификации метода отличаются друг от друга способом организации одномерного поиска вдоль координатных ортов, способом построения ортогонального дополнения к оси x^m—x^{m-1} (методом ортогонализации), а также выбором условия окончания процесса спуска для перехода к очередному повороту осей.

К недостаткам всех вариантов метода Розенброка следует отнести невозможность продолжения процесса оптимизации, если в качестве начальной точки выбрана точка "заклинивания" метода покоординатного спуска. Другим более существенным недостатком является то обстоятельство, что метод применим лишь к задачам оптимизации с одномерными оврагами. При наличии многомерных оврагов метод теряет эффективность, т. к. в нем не принимаются специальные меры для погружения необходимого числа координатных осей (равного размерности оврага) в пространство, образующее дно оврага. В результате целенаправленное изменение ориентации лишь одной из координатных осей не позволяет эффективно продвигаться по многомерному дну.

10.2. Методы обобщенного покоординатного спуска

Пусть решается задача $J(x) \to \min$, $x \in R^n$, $J \in C^2(R^n)$ с овражным (по определению 9.3) функционалом. Таким образом, существует некоторое подмножество $Q \subset R^n$, и при $\forall x \in Q$ для собственных чисел матрицы $J''(x)$ справедливы неравенства

$$\lambda_1 \geq \ldots \geq \lambda_{n-r} >> |\lambda_{n-r+1}| \geq \ldots \geq |\lambda_n|. \qquad (10.2)$$

Число r малых собственных значений определяет размерность оврага (дна оврага). Основной процедурой при реализации рассматриваемого далее класса методов обобщенного покоординатного спуска (ОПС) является процедура приведения матрицы $J''(x)$ к главным осям, т. е. процедура диагонализации, с последующим покоординатным спуском вдоль собственных векторов матрицы. Целесообразность такого подхода вытекает из того, что оси наиболее рациональной системы координат при минимизации квадратичных функционалов (независимо от их выпуклости) методом покоординатного спуска совпадают с собственными векторами матрицы вторых производных. Эта идея неоднократно высказывалась в литературе и даже строились соответствующие алгоритмы. Однако опубликованные результаты численных экспериментов показали низкую эффективность такого подхода. Может быть предложено и объяснение этих результатов. Оно основано на том, что при определении собственных векторов, соответствующих близким или кратным собственным значениям, возникают принципиальные вычислительные трудности. Аналогичные трудности, связанные с ограниченной точностью задания исходной информации, а также последующих вычислений, наблюдаются и при диагонализации плохо обусловленных матриц, имеющих относительно малые по модулю спектральные составляющие. Указанные обстоятельства, по-видимому, явились основным сдерживающим фактором, не позволившим внедрить изучаемые ниже методы в вычислительную практику. Однако из дальнейшего изложения следует, что неудачи при численном экспериментировании были вызваны особенностями реализации метода, рассчитанной на минимизацию выпуклых функционалов. Как показано ниже, при минимизации как выпуклых, так и невыпуклых функционалов достаточно вычислить произвольный ортонормированный базис в инвариантном подпространстве, отвечающем каждой изолированной группе собственных значений. При этом отдельные собственные векторы могут быть вычислены со значительными погрешностями. Можно показать, что отвечающие этим базисам линейные оболочки с высокой точностью совпадают с истинными подпространствами, определяемыми невозмущенной диагонализируемой матрицей.

Эти выводы в известной степени подтверждаются следующей теоремой.

Теорема 10.1. Пусть A — $(n \times n)$-мерная симметричная матрица; $\{u_i\}$ — ортонормированные собственные векторы; $\{\lambda_i\}$ — собственные значения. Тогда при $\lambda_i \neq \lambda_j$ с точностью до величин второго порядка малости имеем

$$\langle u_i + du_i, u_j\rangle = (\lambda_i - \lambda_j)^{-1}\langle (dA)u_i, u_j\rangle, \qquad (10.3)$$

где dA — возмущение матрицы A; du_i — соответствующее возмущение вектора u_i.

Доказательство. Отбрасывая величины второго порядка малости, из равенства $Au_i = \lambda_i u_i$ получим

$$(dA)\, u_i + A du_i = \lambda_i du_i + d\lambda_i u_i.$$

Отсюда, умножая скалярно на u_i, получим

$$\langle (dA)u_i, u_j\rangle + \langle A du_i, u_j\rangle = \lambda_i \langle du_i, u_j\rangle + d\lambda_i \langle u_i, u_j\rangle.$$

Из равенств

$$\langle u_i, u_j\rangle = 0;\ \langle A du_i, u_j\rangle = \lambda_j \langle du_i, u_j\rangle$$

имеем

$$(\lambda_i - \lambda_j)\langle du_i, u_j\rangle = \langle (dA)u_i, u_j\rangle,$$

откуда следует равенство (10.3). Теорема доказана.

Пусть теперь

$$M_1 = \sum_{i=1}^{n-r} a_i u_i;\ M_2 = \sum_{j=n-r+1}^{n} a_j u_j$$

есть два линейных многообразия, порожденных непересекающимися системами собственных векторов

$$\left\{u_i, i = \overline{1, n-r}\right\};\ \left\{u_j, j = \overline{n-r+1, n}\right\}$$

матрицы A. Если соответствующие множества собственных значений

$$\left\{\lambda_i, i = \overline{1, n-r}\right\};\ \left\{\lambda_j, j = \overline{n-r+1, n}\right\}$$

строго разделены: $|\lambda_i| >> |\lambda_j|$, то из равенства (10.3) следует $\langle u_i + du_i, u_j\rangle \cong 0$ при достаточно малом значении $\|dA\| / |\lambda_i|$. Это означает, что все собственные векторы под действием возмущения dA изменяются только в пределах своих линейных многообразий, сохраняя с высокой точностью свойство ортогональности к векторам из дополнительных многообразий. При этом сами вариации векторов при близких $\lambda_i \cong \lambda_j$

собственных значениях в пределах фиксированного линейного многообразия, как это следует из равенства (10.3), могут быть весьма значительными.

Изложенное позволяет в качестве модели программ, реализующих различные методы диагонализации матрицы A, использовать оператор $\Lambda(A)$, ставящий в соответствие произвольной симметричной матрице A ортогональную матрицу V, отличную, вообще говоря, от истинной матрицы U, состоящей из собственных векторов матрицы A. Оператор Λ характеризуется тем, что если спектр матрицы A разделяется на p групп

$$\lambda_i^{\sigma}(A);\ \sum_{\sigma=1}^{p} k_{\sigma} = n,\ i = \overline{1, k_{\sigma}}$$

близких между собой собственных чисел, то каждой группе σ соответствует набор столбцов $\{v_i^{\sigma}\}$ матрицы V, задающий точное линейное многообразие, порожденное соответствующими столбцами $\{u_i^{\sigma}\}$ точной матрицы U.

Рассмотрим квадратичную аппроксимацию

$$f(x) = 1/2\langle Ax, x\rangle - \langle b, x\rangle + c \qquad (10.4)$$

исходного функционала $J(x)$ в окрестности точки $x \in Q$. Допустим, что известны матрица A и ортогональная матрица U, приводящая ее к диагональному виду $U^T A U = \mathrm{diag}(\lambda_i)$. Тогда замена переменных $x = Uy$ приводит квадратичный функционал к сепарабельному виду

$$f(x) = f(Uy) = \sum_{i=1}^{n} f_i(y_i), \qquad (10.5)$$

где f_i — квадратичные функции одной переменной. Таким образом, локально достигается полная декомпозиция исходной задачи и последняя сводится к n независимым оптимизационным задачам. В результате поиск оптимального вектора y^* может осуществляться покомпонентно, ибо связь между аргументами y_i фактически исчезает. В указанной ситуации явление "заклинивания" невозможно, и все вычислительные трудности при применении покоординатных стратегий поиска оптимума, связанные с большими значениями η, полностью устраняются.

В действительности бывает задана не матрица A, а возмущенная матрица $A + dA$, где dA отражает как неопределенность задания исходной матрицы A, так и последующие ошибки округления при проведении собственно процесса диагонализации. В связи с этим вместо точной матрицы U оказывается доступной некоторая матрица $V = \Lambda(A)$. Свойства операто-

ра Λ были рассмотрены выше. Замена переменных $x = Vy$ уже не приводит функционал к виду (10.5). Для изучения создающейся ситуации важное значение имеет следующая теорема.

Теорема 10.2. Пусть собственные значения $\{\lambda_i\}$ и отвечающие им ортонормированные собственные векторы $\{u_i\}$, $i = \overline{1, n}$, некоторой симметричной матрицы A разделены произвольным образом на p групп

$$\lambda_i^\sigma;\ u_i^\sigma;\ \sum_{\sigma=1}^{p} k_\sigma = n,\ i = \overline{1, k_\sigma},\ \sigma = \overline{1, p}$$

так, что

$$u_i^\sigma \neq u_j^s;\ \sigma \neq s\ (i = \overline{1, k_\sigma},\ j = \overline{1, k_s}),$$

где λ_j^σ, u_j^σ — j-е собственное число и соответствующий собственный вектор группы σ. Тогда, если в каждом линейном многообразии M_σ размерности k_σ с базисом $\{u_i^\sigma\}$, $i = \overline{1, k_\sigma}$ задать иной ортонормированный базис $\{w_i^\sigma\}$, $i = \overline{1, k_\sigma}$, связанный с исходным базисом линейным соотношением

$$\omega_i^\sigma = \sum_{m=1}^{k_\sigma} \alpha_{mi}^\sigma u_m^\sigma,\ i = \overline{1, k_\sigma},\ \alpha_{mi}^\sigma \in R^1,$$

то существует такая матрица P перестановок столбцов, что:

1. Преобразование подобия

$$\overline{W}^T A \overline{W},\ W = (w_1^1, ..., w_{k_1}^1, ..., w_{k_p}^p),\ \overline{W} = WP$$

приводит матрицу A к блочно-диагональному виду

$$\overline{W}^T A \overline{W} = \text{diag}\ (A_1, A_2, ..., A_p);\ \overline{W}^T = \overline{W}^{-1}$$

с квадратными $(k_\sigma \times k_\sigma)$-мерными матрицами A_σ на главной диагонали.

2. Собственные значения матрицы A_σ есть

$$\lambda_i^\sigma,\ i = \overline{1, k_\sigma},\ \sigma = \overline{1, p}.$$

Доказательство. Первое утверждение проверяется непосредственно с учетом ортонормированности векторов базиса $\{u_i\}$. Для доказательства второго утверждения достаточно заметить, что вид и расположение матрицы A_m при фиксированном многообразии M_m не зависят от способа задания остальных многообразий M_σ, $\sigma \neq m$.

Поэтому, предположив, что все $k_\sigma = 1$ при $\sigma \neq m$, получим

$$\overline{W}^T A \overline{W} = \operatorname{diag}(\lambda_1^1, \lambda_1^2, \ldots, A_m, \ldots, \lambda_1^p).$$

Учитывая, что преобразование подобия не изменяет спектр матрицы, приходим к требуемому заключению. Теорема доказана.

Теорема 10.3. Пусть $V = \Lambda(A)$, тогда:

1. Замена переменных $x = Vy$ с точностью до нумерации компонент вектора y приводит функционал $f(x)$ вида (10.4) к блочно-сепарабельному виду

$$f_s(y) = f(Vy) = \sum_{\sigma=1}^{p} f_\sigma(y^\sigma), \tag{10.6}$$

где

$$y = (y_1, \ldots, y_n) = (y^1, \ldots, y^p); \; y^\sigma = (y_1^\sigma, \ldots, y_{k_\sigma}^\sigma);$$

$$f_\sigma(y^\sigma) = 1/2\langle A_\sigma y^\sigma, y^\sigma\rangle - \langle b^\sigma, y^\sigma\rangle + c_\sigma, \; c_\sigma \in R^1.$$

2. Собственные значения матрицы f_σ'' равны $\lambda_i^\sigma(A)$, $i = \overline{1, k_\sigma}$, $\sigma = \overline{1, p}$.

Доказательство. Имеем $V = \overline{W}P$, где P — некоторая матрица перестановок столбцов. Поэтому

$$f(x) = \frac{1}{2}\left\langle V^T A V y, y\right\rangle - \left\langle V^T b, y\right\rangle + c =$$

$$= \frac{1}{2}\left\langle P^T \overline{W}^T A \overline{W} P y, y\right\rangle - \left\langle P^T \overline{W}^T b, y\right\rangle + c =$$

$$= \frac{1}{2}\left\langle \overline{W}^T A \overline{W} z, z\right\rangle - \left\langle \overline{W}^T b, z\right\rangle + c, \; z \triangleq Py.$$

Согласно теореме 10.2 матрица $\overline{W}^T A \overline{W}$ имеет блочно-диагональную структуру, что и доказывает первое утверждение. Второе утверждение есть прямое следствие второго утверждения теоремы 10.2.

Следствие. Пусть собственные числа матрицы A удовлетворяют неравенствам (10.2). Тогда:

1. Замена переменных

$$x = Vy, \; V = \Lambda(A),$$

где

$$V = (v_1^1, \ldots, v_{n-r}^1, v_1^2, \ldots, v_r^2);$$

$$v_i^1 = \sum_{m=1}^{n-r} \alpha_{mi}^1 u_m, \; v_i^2 = \sum_{m=1}^{r} \alpha_{mi}^2 u_{n-r+m},$$

с точностью до нумерации компонент вектора y приводит $f(x)$ к виду

$$f_s(y) = f_1(y^1) + f_2(y^2);\ y = (y^1, y^2), \tag{10.7}$$

где

$$y^1 = (y_1, \ldots, y_{n-r});\ y^2 = (y_{n-r+1}, \ldots, y_n).$$

2. $\eta_1 << \eta,\ \eta_2 << \eta$, где η_i — показатели овражности функционалов f_i.

Таким образом, исходная оптимизационная задача локально может быть сведена к двум эквивалентным задачам с существенно меньшими значениями η_i. Представление (10.7) является аналогом идеализированного соотношения (10.5).

Если собственные числа матрицы квадратичного функционала разделяются более чем на две группы, то будет справедливо представление (10.7), содержащее соответствующее число слагаемых.

Согласно соотношению (10.7) появляется возможность независимого решения не связанных между собой оптимизационных задач для функционалов f_i с невысокими показателями овражности.

Полученные результаты носят локальный характер и справедливы в рамках квадратичной аппроксимации исходного функционала $J(x)$. Для неквадратичных функционалов приближенное выполнение соотношений типа (10.7) позволяет говорить о существенном ослаблении связей между различными группами переменных, что определяет достаточно высокую эффективность покоординатного спуска и в общем случае.

Исследование сходимости представленных выше алгоритмов в предположении точной линейной оптимизации вдоль направляющих ортов может быть основано на следующем общем подходе к исследованию алгоритмов нелинейного программирования.

Пусть решается задача

$$J(x) \to \min,\ x \in R^n,\ J(x) \in C^1(R^n).$$

Рассмотрим произвольный алгоритм A, строящий последовательность точек $\{x^k\}$, причем каждая точка x^{k+1} получается последовательной минимизацией функционала $J(x)$ вдоль направлений $d_1, \ldots, d_n$, начиная из точки x^k. Предполагается, что матрица $D = (d_1, \ldots, d_n)$ может зависеть от номера k, являясь при любом k ортогональной. Легко видеть, что метод ПС, метод Розенброка, а также методы ОПС описываются приведенной общей схемой.

Теорема 10.4. Пусть:

1. $J(x) \in C^1(R^n)$.

2. Множество решений, определяемое как $X_* = \{x \in R^n \mid J'(x) = 0\}$, не пусто.

3. Множество $\{x \in R^n \mid J(x) \le J(x^0)\}$, где x^0 — заданная начальная точка, ограничено и замкнуто в R^n.
4. Минимум функционала $J(x)$ вдоль любой прямой в R^n единственен.
5. Если $J'(x^k) = 0$, то алгоритм останавливается в x^k.

Тогда каждая предельная точка последовательности $\{x^k\}$, построенной алгоритмом A, принадлежит множеству X_*.

Доказательство здесь не приводится.

10.3. Реализация методов обобщенного покоординатного спуска

Методы вычисления производных. Для функционалов, заданных аналитически, можно воспользоваться аналитическим методом вычисления производных. Однако в действительности такой подход имеет ограниченное применение, т. к. реальные функционалы имеют достаточно сложную алгоритмическую структуру, не позволяющую воспользоваться аналитическим дифференцированием. Поэтому наиболее часто на практике применяются полуаналитические и численные методы.

Полуаналитические методы занимают промежуточное положение между аналитическими и численными методами построения производных. Эти методы основаны на использовании специальной структуры минимизируемых функционалов и в этом смысле оказываются менее универсальными, чем численные методы.

Рассмотрим характерные *(см. разд. 8.6)* для практических задач конечномерной оптимизации функционалы специального вида:

$$J_1(x) = \nu^{-1} \sum_{k=1}^{m} \varphi_k^{\nu}(x), \ \nu = 2, 3, \ldots; \tag{10.8}$$

$$J_2(x) = \nu^{-1} \sum_{k=1}^{m} \exp(-z_k(x)\nu), \ \nu = 1, 2, \ldots, \tag{10.9}$$

где φ_k, z_k — алгоритмически заданные функции вектора управляемых параметров.

Имеем следующие выражения для составляющих вектора градиента:

$$\frac{\partial J_1(x)}{\partial x_i} = \sum_{k=1}^{m} \varphi_k^{\nu-1}(x) \frac{\partial \varphi_k(x)}{\partial x_i}; \tag{10.10}$$

$$\frac{\partial J_2(x)}{\partial x_i} = -\sum_{k=1}^{m} \exp(-z_k \nu) \frac{\partial z_k(x)}{\partial x_i}, \quad i = \overline{1,n}. \tag{10.11}$$

Естественный способ получения вторых производных состоит в линеаризации функций φ_k, z_k вблизи текущей точки x':

$$\varphi_k(x) \cong \varphi_k(x') + \langle \partial \varphi_k(x')/\partial x, x - x' \rangle;$$
$$z_k(x) \cong z_k(x') + \langle \partial z_k(x')/\partial x, x - x' \rangle. \tag{10.12}$$

Используя (10.12), получаем

$$\frac{\partial^2 J_1}{\partial x_i \partial x_j} \cong (\nu - 1) \sum_{k=1}^{m} \varphi_k^{\nu-2}(x) \frac{\partial \varphi_k}{\partial x_i} \frac{\partial \varphi_k}{\partial x_j}, \nu = 2,3,\ldots; \tag{10.13}$$

$$\frac{\partial^2 J_2}{\partial x_i \partial x_j} \cong \nu \sum_{k=1}^{m} \exp(-z_k(x)\nu) \frac{\partial z_k}{\partial x_i} \frac{\partial z_k}{\partial x_j}, \quad \nu = 1,2,\ldots . \tag{10.14}$$

Из выражений (10.13), (10.14) следует, что вычисление вторых производных минимизируемого функционала может быть сведено к вычислению первых производных функций φ_k, z_k. Последняя задача для многих практических ситуаций решается относительно просто. В частности, возможен численный подход для вычисления $\partial \varphi_k/\partial x_i$, $\partial z_k/\partial x_i$, что значительно проще прямого численного вычисления матрицы Гессе (отсюда название — *полуаналитический метод*).

Дальнейшая детализация структуры решаемой задачи обычно позволяет построить эффективный алгоритм получения указанных производных первого порядка. Например, хорошо известны методы построения производных решений дифференциальных уравнений по параметрам.

Еще одним примером использования специфики задачи оптимизации для вычисления первых производных может служить известный в теории электрических цепей метод, основанный на теореме Теллегена и позволяющий находить частные производные реакций схемы по параметрам компонентов. При этом анализ чувствительности может проводиться как во временной, так и в частотной областях.

Обратимся теперь к численным методам вычисления производных. Достоинством численного подхода кроме его универсальности является низкая стоимость подготовки задачи к компьютерному моделированию. От пользователя требуется лишь написать программу для вычисления значения $J(x)$ при заданном x. Реализованные на основе численных производных методы оптимизации оказываются по существу прямыми методами (так называются методы, не использующие в своей схеме производные функционала $J(x)$). Действительно, заменяя $\partial J/\partial x_i$, например,

конечно-разностным отношением $(J(x + se_i) - J(x)) / s$, где $e_i = (0, ..., 1, ..., 0)$, мы фактически используем лишь значения J, вычисленные при определенных значениях аргумента.

Все рассмотренные в этой книге методы оптимизации строятся на основе использования локальной квадратичной модели минимизируемого функционала, получаемой из общего разложения в ряд Тейлора. Естественно поэтому при выборе формул численного дифференцирования также руководствоваться идеей локальной квадратичной аппроксимации. Исходя из этого целесообразно вместо формул с односторонними приращениями (подобно только что рассмотренным) применять двусторонние конечно-разностные аппроксимации производных, оказывающиеся точными для квадратичных функционалов. Действительно, легко проверить, что если $f(x) = 1/2\langle Ax, x\rangle - \langle b, x\rangle$, то равенство

$$\frac{\partial f}{\partial x_i} = \frac{f(x+se_i) - f(x-se_i)}{2s} \tag{10.15}$$

оказывается точным при любом $s \neq 0$.

Точно так же точным оказывается следующее представление для вторых производных квадратичного функционала:

$$\frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{1}{4s^2}(f(x+se_i+se_j) - f(x-se_i+se_j) -$$

$$- f(x+se_i-se_j) + f(x-se_i-se_j)). \tag{10.16}$$

При использовании соотношений (10.15), (10.16) вычислительные затраты характеризуются числом обращений к вычислению значений $J(x)$: для вычисления градиента — $2n$, для вычисления матрицы Гессе — $(2n^2 + 1)$ обращений, где n — размерность вектора x.

Для излагаемых далее методов оптимизации достаточно определять $J'(x)$ и $J''(x)$ с точностью до множителя, поэтому при реализации формул (10.15), (10.16) деление соответственно на $2s$ и $4s^2$ не производится. Это позволяет устранить известные трудности вычислений, если значение s оказывается относительно малым. Если необходимо работать с различными шагами s_i по отдельным компонентам x_i вектора x, то можно надлежащим образом ввести масштабы независимых переменных, оставляя без изменения стандартную программу вычисления производных.

Методы диагонализации. В качестве основной процедуры приведения симметричной матрицы к главным осям может быть выбран широко известный метод Якоби, несмотря на наличие конкурирующих и, вообще

говоря, более эффективных вычислительных схем. Этот выбор обусловлен следующими обстоятельствами. Во-первых, вычисленные методом Якоби собственные векторы всегда строго ортонормальны с точностью, определяемой точностью компьютера даже при кратных собственных числах. Последнее весьма существенно при использовании этих векторов в качестве базиса, т. к. предотвращается возможность вырождения базиса, существующая, например, в методе Пауэлла. Во-вторых, многие вычислительные схемы имеют преимущество перед методом Якоби лишь при решении частичной проблемы собственных значений. В нашем же случае всегда решается полная проблема и поэтому выигрыш во времени оказывается несущественным при существенно более сложных вычислительных схемах. В-третьих, алгоритмы, основанные на методах Якоби, часто оказываются наиболее доступными, т. к. соответствующие программы имеются в большинстве вычислительных лабораторий. И наконец, определенное влияние на выбор алгоритма оказала простота логики метода Якоби, что приводит к компактности реализующих его программ.

В задачах большой размерности по сравнению с методом Якоби более предпочтительным по объему вычислительных затрат оказывается метод, использующий преобразование Хаусхолдера для приведения матрицы к трехдиагональной форме с последующим обращением к QR-алгоритму определения собственных векторов симметричной трехдиагональной матрицы .

В методе Якоби исходная симметричная матрица A приводится к диагональному виду с помощью цепочки ортогональных преобразований вида

$$A_{k+1} = U^T_k A_k U_k;\ A_0 = A,\ k = 1, 2, ..., \qquad (10.17)$$

являющихся преобразованиями вращения. В результате надлежащего выбора последовательности $\{U_k\}$ получаем $\lim A_k = D = U^T A U$, $k \to \infty$, где $D = \mathrm{diag}(\lambda_i)$ — диагональная матрица; $U = U_0 U_1 U_2 \ldots$ — ортогональная матрица. Поскольку (10.17) есть преобразование подобия, то на диагонали матрицы D расположены собственные числа матрицы A; столбцы матрицы U есть собственные векторы матрицы A.

Элементарный шаг (10.17) процесса Якоби заключается в преобразовании посредством матрицы $U_k = \{u_{ij}\}$, отличающейся от единичной элементами $u_{pp} = u_{qq} = \cos\varphi$, $u_{pq} = -u_{qp} = \sin\varphi$. Угол вращения φ выбирается таким образом, чтобы сделать элемент a_{pq} матрицы A нулем. Вопросы сходимости различных численных схем, реализующих метод Якоби, подробно исследованы в литературе.

За основу может быть взят алгоритм *jacobi* (из известной коллекции алгоритмов Уилкинсона и Райнша, записанных на языке Алгол), реали-

зующий так называемый *частный циклический метод Якоби*. В этом методе аннулируются все элементы верхней треугольной части матрицы A с применением построчного выбора. При таком выборе индексы элементов a_{pq} пробегают последовательность значений (1, 2), (1, 3), ..., (1, n); (2, 3), (2, 4), ..., (2, n); ... ; (n – 1, n). Затем начинается новый цикл перебора элементов в том же порядке.

Эмпирическая оценка трудоемкости процесса построения матрицы $\Lambda(A)$ методом jacobi позволяет выразить необходимое время работы процессора T через размерность n решаемой задачи. Известно, что для матриц до 50-го порядка и длин машинных слов от 32 до 48 двоичных разрядов общее число циклов в процессе вращений Якоби в среднем не превышает 6—10 (под циклом понимается любая последовательность из $(n^2 - n)/2$ вращений). При этом $T = kn^3$, где коэффициент k определяется быстродействием компьютера и приблизительно равен $40t_y$, где t_y — время выполнения операции умножения.

Полученная оценка, а также опыт практической работы, показывают, что при умеренных значениях n время реализации оператора Λ для многих практических случаев невелико и сравнимо с временем однократного вычисления значения минимизируемого функционала. Упоминающаяся выше комбинация метода Хаусхолдера и QR-алгоритма оказывается приблизительно в полтора-два раза быстрее, что может иметь значение только при достаточно больших n.

10.4. Алгоритмы обобщенного покоординатного спуска

Процедура вычисления производных может быть организована пользователем, например, в соответствии с рекомендациями, приведенными в *разд. 10.3*. Однако в качестве основной процедуры для вычисления матрицы Гессе далее выбран метод конечно-разностных соотношений с переменным шагом дискретности s. Последний может определяться автоматически, например, в зависимости от продвижения в пространстве переменных x, задаваемого нормой $\|x^i - x^{i-1}\|$. Чем большее значение s используется, тем шире предполагаемая область справедливости локальной квадратичной модели исходного функционала. Наибольшая точность вычислений по формуле (10.16) применительно к квадратичным зависимостям достигается при работе с максимальным возможным s, т. к. в этом случае вклад погрешностей задания значений J в окончательный результат становится наименьшим. Поэтому чем дальше удалось продвинуться на основе построенной квадратичной аппроксимации

функционала, тем, по-видимому, большие значения s целесообразно выбирать для вычисления производных на следующем этапе поиска. Возможны и другие стратегии регулировки параметра s. Например, сама подпрограмма, реализующая метод оптимизации, может быть настроена на работу с постоянным шагом дискретности. Изменения s в этом случае осуществляются во внешней программе в зависимости от получаемых результатов.

Собственные векторы матрицы не зависят от скалярного множителя, поэтому, как уже указывалось, деление на $4s^2$ в формуле (10.16) не производится.

Полученные в результате диагонализации матрицы Гессе новые координатные орты используются далее для реализации базового алгоритма покоординатного спуска с процедурой выбора шагов продвижения по осям, применяемой в алгоритме GZ1. Переход к новым осям координат целесообразно осуществлять после того, как текущие оси "исчерпали себя" и дальнейшего существенного улучшения ситуации не ожидается. В предлагаемых алгоритмах обновление осей координат происходит после того, как по каждому из координатных направлений вслед за успешным продвижением последовала неудача — возрастание значения $J(x)$. Разумеется, могут существовать и другие критерии для определения момента изменения координатных ортов.

Укрупненное описание алгоритма, реализующего метод ОПС, сводится к следующей последовательности шагов.

10.4.1. Алгоритм SPAC1

- Шаг 1. Ввести данные: x, s.
- Шаг 2. Вычислить матрицу $B = \{b_{ij}\}$ по формулам

$$b_{ij} = J(x + se_i + se_j) - J(x - se_i + se_j) - J(x + se_i - se_j) + J(x - se_i - se_j),$$
$$e_i = (0, \ldots, 1, \ldots, 0),$$
$$i, j = 1, \ldots, n.$$

- Шаг 3. С помощью процедуры jacobi построить ортогональную матрицу U, приводящую матрицу B к диагональному виду U^TBT.
- Шаг 4. В осях $\{u_i\}$, совпадающих со столбцами матрицы U, реализовать процесс покоординатного спуска из точки x до выполнения условия поворота осей; присвоить x полученное лучшее значение, модифицировать s и перейти к шагу 2.

Процесс заканчивается по исчерпании заданного числа обращений к процедуре вычисления $J(x)$. Так же, как и в алгоритме GZ1, должна быть

предусмотрена возможность повторных входов в алгоритм и продолжения вычислений с прерванного места. Практическое применение алгоритма SPAC1 при надлежащем масштабировании и нормализации основных переменных задачи позволяет рекомендовать автоматический выбор шагов дискретности для численного дифференцирования исходя из равенства $s_{i+1} = 0{,}1\|x^{i+1} - x^i\|$.

Построенный алгоритм имеет простую структуру, однако его эффективность может быть достаточно высокой, несмотря на необходимость построения матрицы B. В некоторых случаях более эффективной оказалась модификация метода ОПС, реализованная в алгоритме SPAC2.

В алгоритме SPAC2 матрица вторых производных вычисляется в текущих осях $\{u_i\}$, без возврата к единичному исходному базису $\{e_i\}$. В результате информация о последнем используемом базисе не теряется, что позволяет иногда сократить трудоемкость решения задачи.

10.4.2. Алгоритм SPAC2

- Шаг 1. Ввести исходные данные: x, s.
- Шаг 2. Принять $U = E$, где E — единичная матрица; в качестве координатных векторов взять столбцы $\{u_i\}$ матрицы U.
- Шаг 3. Построить матрицу $B = \{b_{ij}\}$ по формулам

$$b_{ij} = J(x + su_i + su_j) - J(x - su_i + su_j) - J(x + su_i - su_j) + J(x - su_i - su_j),$$

$$i, j = 1, \ldots, n.$$

- Шаг 4. Принять $U := UT$, где T — ортогональная матрица, приводящая матрицу B к диагональному виду T^TBT.
- Шаг 5. В осях $\{u_i\}$ реализовать процесс покоординатного спуска из точки x до выполнения условия поворота осей; присвоить x лучшее полученное значение. Модифицировать s и перейти к шагу 3.

Окончание процесса и выбор шагов дискретности такие же, как и в алгоритме SPAC1.

Дадим необходимые пояснения к алгоритму, касающиеся построения матрицы U на шаге 4.

Выбор в качестве координатных направлений столбцов $\{u_i\}$ некоторой ортогональной матрицы U очевидно эквивалентен замене переменных $x = Uy$. В этом случае изменения компонент y_i вектора y приводят в исходном пространстве к смещениям вдоль одноименных векторов u_i.

Функционал $J(Uy) = I(y)$ как функция y имеет матрицу Гессе вида $I''(y) = U^T J''U$. Действительно, $I'(y) = U^T J'(x)$; $I''(y) = \partial[U^T J'(Uy)]/\partial y = U^T J''(x)U$.

Таким образом, если необходимо работать с функционалом, имеющим матрицу Гессе вида $U^T J''U$, то для этого достаточно в качестве базиса взять столбцы матрицы U. Если требуется изменить матрицу Гессе и привести ее к виду $T^T(U^T J''U)T = U_1^T J U_1$, где T — новая ортогональная матрица; $U_1 = UT$, то в качестве базисных векторов достаточно выбрать столбцы матрицы $U_1 = UT$. Указанная процедура и реализована на шаге 4 сформулированного алгоритма.

Остановимся на некоторых принципиальных отличиях алгоритма SPAC2 от SPAC1. Во-первых, если минимизируемый функционал близок к квадратичному и матрица Гессе меняется относительно мало, то ее повторное построение в осях $\{u_i\}$ опять приведет к диагональной матрице и поэтому число якобиевых циклов вращений при последующей диагонализации вновь полученной матрицы J'' будет сведено к минимуму. В результате матричная поправка T к матрице U, вычисляемая на шаге 4, оказывается близкой к единичной матрице. В алгоритме SPAC1 в указанных условиях весь процесс диагонализации должен каждый раз целиком повторяться; то, что оси $\{u_i\}$ фактически меняются мало при переходе к следующей точке x^i, никак не используется.

Во-вторых, если на каком-то этапе поиска минимума шаг дискретности s оказывается меньше, чем, скажем, $\min_i |\varepsilon_M x_i|$, где ε_M — машинное эпсилон, то это приведет к получению нулевой матрицы на этапе построения матрицы B. Подпрограмма, реализующая алгоритм jacobi, выполнена таким образом, что при этом в качестве диагонализирующей матрицы T получим единичную матрицу. Если указанная ситуация возникает в процессе работы SPAC2, то на шаге 4 не произойдет изменения матрицы U и процесс покоординатного спуска будет продолжен в текущих осях координат. Таким образом, будет сохраняться возможность медленного продвижения, скорость которого затем может вновь возрасти. При использовании алгоритма SPAC1 нулевая матрица B автоматически приведет к получению матрицы $U = E$, что эквивалентно возврату к исходному единичному базису.

В результате вероятность "заклинивания" на участках медленного продвижения для SPAC1 оказывается существенно большей, чем для SPAC2.

Отмеченные особенности SPAC2 не позволяют, однако, полностью отказаться от применения SPAC1. Это объясняется, во-первых, тем, что трудоемкость вычисления матрицы B в SPAC2 оказывается заметно выше, чем в SPAC1, т. к. вектор x варьируется в направлениях u_i, отличных, во-

обще говоря, от единичных векторов e_i. Кроме этого, важное отличие заключается в том, что алгоритм SPAC2 предполагает почти единственный возможный способ построения матриц Гессе, основанный на соотношениях (10.16). При применении же SPAC1 пригоден любой из описанных в *разд. 10.3* методов. Последнее обстоятельство часто оказывается решающим при выборе метода.

10.5. Реализация методов обобщенного покоординатного спуска на основе рекуррентных алгоритмов оценивания

В процессе работы алгоритма ОПС получаются последовательность векторов $\{x^k\}$ и отвечающая им последовательность значений минимизируемого функционала $\{J_k\}$. В указанных алгоритмах используются только те точки x^k, которые приводят к монотонному убыванию $J(x)$, а "неудачные" точки отбрасываются и далее никак не участвуют в процессе поиска. Ниже показано, что соответствующая информация может быть эффективно использована для построения квадратичной модели функционала $J(x)$ с целью последующего вычисления собственных векторов матрицы $J''(x)$ в качестве новых направлений поиска.

Рассмотрим последовательности $\{x^k\}$, $\{J_k\}$, получаемые методом ОПС в текущей системе координат. В этом случае имеются в виду "полные" последовательности, включающие в себя как удачные, так и неудачные шаги. Опишем процедуру, позволяющую по этой информации вычислить матрицу Гессе аппроксимирующего квадратичного функционала. Задачу квадратичной аппроксимации будем решать на основе метода наименьших квадратов:

$$F_N(c) = \sum_{i=1}^{N} [J(x^i) - f(x^i, c)]^2 \to \min_{c \in R^m}; \qquad (10.18)$$

$$x^i = (x_1^i, x_2^i, ..., x_n^i);\ c = (c_1, c_2, ..., c_m);$$

$$m = n^2/2 + 3n/2 + 1;\ N \geq m.$$

Здесь $f(x, c)$ означает аппроксимирующий функционал с неизвестными коэффициентами c_i:

$$f(x, c) = c_1 x_1^2 + c_2 x_1 x_2 + c_3 x_1 x_3 + \ldots + c_n x_1 x_n + c_{n+1} x_2^2 + c_{n+2} x_2 x_3 + \ldots$$
$$\ldots + c_{2n-1} x_2 x_n + \ldots + c_{n^2/2+n} x_n^2 + c_{n^2/2+n+1} x_1 + \ldots + c_{m-1} x_n + c_m. \qquad (10.19)$$

Полагая $y(x) = (x_1^2, x_1x_2, ..., x_1x_n, x_2^2, ..., x_n^2, x_1, ..., x_n, 1)$, представим (10.19) в виде $f(x, c) = \langle c, y\rangle = y^Tc$, что позволяет говорить о линейной регрессионной задаче оценки параметров $c_1, ..., c_m$. Система нормальных уравнений, отвечающая функционалу метода наименьших квадратов (10.18), имеет вид

$$Y_N Y_N^T c = \left(\sum_{k=1}^{N} y_k y_k^T\right) c = Y_N J^N = \sum_{k=1}^{N} J_k y_k, \qquad (10.20)$$

где $Y_N = (y_1, y_2, ..., y_N)$ — $(m \times N)$-мерная матрица; $y_k = y(x^k)$; $J^N = (J_1, J_2, ..., J_N)^T$; $J_i = J(x_i)$. Заметим, что поскольку квадратичный функционал $F_N(c) \geq 0$ и $F''_N(c) = YY^T$, матрица YY^T неотрицательно определена. Можно рассчитать коэффициенты квадратичной модели непосредственно из системы линейных алгебраических уравнений (10.20). Однако более рациональным оказывается другой подход, позволяющий избежать решения линейных систем.

Известно, что для псевдообратной матрицы $(\cdot)^+$ выполняется соотношение

$$(Y^T)^+ = \lim(\delta E + YY^T)^{-1} Y \quad (\delta \to 0, \delta > 0).$$

Отсюда следует, что вместо системы уравнений

$$Y_r Y_r^T c = Y_r J^r \qquad (10.21)$$

можно рассматривать систему

$$(\delta E + Y_r Y_r^T) c = Y_r J^r, \qquad (10.22)$$

решение которой при $\delta \to 0$ сходится к решению (10.21) с минимальной нормой среди всех векторов, минимизирующих величину

$$\|Y_r^T c - J^r\|^2 \to \min_c,$$

что при $r = N$ совпадает с выражением (10.18). Ниже на основе известных рекуррентных алгоритмов оценивания, соответствующих случаю $\delta = 0$, будут построены методы решения регуляризованных систем (10.22) при конечных (малых) значениях параметра δ. В этом случае δ играет роль параметра регуляризации, обеспечивая устойчивость получаемых решений к ошибкам округления. При $\delta = 0$ решение системы (10.21) может наталкиваться на существенные вычислительные трудности, т. к. при $N < m$ матрицы $Y_N Y_N^T$ будут вырождены, а при $N > m$ — плохо обусловлены.

Введем обозначение

$$P_r^{-1} = \sum y_k y_k^T + \delta E = P_{r-1}^{-1} + y_r y_r^T \qquad (10.23)$$

Используя так называемую вторую лемму об обращении матриц (эквивалентную формуле Шермана—Моррисона—Вудбери)

$$(K^{-1} + B^T R^{-1} B)^{-1} = K - KB^T (BKB^T + R)^{-1} BK,$$

получаем рекуррентное соотношение

$$P_r = (P_{r-1}^{-1} + y_r y_r^T)^{-1} = P_{r-1} - P_{r-1} y_r (y_r^T P_{r-1} y_r + 1)^{-1} y_r^T P_{r-1}, \quad (r = \overline{1, N}). \quad (10.24)$$

Поскольку, очевидно, $P_1^{-1} = y_1 y_1^T + \delta E$, из (10.23) следует, что необходимо принять $P_0^{-1} = \delta E$ или $P_0 = \delta^{-1} E$.

Матрица P_0^{-1} симметрична и положительно определена. Предположим, что P_{r-1}^{-1} симметрична и положительно определена, и покажем, что P_r^{-1} обладает этими же свойствами. Последнее, согласно теории симметричных возмущений немедленно следует из представления (10.23), т. к. матрица $y_r y_r^T$ симметрична и неотрицательно определена. Поэтому по принципу индукции все матрицы P_r^{-1}, а вместе с ними и P_r, будут симметричны и положительно определены. Таким образом, все обратные матрицы в соотношении (10.24) существуют.

Построим рекуррентное соотношение для определения оценок c_i^r. Уравнение (10.22) имеет вид

$$P_r^{-1} c^r = \sum_{k=1}^{r} J_k y_k, \quad (10.25)$$

откуда

$$P_r^{-1} c^r = \sum_{k=1}^{r} J_k y_k + y_r J_r = P_{r-1}^{-1} c^{r-1} + y_r J_r. \quad (10.26)$$

Здесь c^r означает оценку вектора c по r вычислениям функционала $J(x)$. Прибавляя и вычитая $y_r y_r^T c^{r-1}$ в правой части (10.26), получаем

$$P_r^{-1} c^r = P_r^{-1} c^{r-1} + y_r (J_r - y_r^T c^{r-1}). \quad (10.27)$$

Из (10.27) имеем окончательное выражение

$$c^r = c^{r-1} + P_r y_r (J_r - y_r^T c^{r-1}). \quad (10.28)$$

По формуле (10.28) может быть вычислена новая оценка c^r вектора параметров при условии, что известна предыдущая оценка c^{r-1}, матрица P_r и вновь полученные значения J_r, $y_r = y(x^r)$.

Последовательный метод оценки параметров квадратичной модели функционала J позволяет заменить процедуру обращения матрицы полной нормальной системы уравнений (10.22) операцией вычисления скаляра, обратного заданному $y_r^T P_{r-1} y_r + 1$, выполняемой на каждом шаге итерационного процесса (10.24).

Непосредственно из построения уравнений видно, что результат c^r для $r = N$, полученный согласно выражениям (10.24), (10.28), приводит к оценке, которая получается из решения полной системы (10.22). При этом нужно принять $c^0 = 0$. Последнее следует из соотношений (10.25), (10.28), записанных для $r = 1$.

Действительно, согласно (10.25), имеем оценку $c^1 = P_1 J_1 y_1$, полученную в результате решения системы (10.22). Из выражения (10.28) следует $c^1 = c^0 + P_1 y_1 (J_1 - y_1^T c^0)$. Поэтому для совпадения обеих оценок c^1, а значит, и последующих оценок, необходимо и достаточно принять $c^0 = 0$.

На основе вычисленных оценок c_i^N, $i = \overline{1,(n^2+n)/2}$, задающих аппроксимацию матрицы $J''(x)$, может быть реализована процедура ОПС. При этом возможны различные стратегии применения изложенного общего подхода, конкретизирующие способ выбора числа "измерений" y_r, J_r, участвующих в коррекции текущей оценки, а также самих точек y_r. Целесообразно после каждого поворота осей обновлять процесс и вновь начинать процедуру построения аппроксимации. Такая тактика позволяет не учитывать "устаревшие" значения J, расположенные достаточно далеко от текущей точки.

Изложенная процедура обладает определенными свойствами адаптируемости по локализации окрестности текущей точки, в которой строится аппроксимирующая квадратичная модель. Действительно, если норма результирующего вектора продвижения в текущих осях достаточно велика, то исходный функционал заменяется квадратичным в достаточно широкой области пространства поиска. Если же оси выбраны неудачно и продвижение мало, то автоматически на формирование квадратичной модели оказывают влияние только близко лежащие точки и тем самым область предполагаемой "квадратичности" функционала $J(x)$ сжимается.

Опыт применения такого типа алгоритмов для целей оптимизации в настоящее время недостаточен. Однако можно ожидать, что в некоторых случаях будут возникать трудности, связанные с рациональным выбором δ, определяющим, в частности, погрешности промежуточных вычислений и их влияние на результат. В этом смысле подбор δ необходимо начинать с относительно больших значений, позволяющих с достаточной точностью получать "малые разности больших величин" при реализации соотношений (10.24). Кроме этого, необходимо учитывать, что реализация методов ОПС на основе рекуррентного оценивания коэффициентов квадратичной модели увеличивает необходимый объем памяти компьютера.

10.6. Тестирование алгоритмов оптимизации

Основным подходом к сравнению алгоритмов с точки зрения их последующих приложений является тестирование, т. е. проверка алгоритмов при решении определенного класса тестовых задач оптимизации, построенных таким образом, чтобы внести в процесс оптимизации характерные трудности, возникающие при решении реальных задач.

Однако выбор класса тестовых функционалов, подлежащих минимизации, еще не решает проблемы. Остается актуальным вопрос об оценке вычислительных затрат алгоритма при решении конкретной тестовой задачи. Наиболее естественен подход, связанный с анализом реального машинного времени, затраченного на поиск минимума функционала с заданной точностью. Однако на практике целесообразно применять метод, основанный на оценке трудоемкости в специальных единицах — Горнерах. В один Горнер (1 Г) оценивается трудоемкость операции однократного вычисления значения минимизируемого функционала. Эффективность алгоритма оптимизации, затратившего на получение результата с заданной точностью наименьшее число Горнеров, считается наивысшей.

Использование метода оценки трудоемкости в Горнерах базируется на экспериментально оправданном предположении, что наиболее трудоемкой при решении реальных задач является процедура вычисления $J(x)$. В задачах оптимизации реальных систем эта процедура связана с решением соответствующей задачи анализа, что требует существенных вычислительных затрат.

При решении тестовых задач, имеющих простую алгоритмическую структуру, основное время расходуется на реализацию собственно стратегии оптимизации, а не на вычисление значений $J(x)$. Поэтому оценки сравнительной эффективности на основе анализа времени работы машины имеют ограниченную ценность.

Весьма часто работа алгоритмов проверяется на задачах минимизации следующих тестовых функций.

1. Квадратичная функция простой структуры

   $F_1(x) = (x_1 - x_2)^2 + (x_1 + x_2 - 10)^2 / 9;\ x^0 = (0,1);\ x^* = (5;5);\ F_1(x^*) = 0.$

   Приведенная функция моделирует ситуацию, когда никаких вычислительных трудностей не возникает и оказываются пригодными почти все методы. Результаты минимизации F_1 могут быть использованы при отладке соответствующих программ.

2. Функция Розенброка

$$F_2 = 100(x_1^2 x_2)^2 + (1 - x_1)^2;$$

$$x^0 = (-1{,}2;\ 1);\ x^* = (1;\ 1);\ F_2(x^*) = 0.$$

Эта функция является часто предлагаемым тестом, используемым во многих работах по методам конечномерной оптимизации. Линии уровня F_2 имеют ярко выраженную овражную структуру с криволинейным дном оврага, расположенным вдоль параболы $x_2 = x_1^2$.

Хотя степень овражности в этом случае не очень высока ($\eta \cong 2500$), работа многих алгоритмов затруднена из-за значительного уменьшения скорости сходимости. В частности, непригодными оказываются алгоритмы покоординатного спуска, а также классические методы спуска по антиградиенту.

3. "Асимметричная долина":

$$F_3 = [(x_1 - 3) / 100]^2 - (x_2 - x_1) + \exp[20(x_2 - x_1)];$$

$$x^0 = (0;\ -1);\ x^* = (3;\ 2{,}850214);\ F_3(x^*) = 0{,}199786.$$

Минимизация F_3 сопряжена с известными трудностями, т. к. это пример "неквадратичной" задачи.

Большинство алгоритмов позволяют достаточно быстро получить значения функции около 0,2, но соответствующее значение x оказывается неудовлетворительным.

4. Функция Пауэлла

$$F_4 = (x_1 + 10x_2^2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4;$$

$$x^0 = (3;\ -1;\ 0;\ 1);\ x^* = (0;\ 0;\ 0;\ 0);\ F_4(x^*) = 0.$$

В точке минимума x^* матрица F''_4 вырождена, а в окрестности этой точки — плохо обусловлена, что затрудняет применение методов ньютоновского типа. Дополнительные экспериментальные данные, касающиеся применения различных методов минимизации к этой функции, содержатся в многочисленных публикациях по методам нелинейного программирования.

5. Функция Зангвилла

$$F_5 = (x_1 - x_2 + x_3)^2 + (x_2 - x_1 + x_3)^2 + (x_1 + x_2 - x_3)^2;$$

$$x^0 = (0{,}5;\ 1;\ 0{,}5);\ x^* = (0;\ 0;\ 0);\ F_5(x^*) = 0.$$

Функция является примером, для которого неприменим первоначальный вариант известного метода Пауэлла, обычно имеющего достаточно высокую эффективность.

6. Пример, связанный с оценкой экспериментальных данных методом наименьших квадратов:

$$F_6 = 10^4 \sum_{i=1}^{7} \left[\left(\frac{x_1^2 + x_2^2 a_i + x_3^2 a_i^2}{1 + x_4^2 a_i} - b_i \right) / b_i \right]^2,$$

$x^0 = (2{,}7; 90; 1500; 10)$; $x^* = (2{,}714; 140{,}4; 1707; 31{,}51)$; $F_6^{\text{я}} = 318{,}57$.

Значения констант a_i, b_i являются компонентами следующих заданных векторов:

$$a = 10^{-3}\,(0; 0{,}428; 1; 1{,}61; 2{,}09; 3{,}48; 5{,}25);$$

$$b = (7{,}391; 11{,}18; 16{,}44; 16{,}20; 22{,}2; 24{,}02; 31{,}32).$$

Эта функция приводит к определенным трудностям при работе многих алгоритмов минимизации. Обычно наблюдается хорошая сходимость по функционалу, а по аргументу процесс оказывается замедленным.

7. Квадратичный функционал с высокой степенью овражности $\eta = 10^{12}$:

$$F_7 = 1/2 \sum_{i=1}^{4} \lambda_i \langle x, u_i \rangle^2 - \langle b, x \rangle; \quad b = (1; 1; 1; 1).$$

Собственные числа матрицы Гессе F''_7 равны: $\lambda_1 = 10^8$, $\lambda_2 = \lambda_4 = 10^{-4}$, $\lambda_3 = 10^6$. Собственные векторы u_i есть:

$$u_1 = (1/\sqrt{3})(1; -1; 1; 0);$$

$$u_2 = (1/\sqrt{6})(1; 2; 1; 0);$$

$$u_3 = (1/\sqrt{3})(1; 0; -1; 1);$$

$$u_4 = (1/\sqrt{6})(1; 0; -1; -2);$$

$x^0 = (0; 0; 0; 0)$; $F_7(x^*) \cong -16\,667$; $x^* \cong (3333{,}3; 13\,333; 10\,000; 6666{,}7)$.

Точные значения компонент вектора x^* задаются выражением

$$x^* = (u_1; u_2; u_3; u_4)(\lambda_1^{-1}\langle b, u_1\rangle; \lambda_2^{-1}\langle b, u_2\rangle; \lambda_3^{-1}\langle b, u_3\rangle; \lambda_4^{-1}\langle b, u_4\rangle)^T \cong$$

$$\cong (u_1; u_2; u_3; u_4) \times (0; 10^4\langle b, u_2\rangle; 0; 10^4\langle b, u_4\rangle)^T.$$

Отсюда видно, что компоненты x^* в основном определяются малыми собственными числами λ_2, λ_4, и уже небольшая погрешность в их представлении приводит к большой ошибке в результате. Необходимо отметить, что при написании тестовой программы, осуществляющей вычисление значений F_7, следует использовать приведенное ранее представление функции в виде суммы. Применение для этой цели

обычного выражения квадратичного функционала $F_7(x) = 1/2\langle Ax, x\rangle - (b, x)$, содержащего в явном виде матрицу $A = F_7''$, недопустимо, т. к. из-за ограниченной точности представления элементов a_{ij} матрицы в памяти компьютера информация о малых собственных числах λ_2, λ_4 может теряться на фоне больших λ_1, λ_3.

Указанное обстоятельство приводит к резкой потере эффективности методов ньютоновского типа, основанных на существенном использовании информации о малых собственных числах при явном представлении аппроксимации матриц Гессе минимизируемого функционала.

Другая особенность функции F_7 заключается в наличии двухмерного оврага, дно которого совпадает с многообразием вида $\{x^* + \alpha u_2 + \beta u_4\}$. Это вносит дополнительные трудности для методов, рассчитанных на минимизацию функционалов с одномерными оврагами.

При оценке результатов тестирования необходимо иметь в виду, что некоторые из приводимых в литературе методов оптимизации используют процедуры одномерной, как правило, квадратичной экстраполяции, дающие им односторонние преимущества при решении задач минимизации функций двух ($n = 2$) переменных. Автор не является сторонником такого подхода, т. к. эти преимущества немедленно теряются для более реальных ситуаций, когда $n > 2$ и увеличивается вероятность появления многомерных оврагов.

В заключение отметим, что при решении реальных задач всегда возникает проблема определения момента окончания вычислений. Основная трудность заключается в том, что точно установить степень приближения решения к искомой оптимальной точке почти невозможно, и поэтому необходимо применять какие-либо косвенные критерии сходимости, позволяющие обрывать вычислительный процесс на основе анализа доступной информации. Далее предполагается, что в качестве такой информации используются свойства последовательностей $\{x^k\}$, $\{J_k\}$, генерируемых методом оптимизации. Требуется определить момент завершения вычислений, который, вообще говоря, может просто являться моментом перехода к другой более перспективной вычислительной процедуре. При этом обычно отказываются от часто рекомендуемого анализа последовательности градиентов $\{J'_k\}$, главным образом, по причине ее малой информативности. При решении плохо обусловленных задач градиент может быть достаточно малым в некоторых точках дна оврага, однако процесс оптимизации в указанных условиях целесообразно продолжать. При этом изменения $\|\Delta x\|$ и $|\Delta J|$ могут быть весьма значительными. Кроме этого, градиент прямо зависит от выбранных масштабов измерения J.

Наиболее надежное решение проблемы окончания процесса оптимизации может быть получено в интерактивном (диалоговом) режиме работы с программой, когда пользователь может всесторонне оценить получаемое решение из соображений, вообще говоря, жестко не связанных с соответствующим значением критерия J.

Например, при решении задачи аппроксимации некоторой функции методами оптимизации может быть проанализирована сама функция, соответствующая параметрам x^k, а не только величина $J(x^k)$, характеризующая усредненную ошибку аппроксимации. Такой наиболее естественный способ оценки результатов позволяет исходить из действительных целей оптимизации, которые подчас не удается адекватно отобразить при формировании скалярного или даже векторного критерия оптимальности.

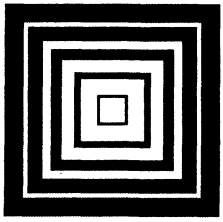
Изложенное не исключает необходимости построения критериев остановки, позволяющих автоматически завершать вычисления или сигнализировать о целесообразности перехода от медленно сходящихся оптимизирующих процедур к процедурам, более приспособленным для решения данной задачи. Важность такого типа критериев очевидна, в частности, при разработке сложных оптимизирующих программных систем, обладающих средствами автоматического перехода от одного метода оптимизации к другому с целью выбора наиболее перспективного алгоритма.

Разумной альтернативой всем применяемым критериям остановки является условие, сочетающее требования как на относительную, так и на абсолютную точность: $|J_{k+1} - J_k| \le \varepsilon_1 |J_k| + \varepsilon_2$, $\varepsilon_1 > \varepsilon_M$. Здесь ε_1, ε_2 — заданные значения относительной и абсолютной точности локализации оптимума по функционалу.

Как было показано ранее, наиболее часто встречающиеся в приложениях функционалы характеризуются медленным изменением при продвижении по дну оврага. Поэтому в данном случае необходимо осуществлять проверку по аргументу $\left|x_i^{k+1} - x_i^k\right| \le \varepsilon_1 \left|x_i^k\right| + \varepsilon_2$, $i = \overline{1,n}$ или использовать комбинированное условие.

В сложных случаях целесообразно прекращать работу программы только после исчерпания выделенного ресурса времени работы, независимо от свойств последовательностей $\{x^k\}$, $\{J_k\}$, $\{J'_k\}$.

Глава 11



Градиентные стратегии конечномерной оптимизации

11.1. Общая схема градиентных методов. Понятие функции релаксации

Классические градиентные схемы, основанные на применении антиградиентов в качестве направлений спуска, непосредственно не приспособлены для минимизации овражных функционалов и по этой причине не позволяют получить решение сколько-нибудь сложной задачи конечномерной оптимизации. Однако далее будет показано, что на их основе могут быть построены методы, обладающие существенно лучшими показателями сходимости в овражной ситуации независимо от характера выпуклости минимизируемых функционалов. С учетом представленных в *главе 9* моделей явления овражности эффективность алгоритмов оценивается по свойствам специально вводимых для этих целей функций релаксации, полностью определяющих локальные характеристики методов.

Пусть решается задача

$$J(x) \to \min_{x}; \quad x \in R^n; \quad J \in C^2(R^n). \tag{11.1}$$

Рассмотрим класс матричных градиентных методов вида

$$x^{k+1} = x^k - H_k(A_k, h_k)J'(x^k) \quad (h_k \in R^1), \tag{11.2}$$

где $A_k = J''(x^k)$, H_k — матричная функция A_k. Предполагается, что в некоторой ζ_k-окрестности $\{x \in R^n \mid \|x - x^k\| \le \zeta_k\}$ точки x^k функционал $J(x)$ достаточно точно аппроксимируется квадратичным функционалом

$$f(x)=1/2\,\langle A_k x, x\rangle - \langle b_k, x\rangle + c_k, \tag{11.3}$$

где A_k — симметричная, не обязательно положительно определенная матрица. Без существенного ограничения общности можно считать, что

$b_k = 0$, $c_k = 0$. Действительно, принимая $\det A_k \neq 0$, $x = x^* + z$, где $x^* = A_k^{-1} b_k$, получим представление

$$f_1(z) = f(x^* + z) = 1/2 \langle A_k z, z \rangle + \bar{c}_k. \qquad (11.4)$$

При этом константу $\bar{c}_k = c_k - 1/2 \langle A_k x^*, x^* \rangle$ можно не учитывать как не влияющую на процесс оптимизации.

Формула (11.2) обладает свойством инвариантности относительно смещения начала координат. Будучи записанной для $f(x)$, она преобразуется в аналогичное соотношение для $f_1(z)$. Для $f(x)$ имеем:

$$x^{k+1} = x^k - H_k(A_k x^k - b_k). \qquad (11.5)$$

Принимая $z^k = x^k - x^*$, получаем из (11.5): $z^{k+1} = z^k - H_k A_k z^k$. А это есть запись метода (11.2) для функционала f_1.

Ставится задача построения таких матричных функций H_k, чтобы выполнялись условия релаксационности процесса $f(x^{k+1}) < f(x^k)$ и чтобы при этом норма $\|x^{k+1} - x^k\|$ ограничивалась сверху только параметром ζ_k, характеризующим область справедливости локальной квадратичной модели (11.3). В настоящее время ситуация такова, что при высокой степени овражности $\eta(x^k)$ для большинства классических схем поиска имеем $\|x^{k+1} - x^k\| << \zeta_k$, что в результате приводит к медленной сходимости.

Определение 11.1

Скалярная функция $R_h(\lambda) = 1 - H(\lambda, h)\lambda$; $\lambda, h \in R^1$ называется *функцией релаксации* метода (11.2), а ее значения $R_h(\lambda_i)$ на спектре матрицы A_k — *множителями релаксации* для точки x^k.

В некоторых случаях для сокращения записей индекс h в выражении функции релаксации будет опускаться.

Здесь $H(\lambda, h)$ означает скалярную зависимость, отвечающую матричной функции $H(A_k, h_k)$ в представлении (11.2).

Напомним, что если A — симметричная матрица и

$$A = T\mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)T^T,$$

где T — ортогональная матрица, столбцы которой есть собственные векторы матрицы A, то

$$F(A) = T\mathrm{diag}[F(\lambda_1), F(\lambda_2), \ldots, F(\lambda_n)]T^T.$$

Матричная функция имеет смысл, если скалярная функция $F(\lambda)$ определена в точках $\lambda_1, \lambda_2, \ldots, \lambda_n$.

Теорема 11.1. Для выполнения условия

$$f(x^{k+1}) \le f(x^k) \tag{11.6}$$

при $\forall x^k \in R^n$ необходимо и достаточно, чтобы

$$|R(\lambda_i)| \ge 1 \ (\lambda_i < 0); \ |R(\lambda_i)| \le 1 \ (\lambda_i > 0) \tag{11.7}$$

для всех собственных чисел λ_i, $i = \overline{1,n}$ матрицы A_k.

Доказательство. Пусть $\{u_i\}$ — ортонормированный базис, составленный из собственных векторов матрицы A_k. Тогда, разлагая x^k по векторам базиса, получаем

$$x^k = \sum_{i=1}^{n} \xi_{i,k} u_i; \ x^{k+1} = x^k - H_k(A_k, h_k) f'(x_k) =$$

$$= (E - H_k A_k) x^k = \sum_{i=1}^{n} \xi_{i,k} (1 - H_k(\lambda_i, h_k)\lambda_i) u_i =$$

$$= \sum_{i=1}^{n} \xi_{i,k} R(\lambda_i) u_i.$$

Из сравнения выражений

$$f(x^k) = \frac{1}{2} \sum_{i=1}^{n} \xi^2_{i,k} \lambda_i;$$

$$f(x^{k+1}) = \frac{1}{2} \sum_{i=1}^{n} \xi^2_{i,k+1} \lambda_i = \frac{1}{2} \sum_{i=1}^{n} \xi^2_{i,k} \lambda_i R^2(\lambda_i) \tag{11.8}$$

следует, что при выполнении (11.7) каждое слагаемое суммы в представлении $f(x^k)$ не возрастает. Достаточность доказана. Докажем необходимость. Пусть существует такой индекс $i = i_0$, для которого $\lambda_{i_0} < 0, \left|R(\lambda_{i_0})\right| < 1$. Выберем $x^k = u_{i_0}$. Тогда

$$f(x^k) = 0{,}5\lambda_{i_0} < f(x^{k+1}) = 0{,}5\lambda_{i_0} R^2,$$

что противоречит условию релаксационности (11.6). Аналогично рассматривается второе неравенство (11.7). Теорема доказана.

Замечания

1. Для строгого выполнения неравенства (11.6) необходимо и достаточно кроме выполнения условий (11.7) потребовать, чтобы существовал такой индекс $i = i_0$, для которого $\xi_{i_0 k} \ne 0$, и соответствующее неравенство (11.7) было строгим.

2. Выражения (11.8) позволяют оценить скорость убывания функционала f в зависимости от "запаса", с которым выполняются неравенства (11.7). Действительно, обозначим через λ_i^+, λ_i^- положительные и отрицательные собственные числа матрицы A_k. Эти же индексы присвоим соответствующим собственным векторам. Суммирование по соответствующим i будем обозначать Σ^+, Σ^-. Тогда

$$2|f(x^k) - f(x^{k+1})| = \Sigma^+\xi_{i,k}^2\lambda_i^+(1 - R^2(\lambda_i^+)) + \Sigma^-\xi_{i,k}^2|\lambda_i^-|(R^2(\lambda_i^-) - 1).$$

Из полученного выражения следует, что наибольшее подавление будут испытывать слагаемые, для которых значение множителя релаксации наиболее существенно отличается от единицы (при выполнении условий (11.7)).

Далее будут рассматриваться в основном зависимости $R_h(\lambda)$, обладающие свойством

$$R_h(\lambda) \to 1 \quad (h \to 0). \tag{11.9}$$

В этом случае из равенства

$$\|x^{k+1} - x^k\| = \sum_{i=1}^{n} \xi_{i,k}^2 [R_{h_k}(\lambda_i) - 1]^2 \tag{11.10}$$

следует, что для $\forall \zeta_k \in R^1$ всегда можно выбрать такой h_k, что $\|x^{k+1} - x^k\| \le \zeta_k$. Таким образом, с помощью параметра h_k можно регулировать норму вектора продвижения в пространстве управляемых параметров с целью предотвращения выхода из области справедливости локальной квадратичной модели (11.3).

Иногда для ограничения нормы (11.10) параметр h может вводиться в схему оптимизации как множитель в правой части (11.2):

$$x^{k+1}(h) = x^k - hH_kJ'(x^k), \quad h \in [0, 1]. \tag{11.11}$$

При этом $\|x^{k+1}(h) - x^k\| = h\|x^{k+1}(1) - x^k\|$, а второе равенство (11.8) трансформируется к виду

$$f(x^{k+1}) = 1/2\sum_{i=1}^{n} \xi_{i,k}^2 \lambda_i \overline{R}^2(\lambda_i),$$

где $\overline{R}(\lambda_i) = (1-h) + hR(\lambda_i)$. Таким образом, новые множители релаксации $\overline{R}(\lambda_i)$ принимают промежуточные значения между 1 и $R(\lambda_i)$, что и требуется для обеспечения свойства релаксационности, определяемого требованиями (11.7).

Введенное понятие функции релаксации позволяет с единых позиций оценить локальные свойства различных градиентных схем поиска. Удобство такого подхода заключается также в возможности использования наглядных геометрических представлений.

Подобно областям устойчивости методов численного интегрирования обыкновенных дифференциальных уравнений, построенных на основе тестового (линейного скалярного) уравнения, можно для любого метода (11.2) построить функцию релаксации, характеризующую область его релаксационности в множестве собственных чисел. При этом роль тестового функционала играет квадратичная зависимость (11.3). Требуемый характер функции релаксации представлен на рис. 11.1; заштрихована запрещенная область, где условия релаксационности (11.7) не выполняются.

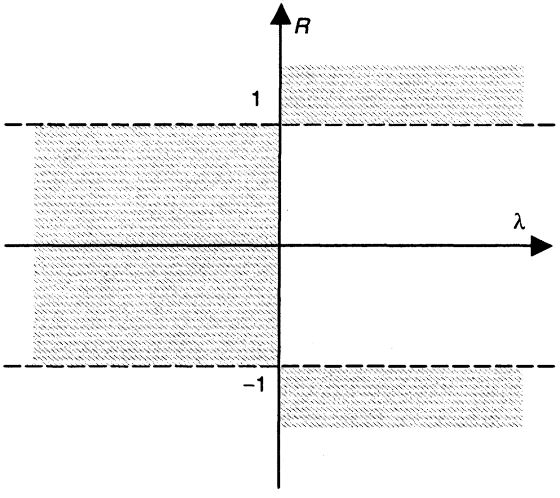


Рис. 11.1. Области релаксационности градиентного метода

Очень важное свойство функций релаксации заключается в возможности использования соответствующих представлений для синтеза новых процедур из класса (11.2), обладающих некоторыми желательными свойствами при решении конкретных классов задач оптимизации.

11.2. Классические градиентные схемы

Рассмотрим некоторые конкретные методы (11.2) и отвечающие им функции релаксации.

Простой градиентный спуск (ПГС). Формула метода ПГС имеет вид

$$x^{k+1} = x^k - hJ'(x^k) \quad (h = \text{const}). \tag{11.12}$$

Соответствующая функция релаксации

$$R(\lambda) = 1 - h\lambda \tag{11.13}$$

линейна и представлена на рис. 11.2.

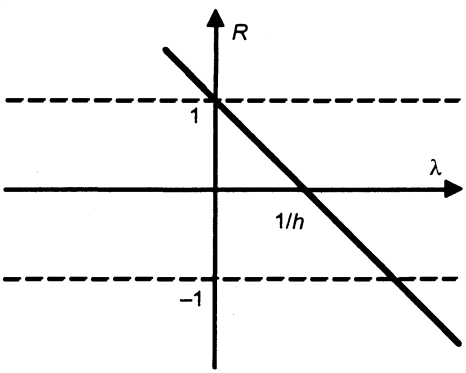


Рис. 11.2. Функция релаксации метода простого градиентного спуска

Пусть собственные значения матрицы A_k расположены в замкнутом интервале $[m, M]$, причем $0 < m \le M$, так что $\eta(x) = M / m >> 1$. В этом случае условие (11.9), очевидно, выполняется, а неравенства (11.7) сводятся к требованию

$$|R(\lambda_i)| \le 1,\ i = \overline{1,n} \text{ или } |1 - h\lambda_i| \le 1,\ i = \overline{1,n}. \tag{11.14}$$

Из неравенства (11.14) следует $h \le 2 / M$, $R(m) = 1 - hm \approx 1$. Точка пересечения прямой $R(\lambda)$ с осью абсцисс есть точка $\lambda = 1 / h$ и, чтобы при $\lambda \in [m, M]$ зависимость $R(\lambda)$ находилась в разрешенной области, необходимо выполнение неравенства $1/h \ge M / 2$ (рис. 11.2). При этом ординаты функции релаксации характеризуют соответствующие множители релаксации, которые в окрестности $\lambda = m$ будут тем ближе к единице, чем больше отношение M / m. Будем считать, что для собственных чисел матрицы A_k выполняются неравенства $\lambda_1 \ge ... \ge \lambda_{n-r} \ge \sigma|\lambda_{n-r+1}| \ge ... \ge \sigma|\lambda_n|$, $\sigma >> 1$, характерные для овражной ситуации. Тогда для точки $x^k \in Q$, где Q — дно оврага, имеем согласно (11.10)

$$\left\|x^{k+1}-x^k\right\| \cong 4\sum_{i=n-r+1}^{n}\xi_{i,k}^2\left(\lambda_i/\lambda_1\right)^2 \le 4\sigma^{-2}\left\|x^k\right\|^2,$$

что может быть существенно меньше ζ_k.

В результате соответствующие малым собственным значениям из окрестности $\lambda = 0$ слагаемые в выражении (11.8) почти не будут убывать, а продвижение будет сильно замедленным. Это и определяет низкую эффективность метода (11.12).

В области $\lambda < 0$ функция (11.13) удовлетворяет условиям релаксации при любом значении параметра h. Параметр h в методе ПГС выбирается из условия монотонного убывания функционала на каждом шаге итерационного процесса. При отсутствии убывания величина h уменьшается до восстановления релаксационности процесса. Существуют различные стратегии выбора h, однако при больших η все эти методы, включая и метод наискорейшего спуска

$$x^{k+1} = \arg\min_{h\ge 0} J[x^k - hJ'(x^k)],$$

малоэффективны даже при минимизации сильно выпуклых функционалов. Так же, как и в методе покоординатного спуска, здесь возможна ситуация "заклинивания", представленная на рис. 11.3.

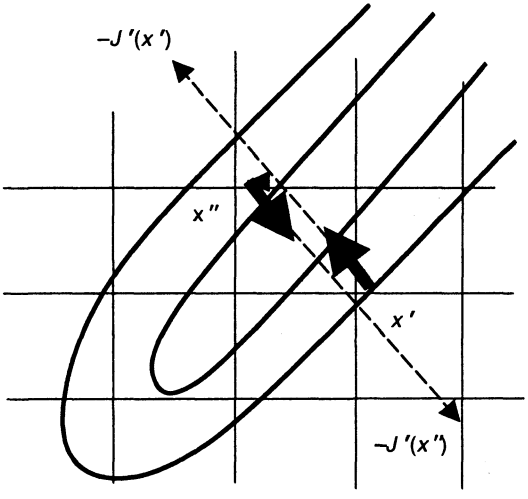


Рис. 11.3. "Заклинивание" метода простого градиентного спуска: $J(x'') > J(x')$

Как указывалось в *разд. 9.2*, метод ПГС представляет определенный интерес как средство оценки локальной степени овражности в окрестности точки замедления алгоритма. Выведем соответствующие соотношения.

Пусть замедление метода ПГС при минимизации некоторого функционала $J(x)$ произошло в окрестности некоторой точки x^0. Тогда можно предположить, что достаточно длинный отрезок последовательности $\{x^k\}$, построенный из точки x^c, будет оставаться в области $\|x - x^0\| \le \zeta_0$ и для $J(x)$ справедлива квадратичная аппроксимация

$$f(x) = 1/2\,\langle Ax, x\rangle - \langle b, x\rangle. \tag{11.15}$$

Метод (11.12) для аппроксимации (11.15) примет вид

$$x^{k+1} = x^k - h(Ax^k - b) = Bx^k + g, \tag{11.16}$$

где $B = E - hA$; $g = hb$.

Записывая (11.16) для двух последовательных номеров k и вычитая полученные равенства, приходим к соотношению

$$y^k = x^{k+1} - x^k = B(x^k - x^{k-1}) = B^k(x^1 - x^0) = B^k y^0. \tag{11.17}$$

Согласно степенному методу определения максимального собственного числа симметричной матрицы в результате проведения процесса (11.17) может быть получена оценка максимального собственного числа матрицы B.

$$\left\|y^{k+1}\right\| / \left\|y^k\right\| \to \max_i \left|\lambda_i(B)\right| \qquad (k \to \infty). \tag{11.18}$$

Пусть шаг h в итерационном процессе (11.16) выбирается из условия релаксационности, тогда можно заключить, что $h \cong 2/M$. Пусть $\lambda_i(A) \in$ $\in [-m, M]$, $M > m > 0$. В результате имеем $\max|\lambda_i(B)| = 1 + mh$. Следовательно, для достаточно больших k

$$\left\|y^{k+1}\right\| / \left\|y^k\right\| = \left\|J'(x^{k+1})\right\| / \left\|J'(x^k)\right\| = \mu_k \approx 1 + mh. \tag{11.19}$$

Приходим к требуемой оценке степени овражности функционала $J(x)$ в окрестности точки x^0:

$$\eta(x^0) = M/m \cong 2/(\mu_k - 1).$$

Рассуждая аналогично для случая $\lambda_i(A) \in 0[m,M]$, получаем вместо (11.19) равенство $\mu_k \approx 1 - mh < 1$ и соответствующую оценку $\eta(x^0) \approx 2/(1 - \mu_k)$. Общая оценка может быть записана в виде $\eta(x^0) \approx 2/|1 - \mu_k|$, причем, сравнивая μ_k с единицей, можно установить характер выпуклости $J(x)$ в окрестности точки x^0, что дает дополнительную полезную информацию.

Метод Ньютона основан на построении квадратичной аппроксимации функционала $J(x)$ в окрестности текущей точки x^k:

$$J(x) \cong f(x) = J(x^k) + \langle x - x^k, J'(x^k)\rangle + 1/2\langle J''(x^k)(x - x^k), x - x^k\rangle.$$

В качестве x^{k+1} выбирается точка, удовлетворяющая уравнению $f'(x) = 0$. В результате приходим к формуле

$$x^{k+1} = x^k - h_k A_k^{-1} J'(x^k); \quad A_k = J''(x^k). \tag{11.20}$$

С помощью дополнительно введенного параметра h_k, как указывалось ранее, осуществляется регулировка нормы вектора продвижения $\|x^{k+1} - x^k\|$.

Таким образом, по построению метод Ньютона является оптимальным для квадратичных функционалов при условии положительной определенности матрицы A_k. Минимум сильно выпуклого квадратичного функционала находится за один шаг при $h_k = 1$.

Главный недостаток метода заключается в следующем. Если функционал $J(x)$ не является выпуклым в окрестности точки x^k, то это может послужить причиной расходимости. Действительно, допустим, что функционал $J(x)$ аппроксимируется невыпуклым квадратичным функционалом. В этом случае ньютоновское направление $p^k = A_k^{-1} J'(x^k)$ указывает на точку x', удовлетворяющую условию $f'(x') = 0$. Такой точкой может оказаться, например, седловая точка, а не точка минимума, как в выпуклой ситуации. В результате направление p^k будет указывать "наверх", а не "вниз" (рис. 11.4).

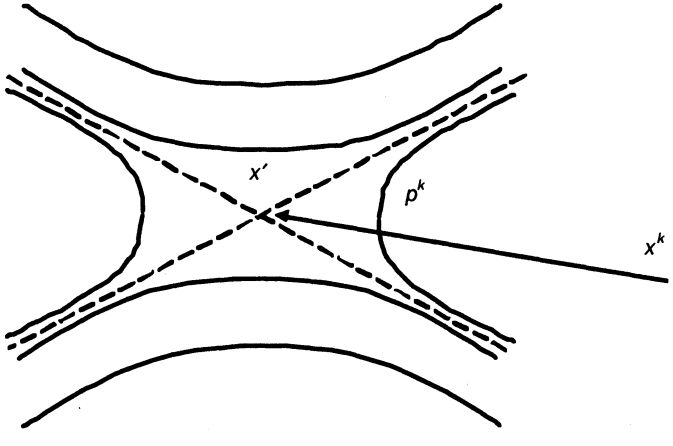


Рис. 11.4. Расходимость метода Ньютона

Заметим, что полная потеря работоспособности алгоритма Ньютона в невыпуклых экстремальных задачах происходит независимо от наличия или отсутствия овражной ситуации. Такая особенность методов ньютоновского типа существенно ограничивает область их практического применения, т. к. уже простейшие задачи оптимизации могут приводить к невыпуклым критериям.

На языке функций релаксации это обстоятельство проявляется в попадании графика функции релаксации метода (11.20)

$$R(\lambda) = 1 - H(\lambda, h_k)\lambda = 1 - h_k$$

в запрещенную область, где не выполняются неравенства (11.7). Действительно, при $h_k = 1$, что соответствует классическому варианту метода Ньютона без регулировки шага, имеем $R(\lambda) \equiv 0$ при $\forall\lambda \neq 0$. И аналогично при любых значениях h_k прямая релаксации $1 - h_k$ параллельна оси абсцисс и захватывает запрещенную область либо при $\lambda > 0$, либо при $\lambda < 0$. Положение ее при $h = 0$ соответствует остановке процесса. В указанных условиях эффективный выбор h_k оказывается затруднительным.

Аналогичным недостатком обладают многие из аппроксимирующих метод Ньютона квазиньютоновских алгоритмов и совпадающих с ними при минимизации квадратичных функционалов методов сопряженных направлений. Все эти методы по эффективности приближаются к методу ПГС, если функционал $J(x)$ в окрестности x^k не является выпуклым.

Традиционное возражение против метода Ньютона, связанное с необходимостью выполнения операции вычисления вторых производных целевого функционала, для реальных задач оптимизации оказывается менее существенным.

Метод Левенберга. Если известно, что собственные значения матрицы A_k расположены в интервале $[-m, M]$, где $M >> m$, то можно построить метод, имеющий нелинейную функцию релаксации (рис. 11.5)

$$R(\lambda) = h' / (h' + \lambda) \quad (h' > 0,\ h = 1 / h'), \tag{11.21}$$

удовлетворяющую требованиям (11.7) при $\forall\lambda \in [-m, M]$, если $h' > m$.

Соответствующий метод предложен Левенбергом и имеет функцию

$$H(\lambda, h) = [1 - R(\lambda)] / \lambda = (h' + \lambda)^{-1}.$$

Схема метода (11.2) с указанной функцией H имеет вид

$$x^{k+1} = x^k - [h'E + J''(x^k)]^{-1} J'(x^k). \tag{11.22}$$

Скаляр h' на каждом шаге итерационного процесса подбирается так, чтобы матрица $h'E + J''(x^k)$ была положительно определена и чтобы $\left\| x^{k+1} - x^k \right\| \leq c_k$, где c_k может меняться от итерации к итерации.

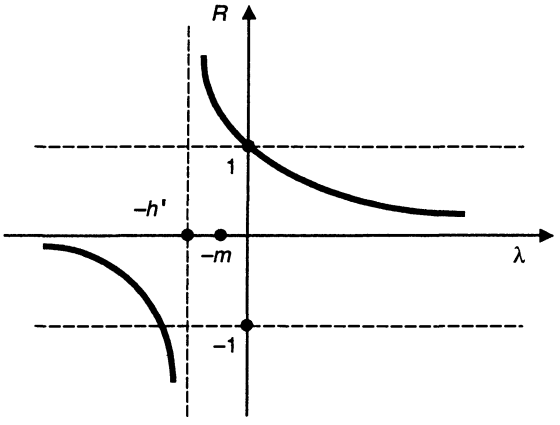


Рис. 11.5. Функция релаксации метода Левенберга

Из последнего выражения следует, что мы имеем, по существу, классическую регуляризованную форму метода Ньютона с параметром регуляризации h'. Данная форма метода Ньютона применялась Левенбергом для решения задач методом наименьших квадратов. Существенно позже данный метод применялся Маркуардтом для решения общих задач нелинейной оптимизации и иногда встречается его описание в литературе под названием "метод Маркуардта".

Реализация метода (11.22) сводится к решению на каждом шаге линейной алгебраической системы

$$\left[h'E + J''(x^k)\right]\Delta x^k = -J'(x^k) \quad \left(\Delta x^k \triangleq x^{k+1} - x^k\right). \tag{11.23}$$

Главный недостаток метода заключается в необходимости достаточно точного подбора параметра h', что сопряжено с известными вычислительными трудностями. Значение m, как правило, неизвестно и не может быть вычислено с приемлемой точностью. При этом оценка для m существенно ухудшается при возрастании размерности n. Лучшее, что обычно можно сделать на практике, это принять

$$h' \geq \max\left\{\varepsilon_M n \left\|J_k''\right\|, \left|\min\lambda_i(J_k'')\right|\right\}. \tag{11.24}$$

Правая часть неравенства (11.24) обусловлена тем, что, как уже указывалось в *разд. 9.2*, абсолютная погрешность представления любого собственного числа матрицы J''_k ввиду ограниченности разрядной сетки равна

$$|d\lambda_i| \leq n\lambda_1\varepsilon_M \approx n\,\|J''_k\|\varepsilon_M.$$

При невыполнении условия $h' > m$ система (11.23) может оказаться вырожденной. Кроме этого, слева от точки $\lambda = -h'$ функция релаксации быстро входит в запрещенную область и метод может стать расходящимся. Попытки использования алгоритмического способа более точной локализации h' приводят к необходимости многократного решения плохо обусловленной линейной системы (11.23) с различными пробными значениями h'.

Легко видеть, что число обусловленности матрицы $h'E + J''(x^k)$ может превышать $\mathrm{cond}[J''(x^k)]$. Действительно, потребуем, например, чтобы $R(-m) = 10$ для обеспечения заданной скорости убывания $J(x)$. Определим необходимое значение параметра h'. Имеем $1 / (1 - hm) = 10$ или $h' = 1 / h = m / 0{,}9$. В этом случае $\lambda_{\min}(h'E + J''_k) = -m + m / 0{,}9 = m / 9 > 0$. Принимая $\lambda_{\max}(h'E + J''_k) \approx \lambda_{\max}(J''_k)$, получаем, что $\mathrm{cond}(h'E + J''_k) \approx \approx 9\lambda_{\max}(J''_k) / |\lambda_{\min}(J''_k)$.

При выборе заведомо больших значений h', что реализуется, например, когда определяющим в (11.24) является первое выражение в скобках, имеем $m << h'$ и $|R(-m)| \approx 1$, что приводит к медленной сходимости. Ограничение h' снизу не позволяет также уменьшить до желаемого значения множитель релаксации для $\lambda > 0$.

Эти трудности возрастают при аппроксимации производных конечными разностями, т. к. при малых значениях $J'(x^k)$ для точек x^k, расположенных на дне оврага, приходим к необходимости получать компоненты вектора градиента как малые разности относительно больших величин порядка $J(x^k)$. В результате компоненты вектора Δx^k будут находиться с большими относительными погрешностями порядка $\eta(x^k) \mid \varepsilon_M J(x^k) \mid / \|J'(x^k)\|$. Коэффициент овражности η в этом случае играет роль своеобразного коэффициента усиления погрешности. Для метода Ньютона справедливо аналогичное замечание. В то же время для метода ПГС точность задания $J'(x^k)$ может оказаться достаточной для правильного указания направления убывания $J(x)$.

Методы, рассмотренные далее в этом разделе, также используют в своей схеме производные, которые вычисляются с теми же погрешностями. Однако их структура такова, что в соответствующих вычислительных схемах не используются окончательные результаты решения плохо обусловленных линейных алгебраических систем. Например, в методах с экспоненциальной релаксацией на участках выпуклости $J(x)$ решение эквивалентной линейной системы выполняется итеративно, причем каждая итерация имеет "физический" смысл, что дает возможность непрерывно контролировать точность вычислений и прерывать процесс, когда накопленная ошибка начинает превышать допустимый уровень.

Несмотря на отмеченные недостатки, метод (11.23) часто оказывается достаточно эффективным и его присутствие в библиотеке методов оптимизации следует признать весьма желательным. Дополнительное положительное свойство соответствующих алгоритмов связано с возможностью их обобщения на решение "больших" задач оптимизации, рассмотренных в *разд. 11.5*.

11.3. Методы с экспоненциальной функцией релаксации

Изучаемые в этом разделе методы могут быть построены по принципу "непрерывных методов" из общей теории численного интегрирования жестких систем обыкновенных дифференциальных уравнений с помощью специальных "системных" алгоритмов численного интегрирования [32]. В настоящей работе использован другой подход к построению алгоритмов, основанный на понятии функции релаксации.

Исходя из изложенных ранее требований к функциям релаксации, естественно рассмотреть экспоненциальную зависимость вида (рис. 11.6)

$$R(\lambda) = \exp(-\lambda h) \quad (h > 0), \tag{11.25}$$

для которой условие (11.7) выполняется при любых значениях параметра h.

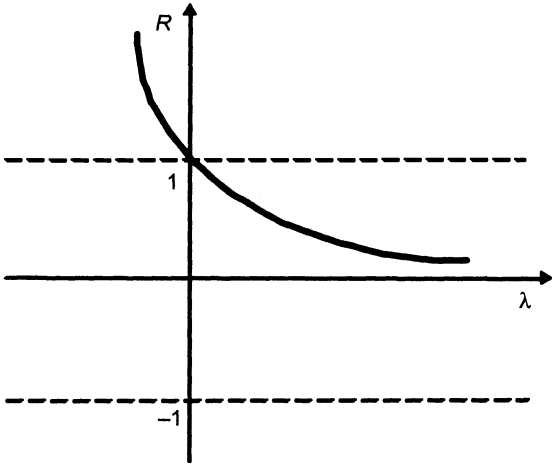


Рис. 11.6. Экспоненциальная функция релаксации

Кроме того, реализуется предельное соотношение (11.9), что позволяет эффективно регулировать норму вектора продвижения независимо от расположения спектральных составляющих матрицы A_k на вещественной оси λ.

Функция (11.25) обобщает ранее рассмотренные функции релаксации и является в определенном смысле оптимальной. Действительно, разлагая экспоненту в ряд Тейлора и ограничиваясь двумя первыми членами разложения, получаем $\exp(-\lambda h) = 1 / \exp(\lambda h) \approx 1 / (1 + \lambda h)$, что совпадает с выражением (11.21). И аналогично, принимая $\exp(-\lambda h) \approx 1 - \lambda h$, приходим к зависимости (11.13). Для достаточно больших значений параметра h имеем $\exp(-\lambda h) \approx 0$ при любых $\lambda \geq m > 0$, что позволяет говорить о вырождении метода в классический метод Ньютона без регулировки шага.

Для построения метода в виде (11.2) необходимо определить соответствующую функцию $H(\lambda, h)$. Имеем $\lambda H(\lambda, h) = 1 - R(\lambda) = 1 - \exp(-\lambda h)$.

Принимая $\lambda \neq 0$, получаем

$$H(\lambda, h) = \lambda^{-1}\left[1 - \exp(-\lambda h)\right] = \int_0^h \exp(-\lambda\tau)d\tau. \qquad (11.26)$$

Доопределяя $H(0, h)$ из условия непрерывности, получаем $H(0, h) = h$. В результате схема метода с экспоненциальной релаксацией (ЭР) примет вид

$$x^{k+1} = x^k - H(A_k, h_k)J'(x^k), \quad A_k = J''(x^k); \qquad (11.27)$$

$$H(A, h) = \int_0^h \exp(-A\tau)d\tau. \qquad (11.28)$$

Параметр h_k определяется равенством

$$h_k = \arg\min_{h \geq 0} J\left[x^k - H(A_k, h)J'(x^k)\right], \qquad (11.29)$$

однако возможны и другие способы выбора h_k.

Принципиальная схема метода ЭР была получена исходя из анализа локальной квадратичной модели минимизируемого функционала. Представляет интерес выяснение возможностей метода в глобальном смысле, без учета предположений о квадратичной структуре $J(x)$.

Можно доказать, что алгоритм (11.27), (11.28) сходится почти при тех же ограничениях на минимизируемый функционал, что и метод наискорейшего спуска, имея в определенных условиях существенно более высокую скорость сходимости.

Следующая теорема устанавливает факт сходимости метода ЭР для достаточно широкого класса невыпуклых функционалов в предположении

достижимости точки минимума (второе условие) и отсутствия точек локальных минимумов (третье условие).

Теорема 11.2. Пусть:

1. $J(x) \in C^2(R^n)$.
2. Множество $X_* = \{x^* \mid J(x^*) = \min J(x)\}$ непусто.
3. Для любого $\varepsilon > 0$ найдется такое $\delta > 0$, что $\|J'(x)\| \geq \delta$, если $x \notin S(X_*)$, где $S(X_*) = \{x \mid \mathrm{d}(x, X_*) \leq \varepsilon\}$; $d(x, X_*) = \min\limits_{x^* \in X_*} \left(x - x^*\right)$;
4. Для любых $x, y \in R^n$ имеем $\|J'(x + y) - J'(x)\| \leq l \|y\|$ $(l > 0)$.
5. Собственные числа матрицы $J''(x)$ заключены в интервале $[-M, M]$, где $M > 0$ не зависит от x.

Тогда независимо от выбора начальной точки x^0 для последовательности $\{x^k\}$, построенной согласно выражениям (11.27), (11.28), выполняются предельные соотношения:

$$\lim \mathrm{d}(x^k, X_*) = 0; \tag{11.30}$$

$$\lim J(x^k) = J(x^*), \quad k \to \infty. \tag{11.31}$$

Доказательство. Используются соотношения

$$J(x+y) = J(x) + \int_0^1 \left\langle J'(x + \vartheta y), y \right\rangle d\vartheta;$$

$$\left| \int_0^1 \left\langle x(\vartheta), y(\vartheta) \right\rangle d\vartheta \right| \leq \int_0^1 \|x(\vartheta)\| \, \|y(\vartheta)\| d\vartheta.$$

Обозначая $J_k = J(x^k)$, $J'_k = J'(x^k)$, $A_k = J''(x^k)$, получаем

$$\begin{aligned} J_k - J\left[x^k - H(A_k, h)J'_k\right] &= \int_0^1 \left\langle J'\left[x^k - \vartheta H(A_k, h)J'_k\right], H(A_k, h)J'_k \right\rangle d\vartheta = \\ &= \left\langle H(A_k, h)J'_k, J'_k \right\rangle - \int_0^1 \left\langle J'_k - J'(x^k - \vartheta H(A_k, h)J'_k), H(A_k, h)J' \right\rangle d\vartheta \geq \\ &\geq \left\langle H(A_k, h)J'_k J'_k \right\rangle - l \left\| H(A_k, h)J'_k \right\|^2 \int_0^1 \vartheta d\vartheta = \left\langle H(A_k, h)J'_k, J'_k \right\rangle - \end{aligned} \tag{11.32}$$

$$- 0{,}5l \left\| H(A_k, h)J'_k \right\|^2 \geq \rho \left\| J'_k \right\|^2 - 0{,}5l \left\| J'_k \right\|^2 R^2 = \alpha \left\| J'_k \right\|^2;$$

$$\alpha = \rho - (l/2)R^2.$$

При этом использованы неравенства

$$\rho\|y\|^2 \le \langle H(A_k,h)y,y\rangle \le R\|y\|^2, \qquad (11.33)$$

где ρ, R — минимальное и максимальное собственные числа положительно определенной матрицы $H(A_k,h)$.

Левое неравенство (11.33) следует из представления минимального собственного числа λ любой симметричной матрицы B в виде $\lambda = \min\limits_{x\neq 0}\left[\langle Bx,x\rangle / \langle x,x\rangle\right]$, а правое — из условия согласования $\|Bx\| < \|B\| \cdot \|x\|$ сферической нормы вектора $\|x\| = \sqrt{\langle x,x\rangle}$ и спектральной нормы симметричной матрицы $\|B\| = \max\limits_i |\lambda_i(B)|$, где $\lambda_i(B), i = \overline{1,n}$ — собственные числа матрицы B. Для значений ρ и R получим:

$$\rho = \min_i \lambda_i[H(A,h)] = \min_i \int_0^h \exp[-\lambda_i(A)\tau]d\tau;$$

$$R = \max_i \int_0^h \exp[-\lambda_i(A)\tau]d\tau.$$

Согласно пятому условию имеем

$$\int_0^h \exp(-M\tau)d\tau \le \int_0^h \exp[-\lambda_i(A)\tau]d\tau \le \int_0^h \exp(M\tau)d\tau,$$

поэтому

$$\rho \ge \int_0^h \exp(-M\tau)d\tau = M^{-1}[1-\exp(-Mh)];$$

$$R \le \int_0^h \exp(M\tau)d\tau = M^{-1}[\exp(Mh)-1];$$

$$\alpha = \rho - 0,5lR^2 \ge M^{-1}[1-\exp(-Mh)] - 0,5lM^{-2}[\exp(Mh)-1]^2.$$

Принимая $h = 1/M$ и считая без ограничения общности, что $M > le(e-1)/2$, где e — основание натуральных логарифмов, получаем

$$\alpha \ge (e-1)[2M - le(e-1)]/(2M^2e) > 0. \qquad (11.34)$$

Из соотношений (11.29), (11.32), (11.33) следует

$$J_k - J_{k+1} \ge J_k - J\left[x^k - H(A_k,M^{-1})J_k\right] \ge \alpha\|J_k\|^2. \qquad (11.35)$$

Следовательно, последовательность $\{J_k\}$ является монотонно невозрастающей и ограниченной снизу величиной $J(x^*)$, поэтому она имеет предел и $J_{k+1} - J_k \to \infty$ при $k \to \infty$.

Из выражения (11.35) следует

$$\|J'_k\|^2 \le \alpha^{-1}(J_k - J_{k+1}),$$

поэтому $\|J'_k\| \to 0$ при $k \to \infty$. А поскольку по условию $\|J'_k\| \ge \delta$ при $x^k \notin S(X_*)$, найдется такой номер N, что $x^k \in S(X_*)$ при $k \ge N$ и, следовательно, справедливо утверждение (11.30).

Обозначим через $\bar{x}^k$ проекцию x^k на множество X_*. Тогда по теореме о среднем

$$J_k - J(\bar{x}^k) = \left\langle J'(x^{kc}), x^k - \bar{x}^k \right\rangle,$$

где

$$x^{kc} = \bar{x}^k + \lambda_k(x^k - \bar{x}^k), \quad \lambda_k \in [0,1].$$

Учитывая, что $J'(\bar{x}^k) = 0$, получаем

$$J_k - J(\bar{x}^k) = \left\langle J'(x^{kc}) - J'(\bar{x}^k), x^k - \bar{x}^k \right\rangle \le$$

$$\le \left\| J'(x^{kc}) - J'(\bar{x}^k) \right\| \cdot \left\| x^k - \bar{x}^k \right\| \le l d^2(x^k, X_*).$$

Из соотношения (11.30) получаем (11.31). Теорема доказана.

Замечания

1. Утверждения теоремы, очевидно, выполняются, если h_k выбирать не из условия (11.29), а из условия

$$J[x^k - H(A_k, h_k)J'_k] = \min_{h \in [0, \bar{h}]} J[x^k - H(A_k, h)J'_k],$$

где $\bar{h} > 0$ — произвольное число. Действительно, легко видеть, что неравенство (11.35) только усилится, если брать любое другое значение h_1 (может быть даже большее чем $2 / [le(e-1)]$) с меньшим значением функционала, чем при $h = 1 / M$, и в то же время, если при $h = 1 / M$ сходимость имеет место, то она сохраняется и при меньших значениях h. Последнее следует из возможности выбора сколь угодно больших значений M при установлении сходимости.

2. Соотношения (11.30), (11.31) сохраняются также при замене условия (11.29) на следующее:

$$J_{k+1} = J[x^k - H(A_k, h_k)J'_k] \le (1-\gamma_k)J'_k + \gamma_k \min_{h>0} J[x^k - H(A_k, h)J'_k]$$

$$(0 < \gamma \le \gamma_k \le 1). \qquad (11.36)$$

Действительно, из условия (11.36) будем иметь

$$J_k - J_{k+1} \ge \gamma_k \{J_k - \min_{h>0} J[x^k - H(A_k, h)J'_k]\} \ge$$

$$\ge \gamma_k \{J_k - J[x^k - H(A_k, h)J'_k]\}$$

и согласно (11.32)

$$J_k - J_{k+1} \ge \gamma_k \alpha \|J'_k\|^2 = \bar{\alpha}\|J'_k\|^2, \quad \bar{\alpha} > 0.$$

Получено неравенство, аналогичное (11.35), и далее доказательство проводится по той же схеме с заменой α на $\bar{\alpha}$..

При сильной выпуклости функционала $J(x)$ удается получить оценку скорости сходимости.

Теорема 11.3. Пусть:

1. $J(x) \in C^2(R^n)$.
2. Для любых $x, y \in R^n$ выполняются условия

$$\lambda\|y\| \le \langle J''(x)y, y\rangle \le \Lambda\|y\|^2, \|J''(x+y) - J''(y)\| \le L\|x\|, \ \lambda > 0, L \ge 0.$$

Тогда независимо от выбора начальной точки x^0 для метода (11.27) справедливы соотношения (11.30), (11.31) и оценка скорости сходимости

$$\|x^{k+1} - x^k\| \le (\Lambda/\lambda)^{1/2} L\|x^k - x^*\|^2 / (2\lambda).$$

Доказательство здесь не приводится.

Таким образом, установлена квадратичная скорость сходимости, характерная для методов ньютоновского типа.

11.4. Реализация и область применимости методов с экспоненциальной функцией релаксации

Алгоритм вычисления матричных функций (11.28) может быть основан на использовании известного рекуррентного соотношения

$$H(A, 2h) = H(A, h)[2E - AH(A, h)]. \qquad (11.37)$$

Поскольку все рассматриваемые матричные функции симметричны и, следовательно, обладают простой структурой, чтобы доказать (11.37), достаточно проверить его для соответствующих скалярных зависимостей, что тривиально.

Формула (11.37) используется в вычислительной практике также для получения обратной матрицы A^{-1}, т. к. выполняется предельное соотношение $H(A,h) \to A^{-1}$ $(h \to \infty)$.

Это еще раз указывает на связь метода ЭР с методом Ньютона, который является предельным вариантом рассматриваемого алгоритма при условии положительной определенности матрицы A.

Выбор параметра h при известной матрице A_k или ее аппроксимации может осуществляться различными способами. В каждом из них приближенно реализуется соотношение (11.29). Наиболее простой прием заключается в следующем.

Задают некоторую малую величину h_0, такую, чтобы матрицу $H(A_k,h_0)$ можно было заменить отрезком соответствующего степенного ряда

$$H(A_k, h_0) \approx h_0 \sum_{i=1}^{m} (-A_k h_0)^{i-1} / i! \qquad (11.38)$$

Далее последовательно наращивают h с помощью соотношения (11.37), вычисляя каждый раз значение $J[x^k - H(A_k, 2^q h_0)J'_k]$, $q = 0, 1, \ldots$. Процесс продолжается до тех пор, пока функция убывает. Точка с минимальным значением J принимается за x^{k+1}. При этом вместо точной реализации соотношения (11.29) оптимальный шаг выбирается на дискретной сетке значений $h_q = 2^q h_0$, $q = 0, 1, 2, \ldots$. Как правило, предельное значение q не превышает 30. Рассмотренная реализация метода ЭР называется *системным алгоритмом оптимизации.*

Во многих случаях более эффективной оказалась реализация метода с элементами адаптации, в которой значение J не вычислялось для всех промежуточных значений q. Функционал вычислялся только для трех значений q: q^*-1, q^*, $q^* + 1$, где q^* — оптимальное значение q, полученное на предыдущей итерации по k. На первой итерации для определения q^* необходимо вычислить все значения J.

С целью более точной локализации минимума на каждом шаге по k могут использоваться процедуры одномерного поиска по h, например, метод золотого сечения. Для этого изложенным ранее грубым способом определяется промежуток $[h_{\min}, h_{\max}]$, содержащий оптимальное в смысле условия (11.29) значение h_*.

Примем

$$\varphi(h) = J[x^k - H(A_k, h)J'(x^k)].$$

Тогда

$$h_{\min} = 2^{q'} h_0,\ h_{\max} = 2^{q'+2} h_0,\ h_* = 2^{q'+1} h_0,$$

причем предполагается, что

$$\varphi(h_{\min}) > \varphi(h_*),\ \varphi(h_{\max}) > \varphi(h_*).$$

Фиксируя число пересчетов q', получаем, что, выбирая $h'_0 \in [h_0, 4h_0]$, имеем $h = 2^{q'} h'_0 \in [h_{\min}, h_{\max}]$. Далее можно принять $\varphi(h) = \psi(h'_0)$, и задача сводится к стандартной задаче минимизации функции одной переменной $\psi(h'_0)$ на заданном промежутке.

Для приближенного вычисления матрицы A_k вторых производных функционала $J(x)$ могут применяться любые методы, изложенные в *разд. 10.3*. Рассмотрим наиболее универсальный алгоритм, основанный на конечно-разностных соотношениях. В результате вычисления по формулам (10.15), (10.16) приходим к матрице $A_k = D_k / \beta_k^2$ и вектору $J'_k = f_k / \beta_k$, где $\beta_k = 2s_k$, s_k — шаг дискретности. Как уже говорилось, производить деление матрицы D_k на β_k^2 или вектора f_k на β_k с целью получения A_k и J'_k нецелесообразно. Поэтому далее принципиальная схема метода ЭР будет преобразована к виду, удобному для непосредственного применения D_k и f_k вместо A_k и J'_k.

Имеем

$$H(D_k, h) = \int_0^h \exp(-D_k\tau)d\tau = \beta_k^{-2}\int_0^h \exp(-A_k\beta_k^2\tau)d\beta_k^2\tau = \beta_k^{-2}\int_0^{\beta_k^2 h} \exp(-A_k t)dt$$

или

$$\beta_k^2 H(\beta_k^2 A_k, h) = H(A_k, h_k);\ h_k = \beta_k^2 h. \tag{11.39}$$

С учетом (11.39) основное соотношение (11.27) приводится к виду

$$\begin{aligned} x^{k+1} &= x^k - H(A_k, h_k)J'(x^k) = \\ &= x^k - \beta_k^2 H(D_k, h_k/\beta_k^2) f_k / \beta_k = \\ &= x^k - 2s_k H(D_k, h) f_k; \\ h &= h_k / 4s_k^2. \end{aligned} \tag{11.40}$$

Имеем также

$$H(D_k, 2h) = H(D_k, h)[2E - D_k H(D_k, h)].$$

Оптимальное значение h в (11.40) находится непосредственно из соотношения

$$J(x^{k+1}) = \min_{h>0} J(x^{k+1} - 2s_k H(D_k, h) f_k].$$

При использовании разностного уравнения (11.40) упрощенная вычислительная схема метода с экспоненциальной релаксацией может быть сведена к следующей последовательности действий.

11.4.1. Алгоритм RELAX

- Шаг 1. Ввести исходные данные x^0, s.
- Шаг 2. Принять $x := x^0$; $J := J(x)$; $x^l = x$; $J_l := J$.
- Шаг 3. Вычислить матрицу $D = \{d_{ij}\}$ и вектор $f = \{f_i\}$ в точке x по формулам

$$d_{ij} = J(x + se_i + se_j) - J(x - se_i + se_j) - J(x + se_i - se_j) + J(x - se_i - se_j)$$

$$(i, j = \overline{1,n}); \qquad (11.41)$$

$$f_i = J(x + se_i) - J(x - se_i) \qquad [i = \overline{1,n}; e_i = (0, ..., 1, ..., 0)]; \qquad (11.42)$$

  принять $h_0 := 0{,}1 / \|D\|$.
- Шаг 4. Принять $k := 0$. Вычислить матрицу $H = H(D, h_0)$:

$$H = \sum_{i=1}^{7} (-D)^{i-1} h_0^i / i! \qquad (11.43)$$

- Шаг 5. Принять $x^l := x - 2sHf$; $J_l := J(x^l)$; $k := k + 1$.
- Шаг 6. Если $J_l < J_l$, принять $x^l := x^l$; $J_l := J_l$.
- Шаг 7. Если $k > 20$, перейти к шагу 8, иначе положить $H := H(2E - DH)$ и перейти к шагу 5.
- Шаг 8. Проверить условия окончания процесса оптимизации в целом; если они выполняются, остановить работу алгоритма; в противном случае принять $x := x^l$; $J := J_l$ и перейти к шагу 3.

Заметим, что выбор на шаге 3 параметра $h_0 = 0{,}1 / \|D\|$ эквивалентен при $A = J''(x)$ равенству $h_0 = 0{,}1 / \|A\|$ в исходной схеме алгоритма. А послед-

нее равенство, как это следует из результатов, полученных при доказательстве теоремы 11.2, гарантирует убывание функционала:

$$J[x^k - H(A_k, h_0)J'_k] < J(x^k). \qquad (11.44)$$

Действительно, как было показано ранее, для выполнения неравенства (11.44) достаточно принять $h_0 \le 1 / M$, где $M > le(e-1)/2 \approx 2{,}3l$. Для квадратичного функционала, аппроксимирующего $J(x)$ в окрестности точки x, имеем $l = \|A\|$, где $A = J''(x)$. Поэтому можно выбрать h_0 из условия $h_0 \le 1 / (2{,}3\|A\|) \approx 0{,}4 / \|A\|$. Замена коэффициента 0,4 на 0,1 позволяет более точно реализовать шаг 4 алгоритма, одновременно гарантируя выполнение (11.44).

Параметр s_k может меняться от итерации к итерации в зависимости, например, от значения нормы $\|x^k - x^{k-1}\|$ так, как это было описано в *разд. 10.3*. Возможны и другие способы регулировки шага.

Обратимся к анализу влияния погрешностей вычислений при реализации методов ЭР.

Рассмотрим итерационный процесс, определяемый рекуррентным соотношением

$$x^{k+1} = x^k - H(A_k, h_k)A_k x^k = g(A_k)x^k, \qquad (11.45)$$

где $g(A) = E - H(A, h)A$. Такой процесс является упрощенной моделью метода ЭР, характеризуя его локальные свойства. Оценим влияние погрешностей в представлении матрицы A_k на характеристики релаксационности последовательности $\{f(x^k)\}$.

Кроме предположения о квадратичном характере $J(x)$ в окрестности точки x^k, неявно введено еще одно допущение. Именно, заменяя в соотношении (11.45) матрицу A_k на возмущенную матрицу $A + dA$ (индекс k у матрицы далее будем опускать), предполагаем, что ошибки в вычислении J'' и J' определенным образом согласованы. В действительности эквивалентное возмущение dA матрицы, определяющей градиент Ax^k, может не совпадать с возмущением матрицы J'', т. к. J' и J'' вычисляются раздельно. Однако с позиций последующего анализа это отличие не является принципиальным.

Предположим, что собственные числа матрицы A разделены на две группы

$$\lambda_1 \ge \ldots \ge \lambda_{n-r} >> |\lambda_{n-r+1}| \ge \ldots \ge |\lambda_n|. \qquad (11.46)$$

Возмущение dA матрицы A приводит к появлению возмущений $d\lambda_i$ для собственных чисел и возмущений du_i для отвечающих им собственных векторов. Согласно результатам, полученным в *разд. 10.2*, будем считать,

что вариации собственных векторов происходят в пределах линейных многообразий

$$M_1 = \sum_{i=1}^{n-r} \alpha_i u_i; \quad M_2 = \sum_{j=n-r+1}^{n} \alpha_j u_j,$$

порожденных собственными векторами $\{u_i, i = 1, ..., n-r\}$ и $\{u_j, j = n-r+1, ..., n\}$ исходной невозмущенной матрицы A. В этом случае матрицы A и $A + dA$ одновременно не приводятся к главным осям, что вносит дополнительный элемент сложности в анализ влияния погрешностей.

Пусть

$$U^T A U = \mathrm{diag}(\lambda_i); \quad U = (u_1, u_2, ..., u_n); \tag{11.47}$$

$$\omega^T (A + dA)\omega = \mathrm{diag}(\lambda_i + d\lambda_i), \quad \omega = (\omega_1, \omega_2, ..., \omega_n).$$

Имеем теперь

$$g(A) = E - \omega D_1 \omega^T \omega D_2 \omega^T = E - \omega D_3 \omega^T,$$

где

$$D_1 = \mathrm{diag}\left\{\int_0^h \exp[(-\lambda_i - d\lambda_i)\tau] d\tau\right\};$$

$$D_2 = \mathrm{diag}(\lambda_i + d\lambda_i);$$

$$D_3 = D_1 D_2.$$

Таким образом, матрица $g(A)$ имеет собственные векторы ω_i и соответствующие им собственные числа $\lambda_i(g) = 1 - \lambda_i(D_3)$.

Принимая

$$x^k = \sum_{i=1}^{n} \xi_{i,k} \omega_i; \quad \omega_i = \sum_{j=1}^{n} \alpha_{ji} u_i,$$

получаем

$$f(x^k) = 1/2\left(Ax^k, x^k\right(= 1/2 \sum_{i=1}^{n} \xi_{i,k}^2 \left(\sum_{j=1}^{n} \alpha_{ji}\right)^2 \lambda_i.$$

Аналогично имеем

$$x^{k+1} = g(A)x^k = \sum_{i=1}^{n} \xi_{i,k} \lambda_i(g) \omega_i = \sum_{i=1}^{n} \xi_{i,k+1} \omega_i;$$

$$f(x^{k+1}) = 1/2 \sum_{i=1}^{n} \xi_{i,k+1}^2 \left(\sum_{j=1}^{n} \alpha_{ji}\right)^2 \lambda_i = 1/2 \sum_{i=1}^{n} \xi_{i,k}^2 \left(\sum_{j=1}^{n} \alpha_{ji}\right)^2 \lambda_i \lambda_i^2(g),$$

где

$$\lambda_i(g) = 1 - (\lambda_i + d\lambda_i)\int_0^{h_k} \exp[(-\lambda_i - d\lambda_i)\tau]d\tau = \\ = \exp[(-\lambda_i - d\lambda_i)h_k]. \tag{11.48}$$

Для выполнения неравенства $f(x^{k+1}) \le f(x^k)$ согласно теореме 11.1 должны выполняться условия релаксационности

$$|\lambda_i(g)| \le 1 \ (\lambda_i > 0); \ |\lambda_i(g)| \ge 1 \ (\lambda_i < 0). \tag{11.49}$$

Теперь легко видеть, что если возмущение $d\lambda_i$ таково, что собственное число меняет знак:

$$\operatorname{sgn}(\lambda_i) \ne \operatorname{sgn}(\lambda_i + d\lambda_i), \tag{11.50}$$

то условия (11.49), вообще говоря, нарушаются. Это приводит к резкому замедлению сходимости, а в некоторых случаях к полной остановке процесса оптимизации.

Пусть вариация dA матрицы A вызывается только погрешностями округления. Тогда неравенство (11.50) невозможно, если все малые собственные числа по модулю ограничены снизу величиной $n\lambda_1\varepsilon_M$. Действительно, в этом случае $\operatorname{sgn}(\lambda_i) = \operatorname{sgn}(\lambda_i + d\lambda_i)$, т. к. $|d\lambda_i| \le n\lambda_1\varepsilon_M \le |\lambda_i|$. Отсюда имеем следующее ограничение степени овражности функционалов, эффективно минимизируемых методами ЭР:

$$\eta(x^k) \le 1 / (n\varepsilon_M). \tag{11.51}$$

Проведенный анализ показывает, что погрешности вычислений при достаточно больших значениях η могут приводить к случайному характеру множителей релаксации для малых собственных чисел, что определяет резкое снижение эффективности метода. Из соотношения (11.51) следует, что трудности возрастают при увеличении размерности n решаемой задачи и уменьшении длины разрядной сетки компьютера. Вычисления с двойной точностью приводят к оценке $\eta(x^k) \le 1 / (n\varepsilon_M^2)$ и позволяют решать существенно более широкий класс задач.

Как показывает практика, наибольшую эффективность методы типа RELAX имеют при решении задач с удовлетворяющим неравенству (11.51) значением η при кусочно-квадратичном характере зависимости $J(x)$. Характер выпуклости $J(x)$ при этом безразличен.

В отличие от методов ОПС матричные градиентные схемы типа RELAX оказываются менее универсальными. Однако там, где они применимы, может быть получен заметный вычислительный эффект. Кроме того, как показано ниже, на базе градиентных методов могут быть построены ал-

горитмы оптимизации с большим числом n оптимизируемых параметров. Следовательно, рассматриваемые классы методов взаимно дополняют друг друга, не позволяя выделить какой-то один наилучший подход.

11.5. Методы оптимизации больших систем

Под *большими системами* будем понимать системы, описываемые моделями с большим числом управляемых параметров. Если степень овражности соответствующих критериев оптимальности достаточно высока, то стандартные вычислительные средства оказываются неэффективными в силу изложенных ранее причин. Методы ОПС, а также методы ЭР неприменимы, т. к. их вычислительные схемы содержат заполненные матрицы размерности $n \times n$, что при больших (около 1000) n определяет чрезмерные требования к объему необходимой памяти компьютера. Это же замечание справедливо для квазиньютоновских алгоритмов типа метода Давидона—Флетчера—Пауэлла, Бройдена, Пирсона, Мак-Кормика, Бройдена—Флетчера—Гольдфарба—Шенно, Пауэлла—Бройдена и т. д.

Наиболее часто в указанной ситуации рекомендуется применять различные нематричные формы метода сопряженных градиентов (СГ). Однако, как показано далее, их возможности также весьма ограниченны. Это вызвано тем, что доступное число итераций оказывается меньше размерности и в результате гарантируется сходимость со скоростью геометрической прогрессии с показателем, близким к единице:

$$\|x^k - x^*\| \le 2t^k \|x^0 - x^*\|, \qquad (11.52)$$

где $x^* = \operatorname{argmin} f(x)$, $x \in R^n$, $t \approx (1 - 2/\sqrt{\eta})$; η — коэффициент овражности минимизируемого сильно выпуклого квадратичного функционала[1]. Таким образом, конечность метода СГ при решении задач минимизации квадратичных функционалов в этом случае не играет роли.

Далее показано, что в рамках класса матричных градиентных схем (11.2) могут быть построены алгоритмы, более эффективные для рассматриваемых задач, чем методы СГ.

Пусть оптимизируемая большая система может быть представлена как совокупность q взаимосвязанных подсистем меньшей размерности. Пусть

[1] В литературе оценка (11.52) часто приводится с ошибкой в множителе.

также требования к выходным параметрам системы могут быть сформулированы в виде неравенств

$$y_j(x^j, x^q) \le t_j \quad (j = \overline{1, q-1}); \quad y_q(x^q) \le t_q, \tag{11.53}$$

где x^j — n_j-мерный частный вектор управляемых параметров; x^q — n_q-мерный вектор управляемых параметров, влияющий на все q выходных параметров и осуществляющий связь отдельных подсистем оптимизируемой системы. Размерность полного вектора управляемых параметров $x = (x^1, x^2, ..., x^q)$ равна

$$n = \sum_{i=1}^{q} n_i. \tag{11.54}$$

Используя технику построения целевых функционалов, представленную в *главе 10*, можно привести задачу решения системы неравенств (11.53) к виду

$$J(x) = \sum_{j=1}^{q} \phi_j(x^j, x^q) \to \min_{x \in R^n}, \tag{11.55}$$

где критерий (11.55) является сглаженным вариантом минимаксного критерия.

Функционалы (11.55) возникают и при других постановках задач оптимизации, не основанных непосредственно на минимаксных критериях. Поэтому задача (11.55) имеет достаточно общий характер.

Далее будут рассмотрены методы решения задачи (11.55) при следующих предположениях.

1. Критерий $J(x)$ обладает относительно высокой степенью овражности, а его выпуклость гарантируется только в окрестности точки минимума.
2. Размерность (11.54) полного вектора управляемых параметров x велика, что, с одной стороны, затрудняет применение стандартных методов оптимизации из-за ограниченного объема доступной памяти компьютера, а с другой — не позволяет реализовать предельно возможные характеристики сходимости алгоритмов.
3. Решение задачи анализа оптимизируемой системы требует значительных вычислительных затрат. Поэтому в процессе оптимизации требуется минимизировать число обращений к вычислению значений $J(x)$.
4. Коэффициент заполнения γ матрицы $A(x) = J''(x)$ достаточно мал. Обычно можно считать $\gamma \sim 1 / q$.

При сделанных предположениях структура матрицы $A(x)$ не зависит от точки x (рис. 11.7).

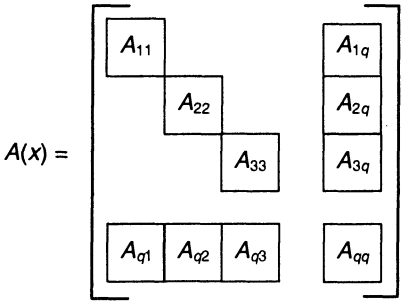


Рис. 11.7. Структурированная матрица Гессе

Подматрица A_{ij} имеет размеры $n_i \times n_j$, а общее число ненулевых элементов равно

$$\sum_{i=1}^{q} n_i^2 + 2n_q \sum_{i=1}^{q-1} n_i.$$

Таким образом, учитывая симметричность матрицы $A(x)$, в памяти компьютера необходимо хранить

$$\sum_{i=1}^{q} \frac{n_i^2 + n_i}{2} + n_q \sum_{i=1}^{q-1} n_i$$

ненулевых элементов. Необходимые сведения о схемах хранения разреженных матриц широко представлены в литературе.

Из изложенных в предыдущих разделах методов только методы Ньютона и Левенберга могут рассматриваться при оптимизации больших систем с достаточно высокими показателями овражности η. Однако неприменимость метода Ньютона в невыпуклой ситуации и отмеченные в *разд. 11.2* недостатки метода Левенберга не позволяют считать вопрос решенным.

Обратимся снова к классу матричных градиентных схем (11.2).

В силу приведенных выше предположений и сформулированных в *разд. 11.1* требований к функциям релаксации наиболее рациональный метод должен иметь функцию релаксации, значения которой резко снижаются от $R = 1$ при $\lambda = 0$, оставаясь малыми во всем диапазоне $[0, M]$. И напротив, при $\lambda < 0$ функция $R(\lambda)$ должна интенсивно возрастать. Кроме того, отвечающая $R(\lambda)$ матричная функция H должна строиться без матричных умножений для сохранения свойства разреженности матрицы $A_k = J''(x^k)$.

Покажем, что в качестве такой $R(\lambda)$ с точностью до множителя могут быть использованы *смещенные полиномы Чебышева второго рода* $P_s(\lambda)$, удовлетворяющие следующим соотношениям:

$$P_1(\lambda) = 1;\ P_2(\lambda) = 2(1 - 2\lambda); \tag{11.56}$$

$$P_{s+1}(\lambda) = 2(1 - 2\lambda)P_s(\lambda) - P_{s-1}(\lambda).$$

График зависимости $P_s(\lambda) / s$ для некоторого s представлен на рис. 11.8.

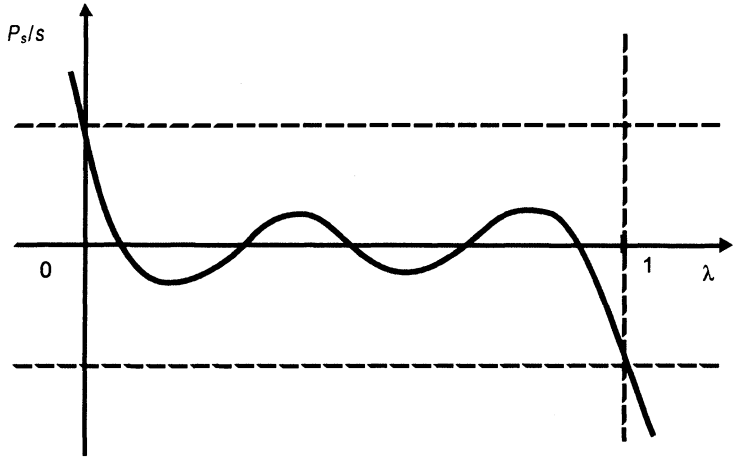


Рис. 11.8. Полином Чебышева

Действительно, полагая $R(\lambda) = P_L(\lambda) / L$ при достаточно большом значении L, получим сколь угодно быструю релаксацию любого слагаемого в представлении *(см. разд. 11.1)*

$$f(x^{k+1}) = 1/2\sum_{i=1}^{n} \xi_{i,k}^2 \lambda_i R^2(\lambda_i), \tag{11.57}$$

где

$$x^k = \sum_{i=1}^{n} \xi_{i,k} u_i.$$

Это утверждение вытекает из известного факта равномерной сходимости последовательности $\{P_s(\lambda) / s\}$ к нулю при $s \to \infty$ на промежутке (0; 1). Далее будем предполагать, что собственные числа матрицы A_k нормированы к промежутку (0;1). Для этого достаточно вместо матрицы J'' рассматривать матрицу $J'' / \|J''\|$, а вместо вектора J' — вектор $J' / \|J''\|$.

Отвечающая принятой $R(\lambda)$ зависимость $H(\lambda)$ имеет вид

$$H(\lambda) = [1 - R(\lambda)] / \lambda = [1 - P_L(\lambda) / L] / \lambda. \qquad (11.58)$$

Построение методов (11.2) непосредственно с функцией (11.58) возможно, но приводит, как и в методе Левенберга, к необходимости решения на каждом шаге по k больших линейных систем уравнений с разреженной матрицей. Ниже показано, что существуют более эффективные приемы реализации.

Действительно, из выражений (11.58) следует, что $H(\lambda)$ является полиномом степени $L - 2$, в то время как $R(\lambda)$ имеет степень $L - 1$. Поэтому для реализации матричного градиентного метода с указанной функцией $H(\lambda)$ нет необходимости решать линейные системы. Метод будет выглядеть следующим образом:

$$x^{k+1} = x^k - (\alpha_1 E + \alpha_2 A_k + \ldots + \alpha_{L-1} A_k^{L-2}) J'(x^k) = x^k - H(A_k) J'(x^k),$$
$$A_k = J''(x^k). \qquad (11.59)$$

Реализация метода (11.59) может быть основана на методах вычисления коэффициентов α_i для различных степеней L. При этом число L должно выбираться из условия наиболее быстрого убывания $J(x)$. Альтернативный, более предпочтительный подход основан на других соображениях.

Для функции

$$H_s(\lambda) = \alpha_1 + \alpha_2 \lambda + \ldots + \alpha_{s-1} \lambda^{s-2}, \; s = 2, 3,$$

из (11.56) можно получить рекуррентное соотношение

$$(s+1) H_{s+1} = 2s(1 - 2\lambda) H_s - (s-1) H_{s-1} + 4s, \qquad (11.60)$$
$$(H_1 = 0, \; H_2 = 2, \; s = \overline{2, L-1}).$$

Из соотношения (11.60) имеем

$$x^{k+1}[s+1] = x^k - H_{s+1} J'_k = x^k - \frac{2s}{s+1}(E - 2A_k) \cdot H_s J'_k + \frac{s-1}{s+1} H_{s-1} J'_k - \frac{4s}{s+1} J'_k$$
$$(s = \overline{2, L-1})$$

или

$$\vartheta_{s+1} = x^{k+1}[s+1] - x^k =$$
$$= \frac{2s}{s+1}(E - 2A_k)\vartheta_s - \frac{s-1}{s+1}\vartheta_{s-1} - \frac{4s}{s+1} J'_k \qquad (11.61)$$
$$(\vartheta_1 = 0, \vartheta_2 = -2J'_k, s = \overline{2, L-1}).$$

Здесь $x^{k+1}[s]$ есть s-е приближение к вектору $x^{k+1} = x^{k+1}[L]$.

Таким образом, при фиксированной квадратичной аппроксимации $f(x)$ функционала $J(x)$ в окрестности $x = x^k$ мы имеем возможность переходить от P_s к P_{s+1} в результате одного умножения матрицы $E - 2A_k$ на вектор ϑ_s, в полной мере используя свойство разреженности матрицы $A_k = J''_k$ и не прибегая к дополнительным вычислениям градиента. Эффективность алгоритма (11.61) при больших значениях η определяется множителями релаксации для малых собственных значений матрицы A_k. Рассмотрим положительную часть спектра ($\lambda > 0$), что особенно важно в окрестности оптимума, где матрица $J''(x)$ положительно определена. Основное достоинство метода с функцией релаксации вида $R_s(\lambda) = P_s(\lambda) / s$ состоит в том, что уже при малых s происходит заметное подавление слагаемых из (11.57) в широком диапазоне значений λ. В табл. 11.1 представлены значения R_s для внутреннего максимума $R_s(\lambda)$ и границы диапазонов $\alpha_s \le \lambda \le \beta_s$, где $|R_s(\lambda)| \le R_s$.

Таблица 11.1. *Характеристики множителей релаксации*

s	3	4	5	6	7	8
R_s	0,333	0,272	0,250	0,239	0,233	0,230
α_s	0,147	0,092	0,061	0,044	0,033	0,025
β_s	0,853	0,908	0,939	0,956	0,967	0,975
$-R'_s(0)$	5,30	10,0	16,0	23,3	32,0	42,0

Можно показать, что значения α_s, β_s для $s > 8$ могут быть вычислены по асимптотической формуле

$$\alpha_s = 1{,}63 / s^2; \quad \beta_s = 1 - \alpha_s; \tag{11.62}$$

при этом $R_s < 0{,}23$. В левой части спектра ($\lambda < 0$) имеем $R_s(\lambda) > 1 + R'_s(0)\lambda$, поэтому значения производных $R'_s(0)$ характеризуют множители релаксации для отрицательных слагаемых в (11.57).

Упрощенная схема алгоритма, построенного на основе соотношения (11.61), может быть реализована с помощью следующей последовательности шагов.

11.5.1. Алгоритм RELCH

- Шаг 1. Задать начальную точку x; вычислить $J := J(x)$; задать L.
- Шаг 2. Вычислить $J' := J'(x)$, $J'' := J''(x)$; принять $J' := J' / \|J''\|$, $J'' := J'' / \|J''\|$, $\alpha := 1$.
- Шаг 3. По формуле (11.61) построить ϑ_L; принять $x' := x + \vartheta_L$.
- Шаг 4. Вычислить $J_t := J(x')$. Если $J_t > J$, перейти к шагу 5; иначе — к шагу 6.
- Шаг 5. Принять $\alpha := \alpha / 2$, $x' := x + \alpha\vartheta_L$ и перейти к шагу 4.
- Шаг 6. Принять $x := x'$, $J := J_t$ и перейти к шагу 2.

Критерий окончания процесса здесь не указан. Как правило, вычисления заканчиваются по исчерпании заданного числа вычислений функционала либо при явной остановке алгоритма. Число пересчетов L по формуле (11.61) является параметром, задаваемым пользователем. Согласно (11.62) первоначально целесообразно принять $L \approx 1{,}3\sqrt{\eta}$, $\eta \approx 1 / \alpha_L$, где η — оценка степени овражности минимизируемого функционала. При таком выборе L множители релаксации в положительной части спектра будут гарантированно меньше 0,23. При конструировании алгоритмических способов задания L необходимо учитывать, что последовательность $\{J_s\}$, где $J_s = J(x^k + \vartheta_s)$ не будет при $s \to \infty$ убывать монотонно. На шаге 5 алгоритма применена регулировка нормы вектора продвижения с целью предотвращения выхода из области справедливости локальной квадратичной модели функционала.

Дадим оценку эффективности метода (11.61) по сравнению с методом сопряженных градиентов (СГ), наиболее конкурентоспособным из стандартных методов решения больших задач оптимизации.

Важная особенность алгоритмов типа RELCH заключается в том, что соответствующие множители релаксации будут определяться только числом итераций L и степенью овражности η задачи независимо от размерности n. В то же время в схемах методов СГ для завершения каждого цикла спуска требуется порядка n итераций; в противном случае согласно (11.52) скорость сходимости может быть очень малой. Кроме того, каждая итерация метода СГ даже для квадратичной функции требует нового вычисления градиента, т. е. дополнительных вычислительных затрат по анализам функционирования оптимизируемой системы.

Будем далее полагать, что алгоритм RELCH реализован с постоянным $L = \sqrt{\eta}$, имея в области $\lambda > 0$ множители релаксации, не превышающие значения 0.23.

Рассмотрим задачу минимизации квадратичного функционала $f(x) = 1 / 2\langle Ax, x\rangle$ с положительно определенной матрицей A. Оценим количество вычислений $f(x)$, требуемое для достижения контрольного вектора x' с нормой $\|x'\| \le 0{,}23$, методом СГ и алгоритмом RELCH из начальной точки x^0 с $\|x_0\| = 1$. По достижении точки x' вся ситуация повторяется, поэтому полученные ниже сравнительные оценки эффективности имеют достаточно общий характер.

Будем предполагать также, что для вычисления производных применяются двусторонние конечно-разностные соотношения (10.15), (10.16).

Для достижения вектора x' по алгоритму RELCH требуется вычислить в точке x^0 слабо заполненную матрицу Гессе с заполненной главной диагональю и вектор градиента $f'(x^0)$. При коэффициенте заполнения γ для этого потребуется около $2\gamma n^2$ вычислений f. Далее выполняется $L = 1{,}3\sqrt{n}$ итераций по формуле (11.61), не требующих дополнительных анализов функционирования.

Чтобы получить вектор x' по методу СГ, потребуется N итераций, где число N определяется из условия (11.52): $\|x^N\| = 2t^N = 0{,}23$, т. е. $N \approx -2{,}2 / \ln t$. Для выполнения каждой итерации необходимо обновление вектора градиента, что связано с $2n$ вычислениями $f(x)$. Общее число вычислений f равно $-4{,}4n / \ln t$. Относительный выигрыш в количестве вычислений f методом RELCH по сравнению с методом СГ задается функцией $\psi(\eta) \approx -2{,}2 / (\gamma n \ln t)$. Очевидно, при $\eta \to \infty$ имеем $t(\eta) \to 1$, $\psi(\eta) \to \infty$. Характерные значения ψ для $\gamma = 0{,}01$ и $n = 1000$ даны в табл. 11.2.

Таблица 11.2. *Функция выигрыша*

η	100	1000	1500	10^4	10^5
ψ	1,0	3,4	4,0	11,0	35,0

Таким образом, для получения сравнимых результатов при $\eta = 10^4$ по алгоритму RELCH потребуется приблизительно в 11 раз меньше вычислений f, чем по методу СГ. Следует однако учитывать, что при увеличении η возрастает число L пересчетов по формуле (11.61). Это может приводить к возрастанию вычислительных погрешностей при вычислении ϑ_s с большими номерами s.

Важным дополнительным преимуществом алгоритма RELCH по сравнению с методом СГ является его достаточно высокая эффективность при решении задач с невыпуклыми функционалами, т. к. функция релаксации метода в левой полуплоскости целиком расположена в разрешенной области и множители релаксации для $\lambda > 0$ быстро растут по абсолютному значению при переходе от ϑ_s к ϑ_{s+1}. Характеристики роста были приведены ранее.

Так же как в методе ЭР, можно показать, что эффективность рассматриваемого подхода сохраняется при степенях овражности, удовлетворяющих неравенству $\eta < 1 / (n\varepsilon_M)$. Области работоспособности алгоритмов RELAX, RELCH в плоскости n,η представлены на рис. 11.9.

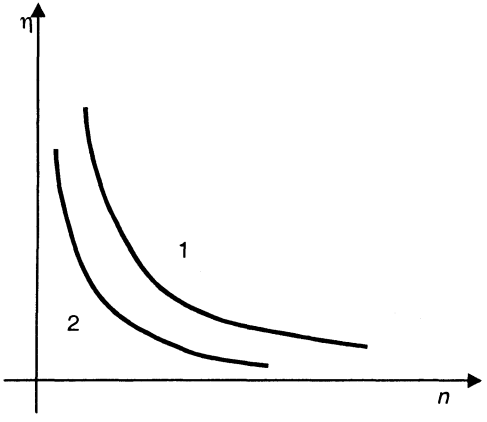


Рис. 11.9. Области работоспособности

Ясно, однако, что при малых размерностях n более эффективными, вообще говоря, оказываются алгоритмы типа RELAX. Они позволяют за меньшее число N_y операций умножения матрицы на вектор получать заданные значения множителей релаксации. При больших η это приводит к существенному уменьшению накопленной вычислительной погрешности.

Для подтверждения данного замечания достаточно проанализировать характер изменения множителей релаксации при применении формул пересчета (11.37) и (11.61). Характерные зависимости для рассмотренных случаев (для фиксированного $\lambda_i > 0$) представлены на рис. 11.10 (1 — RELAX, 2 — RELCH).

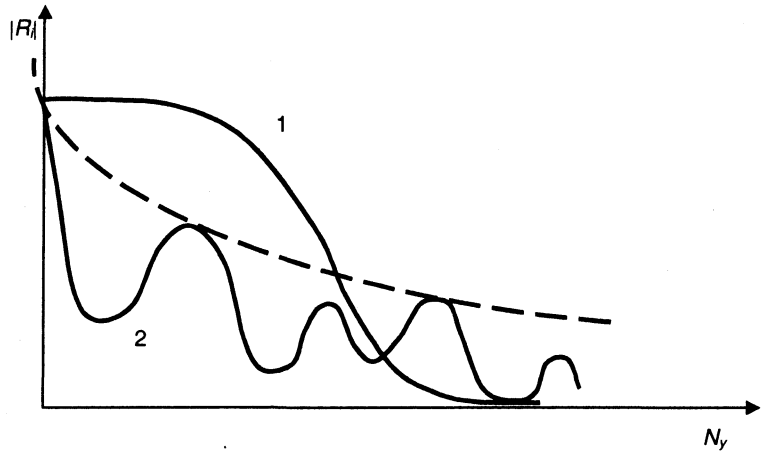
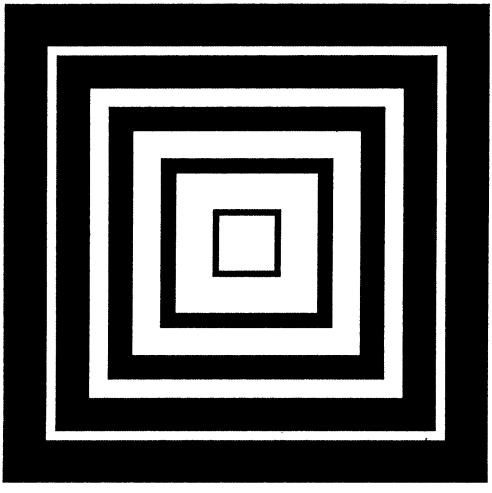


Рис. 11.10. Значения множителей релаксации

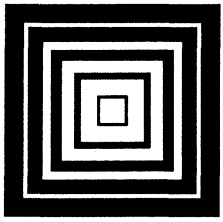
Из рис. 11.10 видно в то же время, что если область локальной квадратичности функционала $J(x)$ невелика (ζ_k мало), то $|R_i| \approx 1$ и более эффективными могут оказаться методы типа RELCH.



ЧАСТЬ III

Экспертные системы принятия решений

Глава 12



Введение

Экспертная система (ЭС) представляет из себя компьютерную программу, позволяющую автоматизировать достоверные рассуждения человека-эксперта в конкретной предметной области. ЭС — это диалоговая система; содержание и форма диалога соответствуют "беседе" эксперта с "заказчиком" или пользователем системы с целью получения экспертных заключений по обсуждаемой проблеме. В результате такой беседы человек-эксперт приходит к определенным выводам и рекомендациям, позволяющим ответить на основной вопрос пользователя. В частности, пользователя может интересовать проблема выбора решения из заданного множества альтернатив. К такому же результату мы приходим, общаясь и с ЭС, которая моделирует поведение человека-эксперта. Традиционным примером предметной области, где целесообразно применение ЭС, является область медицинской диагностики. Основываясь на данных анализов и внешних симптомах проявления болезни, а также на основе имеющейся дополнительной информации ЭС должна указать наиболее правдоподобный диагноз (из заданного множества диагнозов), моделируя рассуждения человека-эксперта, в данном случае — лечащего врача или врача-диагноста.

Основная цель, достигаемая при использовании ЭС, состоит в тиражировании знаний высококвалифицированных экспертов. Это приводит к удешевлению процесса экспертизы (обращение к высококвалифицированному человеку-эксперту не всегда возможно и стоит достаточно дорого), а также, вообще говоря, к повышению достоверности и надежности результатов экспертизы. Последний аспект особенно ярко проявляется в задачах принятия решений в условиях критических ситуаций, когда требуется быстро и безошибочно указать способ поведения реального объекта (например, человека). Как правило, в жестких временных рамках даже высококвалифицированный человек-эксперт (например, офицер наведения ракет в системах противовоздушной обороны) подвержен влиянию различных психологических факторов, затрудняющих процесс выработки рациональных решений. Экспертная же система за счет

своего быстродействия и отсутствия влияния нежелательных человеческих факторов позволяет быстро и непредвзято оценить результаты анализа обстановки и выработать разумную ответную реакцию.

При этом важно понимать, что построение и последующее применение ЭС возможно только при условии наличия эксперта (или группы экспертов), знания которого (или которых) удалось формализовать с помощью соответствующей "базы знаний". Кроме знания основных фактов и данных из конкретной предметной области эксперт владеет своей логикой рассуждений, которая также должна быть отражена при построении ЭС (механизм вывода).

В настоящее время ситуация такова, что существующие и проектируемые ЭС направлены на принятие решений в достаточно узких предметных областях, что, вообще говоря, соответствует практике использования живых экспертов.

Далее рассмотрены основные сведения по внутреннему устройству ЭС. Приводимых сведений достаточно не только для понимания смысла работы реальных ЭС, имеющихся на рынке современных информационных технологий, но и для создания собственных ЭС, предназначенных для решения возникающих специальных задач. Изложенный материал, конечно, не охватывает всего многообразия способов представления знаний и организации процедуры логического вывода. Многие вопросы не рассмотрены. Однако приводимые сведения, по мнению автора, могут явиться основой для более глубокого освоения предмета с помощью широкого спектра специальной литературы.

12.1. Назначение и области применения экспертных систем

В настоящее время системы поддержки принятия решений в виде экспертных систем широко используются в различных областях. Появилась и развивается специальная индустрия по разработке и внедрению ЭС.

Основное назначение ЭС состоит в решении неформализованных задач выбора, являющихся трудными для традиционных методов математического анализа и традиционных методов программирования.

Наибольшее распространение ЭС получили в таких областях, как:

- проектирование заказных интегральных схем;
- автоматизация программирования на основе применения современных CASE-систем и окружений разработки больших программных проектов;

- военные приложения;
- здравоохранение;
- риэлтерская деятельность по подбору и продаже объектов недвижимости; рынок недвижимости;
- финансовый рынок и рынок ценных бумаг;
- автоматизированное комплексирование заказных компьютерных систем, в том числе офисных;
- принятие решений в кризисных ситуациях;
- охрана правопорядка;
- современные информационные образовательные технологии; контроль знаний обучающихся;
- задачи планирования и рационального распределения ресурсов

и т. д.

По своему смыслу многие ЭС, применяемые в вышеперечисленных областях, могут быть отнесены к одному из следующих классов.

Диагностирующие и управляющие системы. Основная задача диагностики может быть в общем виде сформулирована следующим образом. Пусть S — некоторая диагностируемая система (рис. 12.1).

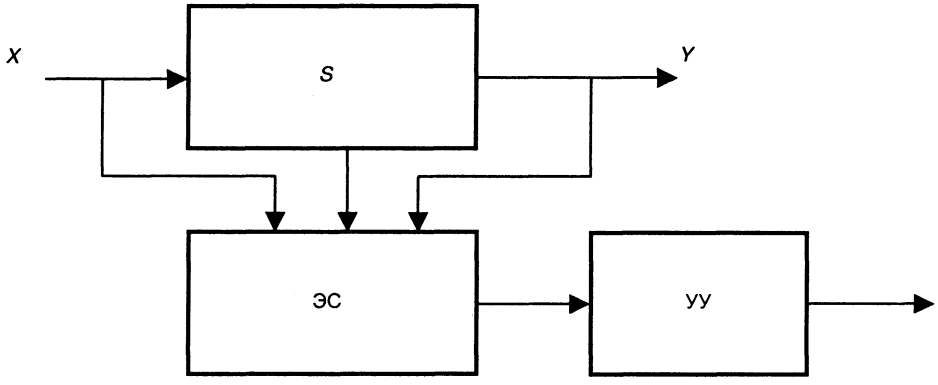


Рис. 12.1. Задача диагностики и управления

X — входные, а Y — выходные сигналы диагностируемой системы. На основе анализа в реальном времени информации о входных и выходных сигналах, а также информации о внутреннем состоянии системы S диагностирующая экспертная система ЭС должна делать заключения о "правильности" функционирования S. При возникновении "нештатных",

критических ситуаций они должны фиксироваться ЭС. Кроме того, должны определяться места "неисправностей" в системе *S* и выдаваться рекомендации устройству управления УУ (это может быть техническое устройство, человек, группа лиц, государственный орган и т. п.) с целью вывода *S* из кризисной ситуации. Таким образом, данная ЭС будет выполнять функции *управления*.

Как уже указывалось, под *S* может пониматься, в частности, пациент некоторого лечебного учреждения, филиал некоторого банка, автомобиль с компьютером на борту или большой программный комплекс, функционирующий в реальном времени. Таким образом, здесь речь идет, по существу, о некоторой системе мониторинга (слежения). В этом случае диагностика состояния *S* и соответствующая интерпретация поступающей информации происходят в реальном времени. ЭС сигнализирует о выходе параметров слежения системы *S* за допустимые пределы, анализирует возможные причины и выдает советы о целесообразной реакции на сложившуюся ситуацию. Подобные системы мониторинга применяются в медицине, экономике, военной области, в системах управления ядерными реакторами и т. д.

В виде соответствующих ЭС существуют и применяются статические диагностирующие системы, например, системы по определению и устранению неисправностей автомобиля при обращении пользователя на станцию технического обслуживания.

Прогнозирующие системы могут также применяться в различных областях. Основная задача прогнозирующей ЭС заключается в анализе развития ситуации (некоторой системы *S*) за определенный отрезок времени и выдаче соответствующих выводов и прогнозов о правдоподобных путях развития этой ситуации в будущем. Например, введение некоторых новых законов в стране требует предварительного анализа возникающих последствий и оценки желательности этих последствий. Точно так же анализ определенных изменений на бирже позволяет указать возможное развитие ситуации в будущем, что может повлиять на принимаемые в настоящий момент решения. Безусловно, существуют прогнозирующие системы, основанные на регулярных математических методах, связанных, в частности, с методами анализа временных рядов, однако в ряде случаев прогноз может быть осуществлен только на основе знаний экспертов в конкретной предметной области, и это область применимости ЭС.

Планирующие системы предназначены для создания плана реализации последовательности действий для достижения поставленных целей. Примерами ЭС, занятых планированием, могут служить системы формирования плана проведения боевой операции в заданных условиях или системы создания плана действий при комплексировании сложной ин-

формационной (например, телекоммуникационной) системы по основным требованиям, сформулированным заказчиком.

В качестве примера планирующей экспертной системы можно также привести ЭС, обслуживающую какое-то количество рабочих станций (терминалов) в торговом зале и позволяющую покупателям компьютеров спланировать покупку — выбрать в диалоговом режиме конфигурацию компьютера, в наибольшей степени соответствующую целям и финансовым возможностям каждого отдельного покупателя.

Аналогичные ЭС могут применяться при продаже любых достаточно сложных объектов и услуг, например, на рынке недвижимости.

Интерпретирующие (анализирующие) системы осуществляют анализ ("расшифровку") поступающей информации о состоянии некоторой системы или объекта и затем дают описание реальной ситуации на стандартном для данной прикладной области языке. Например, военные интерпретирующие системы могут использоваться для идентификации целей на основе данных радиолокационной разведки.

Можно и далее перечислять области применения и функции ЭС. Ясно, что приведенная классификация является в достаточной степени условной и неполной, и одну и ту же ЭС можно описать (проинтерпретировать!) с различных позиций и отнести сразу к нескольким классам. Однако бесспорно, что области практического применения ЭС могут быть связаны с такими ключевыми словами, как интерпретация, управление, диагностика, прогноз, проектирование, планирование, наблюдение, отладка, ремонт, обучение и т. д.

12.2. Структура экспертной системы

ЭС содержит следующие основные компоненты:

- база знаний;
- механизм вывода (средство компьютерного мышления).

Основной процесс заключается в применении механизма вывода к исходным знаниям с целью получения результирующих знаний, представляющих интерес для пользователя ЭС.

Кроме основных компонентов ЭС включает дополнительные подсистемы, обеспечивающие: общение с пользователем, перенос знаний от эксперта в компьютерную программу, объяснение и обоснование результатов вывода и т. д.

Типовая структура ЭС и схема взаимодействия участников процесса построения и использования ЭС представлены на рис. 12.2. Собственно ЭС обведена пунктиром.

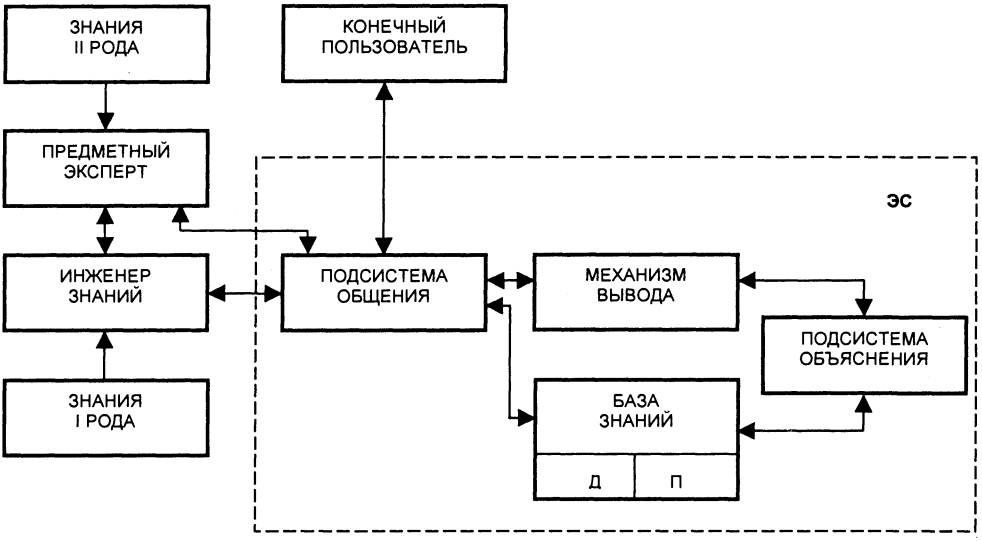


Рис. 12.2. Структура экспертной системы

Знания, которыми владеет эксперт в конкретной предметной области, делятся на *декларативные* (Д) и *процедурные* (П).

Декларативные знания (или *факты*) дают описание фактов и явлений внешнего мира, относительно которых можно установить, есть они в наличии или нет. Например, у больного температура может быть повышенной или нормальной ("есть температура или нет", как обычно говорят). Данный конкретный человек может иметь высшее образование или не иметь и т. п.

Процедурные знания заключаются в правилах манипулирования фактами для получения заключений, приводящих к новым знаниям (как декларативным, так, возможно, и процедурным). Весьма распространенная форма представления процедурных знаний связана с уже упоминавшейся в этой книге *продукцией* ЕСЛИ ... ТО... . Например, запись ЕСЛИ *A* ТО *B*, где *A* и *B* — факты, позволяет по факту *A* установить наличие факта *B*, если указанная продукция (элемент множества процедурных знаний) присутствует в базе знаний ЭС.

На рис. 12.2 представлены знания первого (I) и второго (II) рода. К *знаниям первого рода* относятся общезначимые, общеизвестные декларативные и

процедурные знания, например, отражающие законы сохранения в физике. Такие знания доступны не только эксперту в данной предметной области (предметному эксперту), но и, в частности, инженеру знаний. *Знания второго рода* являются в определенном смысле более ценными. Они включают различные "know-how", эмпирические и интуитивные соображения, которыми владеет данный предметный эксперт. Так же как и знания первого рода, эти знания могут быть как декларативными, так и процедурными.

Предметный эксперт, или просто *эксперт*, — это человек, являющийся признанным специалистом в конкретной предметной области и умеющий (а главное — желающий) ясно объяснить свои методы, приемы и стратегии решения проблем. В ЭС, как правило, моделируются знания одного или нескольких экспертов, а также используются дополнительные доступные знания.

Инженер знаний (инженер по знаниям) должен являться глубоким специалистом в области ЭС и, в частности, должен владеть полной информацией о конкретной ЭС, т. к. именно он осуществляет перенос знаний эксперта в ЭС при ее построении и настройке на конкретную предметную область. Инженер знаний общается с экспертами и форматирует полученные знания для их введения в базу знаний ЭС. Кроме того, желательно, чтобы инженер знаний имел достаточно высокий научный и интеллектуальный потенциал для быстрой адаптации к конкретной предметной области. Инженер знаний участвует в разработке конкретной ЭС и в ее последующем сопровождении.

Конечный пользователь использует ЭС по прямому назначению — для получения ответов на свои вопросы из области компетентности данной ЭС. Важное обстоятельство заключается в том, что пользователь, в отличие от инженера знаний, может не быть специалистом в области информатики. Он общается с ЭС через *подсистему общения* на языке, максимально приближенном к профессиональному языку в конкретной области экспертизы. Тем более от него не требуется знаний в области программирования вообще и программирования ЭС в частности.

Как устроены базы знаний и механизм вывода (иногда говорят — машины логического вывода) показано далее при рассмотрении конкретных ЭС.

12.3. Основные классы и виды экспертных систем

Классификацию объектов производят по каким-либо признакам. В данном случае мы кратко на эвристическом уровне рассмотрим основные виды ЭС в зависимости от методов представления знаний.

Методы, основанные на правилах, — продукционные экспертные системы. Как уже отмечалось, продукционные системы в качестве базы процедурных знаний имеют набор продукций (правил) вида ЕСЛИ *A* ТО *B*, где *A* и *B* — элементы множества декларативных знаний. (В ряде случаев могут быть использованы и более сложные продукционные структуры.) Здесь *A* называется *условием*, а *B* — следствием. Если факт *A* породил *B*, то теперь уже *B* может выступать как условие в новой продукции и т. д. Организуется так называемая цепочка логического вывода, которая заканчивается фактом или фактами, играющими роль результатов экспертизы.

С помощью правил-продукций в отличие от традиционных приемов программирования удается реализовать более гибкую стратегию организации ветвления программы (передачи управления), которое должно управляться самими данными. Кроме того, при этом достигается логическая цельность и ясность программы, что важно как для понимания ее работы во время создания самого программного продукта, так и для его последующей модификации в процессе эксплуатации.

Более подробно способы представления знаний в продукционных ЭС, а также различные механизмы вывода рассмотрены в *главе 13*.

Фреймы (фреймовые системы). Под фреймовыми системами понимаются ЭС, основанные на специальных методах представления знаний в виде *объектов* и *отношений* между объектами. Представление знаний, основанное на фреймах, является по сравнению с продукционными методами альтернативным способом структурирования, хранения и обработки знаний.

По существу речь идет о хорошо известной технологии объектно-ориентированного программирования, применяемой для целей создания ЭС. Основными понятиями являются понятия структуры, объекта, слота, атрибута, значения.

Структура дает общее описание *объекта* с указанием списка *атрибутов* как имен *слотов* (мест хранения информации). При этом объект выступает как некоторая конкретизация структуры, содержащая уже конкретную информацию (значения) по всем слотам.

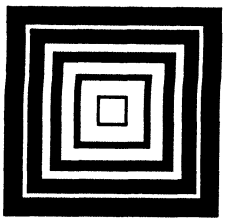
Организация представления знаний на основе объектно-ориентированного подхода имеет ряд особенностей, важнейшим из которых является *принцип наследования*. Во фреймовой ЭС предметная область описывается некоторой иерархической фреймовой структурой, в которой фреймы, занимающие более низкое положение в иерархии, наследуют свойства фреймов более высокого уровня. Это, в частности, позволяет экономить память компьютера (т. к. исключается дублирование при описании свойств объектов), а также уменьшать вероятность возникновения ошибок и противоречий в системе знаний.

К недостаткам фреймовых систем обычно относят их относительно высокую сложность и низкое быстродействие. Кроме того, достаточно сложным оказывается процесс изменения принятой иерархической структуры (родовидовой иерархии).

Более подробное рассмотрение данных вопросов выходит за пределы данной книги.

Прочие методы. Существует множество других концепций построения ЭС, например, семантические сети и нейлоровские диагностирующие системы. Семантические сети тесно связаны с идеями фреймового представления знаний и в этой книге не рассматриваются, а нейлоровские системы, основанные на принципиально ином подходе, далее рассмотрены подробно вплоть до описания работающей ЭС.

Глава 13



Продукционные экспертные системы

В данном разделе рассмотрены основные принципы построения продукционных экспертных систем. Реальные продукционные экспертные системы могут иметь существенно более сложную структуру, их описание можно найти в специальной литературе.

13.1. Основные компоненты продукционной экспертной системы

В соответствии со структурой типовой ЭС (см. рис. 12.2) рассмотрим особенности представления знаний и механизмы вывода в продукционных системах. Основные компоненты продукционной ЭС изображены на рис. 13.1.

Механизм вывода часто называется *интерпретатором правил* или *планировщиком*. Правила — продукции $\alpha_i \to \beta_i$ интерпретируется с помощью конструкции:

$$\text{ЕСЛИ } \alpha_i \text{ ТО } \beta_i.$$

Здесь α_i и β_i могут достаточно сложным образом зависеть от фактов a_i. Например, можем иметь продукцию вида

$$a_1 \wedge a_3 \wedge a_5 \wedge a_9 \to a_{10},$$

где $\wedge$ — знак конъюнкции (логическое "И").

При решении задач оптимизации с помощью некоторой диалоговой системы оптимизации или пакета прикладных программ для выбора метода

конечномерной оптимизации без ограничений может применяться ЭС, содержащая, например, следующее правило:

ЕСЛИ решаемая задача = задача конечномерной оптимизации без ограничений,

И *количество переменных* = 2,

ТО *рекомендуемый метод = метод вращения осей Розенброка.*

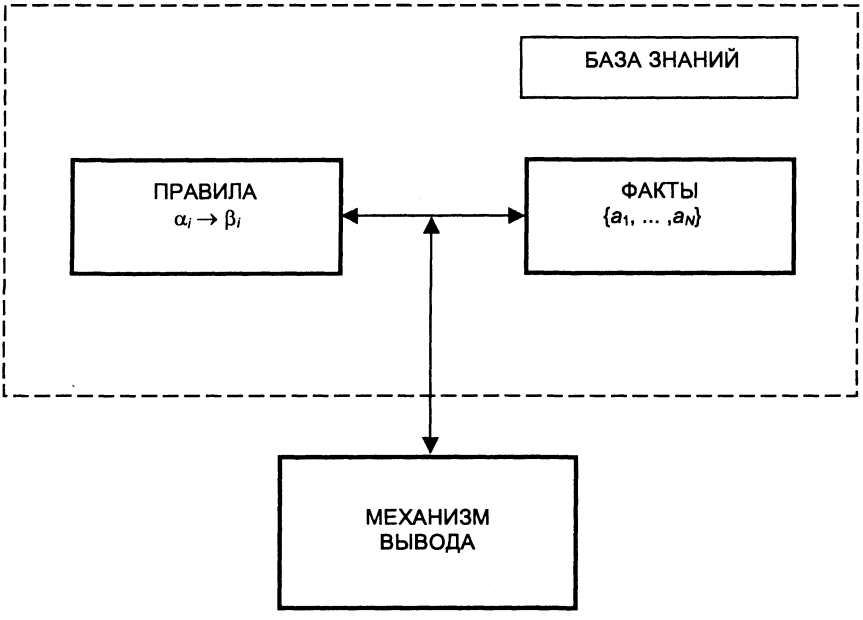


Рис. 13.1. Продукционная экспертная система

Механизм вывода в продукционной ЭС может быть построен в соответствии с различными принципами. Рассмотрим основную идею.

Мы имеем базовое (неизменное) для данной ЭС множество возможных фактов

$$A = \{a_1, \ldots, a_n\},$$

формируемое при создании (разработке) ЭС. При этом важно понимать, что никакие новые факты не могут быть получены в результате работы ЭС. Основная и единственная задача ЭС — устанавливать определенные связи между фактами для конкретной ситуации, интересующей пользователя ЭС.

Будем различать два подмножества A_0, A_1

$$A = A_0 \cup A_1,\ A_0 \cap A_1 = \varnothing$$

исходного множества A. Множество A_1 будет называться *множеством констатированных (или помеченных) фактов*, а множество A_0 — *множеством непомеченных фактов*. Иногда множество A_1 называется *рабочим полем* ЭС.

В начале работы ЭС множество A_1 содержит некоторое количество фактов (исходная информация), например,

$$A_1 = \{a_1, a_2, a_3\}.$$

Далее происходит последовательное пополнение множества A_1 за счет элементов множества A_0. Интерпретатор правил сопоставляет левые части продукций

$$\alpha_i \rightarrow \beta_i$$

с имеющимися во множестве A_1 фактами и выполняет то правило, левая часть которого α_i согласуется с фактами из A_1 (оказывается истинной). В результате множество A_1 пополняется за счет фактов, констатируемых в правой части продукции β_i.

Пример 13.1. Пусть исходное множество A_1 состоит из трех элементов:

$$A_1 = \{a_1, a_2, a_3\}.$$

Интерпретатором правил в базе знаний найдена продукция

$$P_k\colon a_1 \wedge a_2 \wedge a_3 \rightarrow a_4 \wedge a_5.$$

Левая часть оказывается согласованной с множеством A_1, продукция выполняется и новое множество A_1 имеет состав

$$A_1 = \{a_1, a_2, a_3, a_4, a_5\}.$$

Далее процесс повторяется — проверяется согласованность с A_1 оставшихся продукций.

Основная задача продукционной ЭС заключается в определении последовательности правил-продукций, позволяющей по исходным фактам получить интересующий пользователя факт.

Процесс сопоставления левых частей продукций α_i с множеством констатированных фактов A_1 порождает *цепочку вывода*, или *цепочку рассуждений*. Эту цепочку можно изобразить графически (рис. 13.2).

Представленная на рисунке цепочка соответствует последовательному применению продукций

$$a_2 \rightarrow a_4$$

$$a_4 \wedge a_3 \rightarrow a_5$$

к исходному множеству $A_1 = \{a_1, a_2, a_3\}$. Возможно, новый факт a_5 и есть результат, интересующий пользователя.

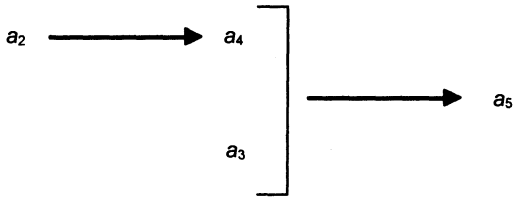


Рис. 13.2. Цепочка вывода

В реальных продукционных ЭС начальное множество A_1 (множество известных пользователю фактов) формируется не однократно, в начале решения задачи, а последовательно, изменяясь в процессе диалога. Таким образом, пополнение A_1 происходит в реальном времени и не только в результате выполнения последовательности продукций, но и как следствие диалога с пользователем. Так, например, результат выполнения некоторого правила может означать некоторое взаимодействие с внешней средой, в том числе и вопрос к пользователю, ответ на который приводит к пополнению множества A_1.

13.2. Прямая и обратная цепочки вывода

Выше мы рассмотрели процесс модификации множества A_1 в соответствии с *прямой цепочкой вывода*. Основная идея заключалась в поиске новых фактов (новой информации) в направлении стрелок, разделяющих левые и правые части правил:

$$\alpha_i \rightarrow \beta_i.$$

Рассмотрим теперь более подробно на конкретном примере, как происходит вывод при прямой и обратной стратегиях вывода. Мы здесь предполагаем, что интерпретатор правил анализирует список правил сверху вниз и выполняет первое же правило, левая часть которого согласуется с множеством текущих фактов A_1.

Пример прямой цепочки вывода. Пусть

$$A_1 = \{a_1, a_2, a_3, a_5, a_7, a_8\},$$

список продукций:

$$P_1 : a_6 \wedge a_2 \rightarrow a_9,$$

$$P_2 : a_3 \wedge a_4 \rightarrow a_6,$$

$$P_3: \quad a_1 \rightarrow a_4.$$

Шаг 1. Просматриваем список продукций сверху вниз, сопоставляя левые части с элементами множества A_1. В результате выполнения P_3 имеем:

$$A_1 = \{a_1, a_2, a_3, a_4, a_5, a_7, a_8\},$$

$$A_1 := A_1 \cup a_4.$$

(Добавлен элемент a_4.)

Шаг 2. Снова просматриваем список правил сверху вниз. На этот раз соблюдаются условия для выполнения продукции P_2:

$$A_1 := A_1 \cup a_6 = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8\}$$

(здесь осуществляется добавление a_6 к последнему варианту множества A_1, полученного на шаге 1).

Шаг 3. Выполняется продукция P_1 и

$$A_1 := A_1 \cup a_9 = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9\}.$$

Цепочка соответствующих рассуждений построена на рис. 13.3.

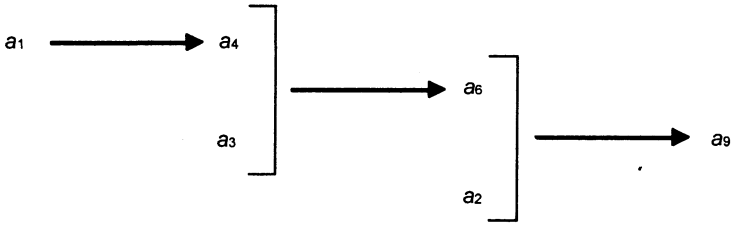


Рис. 13.3. Прямая цепочка вывода

Основной результат: было выведено, что наряду с исходными фактами (ситуациями) существуют ситуации a_4, a_6, a_9. Например, исходные факты могут быть результатом анализов, а факты a_4, a_6, a_9 (или только a_9!) — возможные диагнозы (виды заболеваний).

Предположим теперь, что мы хотим использовать ЭС с этой же самой базой знаний, чтобы установить, существует ли ситуация a_9 (например, болен ли человек конкретной болезнью).

На первый взгляд кажется, что мы уже установили этот факт. Да, это так в данном примере. Но мы попутно установили и другие новые факты (a_4 и a_6), которые теперь нас не интересуют. И если реальная база знаний содержит не три правила, а сотни и тысячи, то в результате прямой цепочки рассуждений будет проделана огромная лишняя (в смысле задачи установления конкретного факта) работа, связанная с построением

большого числа вполне справедливых цепочек вывода и ситуаций, не имеющих отношения к искомому результату.

Поэтому в подобных случаях более выгодной может оказаться *обратная цепочка вывода*. Возвращаясь к предыдущему примеру, заметим, что основная цель состоит в доказательстве существования ситуации (факта) a_9, и мы теперь будем выполнять только те правила, которые относятся к установлению этого факта.

Пример обратной цепочки вывода. Пусть

$$A_1 = \{a_1, a_2, a_3, a_5, a_7, a_8\};$$

$$P_1 : a_6 \wedge a_2 \to a_9;$$

$$P_2 : a_3 \wedge a_4 \to a_6;$$

$$P_3 : \quad a_1 \to a_4.$$

Шаг 1. Системе сообщается, чтобы она подтвердила (установила) существование ситуации a_9. Сначала проверяется наличие a_9 во множестве A_1. Если его нет, то в списке правил P_i имеется правило вида

$$\alpha_i \to a_9 \cup \beta_i.$$

Система находит правило

$$P_1 : a_6 \wedge a_2 \to a_9$$

и решает, что теперь необходимо установить наличие фактов a_2 и a_6, чтобы вывести a_9. (Таких правил, вообще говоря, может быть несколько и процесс разветвляется.)

Шаг 2. Имеем $a_2 \in A_1$, $a_6 \notin A_1$. Находим правило

$$P_2 : a_3 \wedge a_4 \to a_6$$

и задача сводится к установлению a_3, a_4.

Шаг 3. Имеем $a_3 \in A_1$, $a_4 \notin A_1$. Находим правило

$$P_3 : a_1 \to a_4$$

и задача сводится к установлению a_1. Но $a_1 \in A_1$ и поэтому задача установления a_9 решена.

ЭС ставит диагноз: a_9. Основная цель — факт существования a_9 — достигнута.

Как уже указывалось, ЭС является диалоговой системой и поэтому получение начальных данных и сам процесс вывода сопровождаются диалогом с пользователем.

В только что рассмотренных примерах диалог присутствует (может присутствовать) на этапах установления наличия тех или иных фактов в базе знаний (во множестве A_1). Например, для обратной цепочки вывода целесообразны следующие вопросы пользователю:

1. Существует ли a_2? (Шаг 2).

   Ответ: да ($a_2 \in A_1$).

2. Существует ли a_6? (Шаг 2).

   Ответ: нет ($a_6 \notin A_1$).

3. Существует ли a_3? (Шаг 3).

   Ответ: да ($a_3 \in A_1$).

4. Существует ли a_4? (Шаг 3).

   Ответ: нет ($a_4 \notin A_1$).

5. Существует ли a_1?

   Ответ: да ($a_1 \in A_1$).

Ясно, что если некоторые факты являются общезначимыми (знания I рода), то они автоматически присутствуют в базе знаний (в A_1), и соответствующие вопросы не задаются.

13.3. Простая диагностирующая экспертная система

При построении ЭС, как мы видели, возникает проблема организации диалога с пользователем. Диалог должен быть организован таким образом, чтобы задаваемые вопросы поступали к пользователю в нужное время и выглядели бы естественными для сложившейся в процессе вывода ситуации. Поэтому техника ведения диалога должна быть тщательно продумана на стадии создания ЭС. Ниже излагается один из известных (Г. С. Поспелов) подходов к созданию диагностирующих продукционных ЭС. Соответствующие интерпретации могут быть весьма разнообразными и поэтому область применимости обсуждаемых конструкций оказывается достаточно широкой.

Задано множество фактов

$$A = \{a_{ij}\} \cup \{q_i\} = (a_1, a_2, ..., a_n),$$

состоящее из элементов двух типов. Элементы a_{ij} определяют обычные декларативные знания из конкретной предметной области. Элементы q_i определяют вид взаимодействия с внешней средой и в данном случае представляют собой вопросы пользователю в виде альтернативного меню:

$$q_i = a_{i_1}, ..., a_{i_s}$$

Некоторые из q_i имеют другой смысл — результирующих заключений или диагнозов, оформленных в виде соответствующих сообщений пользователю.

Продукции в данной системе имеют вид

$$a_{ij} \rightarrow q_m = \{a_{m_1}, ..., a_{m_k}\}.$$

Все множество фактов и продукций организованы в некоторую систему, представленную в виде графа "ИЛИ". Фрагмент такого диагностирующего графа с вершинами-диагнозами q_9, q_{10}, q_{11}, q_{12} (терминальными вершинами) представлен на рис. 13.4.

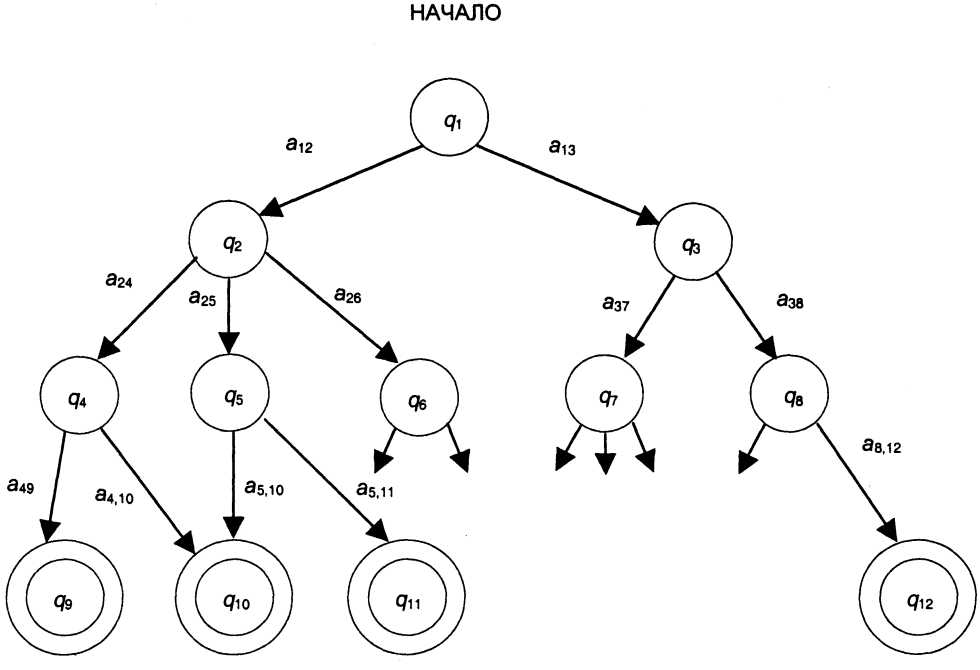


Рис. 13.4. Диагностирующий граф

Принцип работы такой ЭС заключается в следующем. После обращения пользователя к ЭС мы попадаем в вершину q_1, инициирующую вопрос пользователю в виде соответствующего альтернативного меню:

$$q_1 = \{a_{12}, a_{13}, ...\}.$$

Допустим, на вопрос системы: "Какой из фактов a_{12}, a_{13}, ... имеет место?" пользователь ответил: a_{12}. В результате мы заносим a_{12} в рабочее поле и попадаем в новую вершину — вопрос q_2:

$$q_2 = \{a_{24}, a_{25}, a_{26}\},$$

где ситуация повторяется. В конце концов, мы оказываемся в одной из терминальных вершин, где пользователь получает сообщение о результате, характеризующим выбранный ЭС диагноз.

При такой структуре ЭС достаточно просто может быть реализована *подсистема объяснений* как важнейшая составная часть любой ЭС. Для этого достаточно каждой вершине q_i графа сопоставить соответствующий текст, описывающий мотивации выбора в данной вершине. А далее происходит вывод этих текстов при движении снизу (от результата) вверх в соответствии с реализованной цепочкой рассуждений. Для ответа на вопросы пользователя типа "Почему q_i, а не q_j" система, двигаясь по цепочке вывода снизу вверх, определяет место разветвления (какую-то вершину q_k) путей, ведущих к q_i и q_j. Пояснительный текст, ассоциированный с вершиной q_k, и должен содержать ответ на вопрос пользователя. Например, на вопрос "Почему q_9, а не q_{12}?" система ответит "Потому что a_{12}, а не a_{13}", а на вопрос "Почему q_9, а не q_{11}?" получим ответ "Потому что a_{24}, а не a_{25}" и т. п.

Указанная структуризация базы знаний ЭС в данном случае оказывается более естественной и логически оправданной, чем, например, непосредственное использование продукций, построенных в соответствии с различными путями, ведущих от начала процесса к каждой из терминальных вершин, например:

$$a_{12} \wedge a_{24} \wedge a_{49} \rightarrow q_9;$$

$$a_{12} \wedge a_{24} \wedge a_{4,10} \rightarrow q_{10};$$

$$a_{12} \wedge a_{25} \wedge a_{5,10} \rightarrow q_{10}.$$

Мы рассмотрели лишь "принципиальную схему" соответствующей ЭС. Ясно, что в действительности приходится решать много достаточно сложных вспомогательных задач. В частности, важная проблема заключается в разработке и реализации механизма пополнения и модификации знаний, представленных в виде такого диагностирующего графа.

13.4. Формальное представление продукционной экспертной системы

Формальные модели продукционных ЭС играют большую роль не только для изучения конкретных ЭС, но и при построении новых ЭС. В частности, на основе таких формализмов могут изучаться вопросы эффективности различных механизмов вывода, вопросы непротиворечивости и полноты знаний и т. д. Рассмотрим один из возможных подходов.

Будем предполагать, что база знаний ЭС состоит из конечного набора правил:

$$P = \{P_1, ..., P_m\};$$

$$P_i : a_{i_1} \wedge a_{i_2} \wedge ... \wedge a_{i_k} \rightarrow a_j \qquad (13.1)$$

и мысленно возможного конечного набора фактов (ситуаций)

$$A = \{a_1, ..., a_n\}.$$

Часто можно считать, что в правой части импликации (13.1) находится один факт a. Например, при наличии нескольких фактов, объединенных логическим "И", можно соответственно увеличить число импликаций с одной и той же левой частью. Как мы уже видели, в общем случае продукции могут иметь более сложную структуру.

Задача продукционной ЭС — определение цепочки правил, позволяющей получить (подтвердить) интересующий пользователя факт (конечно, из множества A). Как мы видели, процедуры построения этой цепочки могут быть различными — прямой вывод, обратный вывод. Возможен и смешанный вывод. Но в любом случае нас интересует результирующая цепочка от исходных данных к выводимому факту.

Рассмотрим далее процедуру прямого вывода.

С учетом информации, поступающей от пользователя, каждое правило устанавливает новый факт $a_m \in A$, расширяя тем самым набор установленных фактов, находящихся в рабочем поле A_1.

Применимость любого следующего правила зависит только от состояния рабочего поля A_1 (с учетом фактов, введенных пользователем). Множество A_1 рассматривается при этом как состояние самой ЭС, а продукции являются операторами, изменяющими это состояние.

Состояние ЭС описывается с помощью вектора состояния

$$x = (x^1, ..., x^i, ..., x^n),$$

где:

n — количество элементов базового множества фактов A; $x_i = 1$, если $a_i \in A_1$ и $x_i = 0$, если $a_i \in A \setminus A_1$ (факт не установлен).

Продукция

$$P_i : a_{i_1} \wedge a_{i_2} \wedge \ldots \wedge a_{i_k} \rightarrow a_j$$

приводит систему из состояния x в новое состояние, если она применима, т. е. если

$$x_{i_1} \cdot x_{i_2} \cdot \ldots \cdot x_{i_k} = 1.$$

В этом случае получаем новый вектор состояния

$$y = (y_1, \ldots, y_n), \quad P_i x = y,$$

где $y_k = x_k$ при $k \neq j$; $y_j = 1$. (Если уже имели $a_j \in A_1$, то $y = x$ и несмотря на применимость продукции состояние не меняется.)

Если

$$x_{i_1} \cdot x_{i_2} \cdot \ldots \cdot x_{i_k} = 0$$

(что соответствует случаю, когда хотя бы один из фактов $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$ не находится в рабочем поле A_1), то

$$P_i x = x,$$

т. е. состояние не меняется. Продукция неприменима.

Пример 13.2. Пусть имеется список фактов

$$A_1 = \{a_1, a_2, \ldots, a_6\}$$

и три продукции:

$$P_1 : a_2 \wedge a_3 \rightarrow a_5;$$

$$P_2 : a_2 \wedge a_4 \wedge a_5 \rightarrow a_6;$$

$$P_3 : a_6 \rightarrow a_1.$$

Начальное состояние системы

$$x^0 = (011100).$$

Рассмотрим изменение состояния системы при последовательном применении указанных продукций:

$$x^1 = P_1 x^0 = (011110);$$

$$x^2 = P_2 x^1 = (011111);$$

$$x^3 = P_3 x^2 = (111111).$$

Легко видеть, что, например, продукция вида $a_1 \wedge a_2 \rightarrow a_3$ неприменима в состоянии x^0 и, следовательно, не меняет этого состояния. Продукция же $a_2 \wedge a_3 \rightarrow a_4$ применима, но также не меняет состояния x^0.

Указанный формализм позволяет трактовать процесс вывода заключений в продукционной ЭС как процесс эволюции некоторой динамической системы в дискретном времени t:

$$x^{t+1} = P_t x^t,$$

где $\{x^1, x^2, ...\}$ — последовательность состояний ЭС, а $\{P_1, P_2, ...\}$ — последовательность примененных продукций.

Предполагается, что на каждом шаге вывода перед поиском очередной продукции в базе процедурных знаний P система имеет возможность задать вопрос пользователю (так, как это было описано ранее при рассмотрении примеров) для установления дополнительных фактов из A. Стратегия ведения диалога с пользователем определяется при создании конкретных ЭС, настроенных на ту или иную предметную область.

С учетом указанной интерпретации процедуры вывода и привлекая концепцию динамической системы, можно проводить достаточно полный анализ функционирования продукционных ЭС.

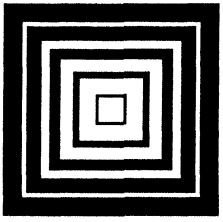
Сама цель работы ЭС формализуется как оценка возможности перехода динамической системы из заданного множества начальных состояний x_0 (в частном случае это может быть одно состояние) в некоторое целевое множество

$$T = \left\{ x^i \middle| x_{i_1} \vee x_{i_2} \vee ... \vee x_{i_k} = 1 \right\}$$

состояний, в которых оказывается установленным хотя бы один из фактов заданного множества B:

$$B = \left(a_{i_1}, a_{i_2}, ..., a_{i_k} \right), \; B \subset A.$$

При этом могут быть привлечены важные факты из теории управляемых динамических систем с целью оптимизации траектории вывода. С помощью данного аппарата решается также важная проблема порядка выбора правил из базы P на каждом шаге вывода, а также проблемы противоречивости знаний. Более подробное рассмотрение этих вопросов выходит за рамки этой книги *(см. Список литературы)*.



Глава 14

Представление и использование нечетких знаний

Одна из проблем, возникающая при создании ЭС, состоит в учете неточности и ненадежности любой информации. Аналогичные проблемы возникают и в других областях компьютерного моделирования, например, в методах численного анализа. Там ставится задача оценки погрешности численного результата при известных погрешностях исходных данных и неточности реализации самого алгоритма (вычислений). Иначе говоря, определяется характер "переноса" погрешностей на результат. Точно так же в ЭС возникает задача оценки степени ненадежности наших выводов и рекомендаций при неточности исходной информации и информации, поступающей в результате диалога с пользователем.

Далее мы рассмотрим один из способов рассуждения при наличии неопределенности, основанный на формуле Байеса теории вероятностей.

14.1. Элементы теории вероятностей

Вспомним необходимые базовые понятия элементарной теории вероятностей.

Пусть σ — некоторый комплекс условий, при осуществлении которого могут происходить события из множества

$$S = \{A,B,C, \ldots\}.$$

Множество S называется *системой случайных событий.*

Если появление события A обязательно сопровождается появлением события B, то говорят, что A *влечет* B; обозначается $A \subset B$.

Если $A \subset B$ и $B \subset A$, то говорят, что события A и B *равносильны*; равносильность событий обозначается как $A = B$. С позиций теории вероятностей такие события неразличимы.

Событие, состоящее в наступлении обоих событий A и B, называется *произведением событий* A и B и обозначается как AB или $A \cap B$.

Событие, состоящее в наступлении хотя бы одного из событий A и B, называется *суммой событий* A и B и обозначается как $A + B$ или $A \cup B$.

Событие, состоящее в том, что A происходит, а B не происходит, называется *разностью событий* A и B и обозначается как $A - B$.

Два события A и $\overline{A}$ называются *противоположными*, если $A+\overline{A}=U$, $A\overline{A}=V$, где U — достоверное событие (происходит всегда при реализации комплекса условий σ); V — невозможное событие.

Два события A и B называются *несовместимыми*, если их совместное появление невозможно: $AB = V$.

При построении математической теории вероятностей исходят из базового понятия пространства (множества) элементарных событий, отождествляя его с достоверным событием U. Природа элементов этого множества заранее не оговаривается. Различные подмножества точек из U и образуют систему событий S.

Пример 14.1. Пусть комплекс условий σ состоит в том, что внутри единичного квадрата наудачу выбирается точка. Тогда все множество точек, расположенных в указанном квадрате, образует множество U. Пусть теперь A и B — события, состоящие в попадании точки внутрь левого и правого кругов соответственно (рис. 14.1).

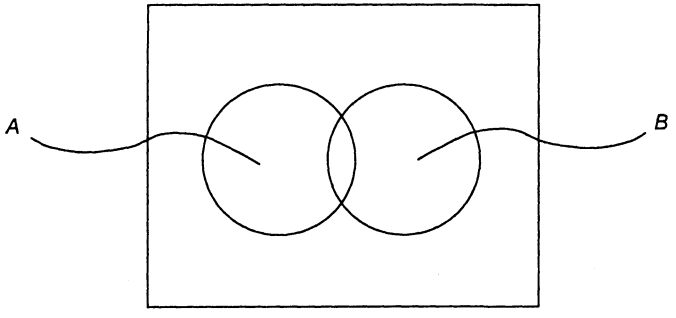


Рис. 14.1. Графическое представление двух событий

Тогда события A, B, $\overline{A}$, $\overline{B}$, $A + B$, AB, U, V могут быть представлены в виде соответствующих заштрихованных подмножеств множества U (рис. 14.2).

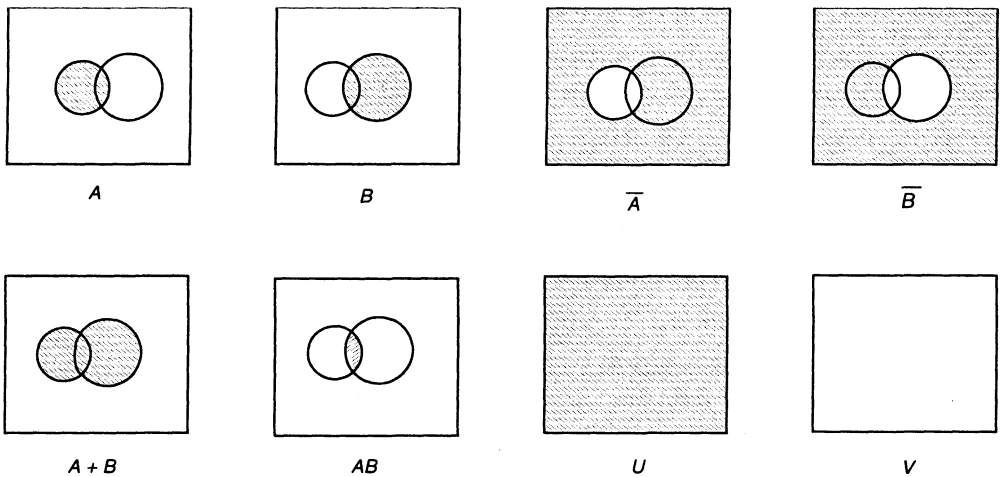


Рис. 14.2. Операции над событиями

Каждому случайному событию A ставится в соответствие неотрицательное число $P(A) \in [0,1]$ — его вероятность. В примере 14.1 вероятность события равна площади соответствующей фигуры.

Справедливы следующие основные законы теории вероятностей:

- $P(\overline{A}) = 1 - P(A)$, где $P(A)$ — вероятность наступления события A, а $P(\overline{A})$ — вероятность его ненаступления;
- $P(A + B) = P(A) + P(B) - P(AB)$;
- $P(AB) = P(A) \cdot P(B / A) = P(B) \cdot P(A / B)$.

  Здесь $P(B / A)$ означает вероятность появления события B при условии наступления события A (условная вероятность);
- $P(U) = 1$, $P(V) = 0$.

Говорят, что событие A *не зависит (независимо)* от события B, если

$$P(A / B) = P(A).$$

Данное равенство означает, что наступление события B не изменяет вероятность наступления события A. Оказывается, что если A не зависит от

B, то и B не зависит от A. Таким образом, в случае независимости событий A и B имеем

$$P(AB) = P(A) \cdot P(B).$$

Выведем теперь основное для наших целей соотношение.

Пусть A и B — два события. Тогда в силу приведенных ранее соотношений получим

$$P(A/B) = \frac{P(AB)}{P(B)}, \quad P(B/A) = \frac{P(AB)}{P(A)}$$

или, исключая $P(AB)$,

$$P(A/B) = \frac{P(B/A)P(A)}{P(B)}.$$

Кроме того, поскольку события A и $\overline{A}$ несовместны, и одно из них обязательно происходит, имеем

$$P(B) = P(B/A)\,P(A) + P(B/\overline{A})\,P(\overline{A}).$$

В результате имеем искомую *формулу (теорему) Байеса*:

$$P(A/B) = \frac{P(B/A)P(A)}{P(B/A)P(A) + P(B/\overline{A})P(\overline{A})}.$$

14.2. Байесовский подход

Одним из исчислений неопределенностей в теории экспертных систем является теория вероятностей и теорема Байеса в частности. С помощью формулы Байеса удается накапливать информацию, поступающую из различных источников, с целью подтверждения или неподтверждения определенной гипотезы (диагноза).

Пусть имеется некоторая гипотеза H и некоторая априорная вероятность того, что гипотеза H истинна. Эта вероятность $P(H)$ либо задается в самом начале как исходное данное, либо является результатом предыдущих преобразований. Далее предполагается, что появляется некоторое свидетельство E, относящееся к данной гипотезе, и мы хотим на основе этой информации уточнить априорную вероятность истинности гипотезы H. Согласно формуле Байеса имеем:

$$P(H/E) = \frac{P(E/H)P(H)}{P(E/H)P(H) + P(E/\overline{H})P(\overline{H})}.$$

Прокомментируем, следуя К. Нейлору (C. Naylor) [23], эту формулу на простом примере из области медицинской диагностики.

Пусть некоторый человек подозревается в заболевании гриппом. Следовательно, в данном случае гипотеза H состоит в том, что он болен гриппом. Можно считать, что в медицинских учреждениях на основе статистических данных известна априорная вероятность $P(H)$ того, что пациенты в данное время года и в данной местности заболевают гриппом. Пусть E означает свидетельство высокой температуры у данного конкретного пациента. Легко видеть, что формула Байеса позволяет получить $P(H / E)$ (вероятность гриппа при наличии высокой температуры). Чтобы воспользоваться формулой Байеса для этого случая, необходимо знать вероятности:

- $P(E / H)$ — вероятность высокой температуры при гриппе;
- $P(E / \overline{H})$ — вероятность высокой температуры при отсутствии гриппа.

Мы предполагаем, что обе эти вероятности нам известны. Они также получаются при обработке имеющихся статистических данных. Ясно, что все три числа $P(H)$, $P(E / H)$, $P(E / \overline{H})$ могут быть получены заранее и имеют универсальный характер, не зависящий от данных по конкретному пациенту.

Теперь, замечая, что

$$P(\overline{H}) = 1 - P(H),$$

мы можем воспользоваться формулой Байеса — все числа в правой части известны.

Пусть

- $P(H) = 0{,}001$; $P(\overline{H}) = 1 - P(H) = 0{,}999$;
- $P(E / H) = 1{,}0$;
- $P(E / \overline{H}) = 0{,}01$.

Тогда по формуле Байеса получим

$$P(H / E) \cong 0{,}009.$$

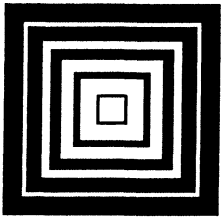
Таким образом, вероятность заболевания гриппом при поступлении свидетельства о высокой температуре увеличилась и составила 0,009 по сравнению с 0,001 (исходная априорная вероятность).

Принципиальная схема работы байесовской ЭС состоит в следующем.

Первоначально мы имеем априорную вероятность $P(H)$, которая хранится в базе знаний. Но, получив свидетельство E и пересчитав эту вероят-

ность по формуле Байеса, мы можем записать ее на место $P(H)$. Получение очередного свидетельства приводит к новому обновлению (увеличению или уменьшению) этой вероятности. Каждый раз текущее значение этой вероятности будет считаться априорным для применения формулы Байеса. В конечном итоге, собрав все сведения, касающиеся всех гипотез (например, диагнозов болезней), ЭС приходит к окончательному заключению, выделяя наиболее вероятную гипотезу в качестве результата экспертизы.

Далее мы уточним и детализируем приведенную принципиальную схему.



Глава 15

Нейлоровские диагностирующие системы

Разработанная К. Нейлором концепция построения ЭС основана на изложенной в *главе 14* общей байесовской схеме. Основные принципы, реализованные в данной ЭС, включают:

- введение верхних и нижних порогов для вероятностей гипотез;
- учет неопределенностей, заключенных в реакции пользователей;
- введение цен свидетельств, определяющих сценарий диалога с пользователем.

Рассмотрим эти вопросы более подробно.

15.1. Элементы механизма логического вывода

Первое усложнение, которое вводится в общую схему байесовского подхода, связано с использованием верхних и нижних порогов для вероятностей отдельных гипотез. Если вероятность $P(H)$ после учета всех свидетельств превосходит *верхний порог* $M1(H)$:

$$P(H) > M1(H),$$

то гипотеза H принимается как основа для возможного заключения. Если же

$$P(H) < M2(H),$$

где $M2(H)$ — *нижний порог*, то гипотеза H отвергается как неправдоподобная.

Есть основания устанавливать верхние и нижние пороги $M1$, $M2$ индивидуально для каждой гипотезы в соответствии с имеющейся в ЭС базой знаний и максимальных возможных уровней вероятностей гипотез с учетом всех принципиально возможных свидетельств. Например, можно полагать

$$M1(H) = 0{,}9PMAX(H),$$

$$M2(H) = 0{,}5M1(H),$$

где $PMAX(H)$ — максимальная возможная вероятность, достижимая для данной гипотезы, при условии, что все свидетельства, имеющиеся в базе знаний и связанные с этой гипотезой, будут подтверждены пользователем в пользу гипотезы H. Величины $PMAX(H)$ для всех H, так же как и $M1(H)$, $M2(H)$, очевидно, могут быть вычислены заранее и также включены в базу знаний ЭС.

Учет неопределенности, заключенной в ответах пользователя на вопросы ЭС, является важным моментом в организации диалога. В идеале мы могли бы предположить, что на вопрос ЭС пользователь отвечает либо "да", либо "нет" (есть высокая температура у пациента, нет высокой температуры и т. д.), т. е. выполняется данное свидетельство E или не выполняется. Более реалистичной является ситуация, когда пользователь по какой-либо причине либо хочет уклониться от ответа (если, например, вопрос слишком сложен и не соответствует квалификации пользователя в данной предметной области), либо стремится дать не слишком определенный ответ. Например, если задается вопрос о наличии повышенной температуры у пациента, то необходимо дать пользователю возможность проранжировать степень повышения температуры, например, в соответствии с 11-балльной шкалой:

- –5 соответствует НЕТ;
- 0 соответствует НЕ ЗНАЮ;
- +5 соответствует ДА.

Присутствуют и все промежуточные целые значения шкалы от –5 до +5.

В результате каждое свидетельство E будет оцениваться по этой шкале на основании ответа пользователя $R \in \overline{R}$

$$\overline{R} = \{-5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5\}.$$

В соответствии с байесовским подходом после обработки очередного свидетельства E мы вычисляли вероятность $P(H / E)$ и заменяли ею предыдущую вероятность $P(H)$. Теперь мы должны предложить способ вычисления не $P(H / E)$, а $P(H / R)$. Это может быть выполнено следующим образом. Во-первых, случай $R = 5$ должен соответствовать вероятности

$P(H / E)$, вычисленной по формуле Байеса. Во-вторых, вариант $R = -5$ должен соответствовать величине $P(H / \overline{E})$. Последняя вероятность может быть найдена из соотношений:

$$P(H) = P(H / E)P(E) + P(H / \overline{E})P(\overline{E}),$$

$$P(\overline{E}) = 1 - P(E)$$

$$P(E) = P(E / H)P(H) + P(E / \overline{H})P(\overline{H}).$$

Здесь величины $P(E / H)$, $P(H)$, $P(E / \overline{H})$, $P(E)$, $P(H / E)$ предполагаются заданными.

В-третьих, случай $R = 0$ (НЕ ЗНАЮ), очевидно, не должен изменять априорную вероятность $P(H)$ и поэтому здесь имеем:

$$P(H / R) = P(H).$$

Мы получили три характерные точки на графике $P(H / R)$ как функции R. Промежуточные значения $P(H / R)$ предлагается восстанавливать с помощью линейной интерполяции. Таким образом, для любого $R \in \overline{R}$ мы получаем соответствующее значение $P(H / R)$, что и требовалось. При этом максимальное значение $P(H / R)$ будет равно $P(H / E)$ и соответствовать случаю $R = 5$, как и указывалось ранее.

15.2. Цены свидетельств — косвенная цепочка рассуждений

Последнее усложнение, которое вносится в базовую схему байесовского подхода, связано с проблемой управления логическим выводом.

Проблема состоит в следующем. В базе знаний имеем конечное множество гипотез

$$H_1, H_2, ..., H_n$$

и конечное множество свидетельств (вопросов)

$$E_1, E_2, ..., E_S.$$

Каждой гипотезе H_i соответствует свое подмножество ассоциированных с ней свидетельств. Согласно стратегии прямой цепочки рассуждений (мы обсуждали этот подход при рассмотрении продукционных ЭС) можно, в принципе, последовательно прорабатывать весь список возможных вопросов и затем выбирать наиболее вероятную гипотезу. При этом, однако, возникают две проблемы.

Во-первых, процедура такого перебора может потребовать больших затрат реального времени, т. к. мы не знаем, на какой H_i остановится в конце концов ЭС, и поэтому вопросы, ассоциированные с H_i, могут поступить пользователю, например, в конце диалога. Во-вторых, такой способ ведения диалога психологически не соответствуют желанию пользователя установить диагноз, т. е. выбрать конкретную гипотезу H. У пользователя создается впечатление, что ЭС бессистемно "прыгает" из одной области в другую, перемежая "общие" вопросы (которые на самом деле желательны лишь в начале диалога) с очень частными.

Поэтому, как мы уже знаем, часто используется обратная цепочка рассуждений: от гипотез — к подтверждающим или опровергающим свидетельствам. В этом случае мы осуществляем перебор не в множестве свидетельств E, а в множестве гипотез H. По своему поведению такая ЭС выглядит более целенаправленной, особенно при проверке конкретного диагноза. Но и здесь в общем случае (когда диагноз не известен и не ставится задача подтверждения или опровержения конкретного диагноза) неясно, в каком именно порядке следует перебирать гипотезы H_i, а также привлекать свидетельства для подтверждения тех или иных гипотез.

Далее мы рассмотрим предложенную К. Нейлором процедуру логического вывода, которая не может быть отнесена ни к прямой, ни к обратной цепочке рассуждений. Она условно может быть названа *косвенной цепочкой рассуждений.*

Рассматриваемый подход сводится к назначению цены каждого свидетельства E_i, отражающей важность данного свидетельства в процессе логического вывода. Далее при построении диалога каждый раз выбираются вопросы (свидетельства) с наибольшими ценами. В процессе вывода цены свидетельств все время пересчитываются в зависимости от получаемых текущих результатов. По-видимому, именно так ведут себя специалисты-эксперты, задавая те вопросы, которые представляются им наиболее важными в текущей ситуации.

В качестве цены свидетельства E предлагается использовать выражение

$$C(E) = \sum_{i=1}^{n} \left| P(H_i / E) - P(H_i / \overline{E}) \right|,$$

где:

$$P(H / E) = \frac{P(E / H)P(H)}{P(E)}, \quad P(E) = P(E / H)P(H) + P(E / \overline{H})P(\overline{H})$$

соответствует формуле Байеса, а выражение для вероятности $P(H / \overline{E})$

$$P(H / \overline{E}) = \frac{(1 - P(E / H)) \cdot P(H)}{1 - P(E)}$$

получается из предыдущей формулы заменой E на $\overline{E}$ и на основании очевидных равенств

$$P(\overline{E} / H) = 1 - P(E / H),$$

$$P(\overline{E}) = 1 - P(E).$$

Таким образом, $C(E)$ определяется как полная сумма максимально возможных изменений вероятностей по всем n гипотезам, имеющимся в базе знаний.

Далее система выбирает свидетельство E с максимальной ценой $C(E)$ и задает соответствующий вопрос пользователю. После получения ответа (в 11-балльной шкале) пересчитываются все вероятности $P(H_i)$ — они заменяются на $P(H_i / R)$ в соответствии с вышеизложенным механизмом линейной интерполяции. После формирования нового массива $P(H_i)$ заново пересчитываются все цены свидетельств и процесс повторяется.

Отметим, что в процессе вывода все вероятности $P(E / H)$, $P(E / \overline{H})$ остаются неизменными — меняется только массив $P(H)$. Из формулы Байеса видно, что

$$\lim P(H / E) = 0 \text{ при } P(H) \to 0;$$

$$\lim P(H / \overline{E}) = 0 \text{ при } P(H) \to 0$$

(при неизменных $P(E / H)$, $P(E / \overline{H})$). Следовательно, цены свидетельств, относящиеся к таким маловероятным гипотезам, будут автоматически падать, что естественно: желательно задавать вопросы, относящиеся к более перспективным гипотезам. И наоборот, цены свидетельств, относящиеся к более вероятным гипотезам, будут расти.

Легко видеть, что введенная формула для $C(E)$ по существу совпадает с, может быть, более "естественной" формулой

$$\overline{C}(E) = \sum_{i=1}^{n} \left\{ \left| P(H_i / E) - P(H) \right| + \left| P(H_i / \overline{E}) - P(H) \right| \right\}.$$

Пример 15.1. Пусть $P(H) = 0{,}001$; $P(E / H) = 0{,}9$; $P(E / \overline{H}) = 0{,}01$. Тогда $P(H / E) \cong 0{,}0083$; $P(H / \overline{E}) \cong 0{,}0001$.

В результате имеем:

$$C(E) = |0,0083 - 0,0001| = 0,0082;$$

$$\overline{C}(E) = |0,0083 - 0,001| + |0,0001 - 0,001| = 0,0073 + 0,0009 = 0,0082.$$

Возможное уточнение изложенного метода ведения диалога на основе использования цен свидетельств может быть связано с механизмом "взвешивания" свидетельств. При этом на шаге $L + 1$ больший вес получают свидетельства, ассоциируемые с гипотезами, имеющими отношение к запрошенному на шаге L свидетельству E_{L-1}. Соответствующие связи представлены на рис. 15.1.

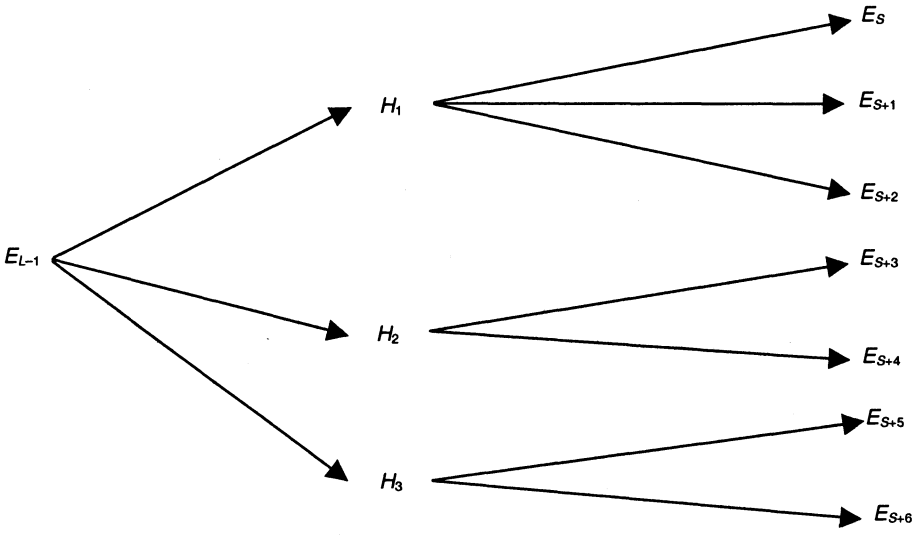


Рис. 15.1. Связь свидетельств и гипотез

Согласно рис. 15.1, на шаге L свидетельства E_S, ..., E_{S+6} получат большие цены, чем непосредственно получаемые по формуле для $C(E)$. В результате шансы на выбор в качестве E_L одного из указанных свидетельств E_S, ..., E_{S+6} возрастают. Указанный подход позволяет внести определенную "инертность" в работу ЭС, что благоприятно отражается на психологическом состоянии пользователя.

Действительно, в этом случае ЭС перестает вести себя "нервозно" и перепрыгивать с одной группы гипотез на другую даже при малых различиях

в соответствующих ценах свидетельств. Реализуется более естественный и оправданный ход диалога, что с точки зрения пользователя создает эффект более "внимательного" отношения ЭС к предмету экспертизы.

15.3. Правила остановки

Мы переходим к важному вопросу определения момента окончания работы ЭС. Ясно, что для этого, вообще говоря, достаточно перебрать все свидетельства и все гипотезы. Однако окончание процесса экспертизы обычно наступает значительно раньше.

В рассматриваемой диагностирующей ЭС с каждой гипотезой H связываются следующие пять чисел:

- $P(H)$ — текущая вероятность истинности гипотезы H;
- $P_{max}(H)$ — текущая максимальая достижимая вероятность для гипотезы H в предположении, что все оставшиеся свидетельства будут свидетельствовать в пользу H;
- $P_{min}(H)$ — текущая минимальная вероятность гипотезы H (все оставшиеся свидетельства говорят о том, что скорее $\overline{H}$, чем H);
- $M1(H)$, $M2(H)$ — верхний и нижний пороги, введены в *разд. 15.1* как величины, пропорциональные $PMAX(H)$.

Как уже указывалось, $PMAX(H)$ означает максимальную достижимую в данной ЭС вероятность истинности H (если все без исключения свидетельства свидетельствуют в пользу H). В самом начале процесса, очевидно, имеем:

$$P_{max}(H) = PMAX(H),$$

затем величина $P_{max}(H)$ будет, вообще говоря, меньше, чем $PMAX(H)$. (Последняя величина является константой для данной базы знаний.)

С учетом указанных пяти величин возможны следующие условия остановки работы ЭС.

1. *Определение наиболее вероятной гипотезы.* Если в некоторый момент обработки списка свидетельств в процессе работы ЭС оказывается, что для какой-то гипотезы H_k выполняются условия:

   $$P_{min}(H_k) > P_{max}(H_i) \text{ для } \forall i \neq k,$$

   то, очевидно, гипотеза H_k оказывается наиболее вероятной и продолжать процесс экспертизы бессмысленно. Ответ пользователю — H_k.

2. *Определение правдоподобной гипотезы.* Иначе говоря, такой гипотезы H, для которой в данный текущий момент времени выполнилось условие

$$P_{\min}(H) > M1(H),$$

где $M1(H)$ — критерий верхнего порога. Данный критерий остановки является в определенном смысле основным и реализуется чаще других. Обычно пользователя интересуют все такие гипотезы и поэтому необходимо дополнительно проверить условие

$$P_{\max}(H) < M1(H)$$

для всех остальных гипотез. Если это условие не выполнено, то процесс работы ЭС может быть продолжен для пополнения списка правдоподобных заключений.

3. *Отсутствие заключения* — констатация факта, что ЭС не может принять решение. Данная ситуация возникает, если мы имеем в какой-то момент времени

$$P_{\max}(H) < M2(H)$$

для всех H.

(Ясно, что для корректности базы знаний мы должны потребовать выполнение условия $PMAX(H) > M2(H)$ для всех H). Кроме этого, мы вынуждены констатировать отсутствие заключения, если при остановке согласно условиям 1 мы имеем для наиболее вероятной гипотезы H_k

$$P(H_k) < M1(H_k).$$

Мы прекращаем обсуждение возможных условий остановки. Ясно, что для реальных ЭС все они должны быть согласованы между собой и с базой знаний ЭС. Необходимо также при создании новых баз знаний осуществлять проверку их корректности и логической непротиворечивости, чтобы процесс экспертизы, например, не закончился в самом начале.

15.4. Структура базы знаний и алгоритм логического вывода

База знаний рассматриваемой диагностирующей ЭС содержит записи, касающиеся знаний о конкретных диагнозах (гипотезах) и знаний о соответствующих симптомах (свидетельствах). Формат записи для описания конкретной гипотезы H (формат 1) может иметь следующий вид:

$$НАЗВ.\ ГИП.;\ P;\ S;\ (j_1;\ p_1^+;\ p_1^-);\ \ldots;\ (j_S;\ p_S^+;\ p_S^-).$$

Здесь *НАЗВ. ГИП.* — название гипотезы H; $P = P(H)$ — априорная (исходная) вероятность данной гипотезы; S — число свидетельств E_i, относящихся к данной гипотезе; j_k — номер свидетельства; $p_k^+ = P(E_{j_k} / H)$ — вероятность выполнения свидетельства для данной гипотезы; $p^- = P(E_{j_k} / \overline{H})$ — вероятность выполнения свидетельства E_{j_k} при неверности данной гипотезы H.

Знания о свидетельствах могут быть представлены в следующем формате (формат 2):

№_свидетельства; Название_свидетельства; Задаваемый_вопрос.

Замечание 15.1

Одни и те же номера свидетельств j_k могут, очевидно, присутствовать в описаниях различных гипотез, но вот соответствующие вероятности p_k^+, p_k^- при этом будут, скорее всего, разными.

Пример 15.2. Рассмотрим фрагмент медицинской базы знаний.

- Описание гипотезы (формат 1):

  Грипп; 0,001; 2; (1; 1; 0,01); (2; 0,9; 0,1)

  (Круглые скобки могут отсутствовать — здесь они оставлены для наглядности записи.)

- Описание свидетельств (формат 2):

  1; Температура; Есть ли у вас высокая температура?

  2; Насморк; Есть ли у вас насморк?

Дадим расшифровку приведенных записей. Априорная вероятность того, что некто болен гриппом, равна 0,001. В данной базе знаний с гриппом связаны два (2) симптома. Первый симптом — высокая температура. Вероятность высокой температуре здесь положена равной 1. Вероятность высокой температуры при отсутствии гриппа равна 0,01.

Второй симптом — насморк. Вероятность насморка при гриппе равна 0,9. Вероятность насморка при отсутствии гриппа равна 0,1.

На симптом № 1 (температура) могут быть ссылки и из записей для других гипотез (болезней), но вероятности p^+, p^- в тройках $(1; p^+; p^-)$ будут уже другими.

Подытожим наше рассмотрение нейлоровских байесовских ЭС в виде следующего простого алгоритма.

15.4.1. Алгоритм логического вывода

- Шаг 1. Сформировать массив $P(H_i)$ априорных вероятностей для всех гипотез. Для этого просмотреть базу знаний (формат 1) и извлечь P для каждой из гипотез (это второй элемент записи в формате 1).
- Шаг 2. Сформировать массив цен свидетельств $C(E_j)$:

$$C(E_j) = \sum_{i=1}^{n} \left| \frac{P(E_j / H_i)P(H_i)}{P(E_j / H_i)P(H_i) + P(E_j / \overline{H}_i)P(\overline{H}_i)} - \right.$$

$$\left. - \frac{(1 - P(E_j / H_i))P(H_i)}{1 - P(E_j / H_i)P(H_i) - P(E_j / \overline{H}_i)P(\overline{H}_i)} \right|,$$

  где j — номер свидетельства, определяет выбор троек $(j; p^+; p^-)$ из базы знаний (формат 1). При этом

$$p^+ = P(E_j / H_i); \quad p^- = P(E_j / \overline{H}_i).$$

  Очевидно, в сумму для $C(E_j)$ входят только те слагаемые, которые соответствуют гипотезам H_i, содержащим в своем описании (формат 1) тройки с номером j.

- Шаг 3. Определить свидетельство E_m с максимальной ценой:

$$E_m : m = \arg\max_j C(E_j).$$

  (Здесь же может быть реализована процедура определения E_m с учетом результата дополнительного взвешивания цен свидетельств, как это было указано в *разд. 15.2*).

- Шаг 4. Задать пользователю вопрос, хранящийся в базе знаний (формат 2) для найденного на шаге 3 свидетельства E_m. Ответ пользователя R_m должен быть дан по шкале от –5 до +5.
- Шаг 5. Пересчитать массив $P(H_i)$ в соответствии с полученным на шаге 4 ответом пользователя *(см. разд. 15.1)*:

$$P(H_i) := P(H_i / R_m).$$

- Шаг 6. С учетом полученных на шаге 5 новых значений для элементов массива $P(H_i)$ пересчитать элементы массива цен свидетельств $C(E_j)$.
- Шаг 7. Вычислить для каждой гипотезы H_i значения $P_{max}(H_i)$, $P_{min}(H_i)$ *(см. разд. 15.3)*.

- Шаг 8. Определить число PM:

$$PM = \max_i P_{\min}(H_i).$$

Это наибольший из возможных достижимых минимумов вероятностей для всех гипотез.

- Шаг 9. Проверить, существует ли такой номер k, для которого

$$P_{\max}(H_k) > PM.$$

Если ДА, то перейти к шагу 3. Если НЕТ, то выбрать гипотезу (гипотезы) H_m:

$$m = \arg\max_i P(H_i)$$

(наиболее вероятный результат).

- Шаг 10. Выдать в качестве результата гипотезу (гипотезы) H_m и вызвать подпрограмму объяснений, которая представляет пользователю протокол с описанием всех выводов, проделанных ЭС. Протокол снабжается необходимыми комментариями.

Как уже указывалось ранее, могут быть реализованы и более сложные критерии окончания работы ЭС.

15.5. Пример базы знаний

Рассмотрим теперь ставший уже классическим пример базы знаний для ЭС, используемой для ремонта автомобилей (К. Нейлор). Пример носит иллюстративный характер и не претендует на использование в реальных задачах.

- Гипотезы H_i:

Сел аккумулятор; 0,1; 5; (1; 0; 0,99); (2; 0,7; 0,05); (4; 0,2; 0,5); (5; 0; 0,99); (6; 1; 0,01)

Нет бензина; 0,05; 2; (2; 1; 0,01); (6; 0,9; 0,02)

Отсырел распределитель зажигания; 0,01; 3; (3; 0,9; 0,1); (4; 0,25; 0,5); (6; 0,9; 0,02)

Загрязнены свечи; 0,01; 2; (4; 0,01; 0,5); (6; 0,9; 0,02)

- Свидетельства (симптомы) E_i:

1; Фары горят; Горят ли фары?

2; Указатель бензина на нуле; Есть ли бензин?

3; Автомобиль отсырел; Не находился ли автомобиль долго под дождем?

4; Автомобиль недавно прошел техобслуживание; Проходил ли автомобиль недавно техобслуживание?

5; Стартер крутится; Крутится ли стартер?

6; Автомобиль не заводится; Автомобиль не заводится?

По данной базе знаний можно сформировать массив $C_1(E_i)$ исходных цен свидетельств (табл. 15.1).

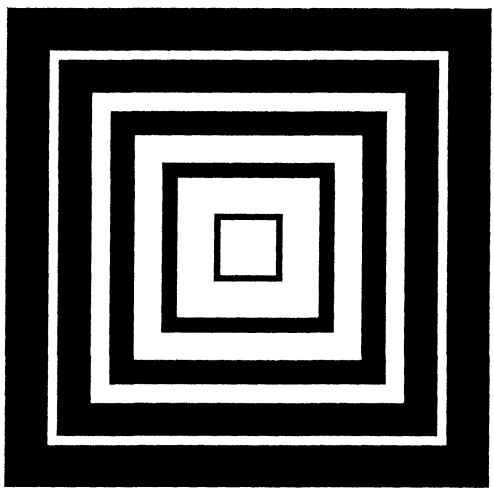
Таблица 15.1. *Цены свидетельств*

i	**Свидетельство**	$C_1(E_i)$	$C_2(E_i)$
1	Фары горят	0,9174	0,9991
2	Указатель бензина на нуле	1,4151	1,2135
3	Автомобиль отсырел	0,0822	0,7554
4	Автомобиль недавно прошел техобслуживание	0,1376	0,8153
5	Стартер крутится	0,9174	0,9991
6	Автомобиль не заводится	2,2381	0,0000

Согласно вычисленным исходным ценам свидетельств $C_1(E_i)$ первым будет всегда задаваться вопрос, связанный со свидетельством E_6 (у него максимальная цена 2,2381):

Автомобиль не заводится?

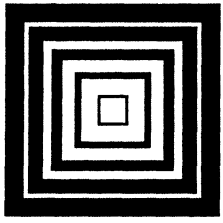
При ответе пользователя ДА ($R = 5$) мы можем пересчитать массив $P(H_i)$ и вычислить новые цены свидетельств $C_2(E_i)$ (они также приведены в табл. 15.1). С учетом новых цен свидетельств будет задан следующий вопрос, связанный со свидетельством E_2 (у него максимальная цена 1,2135). При этом сообщение о том, что автомашина все-таки не заводится, привело к существенному увеличению цен и остальных свидетельств, что выглядит весьма естественно.



ЧАСТЬ IV

Примеры систем поддержки принятия решений

Глава 16



Quick Choice — система многокритериального выбора вариантов

16.1. Область применения системы

Система Quick Choice (QCH) предназначена для решения многокритериальных задач выбора вариантов из заданного конечного множества

$$X = (x_1, ..., x_n).$$

Предполагается, что каждый из вариантов x_i оценивается по m частным критериям F_k. Частные критерии могут иметь как числовые, так и порядковые шкалы. Примером числового критерия в задаче выбора автомобиля из заданного множества (например, при покупке подержанного автомобиля) может служить пробег автомобиля, выраженный в километрах — чем он меньше, тем лучше. Цвет автомобиля естественно оценивать в порядковой шкале, например, вида: черный—белый—коричневый— ... —желтый. Здесь предполагается, что степень предпочтительности убывает слева направо.

В данной системе приняты следующие предположения об исходной задаче:

- множество исходных вариантов конечно, и ЛПР (лицо, принимающее решение, пользователь) может перечислить элементы этого множества;
- ЛПР может определить цель по каждому критерию, а также задать некоторые параметры критериев, описанные ниже;
- ЛПР может ранжировать критерии по важности и указывать равноценные критерии.

Кроме того, предполагается, что ЛПР может принимать или отвергать предлагаемые ему альтернативы, а также указывать не устраивающие его по некоторым критериям оценки отвергаемой альтернативы.

Система QCH поддерживает непрерывный диалог с пользователем, в процессе которого у ЛПР запрашивается вся необходимая информация о задаче и создается отчет о ней. Для запуска этого диалога нужно в меню **Запуск** выбрать пункт **Диалог**.

16.2. Исходные данные

Для работы с системой необходимо задать следующие исходные данные об альтернативах и критериях:

- количество критериев — число m;
- тип каждого критерия (порядковый или числовой);
- цель по каждому критерию (максимум или минимум);
- все принимаемые значения для перечислимых критериев;
- набор ординальной (порядковой) информации вида "критерий P важнее критерия Q";
- набор ординальной информации вида "критерий P равноценен критерию Q";
- количество альтернатив (вариантов), среди которых производится выбор — число n;
- весь набор альтернатив;
- значения всех частных критериев для каждой альтернативы;
- минимальные требования к выбираемой альтернативе по каждому из частных критериев.

16.3. Типы критериев

Для каждого критерия необходимо задать некоторые свойства. Эти свойства описываются далее. Свойство "цель" описывает намерения ЛПР по данному критерию: минимум или максимум. Если целью по данному критерию является максимум, то система предпочитает альтернативы с большим значением данного критерия, если же минимум, то наоборот.

Структура задания типа критерия изображена на рис. 16.1. Для непрерывного критерия ЛПР должен задать максимальное и минимальное значения. Непрерывный критерий принимает значения на заданном промежутке. Перечислимый критерий принимает значения из фиксиро-

ванного набора. Значения перечислимых критериев могут задаваться численно, например:

- отлично — 5;
- хорошо — 4;
- удовлетворительно — 3;
- плохо — 2.

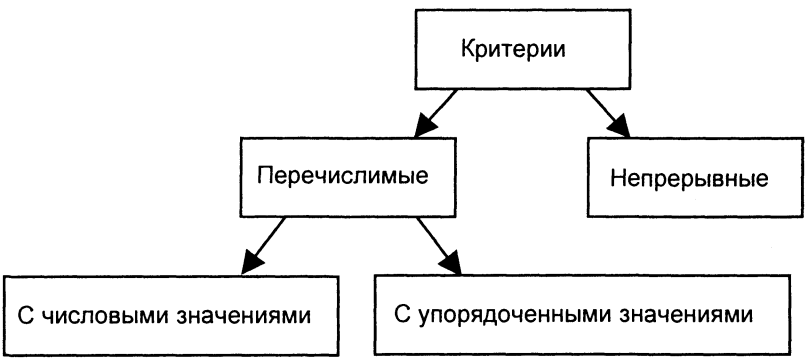


Рис. 16.1. Типы критериев

В этом случае пользователь задает текстовые значения и в системе они заменяются на численные.

Система не требует от ЛПР задания численных значений, если задать тип критерия как "Перечислимый с упорядоченными значениями". В этом случае имеется в виду порядковая шкала, и достаточно задать лишь набор значений данного критерия, а также упорядочить их в порядке предпочтения.

16.4. Функции, реализованные в системе

Описываемая система позволяет выполнять следующие функции:

- задание информации о задаче в диалоге с пользователем;
- построение множества оптимальных вариантов в соответствии с методом t-упорядочения;
- реализация метода ограничений, позволяющего пользователю выбрать одну альтернативу из t-оптимального множества;

- импорт из базы данных;
- импорт данных из текстового файла;
- проверка корректности ординальной информации;
- создание отчета.

16.5. Инсталляция системы

16.5.1. Требования к аппаратуре и окружению

Для работы системы QCH необходима следующая минимальная конфигурация:

- процессор — 486;
- ОЗУ — 8 Мбайт;
- операционная система — Windows 2000/XP.

После инсталляции система занимает на диске 8—10 Мбайт в зависимости от количества устанавливаемых компонентов.

16.5.2. Установка системы

Для запуска процесса инсталляции необходимо запустить файл SETUP.EXE и следовать указаниям системы. Система запросит имя папки, в которую будут скопированы файлы, необходимые для ее работы. Также будет создана папка в меню **Программы**. Для запуска системы необходимо выбрать и запустить программу QCHOICE.EXE. В состав системы также входят компоненты:

- QCHOICE.HLP — файл справки системы;
- подпапка \DEMOS — примеры задач для данной системы;
- AUTO.QCT — пример решения задачи выбора автомобиля из заданного множества вариантов;
- FLAT.TXT — пример создания текстового файла для последующего импорта в систему;
- FLAT.DB — пример таблицы БД Paradox для демонстрации импорта данных в систему из стандартной БД;
- FLAT.QCT — пример решения задачи о выборе квартиры из множества вариантов;
- подпапка \DOCUMENT — "Руководство пользователя" и краткая инструкция для пользователя системы.

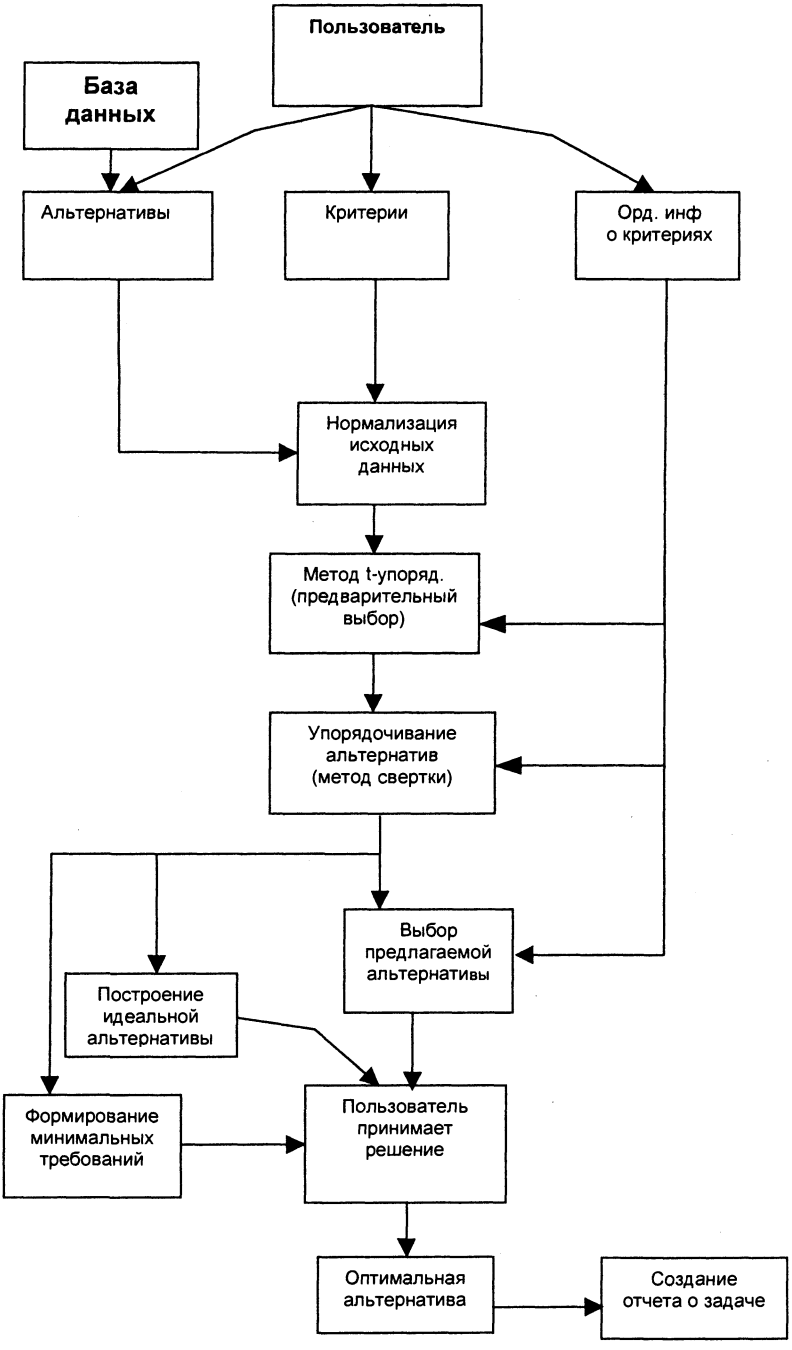


Рис. 16.2. Общая структура системы

На рис. 16.2 показана общая структура данной системы. Исходная информация о задаче задается в диалоге с пользователем. После завершения диалога система имеет информацию об альтернативах, критериях, и ординальную информацию о критериях. В системе также предусмотрено получение данных об альтернативах из текстового файла или баз данных dBase и Paradox. После этого производится нормализация исходных данных и запуск метода *t*-упорядочения.

Выбор результирующей альтернативы производится на основе описанного в этой книге и достаточно хорошо известного метода ограничений. Для более эффективного использования метода ограничений производится предварительное ранжирование альтернатив, по результатам которого определяется порядок предложения альтернатив на рассмотрение пользователя. В качестве метода ранжирования альтернатив применяется метод аддитивной свертки. При этом учитывается ординальная информация о критериях, полученная от пользователя.

При работе метода ограничений помимо текущего варианта пользователю предлагаются "идеальная" альтернатива (с наилучшими возможными оценками по всем критериям) и установленные ограничения. Эти данные предоставляются пользователю как в численном виде, так и в виде диаграммы. При таком представлении пользователь может наилучшим образом оценить достоинства и недостатки текущей оцениваемой альтернативы.

После получения решения система создает отчет о задаче.

16.6. Запуск системы

При запуске система предложит вам выбрать один из следующих вариантов для начала работы:

- *создать новую задачу в диалоге с системой* — вызывается диалог, в результате которого система получит от пользователя необходимые данные об интересующей его задаче и поможет произвести правильный выбор;
- *создать новую задачу и задать параметры вручную* — система выходит из диалогового режима и предлагает пользователю задавать данные о задаче и принимать решение на основе главного меню и диалоговых окон;
- *загрузить ранее созданную задачу* — система предложит пользователю загрузить задачу из файла;
- *загрузить ранее созданную задачу и задать собственные предпочтения* — система предложит пользователю загрузить задачу из файла, а затем

будет запущен диалог, в результате которого пользователь сможет изменить заданные параметры задачи и, задав свои предпочтения, принять решение о выборе какой-либо альтернативы;

- *получить данные из базы данных или текстового файла* — этот пункт следует выбрать, если исходные данные располагаются в текстовом файле или в базе данных. После выбора этого пункта будет реализован соответствующий диалог.

16.7. Получение данных

Система QCH позволяет получать исходные данные об альтернативах из стандартной базы данных Paradox или dBase, а также из текстового файла. Для этого в системе реализованы соответствующие диалоги, которые описаны далее.

16.7.1. Получение данных из текстового файла

Если задано получение (импорт) данных из текстового файла, необходимо задать требуемый файл. Данные об альтернативах в текстовом файле должны располагаться в виде таблицы. Строка таблицы должна соответствовать одной альтернативе, а столбец содержит оценки альтернатив по соответствующим критериям.

Описание каждой альтернативы должно начинаться с новой строки, а значения различных критериев должны разделяться пробелами. Вот пример содержимого такого файла:

```
3  2  text1  5
4  1  text2  5
7  2  text3  9
1  1  text2  2
3  21 text2  3
```

В первой строке такого файла могут быть расположены названия критериев, а в первом столбце — названия альтернатив. Однако это не обязательно, при отсутствии названий система назовет альтернативы и критерии стандартным образом.

После выбора исходного файла система попросит задать количество столбцов в исходном файле и после этого откроется диалоговое окно, показанное на рис. 16.3.

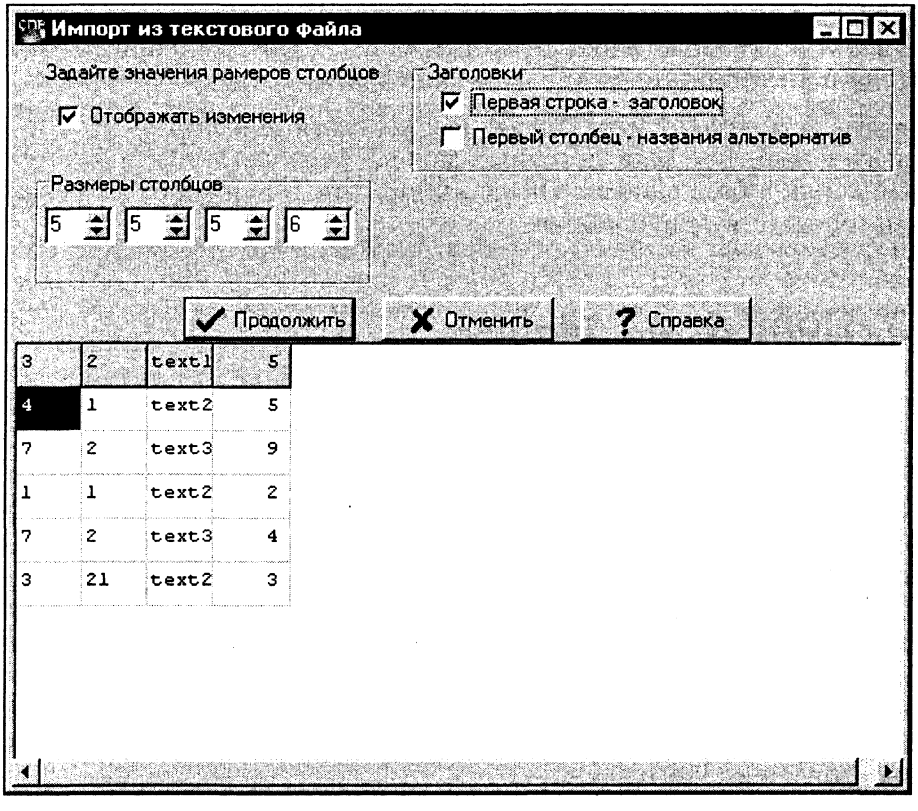


Рис. 16.3. Импорт из текстового файла

Исходный файл будет представлен на экране в виде таблицы. От пользователя требуется задать размеры столбцов таким образом, чтобы все значения, относящиеся к различным критериям, были расположены в различных столбцах этой таблицы, а относящиеся к одному критерию — в одном столбце. Изменение размера столбцов будет непосредственно отображаться на экране.

Если первый столбец файла содержит названия альтернатив, то необходимо установить флажок **Первый столбец - названия альтернатив**. Если первая строка файла содержит названия критериев, то нужно установить флажок **Первая строка - заголовок**. При этом система автоматически определит, какие критерии считать перечислимыми, а какие — непрерывными. После нажатия кнопки **Продолжить** система спросит пользователя о его предпочтениях, предложит упорядочить полученные значения перечислимых критериев и запустит диалог принятия решения.

16.7.2. Получение данных из базы данных

Если задано получение (импорт) данных из базы данных (БД), необходимо задать файл нужной таблицы базы данных.

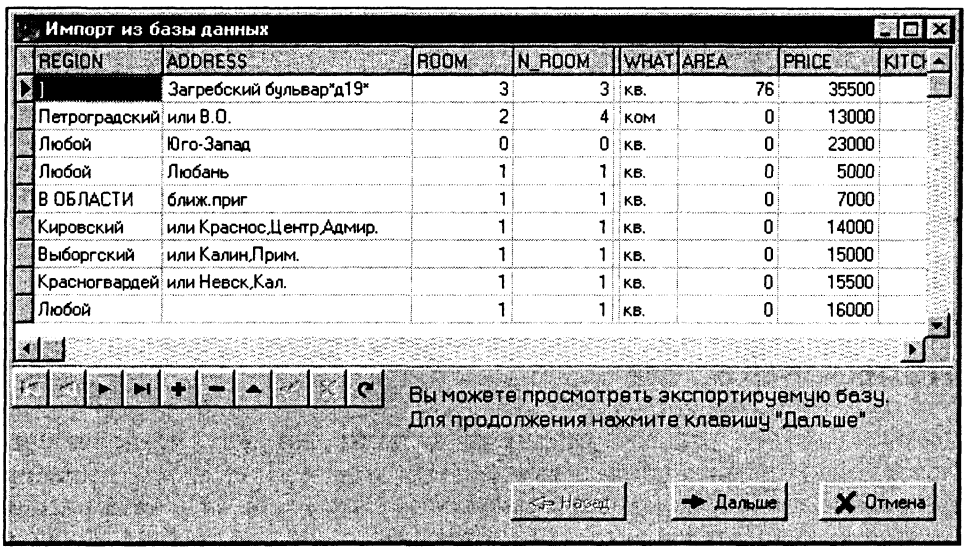


Рис. 16.4. Импорт из БД

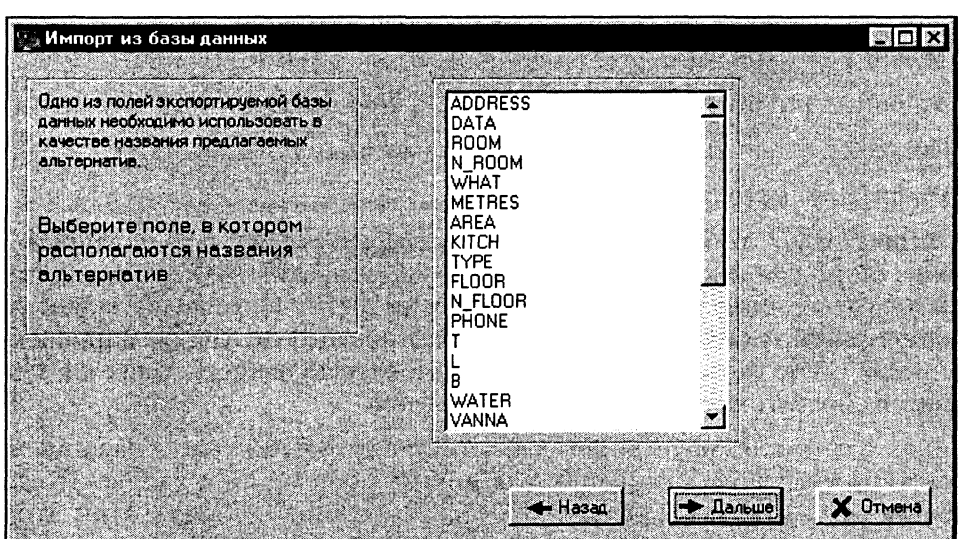


Рис. 16.5. Выбор поля

После этого данная таблица будет показана на экране. Затем надо нажать кнопку **Дальше**. Система предложит пользователю список всех полей данной базы (рис. 16.4). В этом списке нужно выбрать поле (рис. 16.5), в соответствии с которым будут именоваться альтернативы. После этого нужно выбрать поля, которые будут использоваться в качестве критериев в данной системе. Для этого необходимо перенести нужные поля из правого списка в левый. При этом система автоматически определит, какие критерии считать перечислимыми, а какие — непрерывными. После нажатия кнопки **Дальше** система спросит пользователя о его предпочтениях и запустит диалог принятия решения.

После этого будет запущен такой же диалог, как при выборе создания новой задачи в диалоге с пользователем.

16.8. Принятие решений в диалоге с пользователем

В данной системе реализован диалог с пользователем, позволяющий пользователю описать интересующую его задачу и получить решение, отвечая на вопросы, задаваемые системой. Такой подход может расширить круг пользователей системы, потому что от них не требуется специальных знаний в области принятия решений. Процедура принятия решения состоит из пяти этапов:

1. Задание критериев.
2. Задание альтернатив.
3. Задание дополнительной информации о критериях.
4. Предварительный выбор альтернатив (метод t-упорядочения).
5. Реализация метода ограничений.

На первых двух этапах пользователь непосредственно задает данные о задаче, описывая множества критериев и альтернатив. Далее в диалоге с пользователем определяются предпочтения ЛПР. Исходя из заданных предпочтений, система отбирает наиболее предпочтительные варианты. На пятом этапе система предлагает пользователю выбрать одну альтернативу в соответствии с методом ограничений.

Диалог состоит из последовательности диалоговых окон, в каждом из которых пользователь задает ту или иную информацию о задаче. После задания этой информации следует нажать кнопку **Дальше**. При этом открывается очередное диалоговое окно. Система позволяет пользователю возвращаться к предыдущим этапам, чтобы изменить введенную

ранее информацию о задаче. Диалог завершается запуском метода ограничений.

16.8.1. Задание критериев в диалоге с пользователем

В системе QCH критерии можно задавать двумя способами: либо в диалоге, либо при помощи команд главного меню. Рассмотрим задание критериев в диалоге.

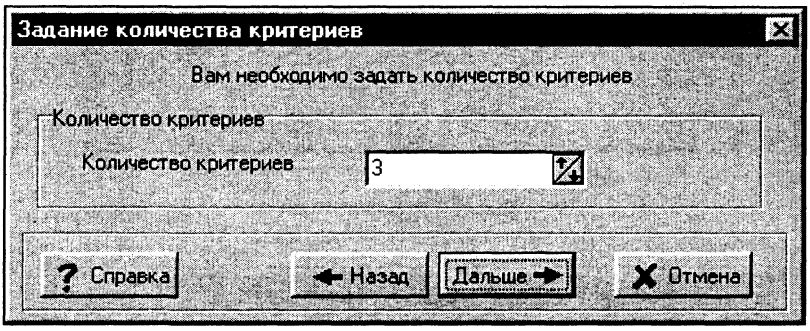


Рис. 16.6. Задание количества критериев

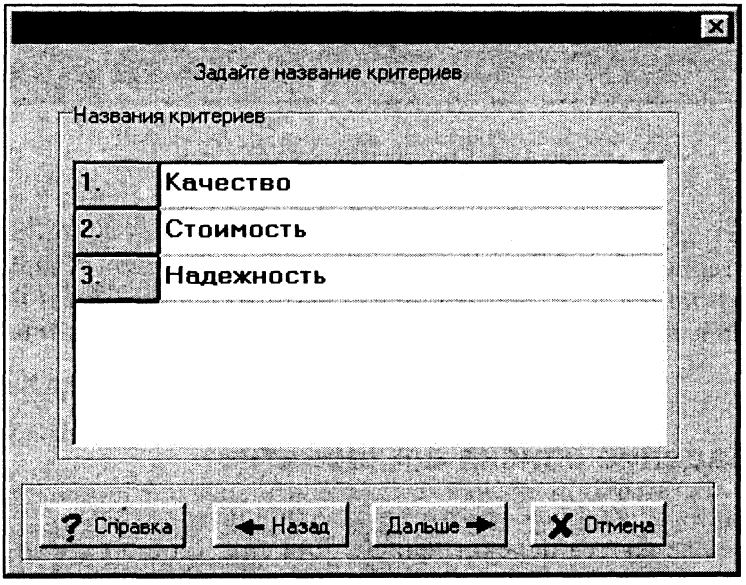


Рис. 16.7. Задание названий критериев

Вначале система запрашивает количество критериев (рис. 16.6) и их названия (рис. 16.7), после этого для каждого критерия система предлагает пользователю определить (рис. 16.8 и 16.9):

- какова его цель по данному критерию;
- может ли он перечислить все значения данного критерия или хочет задать максимальное и минимальное значения;
- может ли он численно оценить каждое принимаемое критерием значение.

Рис. 16.8. Определение цели

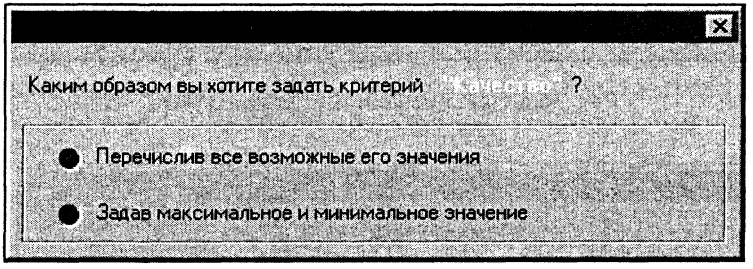


Рис. 16.9. Способ задания критерия

В процессе получения ответов на эти вопросы система формирует параметры критериев. В заключение на экран выводится окно **Свойства критериев**, где пользователь должен либо задать максимальное и минимальное значения, либо перечислить значения, принимаемые данным критерием.

16.8.2. Задание списка альтернатив

После задания критериев требуется задать альтернативы. Альтернативы в системе задаются перечислением. Для каждой альтернативы необходимо задать все значения по каждому из заданных критериев. Для непрерывных критериев задается соответствующее число, а для пере-

числимых критериев система предложит выбрать значение из множества возможных.

Сначала система запрашивает названия альтернатив и их количество, затем для каждой альтернативы предлагает задать ее значения в диалоге (рис. 16.10).

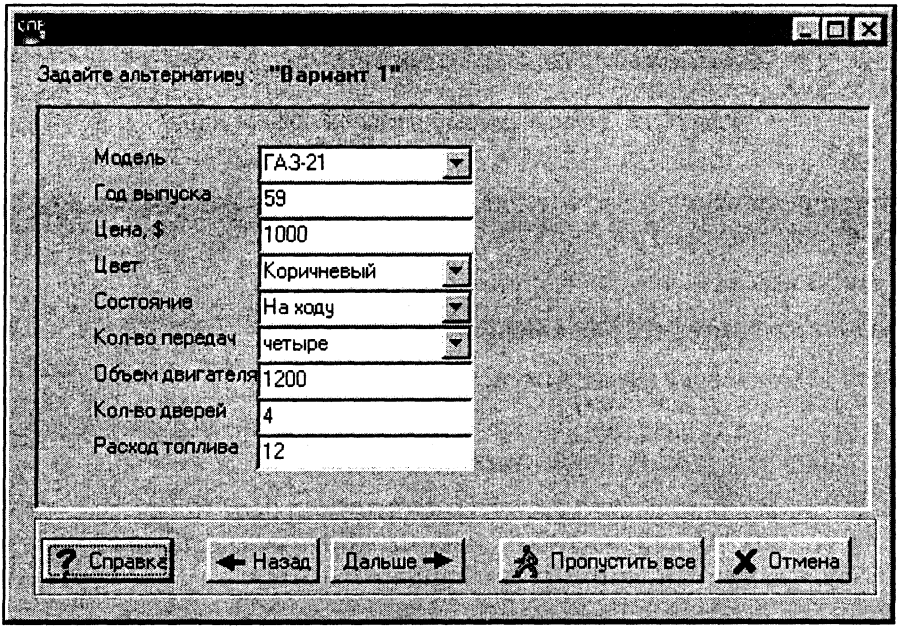


Рис. 16.10. Задание альтернативы в диалоге

При помощи кнопки **Пропустить все** можно отказаться от последовательного задания альтернатив и задать их, например, непосредственно в окне альтернативы после окончания диалога.

16.8.3. Задание дополнительной информации о критериях

Дополнительная информация поможет значительно сократить исходное множество вариантов. Особенно эффективно задание дополнительной информации для непрерывных критериев.

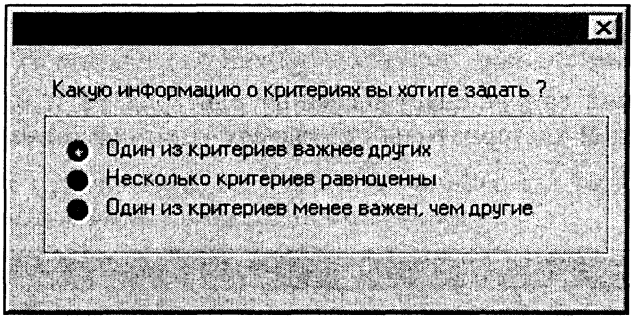


Рис. 16.11. Задание дополнительной информации

Система предложит пользователю выбрать один из трех типов дополнительной информации (рис. 16.11):

- **Один из критериев важнее других**
- **Несколько критериев равноценны**
- **Один из критериев менее важен, чем другие**

Для первого и третьего вариантов будет предложено выбрать критерий, а для второго — указать равноценные критерии.

В случае большей важности одного из критериев необходимо будет ответить на вопрос "По сравнению с какими критериями данный критерий будет более важен?" Возможны три варианта ответа:

- важнее всех критериев;
- важнее какого-нибудь одного;
- важнее нескольких критериев.

Эту информацию система запросит у пользователя.

16.9. Метод ограничений

Для выбора одной альтернативы из выбранного множества альтернатив в данной системе используется метод ограничений *(см. главу 5)*. Суть этого метода заключается в том, что ЛПР предоставляется один из вариантов, который он может принять или отвергнуть. В случае если ЛПР не устраивает данный вариант, ему предоставляется возможность задать минимальные требования по какому-нибудь критерию с целью предложить ему более приемлемую альтернативу. Если же в исходном множест-

ве не имеется альтернатив, удовлетворяющих требованиям ЛПР, то система попросит ослабить требования по какому-нибудь критерию. Внешний вид окна системы при запуске метода ограничений показан на рис. 16.12. В таблице, помимо значений предлагаемой альтернативы, представлены минимальные требования по каждому из критериев и идеальная альтернатива. Этой альтернативы не существует среди заданного набора. Идеальная альтернатива содержит наилучшие значения по каждому из критериев в отдельности. С ее помощью пользователь может видеть: какое наилучшее значение он, в принципе, может получить по каждому критерию. При этом значения по другим критериям, скорее всего, будут хуже.

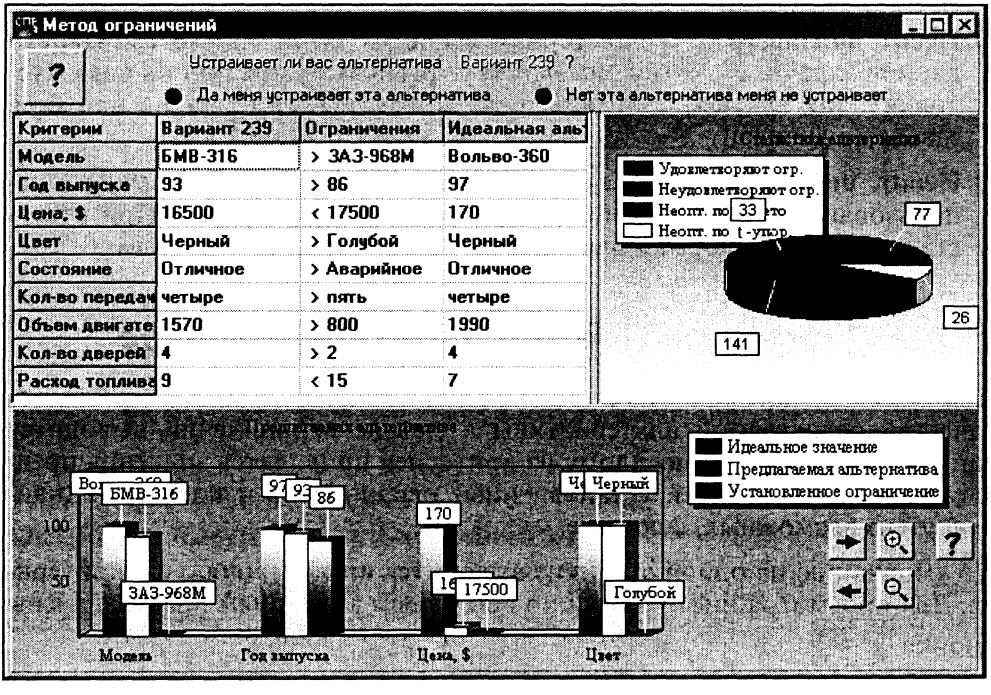


Рис. 16.12. Метод ограничений

Эта процедура будет продолжаться до тех пор, пока ЛПР не согласится с предлагаемой альтернативой. После этого будет запущен процесс создания отчета о задаче.

16.9.1. Диаграмма *Статистика альтернатив*

На данной диаграмме можно увидеть распределение исходного множества альтернатив по следующим группам:

- **Удовлетворяют огр.** — это альтернативы, удовлетворяющие установленным минимальным требованиям и прошедшие предварительный отбор. Пользователь в методе ограничений осуществляет свой выбор именно из этого множества альтернатив. При значительном количестве таких альтернатив, не следует соглашаться с предлагаемой альтернативой, т. к. имеется возможность усилить свои требования по некоторым критериям;
- **Не удовлетворяют огр.** — это альтернативы, не удовлетворяющие ограничениям, но прошедшие предварительный отбор;
- **Неопт. по Парето** — эти альтернативы не входят во множество Парето данной задачи. Поэтому они не могут быть выбраны ни при каких условиях, так как всегда в исходном множестве альтернатив найдется заведомо лучшая;
- **Неопт. по *t*-упор.** — это альтернативы, которые были отсеяны с использованием дополнительной информации о критериях. При отсутствии данной информации это множество всегда будет пустым.

16.9.2. Диаграмма *Предлагаемая альтернатива*

Для большей наглядности и оценки положительных и отрицательных качеств предлагаемой альтернативы в методе ограничений выводится данная диаграмма. По каждому из критериев приводятся значения предлагаемой альтернативы, установленные ограничения и идеальное (наилучшее из возможных) значение.

По умолчанию на одной странице выводится информация о 4-х критериях. Для просмотра информации по остальным критериям предназначены кнопки  .

Для изменения числа критериев, одновременно выводимых на экран, предназначены кнопки  .

16.9.3. Задание параметров метода ограничений

Метод ограничений является основным методом в данной системе, но его эффективность зависит от вспомогательных методов, осуществляющих предварительный отбор альтернатив и выбор предлагаемого вари-

анта в методе ограничений. Параметры этих методов можно задать, выбрав пункт **Опции \ Настройка метода ограничений** главного меню. При этом пользователю будет предложен диалог (рис. 16.13).

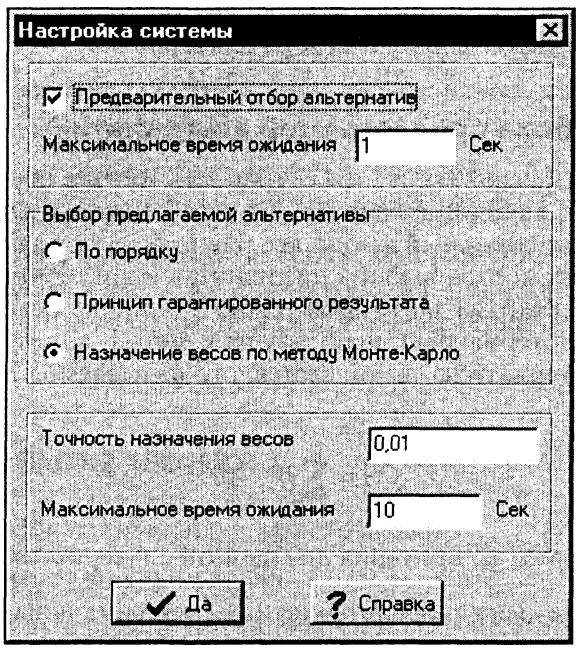


Рис. 16.13. Настройка метода ограничений

Настройка предварительного выбора

Опция **Предварительный отбор альтернатив** определяет, следует ли системе осуществлять предварительный выбор. Если данная опция будет отключена, то система будет отбирать альтернативы только по принципу Парето, не используя ординальную информацию о критериях. Данную опцию разумно отключать при большом (более 10) количестве критериев, когда предварительный отбор занимает достаточно много времени.

В поле **Максимальное время ожидания** задается максимальное допустимое время прохождения предварительного отбора (по умолчанию 10 секунд), по прохождении этого времени пользователю будет предложено

либо отказаться от предварительного отбора, либо изменить эти параметры, либо изменить ординальную информацию о критериях.

При большом количестве критериев и большом количестве информации о них предварительный выбор альтернатив может занимать достаточно много времени. Поэтому имеет смысл ограничить максимальное время прохождения предварительного отбора альтернатив.

Настройка выбора предлагаемой альтернативы

Как уже было отмечено выше, эффективность метода ограничений зависит от разумного выбора предлагаемой альтернативы. Однако в методе ограничений окончательный выбор все равно остается за пользователем, поэтому важность метода выбора предлагаемой альтернативы из множества возможных отходит на второй план. В данной системе для выбора предлагаемой альтернативы используется метод аддитивной свертки. Для каждой альтернативы вычисляется так называемое *обобщенное значение (*ОЗ*)*:

$$\text{ОЗ} = \sum_{i=1}^{N} W_i \cdot Alt_i,$$

где Alt_i — значение i-го критерия для данной альтернативы.

Пользователю предлагается альтернатива с максимальным значением ОЗ из числа альтернатив, удовлетворяющих ограничениям.

Веса критериев (W_i) вычисляются с использованием одного из следующих методов назначения весов:

- метод Монте-Карло;
- принцип гарантированного результата.

16.10. Главное окно

После запуска системы на экране появляется главное окно программы, которе содержит главное меню и рабочие окна.

Данная система является многооконной и содержит набор следующих рабочих окон, предназначенных для отображения и задания определенной информации:

- **Задание альтернатив** — альтернативы и их характеристики по заданным критериям;

- **Критерии** — критерии и их свойства;
- **Ординальная информация о критериях** — ординальная информация о критериях;
- **Нормализованные исходные данные** — нормализованные исходные данные;
- **Результаты выбора** — множество оптимальных альтернатив после предварительного отбора;
- **Информация** — статистика; также в этом окне пользователь может ввести краткий комментарий к решаемой задаче.

Каждое из рабочих окон можно свернуть или развернуть на все окно системы. Также имеется возможность изменения шрифтов каждого из окон системы. Расположение окон и установленные шрифты хранятся в файлах с расширением имени DS.

16.11. Главное меню

Рассмотрим команды (пункты) главного меню.

- **Файл** — работа с файлами задач:
  - **Новый** — создание новой задачи; при выборе этого пункта создается новая задача. Параметры новой задачи можно задать при помощи пункта **Новая задача** меню **Опции**;
  - **Открыть** — открытие существующей задачи; при выборе этого пункта программа реализует стандартный диалог, в котором запрашивается имя загружаемой задачи;
  - **Сохранить** — сохранение задачи; данная команда позволяет сохранить задачу с установленным именем, показанным в верхней части главного окна. В случае, если такой файл существует, программа запросит подтверждение на перезапись данного файла;
  - **Сохранить как** — сохранение задачи с новым именем; данная команда позволяет сохранить задачу, задав при этом новое имя файла;
  - **Импорт данных** — запуск импорта из базы данных или текстового файла;
  - **Выход** — выход из программы; при выходе программа предлагает сохранить имеющуюся задачу;

- **Запуск** — позволяет запускать различные методы.
  - **Предварительный выбор** — запуск метода предварительного выбора, при этом осуществляется запуск алгоритма принятия решения и осуществляется построение множества оптимальных вариантов. После правильного завершения работы алгоритма в окне **Информация** отображается статистика работы алгоритма, а окно **Результаты** становится активным и содержит выбранные варианты и их значения по каждому из критериев и ОП;
  - **Метод ограничений** — запуск метода ограничений;
  - **Диалог** — запуск диалога;
- **Критерии** — позволяет задавать информацию о критериях, отображаемую в окне **Критерии**:
  - **Добавить** — добавление нового критерия;
  - **Удалить** — удаление критерия;
  - **Количество** — задание количества критериев;
  - **Свойства** — задание свойств критерия;
- **Альтернативы** — данные команды позволяют модифицировать набор альтернатив:
  - **Добавить** — добавление альтернативы; эта команда позволяет добавлять новые альтернативы. После появления новой альтернативы необходимо задать ее имя и значения по каждому из критериев;
  - **Удалить** — удаление альтернативы;
  - **Количество** — задание количества альтернатив;
  - **Автоформат** — включение/выключение автоматического задания ширины столбцов в окне **Задание альтернатив;**
- **Доп. информация** — при помощи этих команд осуществляется задание ординальной информации о критериях:
  - **Добавить** — добавление новой строки ординальной информации о критериях;
  - **Удалить** — удаление строки ординальной информации;
  - **Очистить** — удаление всей дополнительной информации;

- **Опции** — задание параметров системы:
  - **Настройка метода ограничений** — определение параметров метода предварительного отбора альтернатив и метода выбора предлагаемой альтернативы;
  - **Шрифты** — позволяет изменять тип и размер надписей во всех окнах.
  - **Новая задача** — задание параметров новой задачи;
  - **Сохранить** — сохранение расположения окон и установленных шрифтов в файле DEFAULT.DS; эти параметры будут загружаться автоматически при запуске;
  - **Сохранить в файл** — сохранение расположения окон и установленных шрифтов в файле, имя которого задается при выборе этой команды;
  - **Загрузить стандартные** — загрузка расположения окон и установленных шрифтов из файла DEFAULT.DS;
  - **Загрузить из файла** — загрузка расположения окон и установленных шрифтов из произвольного файла;
  - **Панель управления** — показать/убрать панель управления с экрана;
  - **Запускать диалог при старте** — если этот пункт включен, то при старте система будет автоматически загружать диалог с пользователем;
  - **Изменить панель управления** — вызов диалога, в котором задаются элементы панели управления;
- **Дополнительно** — некоторые дополнительные функции;
  - **Случайно заполнить таблицу** — этот пункт позволяет заполнить оценки альтернатив случайными значениями критериев. Эта функция используется для тестирования метода, а также может применяться для его более подробного изучения;
- **Окно** — задание расположения окон:
  - **Рядом** — все несвернутые окна располагаются так, что они не накладываются друг на друга;
  - **Каскадом** — все несвернутые окна располагаются так, что они накладываются друг на друга;
  - **Упорядочить все** — упорядочиваются все свернутые окна;
  - **Свернуть все** — все окна сворачиваются;

- **Альтернативы** — окно **Задание альтернатив** становится активным;
- **Критерии** — окно **Критерии** становится активным;
- **Доп. информация** — окно **Ординальная информация о критериях** становится активным;
- **Нормализованные данные** — окно **Нормализованные исходные данные** становится активным;
- **Информация** — окно **Информация** становится активным;
- **Результаты** — окно **Результаты выбора** становится активным;

❒ **Отчет** — позволяет запустить команду на создание отчета:

- **Создание отчета** — создание отчета о текущей задаче;

❒ **Справка** — позволяет получить информацию о системе и вызвать справку:

- **Вызов справки** — показывает на экране справку по QuickChoice;
- **Поиск** — запускает справку и позволяет осуществить по ней поиск;
- **О программе** — выводит информацию о программном продукте.

Следует отметить, что большинство команд главного меню содержится в контекстных меню, открываемых щелчком правой кнопки мыши на соответствующем рабочем окне.

16.12. Рабочие окна

Рассмотрим более подробно каждое из диалоговых окон.

16.12.1. Окно *Задание альтернатив*

В этом окне располагаются альтернативы и их значения по заданным критериям. По строкам располагаются введенные альтернативы, по столбцам — заданные критерии. Например, на рис. 16.14 альтернатива **Вариант 1** характеризуется следующими значениями по заданным критериям.

Модель	ГАЗ-21
Год выпуска	59
Цена, $	1000
Цвет	Коричневый
Состояние	На ходу

Кол-во передач	четыре
Объем двигателя	1200
Кол-во дверей	4
Расход топлива	12

Аналогично задаются все альтернативы.

При щелчке правой кнопки мыши в окне альтернатив открывается контекстное меню со следующими командами:

- **Удалить** — удаление альтернативы;
- **Добавить** — добавление альтернативы;
- **Редактировать имя варианта** — редактирование названия альтернативы; эта команда позволяет редактировать имя альтернативы, которое располагается в таблице альтернатив и таблице результатов;
- **Шрифт** — задание шрифта для данного окна;
- **Автоформат** — включение/выключение автоматической установки ширины столбцов в данном окне.

Задание альтернатив

	Модель	Год выпуск	Цена, $	Цвет	Состояние	Кол-во пе	Объем дв	Кол-во дв	Расход т
Вариант 1	ГАЗ-21	59	1000	Коричневы	На ходу	четыре	1200	4	12
Вариант 2	ГАЗ-21	64	1100	Синий	На ходу	четыре	1200	4	12
Вариант 3	ГАЗ-21	66	1200	Бежевый	На ходу	четыре	1200	4	12
Вариант 4	ГАЗ-24	83	2200	Красный	Хорошее	четыре	1600	4	13
Вариант 5	ГАЗ-24	84	2000	Голубой	Нормальн	четыре	1800	4	13
Вариант 6	ГАЗ-24	82	2500	Белый	Хорошее	четыре	1800	4	13
Вариант 7	ГАЗ-24	78	1500	Белый	Нормальн	четыре	1800	4	13
Вариант 8	ГАЗ-24	84	1900	Черный	На ходу	четыре	1800	4	13
Вариант 9	ГАЗ-24	93	4300	Черный	Отличное	четыре	1800	4	13
Вариант 10	ГАЗ-24	92	4500	Черный	Хорошее	четыре	1800	4	13
Вариант 11	ГАЗ-24	82	2300	Белый	На ходу	четыре	1800	4	13
Вариант 12	ГАЗ-3102	96	7500	Черный	Отличное	четыре	1800	4	15
Вариант 13	ГАЗ-3102	93	5600	Красный	Хорошее	пять	1800	4	15
Вариант 14	ГАЗ-3102	97	7700	Черный	Отличное	пять	1800	4	15
Вариант 15	ГАЗ-3102	96	7000	Белый	Хорошее	четыре	1800	4	15

Рис. 16.14. Окно **Задание альтернатив**

Если навести указатель мыши на значение перечислимого критерия, то система предлагает выбрать одно из его заданных значений (рис. 16.15).

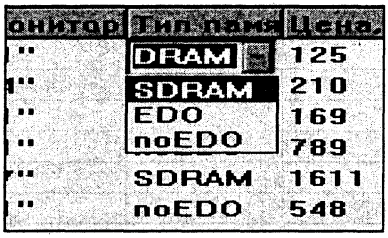


Рис. 16.15. Задание значения перечислимого критерия

Добавление альтернативы

Добавить новые альтернативы к исходному набору можно несколькими способами. Если пользователю нужно добавить одну альтернативу, то можно воспользоваться командой **Добавить** контекстного меню окна альтернатив или командой **Альтернативы \ Добавить** главного меню. Если требуется добавить несколько альтернатив в систему, можно воспользоваться командой **Альтернативы \ Количество** главного меню. При этом будет вызван соответствующий диалог. Следует отметить, что при помощи этой команды можно только добавлять альтернативы.

Удаление альтернативы

Удалить альтернативу из текущего набора можно при помощи соответствующей команды контекстного меню окна альтернатив или при помощи команды **Альтернативы \ Удалить** главного меню.

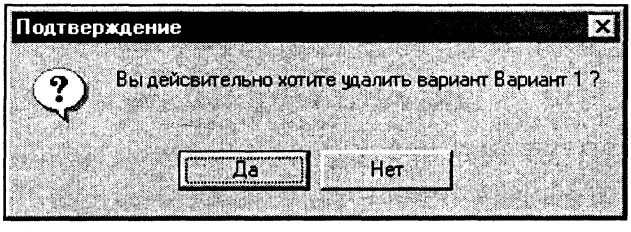


Рис. 16.16. Подтверждение удаления альтернативы

Перед выбором одной из этих команд необходимо указать удаляемую альтернативу при помощи мыши в окне альтернатив. После этого программа запросит подтверждение (рис. 16.16). И если пользователь согласится, то альтернатива будет удалена.

Изменение названия альтернативы

Изменить название альтернативы можно при помощи команды **Редактировать имя варианта** контекстного меню окна альтернатив, открываемого щелчком правой кнопки мыши в этом окне. После этого откроется диалоговое окно, позволяющее задать новое имя варианта (рис. 16.17). Имя варианта, используемое по умолчанию, можно задать при помощи команды **Опции \ Новая задача** главного меню.

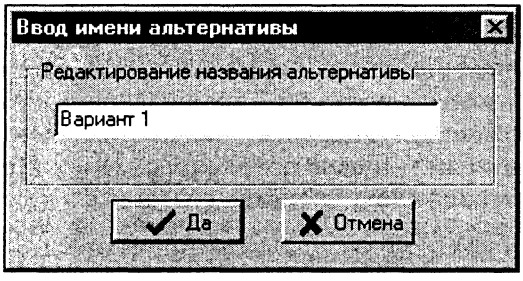


Рис. 16.17. Изменение имени альтернативы

16.12.2. Окно *Критерии*

Это окно (рис. 16.18) предназначено для задания критериев и их свойств. В данном окне критерии представлены в виде таблицы, отображающей названия критериев и их свойства: тип, цель, минимальное и максимальное значения.

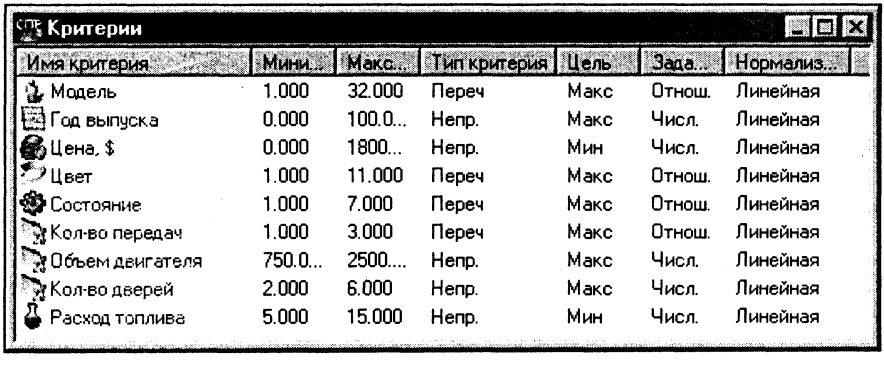


Рис. 16.18. Окно **Критерии**

Щелчок правой кнопки мыши открывает контекстное меню, содержащее следующие команды:

- **Добавить** — добавление нового критерия;
- **Удалить** — удаление критерия;
- **Свойства** — задание свойств критерия;
- **Шрифт** — задание шрифта для данного окна.

Добавление критерия

Добавить новые критерии к исходному набору можно несколькими способами. Если пользователю нужно добавить один критерий, то можно воспользоваться командой **Добавить** контекстного меню окна **Критерии** или командой **Критерии \ Добавить** главного меню. При этом пользователю будет предложено задать свойства созданного критерия.

Если требуется добавить сразу несколько критериев, следует воспользоваться командой **Критерии \ Количество** главного меню. Эта команда открывает диалоговое окно, позволяющее задать количество критериев (рис. 16.19). Нельзя задать количество критериев меньше, чем их было в системе; при такой попытке будет выдано сообщение (рис. 16.20).

Рис. 16.19. Задание количества критериев

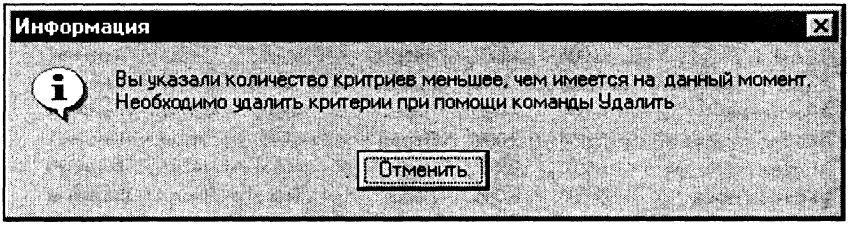


Рис. 16.20. При помощи команды **Количество** нельзя уменьшать количество критериев

При использовании команды **Количество** свойства добавляемых критериев будут заданы по умолчанию. Эти свойства можно задать при помощи команды **Опции \ Новая задача**.

Удаление критерия

Для удаления критерия следует воспользоваться командой **Критерии \ Удалить** главного меню или командой **Удалить** контекстного меню окна **Критерии** (перед этим нужно выбрать удаляемый критерий в окне **Критерии**). Система не может работать с количеством критериев меньше двух и при попытке удаления одного из двух критериев будет выдано сообщение об ошибке.

Изменение свойств критерия

Для изменения свойств критерия следует воспользоваться командой **Критерии \ Свойства** главного меню или командой **Свойства** контекстного меню окна **Критерии** (перед этим нужно выбрать удаляемый критерий в окне **Критерии**). Если критерий не выбран, то откроется диалоговое окно (рис. 16.21), в котором пользователь должен выбрать изменяемый критерий.

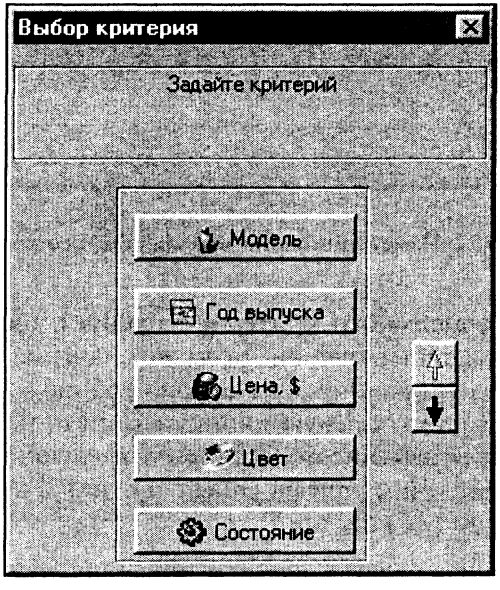


Рис. 16.21. Выбор критерия

Диалоги изменения свойств критерия различны для перечислимых и непрерывных критериев (рис. 16.22 и 16.23). Пользователь должен изменить нужные свойства критерия и нажать кнопку **Да**. Различные типы критериев описаны в *разд. 16.2*. Для перечислимого критерия необходимо задать значения, которые он может принимать, при помощи кнопок **Добавить** и **Удалить**.

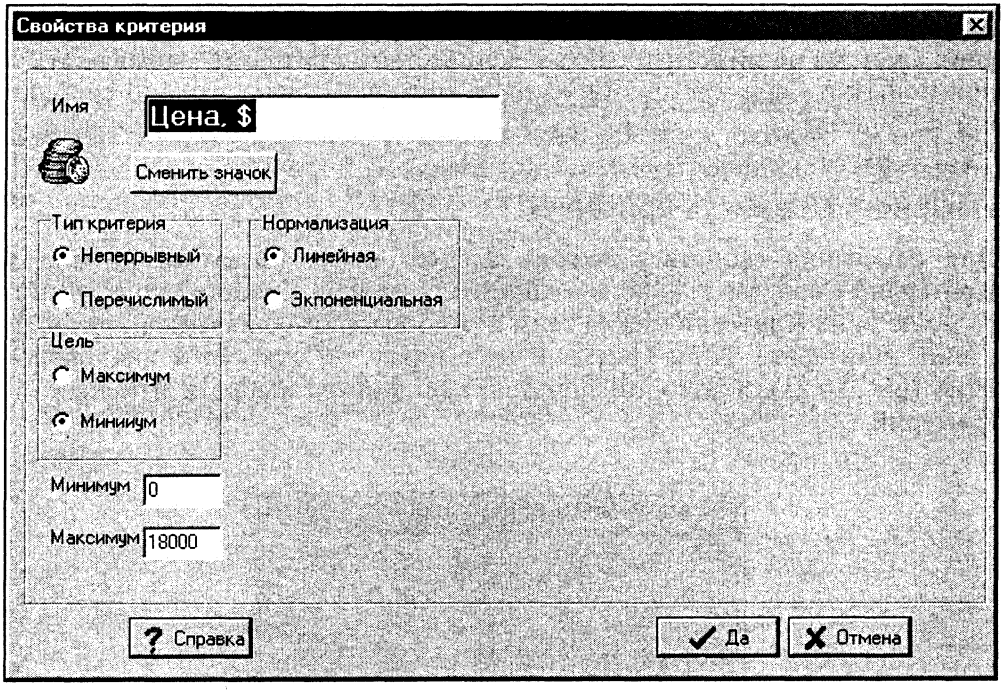


Рис. 16.22. Изменение свойств непрерывного критерия

Если значения перечислимого критерия задаются численно, то необходимо ввести эти численные оценки. Если же значения задаются перечислением, то их нужно упорядочить по предпочтению — навести указатель мыши на критерий, нажать левую кнопку мыши и, не отпуская ее, перетащить значение на нужное место. Изменить введенные значения можно, щелкнув правой кнопкой мыши на списке и выбрав в открывшемся контекстном меню команду **Свойства**.

Для изменения значка редактируемого критерия нужно нажать кнопку **Сменить значок** и выбрать новый значок (рис. 16.24).

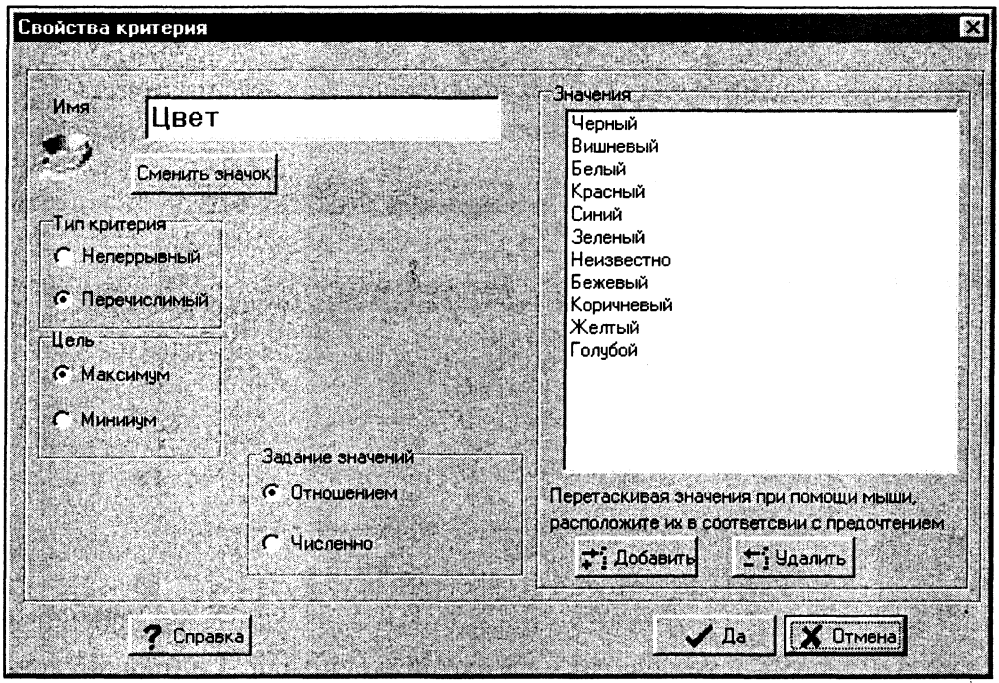


Рис. 16.23. Изменение свойств перечислимого критерия

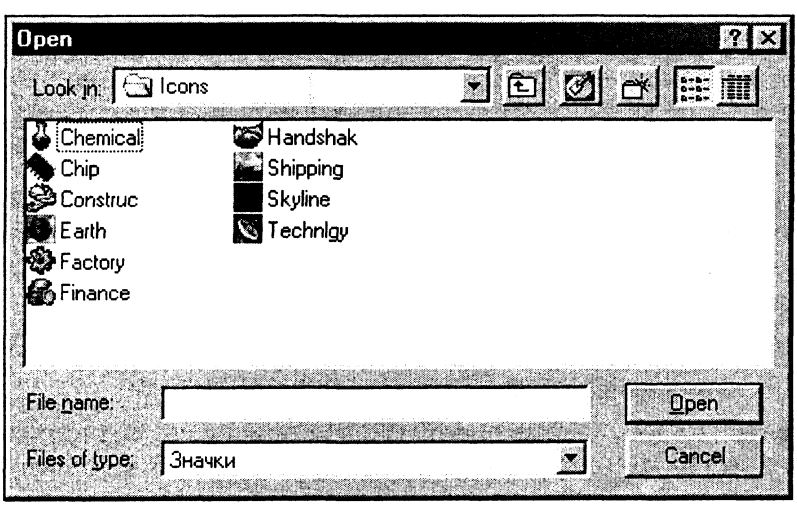


Рис. 16.24. Задание значка

16.12.3. Окно *Ординальная информация о критериях*

В этом окне выводится ординальная информация о критериях (рис. 16.25).

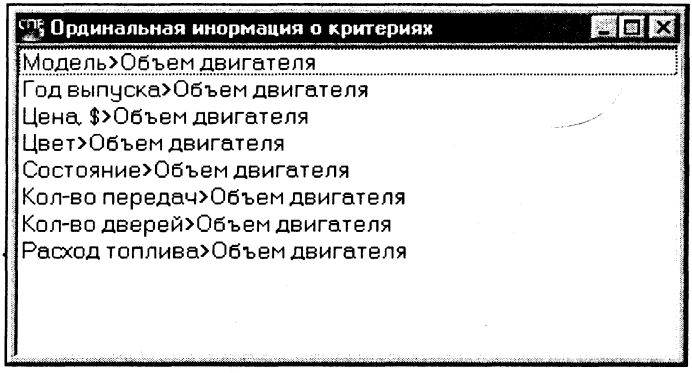


Рис. 16.25. Ординальная информация о критериях

Окно содержит строки ординальной информации типа "Критерий 1 важнее Критерия 2" и "Критерий 1 равноценен Критерию 2". Соотношения важности и равноценности соответственно обозначаются символами ">" и "=".

ЛПР, предоставляя информацию о том, что один критерий важнее другого, предполагает, что после нормализации исходных данных любое увеличение более важного критерия при уменьшении менее важного критерия на такую же величину улучшит оценку.

ЛПР, предоставляя информацию о равноценности критериев, предполагает, что после нормализации исходных данных для него безразлично любое уменьшение одного из равноценных критериев при увеличении второго на такую же величину.

Щелчок правой кнопки мыши на окне ординальной информации открывает контекстное меню, содержащее следующие команды:

- **Добавить** — добавление новой строки ординальной информации о критериях;
- **Удаление** — удаление строки ординальной информации;
- **Шрифт** — задание шрифта для данного окна.

Добавление ординальной информации о критериях

Для добавления ординальной информации о критериях можно воспользоваться командой **Доп. информация \ Добавить** главного меню или командой **Добавить** контекстного меню окна **Ординальная информация о критериях**. При этом откроется диалоговое окно **Ввод дополнительной информации о критериях** (рис. 16.26).

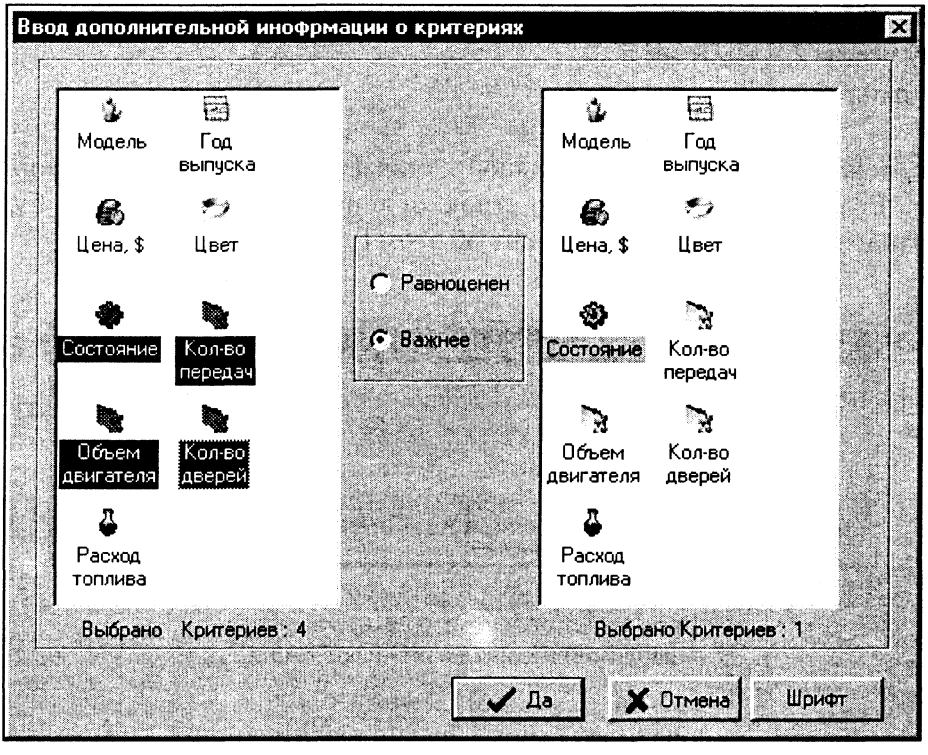


Рис. 16.26. Добавление ординальной информации о критериях

Данная система позволяет задавать сразу несколько строк ординальной информации. Для этого надо выбрать первую группу критериев, выбрать вторую группу и задать отношение. Для выделения нескольких критериев следует указывать выбираемые критерии мышью, удерживая при этом клавишу <Ctrl> нажатой. После этого нужно нажать кнопку **Да**. Добавление строк ординальной информации будет проводиться по следующему принципу: для каждого из выбранных критериев левой части будет

добавляться информация о важности или равноценности по отношению к каждому из критериев, выбранных в правой части. Например, если выбрать в левой части критерии 1, 2, 3, а в правой — критерии 4, 5, то будут добавлены следующие строки ординальной информации:

Критерий 1 > Критерий 4
Критерий 2 > Критерий 4
Критерий 3 > Критерий 4
Критерий 1 > Критерий 5
Критерий 2 > Критерий 5
Критерий 3 > Критерий 5

Так можно быстро ввести большое количество ординальной информации. Если среди добавляемых строк ординальной информации будут противоречия, то система сообщит об этом (рис. 16.27).

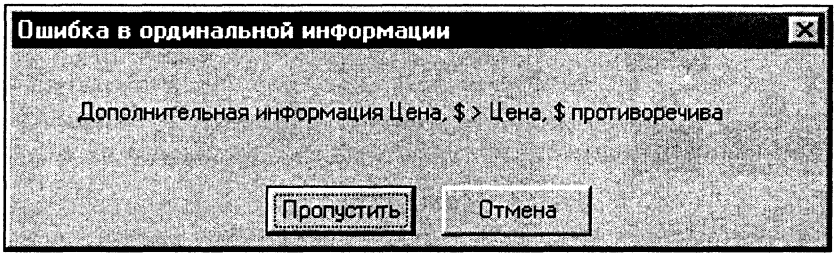


Рис. 16.27. Сообщение о некорректности вводимой ординальной информации

Чтобы проигнорировать эту строку, нужно нажать кнопку **Пропустить**, в этом случае программа продолжит дальнейшее создание строк ординальной информации в соответствии с выбранными критериями. Для отмены выбора следует нажать кнопку **Отмена**. Противоречие появляется, если после дополнения информации о предпочтениях в соответствии с принципом транзитивности возникает, например, строка "Критерий 1 важнее Критерия 1" или одновременно две строки — "Критерий 1 важнее Критерия 2" и "Критерий 2 важнее Критерия 1". Однако в последнем случае сообщение о противоречивости будет выдано только при запуске выбора.

Удаление ординальной информации о критериях

Для удаления строки ординальной информации следует, выделив нужную строку, нажать клавишу <Del> или выбрать соответствующую команду в меню. При этом программа запросит подтверждение на удаление этой строки (рис. 16.28).

Рис. 16.28. Удаление строки ординальной информации

16.12.4. Окно *Нормализованные исходные данные*

В этом окне выводятся нормализованные исходные данные. Поскольку в методе t-упорядочения подразумевается однородность критериев, предварительно необходимо их нормализовать. В данной системе используется следующий способ нормализации исходных данных.

Нормализованные исходные данные

	Модель	Год выпус	Цена, $	Цвет	Состояние	Кол-во пер	Объем дви	Кол-во две	Расход
Вариант 1	---	0.59000	-0.05556	---	---	---	0.25714	0.50000	-0.7000(
Вариант 2	---	0.64000	-0.06111	---	---	---	0.25714	0.50000	-0.7000(
Вариант 3	---	0.66000	-0.06667	---	---	---	0.25714	0.50000	-0.7000(
Вариант 4	---	0.83000	-0.12222	---	---	---	0.48571	0.50000	-0.8000(
Вариант 5	---	0.84000	-0.11111	---	---	---	0.60000	0.50000	-0.8000(
Вариант 6	---	0.82000	-0.13889	---	---	---	0.60000	0.50000	-0.8000(
Вариант 7	---	0.78000	-0.08333	---	---	---	0.60000	0.50000	-0.8000(
Вариант 8	---	0.84000	-0.10556	---	---	---	0.60000	0.50000	-0.8000(
Вариант 9	---	0.93000	-0.23889	---	---	---	0.60000	0.50000	-0.8000(
Вариант 10	---	0.92000	-0.25000	---	---	---	0.60000	0.50000	-0.8000(
Вариант 11	---	0.82000	-0.12778	---	---	---	0.60000	0.50000	-0.8000(
Вариант 12	---	0.96000	-0.41667	---	---	---	0.60000	0.50000	-1.0000(
Вариант 13	---	0.93000	-0.31111	---	---	---	0.60000	0.50000	-1.0000(
Вариант 14	---	0.97000	-0.42778	---	---	---	0.60000	0.50000	-1.0000(
Вариант 15	---	0.96000	-0.38889	---	---	---	0.60000	0.50000	-1.0000(

Рис. 16.29. Окно **Нормализованные исходные данные**

Если критерий P максимизируется, то нормализованное значение вычисляется по формуле:

$$Alt(i,j)=\frac{Alt^*(i,j)-K_{p\min}(j)}{K_{p\max}(j)-K_{p\min}(j)}.$$

Если критерий P минимизируется, то нормализованное значение вычисляется по формуле:

$$Alt(i,j)=\frac{K_{p\max}(j)-Alt^*(i,j)}{K_{p\max}(j)-K_{p\min}(j)}.$$

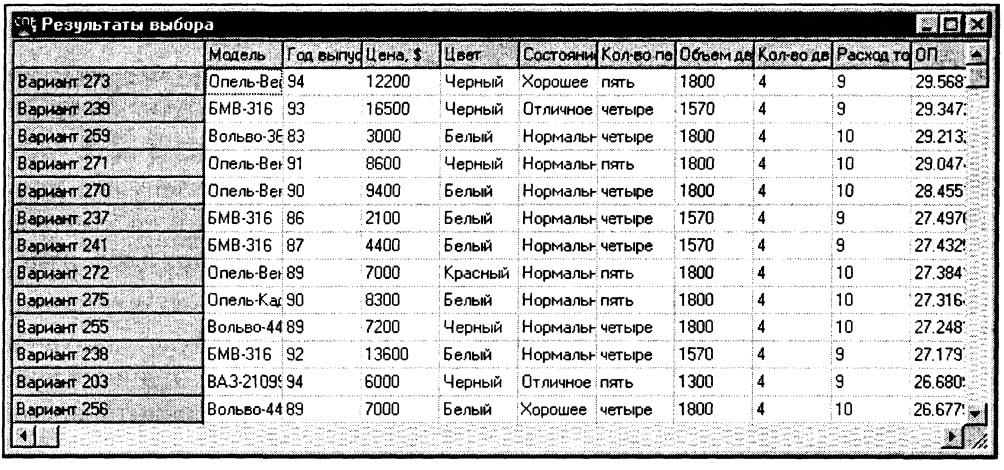
Здесь: Alt^* — ненормализованные исходные данные; Alt — нормализованные исходные данные; $K_{p\max}(j)$, $K_{p\min}(j)$ — максимальные и минимальные значения критериев, заданные пользователем.

Окно **Нормализованные исходные данные** показано на рис. 16.29.

16.12.5. Окно *Результаты выбора*

В окне **Результаты выбора** (рис. 16.30) представлены альтернативы, отобранные после предварительного выбора. Для запуска выбора необходимо воспользоваться главным меню или нажать клавишу <F9>. При этом осуществляется запуск алгоритма принятия решения и построения множества оптимальных вариантов. В случае если исходная информация некорректна, будет выдано соответствующее сообщение об ошибке (рис. 16.31). Такая ошибка может быть вызвана некорректностью ординальной информации. После правильного завершения работы алгоритма в окне **Информация** выводится статистика работы алгоритма, а окно **Результаты выбора** становится активным. Альтернативы в окне упорядочены по обобщенному значению и в соответствии с данным порядком будут предлагаться в методе ограничений. ОЗ располагается в правом столбце таблицы выбранных альтернатив. Построение ОЗ кратко описано в *разд. 16.5*. При большом (более 10) количестве критериев предварительный отбор может выполняться достаточно долго.

Имеется возможность отключить предварительный отбор, а также задать максимальное время его прохождения. Если выбор будет продолжаться дольше установленного времени, система выдаст сообщение об ошибке (рис. 16.32).



	Модель	Год выпус	Цена, $	Цвет	Состояни	Кол-во пе	Объем дв	Кол-во дв	Расход то	ОП
Вариант 273	Опель-Век	94	12200	Черный	Хорошее	пять	1800	4	9	29.568
Вариант 239	БМВ-316	93	16500	Черный	Отличное	четыре	1570	4	9	29.347
Вариант 259	Вольво-3Е	83	3000	Белый	Нормальн	четыре	1800	4	10	29.213
Вариант 271	Опель-Век	91	8600	Черный	Нормальн	пять	1800	4	10	29.047
Вариант 270	Опель-Век	90	9400	Белый	Нормальн	четыре	1800	4	10	28.455
Вариант 237	БМВ-316	86	2100	Белый	Нормальн	четыре	1570	4	9	27.497
Вариант 241	БМВ-316	87	4400	Белый	Нормальн	четыре	1570	4	9	27.432
Вариант 272	Опель-Век	89	7000	Красный	Нормальн	пять	1800	4	10	27.384
Вариант 275	Опель-Ка	90	8300	Белый	Нормальн	пять	1800	4	10	27.316
Вариант 255	Вольво-44	89	7200	Черный	Нормальн	четыре	1800	4	10	27.248
Вариант 238	БМВ-316	92	13600	Белый	Нормальн	четыре	1570	4	9	27.179
Вариант 203	ВАЗ-2109	94	6000	Черный	Отличное	пять	1300	4	9	26.680
Вариант 256	Вольво-44	89	7000	Белый	Хорошее	четыре	1800	4	10	26.677

Рис. 16.30. Результаты выбора

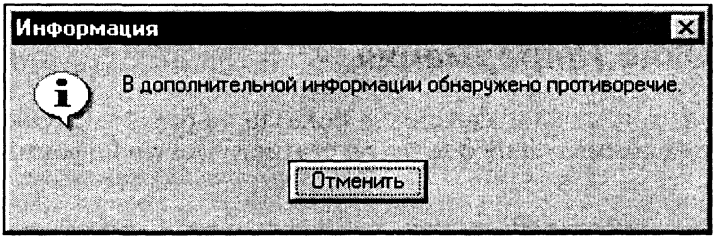


Рис. 16.31. Некорректность ординальной информации

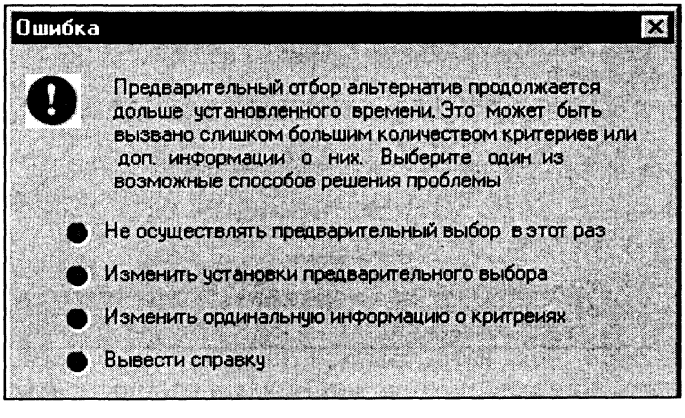


Рис. 16.32. Время предварительного отбора превысило допустимое

При этом пользователю предлагается выбрать один из вариантов:

- **Не осуществлять предварительный выбор в этот раз** — предварительный выбор не будет закончен, однако при повторном запуске он будет проводиться вновь;
- **Изменить установки предварительного выбора** — будет запущен диалог задания параметров метода ограничений, в котором пользователь сможет либо совсем отключить предварительный выбор, либо изменить максимальное время его прохождения;
- **Изменить ординальную информацию о критериях** — будет запущен диалог, позволяющий пользователю отредактировать ординальную информацию о критериях. Это может позволить уменьшить время предварительного отбора;
- **Вывести справку** — выводится справка, которая описывает причины сложившейся проблемы и возможности ее решения.

16.12.6. Окно *Информация*

Внешний вид этого диалогового окна показан на рис. 16.33. Оно содержит поле для ввода комментария о задаче и статистическую информацию.

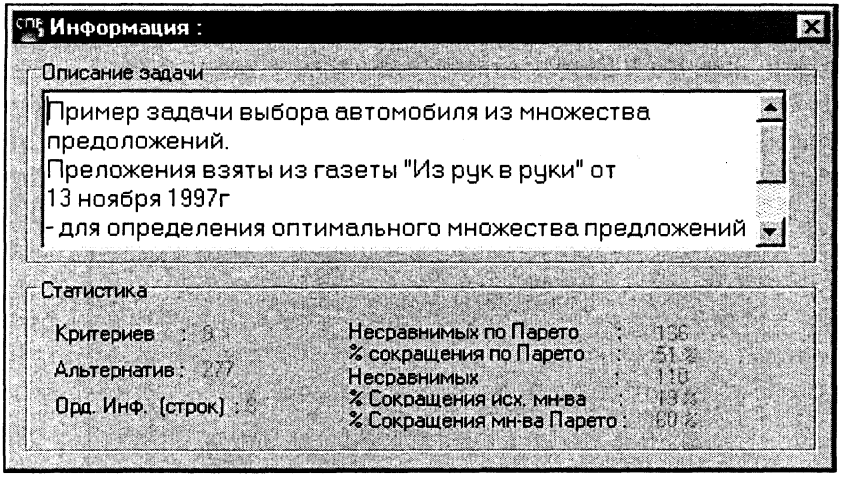


Рис. 16.33. Диалоговое окно **Информация**

16.13. Создание, загрузка и сохранение задачи

16.13.1. Создание новой задачи

Для создания новой задачи нужно выбрать пункт **Файл \ Новый** главного меню. Параметры, принимаемые при создании новой задачи, можно задать при помощи команды **Новая задача** меню **Опции**, открывающей диалоговое окно **Задание параметров новой задачи** (рис. 16.34).

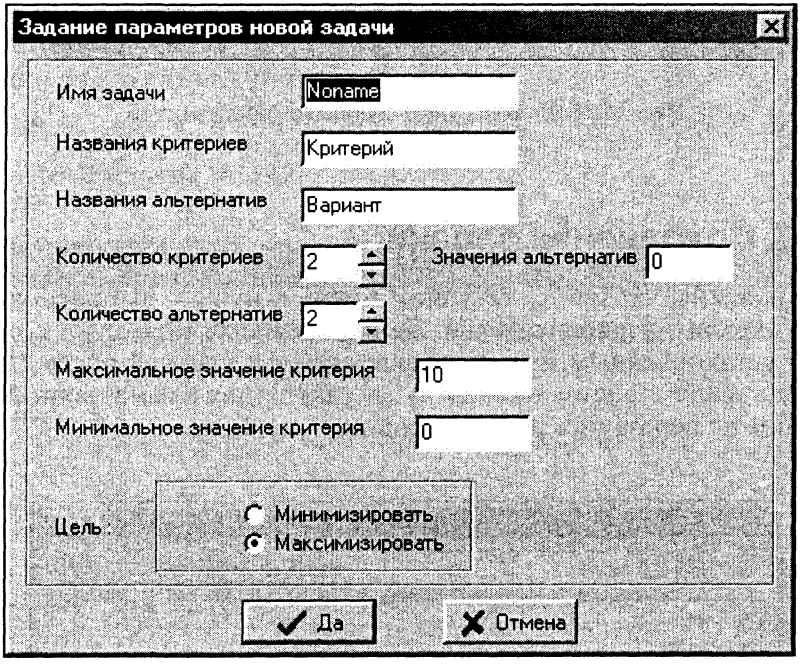


Рис. 16.34. Диалоговое окно задания параметров новой задачи

16.13.2. Загрузка существующей задачи

Загрузка существующей задачи производится при помощи команды **Файл \ Открыть** главного меню. При этом программа открывает стандартное диалоговое окно, в котором запрашивается имя загружаемой задачи (рис. 16.35).

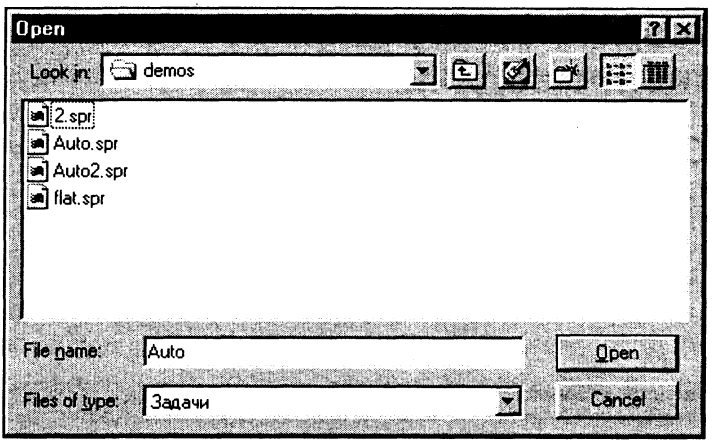


Рис. 16.35. Выбор существующего файла задачи

16.13.3. Сохранение задачи

Для сохранения задачи с текущим именем нужно выбрать команду **Файл \ Сохранить** главного меню. Данная команда позволяет сохранить задачу с установленным именем, показанным в верхней части главного окна. Если файл с таким именем существует, программа запросит подтверждение на перезапись данного файла.

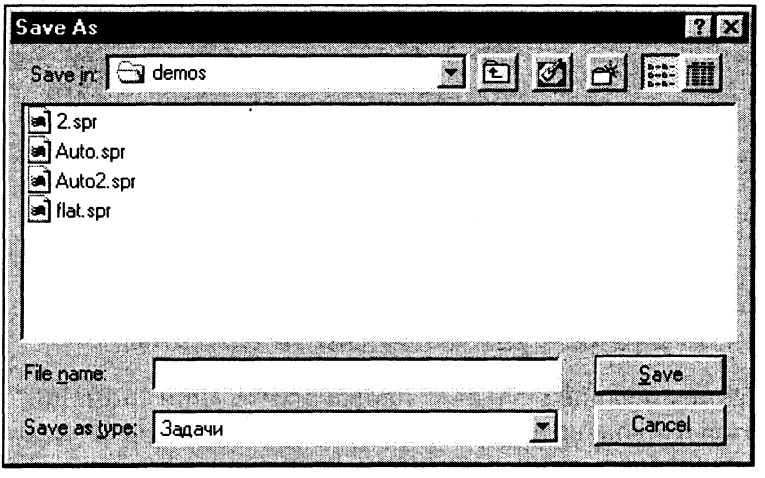


Рис. 16.36. Диалоговое окно сохранения задачи с новым именем

Для сохранения задачи в файле с новым именем следует выбрать команду **Файл \ Сохранить как** главного меню. При этом открывается стандартный диалог, в котором можно ввести новое имя файла (рис. 16.36).

16.14. Создание отчета

Система Quick Choice позволяет сохранить в файле описание критериев, альтернатив, ординальную информацию, а также результаты выбора. Для создания файла отчета нужно выбрать в главном меню команду **Отчет \ Создание отчета**. При этом на экране появится диалоговое окно **Создание файла отчета** (рис. 16.37). Данная система предполагает выбрать один из двух типов отчетов:

- **Текстовый файл**;
- **Документ Microsoft Word**.

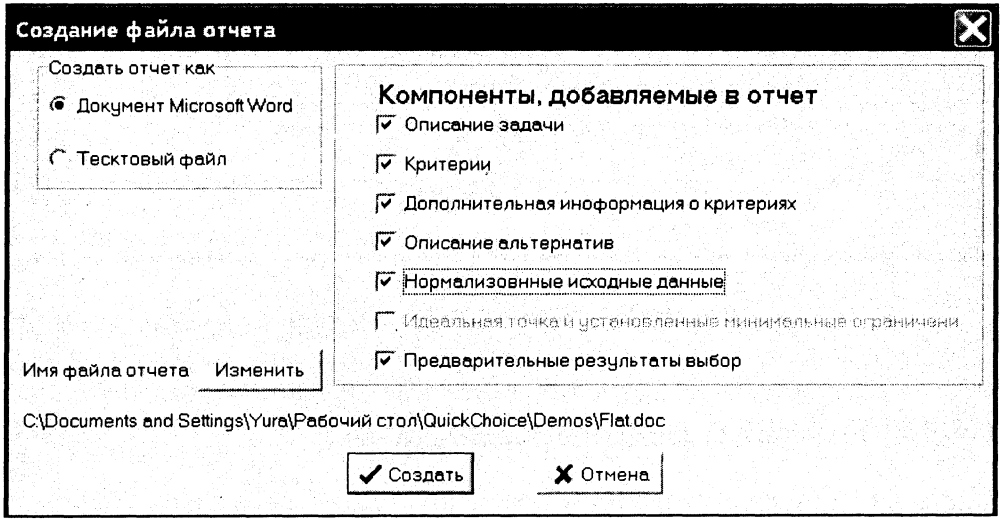


Рис. 16.37. Диалоговое окно **Создание файла отчета**

Для создания отчета в формате Microsoft Word необходимо, чтобы это приложение было уже установлено на компьютере. Создание отчета в этом формате при большом количестве альтернатив и критериев может занять довольно много времени.

В этом диалоговом окне можно задать имя файла, в который будет помещен отчет, и указать компоненты, которые следует добавлять в отчет. Для добавления в отчет результатов выбора необходимо сначала провести вычисление.

После задания параметров отчета следует нажать кнопку **Создать**. После этого отчет будет создан в файле с заданным именем (рис. 16.38).

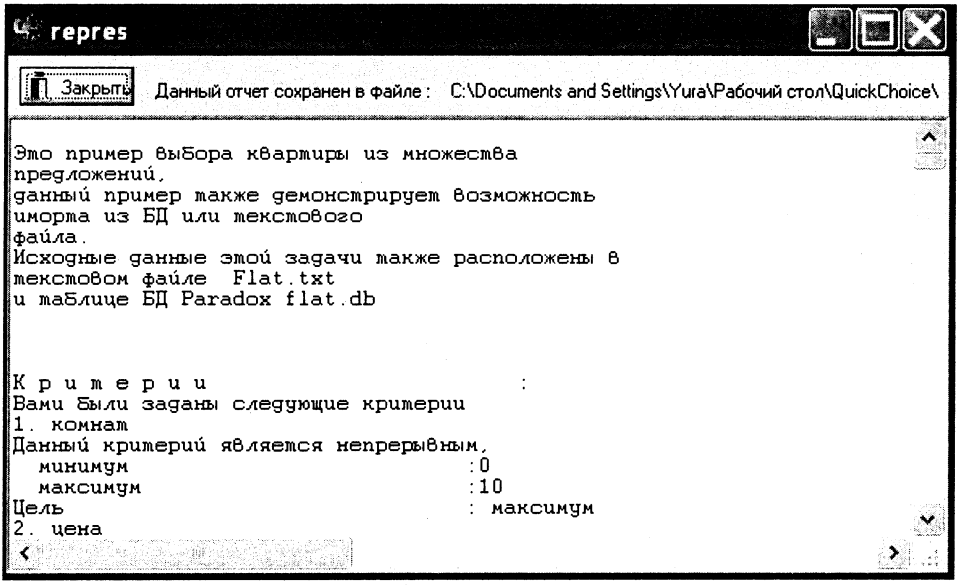


Рис. 16.38. Отчет в текстовом формате

16.15. Пример решения задачи

В качестве примера рассмотрим следующую задачу. Имеется набор различных предложений по продаже автомобилей; пользователю необходимо выбрать наиболее устраивающий его вариант. Альтернативы описываются следующими критериями:

- Модель
- Год выпуска
- Цена, $
- Цвет

- Состояние
- Кол-во передач
- Объем двигателя
- Кол-во дверей
- Расход топлива

Для каждого из перечисленных критериев необходимо определить тип и задать соответствующие параметры. Для перечислимых критериев необходимо перечислить все принимаемые значения.

После запуска системы выбираем пункт **Создание новой задачи в диалоге с системой**. Затем задаем количество критериев — 9 и вводим названия критериев. Теперь система будет задавать вопросы по каждому из критериев. Например, для критерия "Модель" вопросы представлены в табл. 16.1.

Таблица 16.1. *Диалог для критерия "Модель"*

Вопрос системы	Ответ пользователя
Какова ваша цель по критерию "Модель"?	Максимум
Каким образом вы хотите задать критерий "Модель"?	Перечислив все его значения
Можете ли вы каждому перечисленному значению критерия "Модель" сопоставить числовое значение?	Нет

Затем необходимо задать все значения критерия "Модель" и расположить их в порядке предпочтения. После этого аналогичный диалог будет проведен для каждого из критериев. Возможные типы критериев и их свойства показаны в табл. 16.2, а принимаемые значения критериев — в табл. 16.3.

Таблица 16.2. *Возможные типы критериев и их свойства*

№	Критерий	Тип	Задание	Цель	Мин.	Макс.
1	Модель	Перечислимый	Отношением	Максимум	—	—
2	Год выпуска	Непрерывный	Численно	Максимум	0	100
3	Цена, $	Непрерывный	Численно	Минимум	0	18 000
4	Цвет	Перечислимый	Отношением	Максимум	—	—

Таблица 16.2 (окончание)

№	Критерий	Тип	Задание	Цель	Мин.	Макс.
5	Состояние	Перечислимый	Отношением	Максимум	—	—
6	Кол-во передач	Перечислимый	Отношением	Максимум	—	—
7	Объем двигателя	Непрерывный	Численно	Максимум	750	2500
8	Кол-во дверей	Непрерывный	Численно	Максимум	2	6
9	Расход топлива	Непрерывный	Численно	Минимум	5	15

Таблица 16.3. Принимаемые значения критериев

Критерий	Принимаемые значения
Модель	БМВ-520 БМВ-320 Вольво-360 Опель-Вектра БМВ-316 ... М-412 ИЖ-412 ЗАЗ-968М
Цвет	Черный Вишневый Белый Красный Синий Зеленый Неизвестно Бежевый Коричневый Желтый Голубой

На этом задание критериев заканчивается и нужно задать альтернативы. Система предложит задать количество альтернатив и ввести значения альтернатив по каждому из критериев. В табл. 16.4 показан пример задания альтернатив.

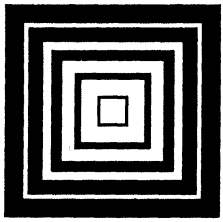
Таблица 16.4. Альтернативы

№	Модель	Год выпуска	Цена, $	Цвет	Состояние	Колво передач	Объем двигателя	Колво дверей	Расход топлива
Вариант 1	ГАЗ-21	59	1000	Коричневый	На ходу	четыре	1200	4	12
Вариант 2	ГАЗ-21	64	1100	Синий	На ходу	четыре	1200	4	12
Вариант 3	ГАЗ-21	66	1200	Бежевый	На ходу	четыре	1200	4	12
Вариант 4	ГАЗ-24	83	2200	Красный	Хорошее	четыре	1600	4	13
Вариант 5	ГАЗ-24	84	2000	Голубой	Норм.	четыре	1800	4	13
Вариант 6	ГАЗ-24	82	2500	Белый	Хорошее	четыре	1800	4	13
Вариант 7	ГАЗ-24	78	1500	Белый	Норм.	четыре	1800	4	13
...									
Вариант 275	Опель-Кадетт	90	8300	Белый	Нормальное	пять	1800	4	10
Вариант 276	Опель-Кадетт	88	8500	Белый	Нормальное	пять	1800	4	10
Вариант 277	Опель-Кадетт	94	4200	Желтый	Нормальное	пять	1800	4	10

После задания альтернатив требуется задать ординальную информацию о критериях. Пусть для ЛПР наиболее важными являются критерии "Цена, $" и "Год выпуска". Система предложит выбрать один из видов ординальной информации, которые описаны ранее. Зададим информацию о большей важности одного критерия по сравнению с другими. Выберем критерий "Цена, $" и поставим его более важным, чем все остальные критерии. Аналогичным образом зададим информацию о критерии

"Год выпуска". После задания дополнительной информации будет проведен предварительный выбор. Затем будет запущен метод ограничений. Пользователю будет предложена одна из альтернатив, например, "Вариант 7". Пусть эта альтернатива не устраивает пользователя. Выбираем вариант **Нет, эта альтернатива меня не устраивает**. После этого система попросит указать критерий, по которому данная альтернатива не удовлетворяет пользователя (например, "Модель"), и попросит задать наихудшее значение данного критерия, которое удовлетворит пользователя. Далее будет предложена следующая альтернатива "Вариант 75". Эта альтернатива не удовлетворяет пользователя по цене. После задания верхнего предела цены осталось две альтернативы, удовлетворяющие ЛПР. После согласия ЛПР с предлагаемой альтернативой система предложит указать те части задачи, которые пользователь хочет поместить в отчет, и создаст необходимый отчет о задаче.

Глава 17



NEYDIS — инструментальное средство построения нейлоровских диагностирующих экспертных систем

17.1. Назначение и структура системы

Одна из проблем, возникающих при создании экспертной системы, состоит в учете неточности и ненадежности любой информации, полученной от пользователя. Перед нами стоит задача оценки степени ненадежности наших выводов и рекомендаций при неточности исходной информации и информации, поступающей в результате диалога с пользователем, т. е. работа в условии неопределенности. Существуют различные способы работы в условиях неопределенности. Основные из них — это использование аппарата нечеткой логики и методика, основанная на теории Байеса *(см. главу 15)*. Второй подход весьма неплохо зарекомендовал себя на практике.

Данное инструментальное средство предназначено для создания экспертных систем в произвольной предметной области при условии наличия ненадежных (неопределенных) данных. Оно использует концепцию, предложенную К. Нейлором, основанную на байесовском подходе.

17.2. Функции, реализованные в системе

В системе реализованы следующие функции:

- создание и редактирование базы знаний;
- редактирование названия базы знаний;

- добавление, удаление, редактирование свидетельств;
- добавление, удаление, редактирование гипотез;
- возможность просмотра всех взаимосвязей свидетельств с гипотезами;
- защита от грубых опечаток эксперта;
- подсказки при добавлении свидетельств;
- проверка на уникальность свидетельств и гипотез;
- сохранение/загрузка базы знаний на диске/с диска. Сохранение базы знаний на диске осуществляется только в текстовом формате.

  *.INI — файл, содержащий название и дату последнего редактирования, *.NKB — файл, содержащий гипотезы и все вероятности, *.SKB — файл, содержащий свидетельства и вопросы, задаваемые пользователю.

  После любого законченного действия (например, добавления свидетельства) база автоматически сохраняется на диске;
- отображение результатов работы экспертной системы в процессе работы с последующим сохранением в файле отчета.

17.3. Структура программного средства

Разработанное программное средство состоит из двух независимых модулей:

- редактор базы знаний;
- оболочка экспертной системы.

17.3.1. Редактор базы знаний

Главное окно редактора базы знаний представлено на рис. 17.1. Редактор предназначен для создания, редактирования и последующего сохранения на диске файлов базы знаний. Созданные в редакторе баз файлы могут быть в любой момент загружены для добавления новой информации или редактирования уже имеющейся.

Следует отметить, что в соответствии с терминологией, принятой в теории ЭС, пользователь редактора базы знаний выступает в качестве эксперта или инженера по знаниям.

Пользователь ЭС как конечного результата работы описываемого инструментального средства не имеет возможности работать с редактором базы знаний. Для него предназначена оболочка экспертной системы.

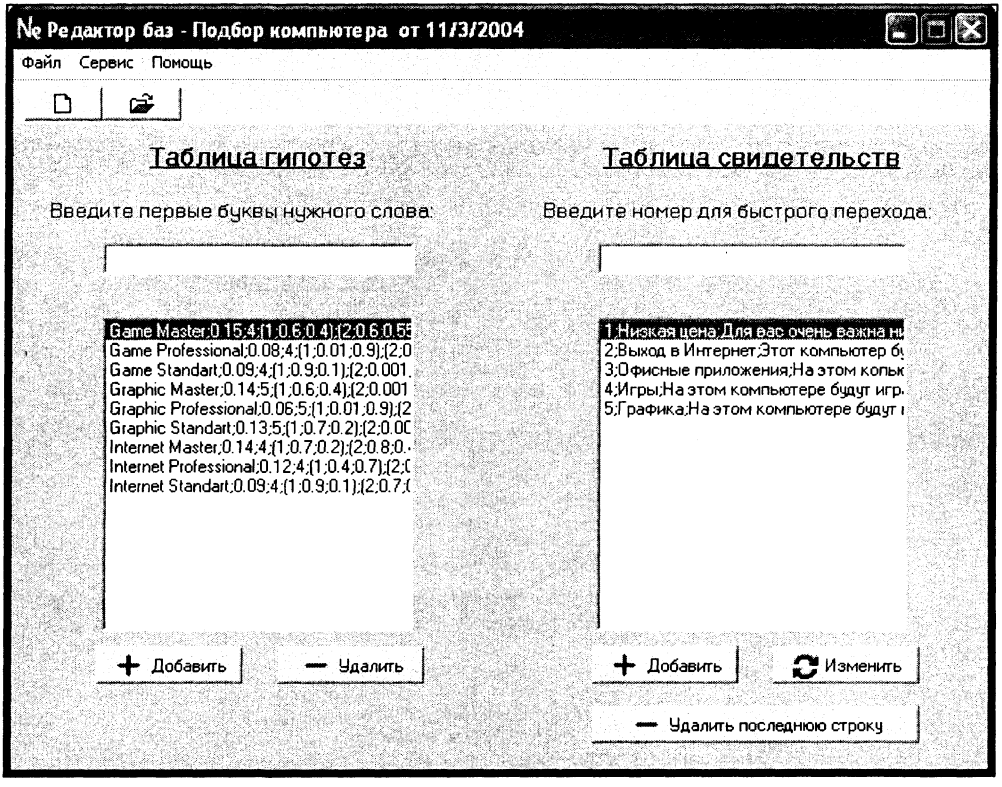


Рис. 17.1. Главное окно редактора базы знаний

Пользователем оболочки по созданию базы знаний должен быть эксперт в той области, в которой создается экспертная система, т. к. будет осуществляться тиражирование его знаний в базу знаний для экспертной системы.

17.3.2. Оболочка экспертной системы

Окно оболочки экспертной системы во время работы показано на рис. 17.2. Данный модуль представляет собой "пустую экспертную систему", которая после загрузки файла с базой знаний "становится" экспертной системой в конкретной предметной области. В процессе работы система в каждый момент определяет наиболее приоритетный вопрос и задает его пользователю. При получении ответа система определяет, может ли она выдать конечный результат; если нет, то определяет следующий задаваемый вопрос.

При нахождении решения система сообщает об этом пользователю. Во время работы отображаются все текущие вероятности гипотез, также их можно посмотреть в файле отчета, который создается автоматически.

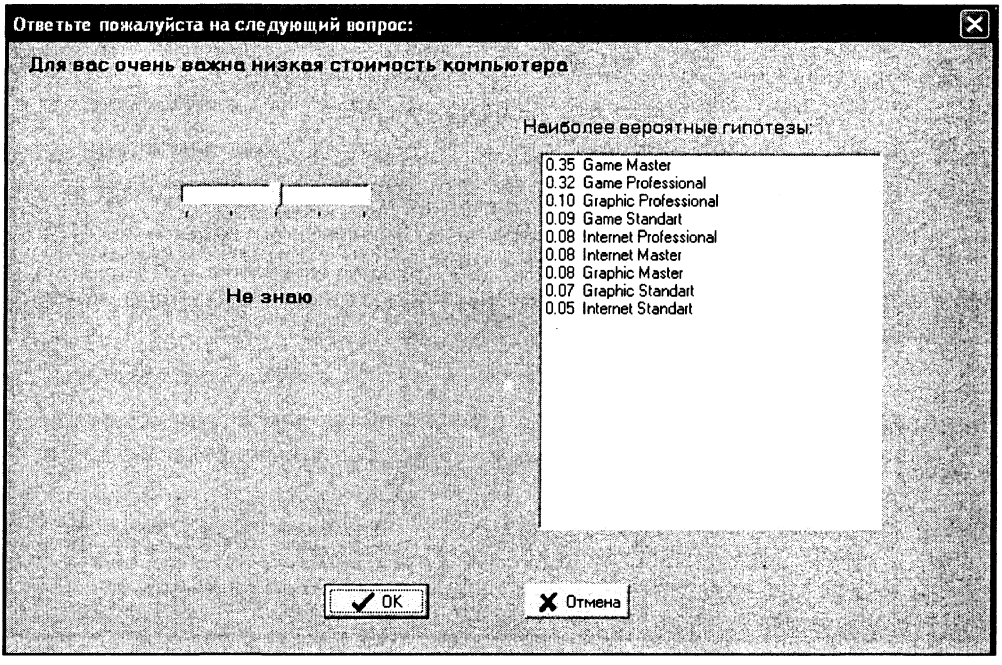


Рис. 17.2. Окно оболочки экспертной системы

Конечным результатом работы данного инструментального средства является конкретная экспертная система в некоторой предметной области. На базе инструментального средства могут быть созданы экспертные системы в таких предметных областях, как медицина, вычислительная техника, геология, математика, управление электроника и др.

17.4. Общая характеристика системы. Системные требования

Для работы системы необходима следующая минимальная конфигурация:

- любой компьютер под управлением операционной системой Windows 95/ 98/98SE/ME/ NT4.0/2000/XP;
- от 20 Мбайт свободного места на жестком диске.

Данная система создания нейлоровских диагностирующих систем была реализована на языке Delphi 6.0.

17.4.1. Представление знаний в экспертных системах

База знаний состоит из трех блоков (файлов):

- общая информация;
- список гипотез и связанные с ними вероятности;
- список свидетельств и вопросы, задаваемые пользователю.

Вся база знаний для экспертной системы реализуется на языке представления знаний и сохраняется на диске в текстовом формате.

17.4.2. Характеристики решаемых задач и квалификация пользователя

Решаемые с помощью данного программного средства задачи имеют широкую область применения. Главная задача — это создание собственных экспертных систем в конкретных предметных областях. Освоив данное инструментальное средство, вы сможете достаточно быстро сгенерировать заказанную вам экспертную систему при наличии необходимой исходной информации. Основная сложность при этом состоит в задании большого количества исходных вероятностных характеристик событий. Эти вероятности часто оказываются неизвестными или известными с ограниченной надежностью. В этом случае необходимо провести соответствующие дополнительные исследования либо ограничиться заданием таких вероятностей, которые и характеризуют ваше незнание. На этом уровне применения описываемого программного продукта требуется достаточно полное знание теории, представленной в соответствующих разделах настоящей книги. Работа же с готовыми экспертными системами оказывается доступной широкому пользователю и не требует дополнительной специальной подготовки, а только умения работать на компьютере.

17.5. Инсталляция системы

Для установки инструментального средства создания нейлоровских ЭС необходимо запустить исполняемый файл SETUP.EXE с инсталляционного диска. После запуска программа выведет на экран краткую инфор-

мацию об устанавливаемом программном средстве. Следуйте указаниям мастера установки.

- Окно приветствия мастера установки. В этом окне отображается общая информация о программе проводящей инсталляции. Для перехода к следующему пункту здесь и дальше необходимо нажимать кнопку **Next** (рис. 17.3).
- Окно **License Agreement**. В этом окне пользователю необходимо прочитать лицензионное соглашение. При согласии следует выбрать пункт **I accept the terms in the license agreement**.
- Окно **Customer Information**. Здесь необходимо ввести информацию о пользователе: его имя и, если нужно, организацию.
- Окно **Destination Folder**. Данное окно предназначено для выбора папки, в которой будет установлена система. В окне отображается путь, предлагаемый компьютером по умолчанию. Для изменения пути установки нажмите кнопку **Change** (рис. 17.4).

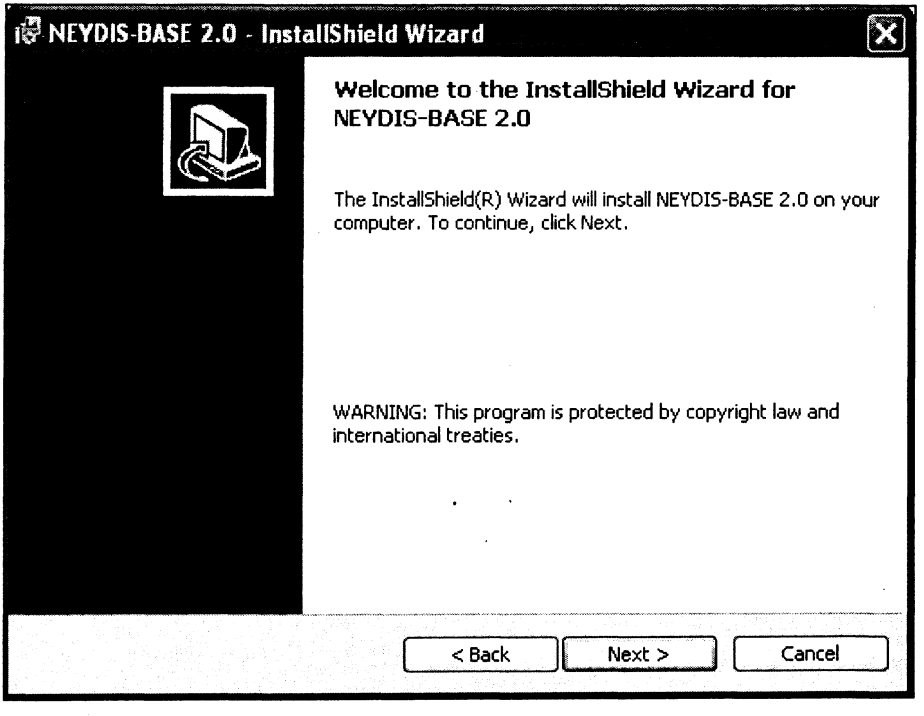


Рис. 17.3. Окно приветствия мастера установки

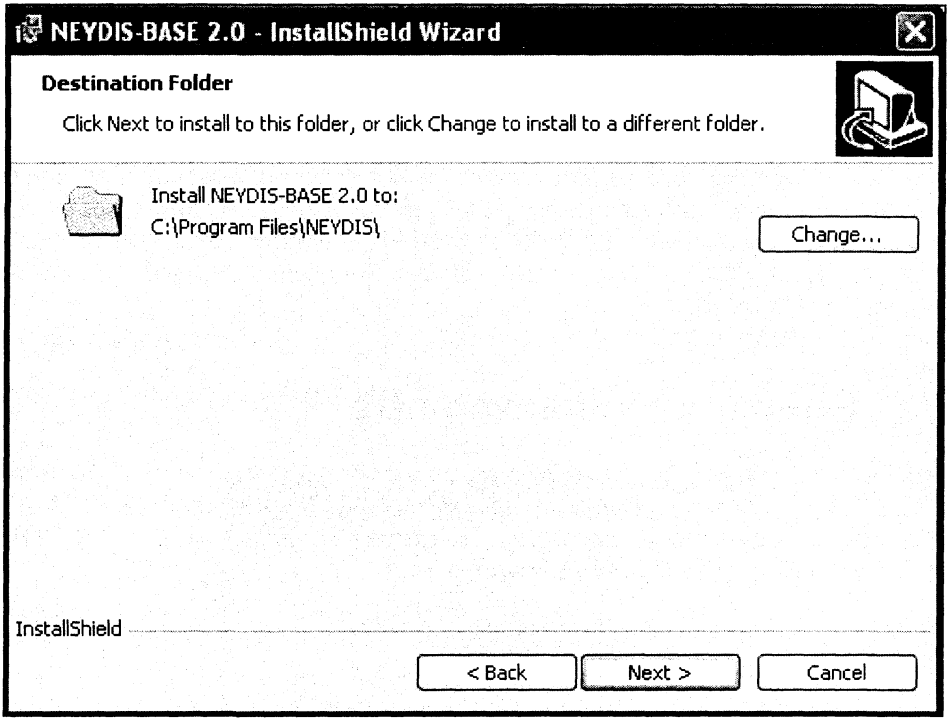


Рис. 17.4. Окно **Destination Folder**

- Окно **Ready to Install the Program**. Это окно показывает всю информацию, введенную пользователем, такую как имя и путь установки. Для продолжения нажмите кнопку **Install**. После окончания установки нажмите кнопку **Finish** — установка завершена.

Программой установки в меню **Пуск \ Программы** будет добавлен пункт **NEYDIS**, открывающий меню с командами запуска оболочки базы знаний и диагностирующей части.

Следует помнить, что данный программный продукт должен быть переписан на ваш компьютер только с помощью программы установки. Не допускается копирование файлов инструментального средства на другой компьютер — программа не будет правильно функционировать. При запуске исполняемых файлов инструментального средства будет произведена проверка того, было ли оно установлено на данном компьютере, и при отрицательном результате программа не будет запущена.

17.6. Создание собственной экспертной системы

Для создания экспертной системы в какой-либо области необходимо создать базу знаний для этой области. Для этого предназначен редактор базы знаний. Как было сказано ранее, редактор баз знаний необходим для создания, редактирования и сохранения в файле базы знаний разрабатываемой ЭС. Файл базы знаний хранит накопленные знания в данной предметной области.

17.7. Описание редактора БЗ

Редактор БЗ является приложением операционных систем Windows 95/98/2000/XP. Исполняемый файл редактора БЗ называется NEYDIS-BASE.EXE. Запустить редактор можно, выбрав пункт **NEYDIS \ NEYDIS-BASE** меню **Пуск \ Программы**.

17.7.1. Главное окно

При запуске NEYDIS-BASE на экране появится окно заставки с информацией о программе и разработчиках. Через четыре секунды окно исчезнет и перед пользователем появится главное окно программы (см. рис. 17.1).

Главное окно редактора БЗ содержит следующие элементы:

- заголовок;
- меню;
- панель инструментов;
- дочерние окна редактирования БЗ.

17.7.2. Система меню

Редактор БЗ является однодокументным приложением. Это означает, что в нем можно одновременно редактировать только одну базу знаний.

Главное меню показано на рис. 17.5 и содержит следующие пункты:

- **Файл**;
- **Сервис**;
- **Помощь**.

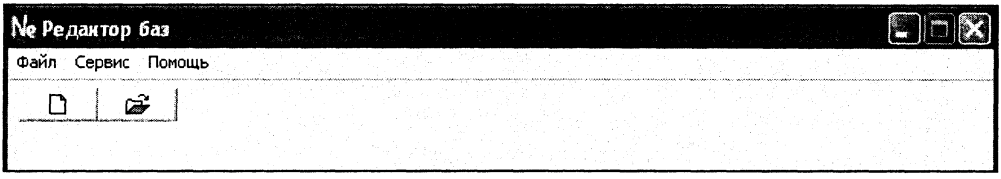


Рис. 17.5. Система меню

Меню *Файл*

Команды меню **Файл** показаны на рис. 17.6.

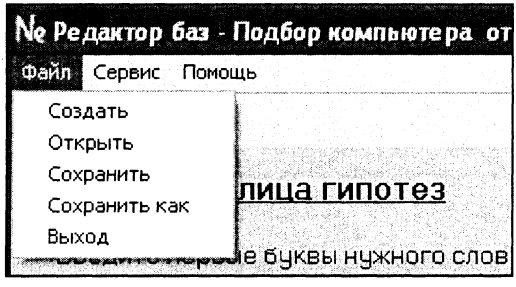


Рис. 17.6. Меню **Файл**

- **Создать** — создание новой базы знаний;
- **Открыть** — загрузка базы знаний из файла;
- **Сохранить** — запись базы знаний на диск;
- **Сохранить как** — запись базы знаний на диск под другим именем;
- **Выход** — выход из редактора БЗ.

Рассмотрим работу с командами меню **Файл** подробнее. После загрузки редактора, если требуется создать новую базу знаний, нужно выбрать команду **Создать**. Откроется окно (рис. 17.7).

Здесь нужно выбрать папку и ввести имя файла, в котором будет храниться база знаний, а затем нажать кнопку **Сохранить**. Тем самым вы начнете работу над новой БЗ.

Если требуется открыть ранее созданную базу, нужно выбрать команду **Открыть** меню **Файл** (рис. 17.8). Откроется окно, в котором нужно выбрать файл с БЗ.

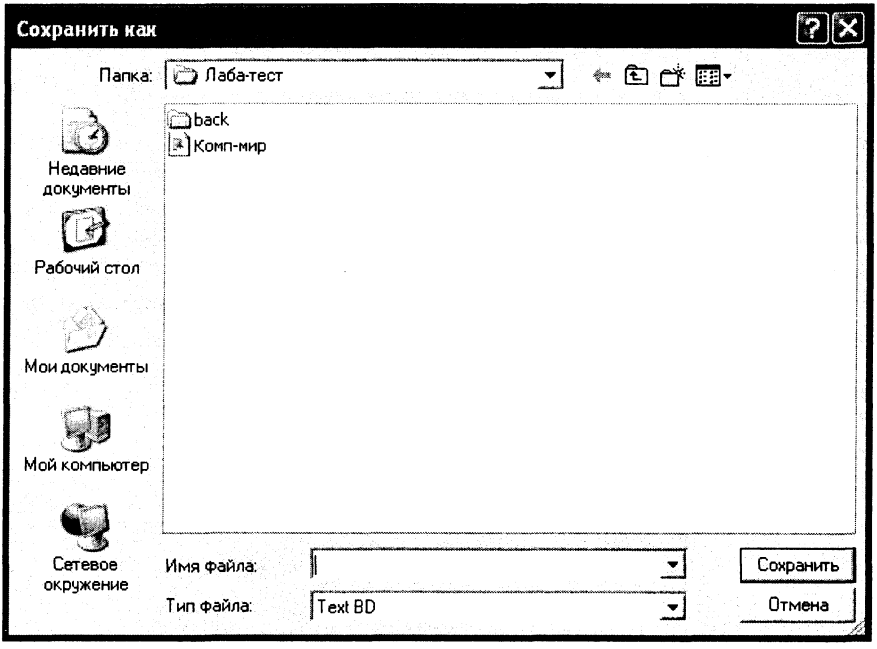


Рис. 17.7. Окно создания новой базы знаний

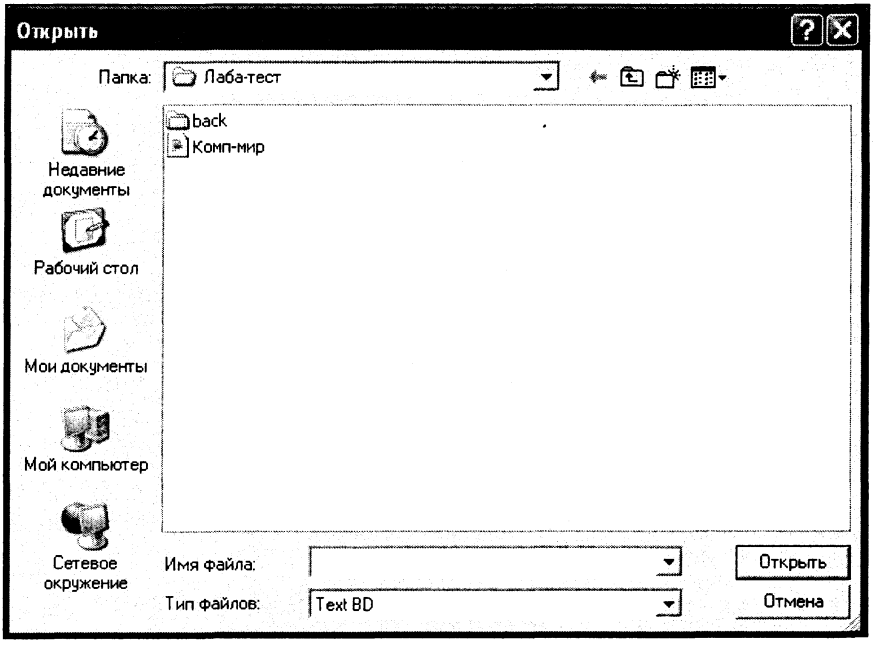


Рис. 17.8. Окно открытия существующей БЗ

Для сохранения БЗ на диске следует выбрать команду **Сохранить** меню **Файл**. БЗ сохраняется на диске без появления каких-либо окон.

Для сохранения БЗ на диске под другим именем следует выбрать команду **Сохранить как** меню **Файл**. Откроется такое же окно, как и при создании новой БЗ (см. рис. 17.7), в котором нужно ввести новое имя файла и нажать кнопку **Сохранить**.

Меню *Сервис*

Меню **Сервис** содержит следующие команды (рис. 17.9):

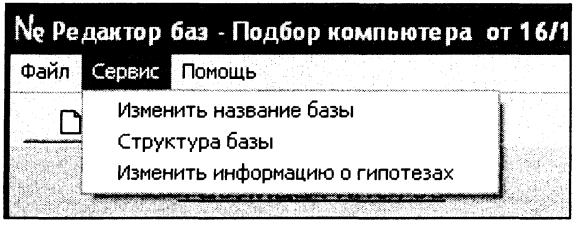


Рис. 17.9. Меню **Сервис**

- **Изменить название базы** — позволяет в любой момент времени изменить название базы (рис. 17.10);

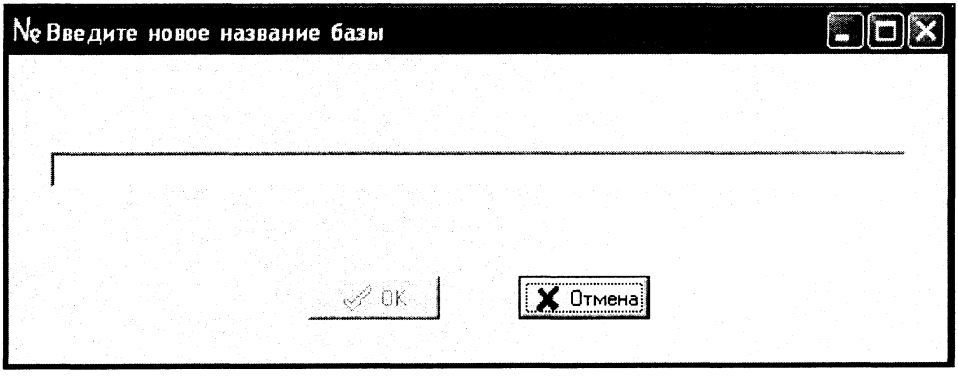


Рис. 17.10. Изменение названия базы знаний

- **Структура базы** — позволяет посмотреть, какие гипотезы используют конкретное свидетельство (рис. 17.11);

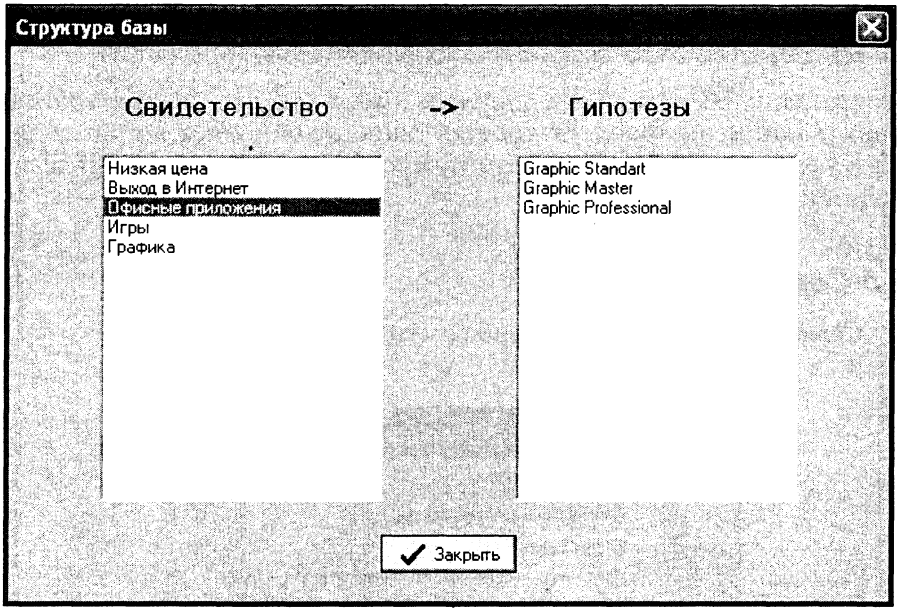


Рис. 17.11. Структура базы знаний

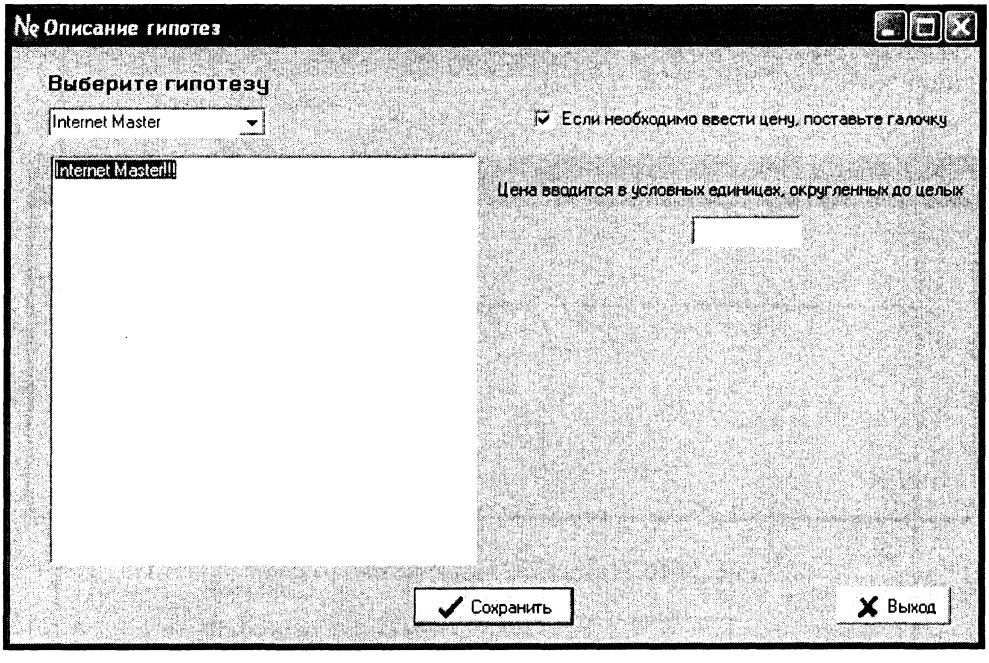


Рис. 17.12. Описание гипотез

- **Изменить информацию о гипотезах** — позволяет добавить к каждой гипотезе ее подробное описание (рис. 17.12). Имеется возможность добавить и изменить цену конкретного товара (гипотезы).

Меню *Помощь*

Меню **Помощь** содержит команду **О программе** (рис. 17.13), открывающую диалоговое окно с информацией о названии программы, разработчиках, номере версии (рис. 17.14).

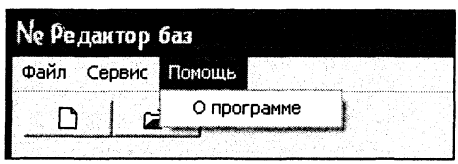


Рис. 17.13. Меню **Помощь**

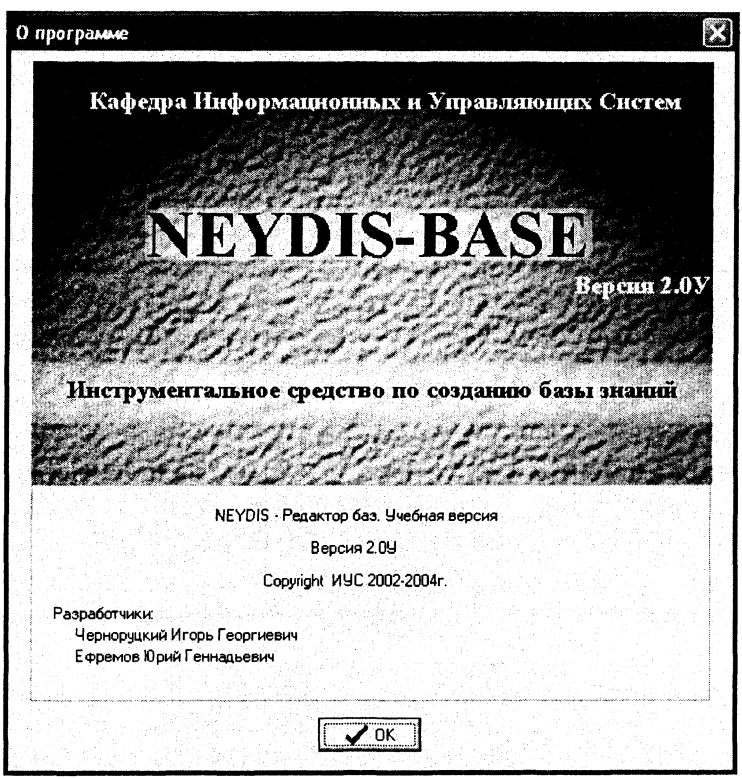


Рис. 17.14. Диалоговое окно **О программе**

Панель инструментов

Панель инструментов (рис. 17.15) содержит две кнопки, нажатие которых аналогично выбору команд **Файл \ Создать** и **Файл \ Открыть** главного меню. При наведении указателя мыши на кнопку панели инструментов появляется всплывающая подсказка с описанием назначения этой кнопки.

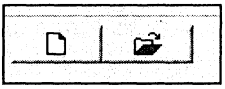


Рис. 17.15. Панель инструментов

17.7.3. Добавление и удаление свидетельств

Переходя непосредственно к созданию базы знаний для экспертной системы в конкретной предметной области, вы выступаете в роли эксперта или инженера по знаниям. На данном этапе для работы с базой знаний ЭС предназначено окно редактирования БЗ (см. рис. 17.1). Оно содержит две главные таблицы (два главных списка):

- таблица гипотез;
- таблица свидетельств.

Например, если речь идет о медицинской экспертной системе, то вариантам решения будут соответствовать различные заболевания (гипотезы), а симптомами, доказывающими заболевание, будут свидетельства, например, высокая температура.

После того как была создана новая или загружена уже имеющаяся БЗ, внешний вид окна редактора баз станет таким, как это показано на рис. 17.1. Список свидетельств отображается в правой части в том порядке, в котором они были добавлены. Каждому свидетельству присваивается уникальный номер, который затем используют гипотезы, поэтому последовательность свидетельств лучше продумать заранее.

Для добавления нового свидетельства следует нажать кнопку **Добавить**, расположенную под списком свидетельств. Откротся окно, показанное на рис. 17.16.

Введите название свидетельства и вопрос, который затем будет задаваться во время диагностики. Вопрос следует сформулировать таким образом, чтобы не возникало двусмысленности, лучше не использовать частицу "не". Например, на вопрос "Не будет ли этот компьютер использоваться в Интернете?" непонятно, как отвечать, — то ли "Да, не будет...", то ли "Нет,

не будет...". Программа вас предупредит, что лучше не использовать частицу "не", если увидит, что вы ее ввели (рис. 17.17).

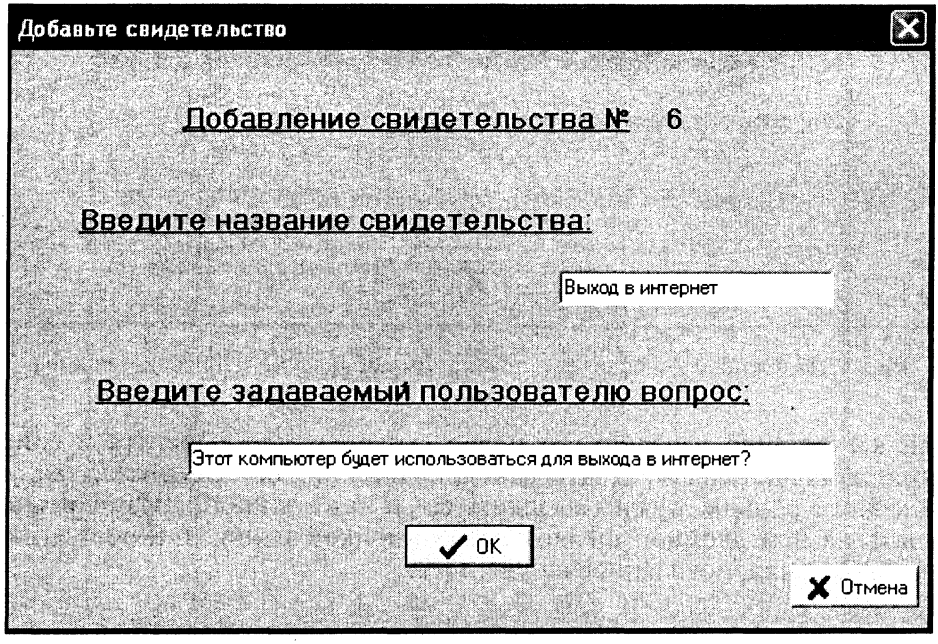


Рис. 17.16. Окно добавления нового свидетельства

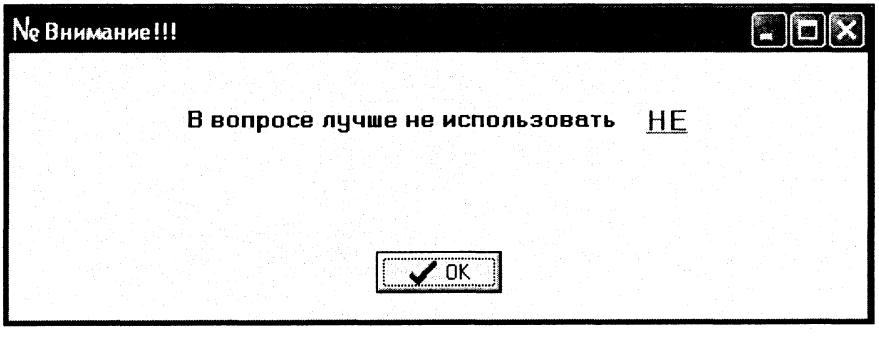


Рис. 17.17. Окно предупреждения

Если добавляемое свидетельство уже присутствует в базе, то откроется соответствующее окно предупреждения (рис. 17.18), и изменений в БЗ не последует.

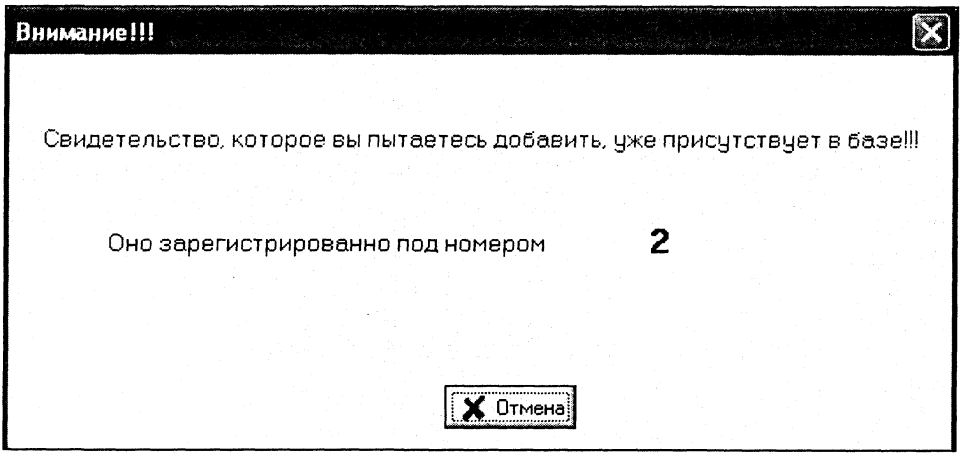


Рис. 17.18. Окно, сообщающее о повторном добавлении в БЗ свидетельства

Если все прошло успешно, то новое свидетельство появится в списке (таблице) свидетельств. Для редактирования любого свидетельства, добавленного раньше, нужно выделить его и нажать кнопку **Изменить** или сделать на нем двойной щелчок левой кнопкой мыши. Откроется окно редактирования, показанное на рис. 17.19.

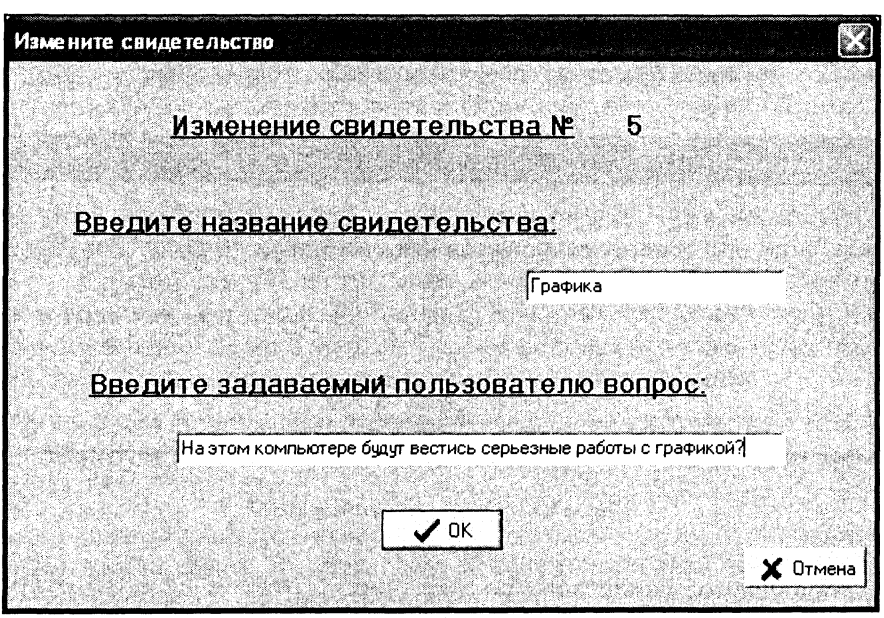


Рис. 17.19. Окно редактирования свидетельства

После всех сделанных изменений нужно нажать кнопку **ОК**. Если это свидетельство уже использовалось хотя бы в одной гипотезе, то появится окно предупреждения (рис. 17.20).

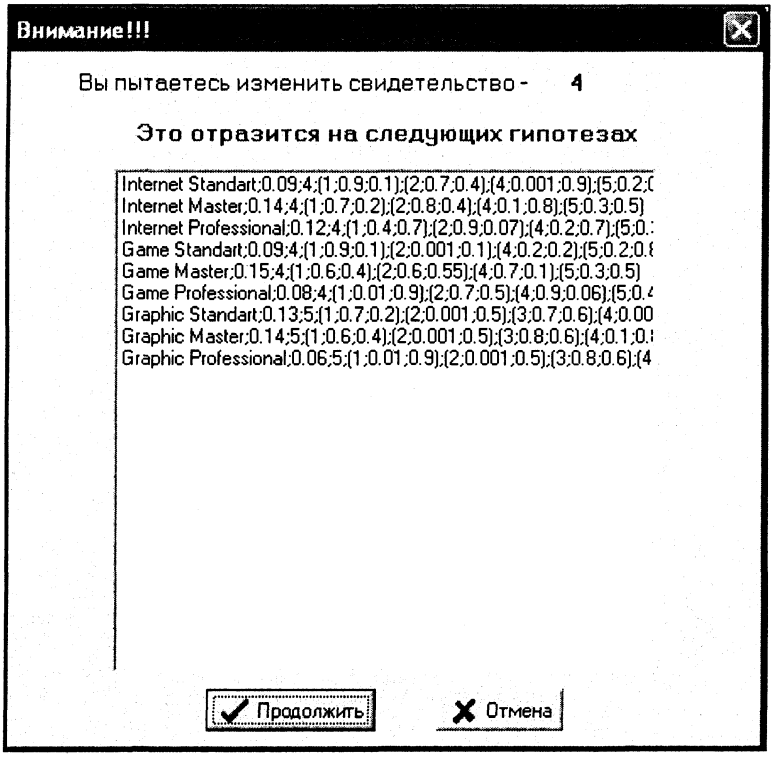


Рис. 17.20. Окно предупреждения при изменении свидетельства

Это окно показывает список гипотез, зависящих от изменяемого свидетельства.

При попытке удалить свидетельство, на которое ссылается хотя бы одна гипотеза, появится окно, показанное на рис. 17.21.

17.7.4. Добавление и удаление гипотез

После добавления всех необходимых свидетельств приступим к добавлению гипотез. Для этого нужно нажать кнопку **Добавить**, расположенную под списком (таблицей) гипотез (см. рис. 17.1). Откротся окно добавления гипотез (рис. 17.22).

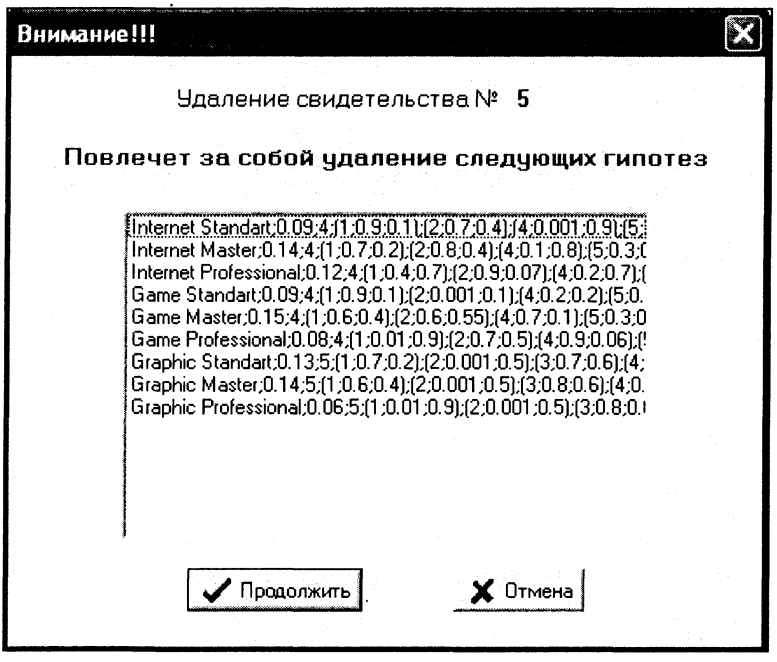


Рис. 17.21. Окно предупреждения при удалении свидетельства

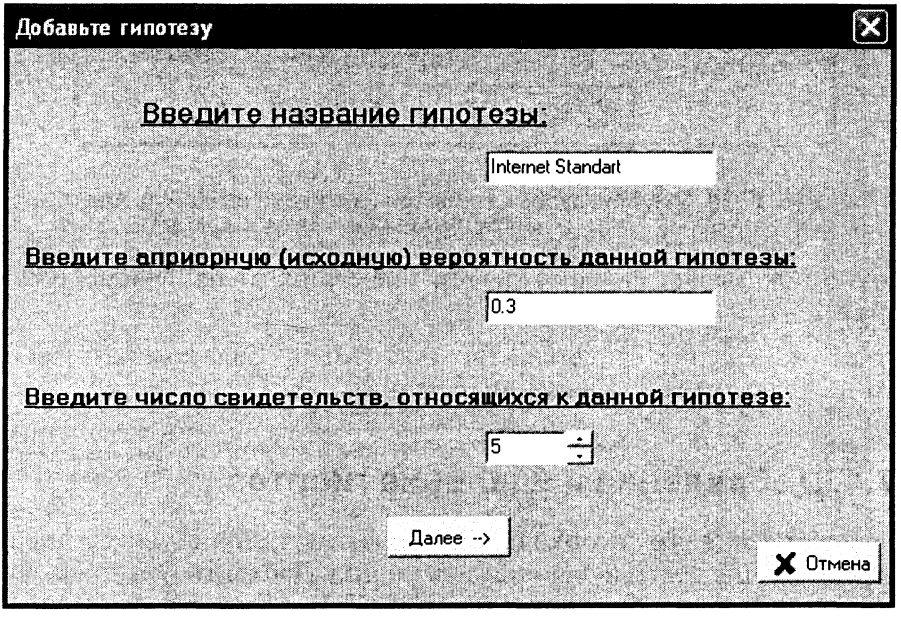


Рис. 17.22. Окно добавления гипотезы

Введите всю необходимую информацию (вводимое число свидетельств, относящихся к данной гипотезе, не может превышать число свидетельств, уже внесенных в БЗ) и нажмите кнопку **Далее**. Если в предыдущем окне была задана верная информация, то откроется следующее окно (рис. 17.23). Если в предыдущем окне была задана неверная информация, то новое окно не откроется — останется открытым предыдущее окно, где необходимо проверить вводимые данные и затем снова нажать кнопку **Далее**.

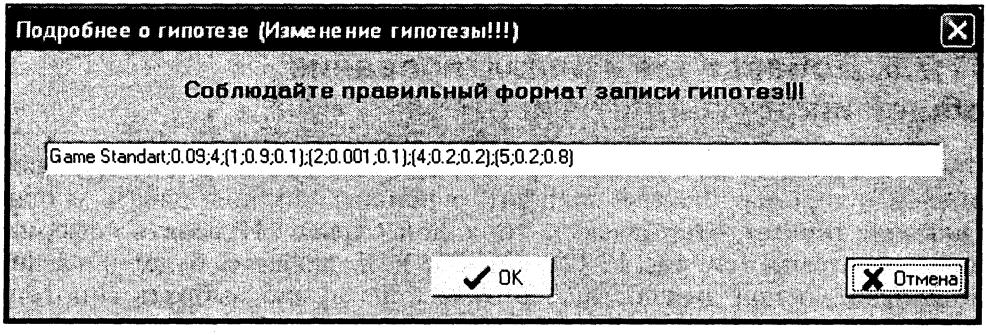


Рис. 17.23. Окно, связывающее гипотезу со свидетельствами

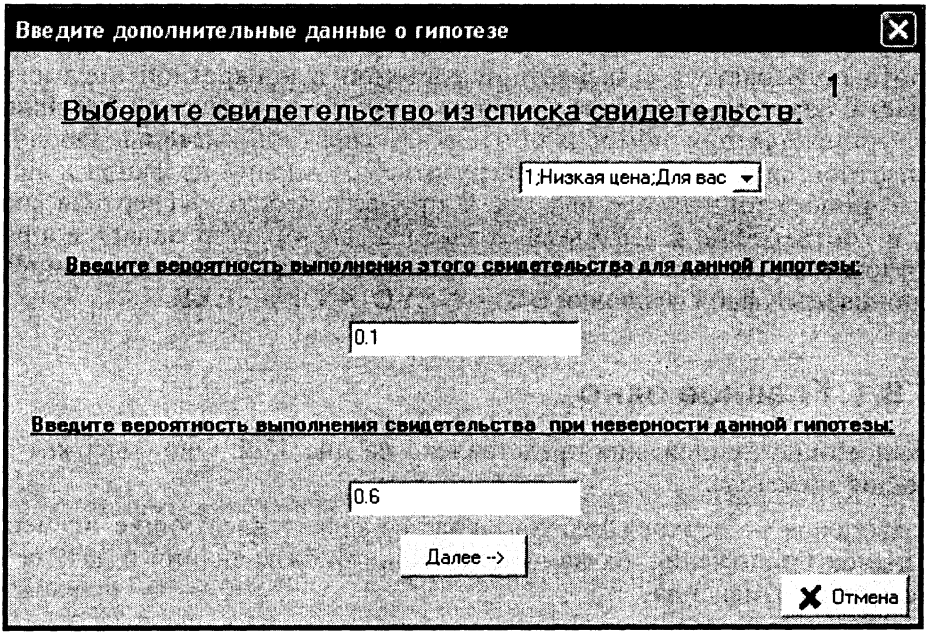


Рис. 17.24. Окно изменения гипотезы

В окне ввода дополнительных данных о гипотезе (см. рис. 17.23) нужно выбрать свидетельство в списке имеющихся и затем ввести все требуемые вероятности, после чего нажать кнопку **Далее**. Когда все связи между гипотезой и свидетельствами будут установлены, новая гипотеза будет добавлена в список гипотез.

Для редактирования гипотезы следует найти ее в списке гипотез и сделать на ней двойной щелчок левой кнопкой мыши. Откроется окно изменения гипотезы (рис. 17.24).

17.7.5. Добавление и редактирование общей информации о гипотезе

Для удобства работы гипотезы проще называть коротким или условным именем, например, Internet Standart, а полное описание давать в окне **Описание гипотез**, открываемом командой **Сервис \ Изменить информацию о гипотезах** (см. рис. 17.12), которое в дальнейшем будет показано пользователю при диагностике. В этом окне нужно выбрать гипотезу, отредактировать ее описание и затем нажать кнопку **Сохранить**.

17.8. Работа готовой экспертной системы

Работа пользователя с экспертной системой в конкретной предметной области, база знаний которой создана с помощью редактора баз знаний, осуществляется при помощи оболочки экспертной системы. Оболочка экспертной системы позволяет загрузить базу знаний из файла и запустить процесс логического вывода. В процессе работы экспертная система, в соответствии с алгоритмом вывода, выбирает и задает вопросы пользователю, получает ответы на них и вычисляет вероятности гипотез. Исполняемый файл оболочки ЭС — NEYDIS_DIAG.EXE.

17.8.1. Главное окно

Главное окно приложения представлено на рис. 17.2. Оно содержит следующие элементы:

- заголовок — верхняя строка окна; содержит следующие элементы: значок приложения, название загруженной базы знаний и дату ее последнего изменения;
- меню (рис. 17.25);
- рабочая область.

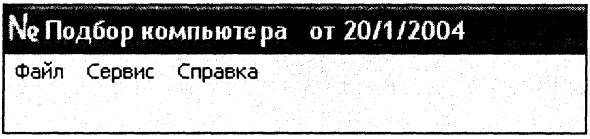


Рис. 17.25. Меню

Рассмотрим меню подробнее:

- меню **Файл** (рис. 17.26) содержит команды:

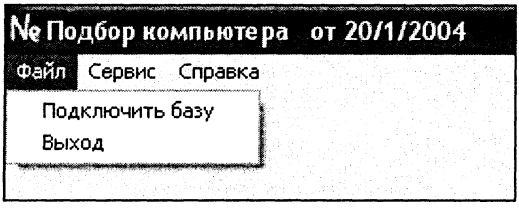


Рис. 17.26. Меню **Файл**

  - **Подключить базу** — позволяет подключить (загрузить) базу знаний из файла;
  - **Выход** — закончить работу с программой;
- меню **Сервис** (рис. 17.27) содержит одну команду:

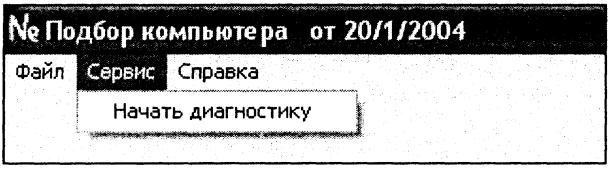


Рис. 17.27. Меню **Сервис**

  - **Начать диагностику** — запускает процесс диагностики с использованием ранее загруженной базы знаний;
- меню **Справка** (рис. 17.28) содержит одну команду:
  - **О программе** — открывает диалоговое окно с информацией о названии программы, разработчиках и номере версии (рис. 17.29).

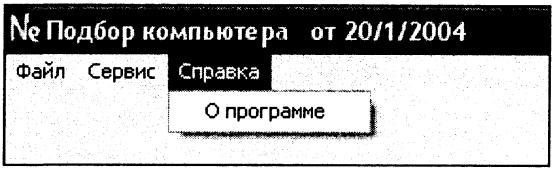


Рис. 17.28. Меню **Справка**

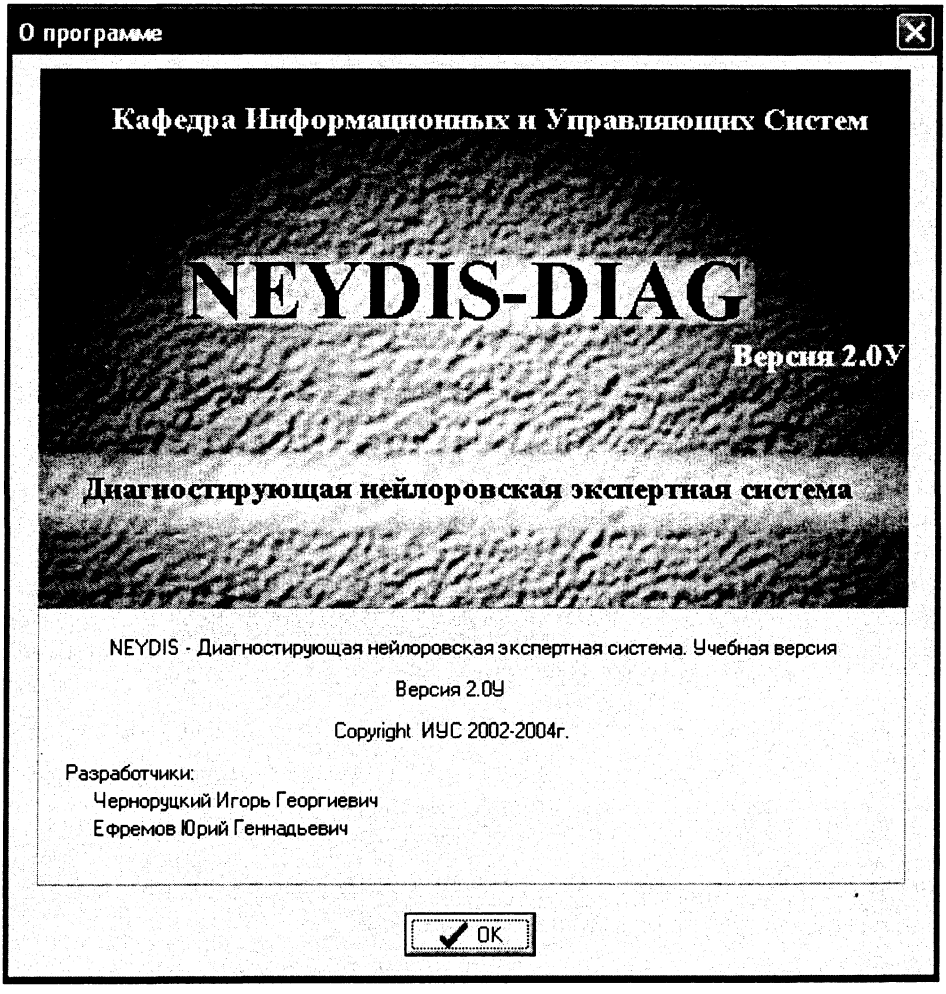


Рис. 17.29. Диалоговое окно **О программе**

17.8.2. Процесс диагностики

После загрузки базы знаний на экране появляется информационное окно (рис. 17.30).

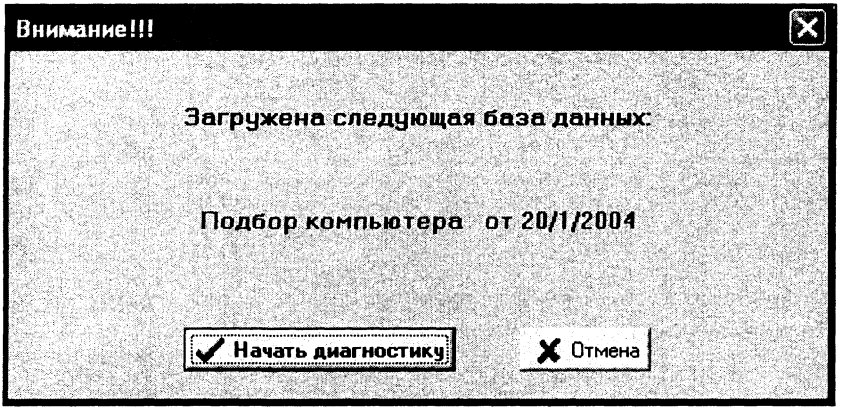


Рис. 17.30. Окно с информацией о загруженной базе знаний

Для начала диагностики следует нажать кнопку **Начать диагностику**. Откроется диалоговое окно (рис. 17.31), которое содержит три основные области:

- *область вопросов* — располагается в верхней части диалогового окна и отображает текущий вопрос, заданный пользователю. Например, на рис. 17.31 показан вопрос: "На этом компьютере будут играть в различные динамические игры, например, Quake3?";
- *область вероятностей* — располагается в правой части окна и отображает текущие вероятности "победы" гипотез. В начальный момент в этой области показаны априорные вероятности гипотез;
- *область ответов* — располагается в левой части окна; содержит ползунковый регулятор, перетаскивая который влево или вправо, пользователь дает ответ, и текстовую интерпретацию ответа. Есть пять вариантов ответов, которые может дать пользователь:
  - **Нет** (рис. 17.32);
  - **Скорее нет, чем да** (рис. 17.33);
  - **Не знаю** (рис. 17.34);
  - **Скорее да, чем нет** (рис. 17.35);
  - **Да** (рис. 17.36).

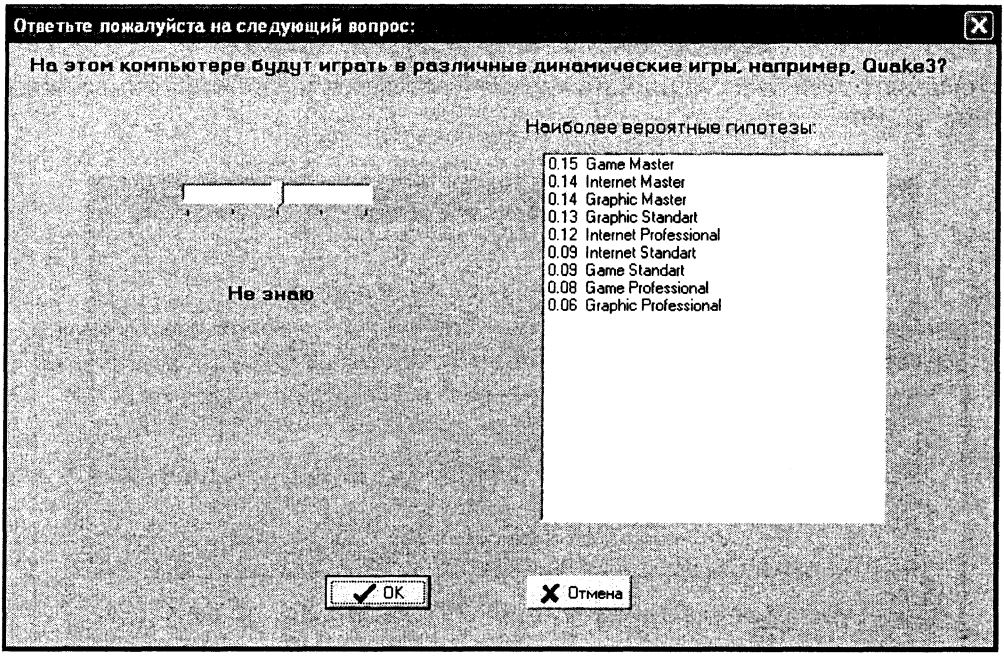


Рис. 17.31. Диалоговое окно диагностики

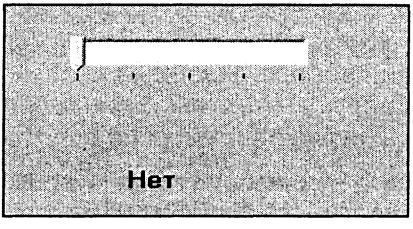


Рис. 17.32. Состояние ползункового регулятора при ответе **Нет**

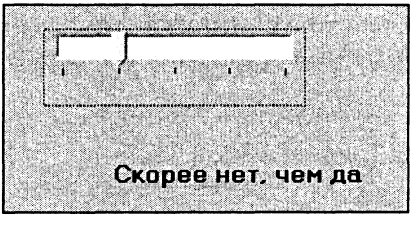


Рис. 17.33. Состояние ползункового регулятора при ответе **Скорее нет, чем да**

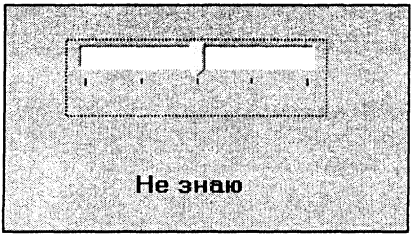


Рис. 17.34. Состояние ползункового регулятора при ответе **Не знаю**

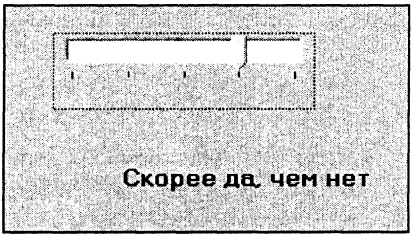


Рис. 17.35. Состояние ползункового регулятора при ответе **Скорее да, чем нет**

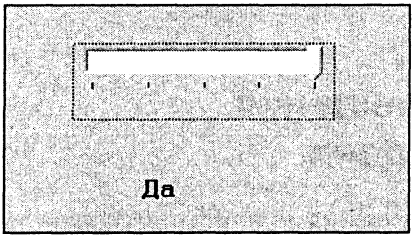


Рис. 17.36. Состояние ползункового регулятора при ответе **Да**

После того как пользователь ответит на все необходимые для принятия решения вопросы, система завершит работу, открыв окно с заключительным результатом (рис. 17.37).

Вся проделанная ЭС работа записывается в файл с именем LOG.LOG, который можно найти в папке БЗ. Открыть его можно в любом текстовом редакторе, например, в Блокноте. В этом файле хранятся все вопросы, заданные пользователю ЭС, все ответы пользователя, а также реакция системы в виде вероятностей гипотез в каждый момент времени.

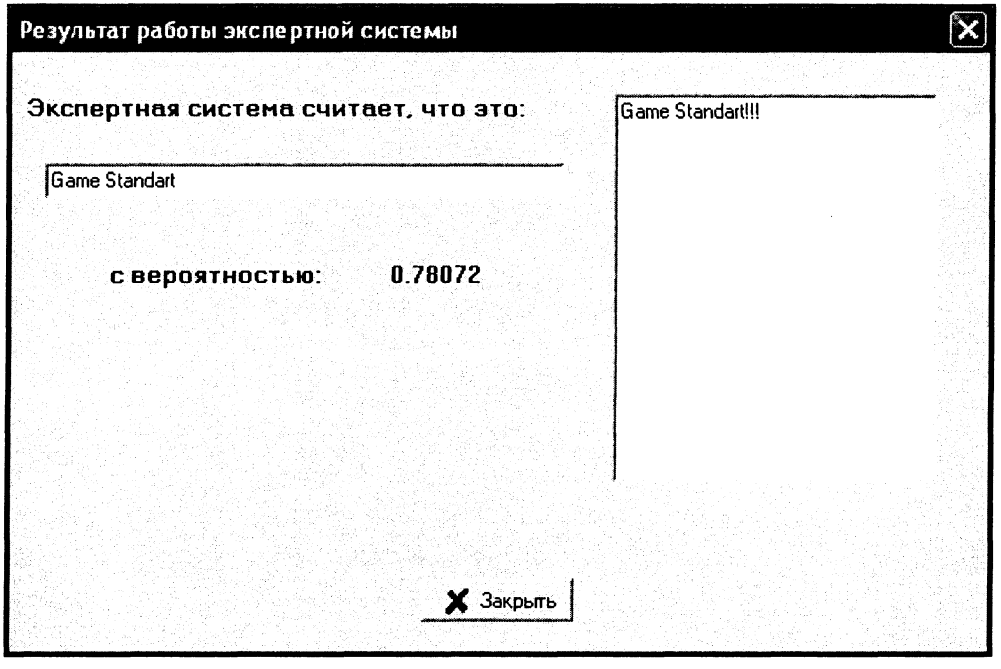


Рис. 17.37. Информационное окно с результатом

17.9. Пример решения модельной задачи

Рассмотрим пример решения конкретной задачи — создание нейлоровской диагностирующей экспертной системы для подбора компьютера с помощью инструментального средства NEYDIS. Необходимо отметить, что заданная экспертная система будет очень простой, с небольшой по размерам БЗ, однако этот пример позволит изучить различные стадии процесса создания экспертных систем с помощью данного средства.

Для создания экспертной системы по подбору компьютера при покупке его пользователем, в зависимости от потребностей, используем прайслист компании "Компьютерный мир" от 26.02.2003. Из него возьмем всю техническую и ценовую информацию:

Internet Standart. Celeron1200/ MB Gigabyte-6OXTA/128 - DIMM/ 20Gb Samsung 7200/FDD/ATI Radeon 7000 32Mb DDR/ CD-ROM/Modem Genius int PCI /sound - 339$

Internet Master. Pentium4 1700/ MB Microstar 6566Ultra/256 DDR2100 / 40Gb WD 7200/FDD/ATI Radeon 7500 64Mb DDR/CD-ROM/Modem USR int Hardware /sound – 579$

Internet Professional. Pentium4 2000/ MB Microstar 6566Max/256 DDR2100/ 80Gb Seagate 7200/FDD/ATI Radeon 9000 64Mb DDR/ CD-ROM/ Modem USR Courier ext /sound – 800$

Game Standart. Celeron 1700/MB Microstar 6547 Max-U/ 128 DDR2100/ 20Gb Seagate7200/FDD/Geforce2MX400 32Mb DDR/CD-ROM/sound - 390$

Game Master. Pentium4 1800/MB Microstar 6566Max/256 DDR2100/ 40Gb Samsung 7200/FDD/ GeForce4MX440 64 DDR/ CD-ROM/ SB Live!4.1/ Modem Genius int – 616$

Game Professional. Pentium4 2400/ MB Intel D850EMV2/RIMM 512 PC-800/ 80Gb Seagate 7200/FDD/GeForce4 Ti4400 128Mb Deluxe/CD-ROM / SB Live! 5.1/ Modem Zyxel int PCI – 1158$

Graphic Standart. Pentium4 1800/ MB Microstar 6566Ultra/ 256 DDR2100/ 40Gb WD/ FDD/ ATI Radeon 7000 64 Mb DDR/ CD-ROM/sound – 531$

Graphic Master. Pentium4 2000/MB Gigabyte 8PE667 / 512Mb DDR2100/ 60Gb Samsung 7200 /FDD/ Matrox G450 32Mb/ CD-ROM/ sound – 675$

Graphic Professional. Pentium4 2800/MB Intel KD845PEBT2/ 512Mb DDR2700/ 80Gb Seagate 7200/ FDD/ Matrox Parhelia 128 Mb DDR/ CD-RW 48x12x48 Teac/ sound – 1390$

- Этап 1 — выделим возможные готовые конфигурации (гипотезы):

  Internet Standart

  Internet Master

  Internet Professional

  Game Standart

  Game Master

  Game Professional

  Graphic Standart

  Graphic Master

  Graphic Professional

- Этап 2 — определим критерии (свидетельства), которые помогут выбрать ту или иную конфигурацию, как оптимальную:

  Низкая цена

  Выход в Интернет

  Офисные приложения

  Игры

  Графика

- 3 этап — придумаем к каждому свидетельству вопрос, который затем будет задаваться пользователю:

  Низкая цена - "Для вас очень важна низкая стоимость компьютера?"

  Выход в Интернет - "Этот компьютер будет использоваться для выхода в Интернет?"

  Офисные приложения - "На этом компьютере будут работать с офисными приложениями?"

  Игры - "На этом компьютере будут играть в различные динамические игры, например, Quake3?"

  Графика - "На этом компьютере будут вестись серьезные работы с графикой?"

- 4 этап — определим связи (зависимости) между гипотезами и свидетельствами, укажем значения условных вероятностей. Например, ясно, что если пользователю требуется самый мощный игровой компьютер, то его цена с максимальным значением вероятности будет одной из самых высоких (или с минимальным значением вероятности его цена будет одной из самых низких). Иначе говоря, если гипотеза "Game Professional" верна, то свидетельство "Низкая цена" имеет место с вероятностью 0,01. При этом если гипотеза "Game Professional" ложна, то свидетельство "Низкая цена" может иметь место с вероятностью 0,9, т. к. почти все остальные конфигурации стоят дешевле. Итак, мы установили связь между гипотезой "Game Professional" и свидетельством "Низкая цена". Зададим априорную (исходную) вероятность гипотезы "Game Professional" — 0,08; эта вероятность определяется как количество людей, которые купят этот компьютер (в процентном соотношении), если не получить от них никакой дополнительной информации. Аналогично задаются все остальные связи и априорные вероятности.

  Получаем набор строк следующего вида:

  Internet Standart;0.09;4;(1;0.9;0.1);(2;0.7;0.4);(4;0.001;0.9);(5;0.2;0.9)

  Internet Master;0.14;4;(1;0.7;0.2);(2;0.8;0.4);(4;0.1;0.8);(5;0.3;0.5)

  Internet Professional;0.12;4;(1;0.4;0.7);(2;0.9;0.07);(4;0.2;0.7);(5;0.3;0.4)

  Game Standart;0.09;4;(1;0.9;0.1);(2;0.001;0.1);(4;0.2;0.2);(5;0.2;0.8)

  Game Master;0.15;4;(1;0.6;0.4);(2;0.6;0.55);(4;0.7;0.1);(5;0.3;0.5)

  Game Professional;0.08;4;(1;0.01;0.9);(2;0.7;0.5);(4;0.9;0.06);(5;0.4;0.3)

  Graphic Standart;0.13;5;(1;0.7;0.2);(2;0.001;0.5);(3;0.7;0.6);(4;0.001;0.9);(5;0.2;0.9)

  Graphic Master;0.14;5;(1;0.6;0.4);(2;0.001;0.5);(3;0.8;0.6);(4;0.1;0.8);(5;0.6;0.1)

  Graphic Professional;0.06;5;(1;0.01;0.9);(2;0.001;0.5);(3;0.8;0.6);(4;0.5;0.2);(5;0.9;0.07)

- 5 этап — создание базы знаний при помощи редактора базы знаний.
  - Внесем все свидетельства и вопросы, в том порядке, как это задано на этапе 3.
  - Добавим гипотезы и вероятности.
- 6 этап — проверка корректности работы ЭС. Необходимо запустить оболочку ЭС, загрузить созданный с помощью редактора БЗ файл базы знаний и начать тестирование.

Тест 1

Предположим, пользователю требуется средний по цене компьютер, с возможностью доступа в Интернет, для нечастых игр, для работы с графикой на невысоком уровне. Я посоветовал бы этому пользователю конфигурацию "Game Master", хотя не очень уверенно, т. к. пользователь не просит ничего конкретного, все его пожелания усредненные и расплывчатые. Посмотрим, что ему посоветует система.

Для этого проведем тестирование, ответив на вопросы, с учетом пожеланий пользователя и проверим файл с отчетом LOG.LOG:

Диалог экспертной системы с пользователем:

1. Вопрос ЭС:

   На этом компьютере будут играть в различные динамические игры, например, Quake3?

   Ответ пользователя:

   Скорее да, чем нет

   Реакция ЭС:

   0.35 Game Master
   0.32 Game Professional
   0.10 Graphic Professional
   0.09 Game Standart
   0.08 Internet Professional
   0.08 Internet Master
   0.08 Graphic Master
   0.07 Graphic Standart
   0.05 Internet Standart

2. Вопрос ЭС:

Для вас очень важна низкая стоимость компьютера?

Ответ пользователя:

Скорее да, чем нет

Реакция ЭС:

0.40 Game Master
0.28 Game Standart
0.17 Internet Standart
0.16 Internet Master
0.16 Game Professional
0.13 Graphic Standart
0.10 Graphic Master
0.06 Internet Professional
0.05 Graphic Professional

3. Вопрос ЭС:

На этом компьютере будут вестись серьезные работы с графикой?

Ответ пользователя:

Скорее да, чем нет

Реакция ЭС:

0.34 Game Master
0.25 Graphic Master
0.23 Graphic Professional
0.19 Game Professional
0.18 Game Standart
0.13 Internet Master
0.11 Internet Standart
0.08 Graphic Standart
0.06 Internet Professional

4. Вопрос ЭС:

Этот компьютер будет использоваться для выхода в Интернет?

Ответ пользователя:

Скорее да, чем нет

Реакция ЭС:

0.35 Game Master
0.24 Internet Professional
0.21 Game Professional
0.18 Internet Master
0.14 Internet Standart
0.12 Graphic Master
0.11 Graphic Professional
0.09 Game Standart
0.04 Graphic Standart

5. Результат работы ЭС:

Game Master с вероятностью: 0.35258

Система тоже рекомендует конфигурацию "Game Master" — наши мнения совпали, что доказывает корректность работы, а низкая вероятность подтверждает то, что ответы пользователя очень "усредненные".

Тест 2

Нам требуется игровой компьютер, сумма не ограничена. Естественно, я считаю, что таким требованиям соответствует только конфигурация "Game Professional". Посмотрим, что скажет система.

Диалог экспертной системы с пользователем:

1. Вопрос ЭС:

На этом компьютере будут играть в различные динамические игры, например, Quake3?

Ответ пользователя:

Да

Реакция ЭС:

0.57 Game Professional
0.55 Game Master

0.14 Graphic Professional
0.09 Game Standart
0.04 Internet Professional
0.02 Internet Master
0.02 Graphic Master
0.00 Internet Standart
0.00 Graphic Standart

2. Вопрос ЭС:

Для вас очень важна низкая стоимость компьютера?

Ответ пользователя:

Нет

Реакция ЭС:

0.93 Game Professional
0.61 Graphic Professional
0.45 Game Master
0.07 Internet Professional
0.01 Internet Master
0.01 Graphic Master
0.01 Game Standart
0.00 Internet Standart
0.00 Graphic Standart

3. Вопрос ЭС:

Этот компьютер будет использоваться для выхода в Интернет?

Ответ пользователя:

Да

Реакция ЭС:

0.95 Game Professional
0.50 Internet Professional
0.47 Game Master
0.02 Internet Master

0.00 Internet Standart
0.00 Graphic Standart
0.00 Graphic Professional
0.00 Graphic Master
0.00 Game Standart

4. Результат работы ЭС:

Game Professional с вероятностью: 0.94758

С большой уверенностью система рекомендует пользователю вариант "Game Professional", что очень логично и полностью совпадает с моим мнением.

17.10. Заключение

Данный программный продукт является инструментальным средством создания нейлоровских экспертных систем. Он может быть применен для создания экспертных систем в любых областях человеческой деятельности, где возникает ситуация наличия ненадежных данных. Для работы с ненадежными данными был реализован подход, основанный на принципах, предложенных К. Нейлором. Система не требует от пользователя специальных знаний в области искусственного интеллекта и теории экспертных систем. Все необходимые ему сведения содержатся в пользовательской документации, поставляемой вместе с программным средством. Система имеет удобный пользовательский интерфейс, систему установки (инсталляции) на ПК пользователя, позволяет создавать и просматривать отчет о работе экспертной системы.

Приложение

Основные обозначения и терминологические замечания

Обозначение	Значение
R	Множество вещественных (действительных) чисел
R^n	n-мерное евклидово пространство
$\forall$, $\exists$	Кванторы всеобщности и существования
$f: X \to Y$	Отображение f множества X во множество Y
$k \in [1:N]$	Число k принимает последовательно все значения из множества натуральных чисел от 1 до N включительно
$\{x^n\}$	Последовательность элементов x^n
$(a,b), \langle a,b \rangle$	Скалярное произведение векторов a и b
$A > 0$	Матрица A положительно определена
$\triangleq$	Равно по определению
$D = \{x \in H / P(x)\}$	Подмножество элементов множества H, обладающих свойством $P(x)$
$\varnothing$	Пустое множество
$C^k(D)$	Множество k раз непрерывно дифференцируемых на множестве D функций
$(\cdot)^+$	Псевдообратная матрица
МНК	Метод наименьших квадратов
$\mathrm{diag}\lambda_i$, $\mathrm{diag}(\lambda_i)$	Диагональная матрица
$\mathrm{cond}[A]$	Спектральное число обусловленности матрицы A

(окончание)

Обозначение	Значение
J', J''	Вектор градиента и матрица вторых производных функционала $J(x)$
$\operatorname{argmin} J(x)$	Минимизатор функционала $J(x)$
$\operatorname{Argmin} J(x)$	Множество всех минимизаторов функционала $J(x)$

Сделаем терминологические замечания по тексту учебника.

Согласно общепринятым математическим канонам термин *функционал* означает (однозначное) отображение произвольного множества во множество вещественных чисел. Следовательно, вещественная функция от вещественного переменного, а также вещественная функция от нескольких вещественных переменных также являются функционалами.

Под задачами *математического программирования* будем понимать конечномерные задачи оптимизации, т. е. задачи поиска максимума или минимума функционала, определенного на некотором подмножестве *конечномерного* евклидова пространства. В этом случае, например, задачи теории оптимального управления, формулируемые как задачи поиска оптимальных управляющих функций из некоторого подмножества бесконечномерного пространства, уже не будут относиться нами к задачам математического программирования.

Мы будем следовать сложившейся в математике традиции и называть определитель матрицы Гессе *гессианом*. В некоторых книгах и учебниках по математическому программированию и оптимизации сама матрица Гессе называется гессианом, что на наш взгляд является неоправданным.

Список литературы

1. Айзерман М. А., Алескеров Ф. Т. Выбор вариантов: основы теории. — М.: Наука, 1990.
2. Алексеев А. В., Борисов А. Н., Вилюмс Э. Р., Слядзь Н. Н., Фомин С. А. Интеллектуальные системы принятия проектных решений. — Рига: Зинатне, 1997.
3. Арсеньев Ю. Н., Шелобаев С. И., Давыдова Т. Ю. Принятие решений. Интегрированные интеллектуальные системы: Учеб. пособие для вузов. — М.: ЮНИТИ-ДАНА, 2003.
4. Березовский Б. А., Барышников Ю. М., Борзенко В. И., Кемпнер Л. М. Многокритериальная оптимизация: Математические аспекты. — М.: Наука, 1989.
5. Борисов А. Н., Вилюмс Э. Р., Сукур Л. Я. Диалоговые системы принятия решений на базе МИНИ-ЭВМ: Информационное, математическое и программное обеспечение. — Рига: Зинатне, 1986.
6. Брукинг А., Джонс П., Кокс Ф. и др. Экспертные системы. Принципы работы и примеры. — М.: Радио и связь, 1987.
7. Варфоломеев В. И., Воробьев С. Н. Принятие управленческих решений: Учеб. пособие для вузов. — М.: КУДИЦ-ОБРАЗ, 2001.
8. Вентцель Е. С. Исследование операций. — М.: Советское радио, 1972.
9. Волков И. К., Загоруйко Е. А. Исследование операций: Учебник для вузов. — М.: Изд-во МГТУ им. Н. Э. Баумана, 2000.
10. Вязгин В. А., Федоров В. В. Математические методы автоматизированного проектирования: Учеб. пособие для втузов. — М.: Высш. шк., 1989.
11. Гаврилова Т. А., Хорошевский В. Ф. Базы знаний интеллектуальных систем. — СПб.: Питер, 2000.

12. Глухов В. В., Медников М. Д., Коробко С. Б. Математические методы и модели для менеджмента. — СПб.: Лань, 2000.

13. Джексон П. Введение в экспертные системы: Пер. с англ.: Учеб. пособие. — М.: Вильямс, 2001.

14. Дубов Ю. А., Травкин С. И., Якимец В. Н. Многокритериальные модели формирования и выбора вариантов систем. — М.: Наука, 1986.

15. Дэннис Дж., Шнабель Р. Численные методы безусловной оптимизации и решения нелинейных уравнений. — М.: Мир, 1988.

16. Змитрович А. И. Интеллектуальные информационные системы. — Минск: ТетраСистемс, 1997.

17. Кини Р. Л., Райфа Х. Принятие решений при многих критериях: предпочтения и замещения/Под ред. И. Ф. Шахнова. — М.: Радио и связь, 1981.

18. Ларичев О. И. Теория и методы принятия решений, а также Хроника событий в Волшебных странах: Учебник. — М.: Логос, 2003.

19. Льюс Р. Д., Райфа Х. Игры и решения. — М.: Изд-во иностр. лит-ры, 1961.

20. Малыхин В. И. Финансовая математика: Учеб. пособие для вузов. — М.: ЮНИТИ-ДАНА, 2000.

21. Миркин Б. Г. Проблема группового выбора. — М.: Наука, 1974.

22. Моисеев Н. Н. Математические задачи системного анализа. — М.: Наука, 1981.

23. Нейлор К. Как построить свою экспертную систему. — М.: Энергоатомиздат, 1991.

24. Ногин В. Д., Чистяков С. В. Применение линейной алгебры в принятии решений: Учеб. пособие. — СПб.: Изд-во СПбГТУ, 1998.

25. Ногин В. Д. Принятие решений в многокритериальной среде. — М.: Физматлит, 2002.

26. Перегудов Ф. И., Тарасенко Ф. П. Введение в системный анализ. — М.: Высш. шк., 1989.

27. Подиновский В. В. Многокритериальные задачи с упорядоченными по важности однородными критериями//Автоматика и Телемеханика, 1976. — № 11. — С. 118—127.

28. Подиновский В. В., Ногин В. Д. Парето-оптимальные решения многокритериальных задач. — М.: Наука, 1982.

29. Попов Э. В. Экспертные системы: Решение неформализованных задач в диалоге с ЭВМ. — М.: Наука, 1987.

30. Поспелов Г. С. Искусственный интеллект — основа новой информационной технологии. — М.: Наука, 1988.

31. Построение экспертных систем: Пер. с англ. / Под ред. Ф. Хейеса-Рота, Д. Уотермана, Д. Лената. — М.: Мир, 1987.

32. Ракитский Ю. В., Устинов С. М., Черноруцкий И. Г. Численные методы решения жестких систем. — М.: Наука, 1979.

33. Растригин Л. А. Современные принципы управления сложными объектами. — М.: Сов. радио, 1980.

34. Розен В. В. Цель — оптимальность — решение. — М.: Радио и связь, 1982.

35. Табак Д., Куо Б. Оптимальное управление и математическое программирование. — М.: Наука, 1975.

36. Таха Х. Введение в исследование операций: В 2-х книгах. — М.: Мир, 1985.

37. Уотшем Т. Дж., Паррамоу К. Количественные методы в финансах: Учеб. пособие для вузов. — М.: Финансы, ЮНИТИ, 1999.

38. Федоренко Р. П. Приближенное решение задач оптимального управления. — М.: Наука, 1978.

39. Химмельблау Д. Прикладное нелинейное программирование. — М.: Мир, 1975.

40. Хованов Н. В. Анализ и синтез показателей при информационном дефиците. — СПб.: Изд-во СПбГУ, 1996.

41. Черноруцкий И. Г. Оптимальный параметрический синтез: электротехнические устройства и системы. — Л.: Энергоатомиздат, 1987.

42. Черноруцкий И. Г. Методы принятия решений: Учеб. пособие. — Л.: Изд-во ЛПИ, 1990.

43. Черноруцкий И. Г. Методы оптимизации: Учеб. пособие. — СПб., Изд-во СПбГТУ, 1998.

44. Черноруцкий И. Г. Методы оптимизации и принятия решений. — СПб.: Лань, 2001.

45. Черноруцкий И. Г. Методы оптимизации в теории управления. — СПб.: Питер, 2004.

46. Шрейдер Ю. А. Равенство, сходство, порядок. — М.: Наука, 1971

47. Брукинг А., Джонс П., Кокс Ф. и др. Экспертные системы. Принципы работы и примеры / Под ред. Р. Форсайта. — М.: Радио и связь, 1987.

Предметный указатель

Д

Е, Ж

З

И

К

Л

М

Н

О

П

Р

С

T

У

Ф

Х

Ц

Ч

Ш

Э

Я