

краткий курс

МАТЕМАТИЧЕСКИЕ МЕТОДЫ ИССЛЕДОВАНИЯ ОПЕРАЦИЙ В ЭКОНОМИКЕ

П. Конюховский

УЧЕБНОЕ ПОСОБИЕ

**Санкт-Петербург**

**Москва • Харьков • Минск**

**2000**

**Конюховский П. В.**

Математические методы исследования операций в экономике

*Серия «Краткий курс»*

|  |  |
| --- | --- |
| Главный редактор издательства  Заведующий редакцией  Художественный редактор  Литературный редактор  Верстка | *В. Усманов*  *Л. Волкова*  *В. Земских*  *Е. Маслова*  *Е. Маслова* |

ББК22.183я7+65.529 УДК519.8(075)+658.012.122(075)

**Конюховский П. В.**

К65 Математические методы исследования операций в экономике — СПб.: Издательство «Питер», 2000. — 208 с. — (Серия «Краткий курс»).

ISBN 5-8046-0190-3

В настоящем учебном пособии представлены основные разделы исследо­вания операций. Упор делается на изложении теоретических и практических аспектов алгоритмов решения экстремальных задач, которые формулируются на базе известных экономико-математических моделей. Отдельное внимание уделяется вопросам содержательной экономической интерпретации формаль­ных математических понятий.

Серия книг «Краткий курс» предназначена для студентов экономических и управленческих специальностей всех форм обучения, а также для всех инте­ресующихся соответствующей темой.

© Конюховский П. В., 2000

© Серия, оформление, ЗАО «Издательство «Питер», 2000

Все права защищены. Никакая часть данной книги не может быть воспроизведена в какой бы то ни было форме без письменного разрешения владельцев авторских прав.

ISBN 5-8046-0190-3

Издательство «Питер». 196105, С.-Петербург, Благодатная ул., 67. Лицензия ЛР № 066333 от 23.02.99.

Подписано к печати 15.09.99. Формат 60х90/16. Усл. п. л. 13.

Тираж 5000. Заказ 4418.

Отпечатано с фотоформ в АООТ «Типография „Правда”».

119. С.-Петербург, Социалистическая ул., 14.

## ПРЕДИСЛОВИЕ

Последние годы ознаменовались выходом большого количества книг, посвященных тем или иным разделам экономической науки. Среди них ведущее место, как по количеству наименова­ний так и по тиражу, занимают издания, посвященные финан­сам банковскому делу, теоретическим и практическим вопро­сам управления ценными бумагами. В значительной степени сложившаяся ситуация объясняется острым дефицитом подоб­ной литературы в предыдущие десятилетия. Однако обратной стороной этого несомненно позитивного процесса стало воз­никновение определенного дидактического вакуума вокруг дру­гих экономических тем. Стремление восполнить один из таких пробелов послужило стимулом для выпуска настоящей книги, посвященной основам исследования операции.

Формирование исследования операций как самостоятель­ной ветви прикладной математики относится к периоду 40-х и 50-х годов. Последующие полтора десятилетия были отмечены широким применением полученных фундаментальных теорети­ческих результатов к разнообразным практическим задачам и связанным с этим переосмыслением потенциальных возможно­стей теории. В результате исследование операции приобрело черты классической научной дисциплины, без которой немыс­лимо базовое экономическое образование.

Обращаясь к задачам и проблемам, составляющим предмет исследования операций, нельзя не вспомнить о вкладе, внесен­ном в их решение представителями отечественной научной школы, среди которых в первую очередь должен быть назван Л. В. Канторович, ставший в 1975 г. лауреатом Нобелевской премии за свои работы по оптимальному использованию ресур­сов в экономике.

Предлагаемая вашему вниманию книга задумана как учебное пособие для специалистов в области экономики и управления предприятиями, и призвано создать общее представление о типах задач, изучаемых в исследовании операции, методах их решения и принципиальных идеях, лежащих в основе этих методов. Математические аспекты предмета по отношению к постав­ленным перед настоящим изданием целям не являются перво­степенными. Однако, по мнению автора, попытки излагать те или иные результаты в полном отрыве от математического ап­парата, на основе которого они получены, являются несостоя­тельными и выхолащивающими объективную количественную сторону изучаемых объектов. Поэтому для понимания излага­емого здесь материала от читателя потребуется определенное владение базовыми знаниями из соответствующих разделов ма­тематического анализа и линейной алгебры. Необходимо при­знать, что любая книга экономико-математического направле­ния может быть подвергнута критике либо за чрезмерную перегруженность математическими изысками и оторванность от реальной экономической проблематики, либо за отсутствие математической строгости и корректности, и, естественно, каж­дый автор, исходя из своих вкусов, представлений и умения, ищет оптимальное сочетание того и другого. Ну а о том, на­сколько это удается, судит читатель...

Книга способствует расширению читательского кругозора и помогает ориентироваться среди разделов, задач и методов ис­следования операций. В этой связи многие темы представлены в обзорном ключе.

Несколько замечаний по используемым в ходе изложения условным обозначениям:

* базовые понятия предмета при их первом появлении в тексте выделяются *курсивом*, а наиболее важные их них (те, кото­рые стоит не забывать и после прочтения!) — **жирным шрифтом;**
* перед фундаментальными определениями стоит символ ☞;
* количество приводимых в данной книге теорем минимизиро­вано (это, однако, не должно создать у неподготовленного читателя превратного впечатления об их действительном количестве); в тех местах, где встречается теорема, ее фор­мулировка выделяется слева двойной чертой;
* доказательство теоремы завершается символом ✌.

## ВВЕДЕНИЕ

Начало развития исследования операций как науки традицион­но связывают с сороковыми годами двадцатого столетия. Среди первых исследований в данном направлении может быть назва­на работа Л. В. Канторовича «Математические методы органи­зации и планирования производства», вышедшая в 1939 г. В за­рубежной литературе отправной точкой обычно считается вышедшая в 1947 г. работа Дж. Данцига, посвященная реше­нию линейных экстремальных задач.

Следует отметить, что не существует жесткого, устоявше­гося и общепринятого определения предмета исследования опе­раций. Часто при ответе на данный вопрос говорится, что «**ис­следование операций** представляет собой комплекс научных методов для решения задач эффективного управления органи­зационными системами» [14].

Природа систем, фигурирующих в приведенном определении под именем «организационных», может быть самой различной, а их общие математические модели находят применение не толь­ко при решении производственных и экономических задач, но и в биологии, социологических исследованиях и других практи­ческих сферах. Кстати, само название дисциплины связано с применением математических методов для управления военны­ми операциями.

Несмотря на многообразие задач организационного управ­ления, при их решении можно выделить некоторую общую по­следовательность этапов, через которые проходит любое опе­рационное исследование. Как правило, это:

1. Постановка задачи.

2. Построение содержательной (вербальной) модели рассматриваемого объекта (процесса). На данном этапе происходит формализация цели управления объектом, выделение возмож­ных управляющих воздействий, влияющих на достижение сфор­мулированной цели, а также описание системы ограничений на управляющие воздействия.

3. Построение математической модели, т. е. перевод сконст­руированной вербальной модели в ту форму, в которой для ее изучения может быть использован математический аппарат.

4. **Решение задач, сформулированных на базе постро­енной математической модели.**

5. Проверка полученных результатов на их адекватность природе изучаемой системы, включая исследование влияния так называемых внемодельных факторов, и возможная коррек­тировка первоначальной модели.

6. Реализация полученного решения на практике.

Центральное место в данной книге отведено вопросам, отно­сящимся к четвертому пункту приведенной выше схемы. Это делается не потому, что он является самым важным, сложным или интересным, а потому, что остальные пункты существенно зависят от конкретной природы изучаемой системы, в силу чего для действий, которые должны производиться в их рамках, не могут быть сформулированы универсальные и содержательные рекомендации. По этому поводу, например, X. Таха заметил, что исследование операций одновременно является как наукой, так и искусством [27].

*Математическое моделирование* в исследовании операций является, сводной стороны, очень важным и сложным, а с дру­гой — практически не поддающимся научной формализации процессом. Заметим, что неоднократно предпринимавшиеся по­пытки выделить общие принципы создания математических мо­делей приводили либо к декларированию рекомендаций самого общего характера, трудноприложимых для решения конкрет­ных проблем, либо, наоборот, к появлению рецептов, примени­мых в действительности только к узкому кругу задач. Поэтому более полезным представляется знакомство с техникой мате­матического моделирования на конкретных примерах.

В качестве таких примеров приведем несколько классиче­ских экономико-математических моделей и задач, которые мо­гут быть сформулированы на их основе.

**Управление портфелем активов.** Рассмотрим проблему принятия инвестором решения о вложении имеющегося у него капитала. Набор характеристик потенциальных объектов для инвестирования, имеющих условные имена от А до F, задается следующей таблицей.

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Доходность (в%) | Срок выкупа (год) | Надежность (в баллах) |
| А | 5,5 | 2001 | 5 |
| В | 6,0 | 2005 | 4 |
| С | 8,0 | 2010 | 2 |
| D | 7,5 | 2002 | 3 |
| Е | 5,5 | 2000 | 5 |
| F | 7,0 | 2003 | 4 |

Предположим, что при принятии решения о приобретении активов должны быть соблюдены условия:

a) суммарный объем капитала, который должен быть вложен, составляет $ 100 000;

b) доля средств, вложенная в один объект, не может превы­шать четверти от всего объема;

c) более половины всех средств должны быть вложены в дол­госрочные активы (допустим, на рассматриваемый момент к тако­вым относятся активы со сроком погашения после 2004 г.);

d) доля активов, имеющих надежность менее чем 4 балла, не может превышать трети от суммарного объема.

Приступим к составлению экономико-математической моде­ли для данной ситуации. Целесообразно начать процесс с опре­деления структуры управляемых переменных. В рассматривае­мом примере в качестве таких переменных выступают объемы средств, вложенные в активы той или иной фирмы. Обозначим их как *хА, хВ, хC, хD, хЕ, хF*. Тогда суммарная прибыль от раз­мещенных активов, которую получит инвестор, может быть представлена в виде



На следующем этапе моделирования мы должны формально описать перечисленные выше ограничения a-d на структуру портфеля.

a) Ограничение на суммарный объем активов:

*хA + хB + хC + хD + хE + хF* ≤ 100 000. (2)

b) Ограничение на размер доли каждого актива:

*хA* ≤ 25 000, *xB* ≤ 25 000, *xC* ≤ 25 000,

*xD*≤ 25 000, *xE*≤ 25 000, *xF*≤ 25 000. (3)

c) Ограничение, связанное с необходимостью вкладывать по­ловину средств в долгосрочные активы:

*xB + xC* ≥ 50 000. (4)

d) Ограничение на долю ненадежных активов:

*хC + хD* ≤ 30 000. (5)

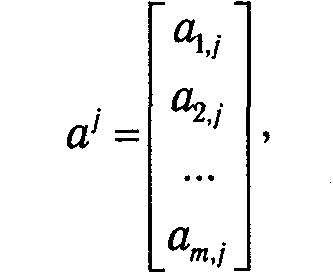
Наконец, система ограничений в соответствии с экономиче­ским смыслом задачи должна быть дополнена условиями неот­рицательности для искомых переменных:

*хA* ≥ 0, *хB* ≥ 0, *хС* ≥ 0, *хD* ≥ 0, *хE* ≥ 0, *хF* ≥ 0. (6)

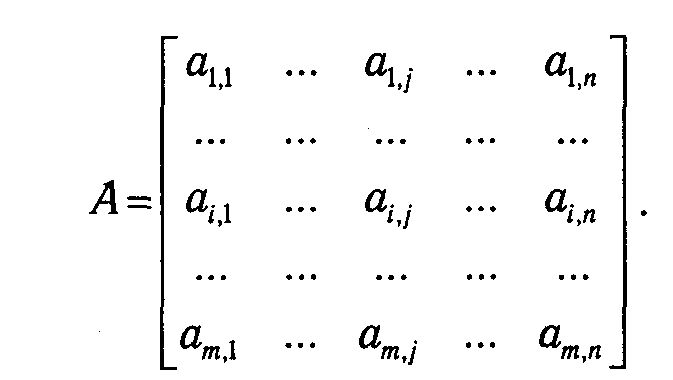
Выражения (1)-(6) образуют математическую модель поведения инвестора. В рамках этой модели может быть по­ставлена задача поиска таких значений переменных *хA, xB, xC, xD, xE, xF*, при которых достигается наибольшее значение при­были (т. е. функции (1)) и одновременно выполняются ограничения на структуру портфеля активов (2)-(6).

Перейдем теперь к рассмотрению более общих моделей и задач.

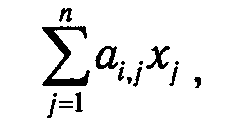
**Простейшая задача производственного планирования**. Пусть имеется некоторый экономический объект (предприятие, цех, артель и т. п.), который может производить некоторую продукцию *n* видов. В процессе производства допустимо исполь­зование *m* видов ресурсов (сырья). Применяемые технологии ха­рактеризуются нормами затрат единицы сырья на единицу произ­водимого продукта. Обозначим через *аi,j* количество *i*-го ресурса (*i 1: m*), которое тратится на производство единицы *j*-го продук­та (*j* 1:n). Весь набор технологических затрат в производстве j-го продукта можно представить в виде вектора-столбца



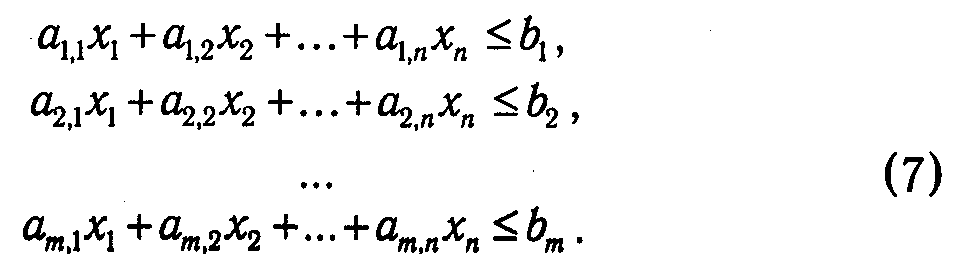
а технологию рассматриваемого предприятия (объекта) в виде прямоугольной матpицы pазмеpности *m* на *n*:



Если *j*-й продукт производится в количестве *xj*, то в рамках описанных выше технологий мы должны потратить *a*1,*jxj*  перво­го ресурса, *a*2*,jxj* — второго, и так далее, *am,jxj - m*-го. *Свод­ный план производства* по всем продуктам может быть пред­ставлен в виде *n*-мерного вектора-строки *х = (х1, х2,...,хj,...,хn).* Тогда общие затраты по *i*-му ресурсу на производство всех про­дуктов можно выразить в виде суммы



представляющей собой скалярное произведение векторов *аj* и *х.* Очевидно, что всякая реальная производственная система име­ет ограничения на ресурсы, которые она тратит в процессе производства. В рамках излагаемой модели эти ограничения по­рождаются *m*-мерным вектором *b=(b1, b2,...,bm),* где *bi* — макси­мальное количество *i*-гo продукта, которое можно потратить в производственном процессе. В математической форме данные ог­раничения представляются в виде системы *m* неравенств:



Применяя правила матричной алгебры, систему (7) можно записать в краткой форме, представив левую часть как произве­дение матрицы *А* на вектор *х*, а правую — как вектор *b*:



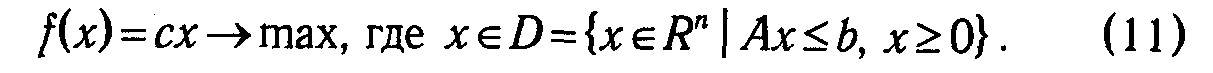
К системе (8) также должны быть добавлены естественные ограничения на неотрицательность компонентов плана произ­водства: *х*1 ≥ 0,..., *хj* ≥0, .... *хn* ≥ 0, или, что то же самое,



Обозначив через *сj* цену единицы *j*-го продукта, получим вы­ражение суммарного дохода от выполнения плана производства, задаваемого вектором *х:*



Формулы (8)-(10) являются не чем иным, как простейшей математической моделью, описывающей отдельные стороны функционирования некоторого экономического объекта, пове­дением которого мы хотим управлять. В рамках данной модели, вообще говоря, можно поставить различные задачи, но, скорее всего, самой «естественной» будет задача поиска такого плана производства *х Rn,* который дает наибольшее значение сум­марного дохода, т. е. функции (10), и одновременно удовлетво­ряет системе ограничений (8)-(9). Кратко такую задачу можно записать в следующем виде:



Несмотря на явную условность рассматриваемой ситуации и кажущуюся простоту задачи (11), ее решение является далеко не тривиальным и во многом стало практически возможным только после разработки специального математического аппа­рата. Существенным достоинством используемых здесь мето­дов решения является их универсальность, поскольку к моде­ли (11) могут быть сведены очень многие как экономические, так и неэкономические проблемы.

Поскольку любая научная модель содержит упрощающие предпосылки, для корректного применения полученных с ее по­мощью результатов необходимо четкое понимание сути этих уп­рощений, что, в конечном счете, и позволяет сделать вывод об их допустимости или недопустимости. Наиболее «сильным» уп­рощением в рассмотренной модели является предположение о прямо пропорциональной (линейной) зависимости между объе­мами расхода ресурсов и объемами производства, которая зада­ется с помощью норм затрат *аi,j.* Очевидно, что это допущение далеко не всегда выполняется. Так, объемы расхода многих ресурсов (например, основных фондов) изменяются скачкооб­разно в зависимости от изменения компонентов объема про­изводства *х*. К другим упрощающим предпосылкам относятся предположения о независимости цен *сj* от объемов *хj*, что спра­ведливо лишь для определенных пределов их изменения, пре­небрежение эффектом кооперации в технологиях и т. п. Данные «уязвимые» места важно знать еще и потому, что они указыва­ют принципиальные направления совершенствования модели.

**Транспортная задача**. Рассмотрим проблему организации перевозки некоторого продукта между пунктами его производ­ства, количество которых равно *m,* и *n* пунктами потребления. Каждый *i*-й пункт производства (*i* 1*:m*) характеризуется запа­сом продукта а*i* ≥ 0, а каждый j-и пункт потребления (*j* 1: *n*) — потребностью в продукте *bj* ≥ 0. Сеть дорог, соединяющая сис­тему рассматриваемых пунктов, моделируется с помощью мат­рицы С размерности *m* на *n*, элементы которой *сi,j* представля­ют собой нормы затрат на перевозку единицы груза из пункта производства *i* в пункт потребления *j.* План перевозки груза в данной транспортной сети представляется в виде массива эле­ментов размерности *m* х *n*:

*х = (x1,1…x1,n, x2,1….,x2,n,… , xi,1, …, xi,n,…, xm,1,…,xm,n).* (12)

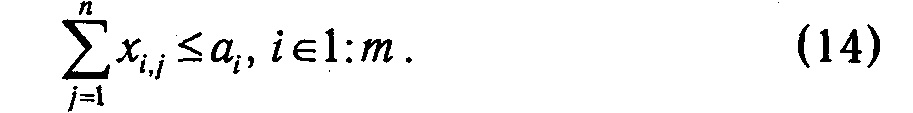
В (12) план перевозок *х* может рассматриваться как вектор, распадающийся на *m* групп, по n элементов в каждой, причем *i*-я группа соответствует объемам груза, вывозимым из *j*-го пун­кта производства во все возможные пункты потребления. Если реальная перевозка между пунктами *i* и *j* отсутствует, то пола­гают *хi,j* = 0.

Ограничения на возможные значения *х*  *Rmn* имеют вид:

1. Ограничение на удовлетворение потребностей во всех пун­ктах потребления:



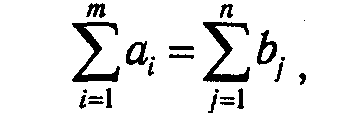
2. Ограничения на возможности вывоза запасов из всех пун­ктов производства:



3. Условия неотрицательности компонентов вектора плана:

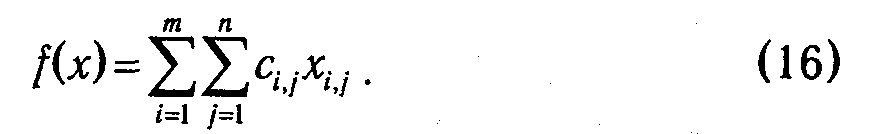
*х, хi,j* ≥ 0, *i* l : *m,* *j* l : *n.* (15)

Существенной характеристикой описываемой модели яв­ляется соотношение параметров *аi* и *bj.* Если суммарный объем производства равен суммарному объему потребления, а именно,

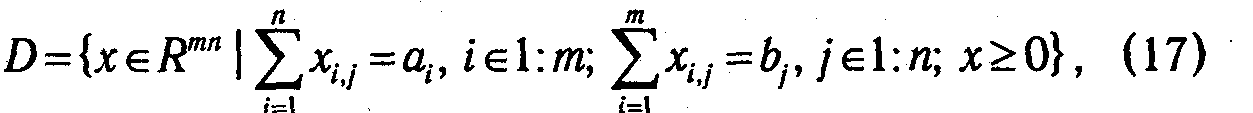


то система называется *сбалансированной.* При выполнении условия сбалансированности разумно накладывать такие огра­ничения на суммарный ввоз и вывоз груза, при которых полно­стью вывозится весь груз и не остается неудовлетворенных потребностей, т. е. условия (13) и (14) приобретают форму ра­венств.

По аналогии с задачей производственного планирования предположим, что затраты на перевозку прямо пропорцио­нальны количеству перевозимого груза. Тогда суммарные за­траты на перевозку в системе примут вид:



Функция (16) и описанные выше ограничения, записанные в форме



задают *транспортную модель*. На ее основе может быть сформулирована задача минимизации суммарных затрат на пе­ревозки:

*f(x)=cx* → min, *x* D, (18)

которая в литературе получила название *транспортной зада­чи в матричной постановке*. Вообще говоря, транспортная задача является частным случаем задачи (11), но в силу ряда особенностей для ее решения применяются специфические ме­тоды, которые, помимо прочего, позволяют прийти к важным теоретическим обобщениям.

Общим для рассмотренных выше задач является то, что в них стоит проблема поиска наибольшего или наименьшего (**оп­тимального**) значения некоторой функции, отражающей цель управления системой, или, как еще говорят, **целевой функции**. Поиск оптимального значения осуществляется на некотором подмножестве допустимых значений переменных, описывающих состояние этой системы, именуемом **множеством допустимых планов**.

Пусть на некотором множестве *D* определена функция *f(x).* Напомним, что точка *х\**, принадлежащая *D (х\* D),* называет­ся **точкой глобального максимума**, если для любого xD выполняется неравенство *f(x) ≤ f(x\*).* В этом случае значение *f(x\*)* называется **глобальным максимумом функции**. Точ­ка *х̀́* называется **точкой локального максимума**, если су­ществует некоторая окрестность этой точки, в любой точке ко­торой значение функции меньше, чем в *х́̀ (f(x) ≤ f(x́̀)).* По аналогии, с точностью до знака неравенства, определяются **глобальный** и **локальный** **минимумы**. Обобщающим понятием для максимума и минимума является таксой термин, как **эк­стремум (оптимум).**

Необходимо отметить, что далеко не всегда весь комплекс целей и задач, стоящий перед моделируемым объектом, может быть выражен в форме некоторой целевой функции. Более того, осознание и осмысление этой проблемы стало своего рода пере­ломным этапом в истории развития исследования операций как науки, положившим конец многим необоснованным ожидани­ям и одновременно давшим толчок к развитию новых направле­ний, связанных с методами многокритериальной оптимизации. Однако следует иметь в виду, что все они базируются на фундаменте методов однокритериальной оптимизации, без ясного понимания которых невозможна работа с более сложным мате­матическим аппаратом.

Как уже отмечалось выше, работа с данной книгой предпола­гает знакомство читателя с классическими методами поиска экст­ремума, основывающимися на аппарате дифференциального исчисления. Но непосредственное применение классических ме­тодов для оптимизации функций, зависящих от большого числа переменных, при наличии значительного количества ограниче­ний наталкивается на серьезные вычислительные трудности, что делает соответствующий аппарат неэффективным.

Мощным инструментом разрешения подобного рода задач стали специальные методы поиска экстремума, составляющие содержание раздела исследования операций, который называ­ется *математическое программирование*. В данном случае понятие программирование употребляется в смысле *планиро­вание* (в отличие от программирования для ЭВМ).

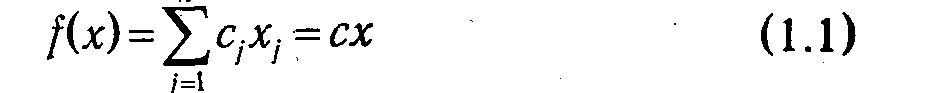
В свою очередь, в зависимости от вида решаемых задач, в ма­тематическом программировании выделяют такие области, как *линейное, нелинейное, дискретное, динамическое, геомет­рическое, стохастическое* программирование. Первые четыре раздела, а также их применение для решения экономических задач составили содержание последующих глав.

# ГЛАВА 1. ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ

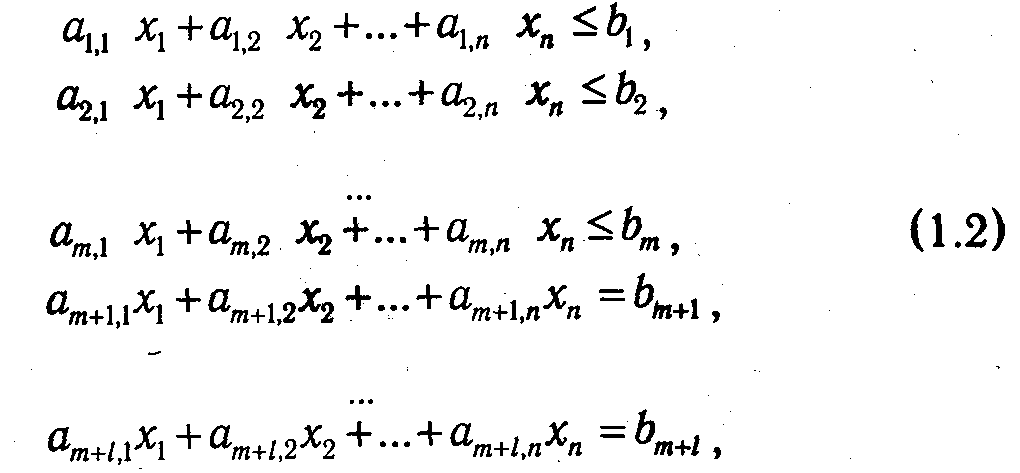
## ПОСТАНОВКА ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

**1.1.1. Формы задачи линейного программирования.**

В общем виде задача линейного программирования\* (в дальней­шем ЗЛП) может быть сформулирована как задача нахождения наибольшего значения линейной функции



на некотором множестве *D Rn*, где *х D* удовлетворяют сис­теме ограничений



и, возможно, ограничениям

*х*1 ≥0, *х*2 ≥0,..., *хj* ≥0,..., *хn* ≥0. (1.3)

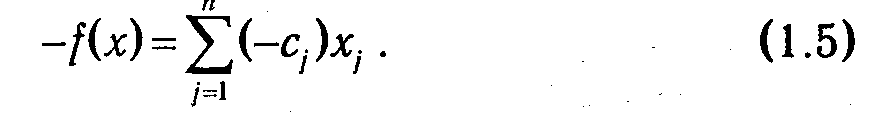
\* Напомним, что частные примеры, сводящиеся к задаче линейного программирования, были описаны во введении.

Не умаляя общности, можно считать, что в системе (1.2) пер­вые *m* ограничений являются неравенствами, а последующие — *l*-уравнениями. Очевидно, этого всегда можно добиться за счет простого переупорядочения ограничений. Относительно направ­ления знака неравенства будем предполагать, что левая часть меньше или равна правой. Добиться этого можно, умножив на (-1) обе части тех неравенств, которые имеют противопо­ложный знак. Ограничения (1.3), вообще говоря, могут быть рассмотрены как частный случай ограничений в форме нера­венств, но в силу особой структуры их обычно выделяют от­дельно и называют *условиями неотрицательности* (или *три­виальными ограничениями).*

Дополнительно следует заметить, что выбор типа искомого экстремума (максимума или минимума) также носит относитель­ный характер. Так, задача поиска максимума функции



эквивалентна задаче поиска минимума функции



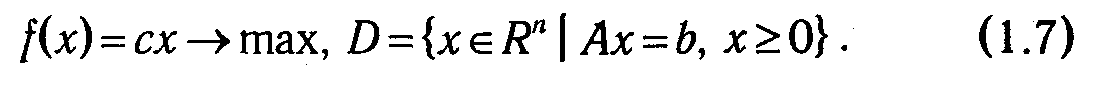
Часто условия задачи (1.1)-(1.3), содержащей ограничения только типа неравенств, бывает удобно записывать в сокращен­ной матричной форме

*f(x) = сх* → max, *Ax ≤ b, х* ≥ 0, (1.6)

где *с* и *х* — векторы из пространства *Rn, b* — вектор из простран­ства *Rm*, а *А* — матрица размерности *m* х *n*.

Задачу линейного программирования, записанную в форме (1.1)-(1.3), называют **общей задачей линейного програм­мирования** (ОЗЛП).

Если все ограничения в задаче линейного программирования являются уравнениями и на все переменные *хj* наложены усло­вия неотрицательности, то она называется задачей линейного программирования в канонической форме, или **канонической задачей линейного программирования** (КЗЛП). В матрич­ной форме КЗЛП можно записать в следующем виде:



Поскольку любая оптимизационная задача однозначно оп­ределяется целевой функцией *f* и областью *D*, на которой отыс­кивается оптимум (максимум), будем обозначать эту задачу парой *(D*, *f*).

Условимся относительно терминологии, которая использу­ется в дальнейшем и является общепринятой в теории линейно­го программирования.

**Планом** ЗЛП называется всякий вектор *х* из пространства *Rn.*

**Допустимым планом** называется такой план ЗЛП, кото­рый удовлетворяет ограничениям (1.2)-(1.3), т. е. содержится в области *D.* Сама область *D* называется при этом *областью допустимых планов*. **Оптимальным планом** *х\** называется такой допустимый план, при котором целевая функция достигает оптимального (в нашем случае — максимального) значения, т. е. план, удовлетворяющий условию

max *f(x) = f(x\*).*

Величина *f\* =f(х\*)* называется *оптимальным значением целевой функции.*

**Решением задачи** называется пара (*х\*, f\**), состоящая из oптимального плана и оптимального значения целевой функции, а *процесс решения* заключается в отыскании множества всех решений ЗЛП.

**1.1.2. Переход к канонической форме**. Подавляющее боль­шинство известных методов решения задач линейного програм­мирования предназначены для канонических задач. Поэтому начальный этап решения всякой общей задачи линейного про­граммирования обычно связан с приведением ее к некоторой эквивалентной канонической задаче.

Общая идея перехода от ОЗЛП к КЗЛП достаточно проста:

* *ограничения в виде неравенств преобразуются в уравне­ния за счет добавления фиктивных неотрицательных переменных хi, (i* 1*:m), которые одновременно входят в целевую функцию с коэффициентом 0, т. е. не оказывают влияния на ее значение;*
* *переменные, на которые не наложено условие неотрица­тельности, представляются в виде разности двух новых неотрицательных переменных:*

- = - =

*хj = хj – хj, (xj ≥* 0*, хj ≥* 0*).*

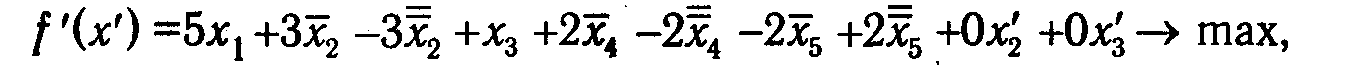
Проиллюстрируем применение описанных выше рекоменда­ций на примере. Пусть задана задача линейного программирова­ния *(D, f)* в общей форме с целевой функцией

*f(x)* = 5*х*1 + 3*x*2 + *x*3 + 2*х*4 -2*х*5 → max

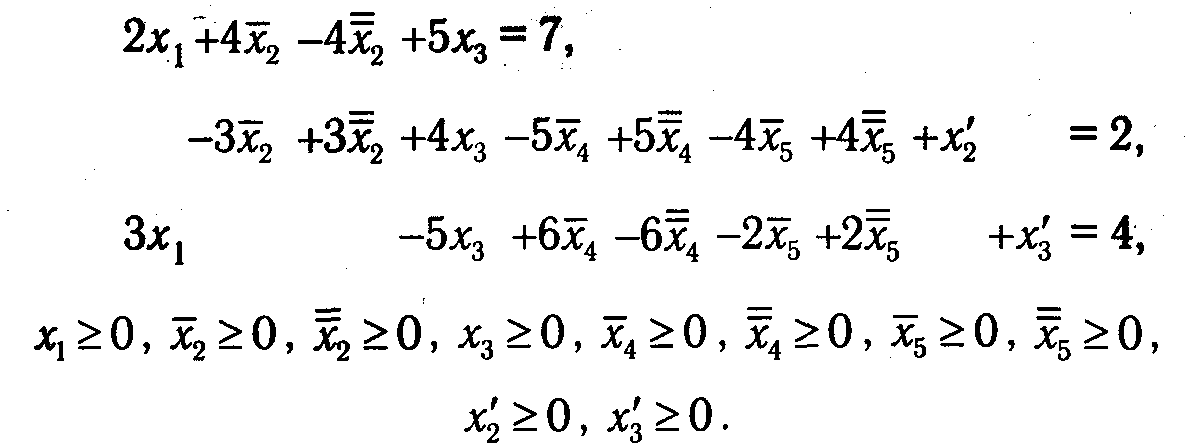
и множеством допустимых планов *D*, определенным системой уравнений и неравенств,

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2*х*1 | + 4*х*2 | + 5*x*3 |  | =7, |
|  | - 3*x*2 | + 4*x*3 | – 5*x*4 – 4*x*5 | ≤ 2, |
| 3*х*1 |  | - 5*x*3 | + 6*x*4 – 2*x*5 | ≤ 4, |
| *х*1≥ 0, |  | *x*3 | ≥ 0. |  |

Тогда в соответствии со сформулированными правилами эк­вивалентная каноническая задача будет иметь вид (*D',f'*), где:



а множество *D'* определено как:



Нетрудно заметить, что «платой» за переход от общей фор­мы задачи линейного программирования к канонической явля­ется рост ее размерности, что, при прочих равных условиях, является фактором, усложняющим процесс решения.

## 1.2. ОСНОВНЫЕ СВОЙСТВА ЗЛП И ЕЕ ПЕРВАЯ ГЕОМЕТРИЧЕСКАЯ ИНТЕРПРЕТАЦИЯ

**1.2.1. Основные понятия линейной алгебры и выпук­лого анализа, применяемые в теории математического программирования**. Кратко напомним некоторые фундамен­тальные определения и теоремы линейной алгебры и выпуклого анализа, которые широко применяются при решении проблем как линейного, так и нелинейного программирования.

Фундаментальным понятием линейной алгебры является *ли­нейное (вещественное) пространство*. Под ним подразуме­вается множество некоторых элементов (именуемых *вектора­ми* или *точками*), для которых заданы операции сложения и умножения на вещественное число (скаляр), причем элементы, являющиеся результатом выполнения операций, также в соот­ветствии с определением должны принадлежать исходному про­странству.

Частными случаями линейных пространств являются веще­ственная прямая, плоскость, геометрическое трехмерное про­странство.

Вектор λ1*a*1 + λ2*a*2 + …+ λm*a*m называется *линейной комбина­цией* векторов *а*1 *а*2,..., *а*m с коэффициентами λ1, λ2, λm,

Система векторов линейного пространства *а*1 *а*2,..., *а*m называется *линейно зависимой*, если существуют такие числа λ1, λ2, λm не равные одновременно нулю, что их линейная комбинация λ1a1 + λ2a2 + …+ λmam равняется нулевому вектору (вектору, все компоненты которого равны нулю). В против­ном случае систему *а*1, *а*2,..., *а*m называют *линейно независи­мой*, т. е. линейная комбинация данных векторов может быть равна нулевому вектору только при нулевых коэффициентах λ1, λ2, …, λm

Максимально возможное количество векторов, которые могут образовывать линейно независимую систему в данном линейном пространстве, называют *размерностью простран­ства*, а любую систему линейно независимых векторов в коли­честве, равном размерности, — *базисом пространства*.

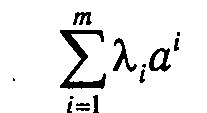
Линейное пространство обычно обозначают как *Rn*, где *n* — его размерность.

Любое подмножество данного линейного пространства, ко­торое само обладает свойствами линейного пространства, назы­вается *линейным подпространством.* Множество *Н*, получае­мое сдвигом некоторого линейного подпространства *L Rn* на вектор *a Rn: H=L+a*, называется *аффинным множеством* (пространством). Если фундаментальным свойством любого ли­нейного пространства или подпространства является принад­лежность ему нулевого вектора, то для аффинного множества это не всегда так. На плоскости примером подпространства явля­ется прямая, проходящая через начало координат, а аффинного множества — любая прямая на плоскости. Характеристическим свойством аффинного множества является принадлежность ему любой прямой, соединяющей две любые его точки. Размерность аффинного множества совпадает с размерностью того линейно­го подпространства, сдвигом которого оно получено.

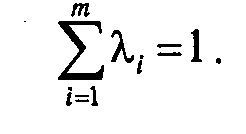
Если рассматривается некоторое линейное пространство *Rn*, то принадлежащие ему аффинные множества размерности 1 на­зываются *прямыми,* а размерности (*n*-1)—*гиперплоско­стями*. Так, обычная плоскость является гиперплоскостью для трехмерного геометрического пространства *R3*, а прямая — ги­перплоскостью для плоскости *R*2. Всякая гиперплоскость делит линейное пространство на два *полупространства.*

Множество V векторов (точек) линейного пространства *Rn* называется *выпуклым,* если оно содержит отрезок прямой, соединяющей две его любые точки, или, другими словами, из того, что *a V* и *bV* , следует, что *х* = (1- λ) х *а+ λ* х *b* *V* , где 0 ≤λ≤ 1.

Линейная комбинация



векторов *а*1, *а*2... *а*m называется *выпуклой*, если λ*i* ≥0, i1:*m* и



Множество, содержащее все возможные выпуклые комбинации точек некоторого множества *М*, называют *выпуклой оболочкой* данного множества. Можно показать, что выпуклая оболочка множества *М* является наименьшим выпуклым множеством, со­держащим *М*.

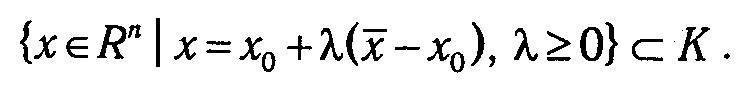
Выпуклая оболочка конечного множества точек называется *выпуклым многогранником*, а непустое пересечение конечного числа замкнутых полупространств — *многогранным выпук­лым множеством*. В отличие от выпуклого многогранника пос­леднее может быть неограниченным.

Точка *v* выпуклого множества *V* называется его *угловой (крайней)* точкой, если она не является внутренней точкой ни для какого отрезка, концы которого принадлежат множеству *V*. Угловые точки выпуклого многогранника являются его *верши­нами,* а сам он — *выпуклой оболочкой* своих вершин.

Множество *К* называется *конусом* с вершиной в точке *x0*, если *x*0  *К* , и из того, что некоторая точка *х* принадлежит *К ( х К* ), следует, что в *К* содержится и луч, начинающийся в *х0* и проходящий через *х*, т. е.



или

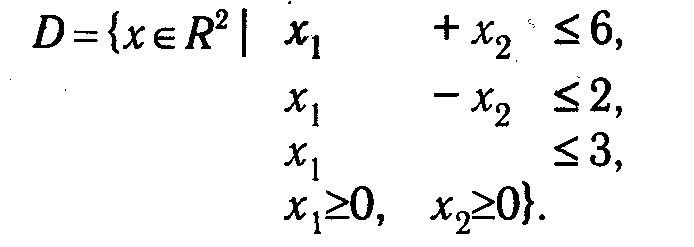


Выпуклая оболочка конечного множества лучей, исходя­щих из одной точки, называется *многогранным выпуклым ко­нусом* с вершиной в данной точке.

**1.2.2. Первая геометрическая интерпретация ЗЛП и графический метод решения**. Рассмотрим следующий пример. Пусть дана задача максимизации линейной целевой функции

*f(x)* = 3*х*1 + *х*2 → max

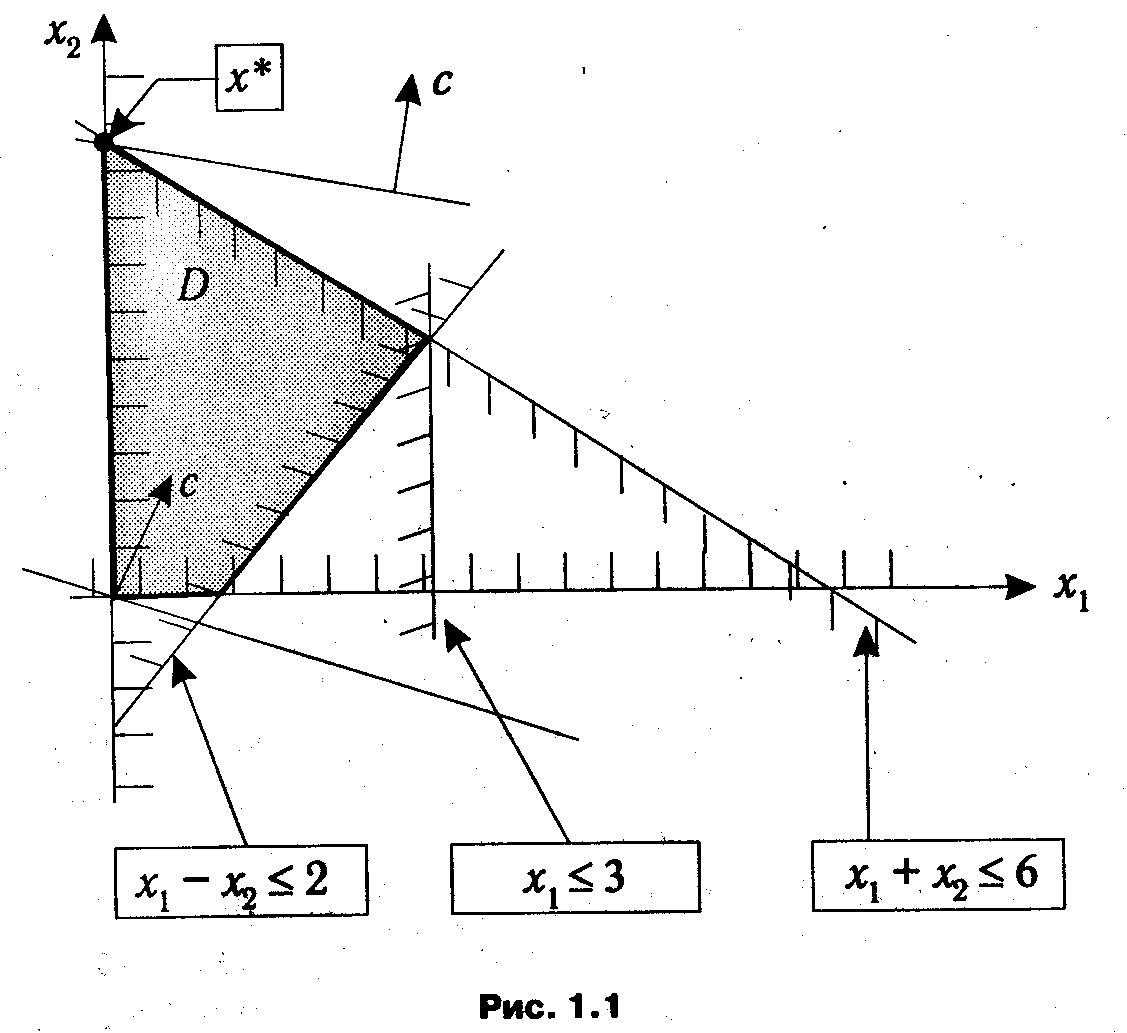
на множестве



Так как количество переменных в неравенствах, задающих область допустимых планов задачи, равно двум, то ее можно изобразить на координатной плоскости (см. *рис. 1.1*).

На *рис. 1.1* показано, что каждое неравенство определяет некоторую полуплоскость. Соответствующие области для каж­дого ограничения отмечены штрихами. Пересечение D данных полуплоскостей (т. е. множество точек, которые одновременно принадлежат каждой их них) является областью допустимых планов задачи. Поведение целевой функции *f(x)* = 3*х*1 + *х*2 в рамках двумерной иллюстрации может быть охарактеризовано с помощью линий уровня.

Напомним, что *линией уровня функции* называется множе­ство точек из ее области определения, в которых функция при­нимает одно и то же фиксированное значение. *Градиентом* функции *f(x)* называется вектор

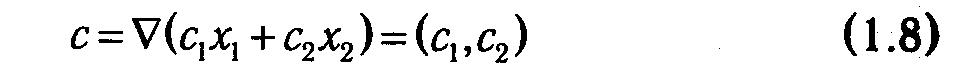


Δ*f(x) = df,…, df*

*dx*1 *dxn*

указывающий направление наиболее быстрого возрастания фун­кции, и, стало быть, ориентированный перпендикулярно лини­ям уровня.

Для линейной функции двух переменных линия уровня пред­ставляет собой прямую, перпендикулярную вектору *с*, который служит градиентом данной функции. Следовательно, если ли­ния уровня определяется уравнением *f(x)=c1x1+ c2x2 =const*, то этот вектоp имеет вид



и указывает направление возрастания функции.

Таким образом, с геометрической точки зрения задача мак­симизации сводится к определению такой точки области *D*, че­рез которую проходит линия уровня, соответствующая наи­большему из возможных значений. Последнее означает, что для нахождения точки экстремума в задаче линейного программирования мы должны сначала построить линию уровня для некоторого произвольного значения целевой функции. Затем необходимо осуществлять ее параллельное передвижение (так, чтобы она оставалась перпендикулярной вектору *с*) до тех пор, пока не достигнем такой точки области допустимых планов *D*, из которой смещение в направлении вектора *с* было бы невоз­можно. Такой метод решения получил название *графического*. Заметим, что решение задачи поиска минимума линейной функ­ции осуществляется аналогично, с той лишь разницей, что дви­жение по линиям уровня должно производиться в направлении, обратном градиенту целевой функции, т. е. по вектору *(-с*).

На *рис. 1.1* изображен некоторый частный случай, для кото­рого решение ЗЛП достигается в угловой точке *х\** = (0, 6) обла­сти *D*. Нетрудно представить, что возможны и другие варианты. Они изображены на *рис. 1.2.*

Рисунок (*а*) иллюстрирует ситуацию неограниченности це­левой функции *f(x)=cx* на множестве *D*, т.е. сколько бы мы ни перемещались по линиям уровня в направлении вектора *с*, ее значение будет возрастать.

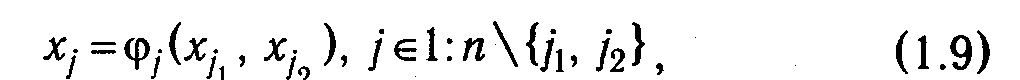
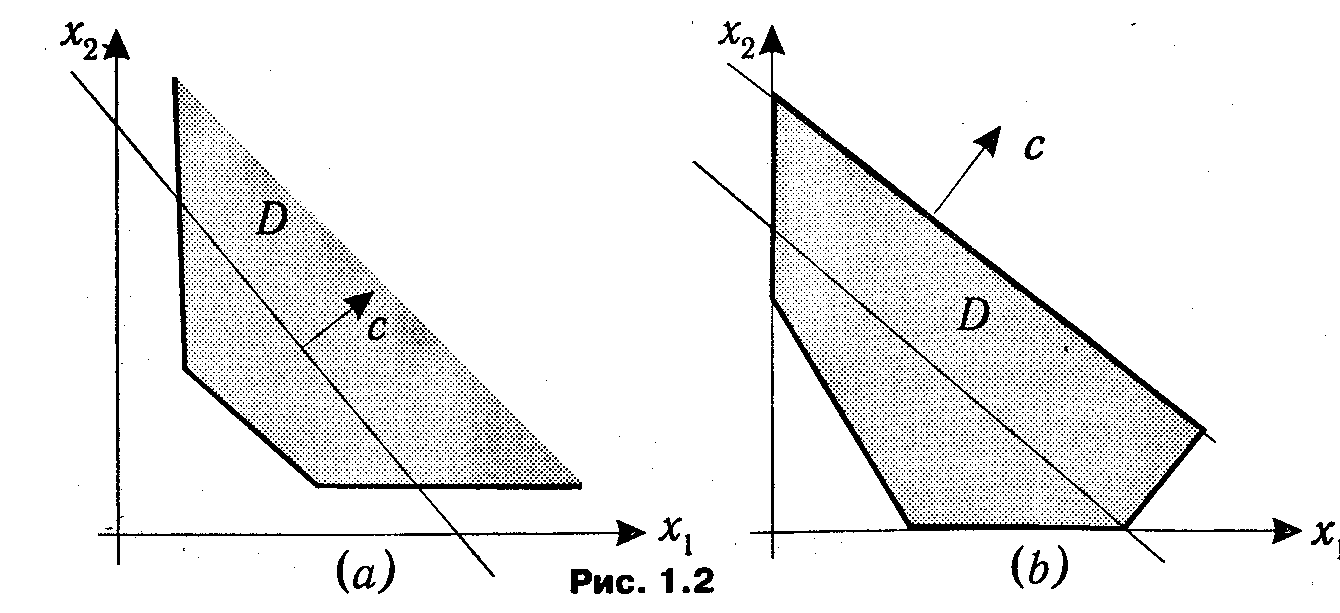
В случае, изображенном на рисунке (*b*), линия уровня, со­ответствующая максимальному значению *f(x),* касается грани множества *D*, и, соответственно, все точки, лежащие на этой гра­ни, являются оптимальными планами.

Во всех рассмотренных иллюстрациях допустимые планы ЗЛП представлялись в виде некоторого многогранного выпук­лого множества на плоскости. Такое их представление в лите­ратуре получило название *первой геометрической интерпре­тации задачи линейного программирования*.

Заметим также, что аналогичным образом могут быть пост­роены интерпретации ЗЛП для случая трехмерного простран­ства *R*3, где множеству *D* будет соответствовать некоторый ограниченный или неограниченный многогранник, а поведе­ние целевой функции будет характеризоваться поверхностями (плоскостями) уровня.

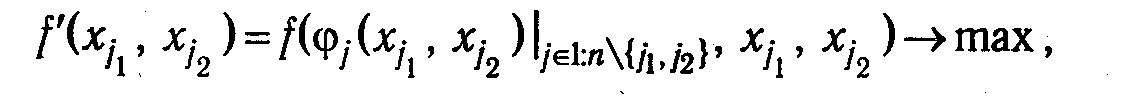
Несмотря на свою очевидную ограниченность, графический метод решения ЗЛП часто оказывается полезным. В частности, он может быть применен не только к задачам с двумя перемен­ными и ограничениями в виде неравенств, но и к каноническим задачам вида (1.7), у которых *n - m* = 2, где *n* — количество пе­ременных, а *m* — ранг матрицы *А*.

Действительно, можно выбрать две произвольные перемен­ные *хj1,xj2* и, используя систему уравнений, выразить через них остальные переменные

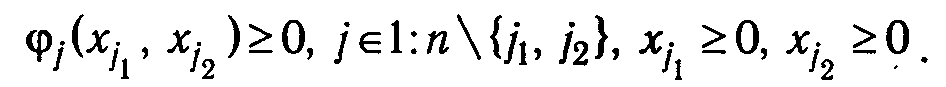


где φj(*xj*1, *xj*2 ) —линейные функции.

Подставив выражения (1.9) в целевую функцию, мы получим эквивалентную задачу



при ограничениях



Последняя ЗЛП может быть решена графически.

**1.2.3. Основные теоремы линейного программирова­ния**. Рассмотрим некоторые теоремы. Отражающие фундамен­тальные свойства задач линейного программирования и лежа­щие в основе методов их решения. Они по существу обобщают на случай задач с произвольным количеством переменных те свойства, которые мы наблюдали в двумерном случае.

*Теорема 1.1. Если целевая функция f принимает мак­симальное значение в некоторой точке множества до­пустимых планов D, то она принимает это значение и в некоторой угловой точке данного множества.*

**Доказательство.**

Чтобы не усложнять изложение, ограничимся тем случаем, когда множество *D* ограничено, и, следовательно, является вы­пуклым многогранником.

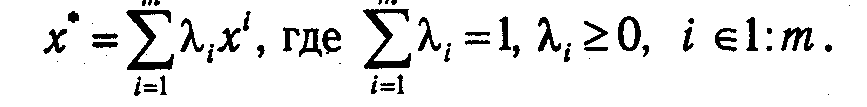
Для доказательства воспользуемся следующим известным свойством ограниченных выпуклых множеств:

|  |
| --- |
| *Если D — замкнутое ограниченное выпуклое множе­ство, имеющее конечное число угловых точек, то лю­бая точка х D может быть представлена в виде вы­пуклой комбинации угловых точек D\*.* |

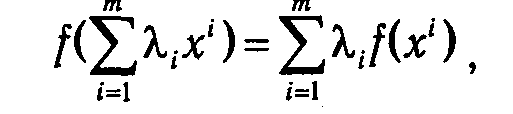
\* Строгое доказательство данного утверждения см., например, в [14].

Пусть *х*1, *х*2,…,*хm*— угловые точки множества *D*, а *х\** — точка, в которой целевая функция *f* достигает максимума.

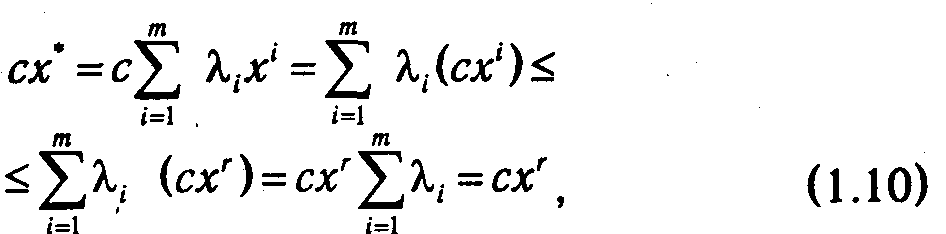
На основе сформулированного выше утверждения точку *х\** можно представить в виде выпуклой комбинации угловых то­чек *х*1, *х*2,..., *xm*



Так как *х\** — точка максимума, то для любого *х D сх\* ≥ сх*. Функция *f(x)* — линейная, поэтому



cледовательно,



где *xr* — угловая точка, удовлетворяющая условию

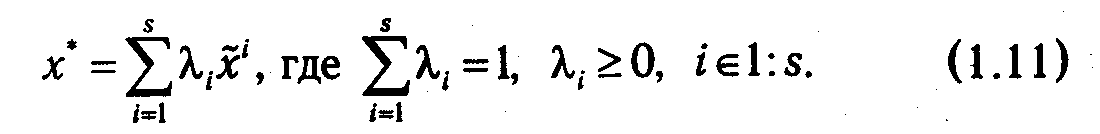


Из (1.10) видно, что *сх\* ≤ схr*. В то же время справедливо об­ратное неравенство: *сх\* ≥ схr*. Откуда следует, что *сх\* = схr*, т. е. существует по крайней мере одна угловая точка *хr*, в которой целевая функция принимает максимальное значение. ✌

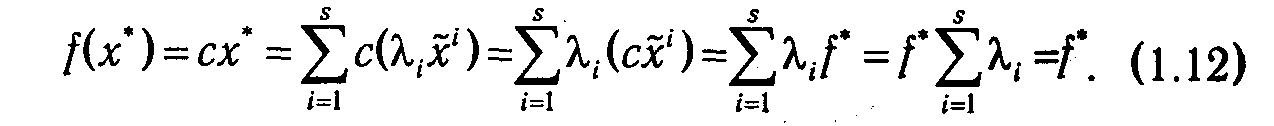
|  |
| --- |
| *Теорема 1.2. Если целевая функция f принимает мак­симальное значение в нескольких точках множества D, то она принимает это же значение в любой точке, яв­ляющейся их выпуклой комбинацией.* |

**Доказательство.**

Пусть максимальное значение функции *f* достигается в точ­ках *х̃*1, *х̃*2,...,*x̃s* , т. е. *сх~i=f\*,* *i* l*:s.* Рассмотрим произвольную выпуклую комбинацию этих точек



Найдем значение целевой функции в точке *х\**

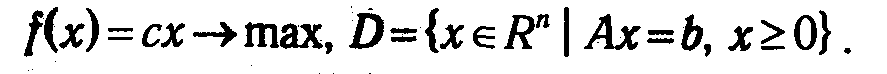


Итак, для произвольной выпуклой комбинации *х\** точек *х̃1*, *х̃*2,...,*x~s* справедливо равенство

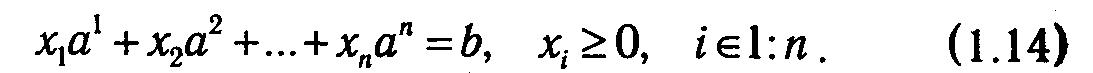


## 1.3. БАЗИСНЫЕ РЕШЕНИЯ И ВТОРАЯ ГЕОМЕТРИЧЕСКАЯ ИНТЕРПРЕТАЦИЯ ЗЛП

**1.3.1. Векторная форма записи КЗЛП и ее примене­ние**. Рассмотрим каноническую задачу линейного программи­рования



Обозначим через *аj* столбцы матрицы *А* и будем рассматри­вать их как векторы пространства *Rm.* Тогда каждому допусти­мому плану КЗЛП — *n*-мерному вектору *х* — соответствует неотрицательная линейная комбинация столбцов *аj*, равная столбцу *b Rm:*



Такое представление ограничений КЗЛП обычно называют *векторной формой записи.*

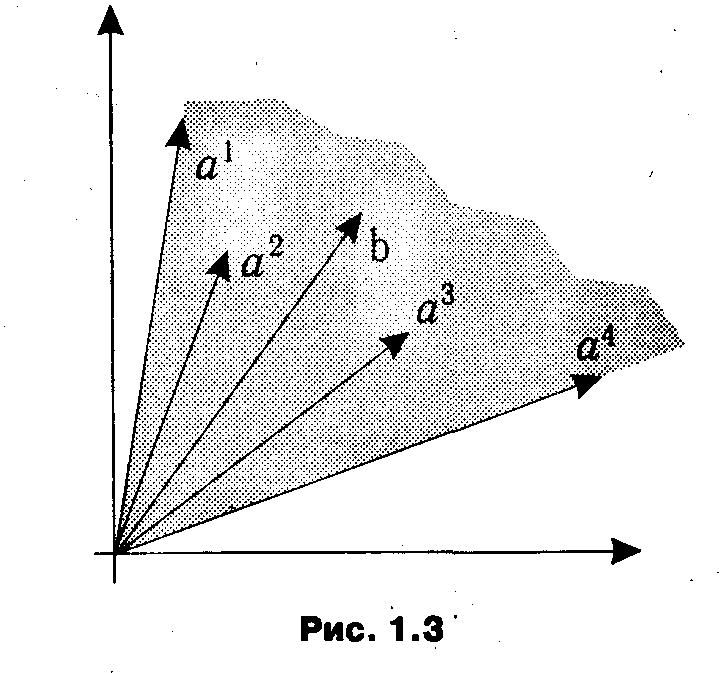
Векторы *аj, j*  l*:n* будем называть *векторами требований* задачи (*D, f*), а вектор *b* — *вектором ограничений*. Множе­ство всех неотрицательных линейных комбинаций столбцов аj с геометрической точки зрения может быть представлено как многогранный выпуклый конус, натянутый на систему векто­ров *аj* в пространстве *Rm (рис. 1.3).*

Соответственно, вопрос о существовании допустимого пла­на задачи (*D, f*) равнозначен вопросу о принадлежности векто­ра *b* данному конусу, а компоненты *хj* некоторого допустимого плана *х D* являются не чем иным, как *коэффициентами раз­ложения вектора ограничений задачи b по векторам, требо­ваний аj.*

Такое представление КЗЛП получило название ***второй гео­метрической интерпретации.***

В дальнейшем без ограничения общности можем предпола­гать, что число уравнений, задающих множество *D*, меньше или равно числу переменных задачи (*m ≤ n*). Действительно, если это не так, то либо система уравнений *Ах* = *b* несовместна (и, зна­чит, множество *D* пустое), либо содержит избыточные (линейно зависимые) уравнения.

Если некоторые *т* столбцов *аj*1 ,*аj*2 ,...,*аjm*матрицы *A* явля­ются линейно независимыми, то они образуют базис в простран­стве *Rm*, и их, вообще говоря, будет достаточно для представ­ления вектора *b* в виде линейной комбинации указанных столбцов. Это означает, что остальные столбцы войдут в данное разложение с нулевыми коэффициентами. Если к тому же коэффициенты линейной комбинации окажутся неотрицатель­ными, то мы получаем так называемый базисный допустимый план *х*, у которого не более *m* компонентов отличны от нуля. Сформулируем определение базисного плана более строго, так как это одно из фундаментальных понятий теории линейного пpогpаммиpования.



* *Пусть задана некоторая каноническая ЗЛП (D,f), А —матрица системы ограничений задачи, и β= {аj1, аj2,..., аjm} —линейно независимая система столбцов матрицы А, образующая базис Rm. Обозначим множе­ство номеров столбцов, входящих в систему b, через N (β) = {j1, j2,..., jm}. План х называется* базисным планом *задачи (D,f), если его компоненты, соответствующие базисным столбцам и называемые базисными компонен­тами, больше или равны нулю (хj ≥0, j N (β)}, а все остальные компоненты* (небазисные) *— равны нулю (xj = 0, j ∉ N (β)).*
* *Базисный план х называется* невырожденным*, если все его базисные компоненты строго*

*положительны, и* вы­рожденным *в противном случае.*

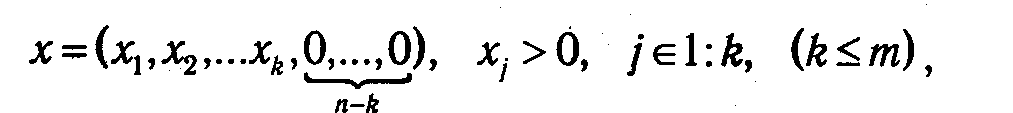
**1.3.2. Свойства базисных планов**. Следующая теорема трактует понятие базисного плана в терминах первой геометри­ческой интерпретации ЗЛП.

|  |
| --- |
| ***Теорема 1.3.*** *Каждый допустимый базисный план яв­ляется угловой точкой множества допустимых пла­нов D.* |

**Доказательство.**

Ради простоты положим, что базисными векторами являют­ся первые *m* столбцов матрицы *A*, т. е. β={*a*l, *a*2,...,*am*}. Тогда утверждение теоремы 1.3 может быть переформулировано сле­дующим образом:

*Если существует такой n-мерный вектор*

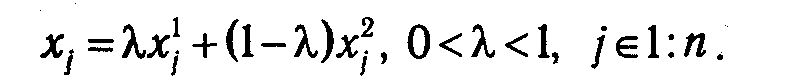


*что* *x*1 *a*1 +*х*2 *а*2 +...+*xk ak* =*b*, *то х есть угловая точка множе­ства D.*

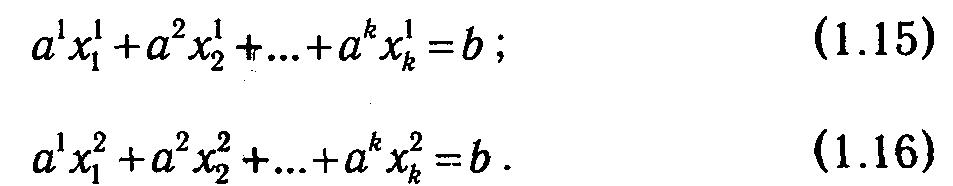
Проведем доказательство от противного, т. е. предположим, что рассматриваемый базисный план *х* не является угловой точ­кой множества *D*. Тогда ее можно представить в виде выпуклой комбинации некоторых двух различных допустимых планов *х*1 и *х*2:

*x* = λ*x*1+(1-λ)*x*2, 0<λ<1.

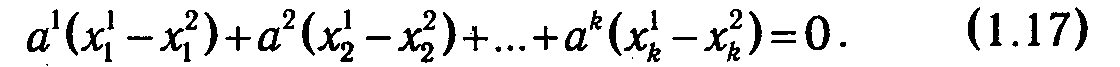
В координатной форме последнее утверждение означает



Поскольку последние (*n - k*) компоненты вектора *х* по пред­положению равны 0, а числа *хj*1, *xj*2 ≥ 0 и λ, (1 -λ )>0, то эти же компоненты в векторах *x*1 и *х*2 также равны 0. Поэтому, с учетом допустимости планов *х*1 и *х*2, можно утверждать, что



Вычитая из (1.15) (1.16), получим



Так как векторы *а*1, *а*2,...,*аk* — линейно независимы, то коэф­фициенты *х*11 –*х*21 =0,..., *х1k –х2k* =0, из чего следует, что *x*1 =*х*2. Это противоречит предположению, что *х*1 и *х*2 являются раз­личными угловыми точками множества *D*. Следовательно*, х* не может быть представлен в виде выпуклой комбинации двух точек *D* и по определению является угловой точкой данного множества. ✌

Интересно отметить, что справедливо и обратное утвержде­ние, которое приведем без доказательства:

*Если х — угловая точка множества D, то она являет­ся допустимым базисным планом задачи (D, f).*

В завершение параграфа необходимо отметить практиче­ское значение установленной связи между угловыми точками и допустимыми базисными решениями: она позволяет формали­зовать (и тем самым существенно упростить) процесс перехода от одной угловой точки к другой.

## 1.4. СИМПЛЕКС-МЕТОД

**1.4.1. Основные этапы симплекс-метода**. Исходя из свойств линейных экстремальных задач, рассмотренных в пре­дыдущих параграфах, можно заключить, что на принципиаль­ном уровне поиск их решений сводится к последовательному перебору угловых точек множества допустимых планов или, что то же самое, перебору соответствующих допустимых ба­зисных планов. Следует подчеркнуть, что такой перебор для реальных многомерных задач возможен только в теоретиче­ском плане и на практике неосуществим (или крайне неэффек­тивен) даже при условии использования мощной вычислитель­ной техники. Средством решения данной проблемы явились прикладные оптимизационные методы, основанные на последо­вательном, целенаправленном переборе базисных планов ЗЛП.

Классическим методом решения ЗЛП стал *симплекс-ме­тод*, получивший также в литературе название *метода по­следовательного улучшения плана*, разработанный в 1947г. американским математиком Джорджем Данцигом.

Идея такого перехода от одного базисного плана к другому, при котором происходит «улучшение» значения целевой функ­ции, может быть продемонстрирована для случая *m* = 2 с помо­щью *рис. 1.4.* Если вектор ограничений b принадлежит конусу, натянутому на некоторые два базисных вектора условий {аj1,аj2}, то существует такой базисный план *х* с базисными компонентами *хj1, хj2,* что *b= хj1aj1 + хj2aj2* . Разумеется, таких планов может быть несколько в зависимости от выбора систе­мы базисных векторов. Чтобы различать их по соответствую­щей величине целевой функции *f(x)=cj*1*xj*1 +*сj*2*хj*2, вводятся так называемые *расширенные векторы условий* и *ограничений*. В общем случае расширенный столбец условий *āj* получается соединением коэффициента целевой функции *сj* и столбца *аj:*

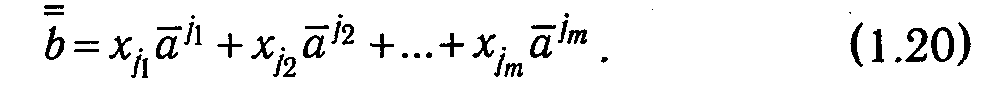


*расширенный вектор ограничений* определим как



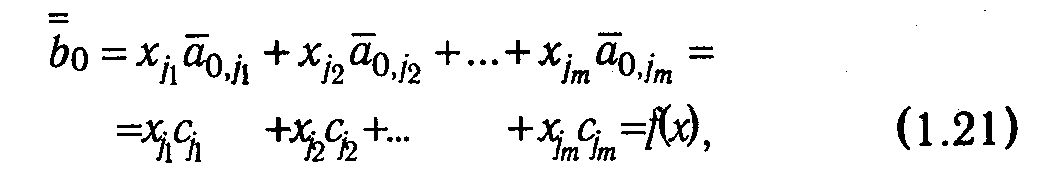
В дальнейшем для удобства нумерации элементов будем счи­тать, что добавляемый коэффициент целевой функции *сj* явля­ется нулевым элементом *j*-го расширенного столбца условий, т. е. *ā0,j = сj*. При изображении расширенных векторов нулевая координата откладывается вдоль вертикальной оси — *оси апп­ликат.*

Рассмотрим также вектор



=

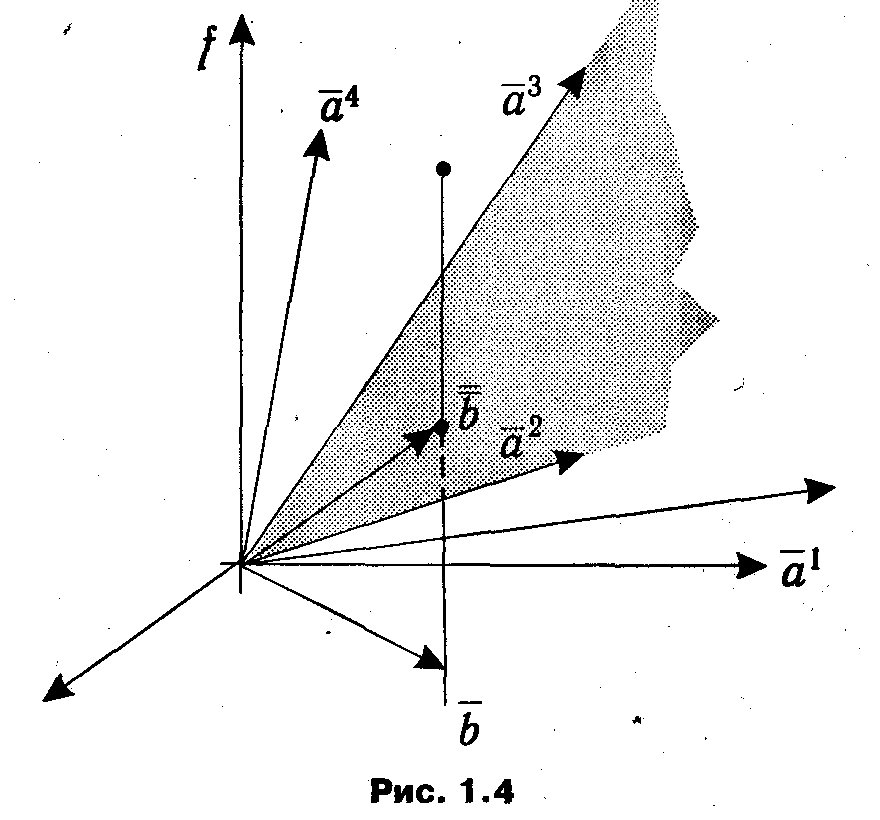
Геометрически определение вектора *b* означает, что он при­надлежит конусу, натянутому на расширенные векторы, а век­тор служит его проекцией. Нулевая координата вектора *b* имеет вид:



т. е. равна значению целевой функции для выбранного базисно­го плана.

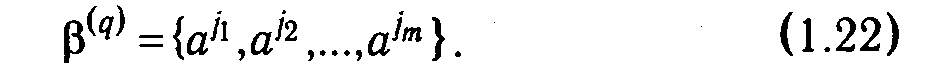
Из геометрической иллюстрации следует, что *для решения задачи мы должны среди векторов аj выбрать такой набор {аj1,аj2}, чтобы, прямая, проведенная через конец вектора параллельно оси аппликат, пересекала конус, натянутый на систему соответствующих расширенных столбцов { āj1, āj2}, в «наивысшей» точке.*

На *рис. 1.4* выделен конус, натянутый на систему расширен­ных столбцов *ā2* и *ā3*, отвечающих текущему допустимому ба­зису. Нетрудно заметить, что данный базис не является опти­мальным, например, для базиса {*а3, а4*} точка пересечения соответствующего конуса и прямой будет находиться выше. Можно, наоборот, указать базис с «худшим» значением целе­вой функции: {*а1 а2*}. Наконец, рассматриваемая геометричес­кая интерпретация КЗЛП иллюстрирует и общую идею крите­рия, который используется в симплекс-методе для определения оптимальности (или неоптимальности) текущего плана*: если существуют векторы-столбцы, лежащие выше плоскости, проходящей через векторы текущего базиса, то он не явля­ется оптимальным и может быть «улучшен».*

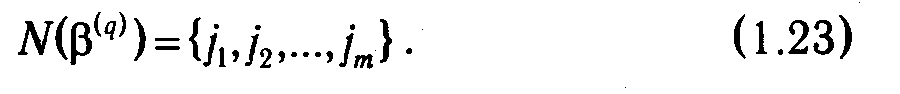


С вычислительной точки зрения критерий оптимальности в симплекс-методе реализуется через нахождение специальных оценок для небазисных векторов-столбцов, относительно теку­щего базиса.

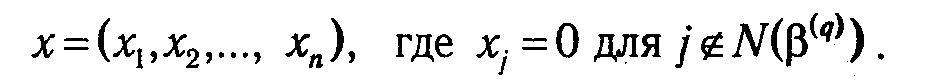
Для удобства дальнейшего изложения введем некоторые обозначения. Учитывая то, что симплекс-метод представляет собой некоторый итеративный вычислительный процесс, то че­рез *q* будем обозначать номер очередной (текущей) итерации. Соответственно, набор базисных столбцов, получаемых на *q*-й итерации, будет обозначаться как *β*(*q*):



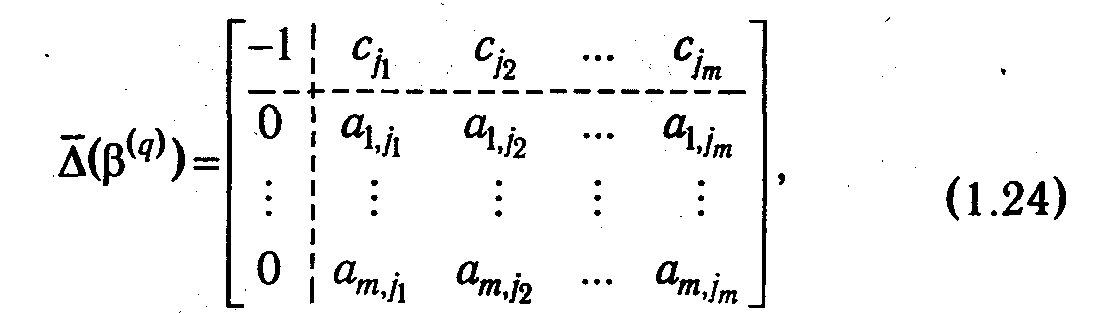
Для того чтобы было легче отличить номер итерации от номе­ров компонентов матриц и векторов, он заключен в круглые скобки. Номера столбцов, входящих в базис, обозначим через *N*(β(q)), а именно



При этом *Nr*(β(q)) =*jr* — номер столбца, занимающего *r*-ю пози­цию в базисе. Тогда текущий базисный план *х* имеет вид:



Обозначим через Δ(ß(q)) матрицу, составленную из столбцов {аj1, аj2,..., аjm}, образующих базис, через Δ(ß(q)) матрицу, об­разованную из соответствующих расширенных столбцов и до­полненную слева столбцом специального вида:

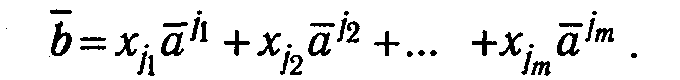


а через ∆-1(β(q)) и ∆-1(β(q)) — матрицы, обратные по отношению к ним. Также представляется удобным ввести отдельные обо­значения для элементов матрицы ∆-1(β(q)):

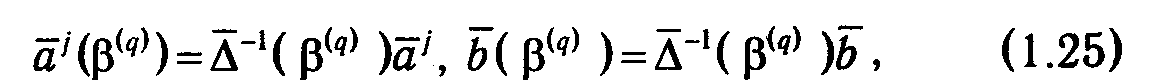
δ*i*(β(q)) — *i*-я строка матрицы ∆ˉ-1(β(q)), (*i* 0: *m* );

δ*ij*(β(q)) — элемент матрицы ∆-1(β(q)), находящийся в *i*-й строке *j*-го столбца (*i* 0: *m*, *j* 0 : *m*).

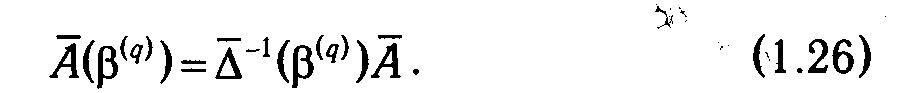
Расширенный вектор ограничений *b* представляется в виде линейной комбинации расширенных векторов условий с коэф­фициентами, равными базисным компонентам текущего базис­ного плана:



Если интерпретировать компоненты векторов а*j* и *b* как ко­ординаты в ортогональном базисе, то их столбцы координат от­носительно произвольного базиса (β(q)), дополненного единич­ным вектором (-1,0,..., 0), направленным противоположно оси аппликат примут вид:



а для расширенной матрицы задачи в целом можно записать



Нулевая строка данной матрицы *a*0(β(*q*)) содержит координаты расширенных векторов условий по оси аппликат. Согласно по­строению, элементы данной строки имеют следующие знаки:

* *a*0,j (β(*q*)) < 0 — *для расширенных векторов условий, рас­положенных выше плоскости, натянутой на систему расширенных базисных векторов;*
* *a*0,j (β(*q*)) > 0 — *для расширенных векторов условий, рас­положенных ниже плоскости, натянутой на систему расширенных базисных векторов;*
* *a*0,j (β(q)) = 0 — *для расширенных базисных векторов.*

Подводя итог сказанному, сформулируем ***критерий опти­мальности допустимого базисного плана*** в симплекс-методе:

☞ *план является оптимальным, если для всех* *j* ∈1: *n* *a*0,j (β(*q*)) ≥ 0, *и неоптимальным в противном*

*случае, т. е. если существует такое l* ∈ l : *n*, что *a*0*l* (β(*q*)) < 0.

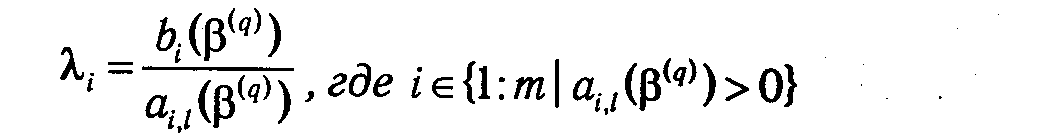
Значения *a*0,j (β(*q*))также называют *оценками столбцов мат­рицы А* относительно текущего базиса, или *симплекс-разнос­тями.*

В случае неоптимальности текущего базиса в алгоритме сим­плекс-метода осуществляется переход к следующему базису. Это делается за счет вывода одного столбца из базиса и ввода другого. Для обеспечения улучшения значения целевой фун­кции в базис должен быть введен вектор-столбец, имеющий от­рицательную оценку. Если таких столбцов несколько, то ***для ввода рекомендуется выбирать столбец***, имеющий *макси­мальную по модулю оценку*. Отметим, что данное правило но­сит относительный характер и не гарантирует наилучшего выбора вводимого столбца. Одновременно на этой стадии тре­буется принять решение о том, какой столбец следует вывести из базиса. Сделать это нужно таким образом, чтобы вновь формируемый базис оказался допустимым. Данное требование может быть легко проиллюстрировано для случая *m* = 2. Например, на *рис. 1.3* векторы {*а*2,*а*3} образуют допустимый ба­зис, а векторы {*а*3,*a*4} —недопустимый, т. к. разложение *b* по *а*3 и *а*4 содержит один отрицательный компонент плана, что противоречит условиям КЗЛП.

Можно доказать, что допустимость базиса, к которому осу­ществляется переход, обеспечивается следующим ***правилом вывода столбца из текущего базиса:***

☞ *для столбца l, претендующего на ввод в базис, и вектора ограничений b рассматриваются*

*отношения*

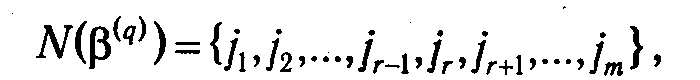


*и определяется такая строка r, что*

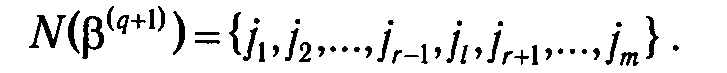


*Полученный индекс r определяет номер столбца в* *N*(β(*q*)), *выводимого из базиса, а именно*, *N*(β(*q*)).

Таким образом, если базис на *q*-й итерации включал столбцы с номерами



то базис на итерации *q* + 1 будет состоять из столбцов с номе­рами:



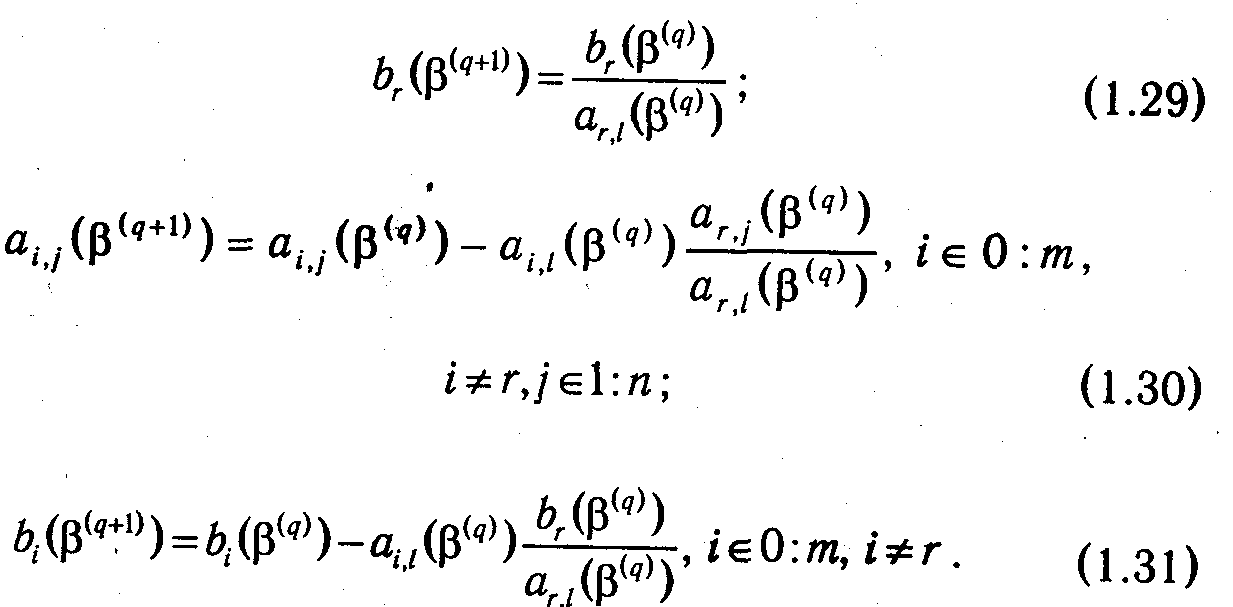
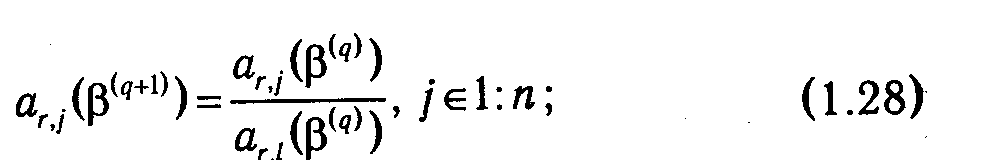
Отдельно следует обсудить тот случай, когда столбец *al* (β(*q*)), претендующий на ввод в базис, не содержит положительных компонентов *al* (β(*q*)) ≤ 0). Это означает, что целевая функция в задаче не ограничена на множестве допустимых значений, т. е. может достигать сколь угодно большого значения. Последнее, очевидно, означает завершение процесса вычислений ввиду *отсутствия оптимального плана.* Геометрически ситуация, когда a*l* (β(*q*)) ≤ 0, соответствует тому, что ось ординат оказыва­ется внутри конуса, натянутого на систему расширенных стол­бцов *аj*, а значит, прямая, проведенная через конец вектора *b* параллельно оси аппликат, однажды «войдя» в этот конус, более никогда из него «не выходит».

Вообще говоря, после перехода от базиса β(*q*) к базису β(*q*+1) мы можем заново сформировать матрицы ∆ (β(*q+*1)), Δ-1 (β(*q*+1)) и, вычислив *А*(β(*q*+1)) = Δ-1 (β(*q*+1))*A*, делать выводы о его оптималь­ности. Однако, учитывая, что β(*q*+1) отличается от β(*q*) всего лишь одним столбцом, с точки зрения техники вычислений представляется рациональным непосредственно переходить от *A* (β(*q*)) и *b* (β(*q*)) к *A* (β(*q*+1)) и *b* (β(*q*+1)) . Дело в том, что у матриц типа *A* (β(*q*)) столбцы, соответствующие базисным векторам, состоят из нулей, за исключением одного элемента, равного единице. Позиция этого ненулевого элемента определяется по­рядковым номером базисного столбца в *N*(β(*q*)). Поэтому для получения матрицы *A* (β(*q*+1)) достаточно с помощью линейных операций над строками матрицы *A* (β(*q*)) привести ее столбец, соответствующий вводимому в базис вектору, к «базисному» виду.

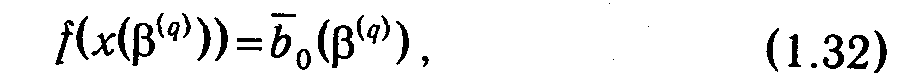
Для это применяется *преобразование Жордана—Гаусса* (так называемый метод полного исключения). В данном случае оно состоит в том, что мы должны «заработать» единицу на мес­те элемента *ar,l*(β(*q*)) (он обычно называется *ведущим*)\* и нули на месте остальных элементов столбца *al*(β(*q*)). Первое достига­ется посредством деления *r*-й строки на ведущий элемент, вто­рое — путем прибавления вновь полученной *r*-й строки, умно­женной на подходящий коэффициент, к остальным строкам матрицы *A*(β(*q*)).

\* Напомним, что *l* — номер столбца, вводимого в базис, а *r* — номер строки в симплекс-таблице, определяющей номер столбца, выводимого из базиса.

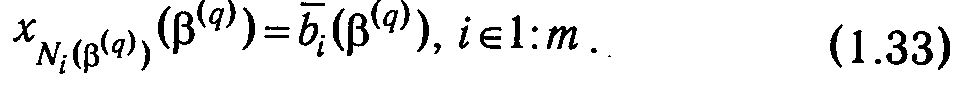
Формально результат выполнения данного преобразования над элементами *A*(β(*q*)) и *b*(β(*q*)) может быть выражен в следующем виде



Следует особо отметить смысл элементов вектора *b*(β(*q*)). Его нулевой компонент *b*0(β(*q*)) в соответствии с построением содержит *значение целевой функции*, достигаемое ею на теку­щем плане



а остальные элементы — *ненулевые компоненты этого плана:*



Название метода произошло от понятия симплекса. Напом­ним, что *m-симплексом* называют выпуклый многогранник, аффинная оболочка\* которого есть аффинное множество размерности *m*. В данном случае можно считать, что система рас­ширенных базисных столбцов {*аj*1 ,*аj*2,..., *аjm*}, рассматривае­мых как точки в *Rm*+1, порождает (*m* -1)-мерный симплекс в пространстве *Rm*+1.

\* *Аффинной оболочкой* множества называют наименьшее аффинное множество, в котором содержится данное множество.

В заключение настоящего пункта обобщим изложенные во­просы и приведем *схему алгоритма симплекс-метода для решения задачи максимизации.* Она включает однократно выполняемый 0-этап и повторяемый конечное число раз I-этап (стандартную итерацию).

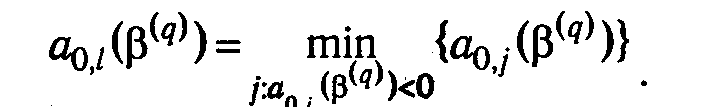
**0-этап. Нахождение допустимого базисного плана** (см. п. 1.4.5). Результатом 0-этапа является допустимый базис­ный план *х*(β(1)), а также соответствующие ему матрица *A*(β(1)) и вектор *b*(β(1)), которые будут использованы на первой итера­ции. Полагаем номер текущей итерации *q* равным 1 и переходим к I-этапу.

**I-этап. Стандартная итерация алгоритма** — выполня­ется для очередного базисного плана *x*(β(*q*)).

1°. *Проверка оптимальности* текущего базисного плана: осуществляется просмотр строки оценок *а*0(β(*q*) ). Возможны два варианта:

1′. *a*0(β(*q*) )≥0 — план, соответствующий текущему базису за­дачи, **оптимален**. Вычислительный процесс закончен. Соглас­но формулам (1.33) и (1.32) выписываются оптимальный план задачи *х*\* = *x*(β(*q*)) и значение целевой функции *f*(*x*\*) = *f*(*x*(β(*q*))).

1″. В строке оценок *а*0(β(*q*)) существует по меньшей мере один элемент *а*0,j (β(*q*))<0, т. е. имеющий отрицательную оцен­ку. Следовательно, план *x*(β(*q*)) —**неоптимален.** Выбирается столбец с номером *l*, имеющий минимальную (максимальную по абсолютной величине) отрицательную оценку



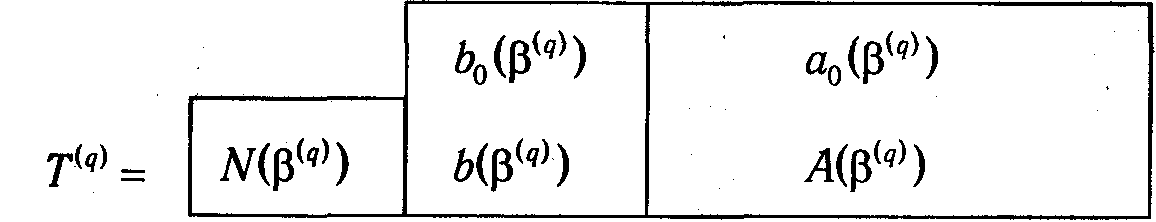
Он называется *ведущим* и *должен быть введен в очередной базис*. Переходим к пункту 2° алгоритма.

2°. *Определение столбца, выводимого из базиса*. Рассмат­ривается ведущий столбец *аl*(β(*q*)) Возможны два варианта:

2'. Для всех *i*∈1:*m* *аi,l*(β(*q*))≤0. Делается вывод о *неогра­ниченности целевой* функции и завершается вычислительный процесс.

2". Существует по крайней мере одна строка с номером *i*∈1:*m*, для которой *аi,l*(β(*q*))>0 . Согласно правилу (1.27) опре­деляются место *r* и номер *Nr*(β(*q*)) = *jr* для столбца, выводимого из базиса. Переходим к пункту 3° алгоритма.

3°. *Пересчет элементов матрицы А и столбца b* относи­тельно нового базиса. В соответствии с формулами (1.28)-(1.31) осуществляем расчет элементов матрицы задачи *А* и век­тора ограничений *b* относительно базиса β(*q+*1), который полу­чается после введения в базис β(*q*) столбца *аl* и выводом из него столбца *аr*. Полагаем номер текущей итерации *q*: = *q*+l и пере­ходим к первому пункту алгоритма.



**Рис.1.5**

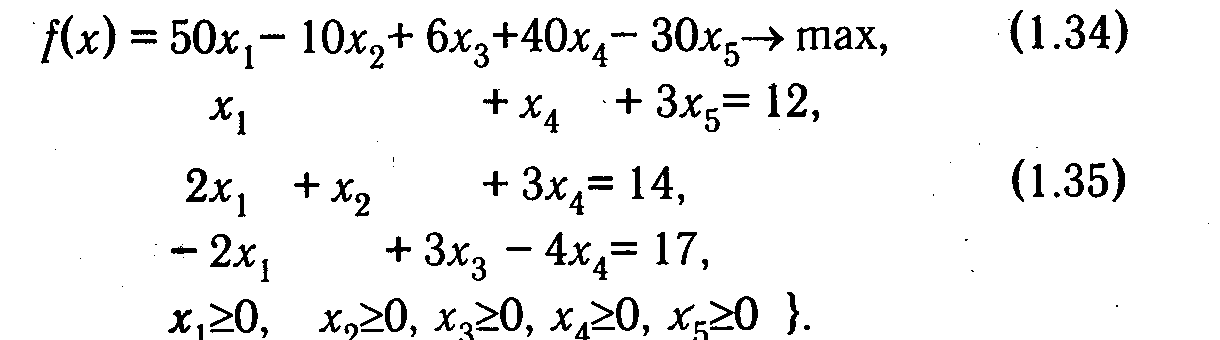
**1.4.2. Табличная реализация симплекс-метода**. С точ­ки зрения обеспечения рациональности и наглядности вычис­лительного процесса выполнение алгоритма симплекс-метода удобно оформлять в виде последовательности таблиц. В различ­ных источниках приводятся разные модификации симплекс-таблиц, отличающиеся друг от друга расположением отдель­ных элементов. Однако это не должно вызывать смущения у читателя, так как все они базируются на одних и тех же прин­ципах. Остановимся на структуре таблицы, показанной на рис. 1.5.\*

\* Настоящая структура симплекс-таблиц строится на идеях и прин­ципах их организации, предложенных в [1].

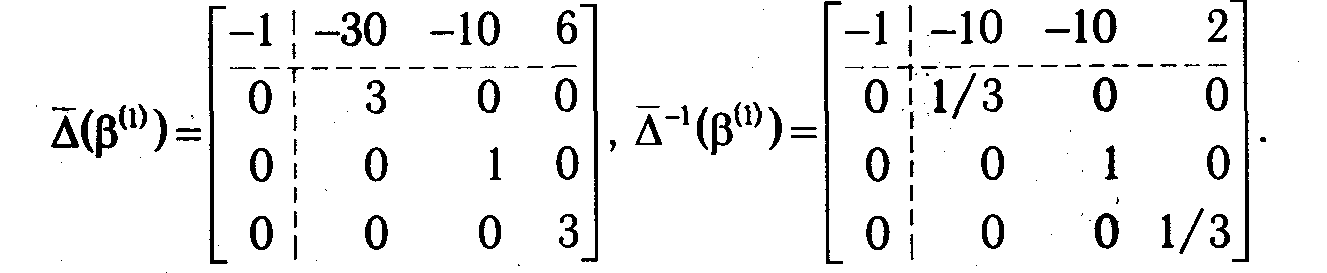
Симплекс-таблица Т(*q*), изображенная на *рис. 1.5*, соответ­ствует допустимому базису КЗЛП β(*q*)), получаемому на *q*-й ите­рации. Столбец *N*(β(*q*)) содержит номера базисных столбцов (в той последовательности, в которой они входят в базис), стол­бец *b*(β(*q*)) —компоненты вектора ограничений относительно текущего базиса β(*q*), *A*(β(*q*)) — компоненты матрицы задачи относительно текущего базиса β(*q*). Наконец, в строке *а*0(β(*q*)) находятся текущие оценки столбцов, а ячейка *b*0(β(*q*)) содержит значение целевой функции, достигаемое для текущего плана.

Безусловно, следует добавить, что табличная модификация симплекс-метода имеет важное практическое значение не столько как удобная форма организации ручного счета, сколь­ко как основа для реализации данного алгоритма в рамках про­граммного обеспечения ЭВМ.

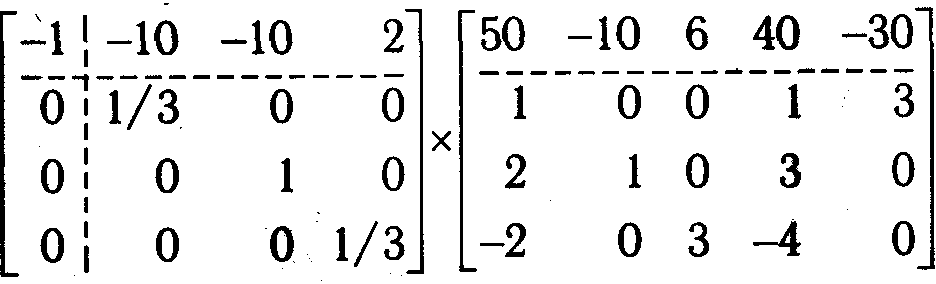
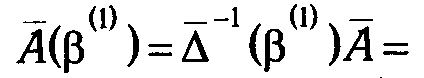
**1.4.3. Пример решения ЗЛП симплекс-методом**. Рас­смотрим на конкретном примере процесс решения КЗЛП таблич­ным симплекс-методом. Пусть дана каноническая задача ЛП:



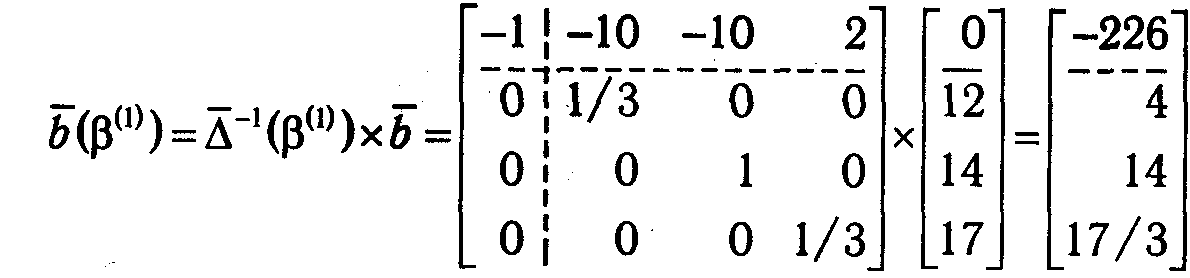
Как видно, столбцы матрицы с номерами {5, 2, 3} являются линейно независимыми. И можно получить разложение по дан­ным столбцам вектора ограничений с положительными коэф­фициентами. Последнее означает, что столбцы {5, 2, 3} образу­ют допустимый базис, с которого можно начать решение задачи. Из столбцов, входящих в базис, с учетом нулевых эле­ментов формируется матрица Δ(β(1)) и обратная по отношению к ней Δ-1(β(1)):



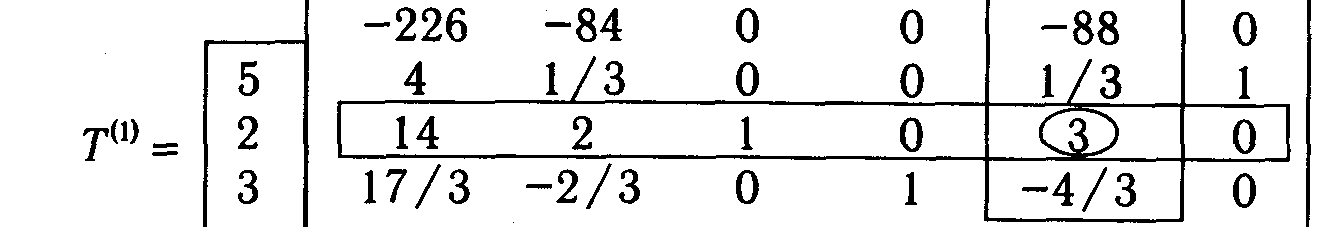
Используя их, по формуле (1.26) получаем



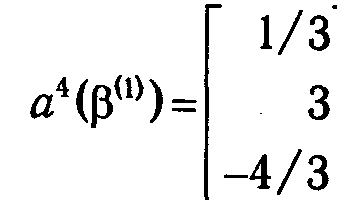
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | -84 | 0 | 0 | -88 | 0 |  |
|  | 1/3 | 0 | 0 | 1/3 | 1 |  |
| = | 2 | 1 | 0 | 3 | 0 | ; |
|  | -2/3 | 0 | 1 | -4/3 | 0 |  |



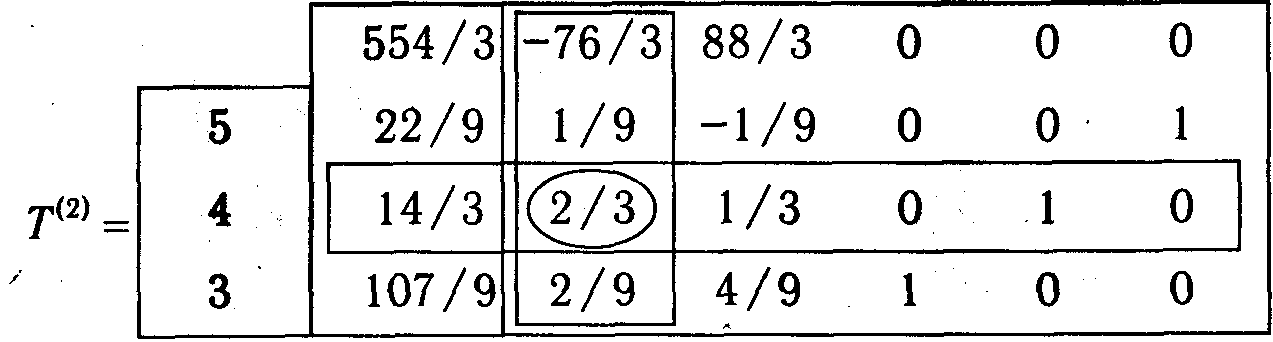
Используя полученные значения *A*(β(1)) и *b*(β(1)), заполняем симплекс-таблицу *T*(1), соответствующую первой итерации (*q*=1).



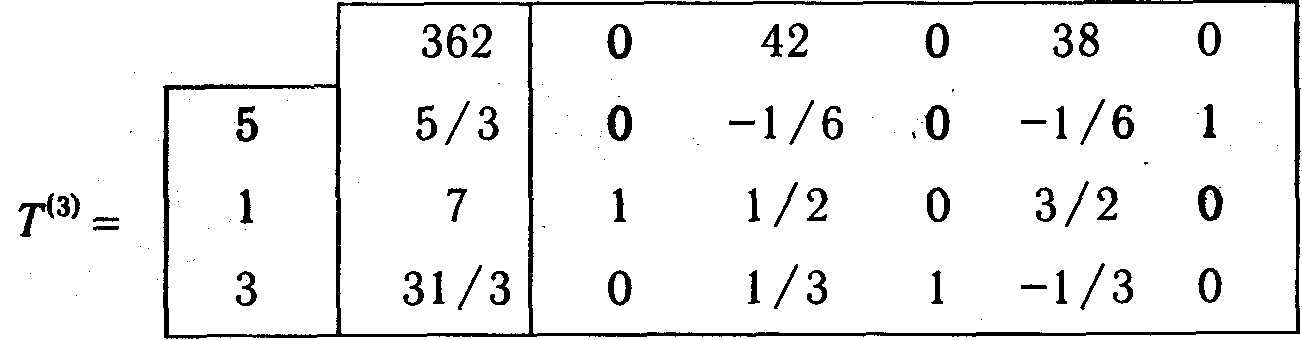
Поскольку строка оценок *a*0(β(1)) в первом и четвертом столбцах содержит отрицательные элементы (*a*0,1(β(1)) = -84, *a*0,4(β(1)) = -88), план *х*(β(1)) = (0,14,17/3,0,4) не является оптимальным, и значение целевой функции *f*(*x*(β(1))) = -226 может быть улучшено. Следуя рекомендации п.1˝ алгоритма симплекс-метода, полагаем номер вводимого в очередной базис столбца *l* = 4 (т.к. |-88| > |-84|). Рассматриваем *ведущий* столбец (выделен пунктирной рамкой)



Он содержит два положительных элемента. Применяя рекомен­дацию п. 2" алгоритма, получаем λ1= 4/(1/3) =12, λ2=14/3, и, стало быть *r* = 2. Следовательно, столбец с номером *N*2(β(1)) = 2 должен быть выведен из базиса. Таким образом, по­лучаем очередной допустимый базис задачи *N*(β(2)) = {5, 4, 3}. Элемент *a*2,3(β(1)) является ведущим (обведен кружком). При­менив формулы (1.28)-(1.31), переходим к симплекс-таблице, соответствующей второй итерации *T*(2), и полагаем индекс те­кущей итерации *q* = 2.



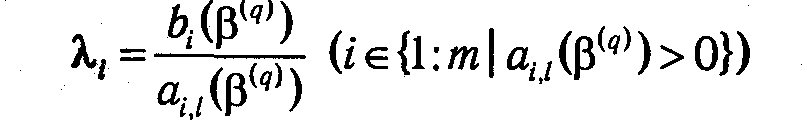
В ходе выполнения второй итерации опять-таки определяют­ся вводимый столбец *а*1, выводимый *а*4 и ведущий элемент *a*2,1(β(2)). Перейдя к итерации *q* = 3, имеем таблицу *T*(3).



Как видно из *T*(3), строка оценок содержит только неотрицатель­ные значения, поэтому достигнутый базис *N*(β(3)) = {5, 1, 3} яв­ляется оптимальным. В итоге мы получаем, что вектор *х*\* = (7, 0, 31/3, 0, 5/3) является оптимальным планом (точ­кой максимума) задачи (1.34)-(1.35), максимальное значение целевой функции равно *f*\* =*f*(*х*\*)=362, а решение КЗЛП имеет вид ((7, 0, 31/3, 0, 5/3); 362).

**1.4.4. Сходимость симплекс-метода. Вырожденность в задачах ЛП**. Важнейшим свойством любого вычислительного, алгоритма является *сходимость*, т. е. возможность получения в ходе его применения искомых результатов (с заданной точно­стью) за конечное число шагов (итераций).

Легко заметить, что проблемы со сходимостью симплекс-ме­тода потенциально могут возникнуть на этапе выбора значения *r* (п. 2") в случае, когда одинаковые минимальные значения от­ношения



будут достигнуты для нескольких строк таблицы Т (*q*) одновре­менно. Тогда на следующей итерации столбец *b*(β(*q*+1)) будет со­держать нулевые элементы. Напомним, что

* *допустимый базисный план канонической задачи ЛП, со­ответствующий базису* β(*q*), *называется* **вырожденным***, если некоторые его базисные компоненты равны нулю, т. е. вектор b*(β(*q*)) *содержит нулевые элементы.*

Задача ЛП, имеющая вырожденные планы, называется ***вы­рожденной***. При «выходе» на вырожденный план мы фактиче­ски получаем разложение столбца *b* по системе из меньшего, чем *m*, количества столбцов *аj* и, следовательно, лишаемся воз­можности корректно определить, ввод какого столбца в базис приведет к росту значения целевой функции. Подобные ситуа­ции, в конечном счете, могут привести к зацикливанию вычис­лительного процесса, т. е. бесконечному перебору одних и тех же базисов.

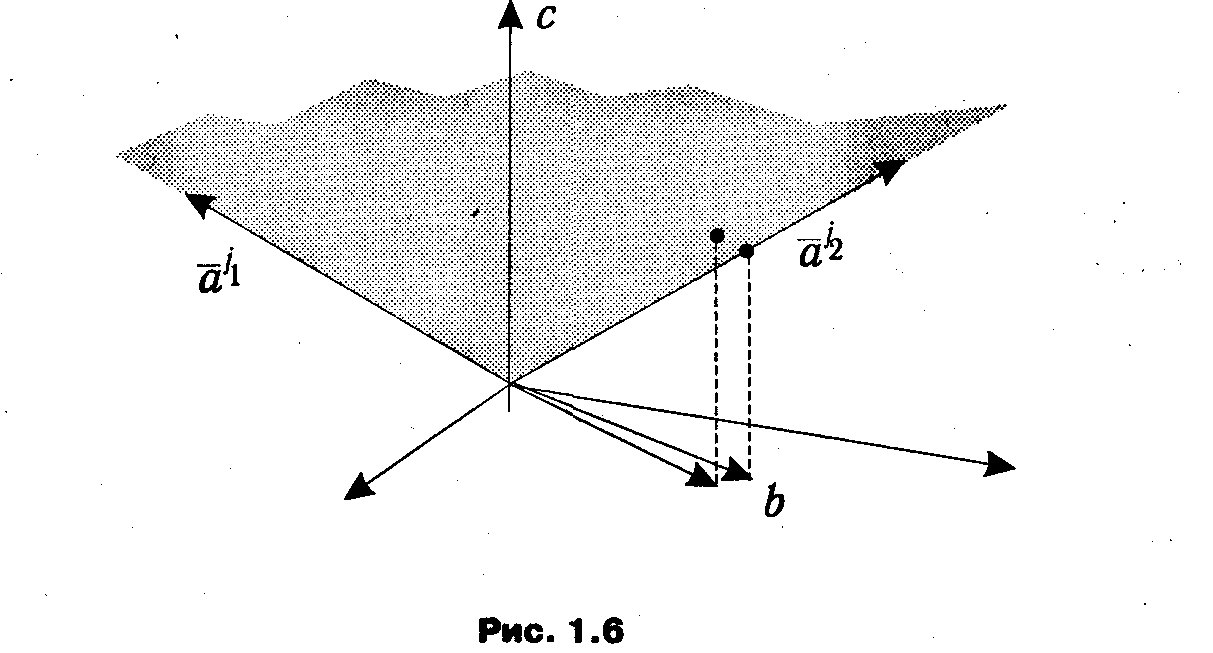
С точки зрения первой геометрической интерпретации ЗЛП ситуация вырожденности означает, что через некоторую угло­вую точку многогранного множества допустимых планов зада­чи, соответствующую текущему базисному плану, проходит более чем *m* гиперплоскостей ограничений задачи. Иными сло­вами, одно или несколько ограничений в данной точке являют­ся избыточными. Последнее предоставляет повод для размыш­лений об экономической стороне проблемы вырожденности как проблемы наличия избыточных ограничений и в некоторых случаях определяет пути ее решения.

С точки зрения второй геометрической интерпретации ЗЛП вырожденность означает «попадание» линии, проходящей через вершину вектора *b* параллельно оси аппликат, в ребро кону­са, натянутого на систему расширенных векторов текущего базиса. Так, в случае, изображенном на *рис. 1.6*, эта линия по­падает в ребро, определяемое вектором *аj*2 , т. е., несмотря на то что текущий базис содержит столбцы *аj*1 и *аj*2, для представле­ния вектора *b* достаточно только одного *аj*2.

Из данной интерпретации вытекает и идея метода решения вырожденных задач ЛП, получившего названия *метода возму­щений*: при выходе на вырожденный план осуществляется не­значительный сдвиг вектора *b* и вырожденность (как это видно из иллюстрации) устраняется.

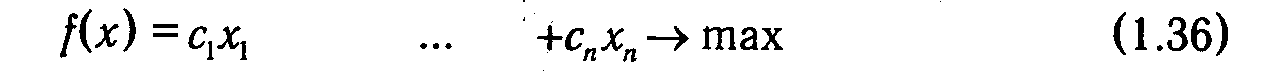
Необходимо, однако, сказать, что рассмотренная здесь про­блема зацикливания для большинства практически значимых задач является достаточно редкой и маловероятной. Более того, она очень часто разрешается за счет ошибок округлений при выполнении расчетов на ЭВМ.

**1.4.5. Нахождение допустимого базисного плана**. В рас­смотренном выше примере исходный базисный план, необходи­мый для начала вычислений по симплекс-методу, был подобран за счет особенностей матрицы условий. Действительно, данная матрица уже содержала необходимое количество «почти базис­ных» столбцов. Очевидно, что для подавляющего большинства задач ЛП невозможно подобным образом сразу и в явном виде указать исходный допустимый базисный план. Вообще говоря, существуют различные приемы решения данной задачи. Мы остановимся на одном из них, получившем название *метода минимизации невязок*. Его сильной стороной, безусловно, яв­ляется универсальность. Xотя, в некоторых частных случаях, он может оказаться слишком громоздким.

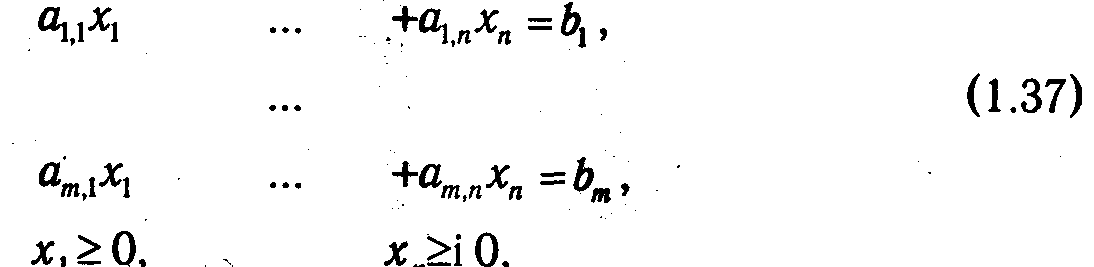


Идея метода минимизации невязок состоит в построении со­ответствующей вспомогательной задачи, для которой можно в явном виде указать исходный базисный план, и решении ее с помощью процедуры симплекс-метода.

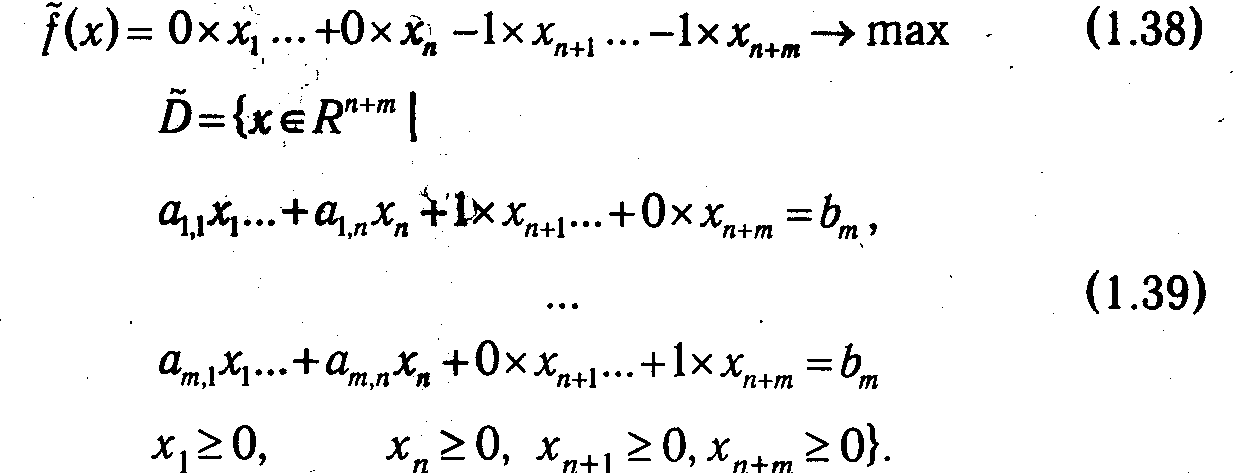
Не умаляя общности, можно считать, что вектор ограниче­ний в исходной задаче неотрицателен, т. е. *bi* ≥ 0, *i* ∈ 1: *m*. Тогда для КЗЛП максимизации функции



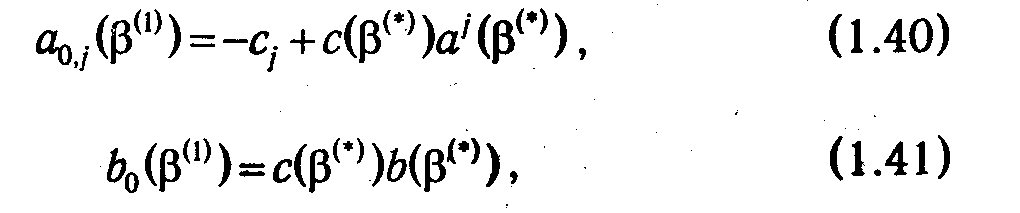
на множестве, определяемом ограничениями



строится вспомогательная задача



Как видно из (1.36)-(1.39), задача (*,*) отличается от за­дачи (*D, f*) матрицей, получаемой в результате добавления к исходной матрице *А* единичной матрицы, которой соответ­ствуют дополнительные (фиктивные) переменные *хn*+1,...,*хn+m*. Данным переменным (собственно говоря, их и называют *не­вязками*) в целевой функции ** отвечают коэффициенты (-1), а переменным *х*1,...,*хn*, которые являются общими для обеих задач, — нулевые коэффициенты. Существенной особенно­стью (*,*) является наличие в ней очевидного исходного ба­зиса, образуемого добавленными столбцами, и соответствую­щего плана с базисными компонентами х*n+i* = *bi* ≥ 0, *i* ∈ 1: *m*. В силу структуры целевой функции *f̃* в процесс поиска ее мак­симума процедурой симплекс-метода фиктивные переменные (невязки) х*n+i* должны минимизироваться желательно до нуля, что может быть достигнуто за счет последовательного перевода их в небазисные компоненты плана. Если в оптималь­ном плане *х̃*, полученном в результате решения вспомогатель­ной задачи, последние *m* компонент (т. е. невязки) равны нулю, то его начальные *n* компонент удовлетворяют системе ограни­чений, определяющих область *D*, и ** легко (простым отбра­сыванием невязок) может быть преобразован в стартовый до­пустимый план основной задачи (*D, f*). При этом все строки симплекс-таблицы, полученной на последней итерации вспо­могательной задачи (за исключением нулевой), могут быть перенесены в первую таблицу основной задачи. Элементы же нулевой строки для вновь формируемой симплекс-таблицы вы­числяются по формулам:

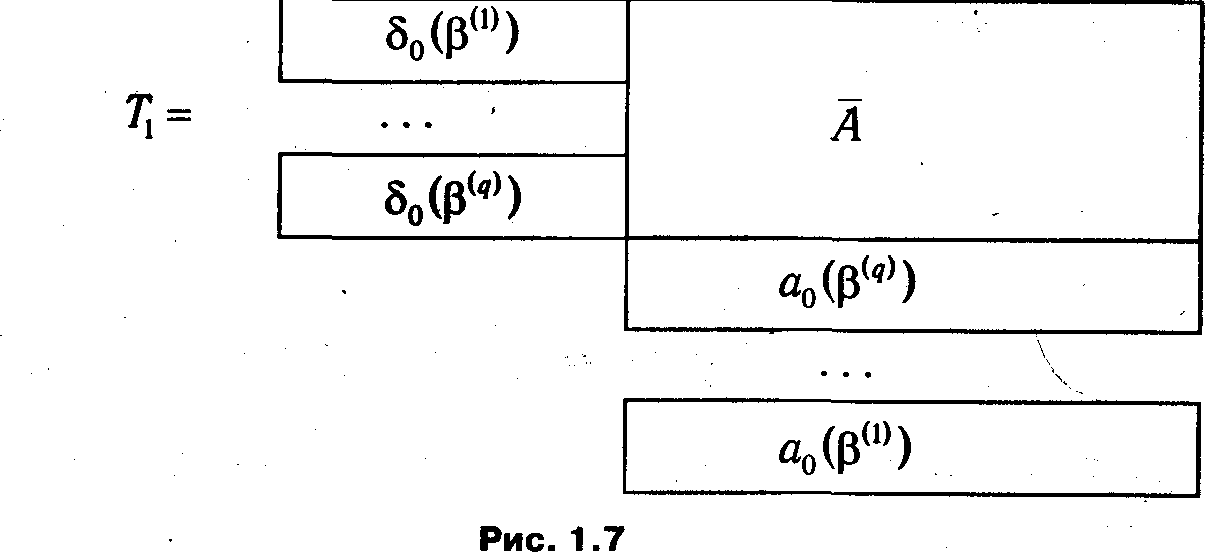


где β(\*) — базис, полученный на последней итерации вспомога­тельной задачи.

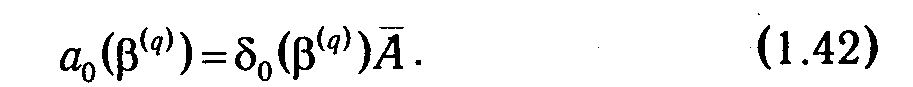
В случае, когда оптимальный план вспомогательной задачи *х̃* все же содержит не равные нулю фиктивные компоненты (т. е. **(**)<0), мы приходим к заключению об *отсутствии до­пустимых планов* у исходной задачи (*D, f*), т. е. *D* = Ø.

## 1.5. МОДИФИЦИРОВАННЫЙ СИМПЛЕКС-МЕТОД

**1.5.1. Вычислительная схема, основанная на преобра­зовании обратных матриц.** Анализируя вычислительную процедуру симплекс-метода с позиций оценки трудоемкости, не­трудно заметить, что наиболее критичным в этом плане являет­ся этап пересчета значений *А* и *b* при переходе от одного базис­ного плана к другому (п. 3 алгоритма). Однако в том случае, когда число ограничений задачи *m* явно меньше количества переменных *n*, можно добиться существенной «экономии», вы­полняя на очередной итерации *q* преобразование Жордана—Га­усса не над матрицей *А*(β(*q*)), а над матрицей Δ-1(β(*q*)). При этом учитывается и то, что при необходимости, применяя формулу (1.26), всегда можно получить *А*(β(*q*)) по Δ-1(β(*q*)). Более того, для выполнения описанных выше действий симплекс-процедуры нам в действительности не требовалась матрица *А*(β(*q*)) целиком. Реаль­но в ней использовались только строка оценок *a*0(β(*q*)) и ведущий столбец *al*(β(*q*)). Данные соображения положены в основу вы­числительной схемы симплекс-метода, основанной на преобра­зовании обратных матриц, которую также называют *модифици­рованным симплекс-методом*. Впервые данный алгоритм был предложен в 1951 г. в работах Л. В. Канторовича.



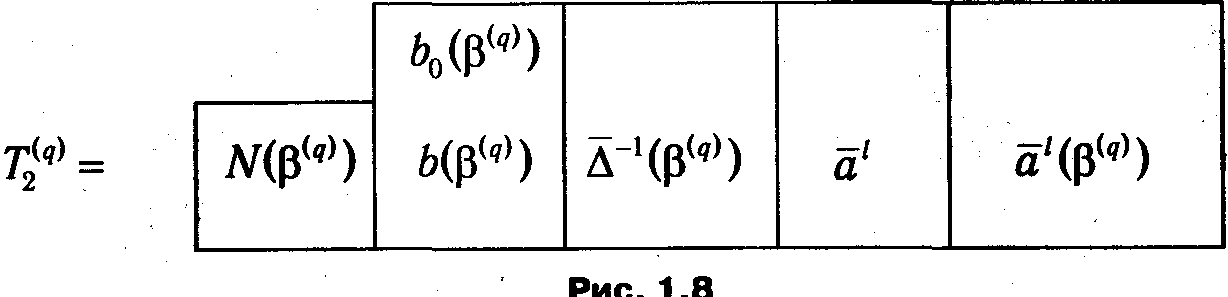
Вычислительной схеме модифицированного симплекс-мето­да соответствует система таблиц *T*1 и *Т*2(*q*). Таблица *T*1 (*рис. 1.7*) является общей для всех итераций и служит для получения строки оценок текущего базисного плана *a*0(β(*q*)). Если обозна­чить через δ*i*(β(*q*)) (*i* ∈ 0 : *m*) строки матрицы Δ-1(β(*q*)), то из (1.26), в частности, следует, что



Как видно из *рис. 1.7*, *T*1 состоит из трех блоков:

* *в центре содержится матрица* *А*;
* *в левом блоке таблицы на каждой итерации дописывают­ся нулевые строки матрицы* Δ-1(β(*q*)) *для текущего ба­зиса*;
* *нижний блок, расположенный под матрицей* *А*, *на каж­дой итерации дополняется строкой оценок текущего плана, вычисленной по формуле (1.42).*

Симплекс-таблица *Т*2(*q*), изображенная на *рис. 1.8*, соответ­ствует допустимому базису КЗЛП β(*q*), получаемому на *q*-й ите­рации. Столбец *N*(β(*q*)) содержит номера базисных столбцов (в последовательности вхождения в базис); столбец *b*(β(*q*)) — компоненты вектора ограничений относительно текущего ба­зиса β(*q*); Δ-1(β(*q*)) — матрица, обратная по отношению к матри­це расширенных столбцов текущего базиса β(*q*); графа *аl* со­держит расширенный вектор условий, вводимый в базис на текущей итерации, а следующая графа — координаты *аl*(β(*q*)) этого же столбца в текущем базисе β(*q*).



По аналогии с п. 1.4.1 опишем формальную схему алгоритма модифицированного симплекс-метода.

**0-этап. Нахождение допустимого базисного плана.**

1. Для поиска допустимого базиса может быть применен ме­тод минимизации невязок, рассмотренный в п. 1.4.5. При этом для решения вспомогательной задачи используется процедура модифицированного симплекс-метода. В результате 0-этапа мы получаем допустимый базисный план *x*(β(1)) и соответствую­щие ему матрицу Δ-1(β(1)) и вектор *b*(β(1)).

2. Заполняем центральную часть таблицы *T*1, содержащую матрицу *А*.

3. Содержимое матрицы Δ-1(β(1)) и вектора *b*(β(1)), получен­ных на этапе поиска допустимого базисного плана, переносит­ся в таблицу *T*2(1).

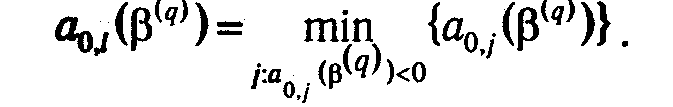
4. Полагаем номер текущей итерации *q* равным 1 и перехо­дим к I-этапу.

**1-этап. Стандартная итерация алгоритма** — выполня­ется для очередного базисного плана *x*(β(*q*)).

1°. *Проверка оптимальности текущего базисного плана*. Содержимое нулевой строки таблицы *T*2(*q*) — δ0(β(*q*)) переписы­вается в соответствующую графу таблицы *T*1. По формуле (1.42) рассчитываем и заполняем строку *a*0(β(*q*)). Осуществля­ем просмотр строки оценок *a*0(β(*q*)). Возможны два варианта:

1΄. *a*0(β(*q*))≥0 —план, соответствующий текущему базису задачи, **оптимален.** Вычислительный процесс закончен. Со­гласно формулам (1.33) и (1.32) выписываются оптимальный план задачи *х*\* = *x*(β(*q*)) и оптимальное значение целевой функ­ции *f*(*х*\*) = *f*(*x*(β(*q*))).

1". В строке оценок *a*0(β(*q*)) существует по меньшей мере один элемент *a*0,*j*(β(*q*))<0, т. е. имеющий отрицательную оцен­ку. Следовательно, план *x*(β(*q*)) — **неоптимален**. Выбирает­ся номер *l*, соответствующий элементу, имеющему минималь­ную (максимальную по абсолютной величине) отрицательную оценку:



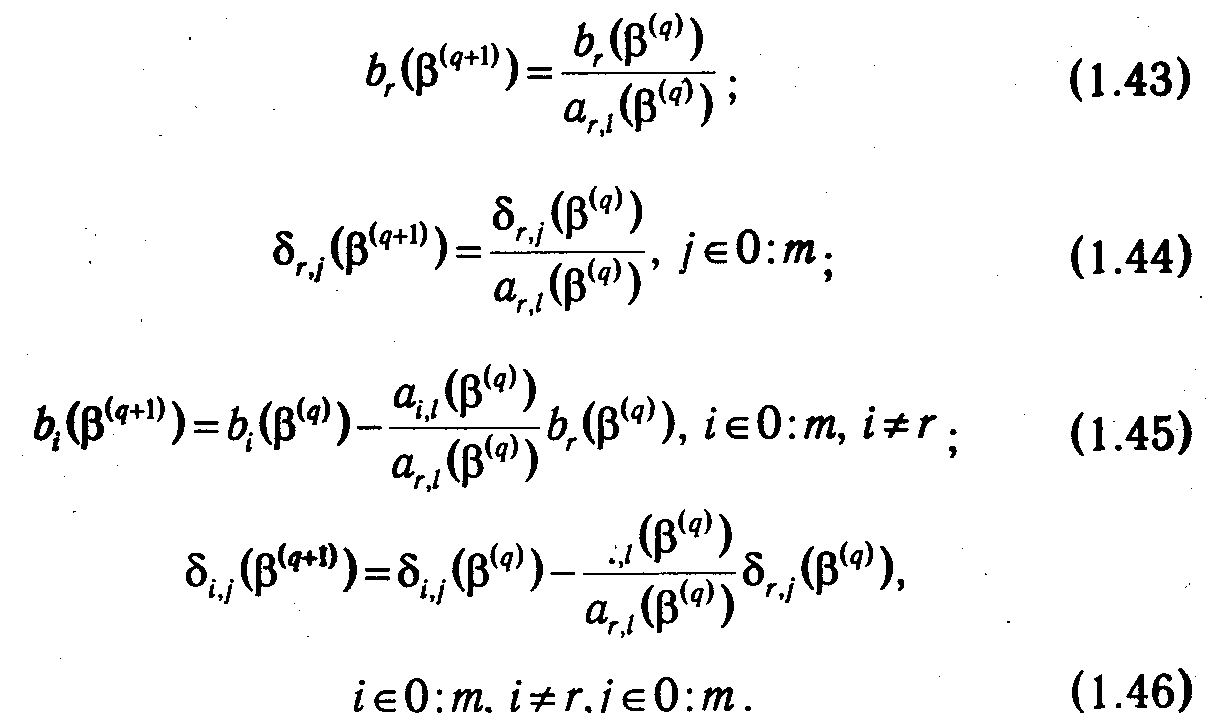
*l*-й столбец матрицы *A* становится *ведущим* и должен быть вве­ден в очередной базис. Переходим к пункту 2° алгоритма.

2°. *Определение столбца, выводимого из базиса*. Перепи­сываем ведущий столбец *аl* из таблицы *T*1 в текущую таблицу *Т*2(*q*). По формуле *аl*(β(*q*)) = Δ-1(β(*q*))*аl* заполняем соответствую­щий столбец в таблице *Т*2(*q*). Возможны два варианта:

2'. Для всех *i* ∈ 1: *m* *аi,l*(β(*q*))≤0. Делается вывод о *неограни­ченности целевой функции* и завершается вычислительный процесс.

2". Существует по крайней мере один индекс *i* ∈ 1: *m*, для ко­торого *аi,l*(β(*q*))>0. Согласно правилу (1.27) определяются мес­то *r* и номер *Nr*(β(*q*)) для столбца, выводимого из базиса. Пере­ходим к пункту 3° алгоритма.

3°. *Пересчет относительно нового базиса элементов столбца b* и *матрицы* Δ-1. Переход к новому базису β(*q*+1), кото­рый получается введением в базис β(*q*) столбца *аl* и выводом из него столбца *аr*, осуществляется по формулам, аналогичным формулам (1.28)-(1.31). Они имеют вид:

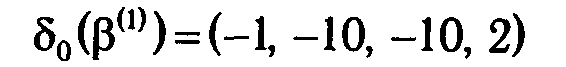


Полагаем номер текущей итерации *q*: =*q*+l и переходим к первому пункту алгоритма.

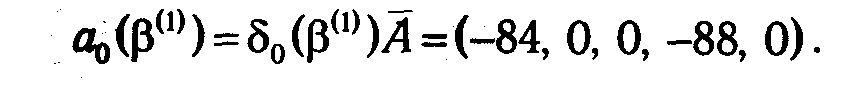
В завершение подчеркнем, что в силу приведенных выше пре­имуществ именно модифицированный симплекс-метод реально применяется в программном обеспечении, предназначенном для решения канонических задач линейного программирования.

**1.5.2. Пример решения ЗЛП модифицированным сим­плекс-методом.** Приведем решение рассмотренной ранее за­дачи (1.34)-(1.35), основанное на использовании процедуры модифицированного симплекс-метода. По аналогии с п. 1.4.3 оно начинается с выбора очевидного исходного базиса, обра­зуемого столбцами {5,2,3}. Для него уже были вычислены Δ-1(β(*q*)) и *b*(β(*q*)), поэтому заполнение начальных таблиц *T*1 и *Т*2(1) не представляет труда.

На первой итерации мы переписываем нулевую строку

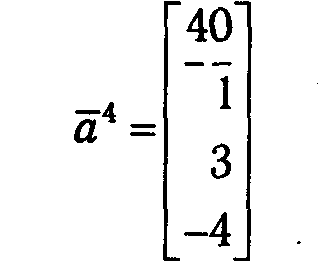


из *Т*2(1) в *T*1 и, умножив ее на матрицу *A*, получаем строку оце­нок

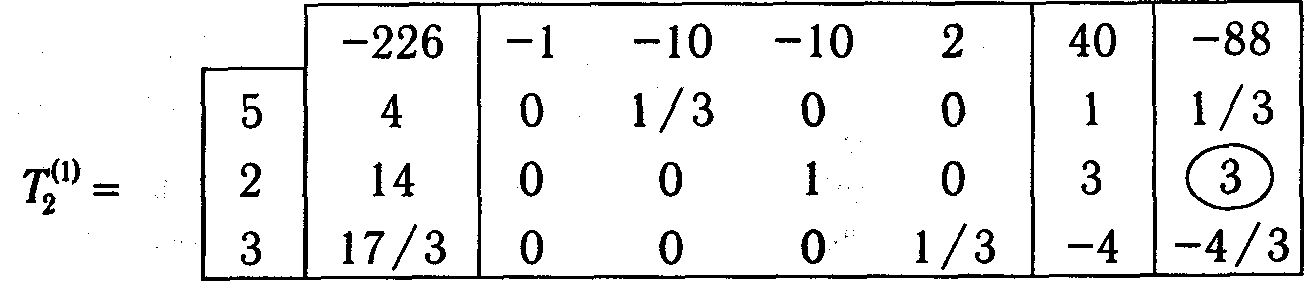
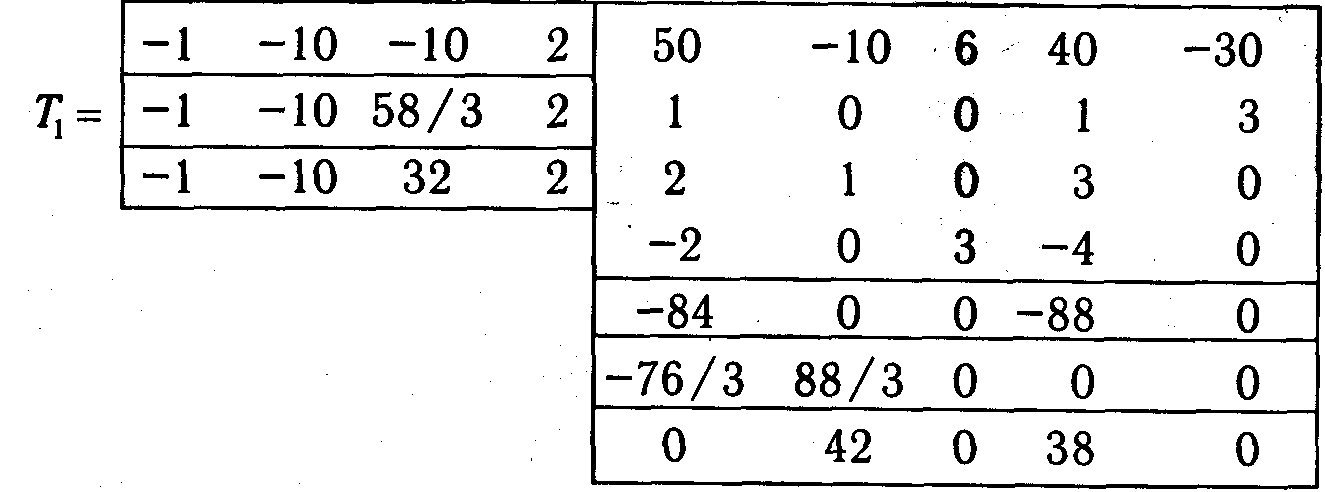


Так как *a*0(β(1)) содержит отрицательные элементы, то делаем вывод о неоптимальности плана, соответствующего базису β(1), и, выбрав наименьшую отрицательную оценку (-88), получаем номер столбца, вводимого в базис, *l* = 4.

Переписываем столбец

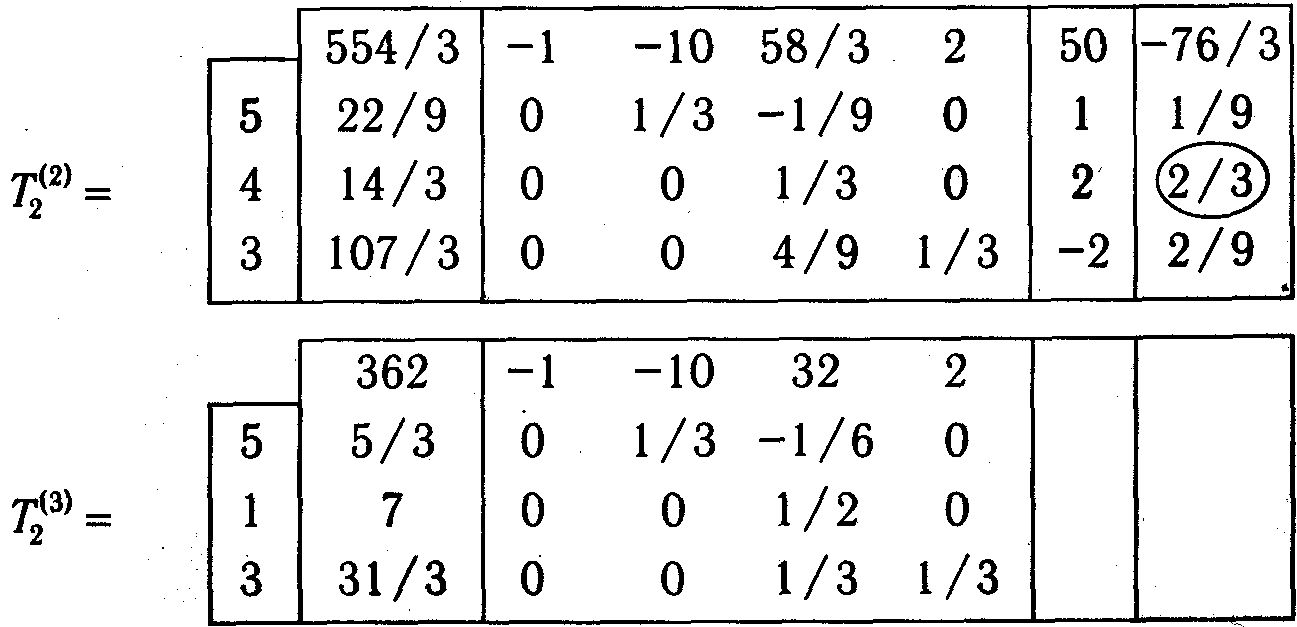


из таблицы *T*1 в *Т*2(1) и пересчитываем его координаты относи­тельно текущего базиса, т. е. умножаем матрицу Δ-1(β(*q*)), рас­положенную в таблице *Т*2(1) слева, на *а*4.



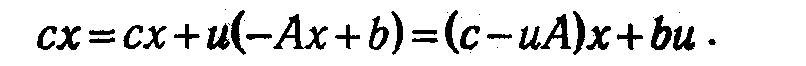
После заполнения таблицы *Т*2(1) данными по вводимому в но­вый базис столбцу можно перейти к определению номера выво­димого столбца. Эта процедура осуществляется в полной ана­логии с обычным симплекс-методом. Рассмотрев отношения элементов *bi*(β(1)) и *ai,l*(β(1)) для {*i*∈1:*m| ai,l*(β(1))>0} и определив минимальное из них, находим, что *r* = 2. Следовательно, стол­бец с номером *N*2(β(*q*)) = 2 должен быть выведен из базиса. Та­ким образом, получаем очередной допустимый базис задачи с *N*(β(2)) = {5, 4, 3}. Элемент *a*2,3(β(1)) является ведущим (обведен кружком). Применив формулы (1.43)-(1.46), переходим к сим­плекс-таблице, соответствующей второй итерации *Т*2(2), и пола­гаем индекс текущей итерации *q* = 2.

Повторяя те же самые действия (их легко проследить по при­водимым здесь таблицам *Т*2(2) и *Т*2(3), на третьей итерации мы получим оптимальный план задачи и оптимальное значение целевой функции, которые извлекаются из второго столбца таблицы *Т*2(3). Легко заметить, что в процессе решения мы «про­шли» по той же самой последовательности допустимых базис­ных планов, которая встречалась в п. 1.4.3.



## 1.6. ТЕОРИЯ ДВОЙСТВЕННОСТИ В ЛИНЕЙНОМ ПРОГРАММИРОВАНИИ

**1.6.1. Понятие двойственной задачи ЛП**. Пусть задана КЗЛП (1.7). Если целевая функция *f*(*x*) = *cx* достигает макси­мального значения на множестве *D*, то обоснованным представ­ляется вопрос о том, каким образом можно построить верхнюю оценку для нее. Очевидно, что если через *и* обозначить некото­рый *m*-мерный вектор, то



Предположим, что *и* можно выбрать таким образом, чтобы *иА ≥* *с*. Тогда при любых *х* ≥ 0 справедливо неравенство



Стремясь получить наилучшую оценку (1.47), мы приходим к формулировке некоторой новой кстремальной задачи, которая в некотором смысле логически сопряжена с задачей (1.7) называется *двойственной*. Оговоримся, что приведенные рассуждения не носят строгого характера и предназначены только для того, чтобы подготовить читателя к приводимому ниже формальному определению двойственной задачи линейного прoграммирования.

☞ *Если задана каноническая задача ЛП*



*то задача ЛП*



*называется* **двойственной** *по отношению к ней. Соот­ветственно, задача* (*D, f*) *no отношению к*

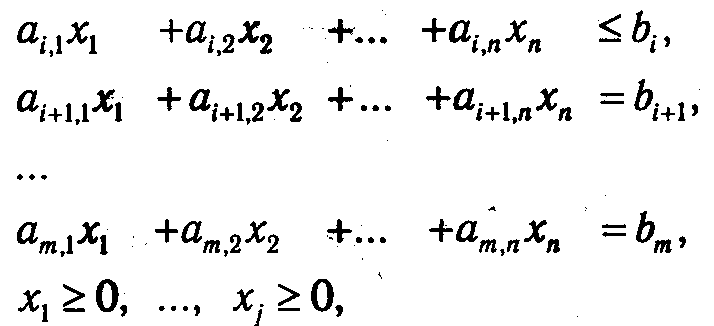
(*D\*,f*\*) *назы­вается* **прямой** *(или исходной).*

**1.6.2. Общая схема построения двойственной задачи.** Приведенное выше определение задачи, двойственной по отношению к канонической ЗЛП, может быть распространено на случай общей задачи линейного программирования.

☞ *Если задана общая задача ЛП*

*f*(*x*) = *c*1*x*1 + ... + *сjхj + сj*+1*хj+*1 *+ ...* + *сnxn →* max, *x* ∈ *D,* (1.50)

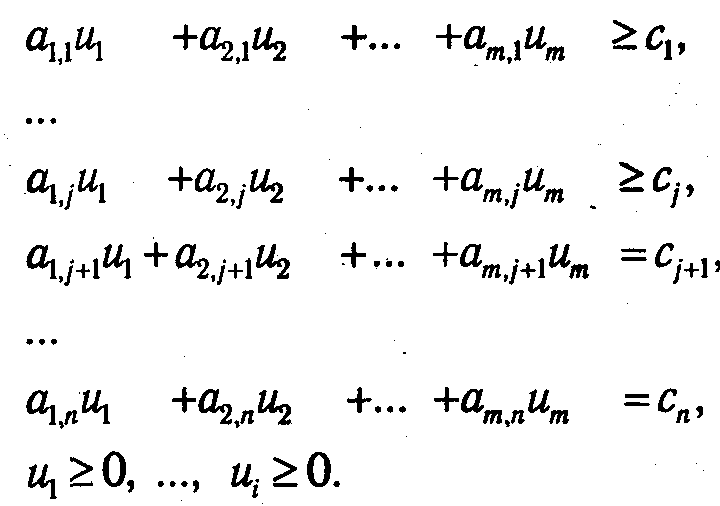
*где D определяется системой уравнений и неравенств:*



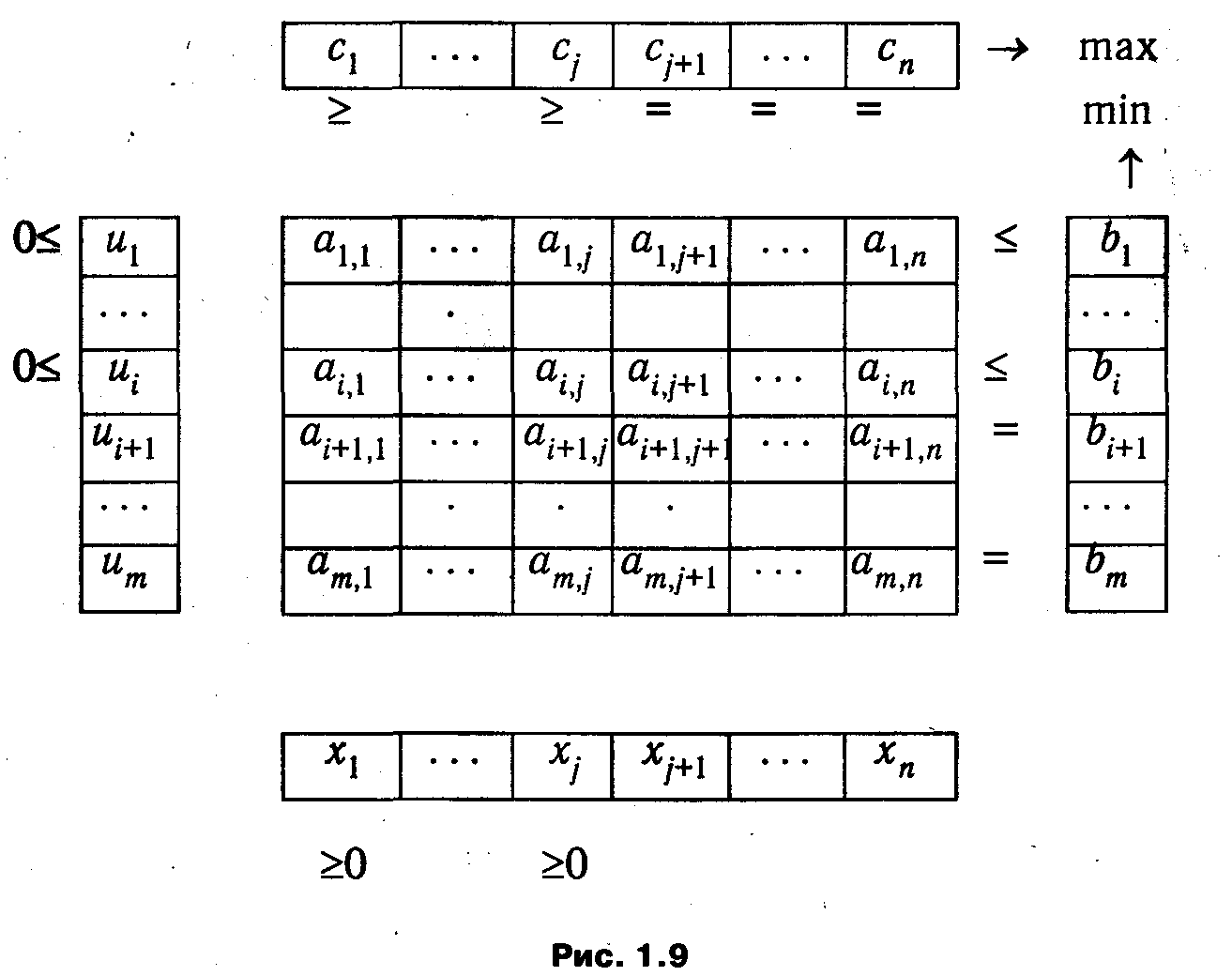
*то* **двойственной** *по отношению к ней называется об­щая задача ЛП*



*где D\* определяется системой уравнений и неравенств:*



Правила построения задачи, двойственной по отношению к ОЗЛП, наглядно представлены схемой, показанной на *рис. 1.9.*



Как следует из приведенной схемы при переходе от прямой задачи ЛП к двойственной:

1. Тип оптимума меняется на противоположный, т. е. макси­мум на минимум, и наоборот.

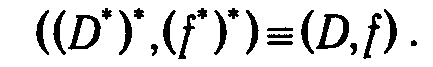
2. Вектор коэффициентов целевой функции *с* и столбец огра­ничений *b* меняются местами.

3. Матрица ограничений задачи *A* транспонируется.

4. Множество индексов переменных, на которые наложено условие неотрицательности в прямой задаче (например, *хj* ≥ 0 или *uj* ≥ 0), определяют номера ограничений, имеющих форму неравенств в двойственной задаче (*aju* ≥ *сj* или *aix* ≤ *bj*).

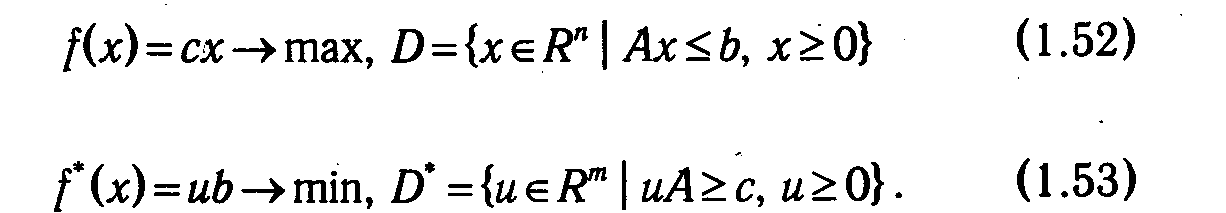
5. Множество номеров ограничений, имеющих форму нера­венств в прямой задаче (например, *aix* ≤ *bj* или *aju* ≥ *сj)*, опреде­ляют множество индексов переменных, на которые накладыва­ется условие неотрицательности, в двойственной задаче (*ui* ≥ 0 или *xi* ≥ 0).

* *Из приведенного определения вытекает важное свой­ство — симметричность отношения двойственности, т. е. задача, двойственная по отношению к двойственной, со­впадает с прямой (исходной) задачей*:

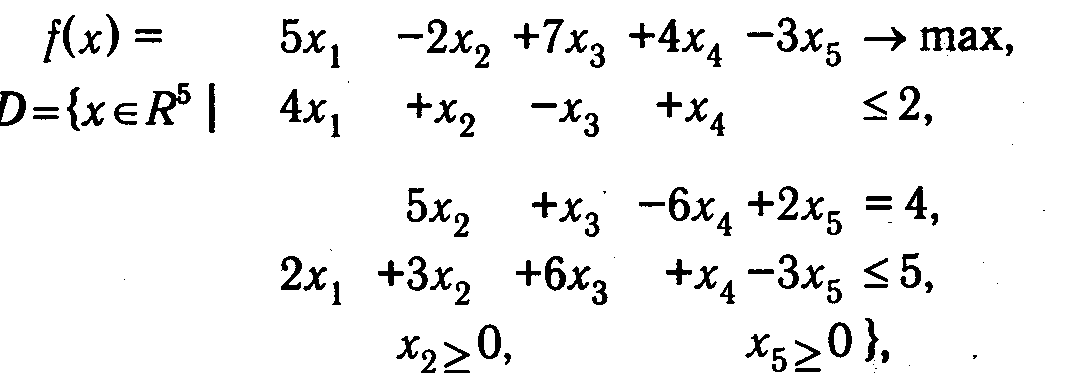


*Тем самым имеет смысл говорить о паре взаимно двой­ственных задач.*

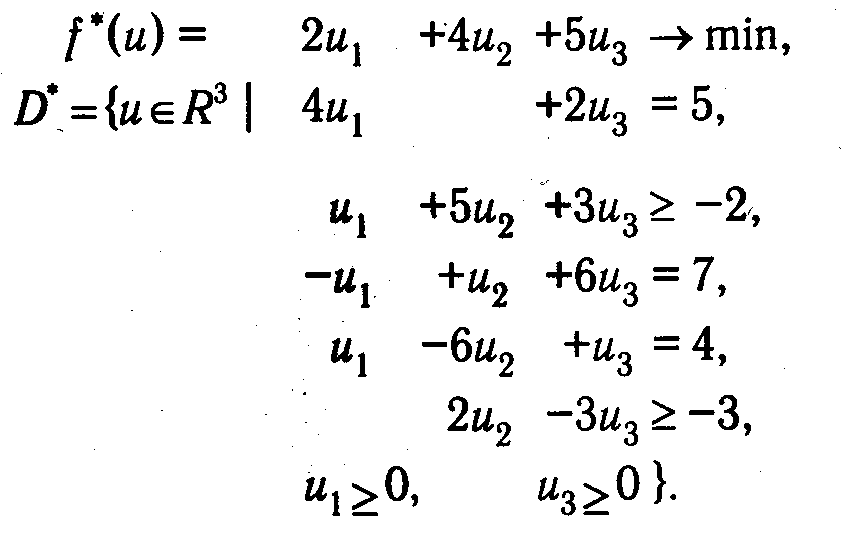
В матричной форме пара двойственных общих задач линей­ного программирования может быть кратко записана как:



Рассмотрим процесс построения двойственной задачи на конкретном примере. Пусть задана ОЗЛП (*D*, *f*):



тогда двойственной к ней будет задача (*D\**, *f\**):

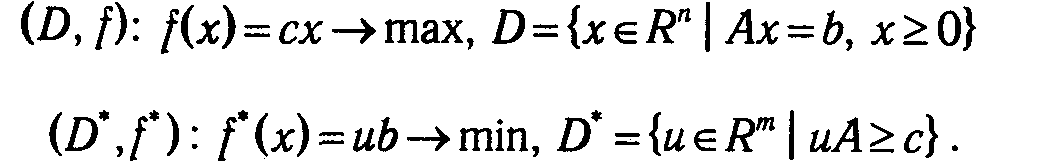


**1.6.3. Теоремы двойственности и их применение**. Фун­даментальные свойства, которыми обладают двойственные за­дачи линейного программирования, могут быть сформулирова­ны в виде приводимых ниже утверждений. Их обычно называют *теоремами двойственности*.

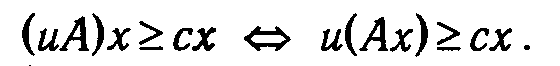
|  |
| --- |
| *Теорема 1.4*. *Если х, и— допустимые планы для пары двойственных задач* (*D,f*) и (*D\*,f\**), *тo* *f*(*x*) ≤ *f*(*u*). |

**Доказательство.**

Достаточно доказать теорему для случая, когда задача (*D*, *f*) является канонической. Рассмотрим пару двойственных задач



Из того, что вектор *и* является допустимым планом задачи (*D*\*, *f*\*), следует, что *иА* ≥ *с*. Умножив левую и правую части дан­ного неравенства на вектор *х* ≥ 0 , получим равносильную сис­тему неравенств



Одновременно для вектора х, являющегося допустимым планом задачи (*D, f*), справедливо равенство *Ax=b*. Тем самым доказа­но, что *иb ≥* *сх.*✌

**Замечание.** Теорема 1.4, разумеется, верна и для оптималь­ных планов взаимно двойственных задач: *f(x\*) ≤ f\*(u\*),* где *х\** и *u\*—*любые оптимальные планы задач (*D, f*) и (*D\*,f*\*). На самом деле, как будет видно из дальнейшего, справедливо равенство *f(x\*) = f\*(u\*).*

|  |
| --- |
| ***Теорема 1.5.*** *Если для некоторых допустимых планов  и  взаимно двойственных задач* (*D, f*) *и* (*D*\*,*f*\*) *вы­полняется равенство f()=f\*(), то  и  являются оптимальными планами для этих задач.* |

**Доказательство.**

Согласно теореме 1.4, для всех допустимых планов *х* задачи (*D, f*) справедливо неравенство *сх* < *b*. По условию теоремы *f()=f()* или, что то же самое, *с = b*. Следовательно, верно утверждение: для любого x ∈ D *с>сх*, т. е. *х* является опти­мальным планом для задачи (*D, f*).

Рассуждения, доказывающие оптимальность плана ** для за­дачи (*D\**,*f*\*), проводятся аналогично. ✌

|  |
| --- |
| *Теорема 1.6.* *Если целевая функция f в задаче (D, f) не ограничена сверху, то двойственная к*  *ней задача (D\*,f\*) не имеет допустимых планов.* |

**Доказательство.**

Если предположить, что у двойственной задачи (*D*\*,*f*\*) су­ществует хотя бы один допустимый план *и̃*, то, согласно теоре­ме 1.4, для любого допустимого плана *х* задачи (*D, f*) справед­ливо неравенство *f(x)* ≤ *f*\*(**) <+∞. Последнее означает, что целевая функция *f* задачи (*D, f*) ограничена сверху. Поскольку это противоречит условию теоремы, предположение о сущест­вовании допустимых планов двойственной задачи (*D\*,f*\*) не­верно. ✌

Следующее утверждение, известное как *теорема равнове­сия*, используется при проверке оптимальности планов ЗЛП.

|  |
| --- |
| *Теорема 1.7. Пусть х\** и *u\* — оптимальные планы ка­нонической задачи (D, f) и двойственной по отноше­нию к ней задачи (D\*,f\*). Если j-я компонента плана х\* строго положительна (xj\*>0), то соответствующее j-e ограничение двойственной задачи выполняется как равенство: а*1,*ju*1\* *+...+аm,jum\*= сj; если же j-й компо­нент плана х\* имеет нулевое значение (хj\* =0), то j-e ограничение двойственной задачи выполняется как неравенство: а*1,*ju*1\* *+...+аm,jum\*≥ сj .* |

**Доказательство.**

Векторы *х\** и *и\*,* будучи допустимыми планами соответствую­щих задач, удовлетворяют условиям: *Ах\* = b*, *х\* >* 0 и *и\*А-с ≥ 0*. Найдем скалярное произведение



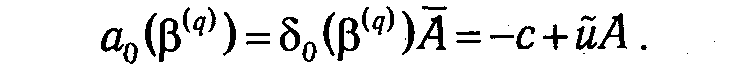
Согласно замечанию к теореме 1.2, оптимальные значения целевых функций взаимно двойственных задач совпадают, т. е*. u\*b=сх\*.* Последнее означает, что *(u\*А-с)х\** = 0 . Однако ска­лярное произведение двух неотрицательных векторов может быть равно нулю только в том случае, когда все попарные про­изведения их соответствующих координат равны нулю. Следо­вательно, если *xj\** > 0, то *u\*аj* – *сj* = 0, если же *xj* = 0, то возмож­но *u\*аj – сj* ≥ 0 , что и утверждается в теореме. ✌

Практическое значение теорем двойственности состоит в том, что они позволяют заменить процесс решения основной задачи на решение двойственной, которое в определенных случаях может оказаться более простым. Например, задача, область до­пустимых значений которой описывается двумя уравнениями, связывающими шесть переменных (*m* = 2, *n* = 6), не может быть решена графическим методом. Однако данный метод может быть применен для решения двойственной к ней задачи, которая име­ет только две переменные.

Еще раз вернемся к таблице *Т*2(*q*) (*рис. 1.8*), получаемой на финальной итерации процедуры модифицированного симплекс-метода. Более подробно рассмотрим нулевую строку матрицы Δ-1(β(*q*)), для которой было введено обозначение δ0(β(*q*)). По­элементно она может быть записана в следующем виде:

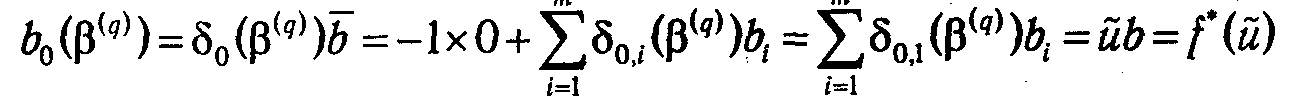


Введем вектор ** = (δ0,1(β(*q*)), δ0,2(β(*q*)),..., δ0,*m*(β(*q*))). Нетруд­но проверить, что строка оценок *a*0(β(*q*)) может быть представ­лена следующим образом:



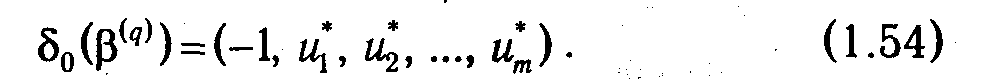
Согласно критерию оптимальности, на последней итерации данная строка неотрицательна, т. е. *ũА≥с*. Следовательно, век­тор и является допустимым планом двойственной задачи.

В то же время элемент *b*0(β(*q*)), содержащий текущее значе­ние целевой функции и равный на последней итерации *f(x\*),* до­пускает представление



Согласно теореме 1.5 из равенства *f*(*х\**) = *f*\*(*ũ*) вытекает, что вектор *ũ* служит оптимальным планом двойственной задачи: *u = ũ.*

Окончательно можно утверждать, что для оптимального базиса



* *Таким образом, существенным преимуществом модифи­цированного симплекс-метода является то, что он по­зволяет одновременно найти оптимальные планы как, прямой, так и двойственной задачи.*

Читателю в качестве самостоятельного упражнения предла­гается построить задачу, двойственную к (1.34)-(1.35), реше­ние которой было приведено в п. 1.5.2, и убедиться, что вектор *u* = (-10, 32, 2), полученный в таблице *Т*2(3), является для нее допустимым и оптимальным планом.

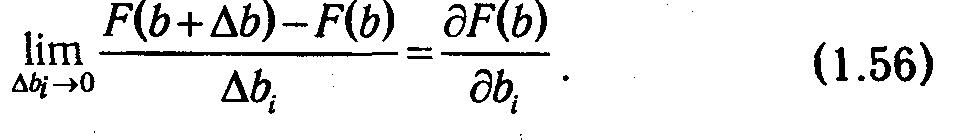
**1.6.4. Экономическая интерпретация.** Традиционная экономическая интерпретация двойственной задачи ЛП бази­руется на модели *простейшей задачи производственного планирования*, описанной во введении. Напомним, что в ней каждый (*j*-й) элемент вектора *х* рассматривается как план вы­пуска продукции данного вида в натуральных единицах, *сj* — цена единицы продукции *j*-го вида, *аj* — вектор, определяющий технологию расходования имеющихся *m* ресурсов на производ­ство единицы продукции *j*-го вида, *b* — вектор ограничений на объемы этих ресурсов.

Предположим, что для некоторых значений *A, b* и *с* найден оптимальный план *х\**, максимизирующий суммарный доход max{*cx*}=*cx*\*. Достаточно естественным представляется во­прос: как будет изменяться оптимальный план *х*\* при измене­нии компонент вектора ограничений *b* и, в частности, при ка­ких вариациях *b* оптимальный план *х*\* останется неизменным? Данная задача получила название *проблемы устойчивости оптимального плана.* Очевидно, что исследование устойчи­вости *х*\* имеет и непосредственное практическое значение, так как в реальном производстве объемы доступных ресурсов *bi*; могут существенно колебаться после принятия планового решения *х*\*.

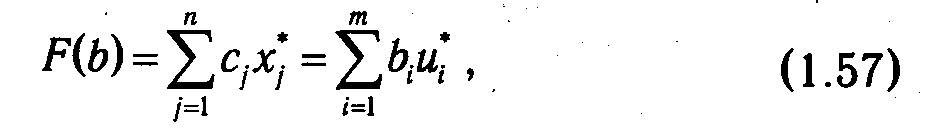
Когда вектор ограничений *b* изменяется на Δ*b* или, как еще говорят, получает приращение Δ*b*, то возникают соответству­ющие вариации для оптимального плана *х\*(b+*Δ*b)* и значения целевой функции f(*х*\*(*b+*Δ*b*)). Допустим, приращение Δ*b* та­ково, что оно не приводит к изменению оптимального базиса задачи, т. е. *х\*(b+*Δ*b)*≥0. Определим функцию *F*(*b*), возвраща­ющую оптимальное значение целевой функции задачи (*D*(*b*), *f*) для различных значений вектора ограничений *b*



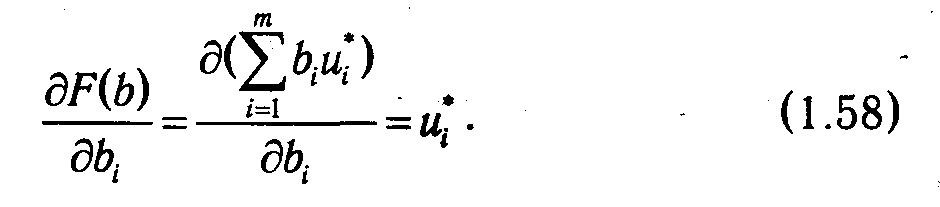
Рассмотрим отношение ее приращения *F(b+*Δ*b)-F(b)* к при­ращению аргумента Δ*b*. Если для некоторого *i* устремить Δ*bi* → 0, то мы получим



Учитывая, что в соответствии с теоремой 1.5



и подставив (1.57) в (1.56), приходим к выражению

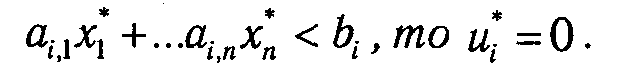


* *Из формулы (1.58) вытекает* ***экономическая интерпре­тация оптимальных переменных двойственной зада­чи****.* Каждый элемент *ui\** может рассматриваться как предель­ная (мгновенная) оценка вклада *i*-го ресурса в суммарный доход *F* при оптимальном решении *х*\*. *Грубо говоря, величи­на ui\** *равна приросту дохода, возникающему при увеличе­нии ресурса i на единицу при условии оптимального ис­пользования ресурсов.*

В различных источниках компоненты оптимального плана двойственной задачи также называются *двойственными оцен­ками* или *теневыми ценами*, а Л. В. Канторович предлагал та­кой термин, как *объективно обусловленные оценки.*

На основе теорем двойственности для пары задач ЛП **в об­щей форме** могут быть сформулированы некоторые важные (с точки зрения экономической интерпретации) следствия.

* *Если при использовании оптимального плана прямой за­дачи i-e ограничение выполняется как строгое неравен­ство, то оптимальное значение соответствующей двой­ственной переменной равно нулю, т.е. если*



В рамках рассматриваемой задачи производственного плани­рования это означает, что если некоторый ресурс *bi*, имеется в избыточном количестве (не используется полностью при реа­лизации оптимального плана), то *i*-e ограничение становится несущественным и оценка такого ресурса равна 0.

* *Если при использовании оптимального плана двойствен­ной задачи j-e ограничение выполняется как строгое не­равенство, то оптимальное значение соответствую­щей переменной прямой задачи должно быть равно нулю, т. е. если a*1,*ju*1\* *+...аm,j иm – сj > 0, то хj\* =*0*.*

Учитывая экономическое содержание двойственных оценок *u*1\*,...,*um*, выражение *а*1*,ju*1\* +…*am,jum\** может быть интерпретиро­вано как удельные затраты на *j*-й технологический процесс. Сле­довательно, если эти затраты превышают прибыль от реализа­ции единицы *j*-го продукта, то производство *j*-го продукта является нерентабельным и не должно присутствовать в опти­мальном производственном плане (*xj\** =0).

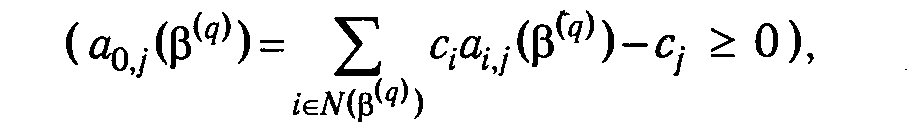
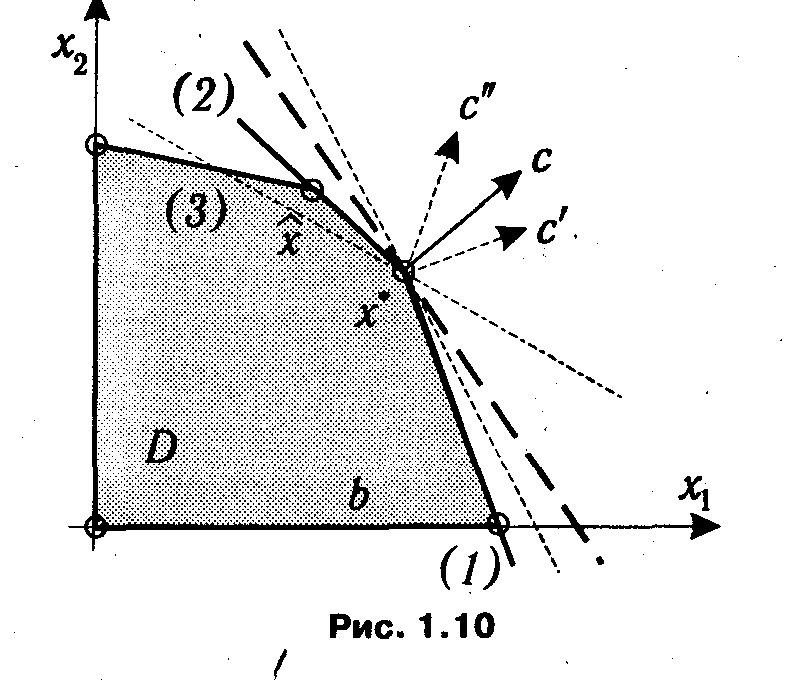
Несмотря на возможные аналогии, которые могут возник­нуть у читателей, знакомых с такими фундаментальными поня­тиями экономической теории, как предельные издержки и пре­дельный доход, двойственные оценки не следует однозначно отождествлять с ценами (хотя такие попытки иногда предпринимались на начальной стадии становления исследования опе­раций как науки). Еще раз подчеркнем, что *переменные двой­ственной задачи по своему смыслу являются оценками по­тенциальной возможности получения дополнительной прибыли за счет увеличения соответствующего ресурса в условиях оптимального функционирования управляемого экономического объекта.*

**1.6.5. Анализ параметрической устойчивости реше­ний ЗЛП**. В предыдущем пункте мы затронули некоторые ас­пекты чувствительности и устойчивости оптимального плана по отношению к изменению вектора ограничений *b*. Очевидно, что аналогичные вопросы могут быть поставлены для случая вариации коэффициентов целевой функции *сj*, *j*∈1:*n*.

С точки зрения экономической интерпретации задача иссле­дования параметрической устойчивости может быть рассмот­рена как изучение тех пределов колебания цен на продукцию управляемого предприятия (фирмы), при которых принятый план выпуска продукции продолжает оставаться оптимальным.

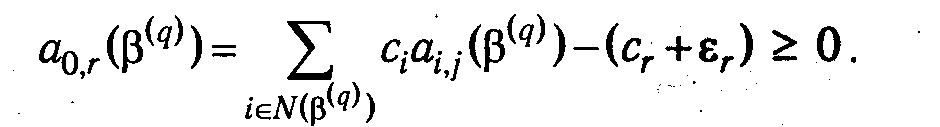
Также содержание проблемы устойчивости оптимального плана ЗЛП по отношению к вариациям целевой функции может быть проиллюстрировано с помощью первой геометрической интерпретации. На *рис. 1.10* изображено множество допусти­мых планов *D* некоторой задачи ЛП. Как видно из рисунка, це­левая функция *f* (ее поведение отражает линия уровня, пока­занная жирным пунктиром) достигает экстремального значе­ния в точке *х*\*, а изменению ее коэффициентов от *с* к *с'* или *с"* на рисунке соответствует поворот линии уровня относительно *х*\*. Активным, т. е. обращающимся в равенство, ограничениям в точке *х*\* соответствуют линии (1) и (2). До тех пор, пока при повороте, вызванном изменением вектора *с*, линия уровня целе­вой функции не выходит за пределы образуемого линиями огра­ничений конуса, *х\** остается оптимальным планом. Как показа­но на *рис. 1.10*, этот план не меняется при переходе от *с* к *с'*, и, наоборот, при переходе от *с* к *с"* линия уровня целевой функции *f(x)=c"x* пересечет линию (2), что вызовет изменение опти­мального базисного плана, которым теперь станет точка *.*

Используя условия оптимальности плана ЗЛП

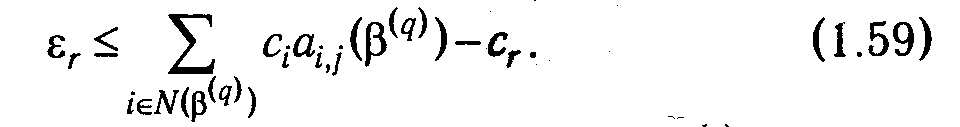


нетрудно получить количественные оценки для пределов коле­баний коэффициентов целевой функции, при которых не проис­ходит изменение оптимального плана. Допустим, вариации под­вергся некоторый элемент *сr* : *сr′* = *сr* + *εr*. Возможны два случая:

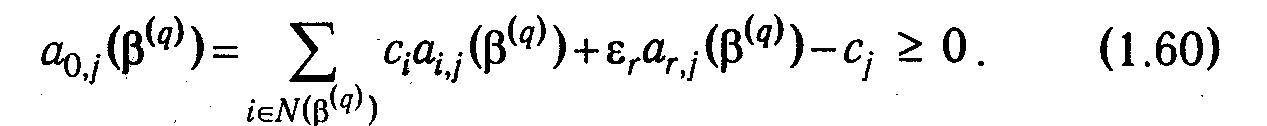
1. Столбец *r* не входит в оптимальный базис (*r* ∈ *N*(β(*q*))). Тог­да для неизменности оптимального плана необходимо и доста­точно выполнение условия



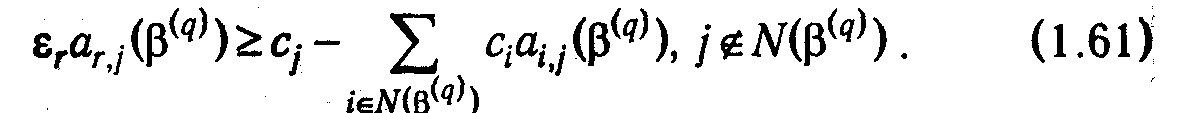
Отсюда можно получить значение для допустимой вариации



2. Столбец *r* входит в оптимальный базис (*r* ∈ *N*(β(*q*))). В этом случае для сохранения оптимальности текущего плана потре­буется выполнение для всех небазисных столбцов (*j* ∉ *N*(β(*q*))) условий



Следовательно, в этом случае допустимая вариация должна удовлетворять условиям



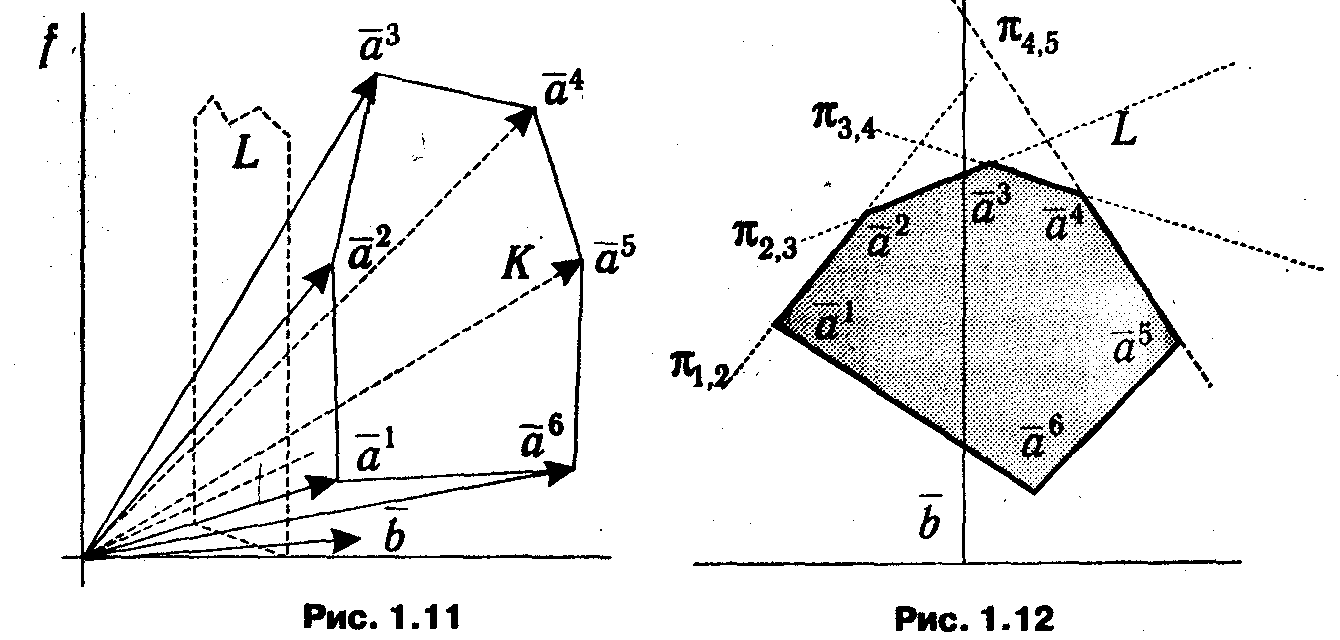
Приведенный пример исследования чувствительности оп­тимального плана по отношению к изменению параметров за­дачи является весьма простым. Очевидно, что существуют и более сложные задачи, в которых, например, исследуются совместные вариации параметров разных типов. Они состав­ляют предмет специального раздела исследования операций, получившего название *параметрического программирова­ния*. Заинтересованный читатель может получить дополни­тельную информацию по данному предмету в [1, 6].

## 1.7. ДВОЙСТВЕННЫЙ СИМПЛЕКС-МЕТОД

**1.7.1. Основные идеи двойственного симплекс-метода.** Непосредственное приложение теории двойственности к вы­числительным алгоритмам линейного программирования позво­лило разработать еще один метод решения ЗЛП, получивший название *двойственного симплекс-метода*, или *метода по­следовательного уточнения оценок*. Впервые он был предло­жен Лемке в 1954 г.

В процессе изложения идей, положенных в основу двойст­венного симплекс-метода, еще раз воспользуемся второй геометрической интерпретацией ЗЛП. Рассмотрим некоторую КЗЛП (1.48). На *рис. 1.11* изображен конус *К* положительных линейных комбинаций расширенных векторов условий *аj* для случая *m* = 2, *n* = 6, а на *рис. 1.12* — (для большей наглядно­сти) — поперечное сечение данного конуса некоторой плос­костью *L*, проходящей параллельно оси аппликат.

С каждым планом *и* задачи, двойственной по отношению к (1.48), можно взаимно однозначно связать гиперплоскость, проходящую через начало координат и перпендикулярную век­тору (-1,*и*). Допустимые планы двойственной задачи (1.49) должны удовлетворять условиям *иА* ≥ *с*, которые можно пере­писать в виде (1, *и*)*А* ≥ 0 . Последнее означает, что всякий рас­ширенный вектор условий *аj* лежит «ниже» плоскости, опреде­ляемой вектором (-1, *и*). Таким образом, любому допустимому плану двойственной задачи в рассматриваемом примере соот­ветствует некоторая плоскость, расположенная выше всех рас­ширенных векторов, а стало быть, и конуса *К*. На *рис. 1.12*, где изображено поперечное сечение, конусу *К* отвечает многогран­ник, а «гиперплоскостям двойственных планов» — пунктирные линии, являющиеся их следами.



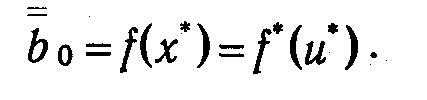
По аналогии с интерпретацией решения прямой задачи про­цесс решения двойственной задачи может быть представлен как *поиск такой гиперплоскости, которая лежит выше конуса К и пересекает прямую, проведенную через конец вектора b параллельно оси аппликат, в самой нижней точке.*

Из геометрической иллюстрации видно, что плоскости, не соприкасающиеся с конусом *К*, заведомо не представляют инте­реса, так как для любой из них легко указать плоскость, у кото­рой искомая точка пересечения лежит ниже. Таким образом, процесс поиска экстремума сводится к последовательному пе­ребору плоскостей, касающихся «верхних» граней конуса, или, как еще говорят, *опорных по отношению к конусу*. Для слу­чая, изображенного на *рис. 1.12*, таковыми являются гиперплоскости π1,2, π2,3, π3,4 и π4,5. Как можно заметить, каждая из рассматриваемых гиперплоскостей натянута на некоторый ба­зисный набор расширенных векторов *аj*, что, собственно, и ис­пользовано для их обозначения (π1,2 ~ {*а*1, *а*2} и т. д.).

* *В дальнейшем систему* β={*aj*1, *aj*2,...*,ajm*} *из т линейно-не­зависимых столбцов матрицы условий прямой задачи А будем называть* **сопряженным базисом***, или* **двойствен­но допустимым базисом***, если всякий вектор и* ∈ *Rm, удовлетворяющий условиям, удовлетворяет также ус­ловиям иаj≥cj(j*∈1:*n), т. е. является допустимым пла­ном двойственной задачи* (1.49).

План двойственной задачи *и*, соответствующий сопряженно­му базису β={*aj*1, *aj*2,...*,ajm*}, называют *опорным планом*. Его «опорность» заключается в том, что в системе ограничений *uаj* ≥ *cj*  (*j*∈1:*n*), задающих область определения двойственной задачи *D*\*, *т* неравенств с номерами *j*∈ *N*(β) обращаются в ра­венства.

Следует обратить внимание на то, что **не всем сопряжен­ным базисам** соответствуют допустимые базисные планы пря­мой задачи. В частности, вектор *b* не может быть разложен с неотрицательными коэффициентами по базисам {*а*1, *а*2}, {*а*3 , *а*4 } или {*а*4, *а*5}. В связи с этим систему коэффициентов разложения вектора *b* по сопряженному базису называют *псев­допланом*. В то же время базис {а2, а3} является допустимым для прямой задачи, и, более того, из иллюстрации видно, что он, с одной стороны, определяет максимум прямой задачи (наивыс­шую точку пересечения прямой, проходящей через конец *b*, с конусом *К*), а с другой — минимум двойственной (низшую точ­ку пересечения этой прямой с лежащей над *К* опорной гипер­плоскостью):



Последний факт является не чем иным, как геометрической иллюстрацией утверждения теоремы 1.5.

Описанные выше свойства пары двойственных задач линейно­го программирования являются идейной основой двойственного симплекс-метода, который представляет собой итеративный про­цесс целенаправленного перебора сопряженных (двойственно допустимых) базисов и соответствующих псевдопланов. В этом и заключено его принципиальное отличие от обычного симп­лекс-метода, в котором последовательно рассматриваются допу­стимые базисные планы прямой задачи\*. Нетрудно догадаться, что при определенной структуре задачи путь, предлагаемый двойственным алгоритмом, может оказаться более коротким.

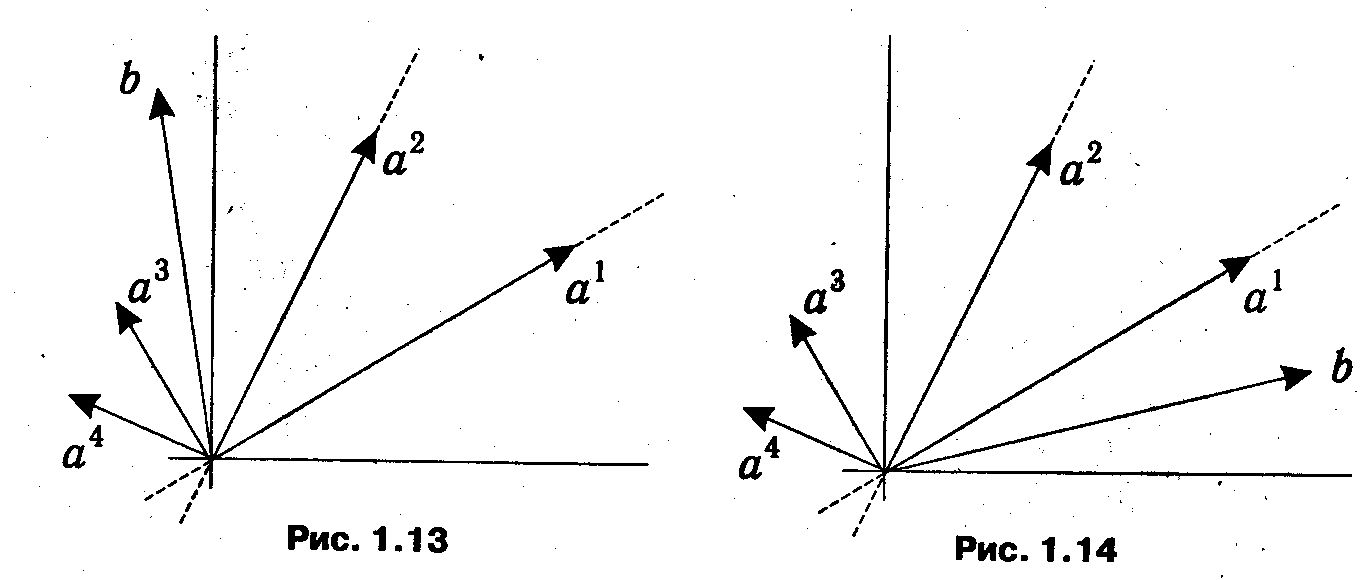
\* В данном пункте используется та же система обозначений, что и в п. 1.4.1.

☞ *Критерий оптимальности псевдоплана х в двойствен­ном симплекс-методе заключается в том,*

*что х одновре­менно должен являться допустимым планом прямой задачи, т. е. все его*

*компоненты должны быть не­отрицательны* (*хj* ≥ 0).

Обратно, если хотя бы одна из компонент псевдоплана яв­ляется отрицательной, то процесс улучшения значения целе­вой функции может быть продолжен. Геометрическая иллюст­рация данной ситуации приведена на *рис. 1.13*. Здесь на плоскости (для *m* = 2) изображена система столбцов ограниче­ний КЗЛП, из которых {*а*1, *а*2} образуют текущий базис. Как видно из рисунка, вектор ограничений имеет отрицательную ко­ординату по направлению, задаваемому вектором *а*1. В то же время очевидны и те базисы (например, {*а*2, *а*3}), в которых *b* будет иметь все положительные координаты. Однако это не все­гда так. Пример на *рис. 1.14* иллюстрирует случай отсутствия допустимых планов у прямой задачи: вектор *b* имеет отрицатель­ную компоненту в текущем базисе {*а*1, *а*2} по направлению *а*2, а для всех остальных небазисных столбцов (*а*3, *а*4) данная коор­дината является положительной, т. е. *b* и столбцы, являющиеся кандидатами на ввод в очередной базис, лежат в разных полу­плоскостях, образуемых прямой, проходящей через вектор *а*1,



и при любых базисах, отличных от текущего, соответствующая координата вектора *b* все равно останется отрицательной.

☞ *Таким образом, в двойственном симплекс-методе признаком отсутствия допустимых планов у*

*решаемой КЗЛП является неотрицательность каких-либо r-х компонент во всех столбцах аj,*

*представленных в текущем базисе* β (*ar,j*(β) ≥ 0, *j* ∈1:*n*) *если br*(β) < 0 l.\*

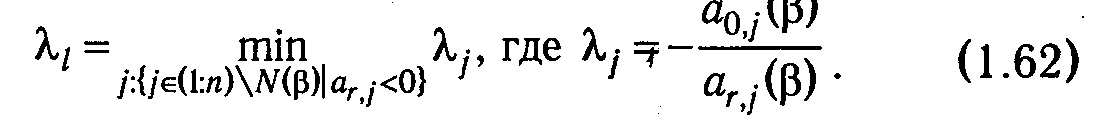
\* Здесь следует подчеркнуть, что двойственный симплекс-метод не­посредственно нацелен на **нахождение решения прямой задачи.**

С другой стороны, если *br*(β)<0 и в строке *аr*(β) существуют отрицательные координаты, то псевдоплан можно улучшить выводя из базиса столбец, находящийся на *r-м* месте, и вводя в него некоторый вектор *al*, имеющий отрицательную *r-ую* координату. При наличии в векторе *b*(β) нескольких отрицательных компонент номер вектора, выводимого из базиса, обычно опре­деляется строкой, содержащей наименьшую (наибольшую по модулю и отрицательную) компоненту.

Принцип выбора столбца, вводимого в базис, определяется необходимостью обеспечивать то, чтобы очередной базис опре­делял опорную плоскость, ниже которой располагаются все не­базисные векторы. Для этого требуется, чтобы оценки всех не­базисных векторов *а*0,*j*(β) ( *j* ∈ *N*(β)), вычисляемые по формулам

*а*0,*j*(β) = -*cj* + *c*(β)*аj*(β)

(см. (1.26)), оставались положительными. Это, в свою очередь, означает, что номер вводимого столбца *l* должен быть опреде­лен таким образом. Чтобы



После выбора выводимого и вводимого векторов производит­ся обычный пересчет симплекс-таблицы по формулам, анало­гичным формулам (1.28)-(1.31), и процесс итеративно повто­ряется. В результате через конечное число шагов будет найден оптимальный план КЗЛП или установлен факт пустоты множе­ства допустимых планов.

**1.7.2. Алгоритм и табличная реализация двойственно­го симплекс-метода**. Обобщая сказанное в предыдущем пун­кте, приведем в сжатом виде алгоритм двойственного симплекс-метода для решения КЗЛП.

**0-этап. Нахождение исходного сопряженного (двой­ственно допустимого базиса).** Результатом 0-этапа являют­ся сопряженный базис β(1) и соответствующие ему псевдоплан *x*(β(1)), матрица *A*(β(1)) и вектор *b*(β(1)), которые будут исполь­зованы на первой итерации. Полагаем номер текущей итерации *q* равным 1 и переходим к I-этапу.

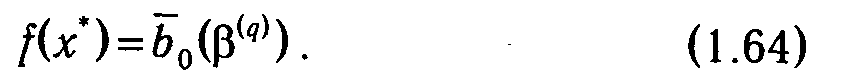
**I-этап. Стандартная итерация алгоритма** — выполня­ется для очередного сопряженного базиса β(*q*).

1°. *Проверка оптимальности* текущего псевдоплана: осу­ществляется просмотр значений *bi*(β(*q*)), *i*∈1:*m*. Возможны два варианта:

1΄. Для всех *i*∈1:*m*, *bi*(β(*q*)) ≥ 0. Тогда текущий псевдоплан *x*(β(*q*)) одновременно является допустимым планом решаемой задачи, т. е. ее **оптимальным** планом. Вычислительный про­цесс закончен. Элементы оптимального плана *х*\* определяют­ся по формуле



а достигаемое на нем значение целевой функции равно



1". Существует по меньшей мере один номер строки *r*∈1:*m*, для которого *br*(β(*q*))<0 . Следовательно, псевдоплан *x*(β(*q*)) — **неоптимален.** Выбирается строка с номером *r*, такая, что



Она становится ведущей и определяет номер столбца *Nr*(β(*q*)), который должен быть выведен из базиса. Переходим к пункту 2° алгоритма.

2°. *Определение столбца, вводимого в базис.* Рассматрива­ется ведущая строка *аr*(β(*q*)). Возможны два варианта:

2'. Для всех *j*∈1: *n* *аr,j*(β(*q*)) ≥ 0. Делается вывод об *отсут­ствии допустимых планов у решаемой задачи* (*D* = Ø) и за­вершается вычислительный процесс\*.

\* Очевидно, что для определения пустоты множества допустимых планов достаточно найти полностью неотрицательную строку *аi*(β(*q*)), соответствующую *bi*(β(*q*)) < 0.

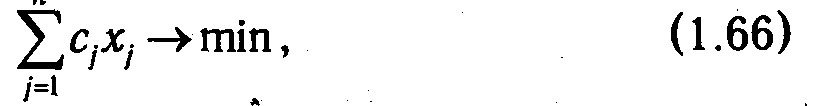
2". В строке *аr*(β(*q*)) существует по крайней мере один эле­мент *аr,j*(β(*q*))<0. Согласно правилу (1.62) определяются *l* — номер столбца, вводимого в очередной базис. Переходим к пун­кту 3° алгоритма.

3°. *Пересчет элементов матрицы А и столбца b* относи­тельно нового базиса. В соответствии с формулами (1.28)-(1.31) осуществляем расчет элементов матрицы задачи *A* и век­тора ограничений *b* относительно нового базиса β(*q*+1), который получается в результате вывода из базиса β(*q*) столбца *аr* и вво­да в него столбца *аl*. Полагаем номер текущей итерации *q*: *= q+*1 и переходим к первому пункту алгоритма.

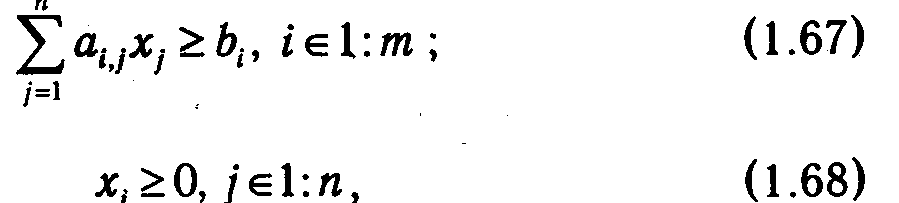
По аналогии со стандартным симплекс-методом вычислитель­ную процедуру двойственного симплекс-метода удобно офор­млять в виде таблиц, приведенных на *рис. 1.5.* Очевидно, что с формальной стороны их структура остается неизменной. Иног­да считается целесообразным добавить к двойственной симп­лекс-таблице строку, содержащую строку со значениями λ*j*, которые вычисляются на этапе определения столбца, вводимо­го в базис.

**1.7.3. Особенности применения двойственного симп­лекс-метода**. Алгоритм двойственного симплекс-метода (как и остальные симплекс-алгоритмы) предполагает знание исход­ного сопряженного базиса. Очевидно, что в общем случае его нахождение является достаточно непростой задачей, сводящей на нет потенциальные преимущества двойственного алгоритма. Однако в ряде случаев исходный псевдоплан может быть опре­делен достаточно легко.

Рассмотрим задачу минимизации:



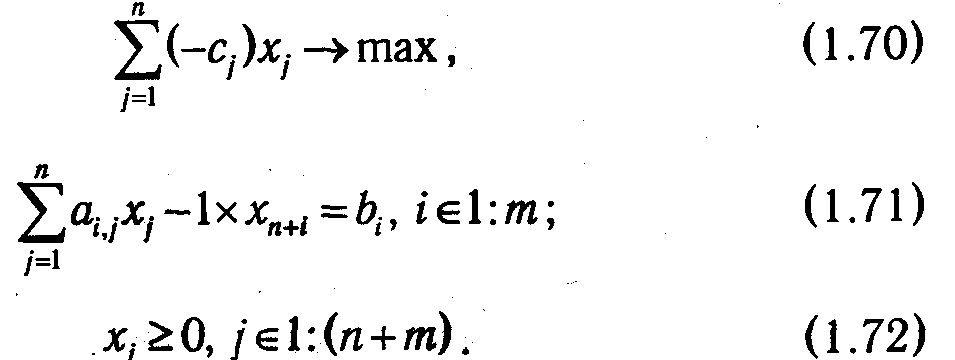
при ограничениях



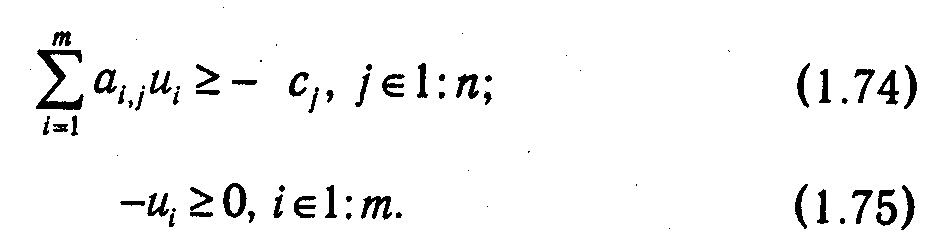
где



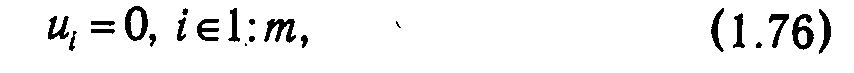
Приведем задачу (1.66)-(1.68) к канонической форме, введя фиктивные переменные *хn*+1, *хn*+2, *...* , *хn+m*:



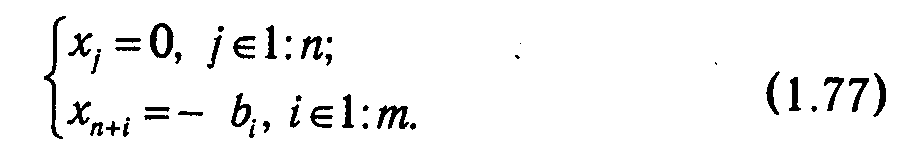
Задача, двойственная к (1.70)—(1.72), будет иметь вид:



Из (1.74)-(1.75) очевиден допустимый план двойственной задачи



и исходный сопряженный базис, образуемый векторами *аn+*1, *аn+*2, …., *аn+m.* При этом начальный псевдоплан равен



Таким образом, при решении задачи вида (1.66)-(1.68) двой­ственный симплекс-метод имеет несомненные преимущества по сравнению с прямым.

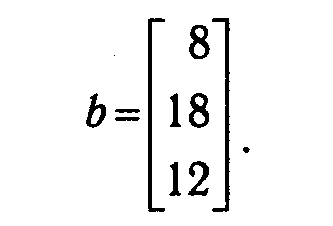
Другое важное направление использования двойственного симплекс-метода связано с поиском оптимальных планов в тех задачах, условия которых претерпели некоторые изменения после того, как они уже были решены с помощью стандартной симплекс-процедуры. Типичными примерами таких изменений являются:

* *изменение компонент вектора ограничений b, что, допу­стим, может быть интерпретировано как корректиров­ка объемов доступных ресурсов в процессе управления экономическим объектом;*
* *добавление новых ограничений к системе условий задачи, что достаточно часто случается при совершенствова­нии используемой экономико-математической модели.*

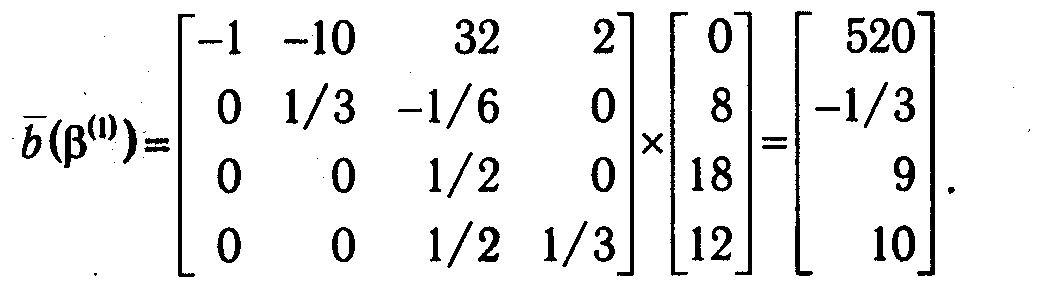
В первом случае, т. е. при изменении вектора *b*, достоинства двойственного симплекс-метода очевидны, так как ранее найден­ный оптимальный базис можно использовать в качестве исход­ного сопряженного базиса при продолжении решения. Второй случай более подробно будет рассмотрен в гл. 4 при рассмотре­нии методов решения целочисленных задач.

В заключение отметим, что в настоящем параграфе был рас­смотрен вариант двойственного алгоритма, соответствующий стандартному симплекс-методу. Нетрудно догадаться, что су­ществует и вариант, построенный на базе модифицированного симплекса (схемы, связанной с преобразованием обратных мат­риц), но, поскольку этот вопрос представляет интерес в основ­ном с точки зрения техники организации вычислений, мы на нем останавливаться не будем. При желании с глубоким и детальным описанием данной версии алгоритма можно ознако­миться в [1]. Отметим лишь, что она обладает теми же прин­ципиальными преимуществами, что и модифицированный симп­лекс-метод.

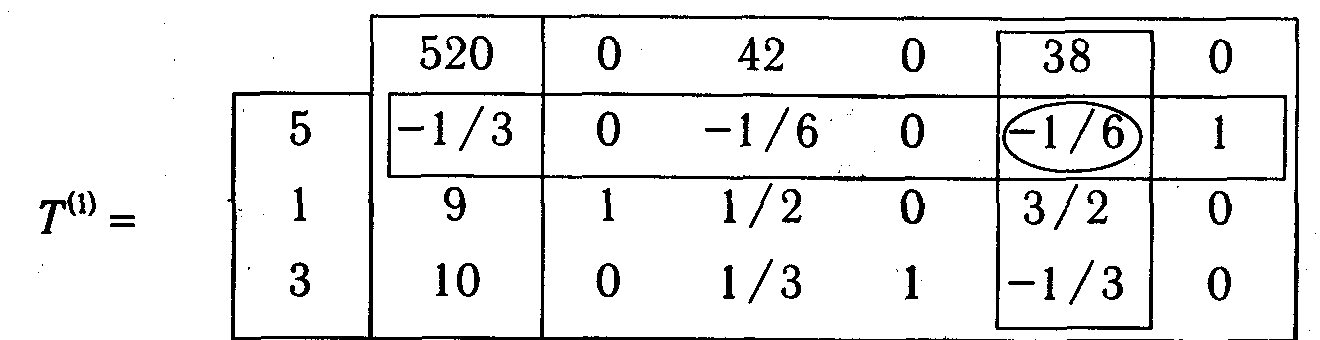
**1.7.4. Пример решения ЗЛП двойственным симплекс-методом**. Рассмотрим на конкретном, примере процесс реше­ния КЗЛП двойственным симплекс-методом. Для этого, опять-таки, вернемся к задаче (1.34)-(1.35), решенной в п. 1.4.3 и п. 1.5.2. Предположим, что произошли изменения в векторе ограничений *b* в результате которых



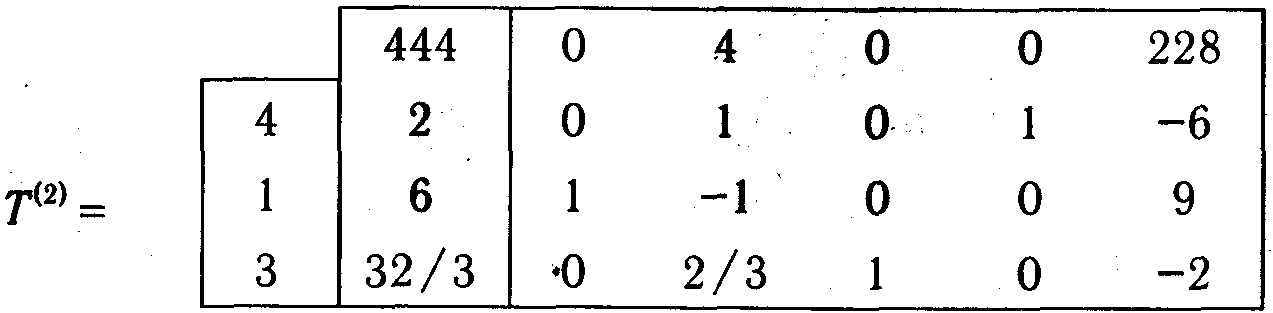
Содержание исходной симплекс-таблицы *T*(1) (за исключением столбца *b*(β(1))) будет идентично содержанию таблицы, полу­чающейся на последнем шаге алгоритма, рассмотренного в п. 1.4.3. Для вычисления значений *b*(β(1)) в данном случае мож­но воспользоваться обратной матрицей, полученной на послед­ней итерации в п. 1.5.2:



В результате имеем:



Как видно из таблицы *Т*(1), в столбце *b*(β(1)) содержатся отрицательные элементы *b*1(β(1)) = - 1/3<0), то есть базис β(1) ={**5, **1, **3} не является оптимальным, но в то же время легко убедиться, что он обладает свойствами сопряженного базиса. Отрицательный элемент в *b*(β(1)) является единствен­ным, поэтому номер столбца, выводимого из базиса, опреде­ляется однозначно — *r* = 1 и *N*1(β(1))=5. Далее рассматриваем строку *a*1(β(1)) = (0, -1/6, 0, -1/6, 1). В ней имеются отри­цательные элементы. Вычисляем λ2 =42:(-(-1/6))=252, λ4 =38:(-(-1/6))=228. λ2> λ4, следовательно, номер столбца, вводимого в базис — *l* = 4. Осуществляем преобразование и получаем симплекс-таблицу *T*(2).



Поскольку *b*(β(2)) >0, то достигнутый базис *N*(β(2)) = {4,1,3} является оптимальным. Из *Т*(2) можно выписать оптимальный план *х*\* = (6, 0, 32/3, 2, 0) и соответствующее ему значение це­левой функции *f*(*x*\*)= 444.

## КЛЮЧЕВЫЕ ПОНЯТИЯ

* Общая задача линейного программирования (ОЗЛП).
* Каноническая задача линейного программирования (КЗЛП).
* Допустимый план.
* Оптимальный план.
* Первая геометрическая интерпретация ЗЛП.
* Базисное решение ЗЛП.
* Вторая геометрическая интерпретация ЗЛП.
* Вырожденный и невырожденный план ЗЛП.
* Симплекс-метод — метод последовательного улучшения плана.
* Критерий оптимальности допустимого базисного плана.
* Метод минимизации невязок.
* Модифицированный симплекс-метод — вычислительная схема, связанная с преобразованием обратных матриц.
* Двойственная задача линейного программирования.
* Симметричность отношения двойственности.
* Теоремы двойственности.
* Экономическая интерпретация двойственных оценок.
* Параметрическая устойчивость решения ЗЛП.
* Двойственный симплекс-метод — метод последовательно­го уточнения оценок.
* Сопряженный (двойственно допустимый) базис.
* Опорный план и псевдоплан.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

1.1. Сформулируйте задачу линейного программирования.

1.2. Дайте определение для следующих понятий: план, допус­тимый план, оптимальный план,

решение задачи.

1.3. Чем отличается общая задача линейного программирова­ния от канонической?

1.4. Всегда ли общую задачу линейного программирования можно привести к каноническому виду?

1.5. Дайте определения для следующих понятий: аффинное множество, гиперплоскость, базис.

1.6. Чем отличается выпуклый многогранник от многогранно­го выпуклого множества?

1.7. В чем отличие понятий «линейная оболочка» и «выпуклая оболочка»?

1.8. Любой ли конус является выпуклым множеством?

1.9. Какая точка выпуклого множества называется угловой?

1.10. В чем заключается первая геометрическая интерпрета­ция задачи линейного программирования?

1.11. В чем заключается вторая геометрическая интерпрета­ция задачи линейного программирования?

В чем ее отли­чие от первой?

1.12. Какой план называется базисным?

1.13. Как связаны базисные планы и угловые точки области оп­ределения задачи линейного

программирования?

1.14. Какой план задачи линейного программирования называ­ется вырожденным?

1.15. Как с точки зрения второй геометрической интерпрета­ции можно представить процесс поиска

оптимального плана в задаче линейного программирования?

1.16. Сформулируйте критерий оптимальности допустимого базисного плана, применяемый в

симплекс-методе.

1.17. Сформулируйте основные этапы стандартной итерации симплекс-метода.

1.18. Для чего применяется преобразование Жордана—Гаусса?

1.19. Какой элемент симплекс-таблицы называется ведущим?

1.20. При каких условиях делается вывод о неограниченности целевой функции в решаемой задаче?

Какая геометриче­ская интерпретация соответствует данному случаю?

1.21. Можно ли заранее точно определить количество итера­ций, которое потребуется для решения

задачи симплекс-методом? Можно ли найти верхнюю границу для данной величины?

1.22. Какая задача называется вырожденной? По каким призна­кам можно узнать, что текущий план

является вырожден­ным?

1.23. Какие проблемы возникают при решении вырожденных задач?

1.24. Какую экономическую интерпретацию имеет ситуация вырожденности?

1.25. В чем основная идея метода возмущений?

1.26. Для чего предназначен метод минимизации невязок?

1.27. Сформулируйте основные отличия модифицированного симплекс-метода по отношению к

стандартному.

1.28. Перечислите преимущества модифицированного симп­лекс-метода.

1.29. Будет ли отличаться количество итераций при решении одной и той же задачи при решении ее

стандартным и мо­дифицированным симплекс-методом?

1.30. Дайте определение двойственной задачи.

1.31. Какими основными свойствами обладает пара двойствен­ных задач?

1.32. В чем заключается экономическая интерпретация пере­менных двойственной задачи?

1.33. Какой смысл вкладывается в понятие «параметрическая устойчивость»?

1.34. Сформулируйте условия для допустимых изменений це­левой функции задачи, при которых ее

оптимальный план остается неизменным.

1.35. Перечислите основные идеи, на которых базируется алго­ритм двойственного симплекс-метода.

1.36. Дайте определение сопряженного базиса.

1.37. Что такое псевдоплан?

1.38. Сформулируйте критерий оптимальности, используемый в алгоритме двойственного симплекс-

метода.

1.39. По каким признакам можно определить, что множество допустимых планов задачи, решаемой

двойственным сим­плекс-методом, пусто?

* 1. В каких ситуациях могут быть реализованы преимуще­ства двойственного симплекс-метода?

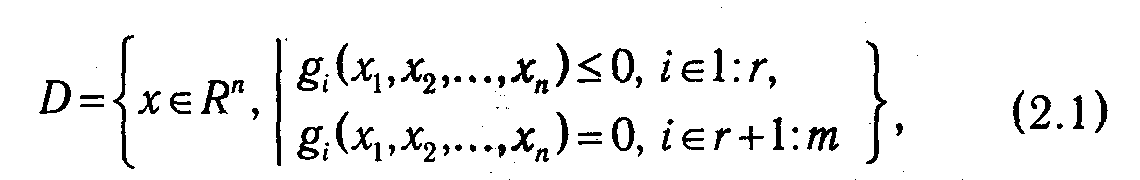
# ГЛАВА 2. НЕЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ

## 2.1. МЕТОДЫ РЕШЕНИЯ ЗАДАЧ

## НЕЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ

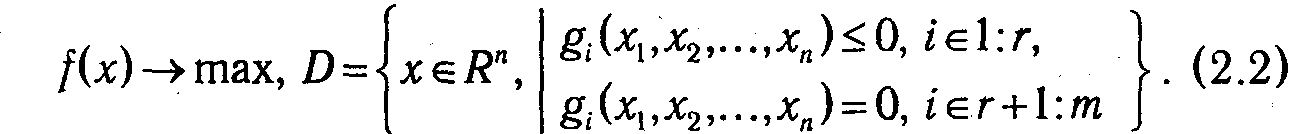
**2.1.1. Постановка задачи.** Как уже упоминалось во введе­нии, предположение о возможности описать зависимости меж­ду управляемыми переменными с помощью линейных функций далеко не всегда адекватно природе моделируемого объекта. Например, в рассмотренных в главе 1 моделях цена товара счи­тается независимой от количества произведенного продукта, однако в повседневной жизни мы постоянно сталкиваемся с тем, что она может зависеть от объема партии товара. Анало­гичные замечания могут быть сделаны и по поводу технологи­ческих ограничений: расход определенных видов сырья и ресур­сов происходит не линейно, а скачкообразно (в зависимости от объема производства). Попытки учесть эти факторы приводят к формулировке более общих и сложных оптимизационных за­дач. Изучение методов их решения составляет предмет научной области, получившей названия *нелинейного программиро­вания*.

Общая задача нелинейного программирования (ОЗНП) оп­ределяется как задача нахождения максимума (или минимума) целевой функции *f*(*x*1, *х*2,..., *xn*) на множестве *D*, определяемом системой ограничений



где *хотя бы одна* из функций *f* или *gi* является нелинейной.

По аналогии с линейным программированием ЗНП однознач­но определяется парой (*D*, *f*) и кратко может быть записана в следующем виде

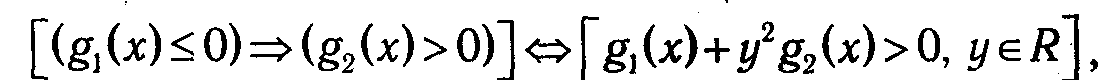


Также очевидно, что вопрос о типе оптимизации не являет­ся принципиальным. Поэтому мы, для определенности, в даль­нейшем по умолчанию будем рассматривать задачи максими­зации.

Как и в ЗЛП, вектор *х*\* = (*x*1\*,*x*2\*,...,*xn\**) ∈ D называется *допус­тимым планом*, а если для любого x ∈ D выполняется неравен­ство *f*(*x*\*) ≥ *f*(*x*), то *х\** называют *оптимальным планом.* В этом случае *х\** является точкой глобального максимума.

С точки зрения экономической интерпретации *f(x)* может рассматриваться как доход, который получает фирма (предпри­ятие) при плане выпуска *х*, а *gi*(*х*) ≤ 0 как технологические ог­раничения на возможности выпуска продукции. В данном слу­чае они являются обобщением ресурсных ограничений в ЗЛП (*аiх – bi* ≤ 0).

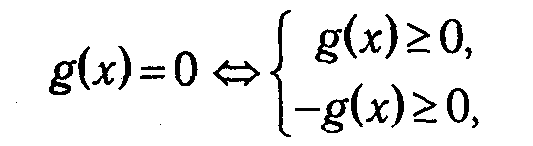
Задача (2.2) является весьма общей, т. к. допускает запись логических условий, например:



или запись условий дискретности множеств:



Набор ограничений, определяющих множество *D*, при необ­ходимости всегда можно свести либо к системе, состоящей из одних неравенств:



либо, добавив фиктивные переменные *у*, к системе уравнений:



Перечислим свойства ЗНП, которые существенно усложня­ют процесс их решения по сравнению с задачами линейного про­граммирования:

1. Множество допустимых планов *D* может иметь очень сложную структуру (например, быть невыпуклым или несвяз­ным).

2. Глобальный максимум (минимум) может достигаться как внутри множества *D*, так и на его границах (где он, вообще гово­ря, будет не совпадать ни с одним из локальных экстремумов).

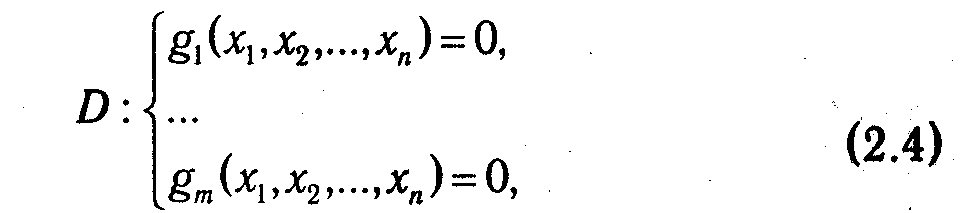
3. Целевая функция *f* может быть недифференцируемой, что затрудняет применение классических методов математическо­го анализа.

В силу названных факторов задачи нелинейного программи­рования настолько разнообразны, что для них не существует общего метода решения.

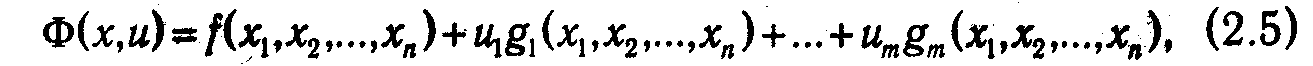
**2.1.2. Решение задач условной оптимизации методом Лагранжа.** Одним из наиболее общих подходов к решению за­дачи поиска экстремума (локального максимума или миниму­ма) функции при наличии связующих ограничений на ее пере­менные (или, как еще говорят, *задачи условной оптимизации*) является **метод Лагранжа**. Многим читателям он должен быть известен из курса дифференциального исчисления. Идея данного метода состоит в сведении задачи поиска условного экcтремума целевой функции



на множестве допустимых значения *D*, описываемом системой уравнений



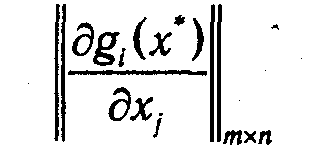
к задаче безусловной оптимизации функции



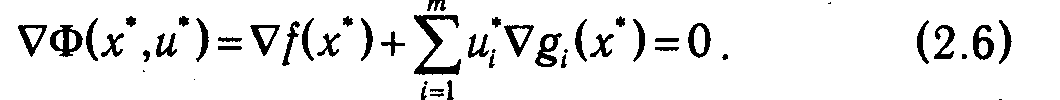
где *u* ∈ *Rm* — вектор дополнительных переменных, называе­мых *множителями Лагранжа.* Функцию Ф(*х,и*), где *x* ∈ *Rn*, *u* ∈ *Rn*, называют *функцией Лагранжа*. В случае дифференцируемости функций *f* и *gi* справедлива теорема, опре­деляющая необходимое условие существования точки услов­ного экстремума в задаче (2.3)-(2.4). Поскольку она непосредственно относится к предмету математического ана­лиза, приведем ее без доказательства.

*Теорема 2.1. Если х\* является точкой условного эк­стремума функции (2.3) при ограничениях*

*(2.4) и ранг матрицы первых частных производных функций*



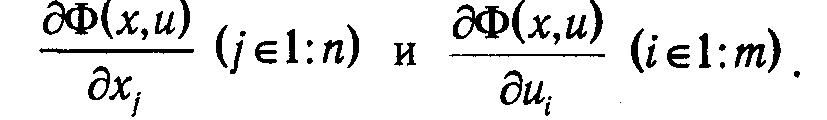
*равен т, то существуют такие и*1\*, *и*2\*,*...,иm\*, не равные одновременно нулю, при которых*



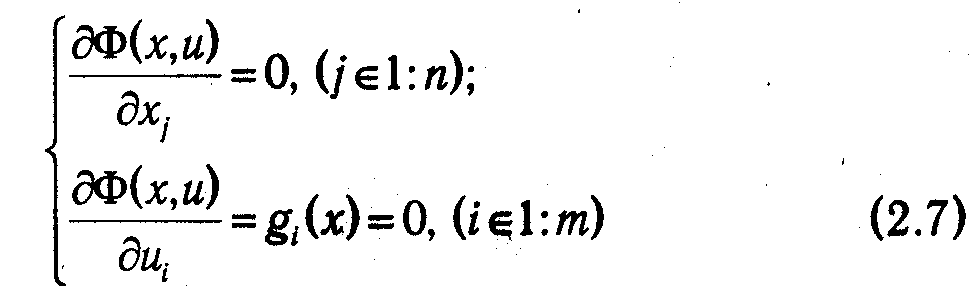
Из теоремы (2.1) вытекает метод поиска условного экстре­мума, получивший название метода множителей Лагранжа, или просто **метода Лагранжа**. Он состоит из следующих этапов.

1. Составление функции Лагранжа Ф(*х,и*).

2. Нахождение частных производных



3. Решение системы уравнений



относительно переменных *x* и *u*.

4. Исследование точек, удовлетворяющих системе (2.7), на максимум (минимум) с помощью достаточного признака экст­ремума.

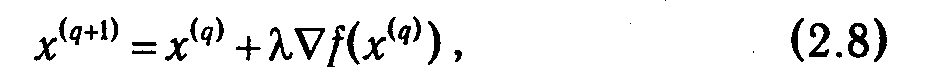
Присутствие последнего (четвертого) этапа объясняется тем, что теорема (2.1) дает необходимое, но не достаточное ус­ловие экстремума. Положение дел с достаточными признаками условного экстремума обстоит гораздо сложнее. Вообще гово­ря, они существуют, но справедливы для гораздо более частных ситуаций (при весьма жестких предпосылках относительно функций *f* и *gi*) и, как правило, трудноприменимы на практике.

Еще раз подчеркнем, что основное практическое значение метода Лагранжа заключается в том, что он *позволяет перей­ти от условной оптимизации к безусловной* и, соответствен­но, расширить арсенал доступных средств решения проблемы. Однако нетрудно заметить, что задача решения системы урав­нений (2.7), к которой сводится данный метод, в общем случае не проще исходной проблемы поиска экстремума (2.3)-(2.4). Методы, подразумевающие такое решение, называются *непря­мыми.* Они могут быть применены для весьма узкого класса за­дач, для которых удается получить линейную или сводящуюся к линейной систему уравнений (2.7). Их применение объясняет­ся необходимостью получить решение экстремальной задачи в аналитической форме (допустим, для тех или иных теоретиче­ских выкладок). При решении конкретных практических задач обычно используются *прямые* методы, основанные на итера­тивных процессах вычисления и сравнения значений оптимизи­руемых функций.

**2.1.3. Градиентные методы решения задач безуслов­ной оптимизации**. Ведущее место среди прямых методов ре­шения экстремальных задач занимает *градиентный метод* (точнее, семейство градиентных методов) поиска стационар­ных точек дифференцируемой функции. Напомним, что *стаци­онарной* называется точка, в которой *f*(*x*)=0 и которая в со­ответствии с необходимым условием оптимальности является «подозрительной» на наличие локального экстремума. Таким образом, применяя градиентный метод, находят множество то­чек локальных максимумов (или минимумов), среди которых определяется максимум (или минимум) глобальный.

Идея данного метода основана на том, что градиент функции указывает направление ее *наиболее быстрого возрастания* в окрестности той точки, в которой он вычислен. Поэтому, если из некоторой текущей точки *х*(1) перемещаться в направлении вектора *f*(*x*(1)), то функция *f* будет возрастать, по крайней мере, в некоторой окрестности *х*(1). Следовательно, для точки *х*(2) = *х*(1) +λ*f*(*x*(1)), (λ>0), лежащей в такой окрестности, справедливо неравенство *f*(*x*(1))≤ *f*(*x*(2)). Продолжая этот про­цесс, мы постепенно будем приближаться к точке некоторого локального максимума (см. *рис. 2.1*).

Однако как только определяется направление движения, сразу же встает вопрос о том, как далеко следует двигаться в этом направлении или, другими словами, возникает проблема выбора шага λ в рекуррентной формулe



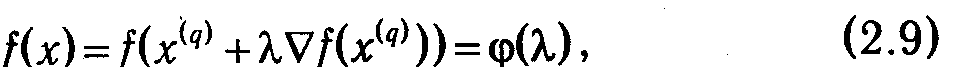
задающей последовательность точек, стремящихся к точке мак­симума.

В зависимости от способа ее решения различают различные варианты градиентного метода. Остановимся на наиболее изве­стных из них.

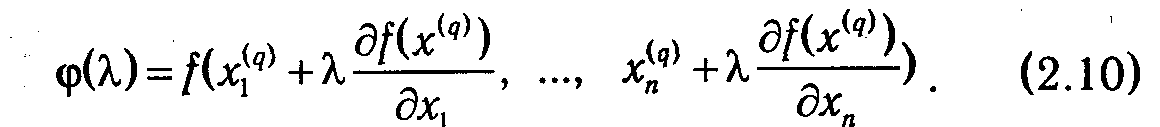
Метод наискорейшего спуска

Название метода можно было бы понимать буквально, если бы речь шла о минимизации целевой функции. Тем не менее по тра­диции такое название используется и при решении задачи на максимум.

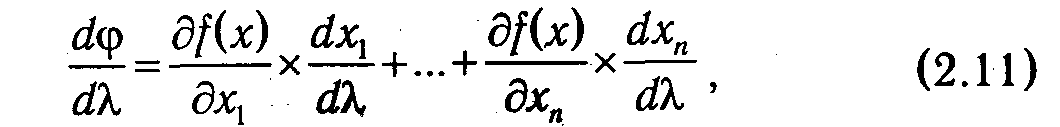
Пусть *f(x)=f(x*1*,x*1*,...xn)* —дифференцируемая функция, заданная на *Rn*, а *х*(*q*) = (*x*1(*q*)*,x*2(*q*)*,...,xn*(*q*)) — некоторая текущая точка. Оговоримся, что каких-либо общих рекомендаций, каса­ющихся выбора исходной точки (или, как еще говорят, началь­ного приближения) *х*(0), не существует, однако по возможно­сти она должна находиться близко от искомого оптимального плана *х*\*. Как уже говорилось выше, если *х*(*q*)— нестационарная точка (т. е. |*f*(*x*(*q*))|>0), то при движении в направлении *f*(*x*(*q*)) функция *f(х*) на некотором промежутке обязательно будет возрастать. Отсюда возникает естественная идея такого выбора шага, чтобы движение в указанном направлении продол­жалось до тех пор, пока возрастание не прекратится. Для этого выразим зависимость значения *f(x)* от шагового множителя λ > 0. полагая х = х(*q*) + λ*f*(*x*(*q*))



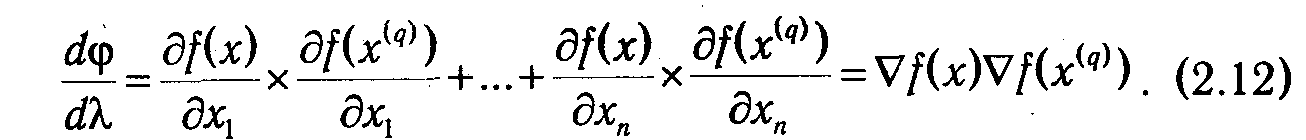
или, в координатной форме,



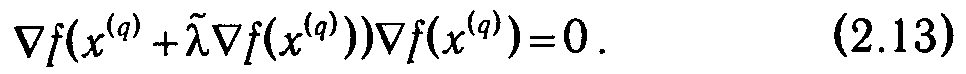
Чтобы добиться наибольшего из возможных значений *f* при движении по направлению *f*(*x*(*q*)), нужно выбрать такое значе­ние , которое максимизирует функцию φ(λ) (φ()= max(φ(λ)). Для вычисления , используется необходимое условие экстре­мума *d*φ(λ)/*d*λ = 0. Заметим, что если для любого λ>0 *d*φ()/*d*λ> 0, то функция *f(х)* не ограничена сверху (т. е. не име­ет максимума). В противном случае, на основе (2.10) получаем



что, в свою очередь, дает



Если считать, что следующая точка *х*(*q+*1) соответствует оп­тимальному значению λ=, то в ней должно выполняться условие *d*φ()/*d*λ = 0, и  следует находить из условия *f*(*x*(*q*+1)) *f*(*x*(*q*))=0 или



Условие (2.13) означает равенство нулю скалярного про­изведения градиентов функции *f* точках *x*(*q*+1) и *x*(*q*). Гео­метрически оно может быть интерпретировано как перпенди­кулярность векторов градиентов функции *f* в указанных точ­ках, что и показано на *рис. 2.2*. Продолжая геометрическую интерпретацию метода наискорейшего спуска, отметим, что в точке *x*(*q*+1) вектор *f*(*x*(*q*+1)), будучи градиентом, перпендику­лярен линии уровня, проходящей через данную точку. Стало быть, вектор *f*(*x*(*q*)) является касательным к этой линии. Итак, движение в направлении градиента *f*(*x*(*q*)) следует продолжать до тех пор, пока он пересекает линии уровня оп­тимизируемой функции.

После того как точка *x*(*q*+1) найдена, она становится текущей для очередной итерации. На практике признаком достижения стационарной точки служит достаточно малое изменение коор­динат точек, рассматриваемых на последовательных итераци­ях. Одновременно с этим координаты вектора Δ*f*(*x*(*q*)) должны быть близки к нулю.

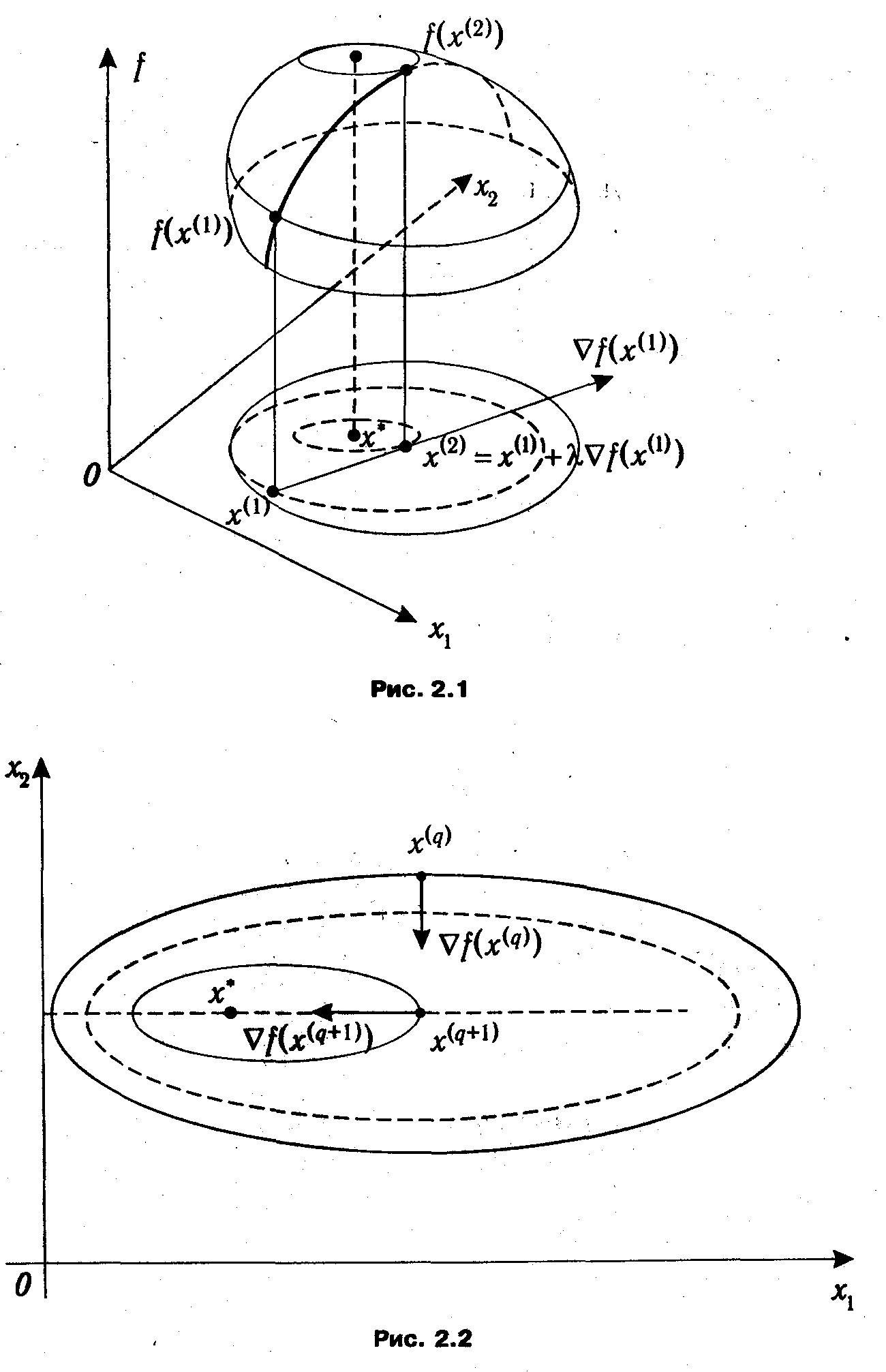
Метод дробления шага

Для нахождения шага λ в методе наискорейшего спуска требу­ется решить уравнение (2.13), которое может оказаться доста­точно сложным. Поэтому часто ограничиваются «подбором» такого значения λ, что φ(λ) > φ(0). Для этого задаются некото­рым начальным значением λ1, (например, λ1=l) и проверяют условие φ(λ1) >φ(0). Если оно не выполняется, то полагают

λ2 = 1/2 λ1

и т. д. до тех пор, пока не удается найти подходящий шаг, с кото­рым переходят к следующей точке *x*(*q*+1). Критерий завершения алгоритма, очевидно, будет таким же, как и в методе наискорей­шего спуска.

**2.1.4. Оптимизационные задачи для выпуклых функ­ций.** Общим недостатком рассмотренных выше методов безус­ловной оптимизации было, с одной стороны, то, что они позволя­ют отыскивать только точки, подозрительные на локальный экстремум, а с другой — то, что найденные решения могут суще­ственно зависеть от начального приближения. Поиск глобально­го оптимума подразумевает перебор найденных точек, который,

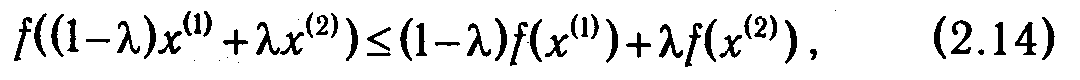


в случае градиентных методов, может быть осуществлен за счет подбора соответствующих начальных приближений *х*(0).

Однако существует один класс функций, для которых гради­ентные методы приводят к нахождению глобального оптимума. Это *выпуклые* функции.

☞ *Функция f(x)* = *f* (*x*1, *x*2, …, *xn*) *называется* **выпуклой** *в области D, если для любых двух точек* *х*(1)*,*

*х*(2) ∈ *D и любо­го* λ ∈ [0,1] *выполняется неравенство*

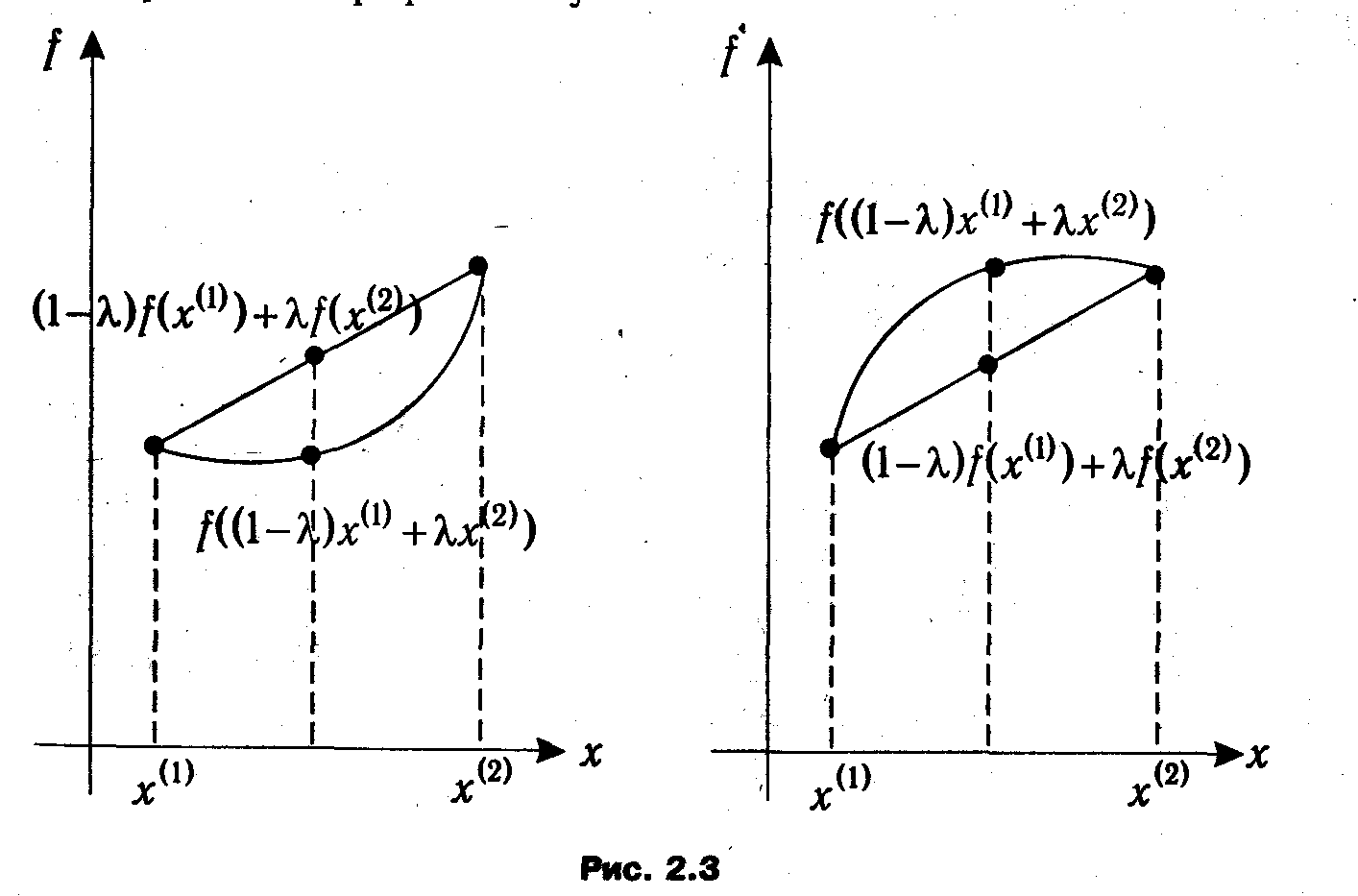


*если же*

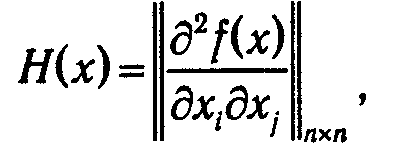


*то функция называется* вогнутой.

Геометрический смысл понятий выпуклости и вогнуто­сти для случая функции одной переменной представлен на *рис. 2.3.* Из него, в частности, видно, что график выпуклой функ­ции лежит *ниже* отрезка, соединяющего точки (*х*(1), *f*(*x*(1))) и (*х*(2), *f*(*x*(2))), а график вогнутой — *выше*.



Можно доказать, что *достаточным условием выпуклости* функции *f* (*x*1, *x*2, …, *xn*) является *положительная определен­ность* матрицы



называемой также *матрицей Гессе*, во всех точках *х* ∈ *D*. Со­ответственно, *достаточным условием вогнутости* являет­ся *отрицательная определенность* матрицы Гессе. В част­ности, для функций одной переменной достаточным условием выпуклости (вогнутости) является выполнение неравенства f*n*(*x*)≥0 (f*n*(*x*)≤0).

Как следует из геометрической интерпретации, для выпук­лой функции локальный экстремум, если он существует, совпа­дает с глобальным. Справедлива теорема.

|  |
| --- |
| ***Теорема 2.2.*** *Если f(x) выпуклая (вогнутая) на Rn функция и* *f*(*x*\*)*=*0*, то х\* — точка глобального мини­мума (максимума).* |

**Доказательство.**

Доказательство достаточно провести для случая вогнутой функции, т. к. для противоположного случая оно будет абсо­лютно аналогичным с точностью до знака.

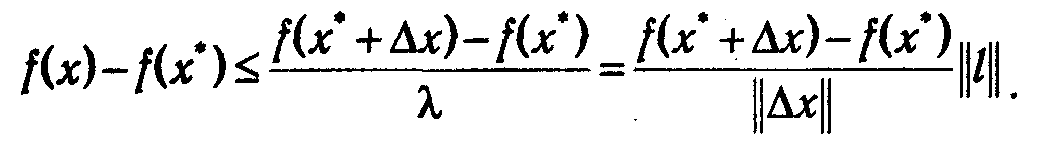
Пусть *х* — произвольная точка, отличная от точки *х*\*. Тогда, для любого λ ∈ [0,1], в силу вогнутости функции *f(x)* будет вы­полняться



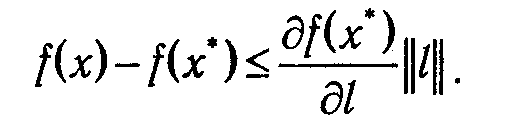
из чего следует



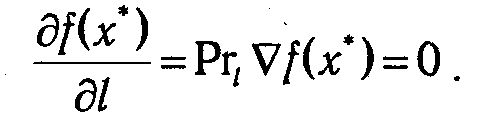
Если ввести вектор *l* = *х - х*\* и обозначить Δ*х* = λ(*x* – *х\**) = λ*l*, то длина вектора Δ*х* будет равна ||Δ*x*||=λ||*l*||. Следовательно,



Устремив λ → 0 и учитывая, что вектор *l* сонаправлен с Δ*х*, получим



По условию теоремы *f*(*x*\*)=0. Это означает, что для любо­го вектора *l* (а, стало быть, для любой точки *х*) согласно формуле, выражающей производную по направлению через градиент,



Следовательно, для любой точки *х*, не равной *х*\*, справедли­во неравенство *f(x)-f(x)≤* 0 <=> *f(x)≤ f(x*\**)*, т.е. *х*\* — точка гло­бального максимума. ✌

Поскольку выпуклые функции обладают столь «полезными» оптимизационными качествами, они занимают исключительно важное место в теории исследования операций. Соответствую­щий раздел получил название *выпуклого программирования*, а общая задача выпуклого программирования формулируется как проблема поиска максимума вогнутой (минимума выпук­лой) функции на выпуклом множестве.

**2.1.5. Метод допустимых направлений**. Данный метод также называется методом возможных направлений или же по имени автора—методом Зойтендейка, см. [16]. Его основную идею будет удобно продемонстрировать на примере ЗНП с огра­ничениями в форме неравенств:



В указанном методе так же, как и в градиентных методах, на­ходится последовательность точек *х*(0) *х*(1),..., *х*(*q*)..., таких, что *f*(*х*(*q*+1)) ≥ *f*(*x*(*q*)) \*. При этом переход от точки *x*(*q*)) к точке *х*(*q*+1)) происходит по некоторому выбранному направлению *s*(*q*) с ша­говым множителем λ*q*:



\* Так как в данных методах при переходе к очередной рассматривае­мой точке происходит улучшение значения целевой функции (в частно­сти, для задачи минимизации — уменьшение), их также называют *ре­лаксационными методами.*

По отношению к векторам, задающим направления переме­щения, вводятся два фундаментальных понятия.

☞ *Направление s называется* **допустимым** *(возможным) в точке x*(*q*) ∈ *D*, *если существует такое*

λ > 0, *что x*(*q*+1) = *x*(*q*) + λ*s* ∈ *D.*

☞ *Направление s называется* **прогрессивным** *в точке* *x*(*q*) ∈ *D*, *если существует такое* λ >0, *что*

*f*(*x*(*q*) + λ*s*)> > *f*(*x*(*q*)) *для задачи максимизации и f*(*x*(*q*) + λ*s*) < *f*(*x*(*q*)) *для задачи минимизации.*

На основе данных определений достаточно просто сформу­лировать критерий проверки оптимальности точки (так назы­ваемый критерий оптимальности в терминах допустимых и про­грессивных направлений):

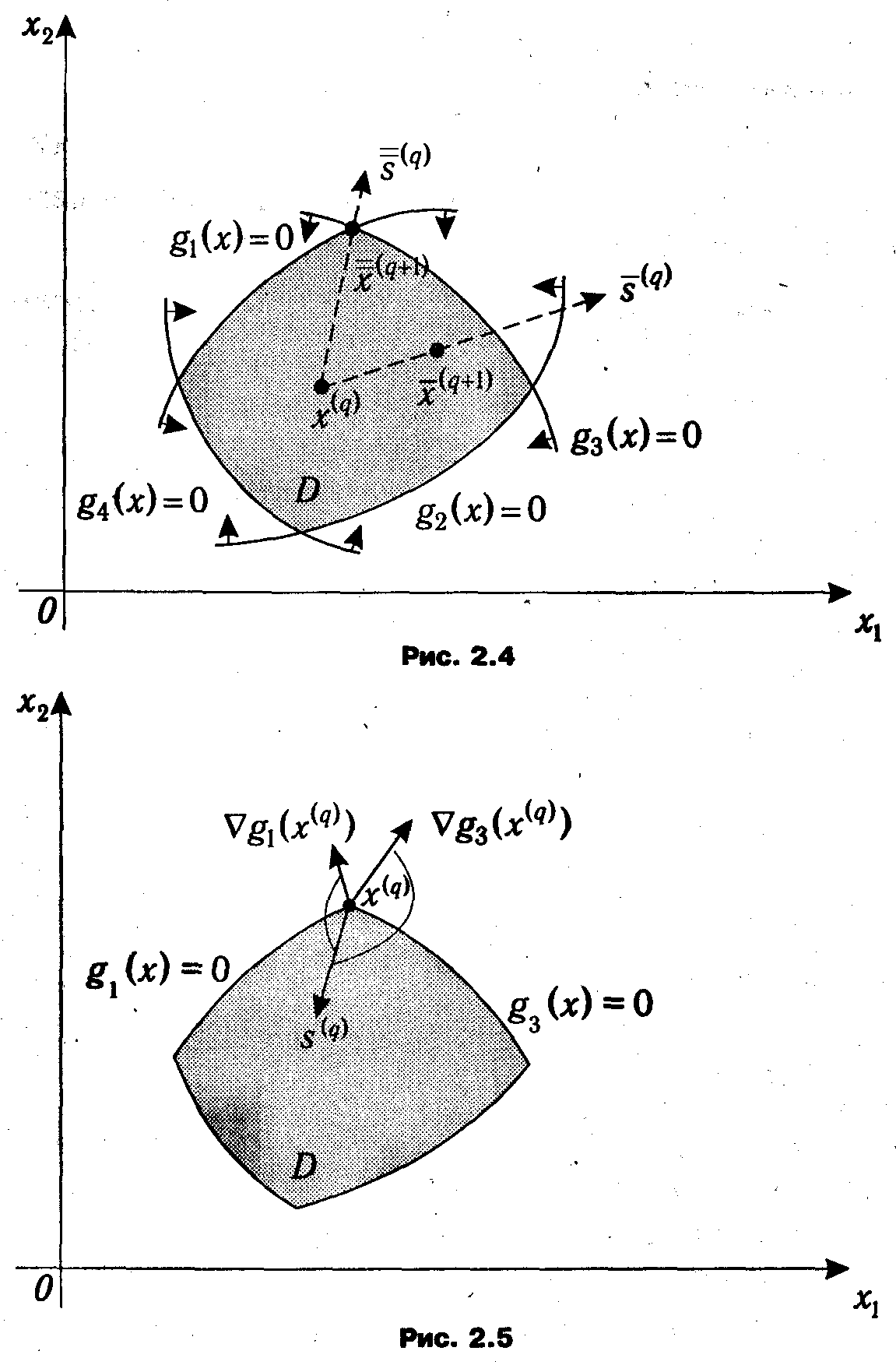
☞ *точка х\* является оптимальным планом задачи* (2.16), *если в ней ни одно допустимое*

*направление не является прогрессивным.*

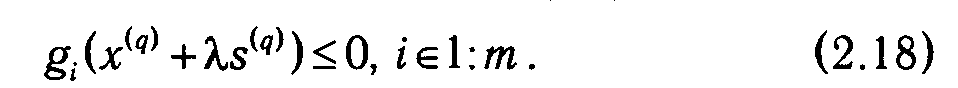
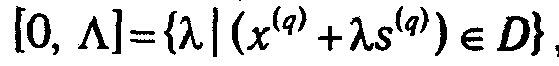
В алгоритме метода допустимых направлений правила выбо­ра точки *х*(*q*+1), к которой происходит очередной переход, раз­личаются в зависимости от того, где находится текущая точка *х*(*q*). Принципиально возможны две ситуации.

**1°. Точка** *х*(*q*) **лежит внутри области** *D*, т. е. для всех *i* ∈ 1: *m* *gi*(*х*(*q*)) < 0 (см. *рис. 2.4*). Очевидно, что для внутренней точки любое направление будет допустимым (при выборе достаточно малого шага), поэтому естественным представляется движение в сторону «гарантированного» возрастания значения функции, а именно в направлении градиента. Значит, для внутренней точ­ки *х*(*q*) целесообразно выбрать *s*(*q*) = *f*(*х*(*q*)).

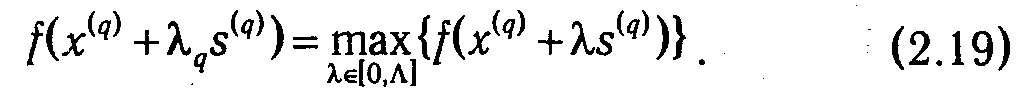
Шаговый множитель λ*q* выбирается так, чтобы, с одной сто­роны, новая точка *х*(*q*+1) принадлежала *D*, а с другой — значе­ние целевой функции в ней  *f*(*х*(*q*+1)) было как можно большим.



С этой целью сначала найдем промежуток [0, ] из условия для чего необходимо решить систе­му неравенств:



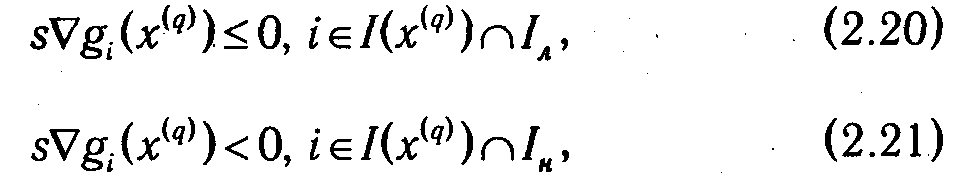
Зная промежуток [0, ], определяем значение шагового множителя λ*q* из условия максимизации значения функции в направлении *s(q)*:



Вновь найденная точка *x*(*q*+1) может находиться или внутри области *D*, или на ее границе. В первом случае (на *рис. 2.4* ему соответствует точка ** (*q*+1) переходим к началу данного пункта и повторяем вышеописанные действия, а во втором (точка **(*q*+1) на *рис. 2.4*) — действуем по рассматриваемой далее схеме.

**2°. Точка** *x*(*q*) **находится на границе области** (см. *рис. 2.5*). Это означает, что одно или несколько неравенств из системы ог­раничений задачи (2.16) выполняются как строгие равенства: *gi*(*x*(*q*)) = 0. Например, на *рис. 2.5* и *g*1(*x*(*q*)) = 0 и *g*3(*x*(*q*)) = 0.

Ограничение, которое в текущей точке выполняется как ра­венство, называют *активным.* Множество номеров активных ограничений в точке *x*(*q*) будем обозначать как *I*(*x*(*q*)). В приме­ре, изображенном на *рис. 2.5*, *I*(*x*(*q*)) = {1, 3}. Также из рисунка видно, что все допустимые направления, исходящие из точки *x*(*q*), должны образовывать тупые углы с векторами градиентов фун­кций, задающих активные ограничения в данной точке. По­следнее условие может быть выражено через задание ограниче­ний на значения скалярных произведений вектора направления *s* на градиенты функции ограничений:

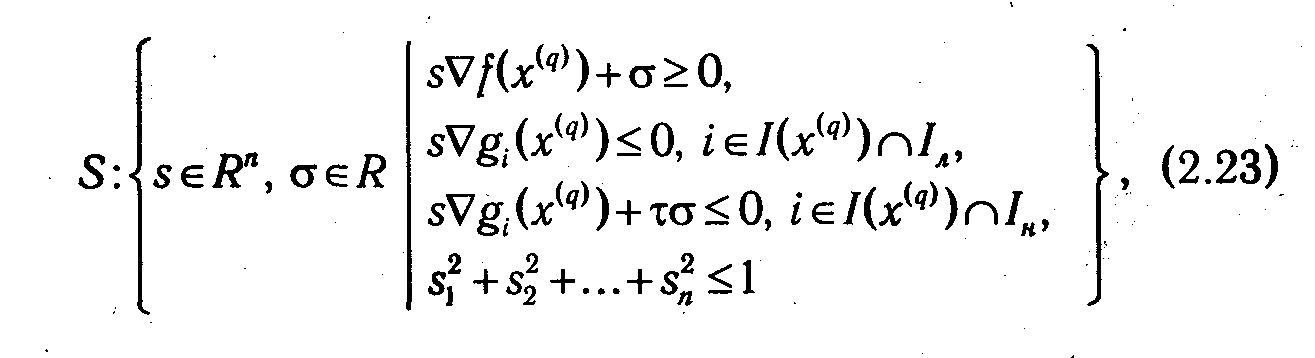


где *Iл* — множество номеров индексов линейных ограничений, *Iн* — множество номеров индексов нелинейных ограничений. Со­ответственно, *I*(*x*(*q*))◠*Iл* —номера линейных активных ограни­чений, а *I*(*x*(*q*))◠*Iн*— номера нелинейных активных ограниче­ний. Отличие условий (2.20) от условий (2.21) заключается в том, что в случае линейного ограничения направление, образу­ющее прямой угол с градиентом ограничивающей функции (т. е. их скалярное произведение равно нулю), будет заведомо допустимым, а в случае нелинейного ограничения — возможно, нет.

Все возможные направления в точке *x*(*q*) образуют так назы­ваемый *конус допустимых направлений*, и из них для следующего перехода, очевидно, нужно выбрать прогрессивное. Если такового не существует, то согласно сформулированному выше критерию точка *x*(*q*) является оптимальной! Для ускорения мак­симизации функции желательно, чтобы угол между искомым допустимым прогрессивным направлением *s*(*q*) и градиентом це­левой функции ∇*f*(*x*(*q*)) был как можно меньше или, что то же самое, как можно большей была бы проекция *s* на ∇*f*(*x*(*q*)) (при условиях нормировки вектора *s*(*q*)). Иными словами, желатель­но, чтобы неравенство *s*(*q*)∇*f*(*x*(*q*))+σ≥0 выполнялось при ми­нимально возможном σ ∈ *R.* Тогда задачу отыскания наилучше­го допустимого прогрессивного направления *s*(*q*) можно свести к задаче минимизации параметра σ:



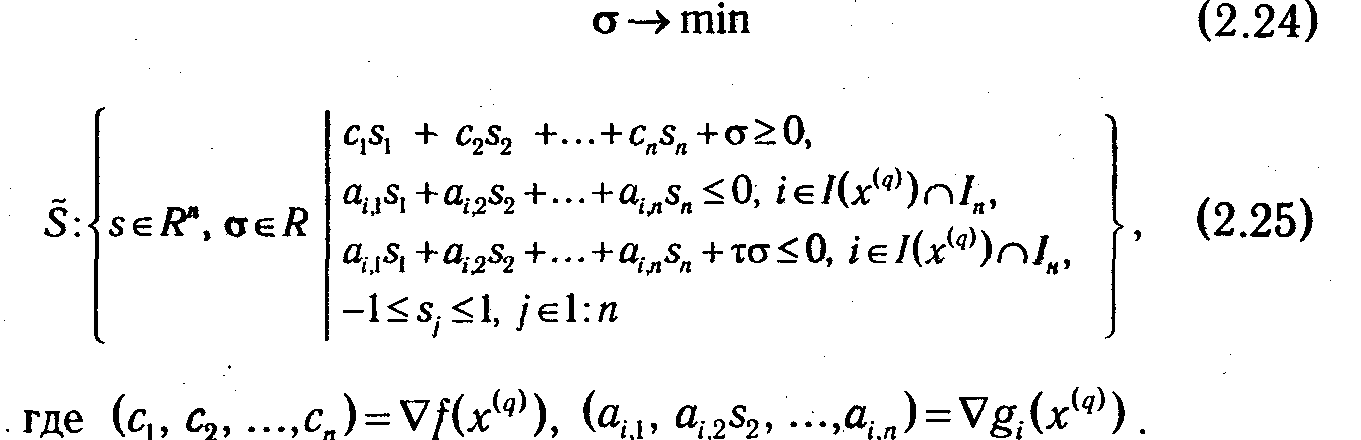
при условиях



где *s*12 *+s*22 *+...+sn*2, ≤ l —условие нормировки, обеспечивающее ограниченность решения;

τ — некоторое достаточно малое число, характеризую­щее «строгость» выполнения неравенств.

В отличие от всех остальных, последнее условие в системе (2.23) является нелинейным, что существенно усложняет про­цесс решения задачи (2.22)-(2.23). Поэтому на практике для определения направления *s*(*q*) (возможно, не лучшего) перехо­дят от данной задачи к задаче линейного программирования пу­тем замены указанных выше условий нормировки на ограниче­ния вида –l ≤ *sj,* ≤ 1:

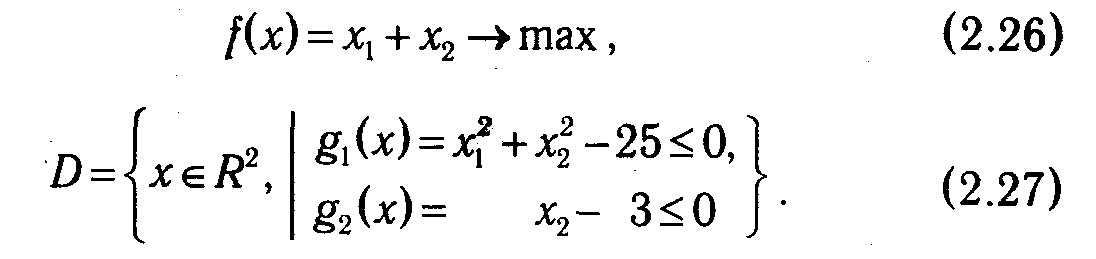


После того как прогрессивное направление *s*(*q*) найдено, ша­говый множитель определяется по методу, описанному в п. 1°.

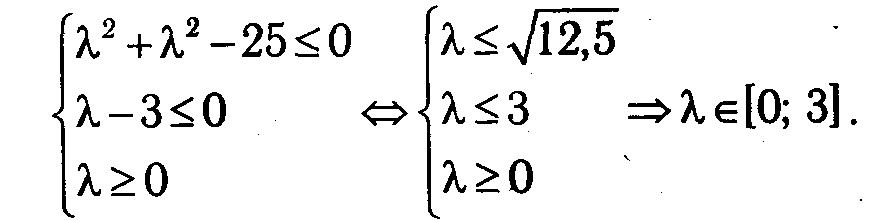
В заключение отметим, что при практических расчетах алго­ритм завершается при достижении такой точки *х*\*, в которой |∇*f*(*x*\*)|<ε, где ε —достаточно малое число.

Представляется полезным обратить внимание читателя и на то, что применяемый для решения задач линейного програм­мирования симплекс-метод может быть рассмотрен как част­ный случай метода допустимых направлений. В частности, этап выбора столбца, вводимого в очередной базис, соответ­ствует определению допустимого прогрессивного направле­ния. Более подробно о такой концепции симплекс-метода можно прочесть в [1].

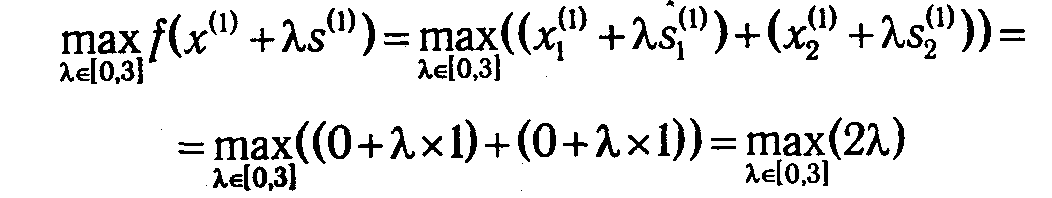
**2.1.6. Пример решения ЗНП методом допустимых на­правлений**. Рассмотрим процесс применения метода допусти­мых направлений на конкретном примере. Пусть дана ЗНП:



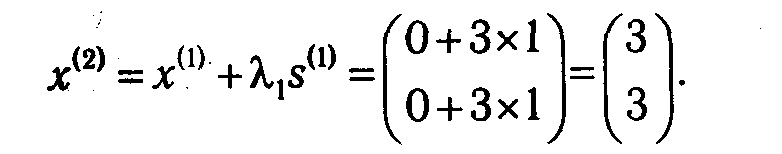
**Итерация 1**. В качестве начального приближения возьмем точку *х*(1) = (0,0). Нетрудно заметить, что она удовлетворяет системе неравенств (2.27), т. е. *х*(1) ∈ *D*. Для *х*(1) все неравенства выполняются как строгие, т. е. множество индексов активных ограничений *I*(*х*(1)) = ∅. Следовательно, в *х*(1) любое направле­ние является допустимым, и нам остается определить, с каким шагом λ1 можно двигаться вдоль градиента целевой функции *s*(1) = ∇*f*(*x*(1))=(1, 1). Система неравенств типа (2.18), из решения которых определяется интервал допустимых значений для λ, для данной задачи примет вид:



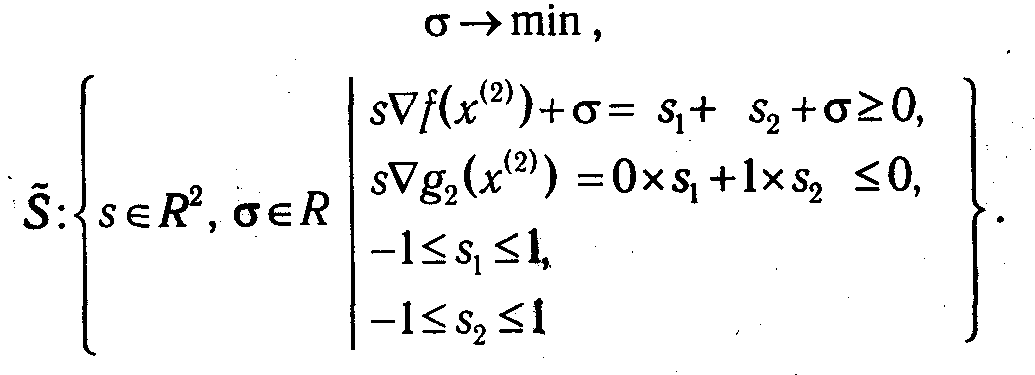
Тогда



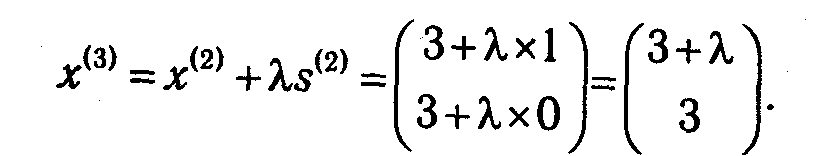
достигается при λ1 = 3 . Отсюда получаем следующую точку



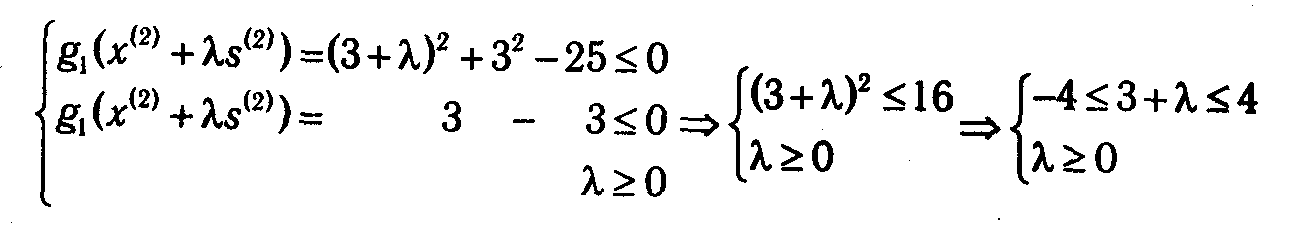
**Итерация 2.** Путем подстановки координат точки *x*(2) в (2.27) определим множество активных ограничений в точке *x*(2): *I*(*x*(2))={2}. Соответственно, задача (2.24) - (2.25), кото­рую требуется решить для определения допустимого прогрес­сивного направления *s*(2) с учетом того, что ∇*f*(*x*(2))=(1, 1) и ∇*g*2(*x*(2))= (0, 1) примет вид:



В данном случае оптимальный план ЗЛП находится доволь­но просто и равен (σ, *s*1, *s*2)\* =(-1, 1, 0). Отбросив дополнитель­ную переменную σ, получаем вектор *s*(2) = (1,0), т. е. очередная точка будет определяться как

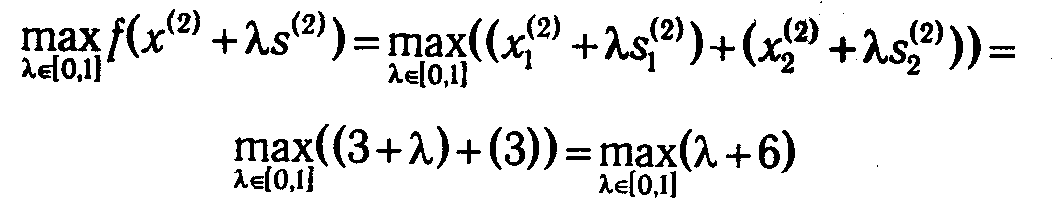


Действуя по аналогии с предыдущей итерацией, для опреде­ления промежутка допустимых значений шагового множителя λ составляем систему неравенств (2.18):



Окончательно имеем λ ∈ [0; 1].

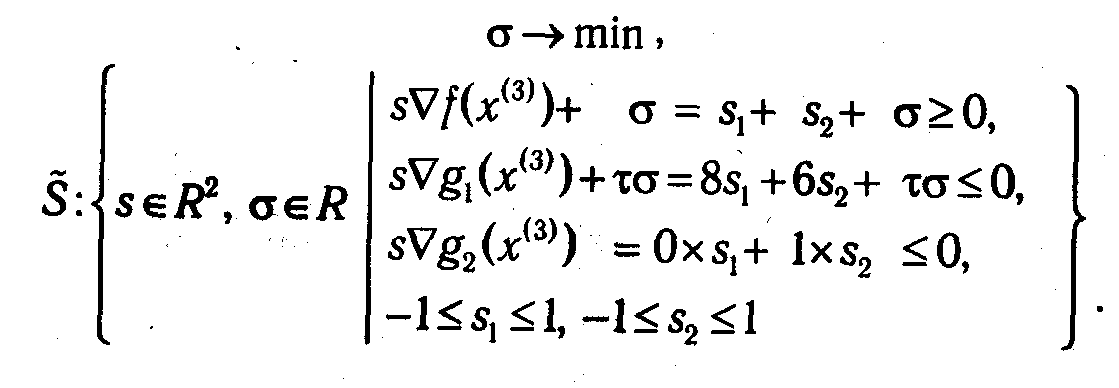
Тогда



достигается при λ2=1. Отсюда получаем следующую точку *x*(3) = (3,4).

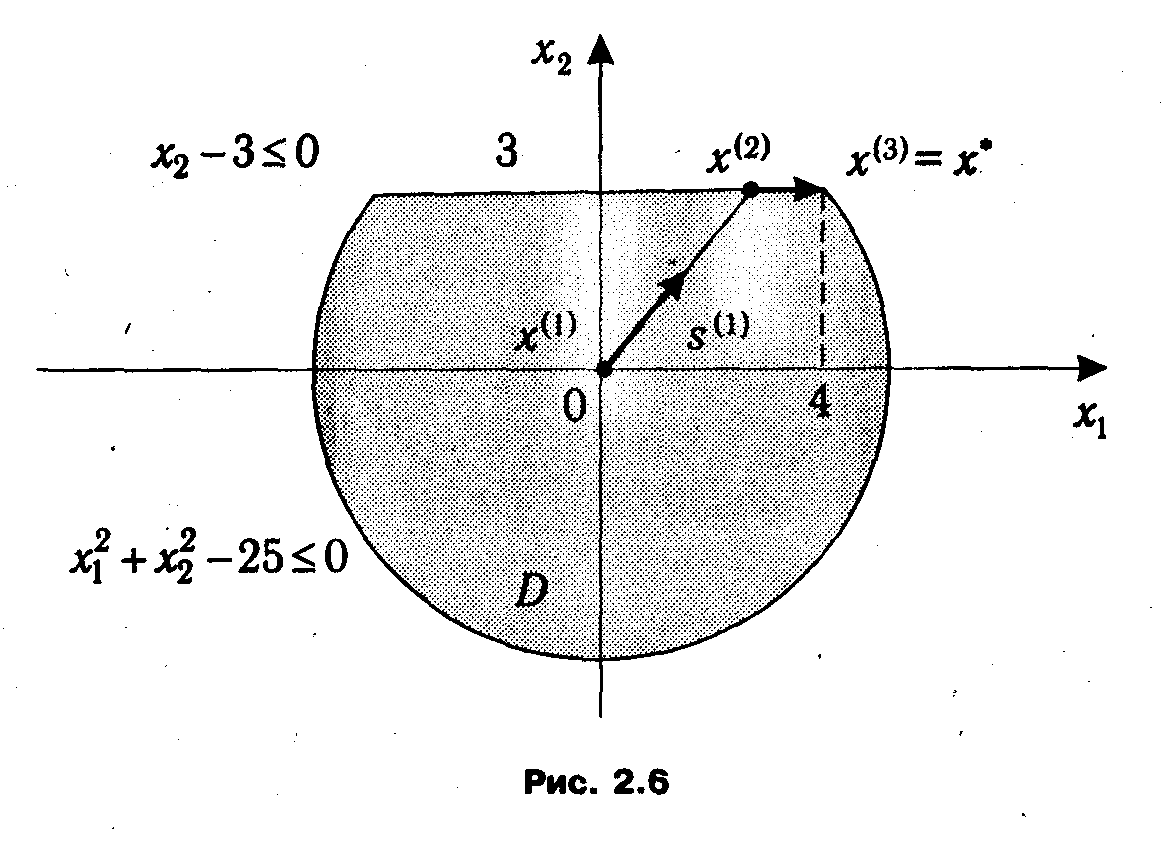
**Итерация 3.** В точке *x*(3) множество активных ограничений будет иметь вид *I*(*x*(3))={1,2}. Найдем значения градиентов: ∇*f*(*x*(3)) = (1, 1), ∇*g*1(*x*(3)) = (2*x*1(3), 2*x*2(3)) = (8, 6) и ∇*g*2(*x*(2)) = (0,1).

Задача определения допустимого прогрессивного направле­ния (2.24)-(2.25) будет иметь вид:



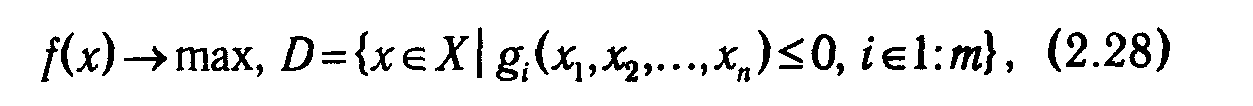
Значение τ из практических соображении следует брать до­статочно малым, например τ = 0,001. Опуская решение данной задачи, приведем интересующие нас компоненты ее оптималь­ного плана: *s*(3) = (0,0). Итак, не существует прогрессивного на­правления, исходящего из точки *х*(3) Таким образом, оптималь­ный план рассматриваемой задачи (2.26)-(2.27) *х*\* = (4,3), а максимальное значение целевой функции *f*\* = *х*1(3) + *х*2(3) = 7.

Графическая иллюстрация проведенного процесса решения представлена графически на *рис. 2.6.*



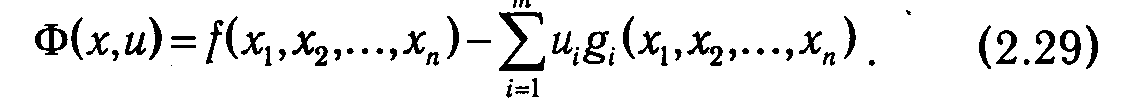
## 2.2. ДВОЙСТВЕННОСТЬ В НЕЛИНЕЙНОМ ПРОГРАММИРОВАНИИ

**2.2.1. Понятие седловой точки.** В настоящем параграфе мы кратко остановимся на некоторых фундаментальных моментах теории нелинейного программирования. Отправной точкой для них является распространение метода Лагранжа для решения ЗНП с ограничениями в форме неравенств:



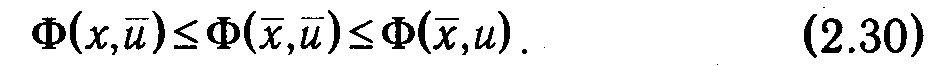
где *X* — некоторая область в пространстве *Rn*.

По аналогии с п. 2.1.2 определим для задачи (2.28) функцию Лагранжа:

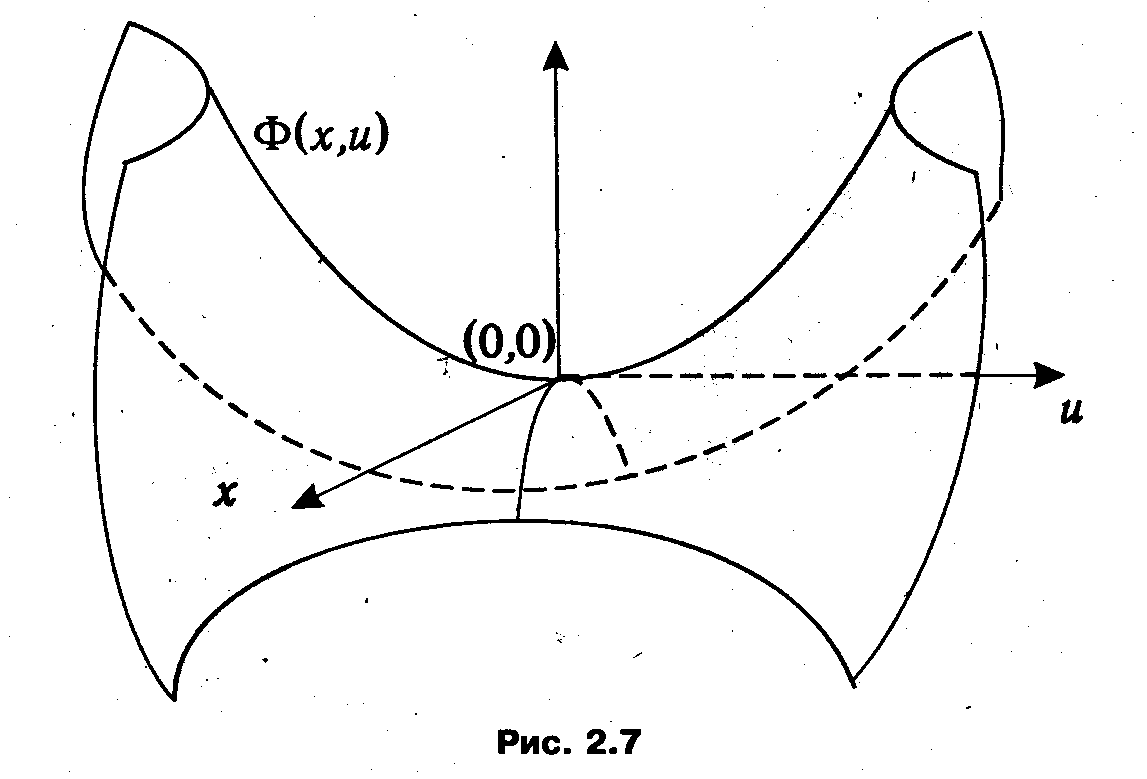


☞ *Пара векторов* (*х, u*) *называется* **седловой точкой** *фун­кции Ф*(*х,* *и*) *в некоторой области X* x *U*, *если*

*для любых x ∈ X и u ∈ U*



Неравенства (2.30) также называют *неравенствами седловой точки*.

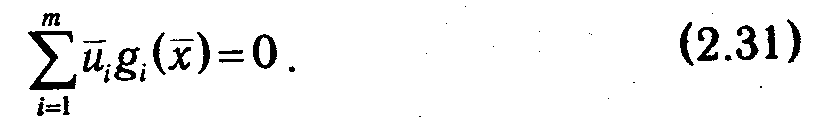


В качестве примера седловой точки может быть приведена точка (0, 0) для функции Ф(*х,и*) = -*х*2 + *и*2, определенной на множестве *R* х *R*. В самом деле, Ф(0,0)=0, Ф(*х*,0)=-*х*2, Ф(0, *и*) = *и*2, а для любых *x*∊*R* и *u*∊*R* выполняются неравенства –*х*2≤0 и 0 ≤ *и*2.

На *рис. 2.7* изображен график функции Ф(*х,и*) (гиперболи­ческий параболоид), и, как видно, в окрестности точки (0,0) он действительно по форме напоминает седло, чем и объясняется происхождение соответствующего термина.

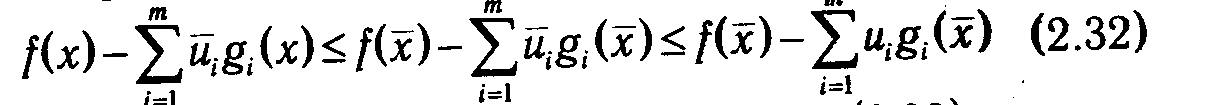
**2.2.2. Теорема Куна—Таккера.** Центральное место в те­ории нелинейного программирования занимает теорема Куна— Таккера, которая связывает решение ЗНП с наличием седловой точки у соответствующей функции Лагранжа.

|  |
| --- |
| **Теорема 2.3. (Достаточное условие экстремума).**  *Если (х, и) — седловая точка функции Лагранжа, в области x*∊*X⊇D*, *и≥*0,*то х является* *оптимальным планом задачи* (2.28), *причем справедливо так называ­емое* **правило дополняющей нежесткости:** |

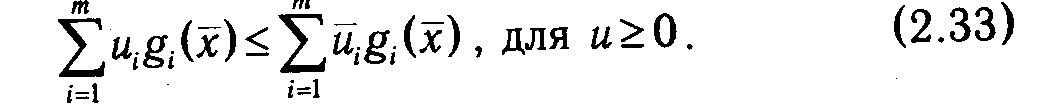


**Доказательство.**

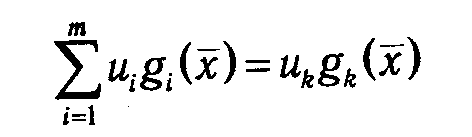
По определению седловой точки



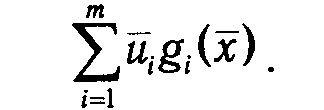
при всех *x*∊*X*, *и≥*0. Из второго неравенства в (2.32) следует, что



Однако (2.33) может иметь место только тогда, когда *gi*(*x*)≤0 при всех *i*∊1:*m*. Действительно, если существует та­кое *k,* что *gk*(*x*)>0, то, положив *иi*=0 для всех *i ≠ k* и выбрав достаточно большое *иk* > 0, можно добиться того, что значение

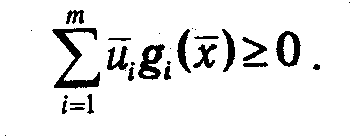


окажется больше постоянного выражения

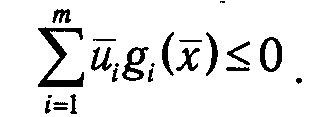


Из того, что для всех *i*∊1:*m* выполняются неравенства *gi*(*x*)≤0, следует, что *х* является допустимым планом задачи (2.28).

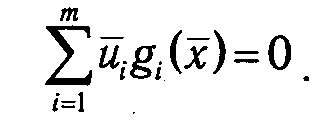
Если в левую часть неравенства (2.33) подставить значения *ui* = 0, *i*∊1:*m*, то получим, что



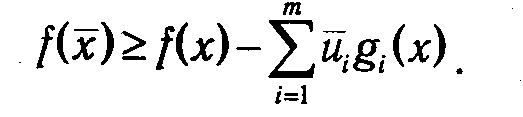
Вместе с тем из того что, *gi*(*x*)≤0 и *ui* ≥0, следует оценка



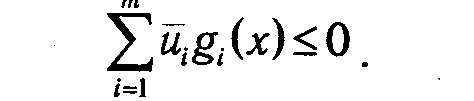
Совместное рассмотрение последних двух неравенств приводит к правилу дополняющей нежесткости в точке *х*:



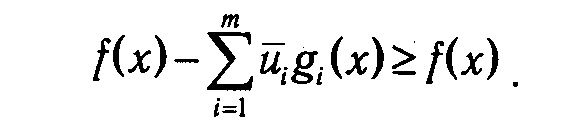
Тогда на основании левой части неравенства седловой точки (2.32) имеем, что для всех *х*∊*Х* (в том числе и для *х*∊*D*)



Но условию ЗНП для любых *х*∊*D* верны неравенства *gi*(*x*)≤0, что, в сочетании с условием *ui* ≥0, позволяет записать



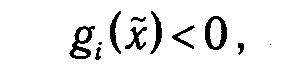
Значит,



Окончательно получаем, что для любых *х*∊*D* справедливо со­отношение *f(x)*≥*f(x),* т. е. *х* — оптимальный план задачи (2.28). ✌

Утверждение, обратное теореме (2.3), т. е. необходимое условие экстремума в ЗНП, оказывается верным только при вы­полнении дополнительных условий, которым должна удовлет­ворять задача (2.28). Важнейшим из них является так называе­мое **условие регулярности Слейтера:**

* *Говорят, что функция gi (х), задающая ограничение в за­даче (2.28), удовлетворяет условию регулярности Слей­тера, если существует такая точка , принадлежащая области допустимых планов D, что*



т. е. ****** является внутренней точкой относительно ограничения *gi(x).* Поэтому данное условие также называют условием *теле­сности.*

Вообще говоря, существуют разные варианты необходимо­го условия Куна—Таккера. Приведем один из них.

|  |
| --- |
| ***Теорема 2.4.* (Необходимое условие наличия экст­ремума).**  *Если (D, f) является задачей выпуклого программиро­вания с решением х, ее целевая функция f(x) и функ­ции ограничений gi(x) — дифференцируемы, нелиней­ные ограничения в форме неравенств удовлетворяют условию регулярности Слейтера, то существует та­кой вектор и* ≥ *0, что (х,и) — седловая точка функции Лагранжа* Ф*(х,и).* |

Мы не будем здесь приводить доказательство теоремы (2.4), которое является довольно сложным. Заинтересованный чита­тель может найти его в таких источниках, как [1, 13].

Значение теоремы Куна—Таккера состоит в том, что она по­зволяет связать процесс решения оптимизационной задачи с поиском седловых точек функции Лагранжа, т. е., грубо гово­ря, с максимизацией этой функции по *х* и минимизацией по *и.*

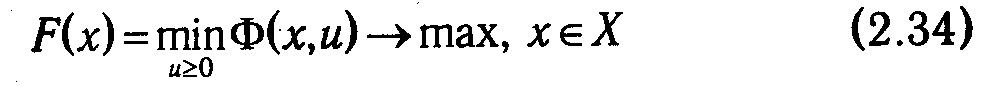
Определим *F(x)* как функцию, ставящую в соответствие каждому значению *х* минимальное значение функции Ф(*х,и*) по *и*:



и по аналогии



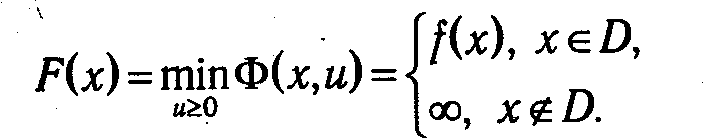
Рассмотрим задачу отыскания максимума функции *F(x)*



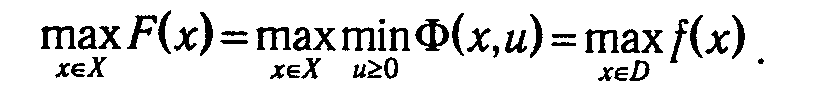
и задачу минимизации G(u)



Очевидно, что



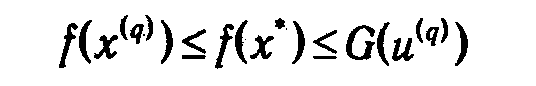
Отсюда следует, что максимум *F(x)* находится в допустимой области *D* и совпадает с максимумом целевой функции *f(x)* зада­чи (2.28):



Таким образом, задача (2.34), в определенном смысле, рав­носильна (2.28). Аналогичные выводы могут быть получены и для (2.35). Задачи (2.34) и (2.35) образуют двойственную пару. Как нетрудно догадаться, данное отношение является обобще­нием отношения двойственности для задач линейного програм­мирования. Соответственно, при определенных условиях пара двойственных задач нелинейного программирования обладает свойствами, аналогичными свойствам двойственных линейных задач. В частности, при любых *х*∊*Х*, *и*≥0



Условие (2.36) находит широкое применение при построе­нии оценок в итеративных методах решения оптимизационных задач. Например, если имеется возможность приблизительно решить прямую и двойственную задачи и получить последова­тельности приближений {*х*(*q*)} и {*и*(*q*)}, то с помощью нера­венств вида



можно определить момент остановки вычислительной про­цедуры.

В заключение отметим, что возможен вариант вывода выра­жений для целевых функций и ограничений пары двойственных задач линейного программирования из общего определения от­ношения двойственности для нелинейных задач. Также отме­тим, что в процессе формирования нелинейных двойственных задач существует большая неоднозначность: их вид можно ва­рьировать, включая в множество *Х* часть ограничений *gi(x)*≤0.

## КЛЮЧЕВЫЕ ПОНЯТИЯ

* Общая задача нелинейного программирования.
* Условная и безусловная оптимизация.
* Прямые и непрямые методы решения оптимизационных задач.
* Стационарная точка.
* Градиентные методы.
* Метод наискорейшего спуска и методы дробления шага.
* Выпуклая и вогнутая функции.
* Матрица Гессе.
* Достаточное условие выпуклости (вогнутости).
* Задача выпуклого программирования.
* Допустимое направление.
* Прогрессивное направление.
* Седловая точка.
* Теорема Куна—Таккера.
* Условие регулярности Слейтера.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

2.1. При каких условиях оптимизационная задача может быть отнесена к классу нелинейных?

2.2. Приведите пример экономической модели, сводящейся к задаче нелинейного программирования.

2.3. Перечислите основные трудности, возникающие в про­цессе решения задачи нелинейного

программирования.

2.4. Какой смысл вкладывается в понятие «условная оптими­зация»?

2.5. Для чего предназначен метод множителей Лагранжа и в чем он состоит?

2.6. Какая точка называется стационарной?

2.7. Какие принципиальные этапы входят в градиентные ме­тоды?

2.8. Для решения каких задач предназначены метод наиско­рейшего спуска и метод дробления шага?

2.9. Дайте определение выпуклой (вогнутой) функции.

2.10. Сформулируйте достаточное условие выпуклости (во­гнутости) функции.

2.11. В чем заключена специфика задач выпуклого программи­рования?

2.12. Перечислите основные этапы, входящие в метод допусти­мых направлений.

2.13. Сформулируйте задачу, которая должна быть решена при определении шага в методе

допустимых направлений.

2.14. Исходя из каких соображений определяется допустимое прогрессивное направление?

2.15. Какое условие используется для определения оптималь­ности текущей точки в методе

допустимых направлений?

2.16. Дайте определение седловой точки. Приведите пример функции, имеющей седловую точку.

2.17. Сформулируйте необходимое и достаточное условия тео­ремы Куна—Таккера. Какое значение

они имеют для ре­шения задач нелинейного программирования?

2.18. В чем состоит условие регулярности Слейтера? Поясните его содержание.

2.19. Какое условие получило название «правила дополняю­щей нежесткости»?

2.20. Приведите пример пары двойственных задач нелинейного программирования.

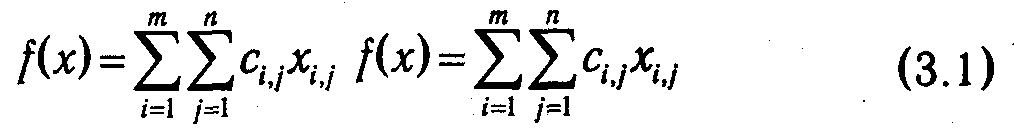
2.21. Какие свойства пары нелинейных двойственных задач мо­гут быть применены для их решения?

# ГЛАВА 3. ТРАНСПОРТНЫЕ И СЕТЕВЫЕ ЗАДАЧИ

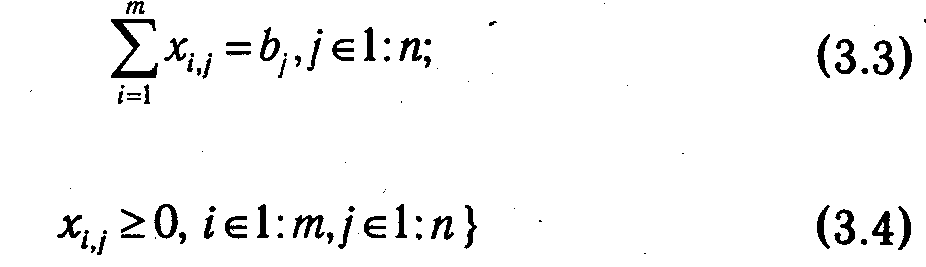
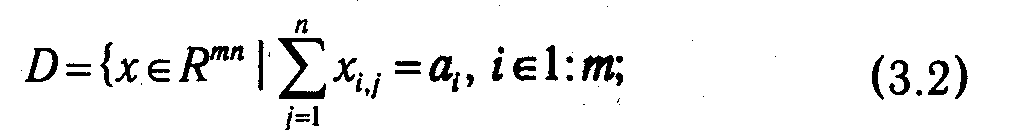
Методы, рассмотренные в предыдущих главах, носили универ­сальный характер и были предназначены для решения очень ши­рокого круга линейных и нелинейных задач. Платой за такую универсальность зачастую является снижение их эффективно­сти, выражающееся в медленной сходимости, высоком объеме вычислений и т. п. В то же время существуют такие классы задач, для которых в силу их специфики разработаны более простые методы решения. Некоторых из них мы коснемся в этой главе.

## 3.1. ТРАНСПОРТНАЯ ЗАДАЧА И МЕТОДЫ ЕЕ РЕШЕНИЯ

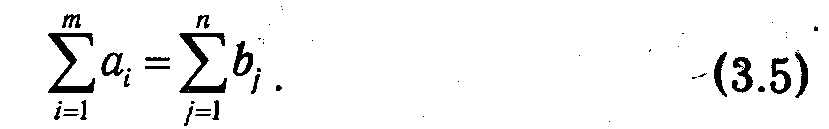
**3.1.1. Транспортная задача в матричной постановке и ее свойства.** Вернемся к *транспортной задаче в матричной, постановке*, о которой мы уже упоминали при рассмотрении вопросов построения математических моделей. Напомним, что данная задача сводится к определению такого плана перевозок некоторого продукта из пунктов его производства в пункты по­требления (║*xi,j*║*m*x*n*), который минимизирует целевую функцию



на множестве допустимых планов



при соблюдении условия баланса

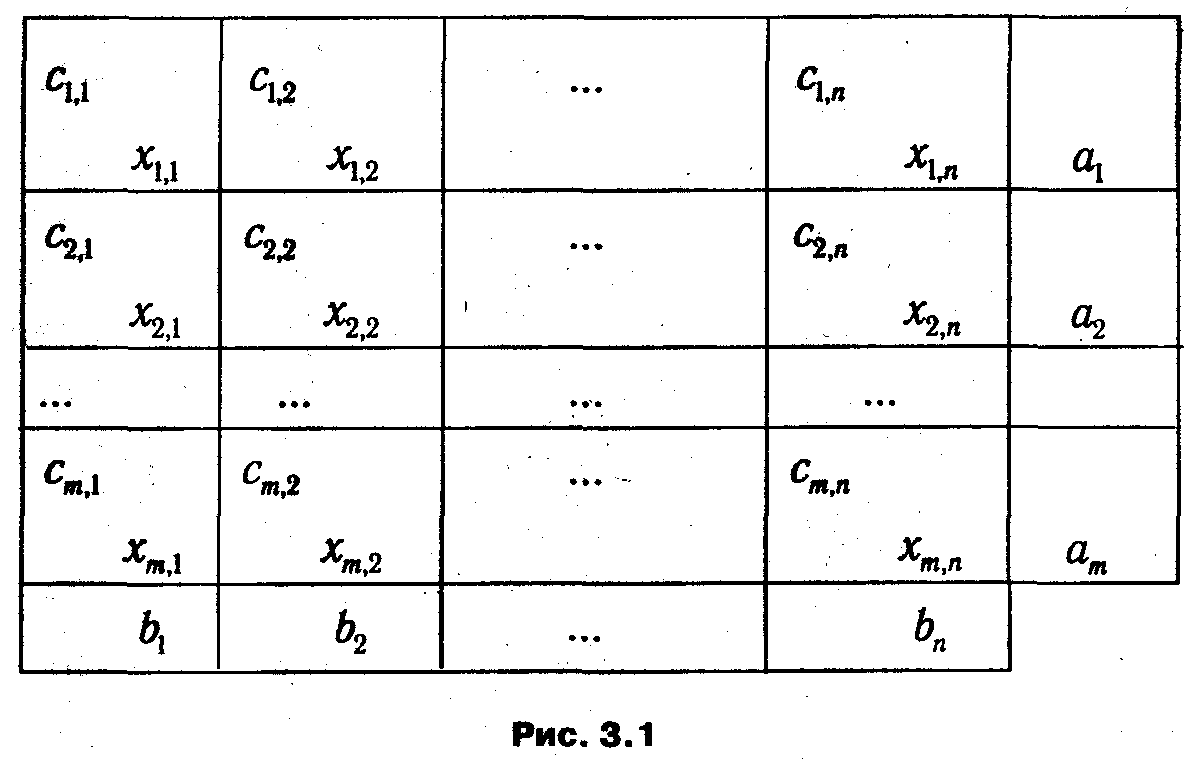


Транспортная задача является представителем класса задач линейного программирования и поэтому обладает всеми каче­ствами линейных оптимизационных задач, но одновременно она имеет и ряд дополнительных полезных свойств, которые позво­лили разработать специальные методы ее решения.

Если привести условия транспортной задачи к канонической форме задачи линейного программирования, то матрица задачи будет иметь размерность (*m*+*n*) х *mn*. Матрицы систем уравне­ний в ограничениях (3.2) и (3.3) имеют ранги, равные соответ­ственно *m* и *n*. Однако, если, с одной стороны, просуммировать уравнения (3.2) по *m*, а с другой — уравнения (3.3) по *n*, то в силу (3.5) получим одно и то же значение. Из этого следует, что одно из уравнений в системе (3.2)-(3.3) является линейной комбинацией других. Таким образом, ранг матрицы транспорт­ной задачи равен *m*+*n*-1, и ее невырожденный базисный план должен содержать *m*+*n*-1 ненулевых компонент.

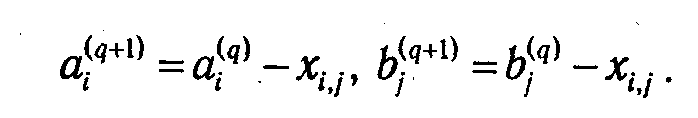
Процесс решения транспортной задачи удобно оформлять в виде последовательности таблиц, структура которых представ­лена на *рис.3.1.*

Строки *транспортной таблицы* соответствуют пунктам производства (в последней клетке каждой строки указан объем запаса продукта *аi*), а столбцы — пунктам потребления (послед­няя клетка каждого столбца содержит значение потребности *bj*). Все клетки таблицы (кроме тех, которые расположены в нижней строке и правом столбце) содержат информацию о пе­ревозке из *i*-го пункта в *j*-й: в левом верхнем углу находится



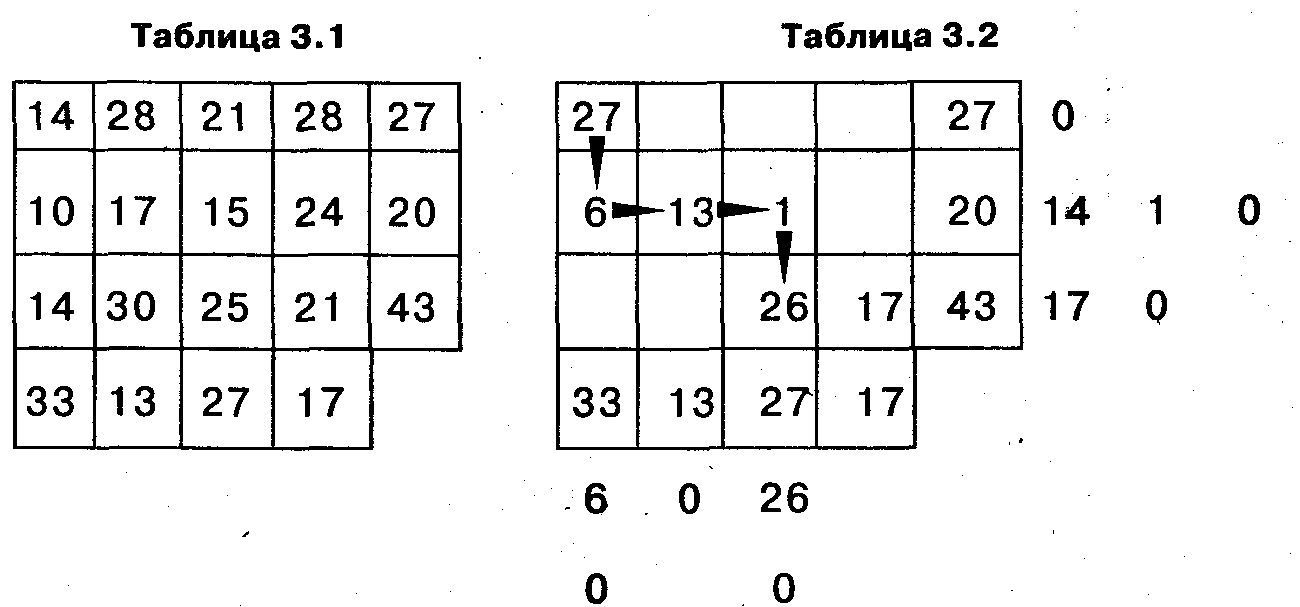
цена перевозки единицы продукта, а в правом нижнем — значе­ние объема перевозимого груза для данных пунктов. Клетки, которые содержат нулевые перевозки (*хi,j* = 0), называют *сво­бодными*, а ненулевые — *занятыми* (*xi,j* >0).

**3.1.2. Построение исходного допустимого плана в транс­портной задаче.** По аналогии с другими задачами линейного программирования решение транспортной задачи начинается с построения допустимого базисного плана. Наиболее простой способ его нахождения основывается на так называемом *мето­де северо-западного угла*. Суть метода состоит в последова­тельном распределении всех запасов, имеющихся в первом, вто­ром и т. д. пунктах производства, по первому, второму и т. д. пунктам потребления. Каждый шаг распределения сводится к попытке полного исчерпания запасов в очередном пункте про­изводства или к попытке полного, удовлетворения потребно­стей в очередном пункте потребления. На каждом шаге *q* вели­чины текущих нераспределенных запасов обозначаются *аi*(*q*), а текущих неудовлетворенных потребностей — *bj*(*q*). Построение допустимого начального плана, согласно методу северо-запад­ного угла, начинается с левого верхнего угла транспортной таб­лицы, при этом полагаем *аi*(0) = *аi*, *bj*(0) = *bj*. Для очередной клетки, расположенной в строке *i* и столбце *j*, рассматриваются зна­чения нераспределенного запаса в *i*-ом пункте производства и неудовлетворенной потребности *j*-ом пункте потребления, из них выбирается минимальное и назначается в качестве объема перевозки между данными пунктами: *xi,j* = min{*аi*(*q*), *bj*(*q*)}. После этого значения нераспределенного запаса и неудовлетворенной потребности в соответствующих пунктах уменьшаются на дан­ную величину:



Очевидно, что на каждом шаге выполняется хотя бы одно из равенств: *аi*(*q*+1) = 0 или *bj*(*q*+1) = 0 . Если справедливо первое, то это означает, что весь запас *i*-го пункта производства исчерпан и необходимо перейти к распределению запаса в пункте произ­водства *i* +1, т. е. переместиться к следующей клетке вниз по столбцу. Если же *bj*(*q*+1) = 0, то значит, полностью удовлетворе­на потребность для *j*-го пункта, после чего следует переход на клетку, расположенную справа по строке. Вновь выбранная клетка становится текущей, и для нее повторяются все пере­численные операции.

Основываясь на условии баланса запасов и потребностей (3.5), нетрудно доказать, что за конечное число шагов мы полу­чим допустимый план. В силу того же условия число шагов ал­горитма не может быть больше, чем *m*+*n*-1, поэтому всегда останутся свободными (нулевыми) *mn*-(*m+n*-1) клеток. Следовательно, полученный план является базисным. Не ис­ключено, что на некотором промежуточном шаге текущий не­распределенный запас оказывается равным текущей неудовлет­воренной потребности (*аi*(*q*) = *bj*(*q*)). В этом случае переход к следующей клетке происходит в диагональном направлении (одновременно меняются текущие пункты производства и по­требления), а это означает «потерю» одной ненулевой компо­ненты в плане или, другими словами, вырожденность построен­ного плана.

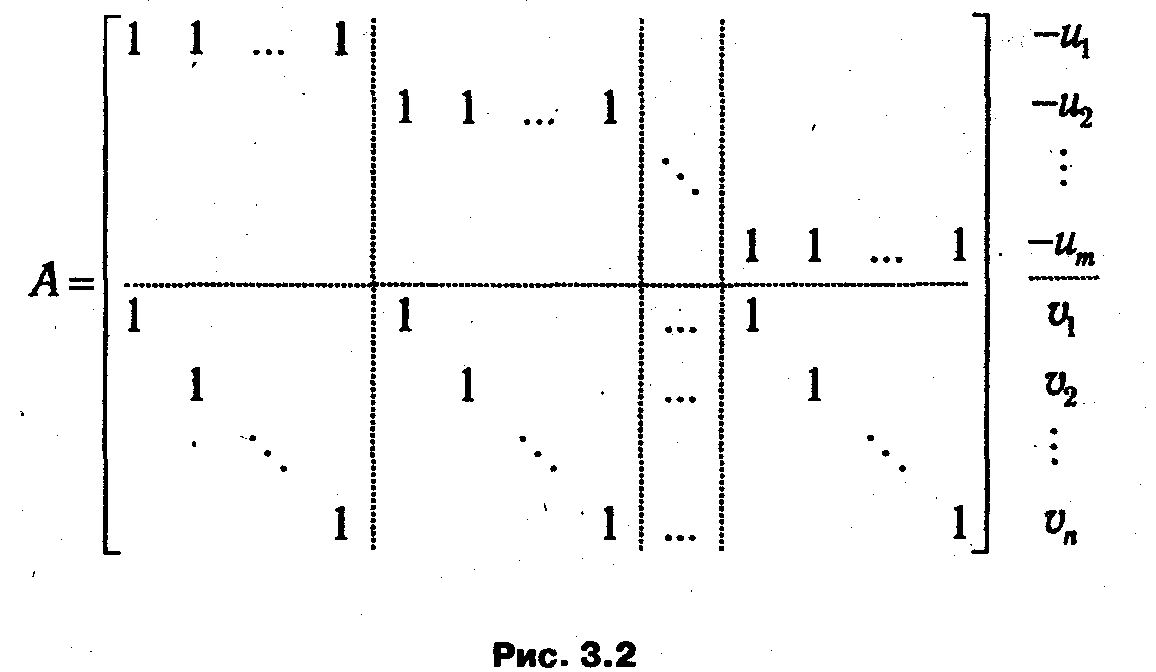


Рассмотрим применение метода северо-западного угла на конкретном примере. Транспортная таблица 3.1 содержит ус­ловия некоторой задачи, а в табл. 3.2 показан процесс поиска допустимого плана, включая последовательное изменение объема нераспределенных запасов и неудовлетворенных по­требностей. Стрелки отражают траекторию перехода по клет­кам транспортной таблицы, а цифры, находящиеся за ее преде­лами, — текущие нераспределенные остатки после назначения объема для очередной клетки.

Особенностью допустимого плана,, построенного методом северо-западного угла, является то, что целевая функция на нем принимает значение, как правило, далекое от оптимально­го. Это происходит потому, что при его построении никак не учитываются значения *сi,j*. В связи с этим на практике для по­лучения исходного плана используется другой способ — *ме­тод минимального элемента*, в котором при распределении объемов перевозок в первую очередь занимаются клетки с наи­меньшими ценами.

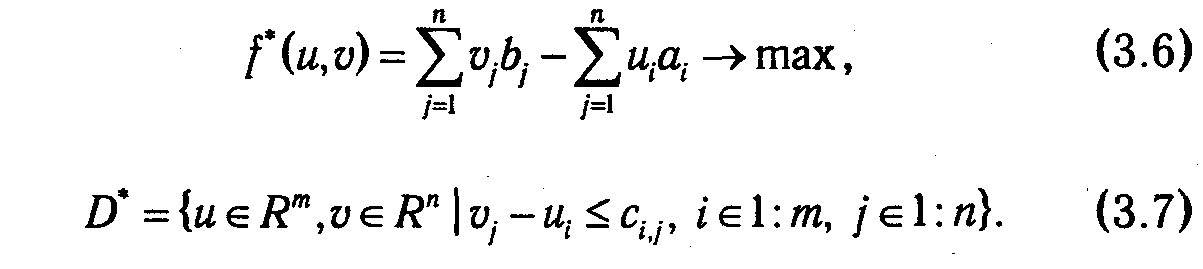
**3.1.3. Критерий оптимальности.** Рассмотрим более по­дробно структуру матрицы транспортной задачи. Схематично она показана на *рис. 3.2.*

Из него видно, что матрица имеет размерность (*m*+*n*) х *mn*, состоит из нулей и единиц и распадается на две группы однотип­ных блоков. Первая (верхняя) соответствует ограничениям на



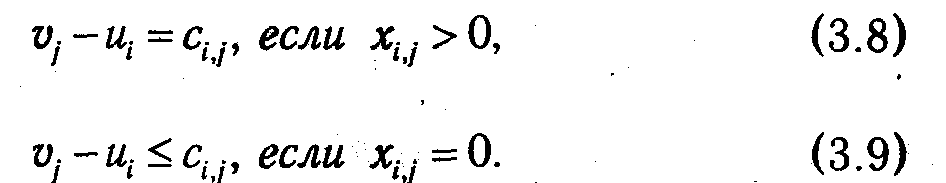
объемы вывоза из пунктов производства, а вторая — ограниче­ниям на удовлетворение потребностей в пунктах производства.

Построим двойственную задачу. С учетом специфической структуры матрицы транспортной задачи вектор двойственных переменных будет иметь размерность *m*+*n*, причем его компо­ненты, соответствующие первой группе ограничений, обозна­чим через (-*ui*), *i*∊1: *m*, а второй — через *vj* , *j*∊1:*n* (*рис. 3.2*). Тогда двойственная задача будет иметь вид:



Переменные *ui* называют *потенциалами пунктов производства*, а *vj* — *потенциалами пунктов потребления*. Применяя доказанные в главе 1 теоремы двойственности (см. теорему 1.7), можно получить критерий оптимальности для плана транспортной задачи:

* *Для того, чтобы допустимый план транспортной зада­чи хi,j был оптимальным, необходимо и достаточно, чтобы нашлись такие потенциалы иi, vj, для которых*

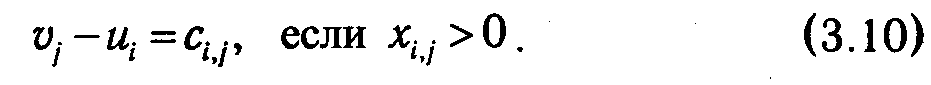


Данные условия имеют содержательную экономическую интерпретацию. Потенциалы *ui*, и *vj* можно рассматривать как цены на перевозимый груз в пунктах производства и потребле­ния (это, кстати, объясняет то, зачем понадобилось обозначать соответствующую двойственную переменную через (-*ui*)). Тогда, согласно условию (3.8), для оптимальности плана пере­возок требуется, чтобы на тех маршрутах, по которым действи­тельно перевозится груз, его цена в пункте потребления воз­растала ровно на цену его перевозки, а в соответствии с условием (3.9) в оптимальном плане цена груза в пункте потреб­ления не может быть меньше его цены в пункте производства с учетом затрат на транспортировку.

**3.1.4. Алгоритм метода потенциалов для транспорт­ной задачи.** Критерий (3.8)-(3.9) положен в основу одного из методов решений транспортной задачи, получившего название *метода потенциалов*. Впервые он был предложен в 1949г. Л. В. Канторовичем и М. К. Гавуриным. Позже на базе общих идей линейного программирования аналогичный метод был предложен Дж. Данцигом.

Точно так же как транспортная задача является частным случаем задачи ЛП, так и метод потенциалов, вообще говоря, может трактоваться как разновидность симплексных проце­дур. Он представляет собой итеративный процесс, на каждом шаге которого рассматривается некоторый текущий базисный план, проверяется его оптимальность, и при необходимости оп­ределяется переход к «лучшему» базисному плану.

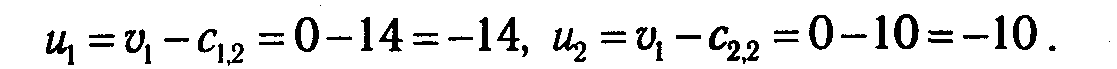
Алгоритм начинается с выбора некоторого допустимого ба­зисного плана (методы его построения были рассмотрены в п. 3.1.2). Если данный план не вырожденный, то он содержит *m* + *n* -1 ненулевых базисных клеток, и по нему можно так оп­ределить потенциалы *ui* и *vj*, чтобы для каждой базисной клет­ки (т. е. для той, в которой *хi,j* > 0) выполнялось условие



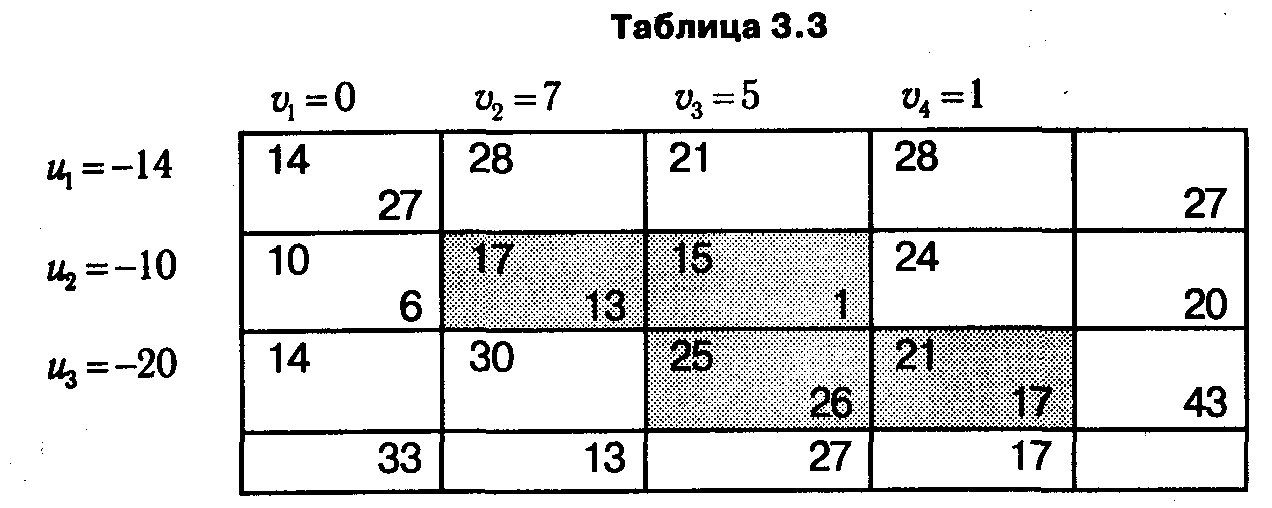
Поскольку система (3.10) содержит *m*+*n*-1 уравнение и *m*+*n* неизвестных, то один из потенциалов можно задать произволь­но (например, приравнять *vj* или *ui* к нулю). После этого ос­тальные неизвестные *ui* и *vj* определяются однозначно.

Рассмотрим процесс определения потенциалов текущего плана транспортной задачи на примере. В табл. 3.3 переписаны условия задачи из табл. 3.1 и ее допустимый базисный план, по­строенный методом северо-западного угла (см. табл. 3.2).

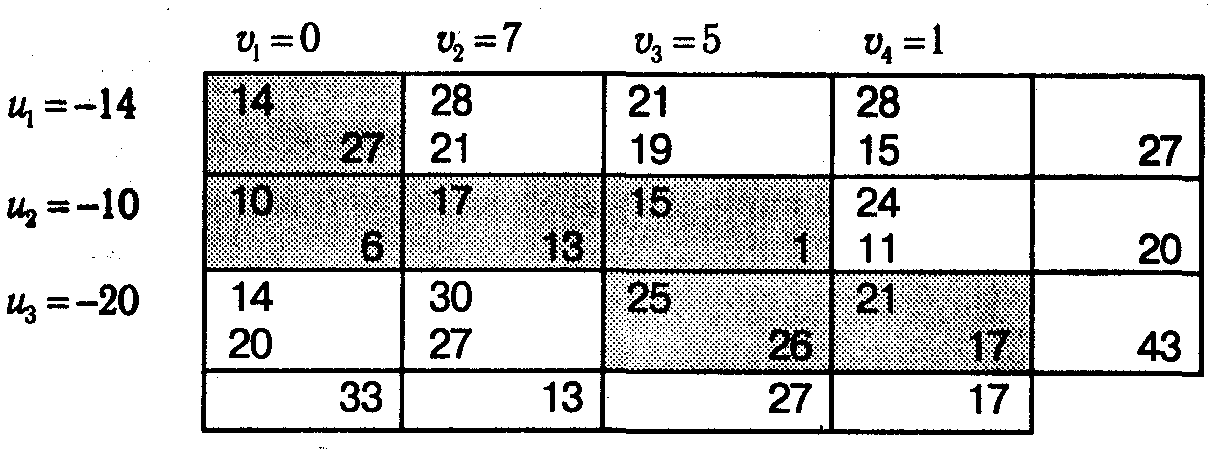
Потенциал первого пункта потребления принимаем равным нулю (*v*1=0). Теперь, зная его, мы можем определить потенци­алы для всех пунктов производства, связанных с первым пунк­том ненулевыми перевозками. В данном случае их два (это пер­вый и второй пункты), получаем:



Имея *u*2 и учитывая, что во второй строке таблицы существу­ют еще ненулевые компоненты *х*2,2 и *х*2,3, можно определить *v*2 = *u*2 + *c*2,2 = -10+17=7 и *v*3 = *u*2 +*c*2,3 = -10+15=5, после чего появляется возможность рассчитать *u*3 = *v*3 – *c*3,3 =5 - 25 = - 20 и, наконец, *v*4 = *u*3+ *c*3,4 = -20 +21=1. В результате получаем полную систему потенциалов, показанную в табл. 3.3.



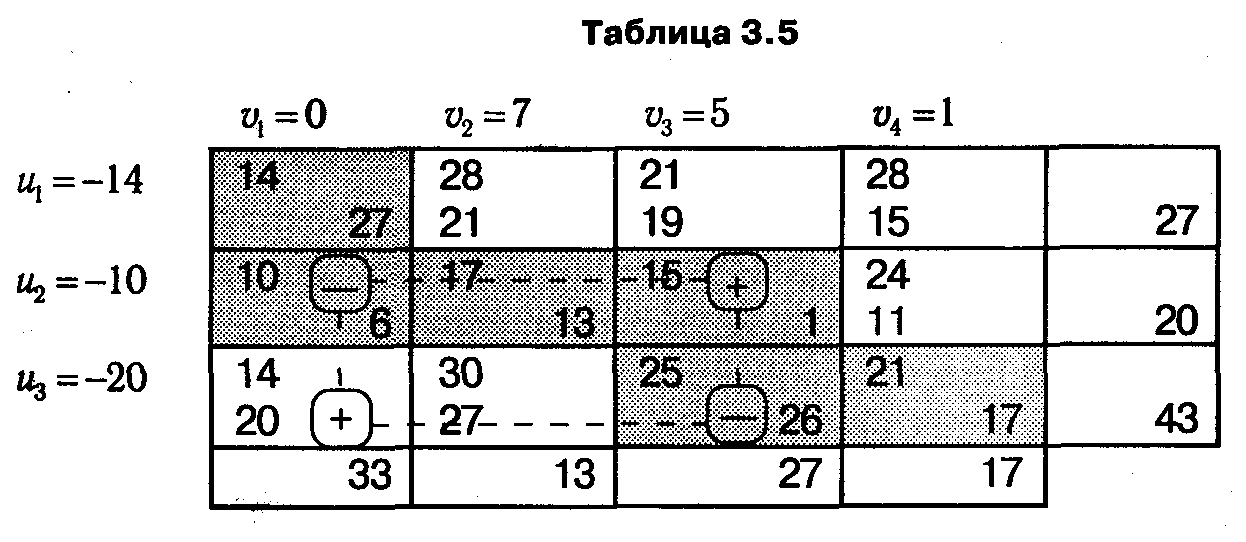
Для свободных клеток транспортной таблицы вычисляют­ся величины α*i,j* = *vj* - *ui*, называемые *разностями потенциа­лов*. В табл. 3.4 они выписаны для всех небазисных клеток под ценами.



Разность потенциалов α*i,j* можно трактовать как увеличение цены продукта при его перевозке из пункта *i* в пункт *j*. Согласно критерию оптимальности (3.8)-(3.9), если все α*i,j* ≤ *сi,j*, то план оптимален, в противном случае, если существует хотя бы одна разность потенциалов α*i,j* > *сi,j*, то он может быть улучшен. Про­цесс «улучшения» плана состоит в определении вводимой и выво­димой клеток, в чем прослеживается содержательная аналогия с соответствующими пунктами симплекс-процедур.

Кандидатом на ввод, очевидно, может быть любая клетка, в которой α*i,j* > *сi,j*, поскольку после ввода в базис будет обеспе­чено равенство α*i,j* = *сi,j*. Для определенности обычно рекомен­дуется брать ту клетку, в которой *оценка* α*i,j* - *сi,j* максималь­на. В рассматриваемом нами примере это будет клетка (3, 1).

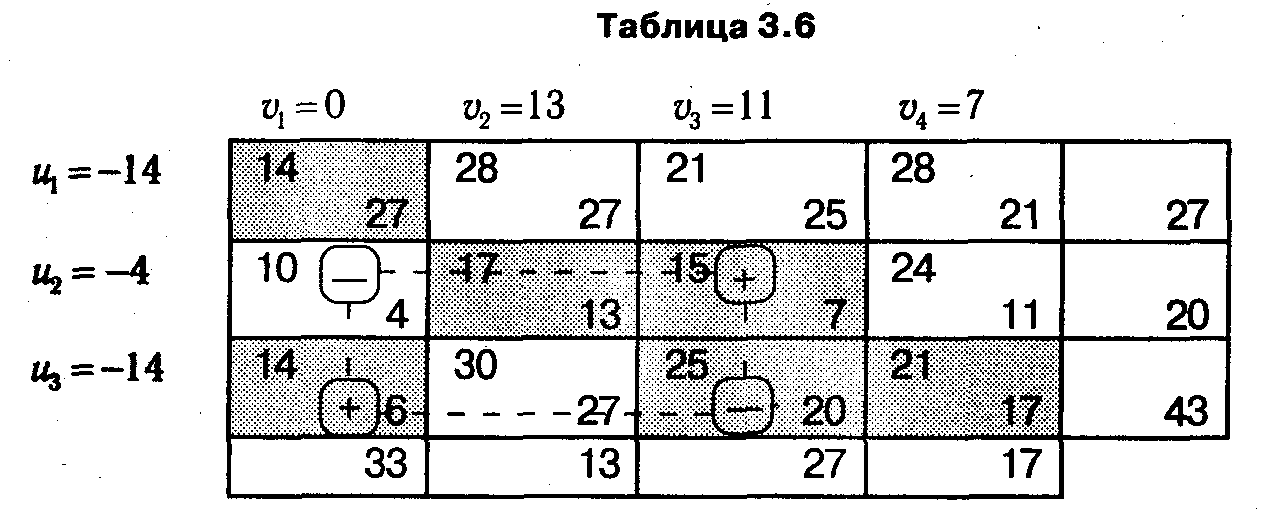
Выводимая клетка определяется с помощью так называемой *цепочки преобразования плана*, описывающей характер пе­рераспределения грузовых потоков. В соответствии со свой­ствами транспортной задачи для невырожденного базисного плана в текущей таблице можно образовать замкнутую цепоч­ку, состоящую только их вертикальных и горизонтальных звень­ев, одной из вершин которой является выбранная свободная клетка, а остальные — занятые клетки. В табл. 3.5 показана це­почка преобразования текущего плана относительно вводимой в него клетки (3, 1).



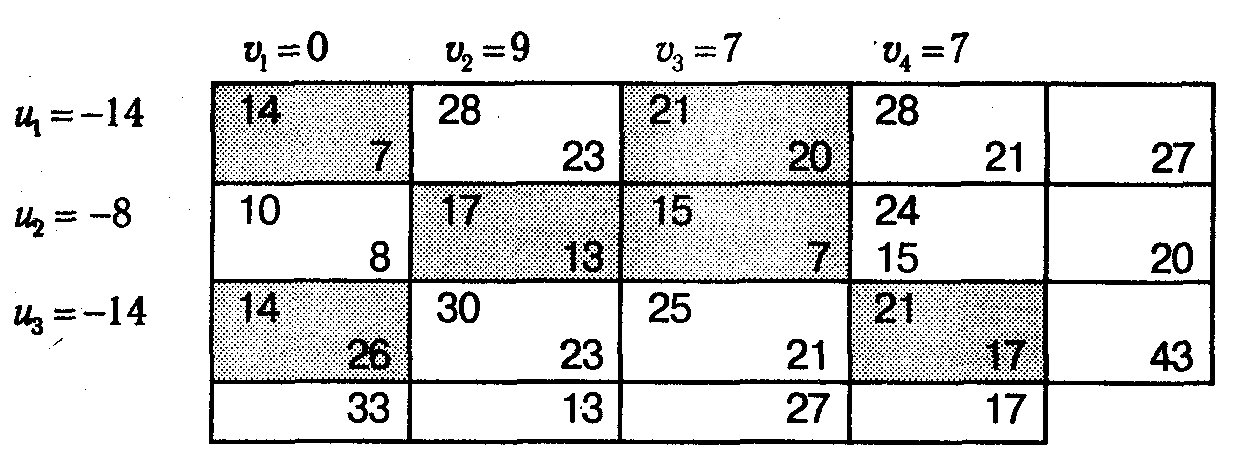
Логика алгоритма построения цепочки достаточно проста: «выйдя» из клетки (3,1) в горизонтальном направлении, мы дол­жны «остановиться» в той занятой клетке плана, из которой сможем двигаться дальше по вертикали. В данном примере это­му требованию удовлетворяют как клетка (3,3), так и клетка (3,4). Однако цепочка от (3,4) не может быть продолжена даль­ше, в то время как двигаясь от (3,3) по вертикали к (2,3) и далее к (2,1), мы возвращаемся к исходной клетке (3,1) и образуем замкнутый цикл.

В построенной цепочке, начиная с вводимой клетки (кото­рая считается первой), помечаются вершины: нечетные — знаком «+», а четные знаком «—». Знаком «+» отмечаются те клетки, в которых объемы перевозок должны увеличиться (та­ковой, в частности, является клетка, вводимая в план, посколь­ку она должна стать базисной). Знаком «—» — те клетки, в ко­торых перевозки уменьшаются с целью сохранения баланса. Среди множества клеток, помеченных знаком «—», выбирает­ся клетка с наименьшим значением *хi,j* (обозначим его θ). Она и становится кандидатом на вывод, т. к. уменьшение объема пе­ревозок на большую величину может привести к отрицатель­ным значениям *хi,j* в других «минусовых» клетках. Затем произ­водится пересчет плана по цепочке: к объемам перевозок в клетках, помеченных знаком «+», добавляется объем θ, а из объемов клеток, помеченных знаком «—», он вычитается. В ре­зультате ввода одной клетки и вывода другой получается новый базисный план, для которого на следующей итерации описан­ные выше действия повторяются.

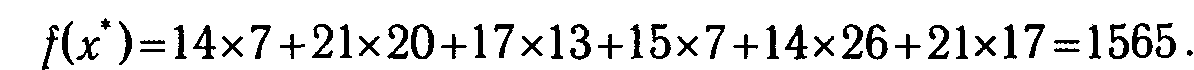
В нашем примере знаком «—» отмечены клетки (2,1) и (3,3), причем *x*2,1 = 6, *x*3,3 = 26. Вычислив значение θ = min{*x*2,1, *x*3,3} = 6, осуществляем преобразование и переходим к следующему ба­зисному плану, показанному в табл. 3.6.



Для вновь полученного плана повторяются действия стан­дартной итерации: рассчитываются потенциалы и оценки для небазисных клеток транспортной таблицы. Как можно видеть, план в табл. 3.6 также не является оптимальным (в клетке (1,3) α*i,j* = 25 > *сi,j* = 21), поэтому вновь строим цепочку преобразо­вания плана и переходим к следующему базисному плану (табл. 3.7).



Из транспортной таблицы 3.7 видно, что полученный план оптимален, так как все разности потенциалов для небазисных клеток α*i,j* = *vj – ui* не превышают соответствующих цен *сi,j*. По данному плану вычисляется оптимальное (наименьшее) значе­ние суммарных издержек на перевозку



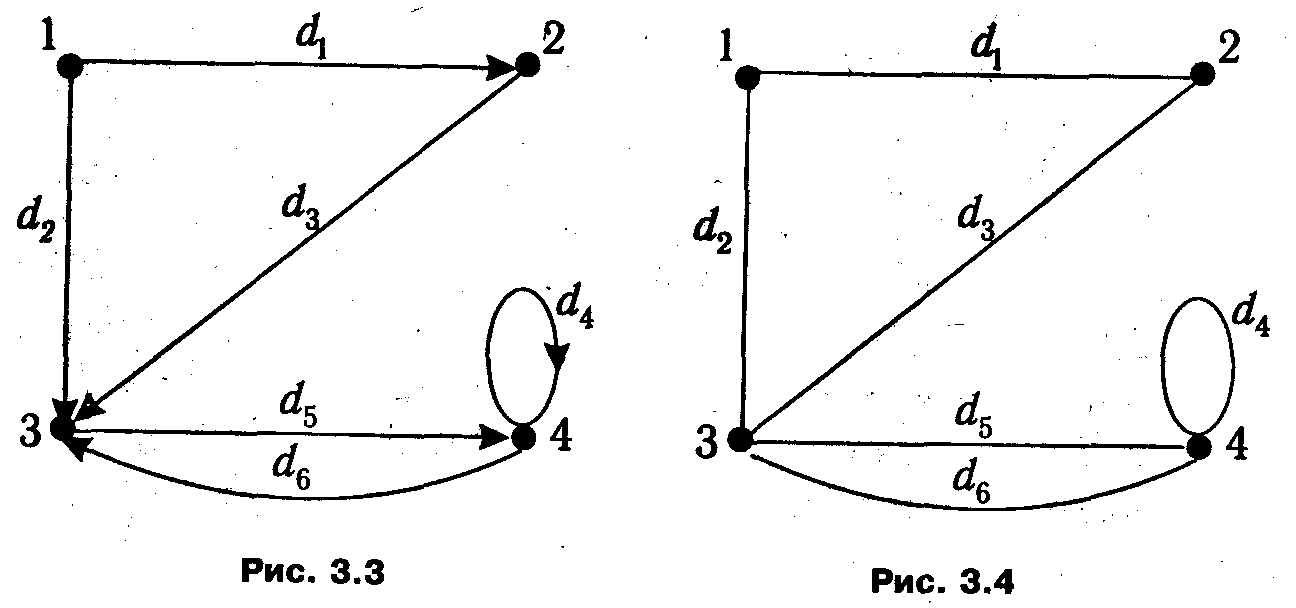
Завершая разговор о методе потенциалов, следует отдельно остановиться на ситуации возникновения вырожденного пла­на. Возможность получения вырожденного плана уже отмеча­лась при описании метода северо-западного угла. Нетрудно за­метить, что вырожденный план также может получиться на этапе преобразования текущего плана по цепочке: если одина­ковое минимальное значение будет достигнуто сразу на не­скольких клетках, помеченных знаком «—», то при вычитании перемещаемого по цепочке объема в новом плане будет меньше чем *m*+*n*-1 ненулевых компонент. Способ преодоления вы­рожденности в транспортной задаче весьма прост, а именно: предлагается дополнить текущий план необходимым коли­чеством нулевых клеток (фиктивными перевозками) таким образом, чтобы они позволяли рассчитать полную систему по­тенциалов, и далее действовать в соответствии с правилами опи­санного выше алгоритма. Фактически здесь мы имеем дело не с чем иным, как с аналогом метода возмущений для транспортной задачи как частного случая ЗЛП. К такому выводу легко прий­ти, если положить, что добавляемые фиктивные клетки содер­жат некоторый малый объем ε.

## 3.2. СЕТЕВЫЕ ЗАДАЧИ

**3.2.1. Основные понятия и определения.** Многие эконо­мические задачи, такие как перевозка грузов, перекачка нефти и газа по трубопроводам, управление запасами и т. п., удобно моделировать и решать в терминах сетей и потоков. Основой подобного рода моделей служат ориентированные или неори­ентированные графы. Приведем некоторые определения.

* **Ориентированным графом** *называется тройка (I, D, G), в которой I — непустое множество* **вершин**, *D* — *множество* **дуг** *и G — отображение, которое каж­дой дуге d∊D ставит в соответствие упорядоченную пару вершин (i, j), где i, j ∊ I.*
* **Неориентированным графом** *называется тройка (I, D, G), в которой I — непустое множество* **вершин**, *D — множество* **ребер** *и G — отображение, которое каж­дому ребру d*∊*D ставит в соответствие неупорядочен­ную пару вершин* [*i, j*]*, где i, j*∊ *I.*

Граф (*I, D, G*) называется *конечным*, если множества *I* и *D* конечны.



Геометрически граф может быть представлен в виде множе­ства точек (изображающих вершины) и соединяющих их линий (со стрелками), соответствующих ребрам (дугам) (*рис. 3.3, 3.4*). Очевидно, что с каждым ориентированным графом можно одно­значно связать неориентированный, заменив дуги на ребра. Если любые две вершины графа соединяются не более чем од­ной дугой (ребром), то граф называется *простым* и может быть задан с помощью пары (*I*, *D*). В этом случае каждая дуга (реб­ро) *d* полностью определяется парой соединяемых вершин (*i*, *j*), что условно записывается в виде: *d=(i,j).* Упорядочен­ная пара вершин (*i, j*), которая ставится в соответствие некото­рой дуге *d*, задает ее ориентацию: *i* называется *началом* дуги, а *j* — ее *концом*, а сама дуга считается *инцидентной* данным вер­шинам.

*Путем длины п* в ориентированном графе (*I*, *D*) называется упорядоченная последовательность различных дуг (*d*1, *d*2,..., *dn*), для которых начало каждой последующей со­впадает с концом предыдущей. Конечный путь, у которого на­чальная вершина совпадает с конечной, называется *контуром.*

Для неориентированного графа аналогом понятия путь яв­ляется *цепь*, а контура — *цикл*.

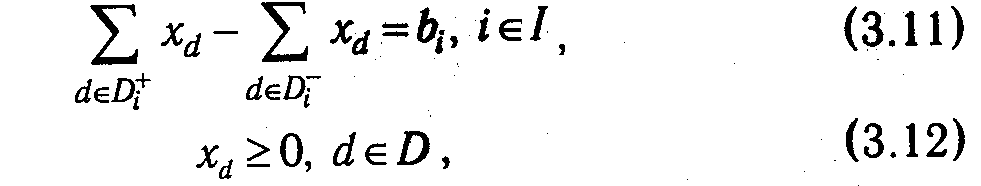
Если две любые вершины неориентированного графа могут быть соединены цепью, то он называется *связным*. Ориентиро­ванный граф называется связным, если ему отвечает связный неориентированный граф.

Связный неориентированный граф, не содержащий циклов, называется *деревом*.

Если *Y* ⊂ *D*, а отображение *GY* является сужением отобра­жения *G* на множество *Y*, то граф (*I*, *Y*, *GY*) называют *частич­ным графом* (реберным подграфом) графа (*I, D, G*).

Рассмотрим задачу: имеется конечный граф (*I, D, G*), каж­дой вершине *i* которого сопоставлено некоторое число *bi*, на­зываемое *интенсивностью вершины*. Граф (*I, D, G*), верши­нам которого сопоставлены значения интенсивностей *bi*, будем называть *сетью*. Если *bi* > 0, то вершина *i* называется *источни­ком*, если *bi* < 0, то — *стоком*, а если *bi* = 0, то — *нейтральной* вершиной. Множество источников, стоков и нейтральных вер­шин обозначим соответственно *I*+, *I*-, *I*0.

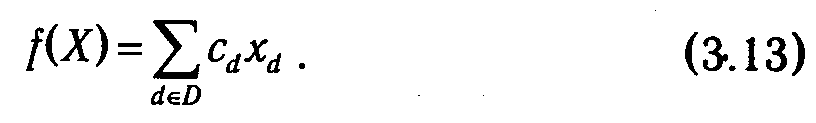
Для определенной выше сети *потоком* называется такая совокупность величин, заданных на множестве дуг, *Х*={*хd*}*d∊D*, что



где *Di*+ — множество дуг, исходящих из вершины *i*, a *Di*- — мно­жество дуг, входящих в нее. Величина *хd* называется *значени­ем потока* по дуге *d* и содержательно интерпретируется как количество продукта, пропускаемого по данной дуге.

Соотношение (3.11) означает, что для любой вершины сети разность выходящего и входящего потоков равна ее интенсив­ности.

На базе введенной терминологии может быть сформулирова­но много различных задач. Рассмотрим наиболее известные из них. Для каждой дуги *d*∊*D* определим значения *cd* ≥ 0, называ­емые *стоимостью перемещения единицы продукта* по дуге, тогда суммарная стоимость потока *Х* примет вид



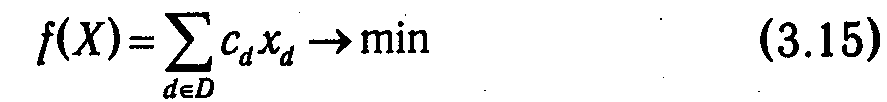
Задачу минимизации функции (3.13) при ограничениях (3.11)-(3.12) обычно называют *линейной сетевой задачей*. Очевидно, что она является задачей линейного программирова­ния. Если дополнительно для каждой дуги сети *d*∊*D* опреде­лить величины *rd* ≥ 0, называемые *пропускными способно­стями*, то, добавив ограничения



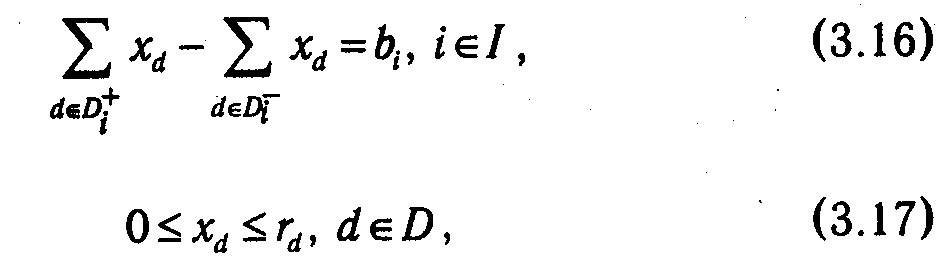
мы получаем задачу о потоке в сети с ограниченными пропуск­ными способностями.

Приведенные формулировки задач специально даны в столь абстрактном виде, что позволяет подчеркнуть их универсаль­ность. К очевидной сфере их приложения относится организа­ция грузоперевозок в транспортной сети. В таких моделях вершины *i* трактуются как пункты, соединенные сетью дорог, и характеризуются потребностями в некотором продукте (*bi*<0) или его запасами (*bi*>0). Задачи определения плана, минимизирующего затраты на перевозки, которые с матема­тической точки зрения полностью идентичны (3.11)-(3.13), (3.14), также называют *транспортными задачами в сете­вой постановке.*

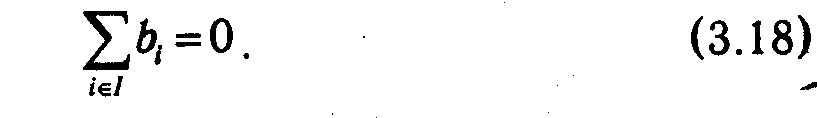
**3.2.2. Метод потенциалов для транспортной задачи в сетевой постановке.** Рассмотрим задачу определения опти­мального потока *Х* в некоторой сети (*I, D, G*), для которого



при ограничениях

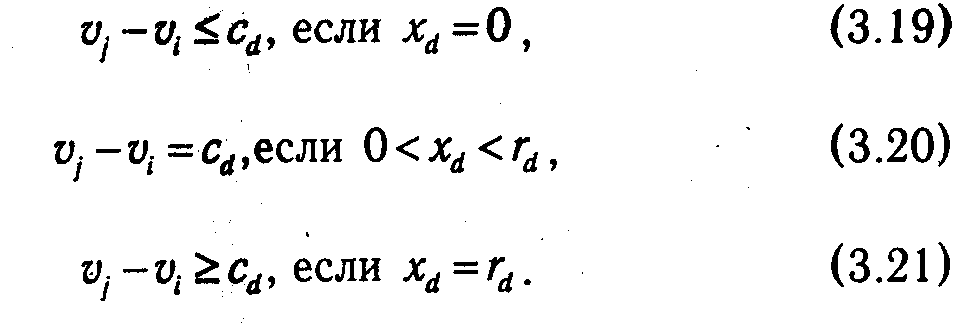


где *rd* ≥ 0. Предполагается также, что сеть является сбаланси­рованной, т. е.



Для задачи (3.15)-(3.17) справедлив критерий оптималь­ности:

* *Для того, чтобы допустимый поток Х={хd}d∊D (т. е. удовлетворяющий условиям* (3.16)-(3.17)) *был опти­мальным, необходимо и достаточно существование для каждой вершины i ∊ I такого числа vi, называемого* **по­тенциалом***, что для всех дуг d = (i, j)*



Заметим, что логика обоснования данного критерия абсо­лютно идентична той, которая использовалась для обоснова­ния критерия оптимальности плана транспортной задачи в мат­ричной постановке: построение двойственной задачи и приме­нение соответствующей теоремы двойственности.

Для решения транспортной задачи в сетевой постановке (3.15)-(3.17) также может быть применен *метод потенциа­лов*, который является обобщением описанного выше мето­да потенциалов для транспортной задачи в матричной поста­новке.

Поскольку задача (3.15)-(3.17) является частным случаем задачи линейного программирования, ее можно привести к ка­нонической форме. При этом достаточно просто устанавлива­ется, что ранг матрицы задачи равен *m*-1, где *m* — количество вершин в сети. Введем дополнительно еще некоторые понятия, используемые при описании свойств сетевых задач.

*Остовом* сети (*I, D, G*) называется любое ее частичное дере­во (частичный граф, являющийся деревом). Справедливо утвер­ждение:

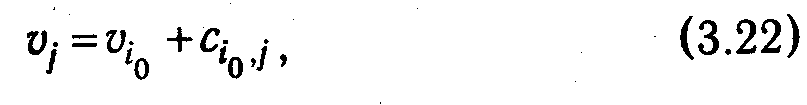
☞ *Произвольному остову сети (I, D, G) соответствует ба­зис задачи* (3.15)-(3.17) *и наоборот.*

Пусть имеется некоторый поток *Х={хd}d∊D*. Рассмотрим множество дуг *D(X)* = {*d* ∊ *D* | 0 < *xd* < *rd*}. *Опорой* потока *Х* назы­вается частичный граф (*I, D(X), G*). Говорят, что поток *Х не­вырожден*, если его опора (7, D(X), G) является остовом сети (*I, D, G*). Иными словами, используя терминологию транспорт­ной задачи, в невырожденном потоке, которому отвечает допу­стимый базисный план задачи, дороги, по которым осуществля­ются перевозки груза, не достигающие по объему ограничения на пропускную способность, образуют остов (связанную под­сеть без циклов) рассматриваемой транспортной сети.

Теперь дадим краткое описание схемы метода потенциалов для транспортной задачи в сетевой постановке.

1°. Предполагается, что в начале очередной итерации *q* имеется некоторый **допустимый невырожденный поток** *Х={хd*(*q*)*}d∊D* (о методах его генерации на начальном этапе бу­дет сказано в дальнейшем).

По имеющемуся потоку *Х*(*q*) строится система потенциалов пунктов сети. Для этого выбирается произвольный пункт *i*0, потенциал которого полагается *vi*0 =0. Множество вершин, смежных с *i*0, обозначим через *I*(*i*0). Тогда для любой вершины *j* ∊ *I*(*i*0) потенциалы рассчитываются по правилу



если (*i*0,*j*) ∊ *D*(*X*(*q*)) (дуга направлена от *i*0),и

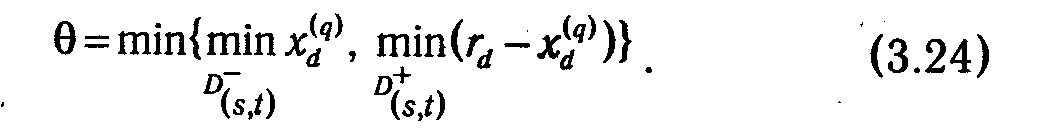


если (*j,i*0) ∊ *G*(*D*(*X*(*q*))) (*j,i*0) ∊ *D*(*X*(*q*)) (дуга направлена к *i*0).

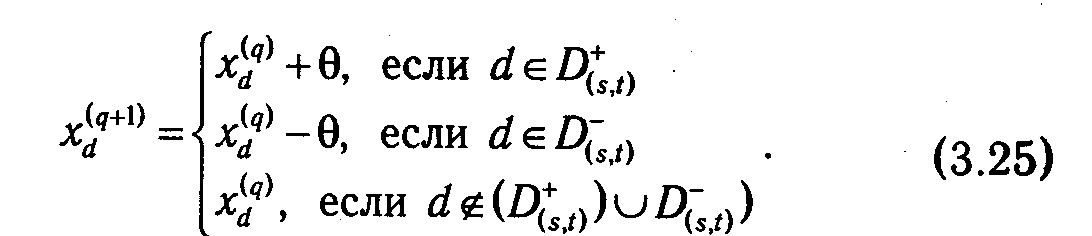
Получив очередную группу вершин с известными потенциа­лами, мы имеем возможность на основе (3.22)-(3.23) вычис­лить потенциалы для следующей группы смежных вершин и т. д., пока не будут определены все потенциалы. Возможность сде­лать это единственным образом вытекает из свойства отсут­ствия циклов у остова сети.

Имея полную систему потенциалов, для всех дуг следует про­верить условия критерия оптимальности (3.19)-(3.21). Если они выполняются, то текущий поток *Х*(*q*) — оптимальный и, следовательно, алгоритм завершен; в противном случае — пе­реходим к построению следующего «улучшенного» потока.

2°. По аналогии с другими методами последовательного улуч­шения плана очередной поток получается за счет «ввода» в него одной дуги и «вывода» другой. Если условия критерия опти­мальности нарушаются сразу для нескольких дуг, то для ввода имеет смысл выбрать ту, на которой достигается максимальное отклонение цены от разности потенциалов соединяемых вер­шин. Пусть для ввода выбрана некоторая дуга *dl* = (*s,* *t*), направ­ленная из вершины *s* в вершину *t*. Из правил построения по­тенциалов следует, что в остове существуют две цепи, одна из которых соединяет базовую вершину *i*0, потенциал которой был принят равным нулю, с *s*, а другая — *i*0 с *t.* Если дополнить остов дугой *dl,* образуется единственный цикл. Построенный цикл является аналогом цепочки преобразования плана в мето­де потенциалов для транспортной задачи в матричной постанов­ке. Обозначим через *D+*(*s,t*) множество дуг данного цикла, ориен­тация которых совпадает с ориентацией дуги *dl* = (*s, t*), а через *D-*(*s,t*) — множество дуг, имеющих противоположную ориента­цию. Определим величину возможной корректировки объемов грузоперевозок, «перемещаемых» по циклу



Идея формулы (3.24) достаточно прозрачна: при циклическом преобразовании текущего потока увеличиваются объемы гру­зоперевозок на тех дугах, которые сонаправлены вводимой дуге, и уменьшаются на дугах, имеющих обратную ориентацию. Соответственно, при добавлении мы должны следить за тем, чтобы не превысить ограничения на пропускные способности (θ ≤ *rd* – *хd*(*q*)), а при вычитании — за неотрицательностью *хd*(*q*). После определения θ происходит пересчет компонент текуще­го потока по формуле



В результате мы получаем новый допустимый поток *хd*(*q*+1)), полагаем номер текущей итерации *q*+1 и переходим к п. 1°.

В описанном алгоритме, как и в случае с матричной транс­портной задачей, мы не гарантированы от возникновения вы­рожденного потока. Как уже упоминалось выше, такому пото­ку будет соответствовать несвязная опора. Для преодоления вырожденности рекомендуется включить в текущий план фик­тивные компоненты с нулевыми объемами так, чтобы соответ­ствующие им дуги дополняли опору до остова сети. Построен­ный таким способом план позволяет выполнить все действия, входящие в стандартную итерацию метода потенциалов.

Отдельно следует остановиться на методах генерации исход­ного допустимого потока. Наиболее простой из них (хотя, воз­можно, и наименее рациональный) основан на идеях, сходных с идеями метода минимизации невязок, используемого для по­строения допустимого базисного плана ЗЛП. Данный метод предполагает решение соответствующей вспомогательной за­дачи, которая получается из основной в результате следующих преобразований:

1. К множеству вершин сети добавляется фиктивная нулевая вершина с нулевой интенсивностью (*b*0= 0).

2. Все вершины, имеющие отрицательную интенсивность (спрос) *bi* < 0, соединяются с добавленной вершиной 0 входя­щими дугами (0, *i*), а вершины, обладающие положительной интенсивностью (запасом) *bi* > 0, — исходящими дугами (*i*, 0). Ограничения на пропускные способности для добавляемых дуг отсутствуют.

3. Стоимости перемещения единицы продукта для вновь до­бавленных дуг полагаются равными 1, а для дуг, соответствую­щих транспортной сети основной задачи, — 0.

Построенная вспомогательная задача обладает очевидным допустимым невырожденным потоком, получаемым назначени­ем объемов, равных интенсивностям вершин, по всем добавлен­ным дугам. Решив вспомогательную задачу, мы либо получим допустимый поток для основной задачи, либо придем к выводу об отсутствии у нее допустимых планов.

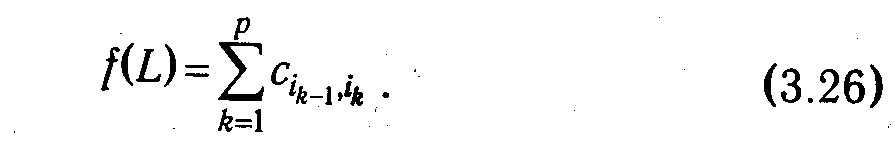
**3.2.3. Задача о кратчайшем пути.** Классическим приме­ром сетевых задач является определение кратчайшего пути между вершинами сети. Пусть задан граф (*I, D, G*), каждой дуге которого поставлено в соответствие число *cd*, называемое *дли­ной*. Также пусть выделены две вершины графа *s* и *t*, и требует­ся найти путь наименьшей длины, ведущий из вершины *s* в вер­шину *t.*

Если в графе имеются «кратные» дуги, соединяющие одина­ковые начало и конец, то достаточно оставить одну — с наи­меньшей длиной, а остальные отбросить. Таким образом, доста­точно рассматривать задачу о кратчайшем пути для простого графа (*I,* *D*), в котором дуги определяются упорядоченными па­рами вершин *d* = (*i*, *j*). Тогда естественно путь *L*, идущий из вер­шины *s* в вершину *t*, задавать в виде упорядоченного набора вер­шин, через которые проходит данный путь:



а длины дуг обозначать как *cd* = *ci,j*.

Длина описанного выше произвольного пути *L* определяет­ся по формуле



Легко заметить, что задача о кратчайшем пути является част­ным случаем транспортной задачи в сетевой постановке (или, что то же самое, задачи об оптимальном потоке). Для этого до­статочно присвоить вершине *s* единичный запас, вершине *t* еди­ничную потребность, все остальные вершины положить нейт­ральными, а дугам присвоить неограниченные пропускные способности. Однако, как правило, более рациональным оказывается использование конкретных свойств данной задачи и ре­шение ее специальными (частными) методами. К их числу отно­сится, например, метод Минти, основные идеи которого мы изложим ниже.

*Метод Минти* решения задачи о кратчайшем пути в сети представляет собой итеративный процесс, в ходе которого строится путь *L=(s=i*0*, i*1*, ..., ip-*1*, ip=t).*

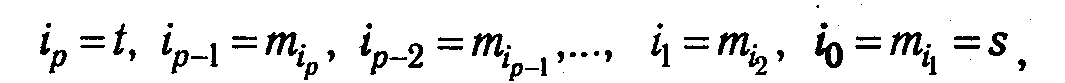
На предварительном (нулевом) этапе алгоритма:

* формируется массив значений так называемых *модифициро­ванных длин i,j,* которые перед началом первой итерации полагаются равными *сi,j* ≥0;
* осуществляется отметка вершины *i*0 *= s* числом *mi*0 = 0.

Стандартная итерация включает этапы:

1°. *Отметка вершин сети*. Обозначим множество вершин cети, отмеченных на предыдущих итерациях, как ** (на первой итерации ** ={*i*0}). Для каждой вершины *i*∊** ищутся дуги, со­единяющие ее с еще не помеченными вершинами-потомками *j*, модифицированная длина которых *i,j* = 0. Найденные таким способом вершины *j* помечаются числом *mj* = *i*, указывающим на «родителя». В том случае, когда сразу несколько дуг, имею­щих *i,j* = 0, заканчиваются в одной и той же вершине *j*, значе­ние для ее пометки выбирается произвольно.

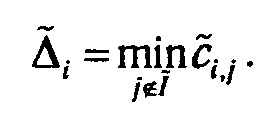
Если среди вновь помеченных вершин окажется вершина *t*, то, значит, найден искомый путь (*i*0, *i*1,..., *i*(*p-*1), *ip*), где



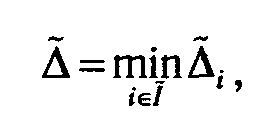
на чем алгоритм завершается.

В случае, если вершины *t* нет среди отмеченных, и одновре­менно нельзя отметить ни одной новой вершины, то переходим к этапу 2.

2°. *Преобразование значений модифицированных длин дуг*. Для каждой вершины *i*∊** ищутся дуги, соединяющие ее с еще не помеченными вершинами *j*, и находятся



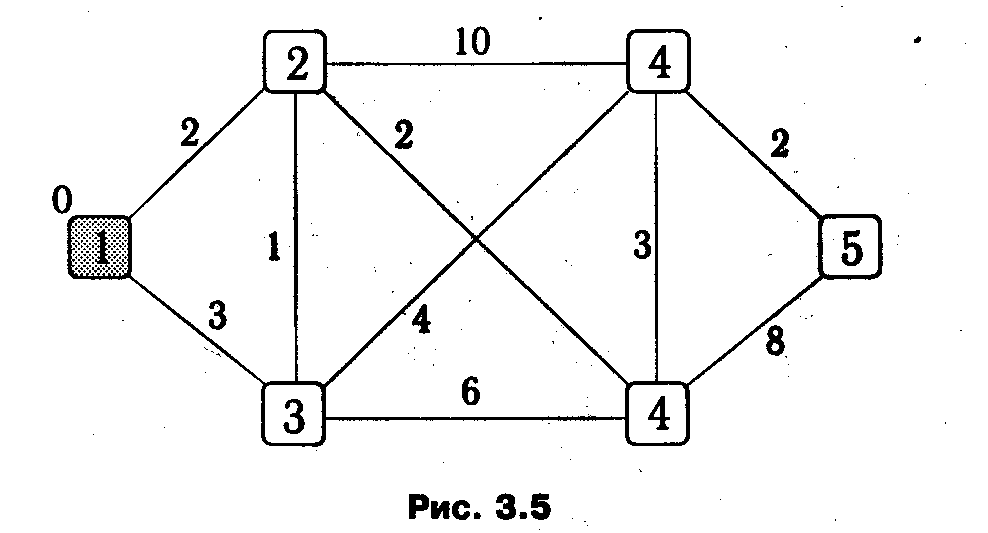
Далее модифицированные длины всех дуг, которые соединя­ют отмеченные вершины с неотмеченными (*i*∊**, *j*∉**), умень­шаются на величину



в результате чего кратчайшие неиспользованные дуги получа­ют нулевую модифицированную длину.

Затем происходит переход к следующей итерации.

Путь, построенный по методу Минти, будет кратчайшим. Это можно доказать с помощью индукции по номеру итерации, на которой была помечена вершина *t*, или, что то же самое, по ко­личеству дуг, составляющих кратчайший путь. Если это про­изошло на первом шаге (что возможно только в случае, если начальная и конечная вершины соединены дугой нулевой дли­ны), то доказываемое утверждение очевидно. Предположим,



что оно верно для всех пунктов, помеченных за первые *r* итера­ций, т. е. тех, которые достигаются переходом по *r* дугам. Тогда, если конечная вершина *t* помечена на (*r* + 1)-ой итерации, то полученный путь также будет кратчайшим, так как данная вер­шина помечается в результате минимально возможного продол­жения одного из путей, полученного за предыдущие *r* итераций и являющегося по предположению кратчайшим.

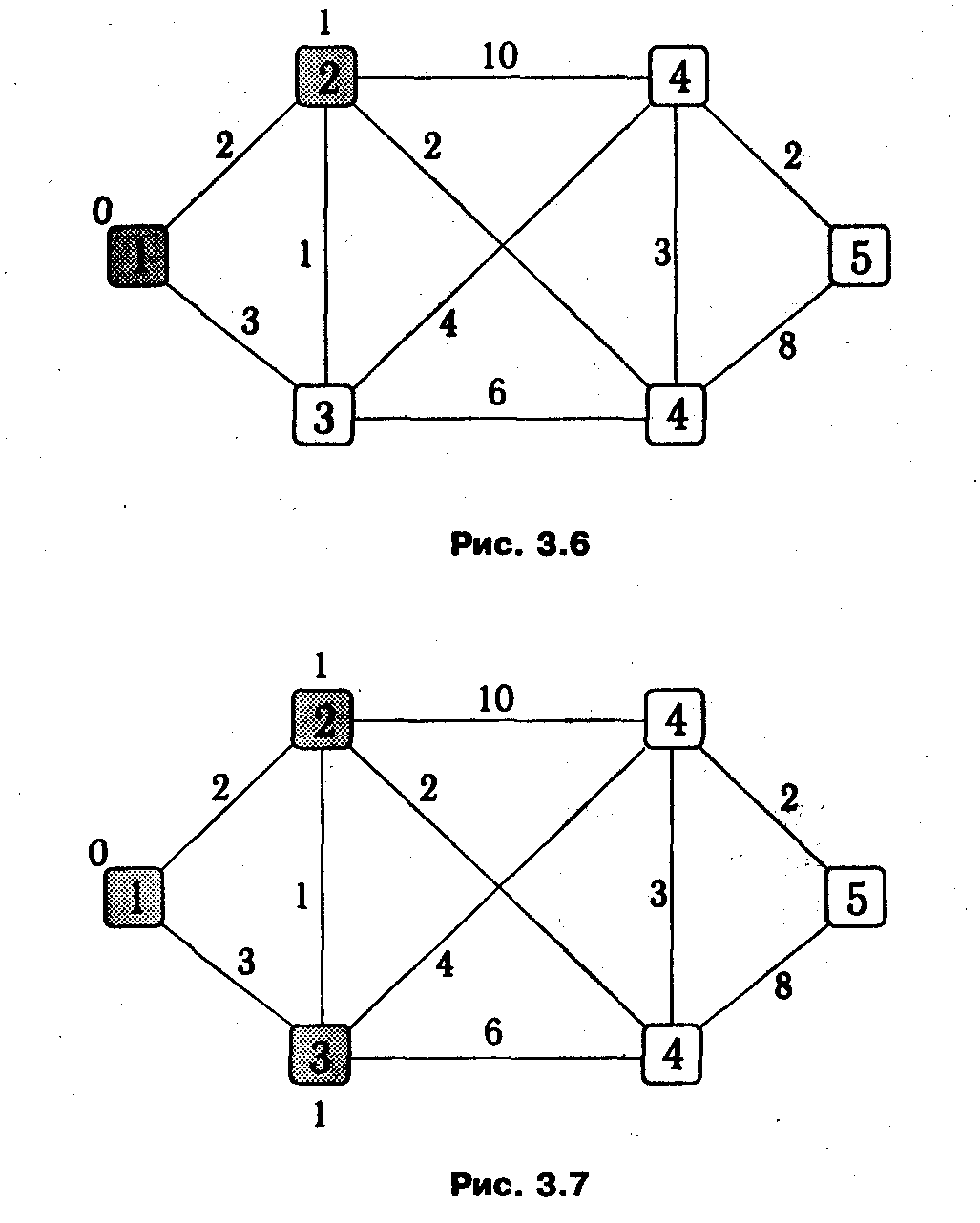
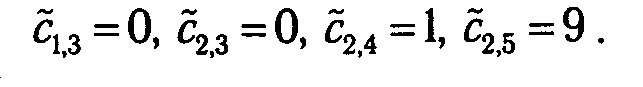
Отметим, что описанный алгоритм пригоден для *построе­ния кратчайших путей на неориентированных графах.*

Рассмотрим изложенный метод на конкретном примере, а именно: определим кратчайший путь из вершины 1 в вершину 6 для неориентированной сети, показанной на *рис. 3.5.*

На предварительном этапе вершина 1 отмечается числом *m*1 = 0, а модифицированные длины совпадают с заданными дли­нами дуг.

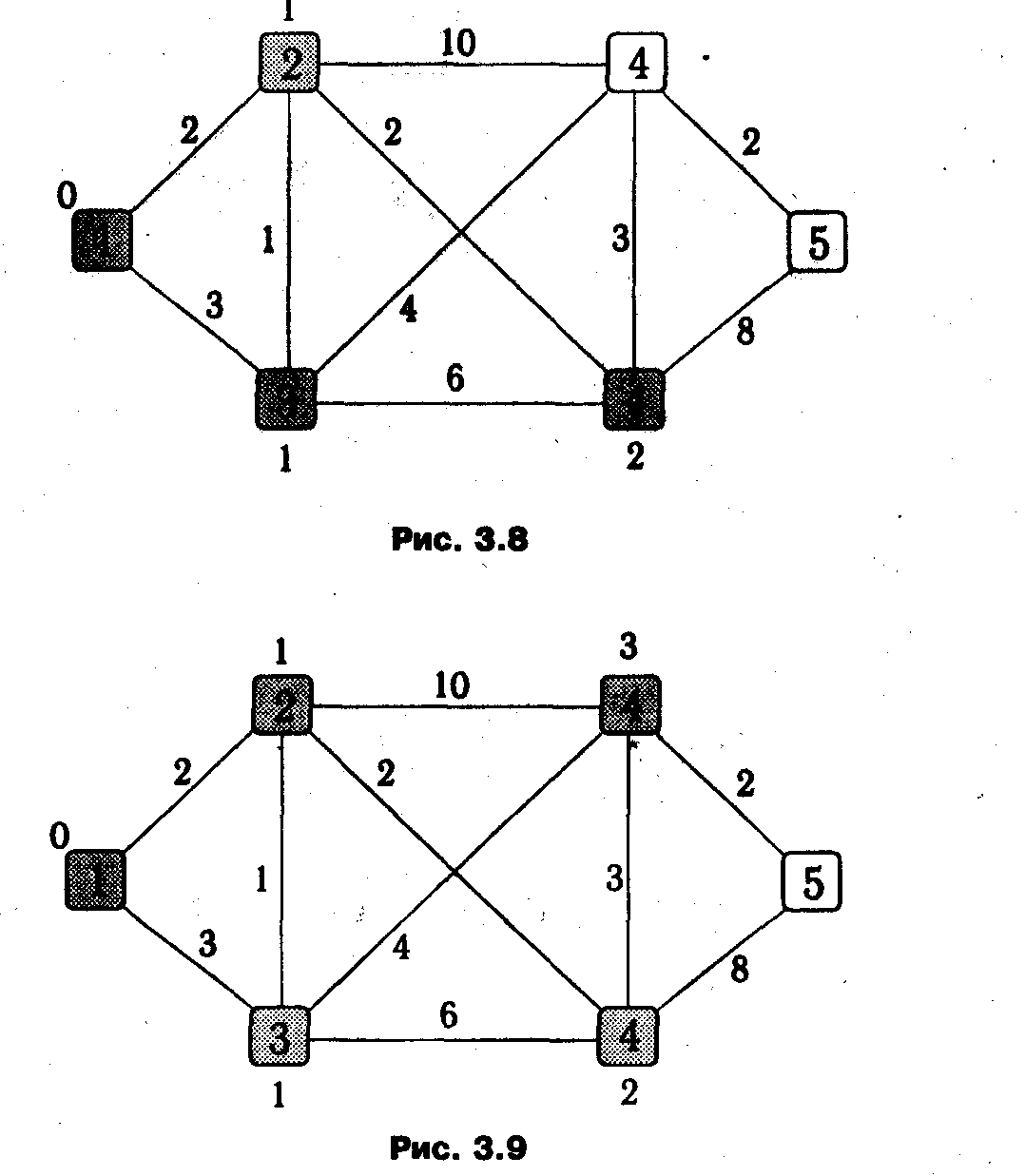
*Итерация 1.* Так как из вершины 1 не выходят дуги нулевой длины, дальнейшая отметка вершин невозможна. Переходим к этапу 2. Смежными с вершиной 1 являются вершины 2 и 3. Для них определяем ∆ = min{**1,2, **1,3}=2 и вычитаем ее из **1,2, **1,3. После преобразования имеем **1,2 = 0, **1,3 = 1.

*Итерация 2.* Помечаем вершину 2 *m*2 = 1 (см. *рис. 3.6*). Дальнейшая пометка невозможна, поэтому переходим к этапу 2. Смежными с помеченными вершинами 1 и 2 являются верши­ны 3,4,5. Из чего определяем ∆ = min{**1,3, **2,3, **2,4, **2,5}=1 и после соответствующего преобразования имеем



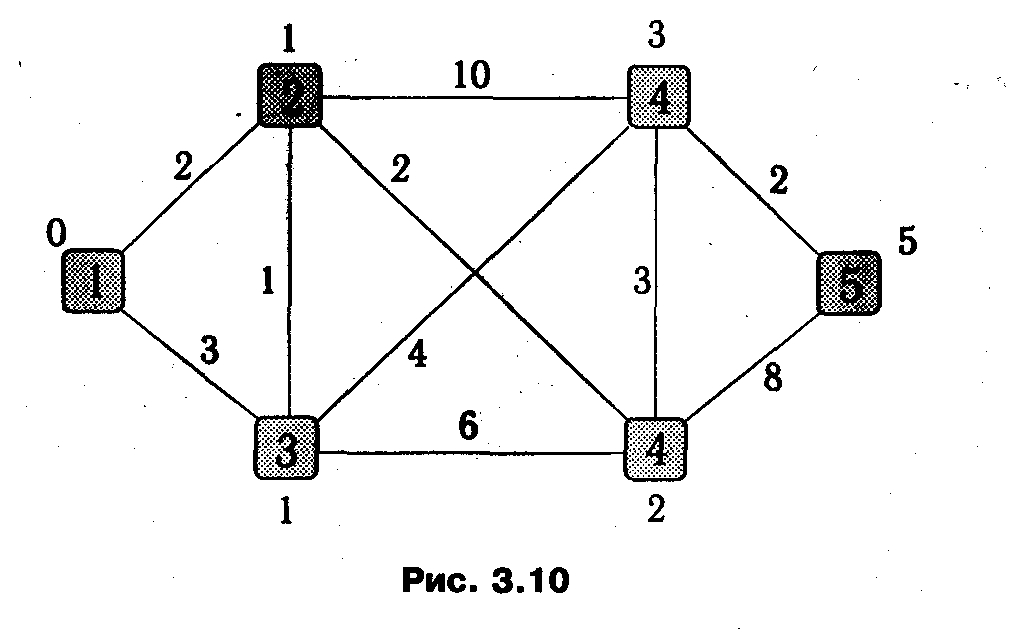
*Итерация 3.* В вершину 3 ведут дуги нулевой длины как из вершины 1, так и из вершины 2. Поскольку выбор здесь может быть произвольным, пометим вершину 3 числом *m*3 = 1 (*рис. 3.7*). Дальнейшая пометка невозможна, поэтому переходим к этапу 2. Смежными с ранее отмеченными вершинами являются вершины 4,5. Из чего определяем ∆ = min{**2,4, **2,5, **3,4, **3,5}=1 и после пре­образования имеем **2,4 = 8, **2,5 = 0, **3,4 = 3, **3,5 = 5.

*Итерация 4.* Помечаем вершину 4 *m*4 =2 (см. *рис. 3.8*). Даль­нейшая пометка невозможна, поэтому переходим к этапу 2. Смежными с ранее помеченными вершинами являются верши­ны 5,6. Из чего определяем ∆ = min{**2,5, **3,5, **4,5, **4,6}=3 и после преобразования имеем **2,5 = 5, **3,5 = 0, **4,5 = 0, **4,6 = 5.



*Итерация 5.* В вершину 5 ведут дуги нулевой длины как из вершины 3, так и из вершины 4. Руководствуясь теми же сообра­жениями, что и на итерации 3, пометим вершину 5 числом *m*5 =3 (*рис. 3.9*). Дальнейшая пометка невозможна, поэтому перехо­дим к этапу 2. Смежной с ранее отмеченными вершинами являет­ся вершина 6. Из чего определяем ∆ = min{**4,6, **5,6}=2 и после преобразования имеем **4,6 = 3, **5,6 = 0.

*Итерация 6.* В вершину 6 ведет дуга нулевой длины из вер­шины 5, поэтому помечаем ее числом *m*6=5 (см. *рис. 3.10*). По­скольку мы отметили конечную вершину маршрута, то алго­ритм завершен и мы можем, используя значения отметок для родителей, выписать искомый кратчайший путь (1, 3, 5, 6).



Следует также добавить, что если бы наш выбор на итераци­ях 3 и 5 был иным, то мы получили бы альтернативный путь той же длины (1, 2, 4, 5, 6), т. е. рассмотренная задача имеет не­сколько решений.

## КЛЮЧЕВЫЕ ПОНЯТИЯ

* Транспортная таблица
* Метод северо-западного угла.
* Потенциал.
* Цепочка преобразования плана.
* Граф (ориентированный и неориентированный).
* Ребра и вершины.
* Путь и контур.
* Цепь и цикл.
* Связность.
* Дерево.
* Частичный граф.
* Транспортная сеть.
* Поток.
* Линейная сетевая задача.
* Остов сети.
* Опора потока.
* Невырожденный поток.
* Задача о кратчайшем пути.
* Алгоритм Минти.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

3.1. Какие специфические свойства позволяют выделить транспортные задачи в отдельный класс из

множества за­дач линейного программирования?

3.2. Опишите метод построения допустимого плана транспор­тной задачи.

3.3. Сколько ненулевых элементов должен содержать невы­рожденный базисный план транспортной

задачи?

3.4. Сформулируйте критерий оптимальности для допустимо­го плана транспортной задачи.

3.5. Что положено в основу метода потенциалов?

3.6. Из чего вытекает критерий оптимальности допустимого плана транспортной задачи?

3.7. Перечислите основные этапы метода потенциалов.

3.8. Какие условия должны быть соблюдены при построении цепочки преобразования плана в методе

потенциалов?

3.9. Что следует делать при возникновении ситуации вырож­денности текущего плана в транспортной

задаче?

3.10. Приведите формулировку линейной сетевой задачи.

3.11. Покажите, что транспортная задача в матричной поста­новке является частным случаем

транспортной задачи в сетевой постановке.

3.12. Дайте определение понятия «остов сети». Какая связь су­ществует между остовом сети и

базисом транспортной за­дачи в сетевой постановке?

3.13. Какой поток называют невырожденным?

3.14. Перечислите основные этапы метода потенциалов для транспортной задачи в сетевой

постановке.

3.15. Каким способом можно получить допустимый поток в транспортной сети?

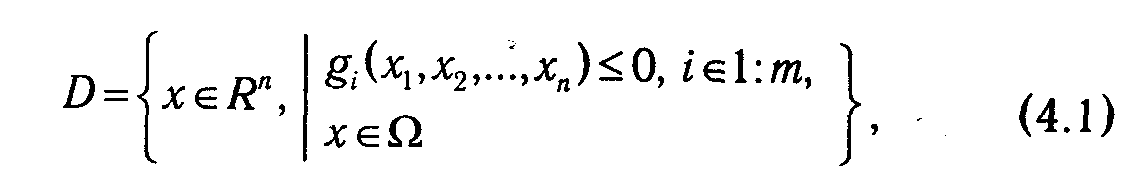
3.16. В чем состоит задача о кратчайшем пути?

3.17. Перечислите основные этапы метода Минти.

# ГЛАВА 4. ДИСКРЕТНОЕ ПРОГРАММИРОВАНИЕ

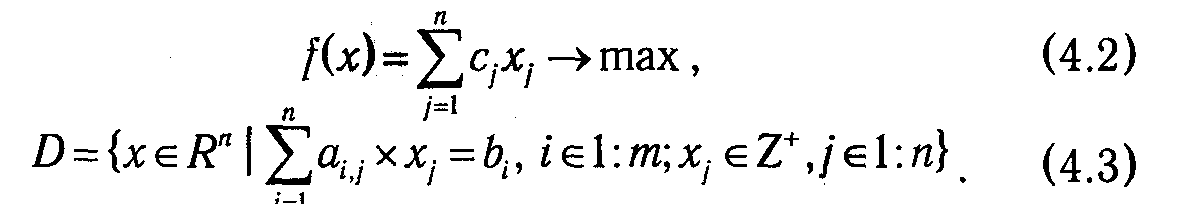
## 4.1. ТИПЫ ЗАДАЧ ДИСКРЕТНОГО ПРОГРАММИРОВАНИЯ

**4.1.1. Основные понятия.** Многие экономические задачи ха­рактеризуются тем, что объемы управляемых ресурсов (в силу тех или иных объективных свойств) могут принимать только целые значения. Математическая формализация данных ситуа­ций приводит к моделям дискретного программирования. В об­щем виде задача дискретного программирования может быть сформулирована как задача нахождения максимума (или мини­мума) целевой функции *f*(*x*1, *x*2,*...,xn*) на множестве *D*, опреде­ляемом системой ограничений



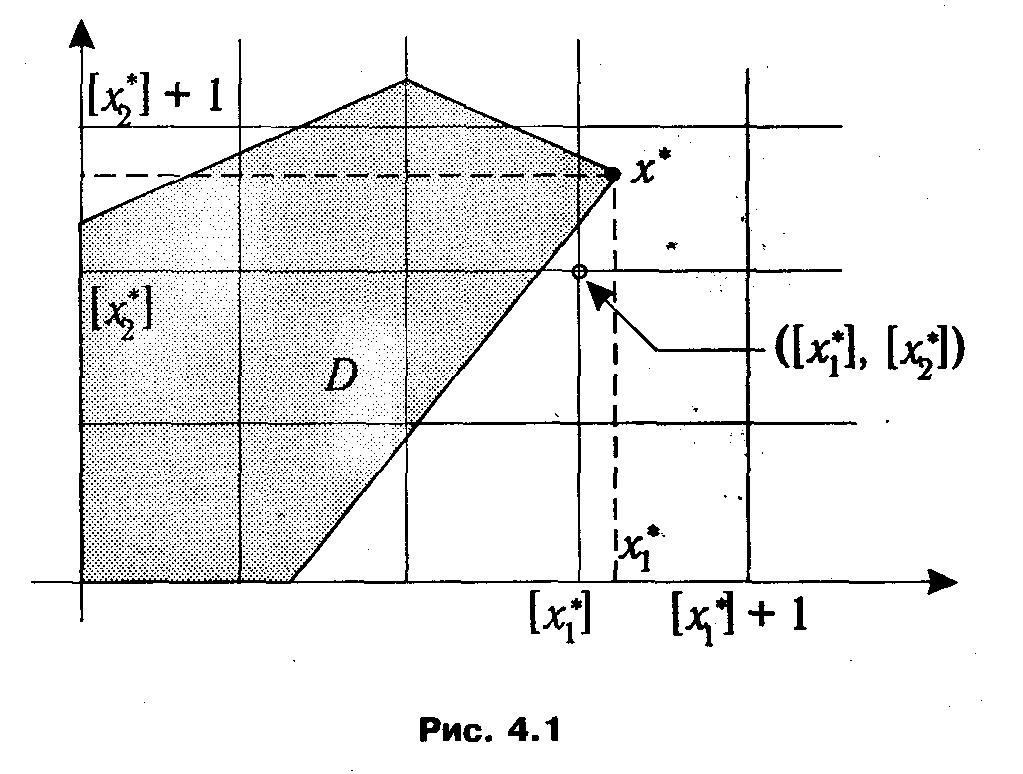
где Ω — некоторое конечное, или счетное\*, множество. Ус­ловие *х*∊Ω. называется *условием дискретности.* Особое место среди дискретных задач занимает *целочисленная за­дача линейного программирования в канонической форме* (ЦКЗЛП):

\* Напомним, что примерами счетных множеств являются множества натуральных, целых и рациональных чисел.



где *Z*+ ={0; 1; 2; ...} — множество неотрицательных целых чи­сел.

Заметим, что в некоторых ситуациях требование «целочисленности» может быть наложено лишь на некоторые перемен­ные *xj*, что кардинально не меняет характера задачи.



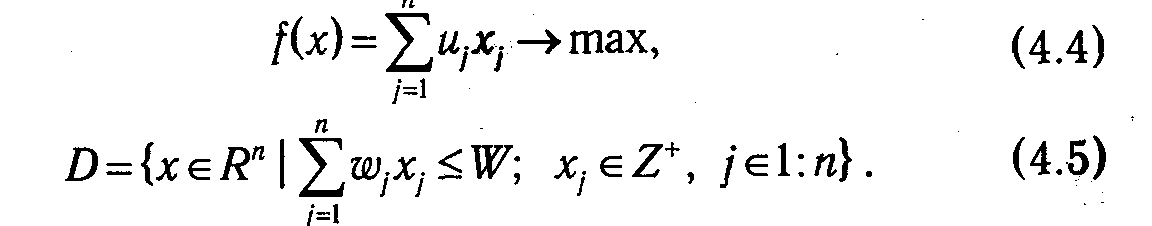
Принципиальная сложность, вызываемая наличием условий целочисленности в системе ограничений оптимизационной задачи, состоит в том, что в значительном количестве случаев невозможно заменить дискретную задачу ее непрерывным ана­логом и, найдя соответствующее решение, округлить его ком­поненты до ближайших целых значений. Пример, показанный на *рис. 4.1*, демонстрирует, что при округлении оптимального плана *х\** обычной задачи ЛП до целых значений получается точ­ка ([*х*1\*],[*x*2\*]), не принадлежащая области допустимых планов задачи *D*. Условимся целую часть числа *хj*. обозначать [*хj*], а дробную — как {*хj*}. Тогда *хj* =[*хj*]+{*хj*}. Отдельно следует до­бавить, что если даже оптимальный план непрерывной задачи, округленный до целых значений компонент, окажется допусти­мым, то целевая функция может вести себя так, что ее значение будет на нем существенно «хуже», чем на оптимальном плане целочисленной задачи.

Перечисленные проблемы предопределили необходимость разработки специальных методов решения дискретных и цело­численных задач. Но прежде чем говорить собственно о мето­дах решения, более подробно остановимся на классификации задач дискретного программирования. В литературе, как пра­вило, выделяют следующие классы дискретных оптимизацион­ных задач:

* задачи с неделимостями;
* экстремальные комбинаторные задачи;
* задачи с разрывными целевыми функциями;
* задачи на несвязных и невыпуклых областях и др.

**4.1.2. Задачи с неделимостями.** В подавляющем большин­стве случаев наличие условий неделимости определяется физи­ческими свойствами моделируемых объектов. Так, например, они могут появиться в качестве дополнительных ограничений в уже рассматривавшейся нами выше задаче производственного планирования, если в ней осуществляется управление выпуском крупной штучной продукции.

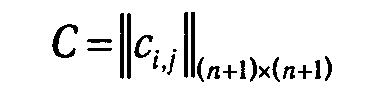
Классическим представителем задач данного класса стала так называемая *задача о ранце*. Ее фабула носит достаточно услов­ный характер и состоит в том, что солдат (или турист), собираю­щийся в поход, может нести груз весом не более *W* кг. Этот груз может состоять из набора предметов *n* типов, каждый предмет типа *j* весит *wj* кг и характеризуется некоторой «полезностью» *uj*, *j* ∊ 1: *n*. В рамках описанной ситуации вполне естественным представляется вопрос: сколько предметов каждого вида нужно положить в ранец, чтобы его суммарная полезность была макси­мальной? Если в качестве компонент плана *хj*. принять количе­ство укладываемых предметов типа *j*, то данную задачу можно записать:



Как нетрудно заметить, представленная математическая мо­дель носит универсальный характер, и к ней могут быть сведены многие экономические задачи. Ярким подтверждением этому служит и тот факт, что в литературе она также известна как задача о загрузке судна.

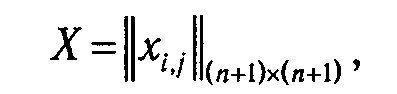
**4.1.3. Комбинаторные задачи.** К данному классу относят­ся задачи оптимизации функции, заданной на конечном множе­стве, элементами которого служат выборки из *n* объектов.

Классическим представителем математических проблем та­кого рода стала *задача о коммивояжере*. Она состоит в состав­лении маршрута посещения торговым агентом, находящимся в некотором начальном пункте, *n* других городов при условии, что задана матрица стоимостей переездов из города в город



(с учетом начального). Причем допустимым является такой мар­шрут, который предусматривает *однократное* посещение *всех* городов и возвращение в исходный пункт. Очевидно, что наилучший маршрут должен минимизировать суммарную сто­имость переездов.

Планом задачи является маршрут коммивояжера, и его мож­но задать с помощью так называемой матрицы смежности



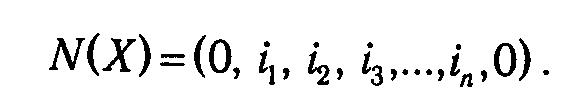
элементы которой определяются следующим образом:

1, если в маршруте предусмотрен переезд из пункта *i* в *j*,

*xi,j* = 0, если в маршруте не предусмотрен переезд из пункта *i* в *j*,

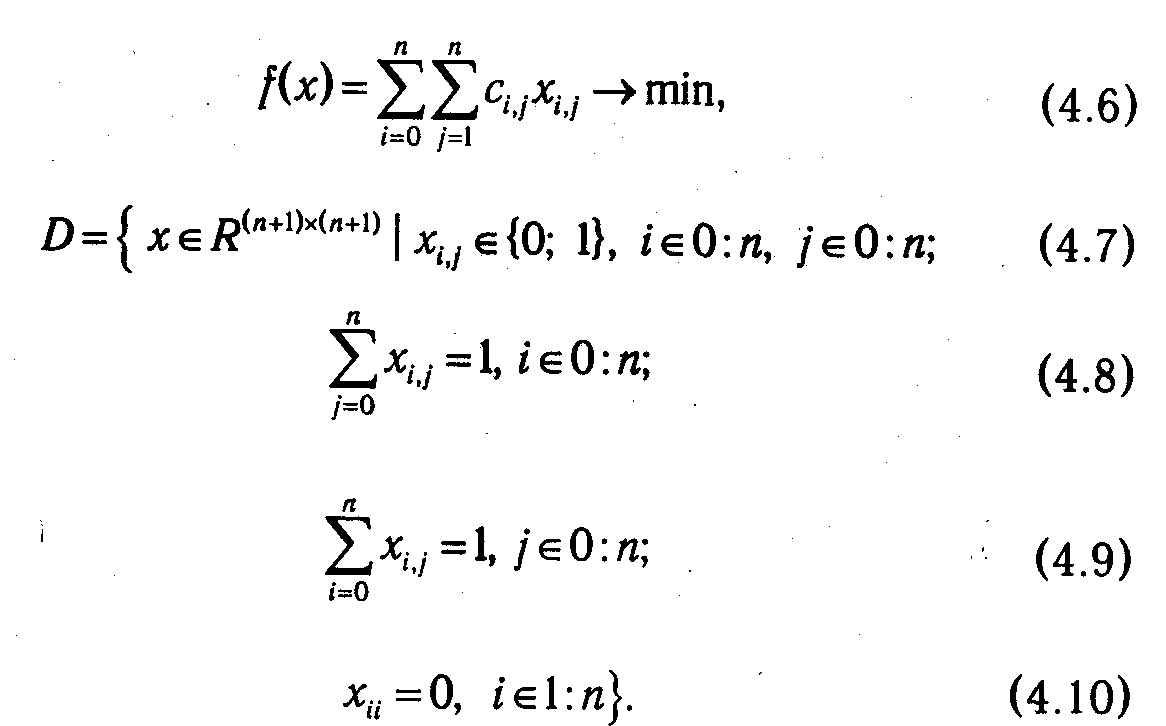
причем по условию задачи *xii* =0,  *i*∊1: *n*.

Допустимыми планами служат связные маршруты, однознач­но определяемые упорядоченным набором посещаемых пунктов:

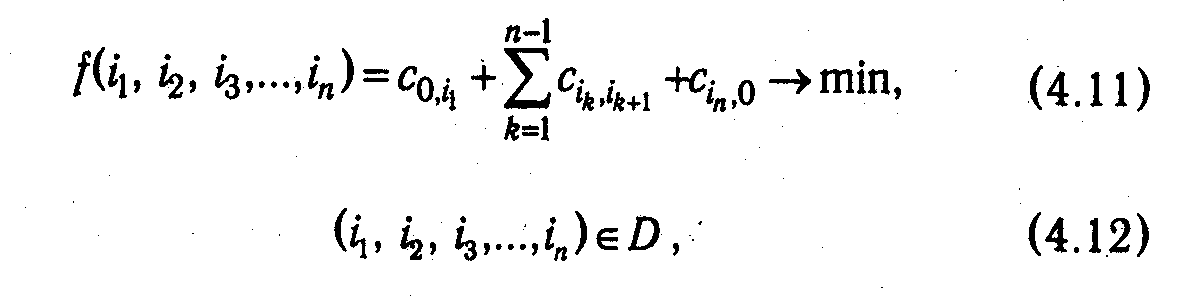


Каждый такой маршрут можно отождествить с перестанов­кой *n* чисел (упорядоченной выборкой из *n* элементов по *n*). В свою очередь, таким перестановкам взаимно однозначно со­ответствуют матрицы *X*, у которых в каждой строке и каждом столбце содержится точно одна единица.

С учетом сказанного задача коммивояжера принимает вид целочисленной задачи линейного программирования:



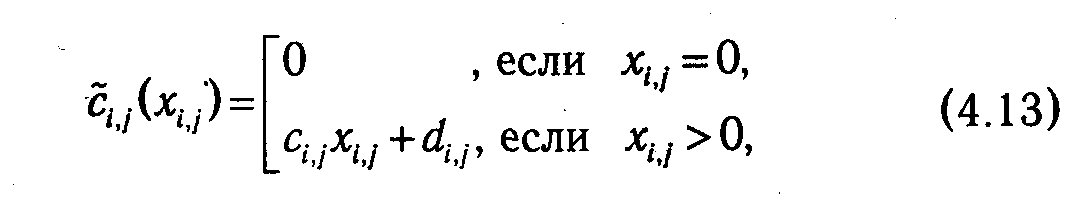
Условия (4.8) и (4.9) с содержательной точки зрения означа­ют, что в каждый пункт можно въехать и выехать только один раз. Приведенная форма записи задачи коммивояжера (4.6)-(4.10) не является самой рациональной и предназначена только для того, чтобы подчеркнуть ее общность с другими задачами дискретного программирования. Существует и другая форма, которая более ярко отражает комбинаторный характер данной проблемы:



где *D* — множество перестановок чисел от 1 до *n*.

Отдельно следует остановиться на том, что задача коммиво­яжера имеет большое количество содержательных аналогов. Скажем, к аналогичной модели приведет задача разработки графика переналадки оборудования, которое может выпускать разные типы изделий, но требует определенных затрат (времен­ных или материальных) при переходе с одного технологическо­го режима на другой.

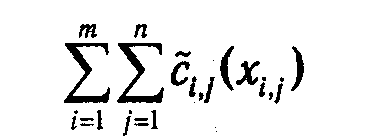
**4.1.4. Задачи с разрывными целевыми функциями.** Как уже упоминалось выше, многие экономические системы ха­рактеризуются наличием так называемых постоянных затрат, которые должны быть произведены независимо от объема про­изводства. Учет в моделях этих и подобных факторов приводит к появлению в них целевых функций, не обладающих свойст­вом непрерывности. В качестве примера может быть приведена *транспортная задача с фиксированными доплатами*. Она отличается от транспортной задачи в матричной постановке, рассмотренной в главе 3, тем, что в ней затраты по перевозке груза из *i*-го пункта производства в *j*-й пункт потребления опре­деляются как



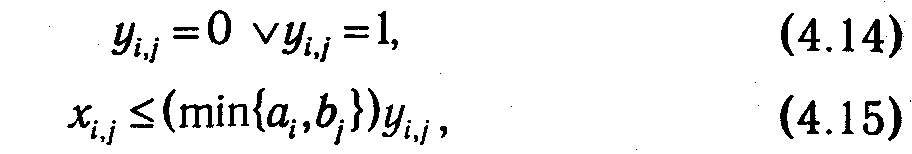
где *сi,j* — по-прежнему издержки на перевозку единицы груза;

*di,j* — фиксированная доплата за аренду транспортных средств.

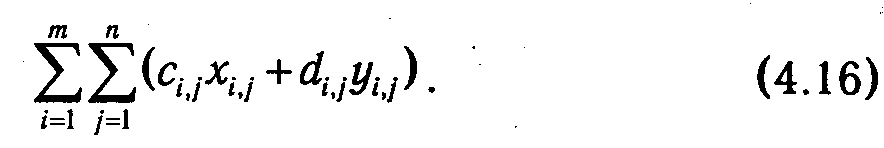
При таких предпосылках целевая функция суммарных затрат на перевозку



содержит «скачкообразные» разрывы, что существенно затруд­няет ее минимизацию, поэтому стандартный метод решения ос­нован на следующем преобразовании. Если ввести вспомога­тельные переменные *уi,j*, такие, что



то целевая функция примет вид



Действительно, если *уi,j* =0 , то переменные *хi,j* =0, а при *уi,j* =1 неравенства (4.15) становятся несущественными, поскольку они и так справедливы для любого опорного плана. Следова­тельно, задача (4.16) эквивалентна исходной задаче (4.13). В силу характера ограничений (4.14)-(4.15) задача (4.16) яв­ляется задачей частично-целочисленного программирования.

Перечисленные примеры далеко не исчерпывают всего мно­гообразия задач дискретного программирования. Однако более подробное их рассмотрение требует привлечения достаточно сложного математического аппарата и выходит за рамки дан­ной книги.

В последующих параграфах мы остановимся на способах ре­шения наиболее известных и хорошо изученных дискретных задач. Излагаемые ниже методы не имеют универсального ха­рактера, с каждым из них связаны определенные ограничения и, соответственно, ответ на вопрос о выборе того или иного из них зависит от конкретных особенностей решаемой задачи. Более того, цель изложения состоит в том, чтобы создать у читателя общие представления об основных идеях и подходах, не углубляясь далеко в вычислительные и математические тонкости, которыми буквально изобилуют алгоритмы дис­кретного программирования. Заметим также, что достаточно эффективный и широко применяемый подход к решению це­лочисленных задач основан на сведении их к задачам транс­портного типа. Это объясняется тем, что если в условиях транс­портной задачи значения запасов (*аi*) и потребностей (*bj*) являются целочисленными, то целочисленным будет и опти­мальный план.

## 4.2. МЕТОД ГОМОРИ

**4.2.1. Основные идеи и принципы**\*. Данный метод, кото­рый также носит название *метода отсекающих плоскостей*, предназначен для решения ЦЗЛП в канонической форме (4.2)-(4.3). Кратко представим его основные идеи.

\* Впервые был предложен Р.Гомори в 1957-1958 гг.

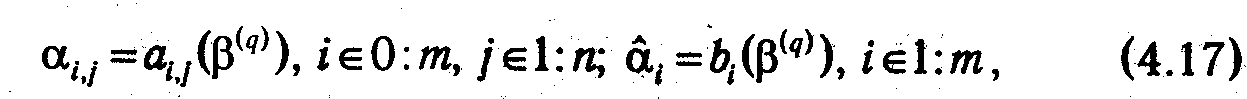
Отправной точкой для решения задачи (4.2)-(4.3) является решение ее непрерывного аналога, т. е. КЗЛП без учета условий целочисленности. Если получаемый в результате оптимальный план *х\** содержит только целые компоненты, то мы автомати­чески получаем и соответствующее решение ЦЗЛП. В против­ном случае к системе ограничений задачи должно быть добавле­но такое ограничение, для которого:

* найденный нецелочисленный оптимальный план *х*\* не удов­летворяет вновь добавляемому ограничению;
* любой допустимый целочисленный план непрерывной задачи (4.2)-(4.3) удовлетворяет вновь добавляемому ограниче­нию.

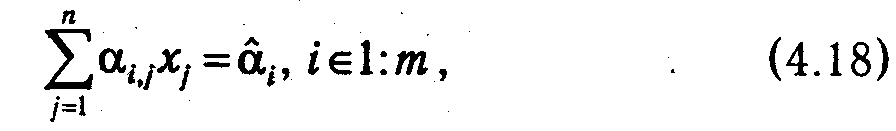
Такое ограничение называют *правильным отсечением.* В первой геометрической интерпретации правильному отсече­нию соответствует гиперплоскость, отсекающая от выпуклого многогранного множества допустимых планов *D* некоторый мно­гогранник, не содержащий целочисленных планов.

Добавив сформированное отсекающее ограничение к уже су­ществующим, мы получаем новую оптимизационную задачу, после чего вычислительный процесс итеративно повторяется.

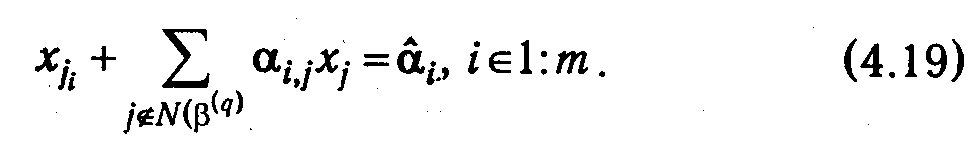
Теперь необходимо несколько более подробно остановиться на принципах формирования отсекающих ограничений. Вос­пользуемся системой обозначений, применявшихся при изло­жении вычислительных методов линейного программирования. Пусть β(*q*) — оптимальный базис, полученный на последней итерации решения нецелочисленной ЗЛП. Если обозначить через α*i,j*  и ά*i*  коэффициенты матрицы задачи и вектора ограниче­ний в текущем базисе (*A*(β(*q*)) и *b*(β(*q*)))



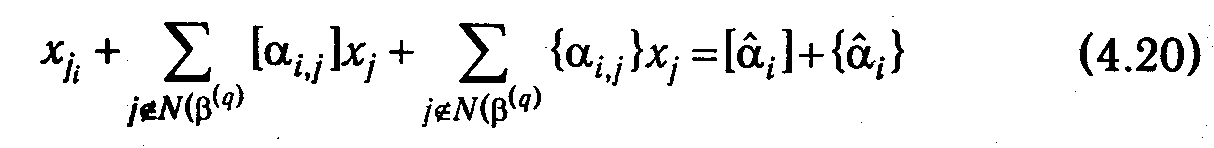
то *i*-ое уравнение в системе ограничений задачи примет вид:



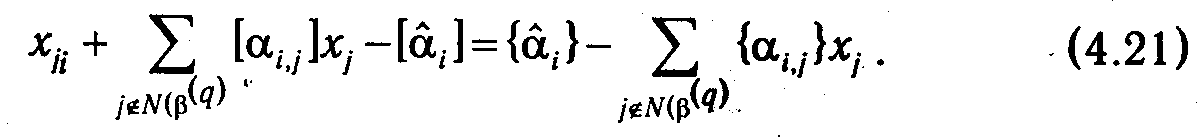
Так как план *x*(β(*q*)) является базисным, то в каждом уравнении все коэффициенты α*i,j*, соответствующие базисным столбцам (*j*∊*N*(β(*q*))), равны нулю за исключением некоторого α*i,ji* =1, на основе чего из (4.18) получаем:



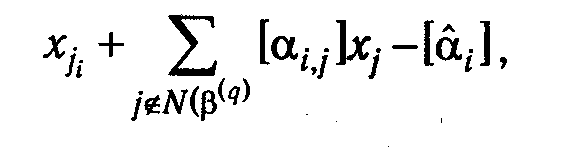
Если представить каждый коэффициент α*i,j* в виде суммы це­лой и дробной частей α*i,j* =[α*i,j*]+{α*i,j*}, то получим



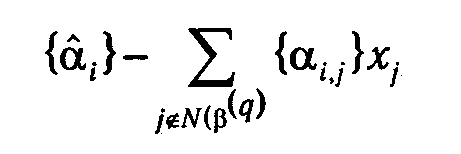
или



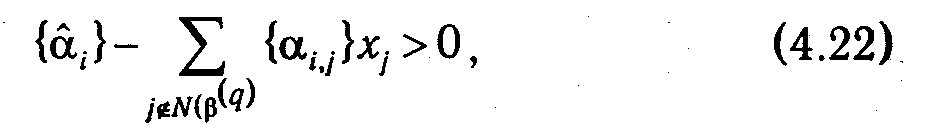
Из (4.21) следует, что если все *хj*, *j*∊1: *n* являются целыми, то целым будет и выражение



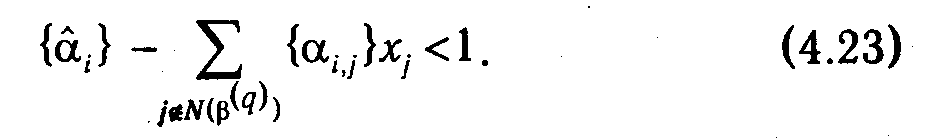
стоящее в левой части (4.21), и, стало быть, правая часть данно­го уравнения:



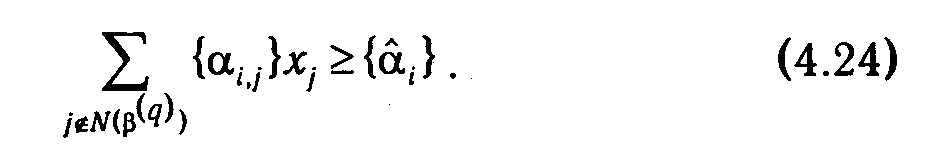
также должна быть целой. Предположим, что



тогда, в силу того, что 0 ≤ {ά*i*} < 1, а {α*i,j*} ≥ 0, *xj* ≥ 0, должно вы­полняться неравенство

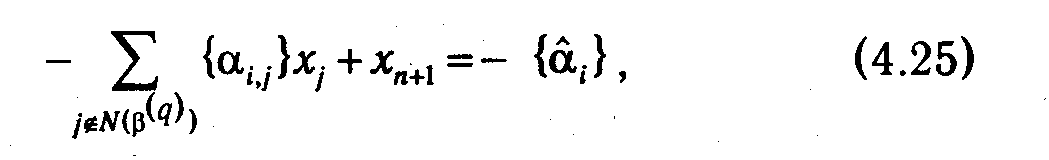


Однако неравенства (4.22) и (4.23) противоречат требуемой целочисленности правой части (4.21) *xj*(β(*q*)). Следовательно, для целочисленных решений должно выполняться условие, проти­воположное неравенству (4.22), или, что то же самое,



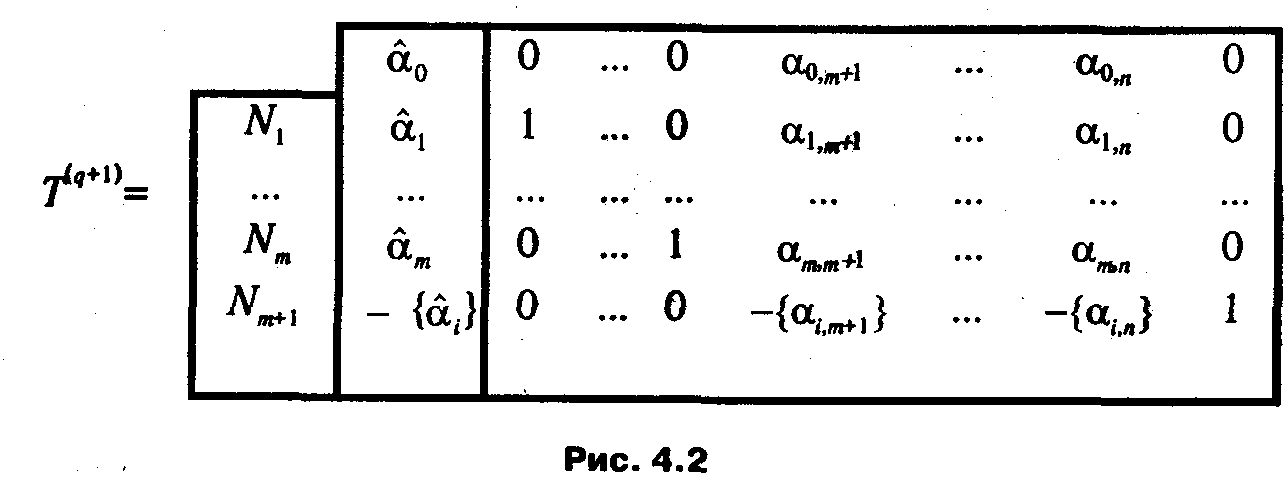
В то же время (4.24) не выполняется для любого нецелочислен­ного *базисного* плана *х*. Действительно, небазисные компонен­ты плана равны нулю: *хj* =0 , *j*∊*N*(β(*q*)), и (4.24) приобретает вид {ά*i*} ≤0 <=> {ά*i*} =0, но это противоречит предположению о нецелочисленности плана *х*, т. к. в базисном плане *хi* = ά*i* . Все сказанное позволяет утверждать, что ограничение (4.24) задает *правильное отсечение.*

Таким образом, с точки зрения организации техники, вычис­лений для осуществления правильного отсечения мы должны к системе ограничений нецелочисленной линейной задачи, реша­емой на *q*-й итерации, добавить условие



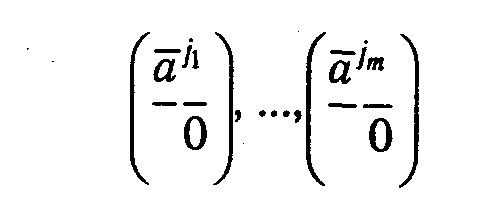
где *xn*+1 ≥0 — фиктивная переменная, добавляемая для преоб­разования неравенства в строгое равенство. Ей соответствует нулевой коэффициент в целевой функции.

Данному преобразованию условий задачи будет соответ­ствовать преобразование симплекс-таблицы, показанное на *рис. 4.2.* На нем по соображениям обеспечения наглядности ис­пользованы обозначения (4.17) и предполагается, что текущий базис β(*q*) состоит из первых *m* столбцов.

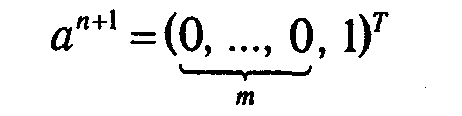


Индекс *i* соответствует выбранной для формирования отсе­чения строке симплекс-таблицы, содержащей нецелочисленное значение *bi*(β(*q*)).

Как видно из *рис. 4.2*, технически преобразование таблицы сводится к дописыванию одной строки и одного столбца. При этом легко убедиться, что модифицированные столбцы



совместно с добавленным столбцом



образуют сопряженный (двойственно допустимый) базис для сформированной задачи, а (ά1, ..., ά*m*, -{ά*i*}) являются ненуле­выми компонентами соответствующего псевдоплана. Исходя из этого, приходим к тому, что *для решения вновь получен­ной задачи может быть эффективно применена процедура двойственного симплекс-метода* (см. параграф 1.7).

Поскольку в начальном псевдоплане имеется только одна от­рицательная компонента (-{ά*i*}), то из базиса должен быть вы­веден соответствующий ей вектор *аn*+1 . Далее, следуя рекомен­дациям алгоритма двойственного симплекс-метода, находим оптимальный план. Если он не является целочисленным, то опи­санные действия итеративно повторяются.

Если в ходе решения дополнительная переменная *хn*+1 вновь становится базисной, ее значение оказывается безразличным для основных переменных. Поэтому строку и столбец, отвеча­ющие ей, вычеркивают. С геометрической точки зрения это можно обосновать так: если псевдоплан оказывается внутри полупространства *хn*+1 ≥0, то дополнительное ограничение, оп­ределяемое гиперплоскостью *хn*+1=0, становится несущест­венным и опускается.

**4.2.2. Описание алгоритма.** Приведем обобщенную схе­му алгоритма Гомори. Структурно он делится на так называе­мые большие итерации. Каждая большая итерация содержит этапы:

1. Решение «текущей» задачи методами линейного програм­мирования (малые итерации). На первой итерации в качестве «текущей» задачи выступает нецелочисленный аналог исходной ЦЗЛП.

2. Определение первой нецелочисленной компоненты в оп­тимальном плане, полученном на этапе 1. Если все компоненты являются целочисленными, то алгоритм завершается.

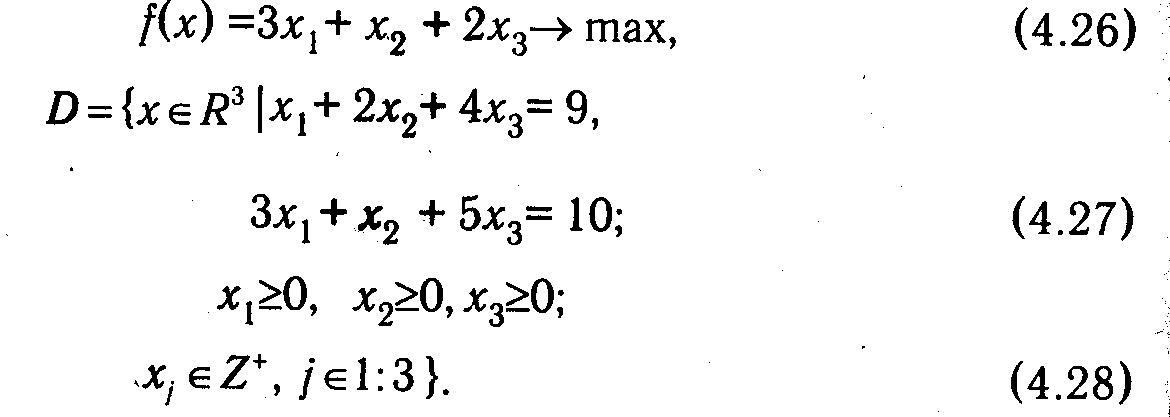
3. Построение для найденной компоненты условия отсече­ния согласно правилу (4.24), добавление сформированного ог­раничения к системе ограничений текущей задачи, т. е. форми­рование новой текущей задачи. Переход на начало следующей большой итерации.

Двойственный симплекс-метод является основой для метода Гомори, так как он позволяет учитывать новые дополнитель­ные ограничения (правильные отсечения) и переходить от те­кущего псевдоплана к новому оптимальному плану.

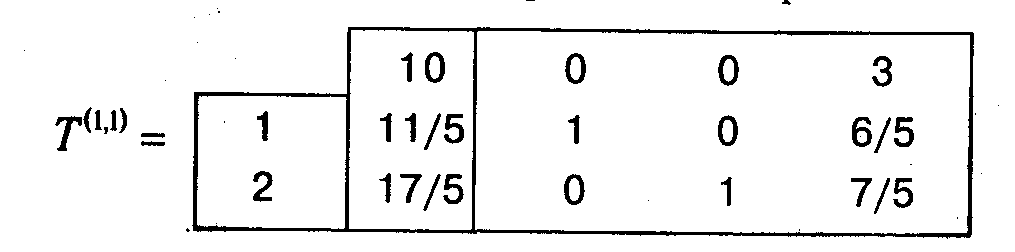
Можно доказать, что приведенный алгоритм конечен. Это означает, что на некотором шаге (итерации) будет найден цело­численный оптимальный план или обнаружен факт отсутствия допустимых целочисленных планов.

В качестве существенного замечания по поводу метода Гомори следует добавить, что при его практической реализации на ЭВМ следует считаться с ошибками округления, т, к. в усло­виях машинной арифметики практически ни один план не будет целочисленным. Кроме того, накапливающиеся погрешности могут внести возмущения в алгоритм и «увести» от оптимально­го целочисленного плана.

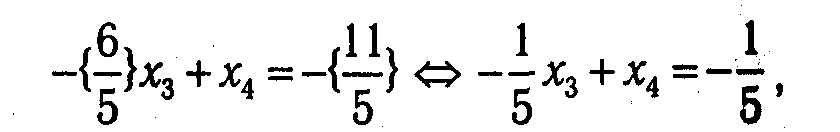
**4.2.3. Пример решения ЦЗЛП методом Гомори.** Рас­смотрим особенности применения метода Гомори на конкрет­ном примере. Пусть дана задача со следующими условиями:



**Итерация 1.** Используя обычный симплекс-алгоритм, реша­ем непрерывный аналог исходной задачи, в котором игнориру­ются условия целочисленности (4.28). В качестве исходного базиса можно взять первый и второй столбцы. На его основе заполняется таблица *T*(1,1) (первый индекс в обозначении табли­цы соответствует «большой» итерации, а второй — «малой»).

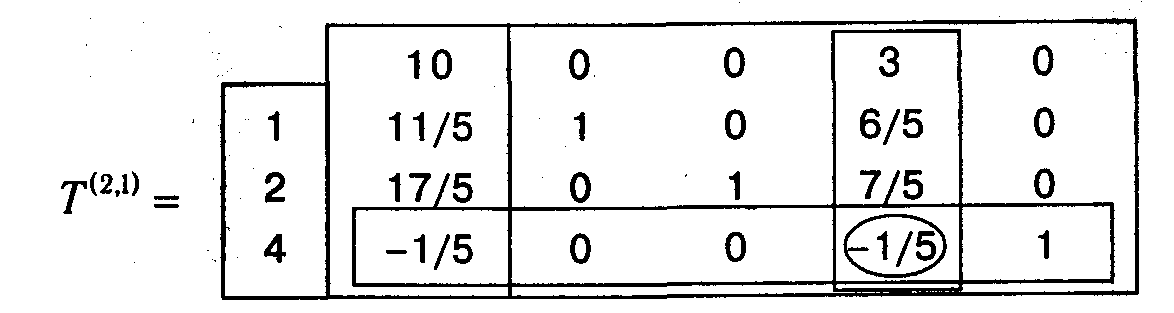


Как видно из строки оценок, данный базис является опти­мальным, однако соответствующий ему план *х* ={11/5,17/5, 0) не является целочисленным, поэтому выбираем из таблицы *T*(1,1) строку, содержащую первый нецелый элемент, и согласно формуле (4.25) строим отсекающее ограничение:

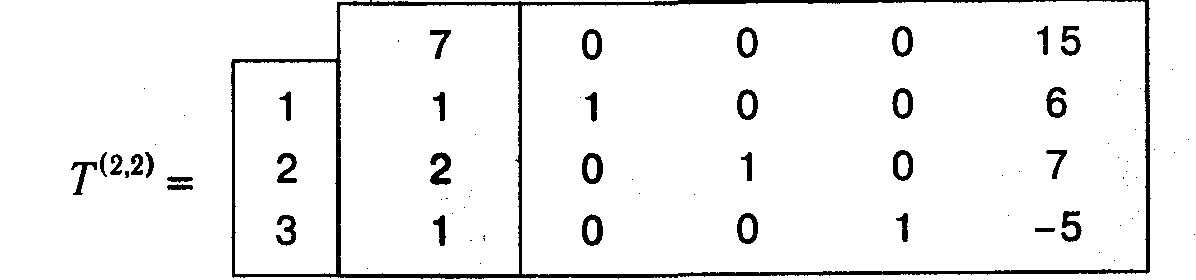


после чего переходим к следующей «большой» итерации.

**Итерация 2.** С учетом сформированного отсекающего ог­раничения заполняем симплекс-таблицу *T*(2,1).



В соответствии с алгоритмом двойственного симплекс-мето­да переходим к следующему базису *N*(β(2,2))={1, 2, 3}.



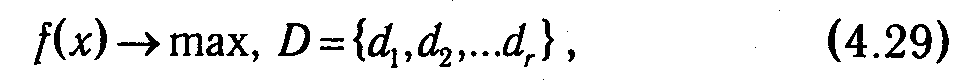
План, достигнутый в таблице *T*(2,2), является не только опти­мальным (*b*(β(2,2))>0), но и полностью состоит из целочислен­ных компонент, т. е. решение задачи найдено: *х*\* = (1, 2, 1) и *f(x)=*7.

## 4.3. МЕТОД ВЕТВЕЙ И ГРАНИЦ

**4.3.1. Общая схема метода «ветвей и границ».** Другим широко применяемым для решения задач дискретного програм­мирования методом является *метод ветвей и границ*. Впервые данный метод для решения ЦЗЛП предложили в 1960 г. Лэнг и Дойг, а его «второе рождение» произошло в 1963 г. в связи с выходом работы Литтла, Мурти, Суини и Кэрел, посвященной решению задачи о коммивояжере [33].

Вообще говоря, термин «метод ветвей и границ» является соби­рательным и включает в себе целое семейство методов, применяе­мых для решения как линейных, так и нелинейных дискретных задач, объединяемое общими принципами. Кратко изложим их.

Пусть стоит задача:

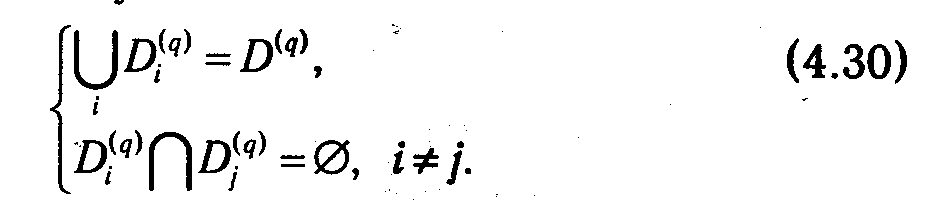


где *D* — конечное множество.

Алгоритм является итеративным, и на каждой итерации про­исходит работа с некоторым подмножеством множества *D*. На­зовем это подмножество *текущим* и будем обозначать его как *D*(*q*), где *q* — индекс итерации. Перед началом первой итерации в качестве текущего множества выбирается все множество *D* (*D*(1)*=D*), и для него некоторым способом вычисляется значе­ние верхней оценки для целевой функции max *f(x)* ≤ ξ( *D*(1)). Стандартная итерация алгоритма состоит из следующих этапов:

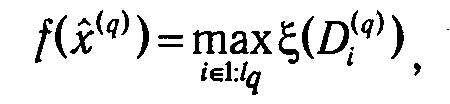
1°. Если можно указать план *x*(*q*)∊*D*(*q*), для которого *f(x*(*q*)*)*≤ξ( *D*(*q*)), то *x*(*q*)=*х\** — решение задачи (4.29).

2°. Если такой план не найден, то область определения *D*(*q*) некоторым образом разбивается на подмножества *D*1(*q*), *D*2(*q*), ..., *Dlq*(*q*), удовлетворяющие условиям:

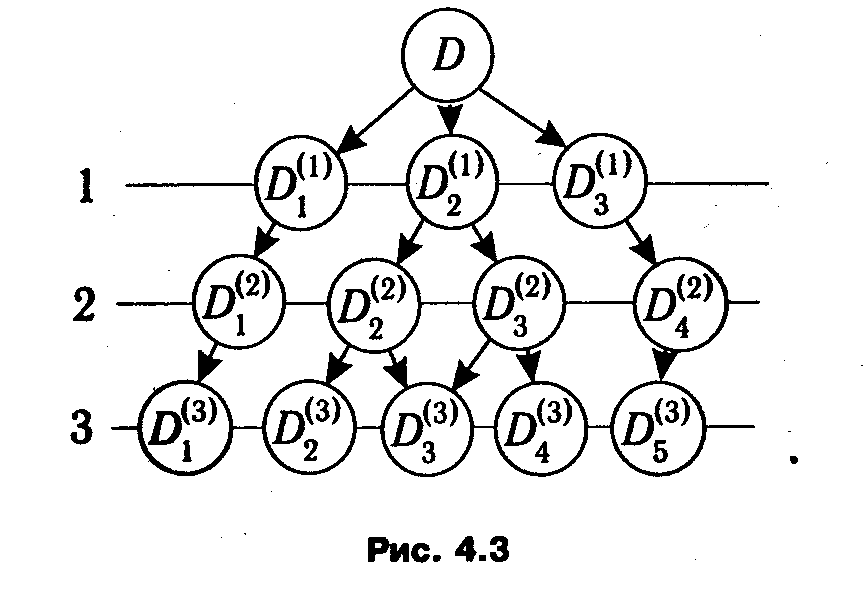


Для каждого подмножества находятся оценки сверху (вер­хние границы) для целевой функции ξ*D*1(*q*), ξ*D*2(*q*), ..., ξ*Dl*1(*q*), уточняющие ранее полученную оценку ξ*D*(*q*), то есть ξ*Di*(*q*) ≤ ξ*D*(*q*), *i*∊1:*lq*. Возможно одно из двух:

2.1. Если существует такой план *х*(*q*), что



то этот план оптимальный.



2.2. Если такой план не найден, то выбирается одно из мно­жеств *Di*(*q*),*i*∊1:*lq* (как правило, имеющее наибольшую оценку

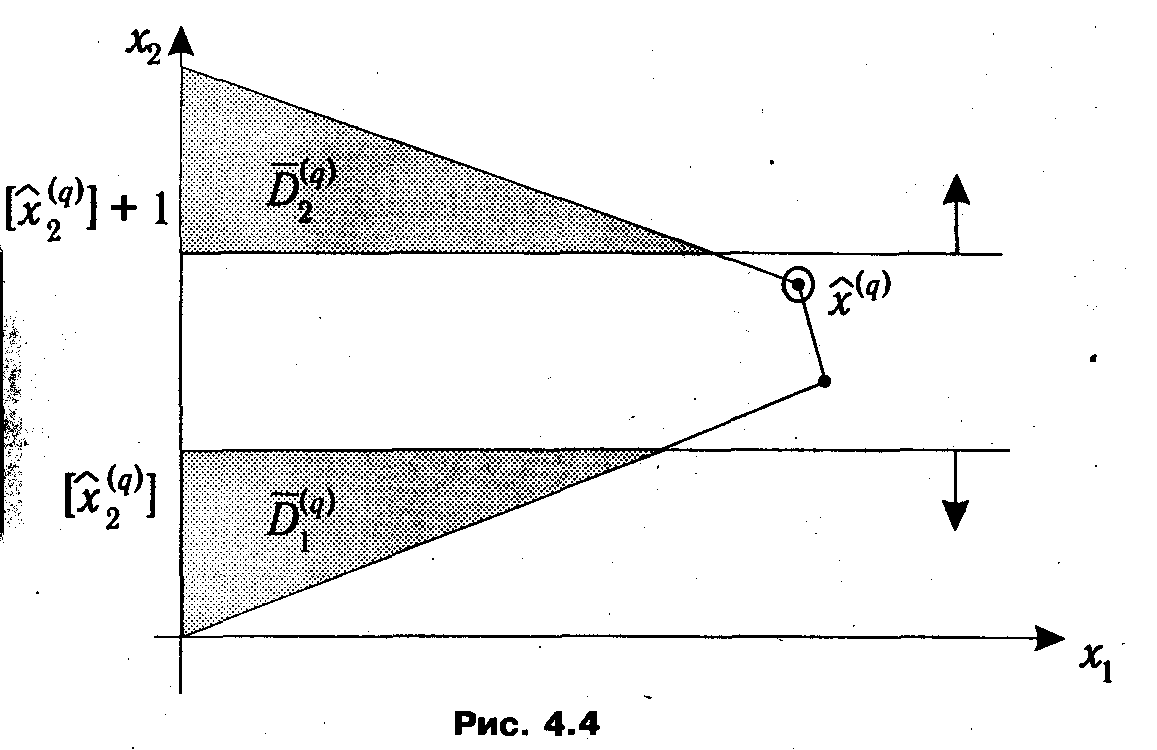


Все имеющиеся к текущему моменту концевые подмножества, т. е. те подмножества, которые еще не подверглись процессу дробления, переобозначаются как *D*1(*q*+1),*D*2(*q*+1),..., *Dl*(*q*+1)(*q*+1),после чего процесс повторяется.

Схема дробления множества *D* представлена на *рис. 4.3* в виде графа. Существуют и более сложные системы индексации подмножеств, при которых не требуется их переобозначение на каждом шаге.

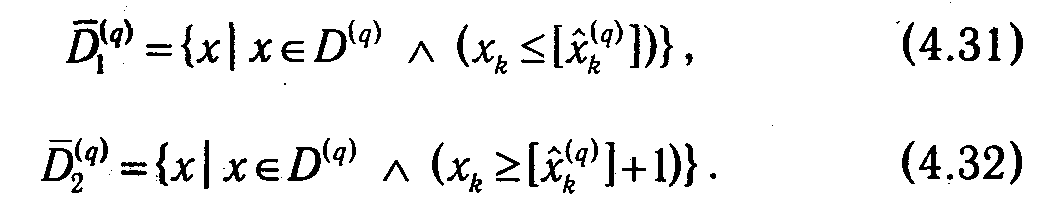
Конкретные реализации метода ветвей и границ связаны с правилами разбиения на подмножества (правилами ветвления) и построения оценок значений целевых функций на данных под­множествах (границ).

**4.3.2. Решение ЦЗЛП методом ветвей и границ.** Рас­смотрим применение алгоритма метода ветвей и границ для решения ЦЗЛП (4.2)-(4.3). Как уже упоминалось, через *D*(*q*) обозначается подмножество множества допустимых планов за­дачи. Перед началом первой итерации (*q* = 1) в качестве теку­щего множества берется все множество *D* (*D*(1) = *D*), после чего решается стандартная задача линейного программирования (*D*(1), *f*). Нетрудно заметить, что она является непрерывным аналогом



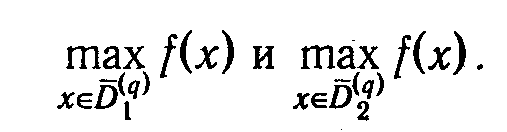
исходной задачи (4.2)-(4.3). Если найденный оптималь­ный план **(1) содержит только целочисленные компоненты, то он является и оптимальным планом для (4.2)-(4.3): ** (1) = *x\**. В противном случае значение *f*(** (1)) становится оценкой (верх­ней границей) значения целевой функции на множестве *D*(1), и мы переходим к выполнению стандартной итерации алгоритма. Опишем входящие в нее этапы.

1) Выбирается некоторая нецелочисленная компонента пла­на *k*(*q*). Поскольку в оптимальном плане она должна быть це­лой, то можно наложить ограничения *xk* ≤ [*k*(*q*)] и *xk* ≥ [*k*(*q*)]+1. Таким образом, *D*(*q*) разбивается на подмножества

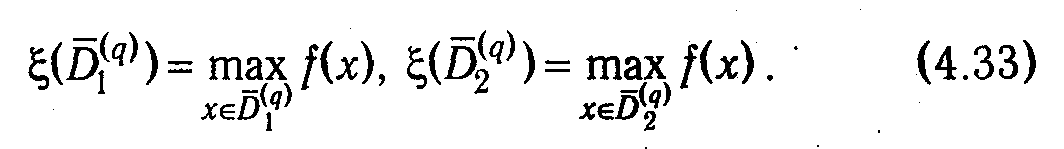


Графическая интерпретация такого разбиения множества *D*(*q*) приведена на *рис. 4.4.*

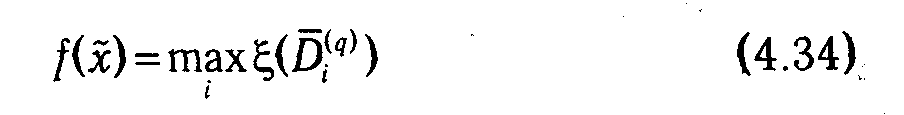
2) Решаются задачи линейного программирования



Соответствующие максимальные значения целевой функции принимаются как ее оценки на этих множествах:



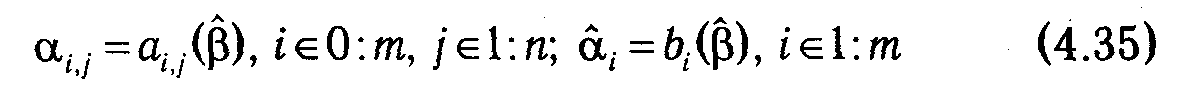
Если оптимальный план ** для одной из решенных задач удов­летворяет условию



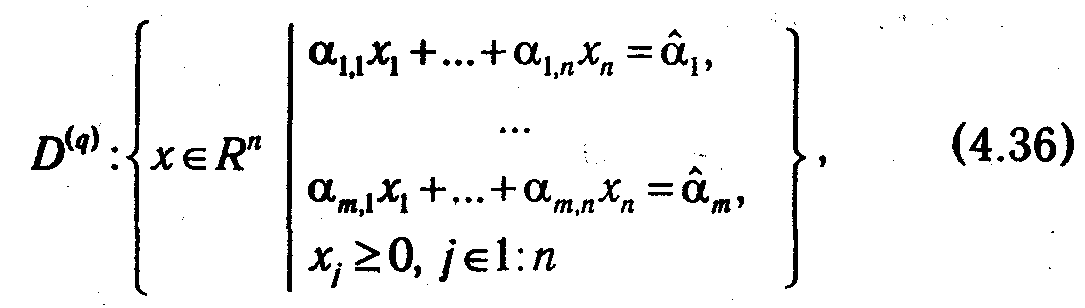
и содержит только целые компоненты, то, значит, найдено ре­шение основной задачи (4.2)-(4.3). В противном случае среди **всех концевых подмножеств**, полученных как на предыду­щих (*Di*(*q*)), так и на текущем (*D*1(*q*), *D*2(*q*)) этапе, выбирается об­ласть с наибольшей оценкой ξ(*Di*(*q*)). Она становится текущим рассматриваемым подмножеством (*D*(*q*+1)). Далее производится перенумерация концевых множеств и вычислительный процесс итеративно повторяется.

При решении задач (*D*1(*q*), *f*) и (*D*2(*q*), *f*) можно воспользовать­ся результатами решения предыдущей задачи (*D*(*q*), *f*). Рас­смотрим вариант организации вычислительного процесса на примере задачи (**1(*q*), *f*) (для (**2(*q*), *f*) он выглядит аналогично с точностью до знаков неравенств).

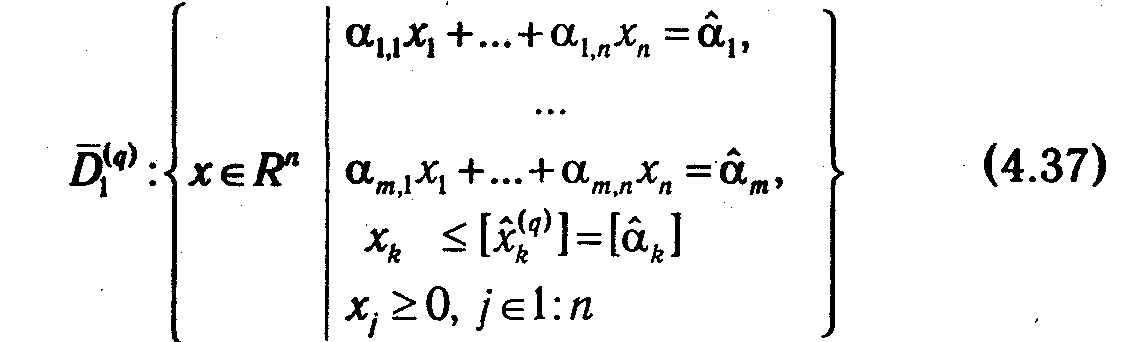
Предположим, что на последнем шаге решения задачи (*D*(*q*), *f*) был получен оптимальный базис β. Без ограничения общности можно считать, что он состоит из первых *m* столбцов матрицы задачи. Данное предположение делается исключитель­но для обеспечения наглядности дальнейшего изложения и оче­видно, что его выполнения можно всегда добиться за счет про­стой перенумерации векторов *аj*. По аналогии с предыдущим параграфом введем обозначения для элементов матрицы задачи (*D*(*q*), *f*) и ее вектора ограничений относительно базиса :



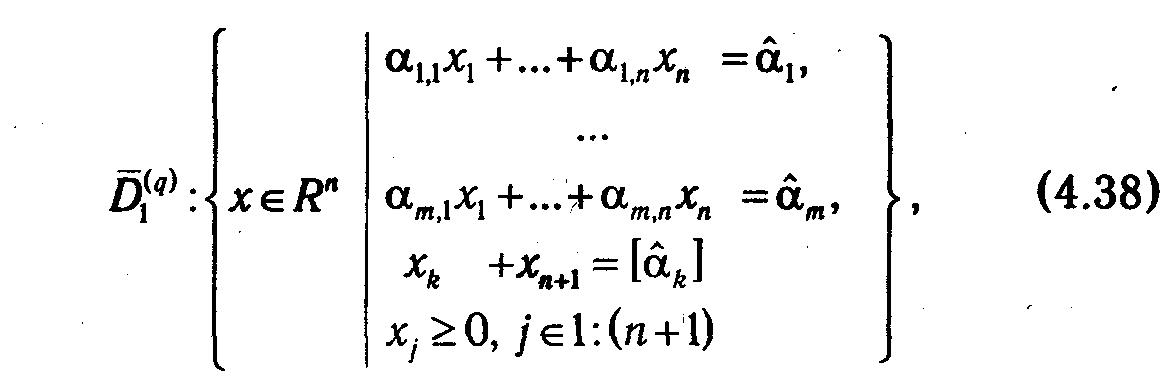
Тогда система ограничений задачи (*D*(*q*), *f*) может быть пред­ставлена как



а получаемая на ее основе система ограничений задачи (**1(*q*), *f*) как

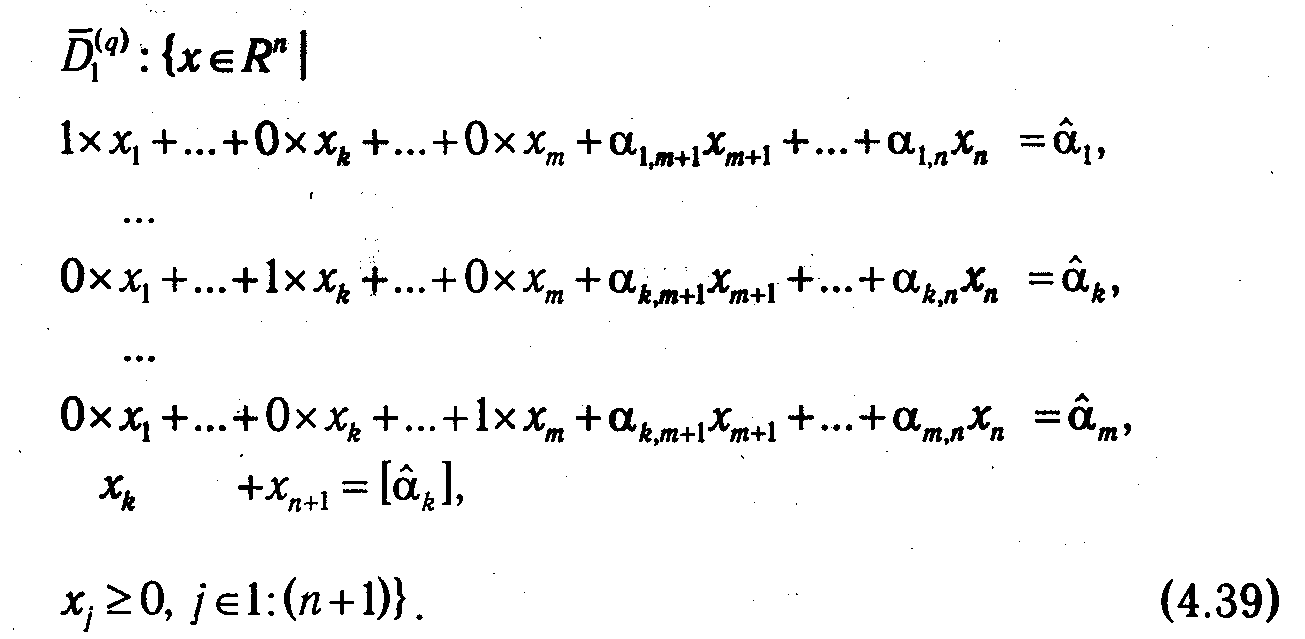


или

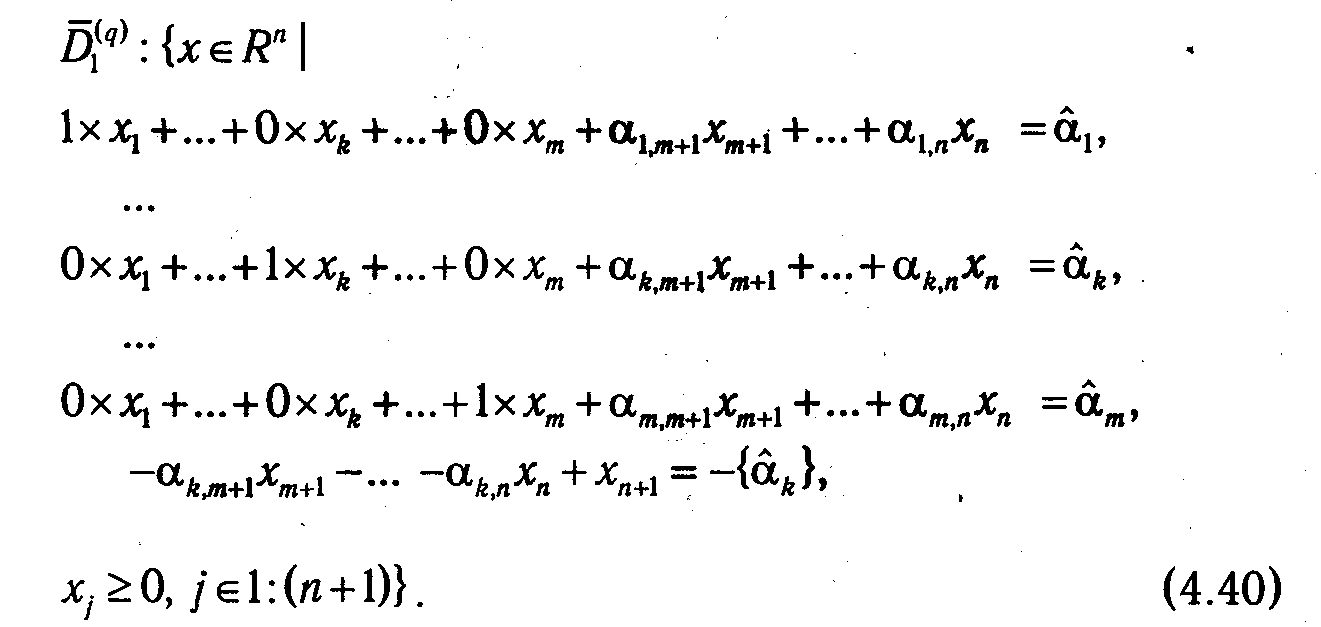


где *хn*+1 ≥ 0 — фиктивная переменная, которой соответствует нулевой коэффициент в целевой функции, добавляемая для пре­образования неравенства в строгое равенство.

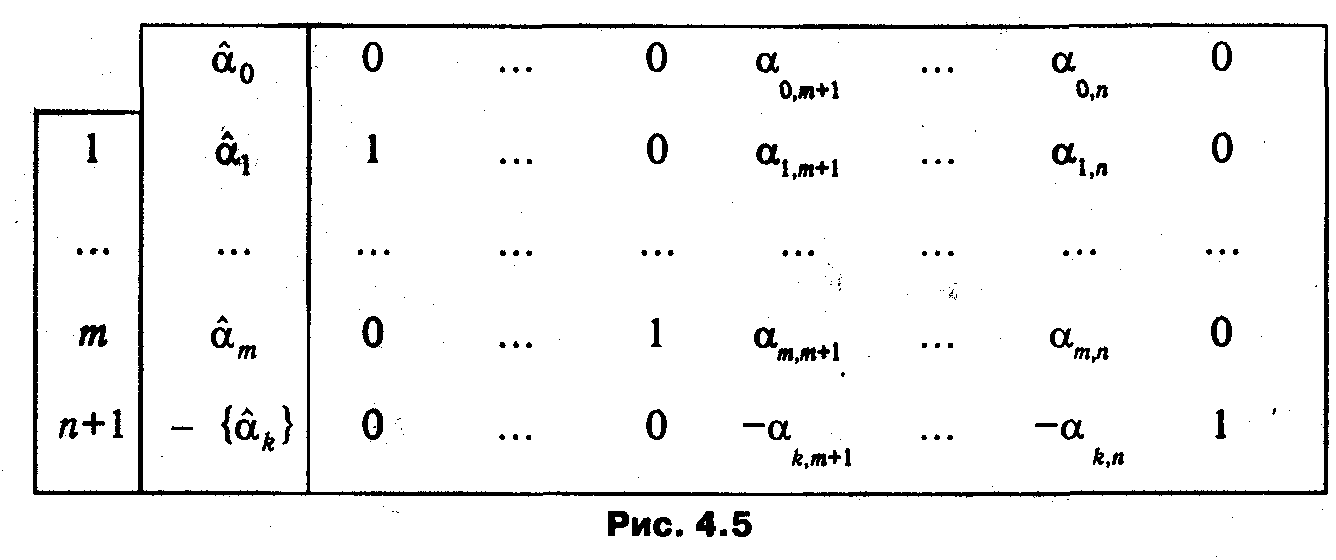
Очевидно, что 1≤*k≤m*, т. к. небазисные компоненты опти­мального плана (*m*+1≤*j≤n*) равны нулю, т. е. являются заведо­мо целочисленными. Тогда с учетом сделанных предположений о виде базиса  можно записать:



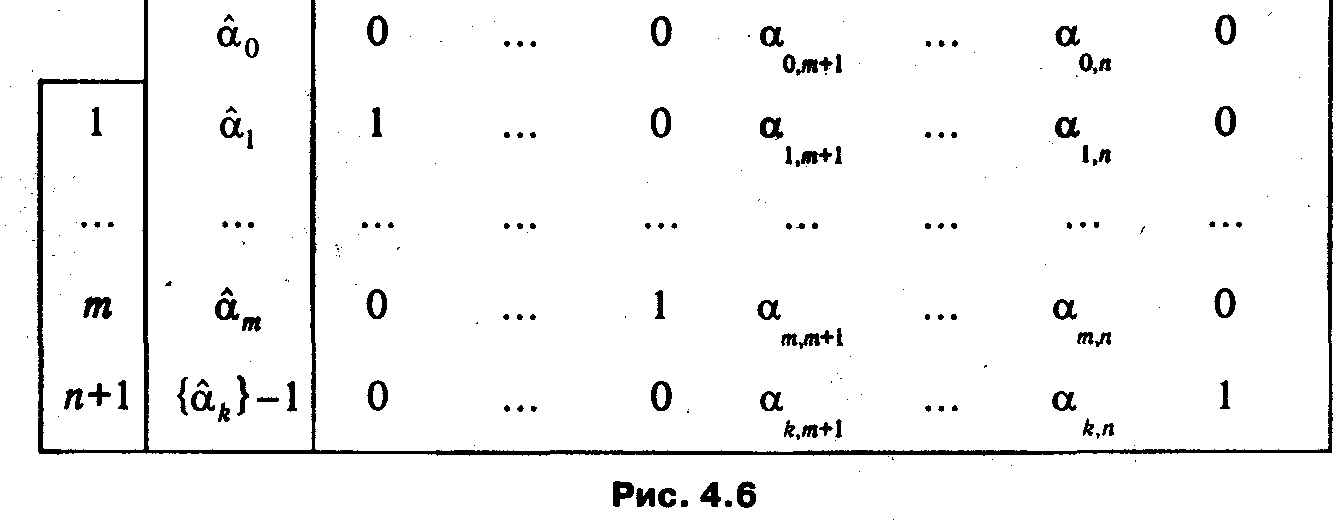
Как видно из (4.39), в *k*-м столбце имеется всего два отлич­ных от нуля элемента: в *k*-й и (*m*+1)-й строках. Если вычесть из (*m*+1)-го уравнения *k*-e, то, учитывая, что [ά*k*] – ά*k* =-{ά*k*}, по­лучим эквивалентную систему:



Проведенные преобразования системы ограничений *D*1(*q*) по­зволили явно выделить сопряженный базис, образуемый столб­цами с номерами 1,..., *m*, *n*+1, и соответствующий ему псевдо­план (ά1, ..., ά*m*, 0,...., 0, -{ά*k*}), т.е. для решения задачи (*D*1(*q*), *f*) может быть применен алгоритм двойственного симплекс-мето­да. Практически вычислительный процесс для данного этапа сводится к преобразованию к симплекс-таблицы, показанному на *рис. 4.5.*



Для случая задачи (*D*2(*q*), *f*) преобразование симплекс-табли­цы, получаемое на базе аналогичных рассуждений, приведено на *рис. 4.6.*



Очевидным недостатком алгоритма метода ветвей и границ при решении задач большой размерности является необходи­мость перебрать слишком большое количество вариантов пе­ред тем, как будет найден оптимальный план. Однако он отчасти может быть преодолен, если ограничиться поиском не опти­мального, а просто «хорошего» (близкого к оптимальному) пла­на. О степени такой близости и скорости приближения к экст­ремуму нетрудно судить по изменению значений оценок.

Подчеркнем, что приведенная реализация метода ветвей и границ является *одной из многих*. Помимо нее, например, очень популярна версия метода решения задачи коммивояжера, в которой для ветвления и построения оценок используют специфические свойства данной модели. При желании о ней мож­но прочесть в [1, 14].

## КЛЮЧЕВЫЕ ПОНЯТИЯ

* Задачи с неделимостями.
* Экстремальные комбинаторные задачи.
* Задачи с разрывными целевыми функциями.
* Правильное отсечение.
* Метод Гомори.
* Методы ветвей и границ.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

4.1. Какие основные проблемы возникают при решении дис­кретных задач?

4.2. Сформулируйте задачу о ранце.

4.3. Какие экономико-математические модели могут быть све­дены к задаче о коммивояжере?

4.4. Приведите пример моделей с разрывными целевыми функ­циями.

4.5. Какой принцип используется для построения правильно­го отсечения в методе Гомори?

4.6. Перечислите основные этапы, входящие в «большую» итерацию метода Гомори.

4.7. Какую роль играет алгоритм двойственного симплекс-ме­тода при решении целочисленной

линейной задачи мето­дом Гомори?

4.8. Перечислите принципиальные идеи, лежащие в основе ме­тодов ветвей и границ.

4.9. Как производится построение отсечения при решении це­лочисленной линейной задачи методом

ветвей и границ?

4.10. Опишите схему решения целочисленной задачи линейно­го программирования методом ветвей и

границ.

4.11. За счет каких преобразований удается построить сопря­женный базис при добавлении

отсекающего ограничения?

# ГЛАВА 5. ДИНАМИЧЕСКОЕ ПРОГРАММИРОВАНИЕ

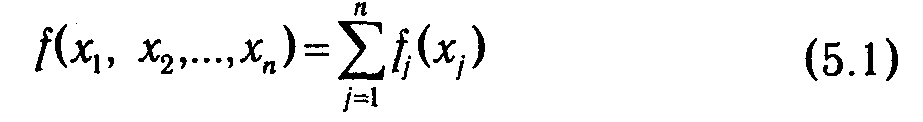
## 5.1. ОБЩАЯ СХЕМА МЕТОДОВ

## ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

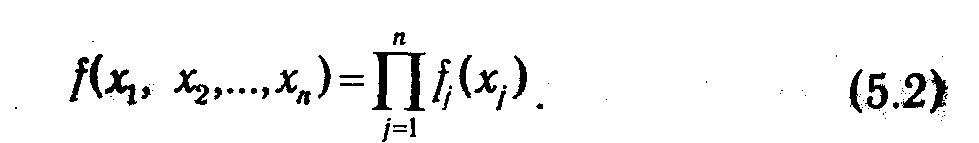
**5.1.1. Основные идеи вычислительного метода динами­ческого программирования.** Некоторые задачи математиче­ского программирования обладают специфическими особенно­стями, которые позволяют свести их решение к рассмотрению некоторого множества более простых «подзадач». В результа­те вопрос о глобальной оптимизации некоторой функции сводится к поэтапной оптимизации некоторых промежуточных целевых функций. В *динамическом программировании\** рас­сматриваются методы, позволяющие путем поэтапной (много­шаговой) оптимизации получить общий (результирующий) оп­тимум.

\* Динамическое программирование как научное направление возник­ло и сформировалось в 1951-1953 гг. благодаря работам Р. Беллмана и его сотрудников.

Обычно методами динамического программирования опти­мизируют работу некоторых *управляемых систем*, эффект ко­торой оценивается *аддитивной*, или *мультипликативной*, целевой функцией. **Аддитивной** называется такая функция не­скольких переменных *f*(*х*1, *x*2, *...,хn*), значение которой вычис­ляется как сумма некоторых функций *fj*, зависящих только от одной переменной *хj*:

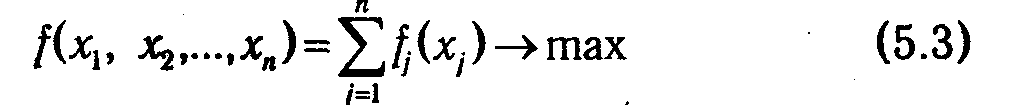


Слагаемые аддитивной целевой функции соответствуют эффек­ту решений, принимаемых на отдельных этапах управляемого процесса. По аналогии, **мультипликативная** функция распа­дается на произведение положительных функций различных пе­ременных:

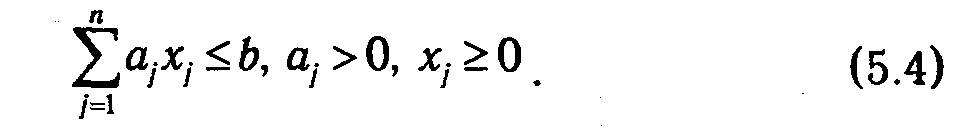


Поскольку логарифм функции типа (5.2) является аддитив­ной функцией, достаточно ограничиться рассмотрением функций вида (5.1).

Изложим сущность вычислительного метода *динамическо­го программирования* на примере задачи оптимизации

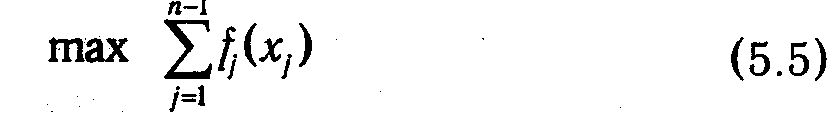


при ограниченияx

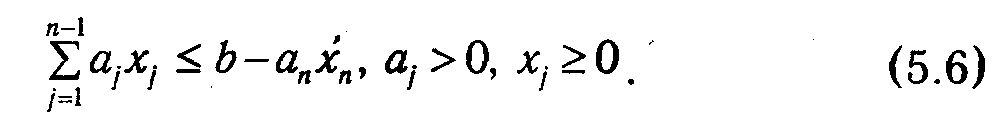


Отметим, что в отличие от задач, рассмотренных в преды­дущих главах, о линейности и дифференцируемости функций *fj*(*xj*) не делается никаких предположений, поэтому примене­ние классических методов оптимизации (например, метода Лагранжа) для решения задачи (5.3)-(5.4) либо проблематично, либо просто невозможно.

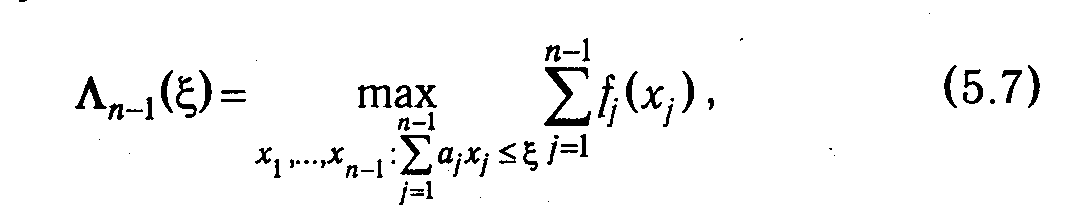
Содержательно задача (5.3)-(5.4) может быть интерпрети­рована как проблема оптимального вложения некоторых ресур­сов *j*, приводимых к единой размерности (например, денег) с помощью коэффициентов *aj*, в различные активы (инвестици­онные проекты, предприятия и т. п.), характеризующиеся фун­кциями прибыли *fj*, т. е. такого распределения ограниченного объема ресурса (*b*), которое максимизирует суммарную прибыль. Представим ситуацию, когда она решается последова­тельно для каждого актива. Если на первом шаге принято реше­ние о вложении в *n*-й актив *xn* единиц, то на остальных шагах мы сможем распределить *b-anxn* единиц ресурса. Абстрагируясь от соображений, на основе которых принималось решение на первом шаге (допустим, мы по каким-либо причинам не могли на него повлиять), будет вполне естественным поступить так, чтобы на *оставшихся* *шагах* *распределение* *текущего* *объе­ма ресурса произошло оптимально*, что равнозначно реше­нию задачи



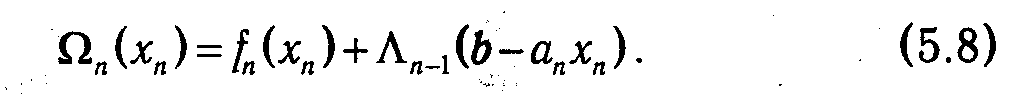
при ограничениях



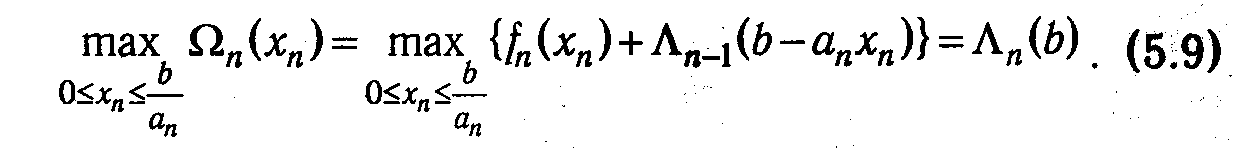
Очевидно, что максимальное значение (5.5) зависит от размера распределяемого остатка, и если оставшееся количество ресур­са обозначить через ξ, то величину (5.5) можно выразить как функцию от ξ:



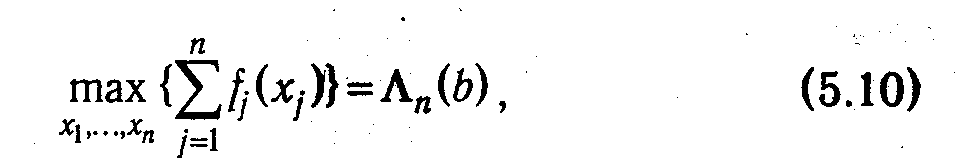
где индекс *п*-1 указывает на оставшееся количество шагов. Тогда суммарный доход, получаемый как следствие решения, принятого на первом шаге, и оптимальных решений, принятых на остальных шагах, будет



Если бы имелась возможность влиять на *xn* , то мы для получе­ния максимальной прибыли должны были бы максимизировать Ω*n* по переменной *xn* , т. е. найти Λ*n*(*b*) и фактически решить задачу:

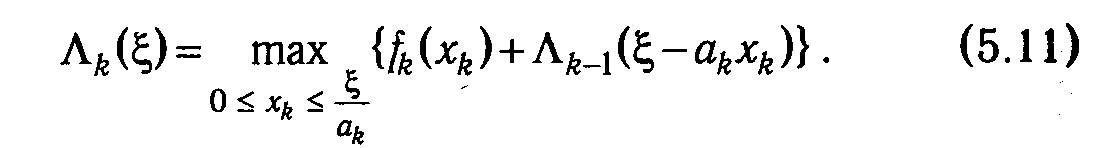


В результате мы получаем выражение для значения целевой функции задачи при оптимальном поэтапном процессе приня­тия решений о распределении ресурса. Оно в силу построения данного процесса равно глобальному оптимуму целевой фун­кции



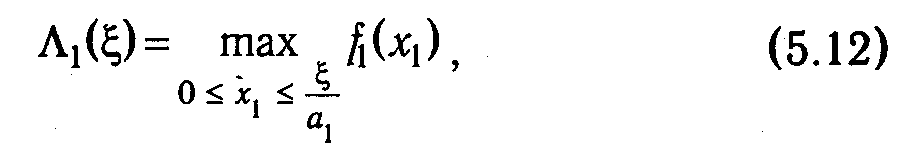
т. е. значению целевой функции при одномоментном распреде­лении ресурса.

Если в выражении (5.9) заменить значения *b* на ξ, и *п* на *k*, то его можно рассматривать как *рекуррентную формулу*, позво­ляющую последовательно вычислять оптимальные значения це­левой функции при распределении объема ресурса ξ за *k* шагов:



Значение переменной *xk* , при котором достигается рассмат­риваемый максимум, обозначим *k* (ξ).

При *k* = 1 формула (5.11) принимает вид



т. е. допускает непосредственное вычисление функций Λ1(ξ) и **1(ξ).

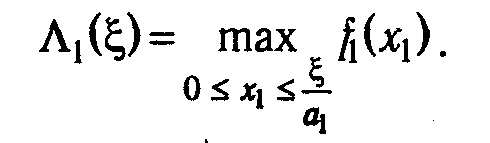
Воспользовавшись (5.12) как базой рекурсии, можно с помо­щью (5.11) последовательно вычислить Λ*k*(ξ) и *k*(ξ), *k*∊2:*n*. Положив на последнем шаге ξ = *b*, в силу (5.9), найдем глобаль­ный максимум функции (5.3), равный Λ*n*(*b*), и компоненту опти­мального плана *хn\** = *n*(*b*). Полученная компонента позволяет вычислить нераспределенный остаток на следующем шаге при оптимальном планировании: ξ= *b* – *аnх\*n* , и, в свою очередь, най­ти *х\*n-*1= *n-*1(ξ*n-*1). В результате подобных вычислений последо­вательно будут найдены все компоненты оптимального плана.

Таким образом, динамическое программирование представ­ляет собой целенаправленный перебор вариантов, который при­водит к нахождению глобального максимума. Уравнение (5.11), выражающее оптимальное решение на *k*-м шаге через решения, принятые на предыдущих шагах, называется *основным рекур­рентным соотношением динамического программирова­ния*. В то же время следует заметить, что описанная схема решения при столь общей постановке задачи имеет чисто тео­ретическое значение, так как замыкает вычислительный про­цесс на построение функций Λ*k*(ξ) (*k*∊1:*n*), т. е. сводит исход­ную задачу (5.3)—(5.4) к другой весьма сложной проблеме. Однако при определенных условиях применение рекуррентных соотношений может оказаться весьма плодотворным. В первую очередь это относится к задачам, которые допускают таблич­ное задание функций Λ*k*(ξ).

**5.1.2. Задачи динамического программирования, до­пускающие табличное задание рекуррентных соотноше­ний.** Рассмотрим процесс решения модифицированного вариан­та задачи (5.3)-(5.4), в котором переменные *хj* и параметры *aj* , *b* могут принимать только целочисленные значения, а ограничение (5.4) имеет вид равенства. В рамках предложенной в п. 5.1.1 ин­терпретации о вложении средств в активы данные предпосылки вполне реалистичны и, более того, могут быть даже усилены тре­бованием о кратности значений *хj* , например, 1000 единицам.

Чтобы не усложнять обозначения, условимся операции це­лочисленной арифметики записывать стандартным образом, по­лагая, что промежуточные результаты подвергаются правиль­ному округлению. Так, например, будем считать, что 12/5=2.

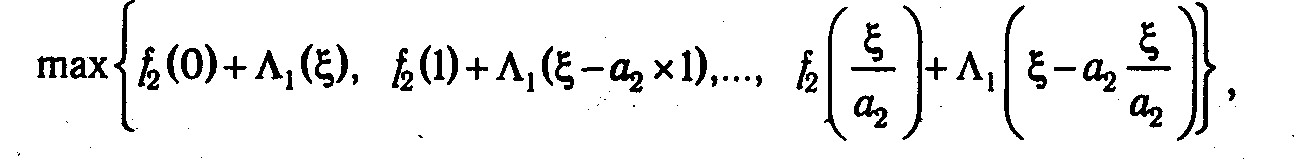
В соответствии с общей схемой вычислительного алгоритма на первом шаге мы должны построить функцию



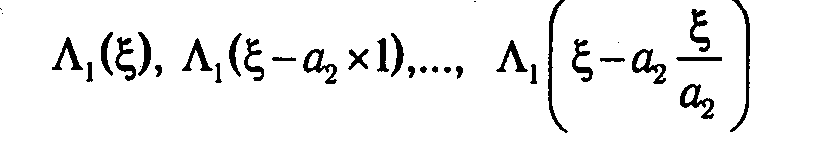
Поскольку ξ≤*b*, *x*1 принимает конечное число целых значений от 0 до *b*/*a*1. Это позволяет, например, путем перебора значе­ний *f*1(*x*1) найти функцию Λ1(ξ) и задать ее в форме таблицы следующей структуры (табл. 5.1).

Последняя колонка табл. 5.1 (**1(ξ)) содержит значение *x*1 , на котором достигается оптимальное решение первого шага. Его необходимо запоминать для того, чтобы к последнему шагу иметь значения всех компонент оптимального плана.

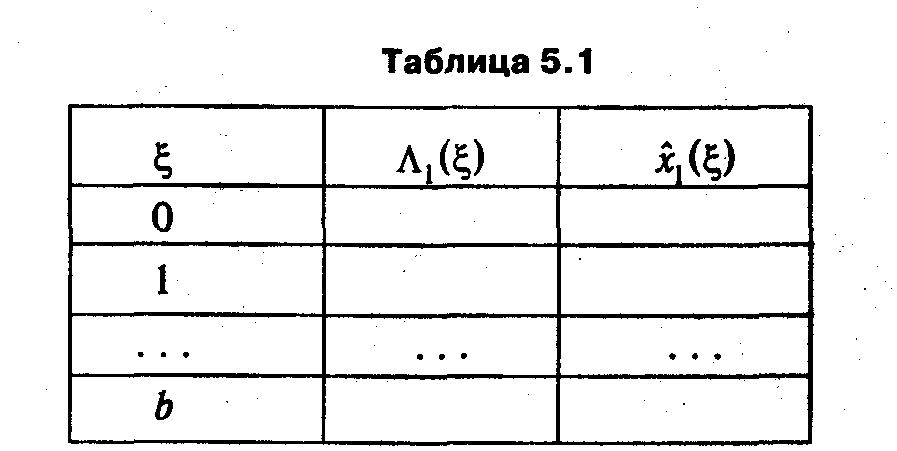
На следующем (втором) шаге можно приступить к вычисле­нию функции Λ2(ξ), значения которой для каждого отдельно взятого ξ∊ 0: *b* находятся как



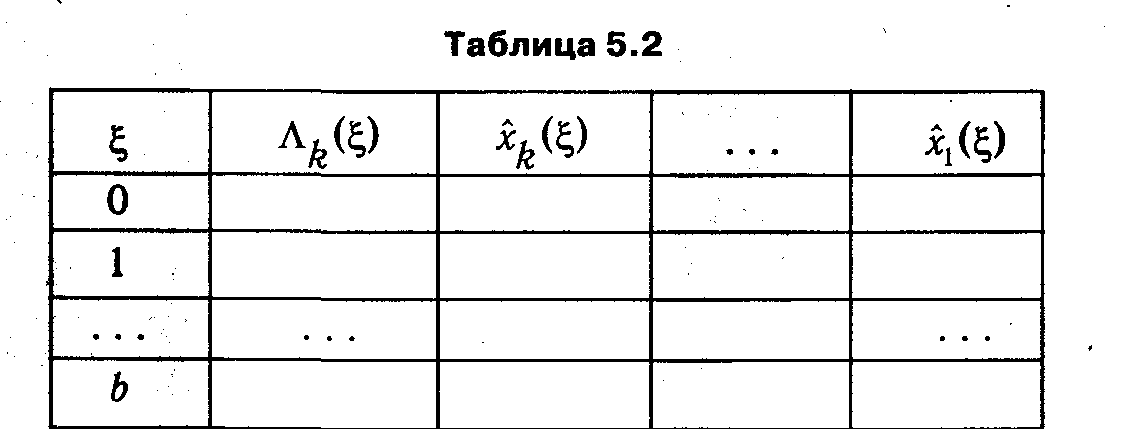
где значения



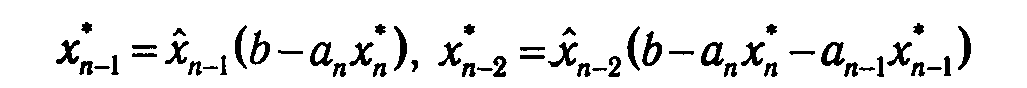
берутся из табл. 5.1. В результате вычислений формируется таблица значений Λ2(ξ), содержащая на одну колонку больше по сравнению с табл. 5.1, так как теперь необходимо запомнить оптимальные решения первого (**1(ξ)) и второго шагов (**2(ξ)).



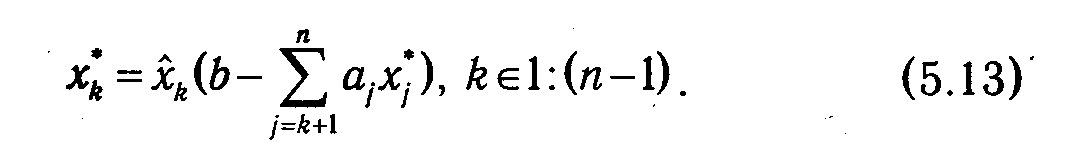
На последующих шагах с номером *k* (*k*∊2:(n-l)) осущест­вляются аналогичные действия, результатом которых стано­вятся таблицы значений Λ*k*(ξ), где ξ ∊{0, 1,..., *b*} (см. табл. 5.2).



На последнем *n*-ом шаге нет необходимости табулиро­вать функцию Λ*n*(ξ), так как достаточно определить лишь Λ*n*(*b*) = *f*(*n*(*b*)). Одновременно определяется и оптимальное значение *n*-й компоненты оптимального плана *x\*n* = *n*(*b*). Далее, используя таблицу, сформированную на предыдущем шаге, определяем оптимальные значения остальных перемен­ных:



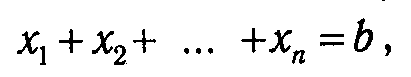
и т. д. или, в общем виде,



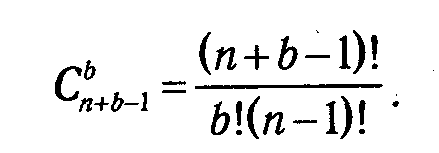
Подчеркнем одно из преимуществ описанного метода с точ­ки зрения его практической реализации в рамках программного обеспечения для ЭВМ: на каждом шаге алгоритма непосред­ственно используется только таблица, полученная на предыду­щем шаге, т. е. нет необходимости сохранять ранее полученные таблицы. Последнее позволяет существенно экономить ресур­сы компьютера.

Выигрыш, который дает применение рассмотренного алго­ритма, может также быть оценен с помощью следующего про­стого примера. Сравним приблизительно по числу операций (состоящих, в основном, из вычислений целевой функции) опи­санный метод с прямым перебором допустимых планов задачи (5.3)-(5.4) при *а*1 = *a*2 = ...*аn* = 1.

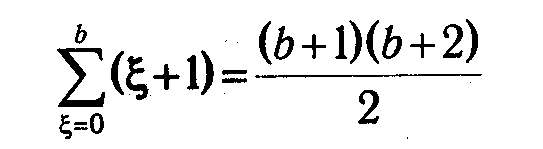
Количество допустимых планов такой задачи совпадает с ко­личеством целочисленных решений уравнения



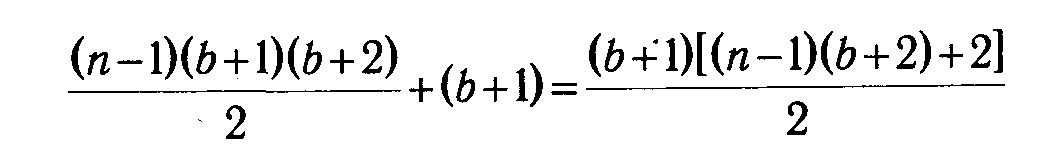
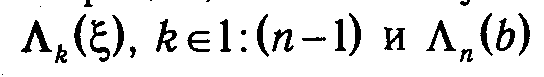
т. е. равно числу сочетаний с повторениями из *n* элементов по *b*. Следовательно, при простом переборе число возможных вари­антов составит



В случае применения метода динамического программирова­ния для вычисления таблицы значений функции Λ*k*(ξ) при фик­сированном ξ необходимо выполнить (ξ+1) операций. Поэто­му для заполнения одной таблицы необходимо проделать



операций, из чего получаем, что для вычисления всех функции потребуется



операций, что при больших *n*  и *b* существенно меньше, чем в первом случае. Например, если *п* = 6 и *b* = 30, то непосредствен­ный перебор потребует выполнения 324 632 операций, а метод динамического программирования — только 2511.

**5.1.3. Принцип оптимальности Беллмана.** Еще раз под­черкнем, что смысл подхода, реализуемого в динамическом программировании, заключен в *замене решения исходной мно­гомерной задачи последовательностью задач меньшей раз­мерности.*

Перечислим основные требования к задачам, выполнение ко­торых позволяет применить данный подход:

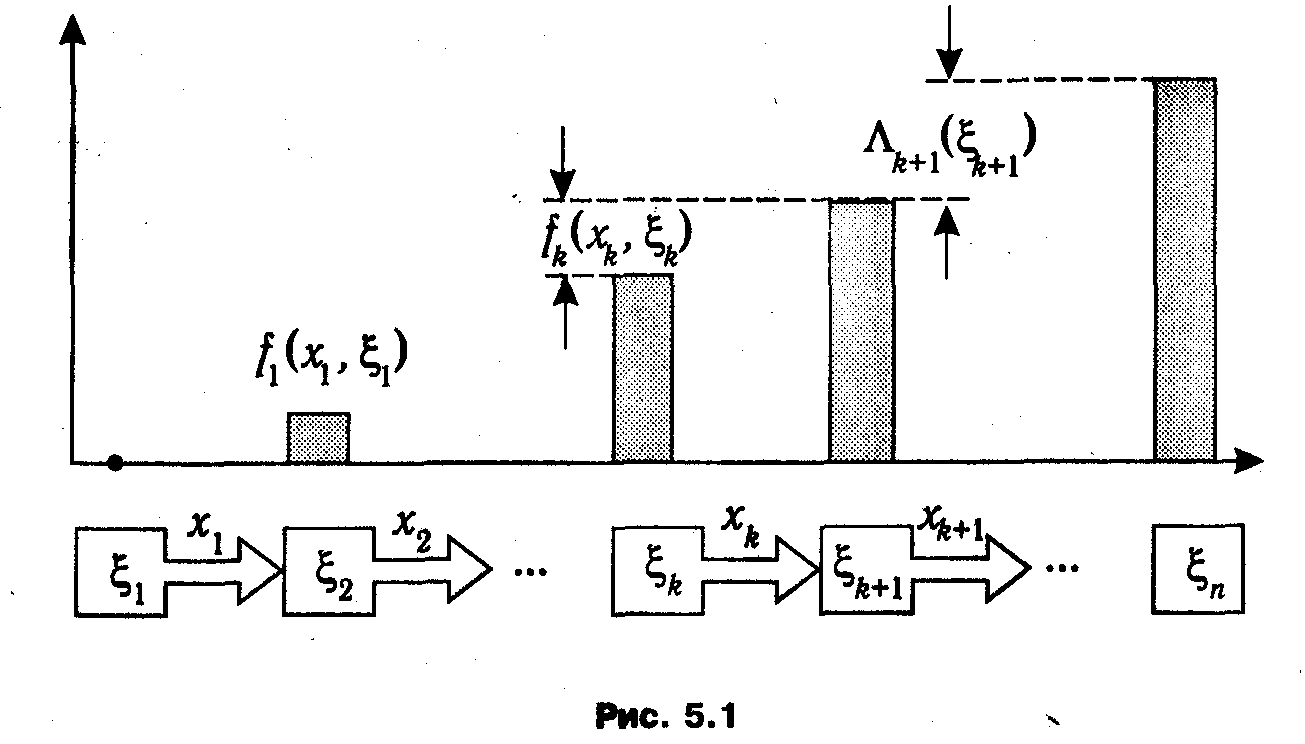
* объектом исследования должна служить *управляемая сис­тема* (объект) с заданными допустимыми *состояниями* и допустимыми *управлениями*;
* задача должна позволять интерпретацию как многошаговый процесс, каждый шаг которого состоит из принятия *реше­ния* о выборе одного из допустимых управлений, приводя­щих к *изменению состояния* системы;

* задача не должна зависеть от количества шагов и быть опре­деленной на каждом из них;
* состояние системы на каждом шаге должно описываться оди­наковым (по составу) набором параметров;
* последующее состояние, в котором оказывается система пос­ле выбора решения на *k*-м. шаге, зависит только от данного решения и исходного состояния к началу *k*-го шага. Данное свойство является основным с точки зрения идеологии дина­мического программирования и называется *отсутствием последействия.*

Рассмотрим вопросы применения модели динамического про­граммирования в обобщенном виде. Пусть стоит задача управ­ления некоторым абстрактным *объектом*, который может пре­бывать в различных *состояниях*. Текущее состояние объекта отождествляется с некоторым набором параметров, обознача­емым в дальнейшем ξ и именуемый *вектором состояния*. Пред­полагается, что задано множество Ξ всех возможных состоя­ний. Для объекта определено также множество *допустимых управлений* (управляющих воздействий) *X*, которое, не умаляя общности, можно считать числовым множеством. Управляю­щие воздействия могут осуществляться в дискретные моменты времени *k*(*k*∊1:*n*), причем управленческое *решение* заключа­ется в выборе одного из управлений *xk*∊*Х*. *Планом* задачи или *стратегией управления* называется вектор *х* = (*х*1, *х*2, .., *xn*-1), компонентами которого служат управления, выбранные на каждом шаге процесса. Ввиду предполагаемого *отсутствия последействия* между каждыми двумя последователь­ными состояниями объекта ξ*k* и ξ*k*+1 существует известная функциональная зависимость, включающая также выбранное управление: ξ*k*+1 = φ*k*(*xk* , ξ*k*), *k*∊1:*п*-1. Тем самым задание на­чального состояния объекта ξ1∊Ξ и выбор плана *х* однозначно определяют *траекторию поведения* объекта, как это показа­но на *рис. 5.1.*

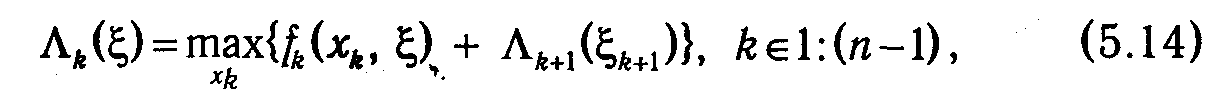
Эффективность управления на каждом шаге *k* зависит от текущего состояния ξ*k* , выбранного управления *xk* и количе­ственно оценивается с помощью функций *fk*(*хk*, ξ*k*), являющих­ся слагаемыми *аддитивной целевой функции*, характеризую­щей общую эффективность управления объектом. (Отметим, что в определение функции *fk*(*хk*, ξ*k*) включается область до­пустимых значений *хk* , и эта область, как правило, зависит от текущего состояния ξ*k* .) *Оптимальное управление*, при задан­ном начальном состоянии ξ1 , сводится к выбору такого опти­мального плана *х\**, при котором достигается *максимум суммы* значений *fk* на соответствующей траектории.

Основной принцип динамического программирования заклю­чается в том, что на каждом шаге следует стремиться не к изолированной оптимизации функции *fk*(*хk*, ξ*k*), а выбирать



оптимальное управление *хk\** в предположении об оптимально­сти всех последующих шагов. Формально указанный принцип реализуется путем отыскания на каждом шаге *k условных оптимальных управлений* *k*(ξ), ξ∊Ξ, обеспечивающих наи­большую суммарную эффективность начиная с этого шага, в предположении, что текущим является состояние ξ.

Обозначим Λ*k*(ξ) максимальное значение суммы функций *fk* на протяжении шагов от *k* до *п* (получаемое при оптимальном управлении на данном отрезке процесса), при условии, что объект в начале шага *k* находится в состоянии ξ . Тогда функции Λ*k*(ξ) должны удовлетворять рекуррентному соотношению:



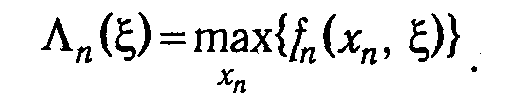
где ξ*k*+1 = φ*k*(*xk*, ξ)

Соотношение (5.14) называют *основным рекуррентным со­отношением* динамического программирования. Оно реализу­ет базовый принцип динамического программирования, извест­ный также как *принцип оптимальности Беллмана*:

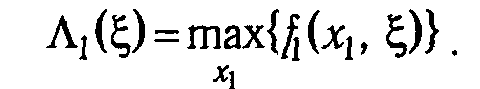
☞ Оптимальная стратегия управления должна удовлетворять следующему условию: *каково бы ни было*

*начальное состо­яние* ξ*k на k-м шаге и выбранное на этом шаге управле­ние хk,, последующие управления (управленческие реше­ния) должны быть оптимальными по отношению к состоянию* ξ*k*+1 = φ*k*(*xk*, ξ*k*)*, получающемуся в результа­те решения, принятого на шаге k.*

Основное соотношение (5.14) позволяет найти функции Λ*k*(ξ) только в сочетании с *начальным условием*, каковым в нашем случае является равенство



Сравнение рекуррентной формулы (5.14) с аналогичными соотношениями в рассмотренных выше примерах указывает на их внешнее различие. Это различие обусловлено тем, что в за­даче распределения ресурсов фиксированным является конечное состояние управляемого процесса. Поэтому принцип Бел­лмана применяется не к последующим, а к начальным этапам управления, и начальное соотношение имеет вид



Важно еще раз подчеркнуть, что сформулированный выше принцип оптимальности применим только для управления объектами, у которых выбор оптимального управления не зави­сит от предыстории управляемого процесса, т. е. от того, каким путем система пришла в текущее состояние. Именно это обсто­ятельство позволяет осуществить декомпозицию задачи и сде­лать возможным ее практическое решение.

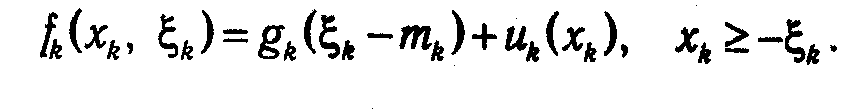
В то же время, говоря о динамическом программировании как о методе решения оптимизационных задач, необходимо отметить и его слабые стороны. Так, в предложенной схеме решения зада­чи (5.3)-(5.4) существенным образом используется тот факт, что система ограничений содержит только одно неравенство, и, как следствие, ее состояние задается одним числом — нераспре­деленным ресурсом ξ . При наличии нескольких ограничений со­стояние управляемого объекта на каждом шаге характеризуется уже набором параметров ξ1, ξ2, ..., ξ*m* , и табулировать значения функций Λ*k* (ξ1, ξ2, ..., ξ*m*) необходимо для многократно больше­го количества точек. Последнее обстоятельство делает приме­нение метода динамического программирования явно нерацио­нальным или даже просто невозможным. Данную проблему его основоположник Р. Беллман эффектно назвал «проклятием многомерности». В настоящее время разработаны определенные пути преодоления указанных трудностей. Подробную информа­цию о них можно найти в специальной литературе [4, 5].

## 5.2. ПРИМЕРЫ ЗАДАЧ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

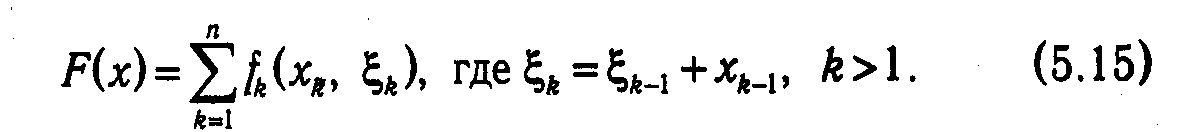
**5.2.1. Задача о найме работников.** Приступим к рассмот­рению вопросов применения методов динамического програм­мирования в конкретных экономико-математических моделях. Отдельно отметим, что данные вычислительные схемы, вообще говоря, достаточно часто используются для решения некото­рых задач, которые уже были затронуты в других главах. Это, прежде всего, задача о ранце, задача о кратчайшем пути, задачи транспортного типа.

Одним из важнейших классов задач, для которых примене­ние динамического программирования оказывается плодотвор­ным, являются задачи последовательного принятия решений. Их особенностью является то, что искомые переменные *х*1, *x*2, .., *хk* ,... должны определяться в строгой временной по­следовательности и не должны меняться местами. В качестве примера опишем так называемую *задачу о найме работников* (задачу об использовании рабочей силы).

В данной задаче рассматривается некоторый экономиче­ский объект (фирма, магазин, завод и т. п.), функционирую­щий в течение конечного числа периодов, обозначаемых но­мерами *k* (*k* ∊ l:*n*). Каждый период *k* характеризуется нормативной потребностью в определенном количестве одно­типных работников *mk*. Тот же объем работ может быть вы­полнен другим количеством сотрудников ξ*k*, что, однако, влечет дополнительные затраты либо за счет нерационально­го использования рабочей силы, либо ввиду повышения опла­ты за интенсивный труд. Размеры этих дополнительных издер­жек описываются функциями *gk* (ξ*k* - *mk*), где (ξ*k* - *mk*) — отклонение фактической численности работающих ξ*k* , от пла­ново необходимой *mk* , причем *gk* (0)=0. Управленческое ре­шение на шаге *k* заключается в выборе величины *изменения* числа сотрудников *хk*∊*Z*, что однозначно определяет количе­ство работающих в течение следующего периода: ξ*k*+1 = ξ*k*+*хk*. Затраты по изменению количества работников (найму и уволь­нению) при переходе от периода *k* к периоду (*k*+1) задаются функцией *uk* (*хk*) , где также *uk* (0)=0. Тогда суммарные издер­жки, вызванные принятым на шаге *k* решением, характеризу­ются значением функции

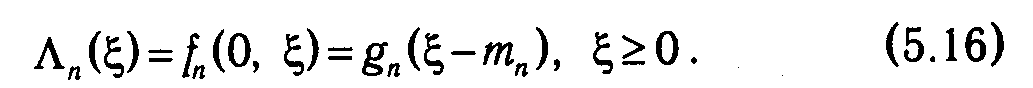


План задачи (стратегия управления) *х* = (*x*1, ..., *хn*-1, 0) за­ключается в выборе поэтапных изменений количества работни­ков, а его суммарная эффективность описывается аддитивной функцией

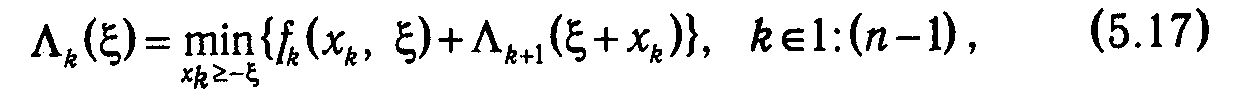


На основе сформулированной модели ставится задача мини­мизации целевой функции (издержек) (5.15). Добавим, что по­становка задачи не будет корректной, если не задать *начальное условие* на количество работников. Существуют две модифика­ции данной задачи, определяемые типом начального условия: в первом случае задается исходное значение на первом этапе *m*1, а во втором — требуемое количество в *n*-м периоде *mn* .

Рассмотрим первый случай. Поскольку фиксированным яв­ляется начальное количество работников и, напротив, ничего не известно о том, каким это количество должно быть на послед­нем этапе, то рассмотрение процесса принятия решений удоб­нее начать с конца. Оптимальное управление на последнем эта­пе *п* по условию равно *х\*n* = *n*(ξ)=0, поэтому минимальные издержки полностью определяются количеством работников в последнем периоде:

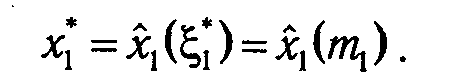


Для остальных предшествующих шагов основное рекуррент­ное соотношение примет вид

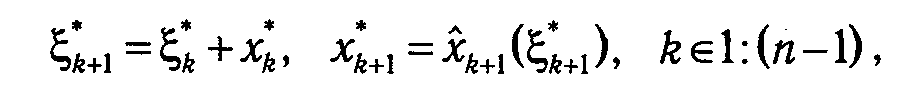


где Λ*k*(ξ) — минимальные затраты с *k*-го по *п*-й периоды, в пред­положении, что количество работников в *k*-й период равно ξ. Точки *л*(ξ), в которых достигаются минимумы (5.17), опреде­ляют условное оптимальное управление на каждом шаге.

Последовательно определяя *л*(ξ) и дойдя до этапа 1, мы смо­жем найти безусловное оптимальное управление *x*1\* из того условия, что на начало первого периода численность работни­ков должна составлять ξ1\* = *m*1 , a именно

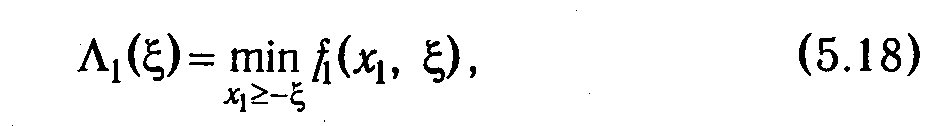


Остальные компоненты оптимального плана *хk\** и состояния ξ*k\**, образующие оптимальную траекторию, последовательно находятся по рекуррентным формулам

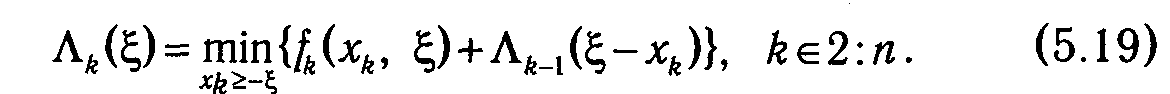


после чего не составляет труда вычислить оптимальное значе­ние целевой функции (5.15).

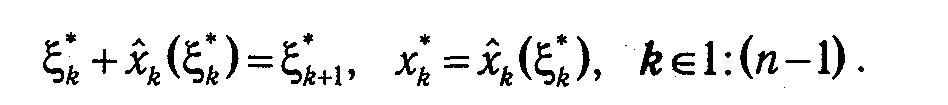
Остановимся теперь на *втором случае*, когда задано фи­нальное состояние управляемого объекта, т. е. желаемое коли­чество работников на последнем периоде ξ*n\**= *mn* . Очевидно, что в данной ситуации следует поступить с точностью «до наобо­рот» и рассмотреть процесс принятия решений от начала к кон­цу. Наилучшее условное управление на первом шаге **1(ξ) будет найдено в процессе вычисления функции



где состояние ξ ≥0 является возможным количеством работ­ников на начальном шаге. Соответственно, основное рекуррент­ное соотношение выразит минимальные издержки вплоть до *k*-го периода через таковые для предыдущих периодов (с перво­го по (*k*-1)-й) при условии, что численность работников в *k*-й период будет равна ξ:



Попутно будут найдены функции *k*(ξ), *k*∊2:*n*, определяю­щие условные оптимальные управления. На последнем перио­де, в силу начального условия, ξ*n\**= *mn*. Отсюда путем последо­вательного решения рекуррентных уравнений могут быть найдены оптимальные численности работников ξ*\*k* и безуслов­ные оптимальные управления:



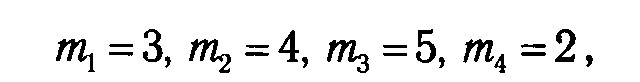
В заключение, как и в первом случае, подсчитывается мини­мальная величина издержек.

Обобщая изложенные схемы решения, можно прийти к вы­воду:

*При использовании алгоритмов динамического програм­мирования, если задано начальное состояние управляемой системы, то задача решается* ***в* обратном направлении**, *а если конечное, тo* — **в прямом.** *Наконец, если заданы как начальное, так и конечное состояния, то задача суще­ственно усложняется.* (В качестве компромисса в этом слу­чае можно отказаться от оптимизации на первом или последнем этапе.)

Продемонстрируем процесс решения задачи о найме работ­ников на конкретном примере:

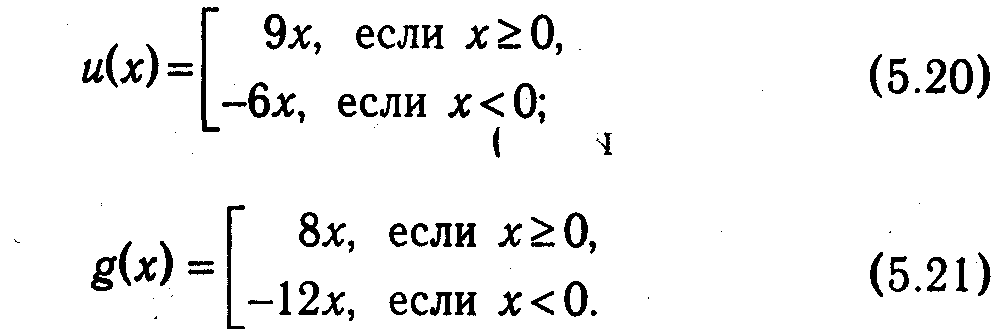
Для функционирования некоторого предприятия в течение четырех месяцев (нумеруемых от 1 до 4) по нормам требуются следующие количества работников одинаковой квалификации:



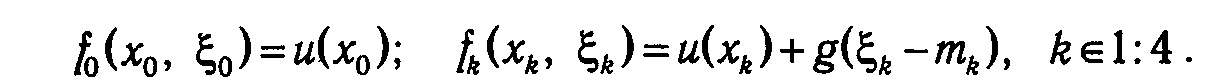
причем перед началом первого месяца (в нулевом месяце) фак­тически имеется ξ0 = 2 сотрудников. Администрация планирует в конце каждого месяца *k* (кроме последнего) корректировать число работающих на величину *xk* , *k*∊0:4, *х*4 = 0. На прием одного сотрудника необходимо затратить 9 у. е., а на увольне­ние — 6 у. е. Предполагается, что расходы на содержание избы­точного работника составляют 8 у. е., а в случае нехватки персо­нала приходится нести затраты в размере 12 у. е. за каждое вакантное место.

Требуется найти оптимальные значения приращений чис­ленности работающих в конце каждого из первых трех меся­цев, при которых суммарные издержки за весь рассматривае­мый период будут минимальными.

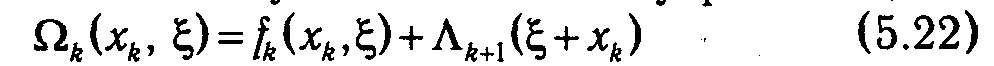
В начале решения запишем в аналитической форме функции издержек на прием-увольнение сотрудников (*и*), а также на содержание ненормативного штата (*g*). С этой целью введем функции



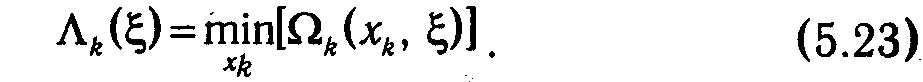
Оценки эффективности управления на каждом шаге имеют вид:



Поскольку в поставленной задаче задано начальное условие ξ\*0 = 2, ее решение начинается с конца, и, следовательно, будут применяться рекуррентные соотношения (5.17). С технической точки зрения будет удобно на каждом шаге составлять две табли­цы значений: функции издержек, получаемых начиная с текуще­го шага в зависимости от текущего состояния и управления,



и функции минимальных издержек в зависимости от текущего состояния



Для сокращения объема табулируемых значений можно вос­пользоваться свойством выпуклости функции Ω*k* (*xk* , ξ), выте­кающим из выпуклости *f* и *g*. Из выпуклости функции Ω*k* (*xk* , ξ) следует, что заполнять таблицу ее значений необходимо лишь до тех пор, пока они уменьшаются, т. е. можно остановиться, как только очередное значение оказывается больше предыду­щего. Отметим, что подобные приемы очень широко использу­ются в динамическом программировании. Разумеется, иллюст­рируемые методы не рассчитаны на ручной счет, поскольку связаны с очень большим объемом рутинных вычислений. Ради краткости ниже приведены только фрагменты таблиц, содержа­щие интересующие нас значения.

*Итерация 1.* Полагаем *k*=4. На данном этапе функция со­стояния Λ4(ξ) может быть найдена непосредственно, если учесть, что *x*4\*=0 и *u*(0)=0:

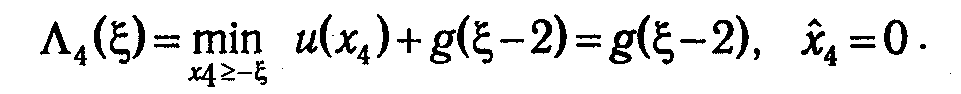
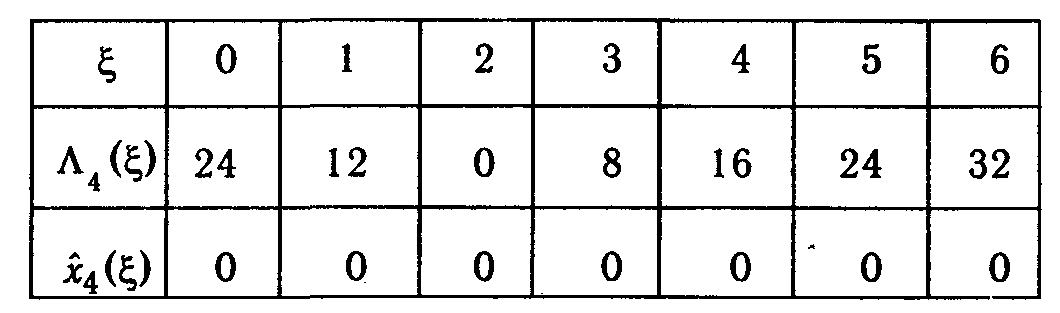
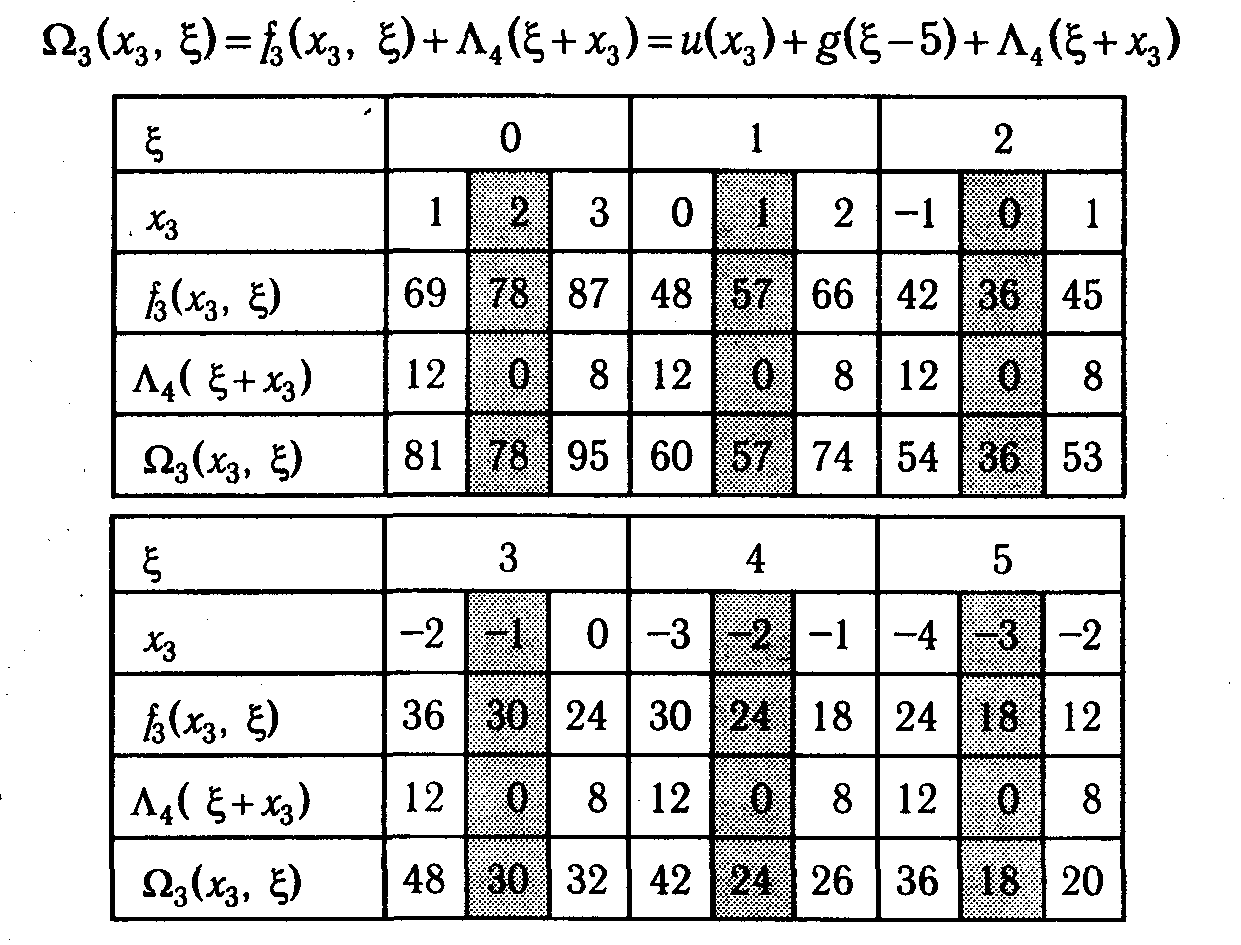


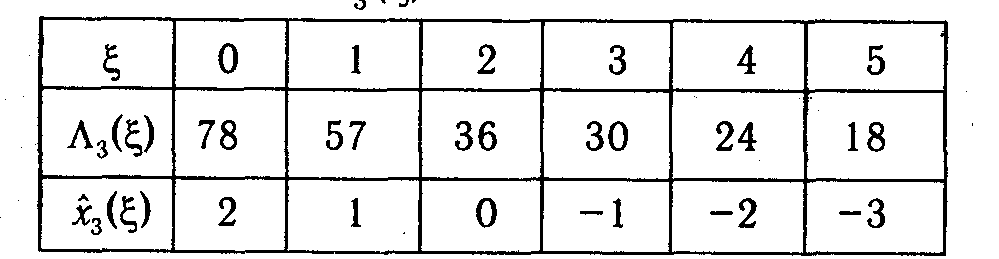
Таблица значений данной функции и условные оптимальные управления имеют вид



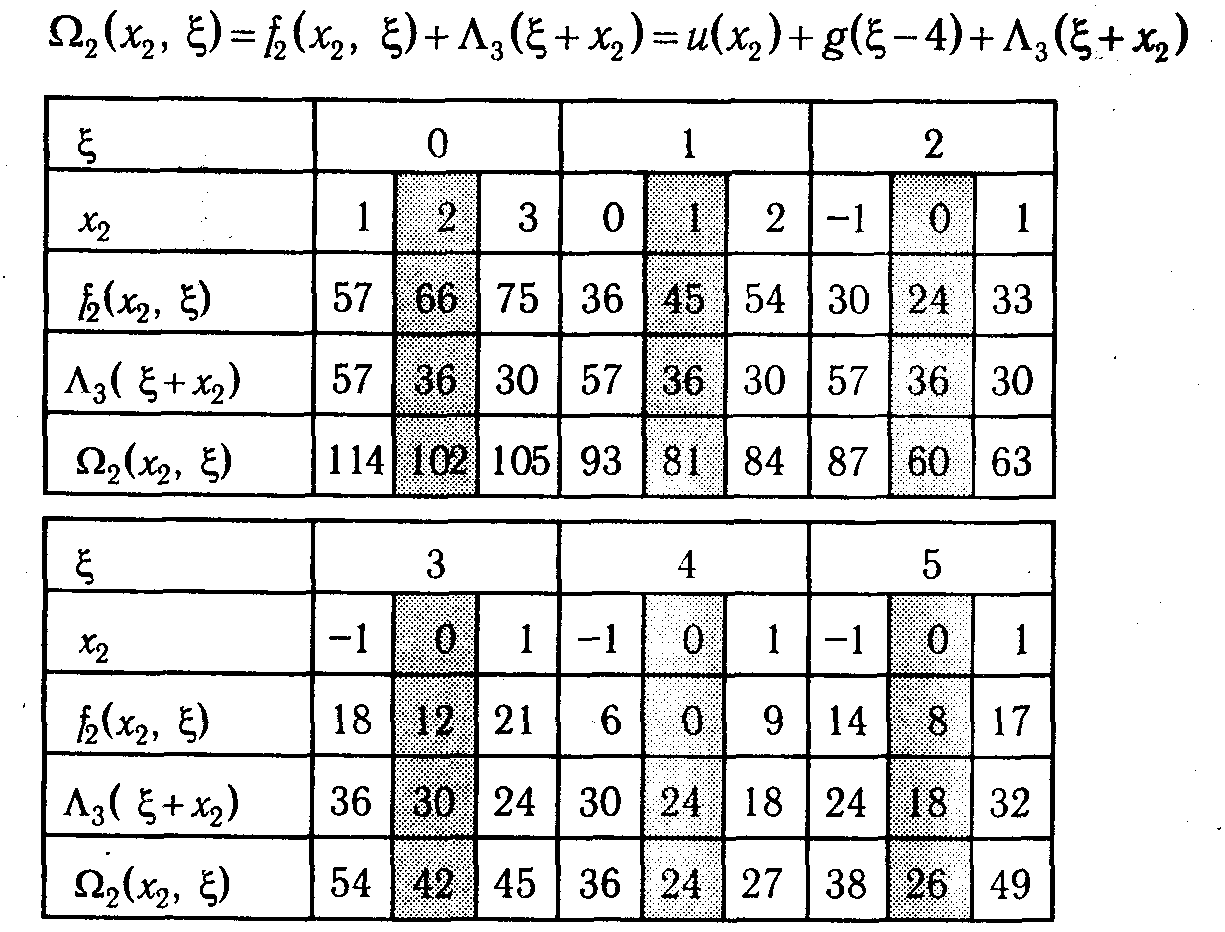
*Итерация* 2. Полагаем *k*=3. Предварительно заполним таб­лицу значений функции Ω3(*x*3, ξ) для достаточно большого множества аргументов согласно формуле:



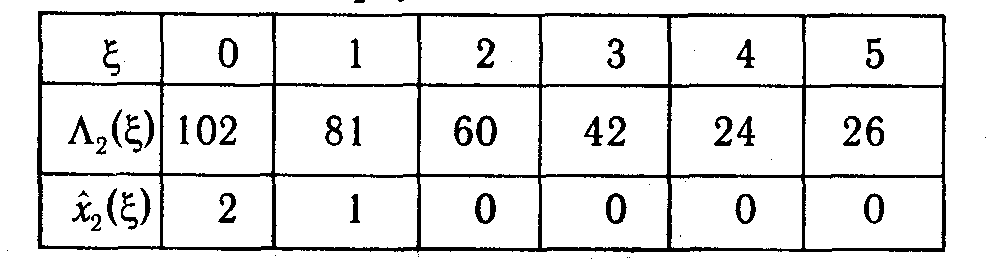
Выбирая минимальные по *х*3 значения Ω3(*x*3, ξ) составим таблицу Λ3(ξ) и соответствующие значения условных опти­мальных управлений **3(ξ):



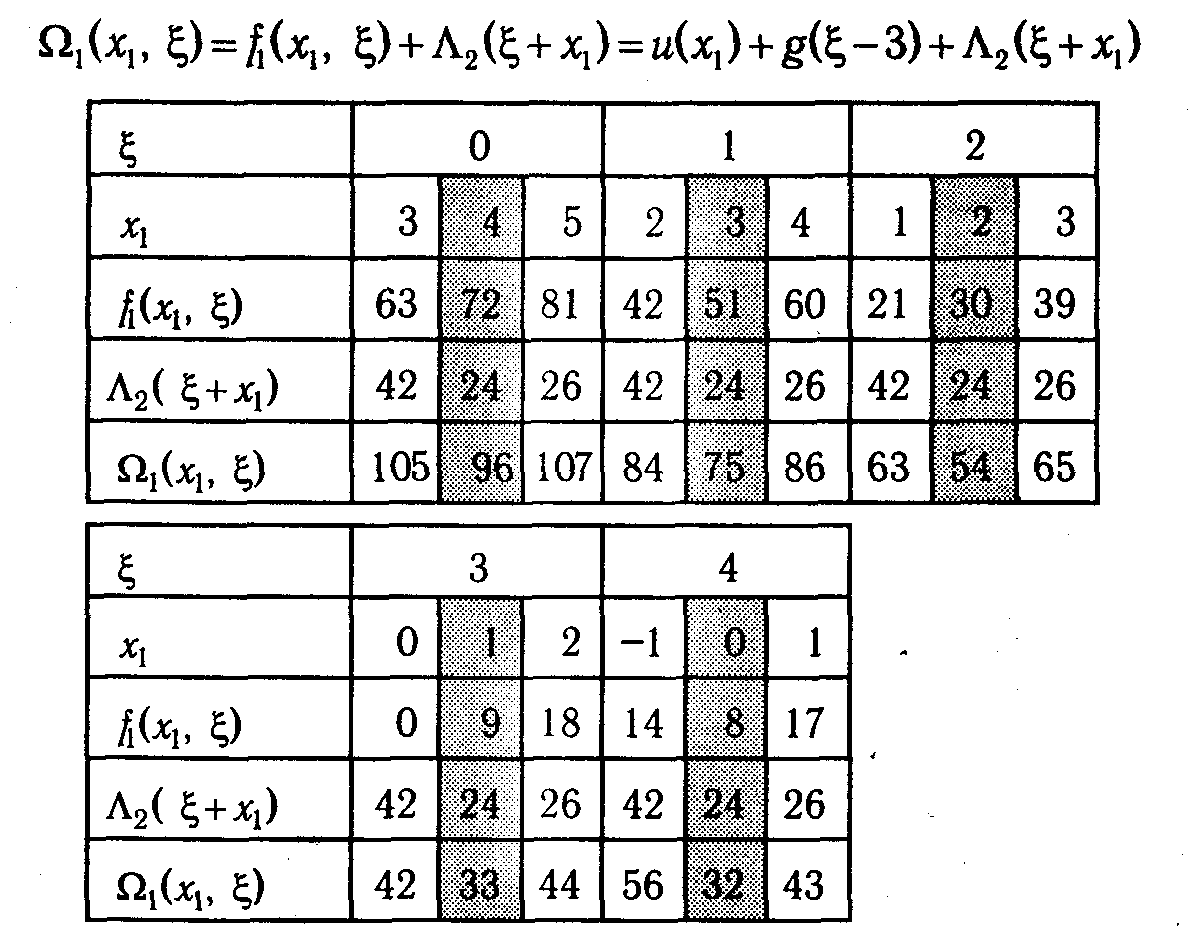
*Итерация 3*. Полагаем *k*=2. Так же, как на предыдущей итерации, заполним таблицу значений функции Ω2(*x*2, ξ) со­гласно формуле:



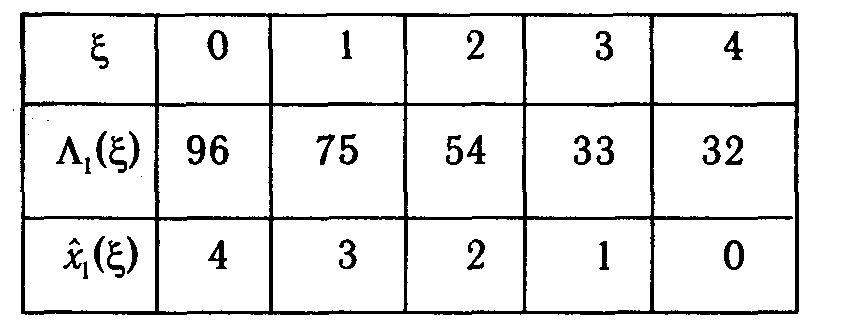
Выбирая минимальные по *х*2 значения Ω2(*x*2 , ξ), составим таблицу Λ1(ξ) и соответствующие значения условных опти­мальных управлений **2(ξ):



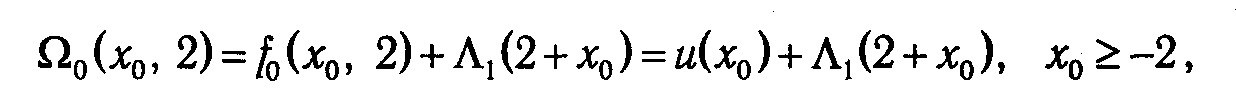
*Итерация 4.* Полагаем *k*=1. Аналогично предыдущему, за­полним таблицу значений функции Ω1(*x*1, ξ) согласно формуле:



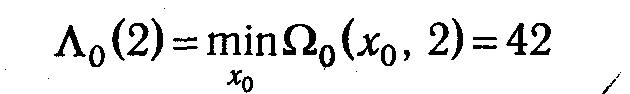
Выбирая минимальные по *х*1, значения Ω1(*x*1, ξ), составим таблицу Λ1(ξ) и соответствующие значения условных опти­мальных управлений **1(ξ):



*Итерация 5.* На последней итерации, в связи с наличием на­чального условия ξ\*0 = 2, достаточно вычислить

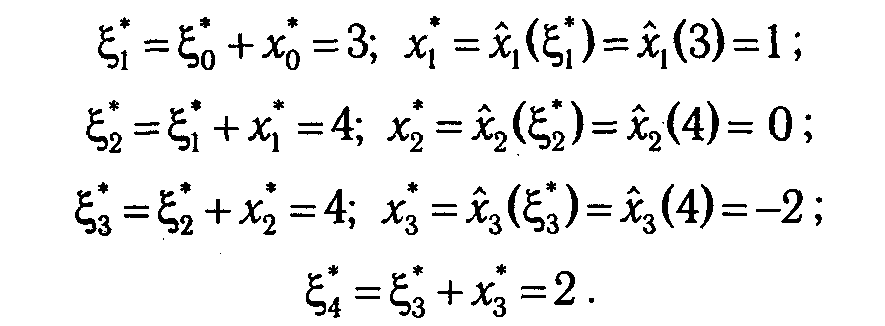


и найти **0(2) как точку минимума Ω0(*x*0, 2) . Простые вычисле­ния показывают, что минимум



достигается при *x*0(2) = 1.

Следовательно, *x\**0 = **0(2)=1, после чего обратным ходом последовательно вычисляются оптимальные управления и оп­тимальные состояния (оптимальная траектория):



Итак, результаты расчета свидетельствуют, что при задан­ной системе расценок в третьем месяце выгоднее не брать 5-го работника, а компенсировать его отсутствие дополнительными выплатами за сверхурочную работу имеющимся сотрудникам.

**5.2.2. Динамические задачи управления запасами.** Одной из наиболее известных сфер приложения методов дина­мического программирования является такая область матема­тической экономики, как *теория управления запасами.* Ее предметом является разработка и исследование математиче­ских моделей систем, занимающих промежуточное положение между источниками (производителями) тех или иных ресурсов и их потребителями. При математической формализации про­цессов управления запасами очень часто приходится исполь­зовать скачкообразные, недифференцируемые и кусочно-не­прерывные функции. Как правило, это обусловливается необ­ходимостью учета эффектов концентрации, фиксированных затрат и платы за заказ. В связи с этим получаемые задачи с трудом поддаются аналитическому решению классическими методами, однако могут быть успешно решены с помощью аппа­рата динамического программирования. Рассмотрим достаточ­но типичную задачу, возникающую в процессе планирования деятельности системы снабжения, — так называемую *динами­ческую задачу управления запасами.*

Пусть имеется некоторая система снабжения (склад, опто­вая база и т. п.), планирующая свою работу на *п* периодов. Ее деятельность сводится к обеспечению спроса конечных потре­бителей на некоторый продукт, для чего она осуществляет за­казы производителю данного продукта. Спрос клиентов (конеч­ных потребителей) в данной модели рассматривается как некоторая интегрированная величина, принимающая заданные значения для каждого из периодов, и он должен всегда удов­летворяться (т. е. не допускаются задолженности и отказы). Также предполагается, что заказ, посылаемый производителю, удовлетворяется им полностью, и временем между заказом и его выполнением можно пренебречь (т. е. рассматривается сис­тема с мгновенным выполнением заказа). Введем обозначения:

*yk* — остаток запаса после (*k*-1)-го периода;

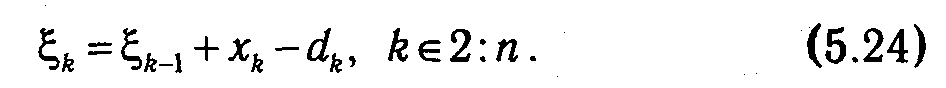
*dk* — заранее известный суммарный спрос в *k*-м периоде;

*хk* — заказ (поставка от производителя) в *k*-м периоде;

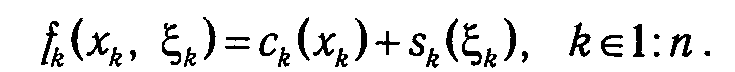
*сk* (*хk*) —затраты на выполнение заказа объема *xk* в *k*-м пери­оде;

*sk* (ξ*k*) — затраты на хранение запаса объема ξ*k* в *k*-м пери­оде.

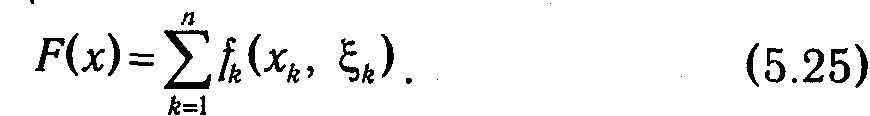
После получения поставки и удовлетворения спроса объем товара, подлежащего хранению в период *k*, составит ξ*k* = *yk* + *хk* - *dk* . Учитывая смысл параметра *yk* , можно записать соот­ношение:



Расходы на получение и хранение товара в период *k* описыва­ются функцией



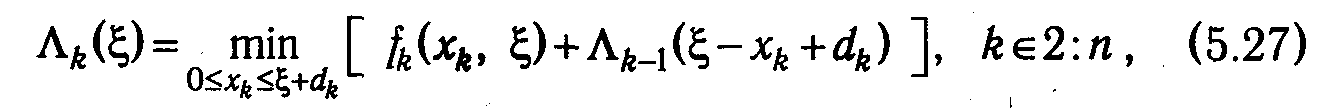
Планом задачи можно считать вектор *х* = (*х*1, *х*2, ..., *хn*), ком­понентами которого являются последовательные заказы в течение рассматриваемого промежутка времени. Соотношение между запасами (5.24) в сочетании с некоторым начальным условием связывает состояния системы с выбранным планом и позволяет выразить суммарные расходы за все *п* периодов функционирования управляемой системы снабжения в форме аддитивной целевой функции:



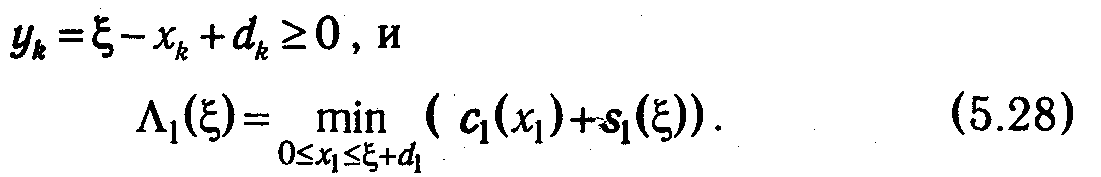
Естественной в рамках сформулированной модели представля­ется задача нахождения последовательности оптимальных управлений (заказов) *x\*k* и связанных с ними оптимальных состояний (запасов) ξ\**k* , которые обращают в минимум (5.25). В качестве *начального условия* используем требование о со­хранении после завершения управления *заданного количест­ва товара* *yn*+1 , а именно



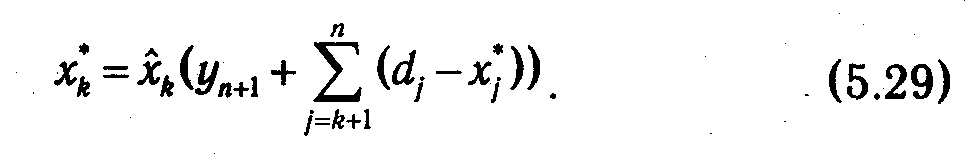
При решении поставленной задачи методом динамического про­граммирования в качестве функции состояния управляемой си­стемы Λ*k*(ξ) логично взять минимальный объем затрат, возни­кающих за первые *k* периодов при условии, что в *k*-й период имеется запас ξ . Тогда можно записать основное рекуррентное соотношение



поскольку

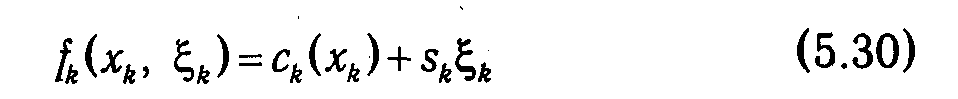


Система рекуррентных соотношений (5.27)-(5.28) позво­ляет найти последовательность функций состояния Λ1(ξ), Λ2(ξ), …, Λ*n*(ξ) и условных оптимальных управлений **1(ξ), **2(ξ), …, *n*(ξ). На *n*-м шаге с помощью начального условия (5.26) можно определить *х\*n* = *n* (*yn*+1). Остальные значения оп­тимальных управлений *x\*k* определяются по формуле:



Особый интерес представляет частный случай задачи (5.24)-(5.25), при котором предполагается, что функции за­трат на пополнение запаса *сk*(*хk*) являются вогнутыми по *хk* , а функции затрат на хранение *sk*(ξ*k*) являются линейными отно­сительно объема хранимого запаса, т. е. *sk*(ξ*k*) = *sk*ξ*k* . Парал­лельно заметим, что обе предпосылки являются достаточно ре­алистичными.

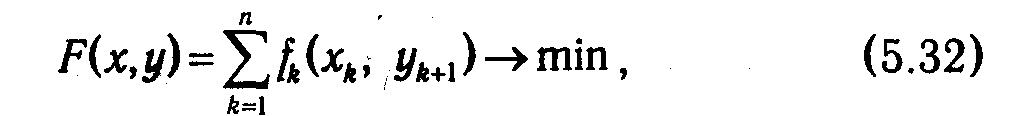
Обозначим функцию затрат в течение *k*-ro периода через



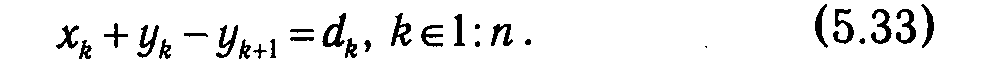
или, что то же самое,



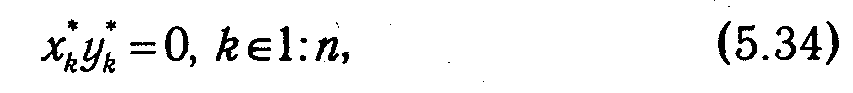
В силу сделанных предположений все функции затрат *fk* (*xk* , *yk*+1) являются вогнутыми (как суммы вогнутой и линей­ной функций). Данное свойство значительно упрощает процесс решения, так как для поиска минимума вогнутых функций *fk* (*xk* , *yk*+1) достаточно рассмотреть только две крайние точки множества, на котором отыскивается минимум. С учетом вве­денного обозначения задачу (5.24)-(5.25) можно записать в виде:



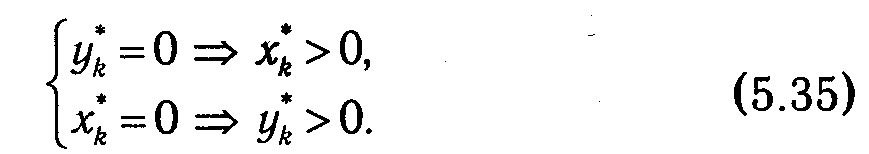
при условиях



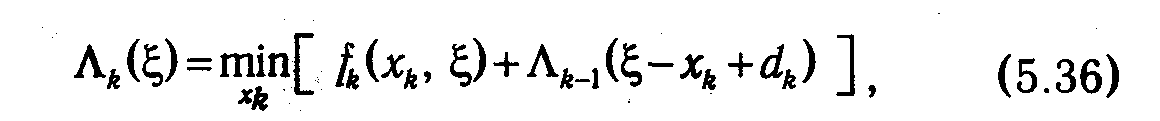
Рассмотрим процедуру решения (5.32)-(5.33). Так как ищет­ся минимум суммы вогнутых функций *fk*(*xk* , *yk*+1), то решение будет достигаться на одной из крайних точек множества, опре­деляемого условиями (5.33). Общее число переменных *xk* и *yk* в системе (5.33) равно *2п*. Однако, учитывая то, что в ней толь­ко *п* уравнений, в оптимальном плане будет не более *п* ненуле­вых компонент, причем для каждого периода *k* значения *xk* и *yk* не могут равняться нулю одновременно (в силу необходимости удовлетворения спроса либо за счет заказа, либо за счет запа­са). Формально это утверждение можно представить в виде ус­ловия дополняющей нежесткости:



где

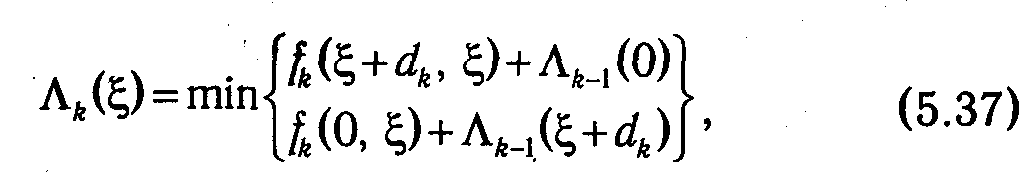


С точки зрения содержательной интерпретации условия (5.34)-(5.35) означают, что при *оптимальном управлении* за­каз поставщику на новую партию не должен поступать, если в начале периода имеется ненулевой запас, или *размер заказа должен равняться величине спроса за целое число периодов*. Отсюда следует, что запас на конец последнего периода должен равняться нулю: *у\*n*+1=0. Последнее позволяет решать задачу в прямом направлении, применяя рекуррентное соотношение

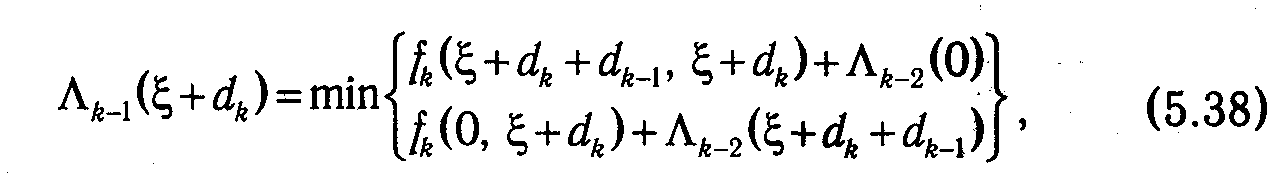


где ξ *= уk*+1*= хk + уk - dk .*

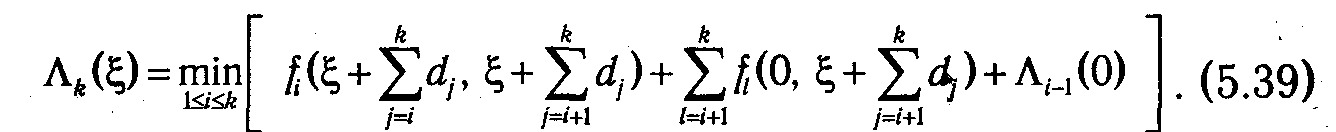
Учитывая (5.34)-(5.35) и вогнутость *f****k*** (*x****k***, ξ), заключаем, что минимум (5.36) достигается в одной из крайних точек *x****k*** =0 или *x****k*** = ξ + *d****k*** поэтому



тогда для предыдущего периода функция состояния может быть выражена как



на oснове чего в общем виде получаем модифицированную фор­му для рекуррентного соотношения



При дальнейших конкретизирующих предположениях о ви­де функций *fk* (*xk* , *уk*+1) можно получить еще более компактные формы для рекуррентных соотношений. Однако эти вопросы носят достаточно частный характер, и мы их рассматривать не будем. Отметим лишь, что приведенные в данном пункте преоб­разования неплохо иллюстрируют общие подходы, применяе­мые в динамическом программировании, а также те свойства задач, которые открывают возможности, для эффективного и плодотворного использования соответствующих методов.

## КЛЮЧЕВЫЕ ПОНЯТИЯ

* Аддитивная и мультипликативная функция.
* Рекуррентное соотношения.
* Принцип оптимальности Беллмана.
* Отсутствие последействия.
* Задача о найме работников.
* Динамическая задача управления запасами.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

5.1. Для решения каких задач предназначен метод динамичес­кого программирования?

5.2. В чем заключена суть метода динамического программи­рования?

5.3. Каким условиям должна удовлетворять задача, чтобы для ее решения мог быть применен

алгоритм динамического программирования?

5.4. Какие трудности связаны с вычислительными алгоритма­ми динамического программирования?

5.5. Что определяет направление решения задачи в алгорит­мах динамического программирования?

5.6. Сформулируйте математическую модель для задачи о най­ме работников.

5.7. Выпишите основное рекуррентное соотношение, исполь­зуемое при решении задачи о найме

работников.

5.8. С какими особенностями задач управления запасами свя­зано применение при их решении

аппарата динамического программирования?

5.9. Какой вид имеет целевая функция в динамической задаче управления запасами?

5.10. Выпишите основное рекуррентное соотношение, исполь­зуемое при решении динамической

задачи управления за­пасами.

# ГЛАВА 6. КРАТКИЙ ОБЗОР

# ДРУГИХ РАЗДЕЛОВ ИССЛЕДОВАНИЯ ОПЕРАЦИЙ

Данная глава занимает особое положение среди остальных. Ее цель — представить читателю предельно краткую характерис­тику так называемых дополнительных разделов исследования операций. Поясним несколько смысл слова «дополнительный». Дело в том, что существуют различные подходы к классифика­ции этих научных отраслей: иногда их считают составными час­тями исследования операций, а иногда рассматривают как само­стоятельные дисциплины. Обе точки зрения имеют право на существование, но даже если придерживаться второй, разделы, о которых пойдет речь в настоящей главе, можно трактовать как смежные области знания, аппарат которых применим для решения задач исследования операций. Еще раз подчеркнем, что каждый параграф настоящей главы лишь обзорно затраги­вает общие вопросы соответствующих тем, а полные учебные курсы по ним существенно превышают объем данной книги.

## 6.1. ТЕОРИЯ ИГР

**6.1.1. Предмет теории игры.** Задачи, рассмотренные в пре­дыдущих главах, формулировались для ситуаций индивидуально­го выбора оптимальных решений, т.е. для случаев, когда решение принимает отдельно взятый субъект, обладающий единственной целью.

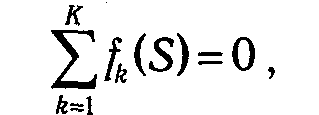
Принципиально иная ситуация возникает при изучении процессов принятия решений несколькими субъектами, инте­ресы которых могут не совпадать. При этом возникают задачи со многими целевыми функциями (критериями). Область мате­матики, изучающая данные проблемы, получила название **тео­рии игр**. Задачи теории игр относятся к области принятия ре­шений в условиях неопределенности, а их специфика состоит в том, что, как правило, подразумевается неопределенность, воз­никающая в результате действий двух или более «разумных» противников, способных оптимизировать свое поведение за счет других. Среди типичных примеров такого поведения могут быть названы действия конкурирующих фирм на одном рынке или планирование военных операций.

Одним из основных вопросов в задачах с коллективным вы­бором решений является *вопрос об определении оптималь­ности*, т. е. вопрос, какие решения следует признавать наилуч­шими в ситуации оптимизации по нескольким критериям, отражающим различные интересы. Многие методы решения проблем теории игр основываются на сведении их к задачам ма­тематического программирования. На наиболее простых из них мы остановимся в настоящей главе.

Теория игр берет начало от работ Э. Бореля (1921 г.), а прин­ципиальным этапом в ее становлении как самостоятельного на­учного направления стала монография Дж. Неймана, вышед­шая в 1944 г. [25].

**6.1.2. Терминология и классификация игр.** Особенно­стью теории игр как научной дисциплины стала употребляемая в ней специфическая терминология. Термин «*игра*» применяется для обозначения совокупности правил и соглашений, которыми руководствуются субъекты, поведение которых мы изучаем. Каждый такой субъект *k*, где *k*∊l:*K*, или *игрок*, характеризует­ся наличием индивидуальной системы целевых установок и *стра­тегий* *s*1*k*, *s*2*k*, ..., *smkk*, т. е. возможных вариантов действий в игре.

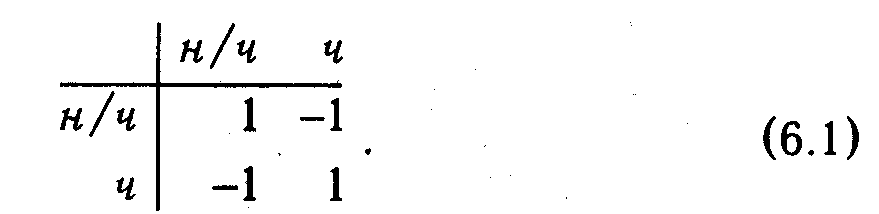
Достаточно распространенный способ математического описания игры основан на задании функций *fk* (*s*1*i*1 , *s*2*i*2 , ..., *skik* , ..., *sKik*), каждая из которых определяет результат (платеж, выигрыш), получаемый *k*-м игроком в зависимости от набора стратегий *S* =(*s*1*i*1 , *s*2*i*2 , ..., *skik* , ..., *sKik*), примененного всеми участ­никами игры. Функции *fk* , *k*∊l:*K* также *называют функциями выигрыша*, или *платежными функциями*. В том случае, если для любых *S*



игра называется *игрой с нулевой суммой*. Игру с двумя участ­никами и нулевой суммой называют *антагонистической*. Ан­тагонистические игры, т. е. игры, в которых выигрыш одного участника равен проигрышу другого, в силу относительно про­стой постановки задачи являются наиболее изученным разде­лом теории игр. Однако содержание теории игр, безусловно, не исчерпывается ими. В классификации игровых моделей вы­деляют игры с конечными и бесконечными наборами стратегий у игроков, выделяют игры по возможным количествам ходов у участников. Также игры делят на *некооперативные* и *коопе­ративные*, т. е. те, в которых функции выигрыша участников зависят от образуемых ими коалиций. Помимо этого игры мож­но различать по объему информации, имеющейся у игроков от­носительно прошлых ходов. В этой связи они делятся на игры с *полной и неполной информацией*. Заинтересованный читатель может обратиться к таким источникам, как [17, 23].

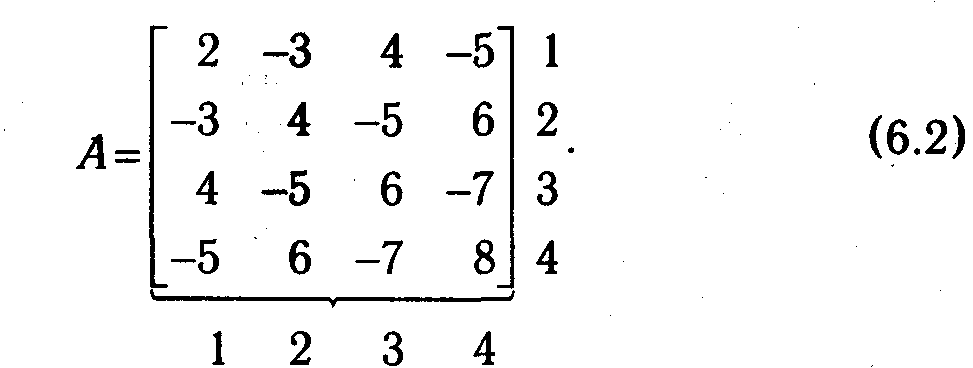
**6.1.3. Матричные игры и понятие седловой точки.** Рас­смотрим более подробно антагонистические игры и их основ­ные свойства. Удобным способом задания игры двух участников с нулевой суммой является *платежная матрица*. Отсюда, кстати, происходит еще одно их название — *матричные игры*. Каждый элемент платежной матрицы *аij* содержит числовое значение *выигрыша игрока* I (проигрыша игрока II), если пер­вый применяет стратегию *i*, а второй — стратегию *j*. Термины *выигрыш* и *проигрыш* следует понимать в широком смысле, т. к. они могут принимать отрицательные значения и с житейской точки зрения означать противоположное. Нетривиальность за­дачи прежде всего заключается в том, что каждый из игроков делает свой выбор, не зная о выборе другого, что существенно осложняет процесс оптимизации выбираемой стратегии.

Классическим примером антагонистической игры является игра с двумя участниками, загадывающими независимо друг от друга числа. Предполагается, что если их сумма оказывается чет­ной, то выигрыш, равный 1, достается первому игроку, а если не­четной, то второму. Положив, что для обоих игроков загадыва­ние нечетного числа является первой стратегией, а четного — второй, можем записать платежную матрицу данной игры:



Строки матрицы (6.1) соответствуют стратегиям игрока I, столб­цы — стратегиям игрока II, а ее элементы — результатам перво­го игрока. Также из определения игры следует, что элементы данной матрицы, взятые с обратным знаком, соответствуют вы­игрышам второго игрока.

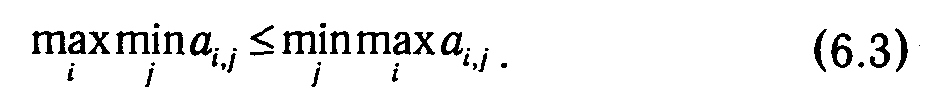
Более сложная и содержательная платежная матрица может быть получена, если несколько модифицировать предложен­ную игру. Допустим, что оба участника имеют право загадывать числа от 1 до 4, что составляет их соответствующие стратегии. В случае, если результат сложения задуманных чисел будет четным, то второй игрок выплачивает первому получившуюся сумму, а если нечетным, то первый — второму. Запишем пла­тежную матрицу для такой игры:



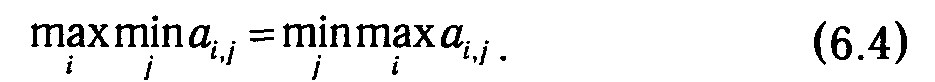
Некоторая условность и искусственность в постановке про­блемы не должны в данном случае нас смущать, так как к подоб­ной форме может быть сведена модель, описывающая, напри­мер, соревнование двух фирм за вновь открывшийся рынок сбыта продукции и т. п.

Как уже отмечалось, важнейшим в теории игр является во­прос об оптимальности решения (выбора стратегии) для каж­дого из игроков. Проанализируем с этой точки зрения некото­рую матричную игру, для которой задана платежная матрица *А*=║*aij*║*m*x*n*. При выборе игроком I стратегии *i* его гарантирован­ный доход независимо от действий игрока II составит min *ai,j*. Поскольку он может выбирать *i* самостоятельно, то целесооб­разно этот выбор сделать таким, чтобы он при любой стратегии противника максимизировал величину гарантированного дохо­да, т. е. обеспечивал получение max (min *ai,j*). Такой принцип вы­бора стратегии получил название «*принцип максимина*». С дру­гой стороны, аналогичные рассуждения могут быть проведены по поводу действий второго игрока. Его наибольший проигрыш при выборе стратегии *j* составит max *ai,j*, и, следовательно, ему следует выбирать стратегию так, чтобы минимизировать вели­чину проигрыша при любых действиях соперника, т. е. обеспе­чить min (max *ai,j*). в этом суть *принципа минимакса.*

Можно доказать справедливость следующего соотношения:

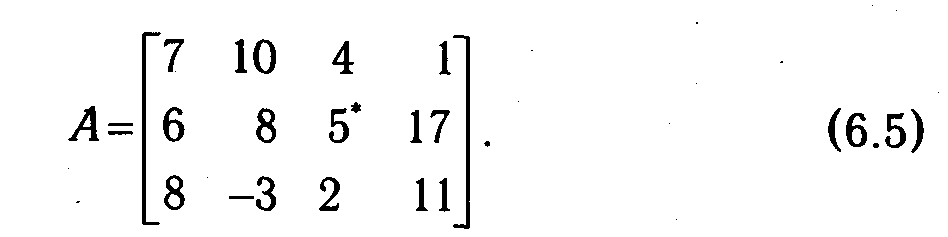


Однако очевидный интерес представляет ситуация, при ко­торой значение выигрыша (платежа), получаемого игроком I при выборе им максиминной стратегии, равно платежу (проиг­рышу) II-го игрока при минимаксной стратегии



В этом случае говорят, что игра имеет *седловую точку*. Совпа­дение значений гарантированных выигрышей игроков при мак­симинной и минимаксной стратегии означает возможность до­стижения в игре некоторого оптимального (стабильного, равновесного) состояния, от которого невыгодно отклоняться ни одному из участников. Понятие «оптимальность» здесь озна­чает, что ни один разумный (осторожный) игрок не стремится изменить свою стратегию, так как его противник, в принципе, сможет выбрать такую стратегию, которая даст худший для первого результат. Стратегии *i\** и *j*\*, образующие седловую точку, называются *оптимальными*, а значение *v* = *ai\*j\** называют *це­ной игры*. Тройка (*i\**, *j\**, *v*) считается *решением матричной игры с седловой точкой.*

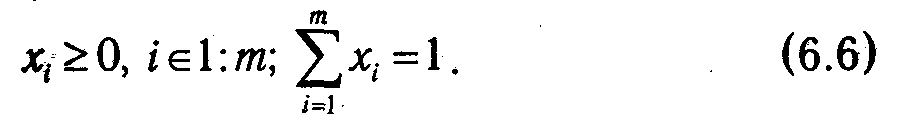
Нетрудно заметить, что не всякая игра обладает седловой точкой. В частности, как игра (6.1), так и игра (6.2) седловой точки не имеют. Примером игры, имеющей седловую точку, яв­ляется игра с платежной матрицей (6.5).



В данной матрице минимальные (гарантированные) выигрыши первого игрока по строкам равны 1, 5 и (-3). Следовательно, его максиминному выбору будет отвечать стратегия 2, гаранти­рующая выигрыш 5. Для второго игрока максимальные проиг­рыши по столбцам матрицы составят 8, 10, 5, 17, поэтому имеет смысл остановиться на стратегии 3, при которой он проиграет только 5. Таким образом, вторая стратегия первого игрока и третья стратегия второго образуют седловую точку со значени­ем 5, т. е. для игры с матрицей (6.5) имеет решение (2; 3; 5).

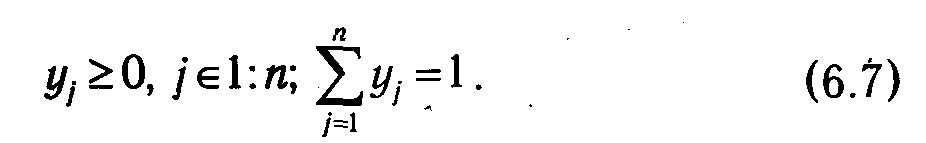
**6.1.4. Смешанные стратегии.** Дальнейшее развитие тео­рии матричных игр основывается на исследовании игры как не­которого повторяющегося процесса. Действительно, вряд ли можно дать содержательные рекомендации по такому вопросу, как следует поступать участникам однократно проводимой игры, не имеющей седловой точки. В случае же ее многократных по­второв естественной и плодотворной представляется идея *ран­домизации* выбора стратегий игроками, т. е. внесение в процесс выбора элемента случайности. Действительно, систематическое отклонение, например, игрока I от максиминной стратегии с це­лью увеличения выигрыша может быть зафиксировано вторым игроком и наказано. В то же время абсолютно хаотичный выбор стратегий не принесет в среднем наилучшего результата.

* **Смешанной стратегией** *игрока I в игре с матрицей A*=║*ai,j*║*m*x*n называется упорядоченный набор действительных чисел xi , i*∊1:*m, удовлетворяющих условиям*



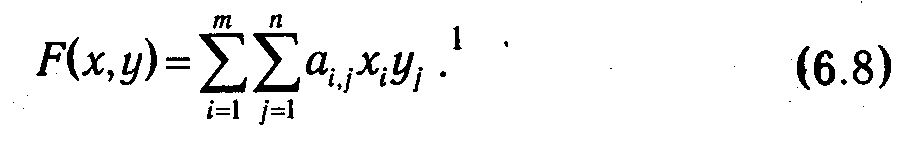
Числа интерпретируются как вероятности применения иг­роком I стратегий 1, 2,..., *m*, которые, в отличие от смешанных, также называют *чистыми стратегиями.*

Аналогично вводится понятие смешанных стратегий игрока II, которые определяются как набор чисел *уj* , *j*∊1:*n* , удовлетво­ряющих условиям



Тогда, если игрок I применяет смешанную стратегию *х* = (*х*1, *х*2,..., *хm*) а игрок II смешанную стратегию *y* = (*y*1, *y*2,..., *yn*), то математическое ожидание выигрыша игрока I (проигрыша игрока II) определяется соотношением

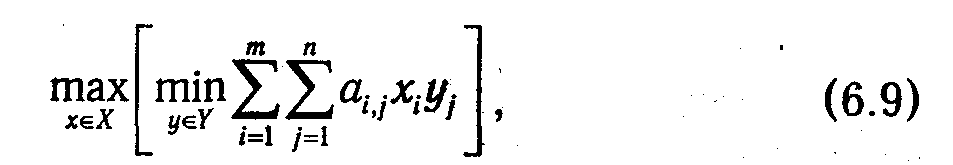
\*



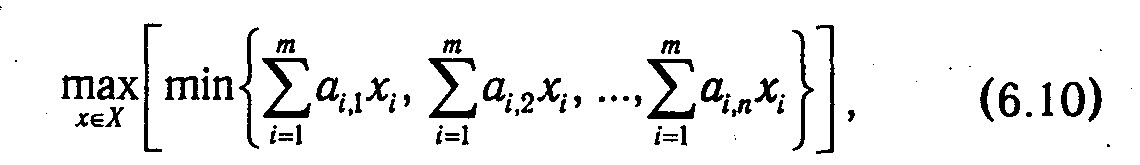
\* Напомним, что при многократном повторении игры средний выиг­рыш близок к математическому ожиданию.

В дальнейшем через *Х* будем обозначать множество допу­стимых смешанных стратегий игрока I, определяемое условием 6.7, а через *Y* — определяемое условием 6.8 множество допу­стимых смешанных стратегий игрока II.

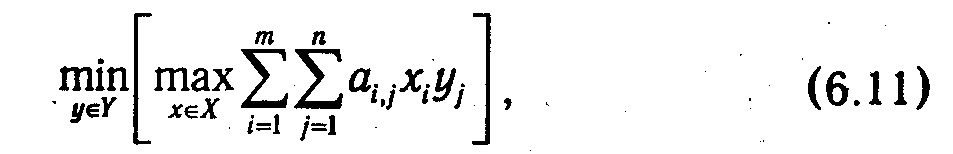
К поиску решения игры в смешанных стратегиях, так же как и в п. 6.1.3, могут быть применены критерии максимина-минимакса. В соответствии с ними игрок I будет выбирать свою сме­шанную стратегию *х* = (*х*1, *х*2,..., *хm*) таким образом, чтобы мак­симизировать наименьший средний выигрыш:



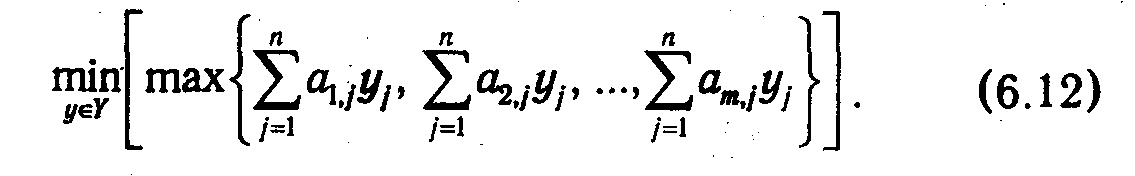
который, как можно доказать, равен



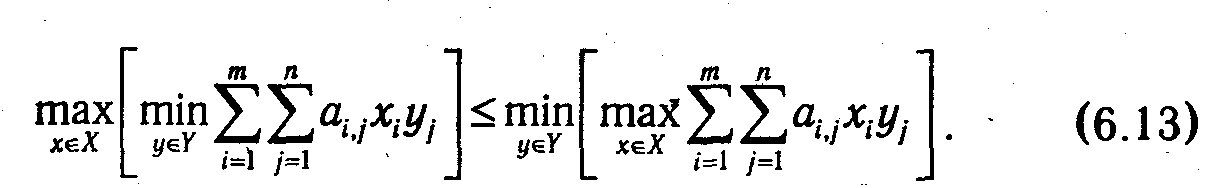
а игрок II — свою смешанную стратегию так, чтобы минимизи­ровать наибольший средний проигрыш:



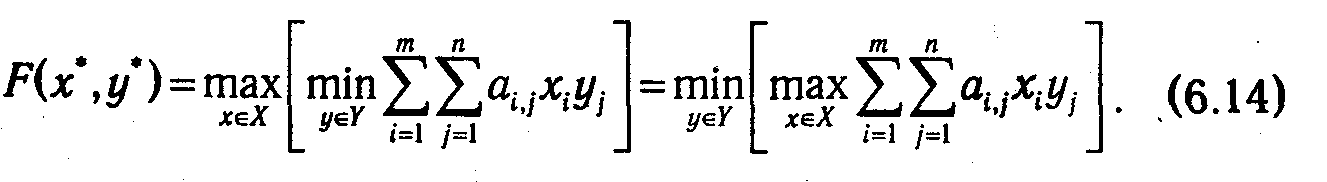
также равный



По аналогии с (6.3) для любых *х*∊*Х* и *y*∊*Y* справедливо не­равенство



* *Стратегии х\**∊*Х* и *y\**∊*Y* *называют* **оптимальными смешанными стратегиями**, *если для любых* *х*∊*Х* и *y*∊*Y* *справедливо равенство*



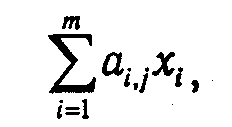
*v =F*(*x\*, у\**) называют ценой игры, и если *х\** и *у\** существу­ют, то говорят, что *игра имеет решение в смешанных стра­тегиях* (*х\*, у\*, v*).

Справедлива фундаментальная теорема Дж. Неймана, кото­рую мы приведем без доказательства.

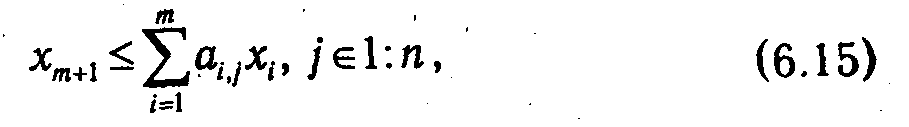
|  |
| --- |
| **Теорема 6.1** (основная теорема матричных игр). *Любая матричная игра имеет решение в смешанных страте­гиях.* |

Значение и нетривиальность теоремы (6.1) обусловлены прежде всего тем, что, как было показано в п. 6.1.3, в общем случае матричные игры в чистых стратегиях решения не имеют.

**6.1.5. Решение матричных игр методами линейного программирования.** Рассмотрим некоторые способы ре­шения матричных игр. Задача, решаемая первым игроком, (6.10) была сформулирована как максимизация наименьшей из сумм



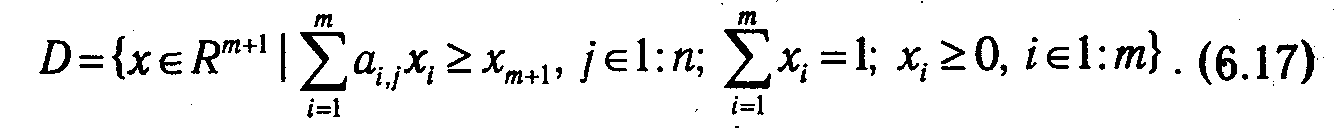
но если определить некоторое *хm*+1, для которого выполняется



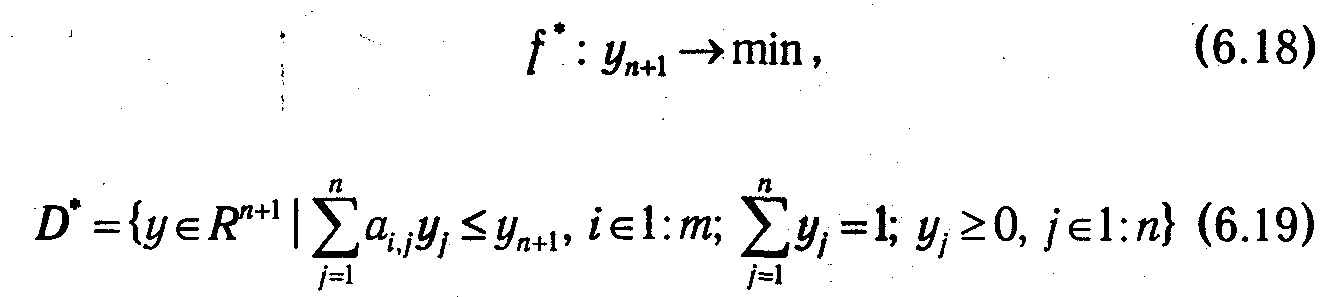
то она может быть сведена к задаче линейного программиро­вания:



при ограничениях



Проведя аналогичные рассуждения, приходим к тому, что задача минимизации наибольшего ожидаемого проигрыша, решаемая игроком II (6.12), сводится к задаче линейного програм­мирования

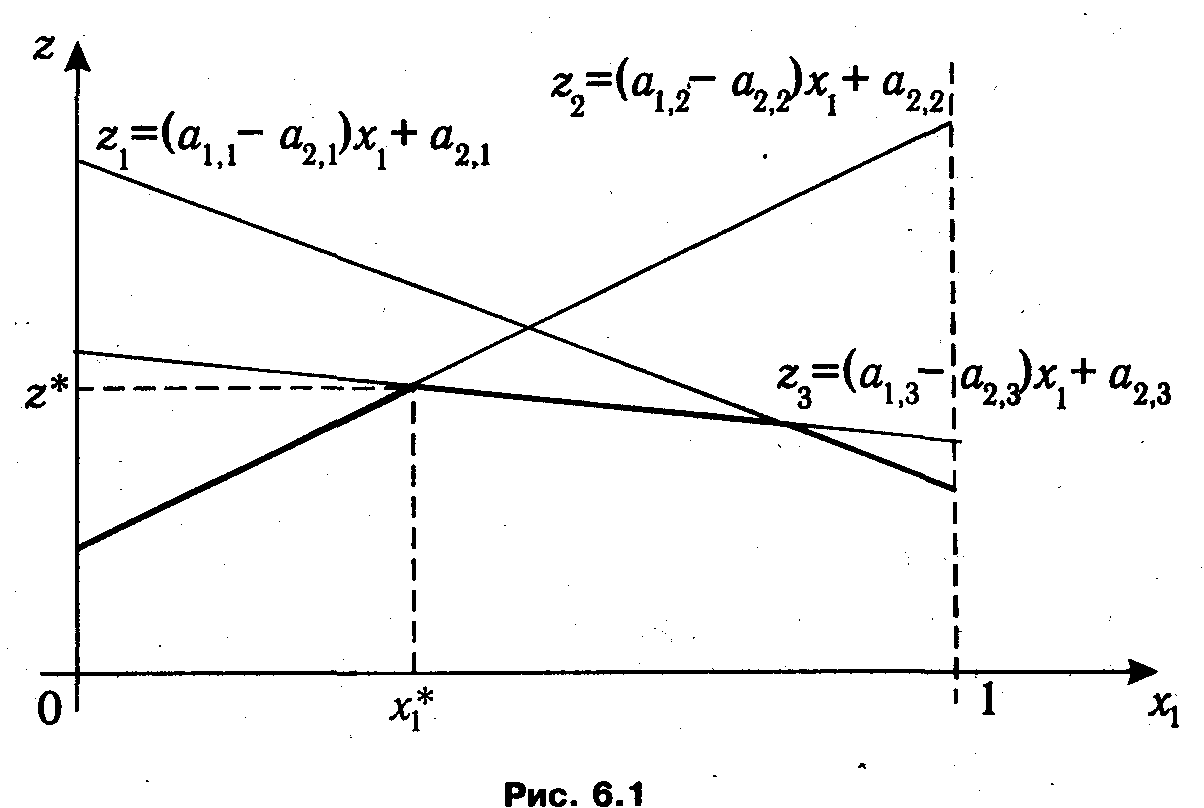
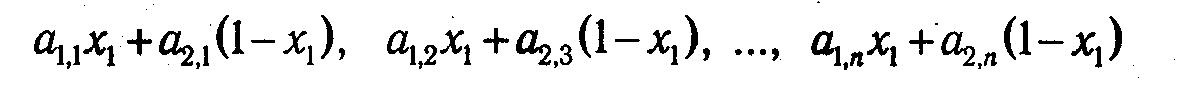


Таким образом, мы получаем возможность применять все возможности аппарата линейного программирования для поис­ка оптимальных стратегий обоих игроков.

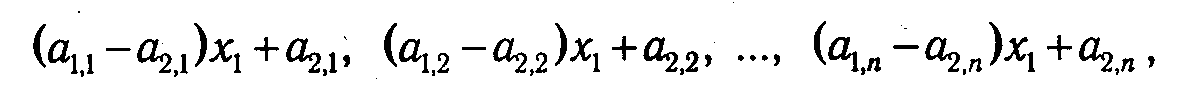
Достаточно легко проверить, что задачи (6.16)-(6.17) и (6.18)-(6.19) образуют двойственную пару. Здесь в определен­ном смысле мы вернулись к проблемам, уже рассматривавшим­ся во второй главе, а именно к взаимосвязи между наличием ре­шения у некоторой оптимизационной задачи и существованием седловой точки у соответствующей функции Лагранжа. В дан­ном случае аналогичная связь прослеживается между седловой точкой игры и решением пары задач оптимизации.

**6.1.6. Графические методы решения игр.** Следует отме­тить, что применение для решения задач (6.16)-(6.17), (6.18)-(6.19) стандартных алгоритмов линейного программирования далеко не всегда является рациональным. Помимо этого суще­ствуют иные методы, которые основываются на использовании специфики данных задач. В настоящем пункте мы остановимся на очень простом классическом способе поиска оптимальных сме­шанных стратегий в матричных играх, где один из участников имеет только две стратегии (это так называемые 2 х *п*  и *т* х 2 игры).

Для определенности положим, что игрок I имеет возмож­ность выбирать между двумя стратегиями с вероятностями *x*1 и *x*2 = 1-*x*1, тогда его ожидаемые выигрыши, соответствующие чистым стратегиям игрока II, примут вид



или

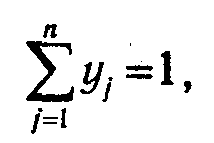


т. е. ожидаемые выигрыши могут быть представлены в виде графи­ков линейных функций, зависящих от переменной *x*1 ∊ [0; 1] (*рис. 6.1*, где предполагается, что игрок II имеет три стратегии).

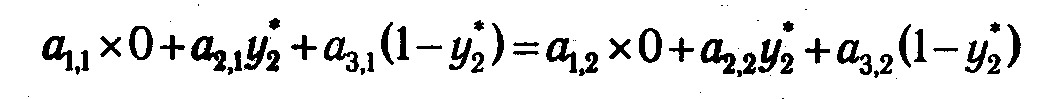
Линии, изображенные на *рис.* *6.1*, задают зависимости сред­него выигрыша игрока I от значения вероятности *x*1 , с которой он выбирает свою первую стратегию, для случаев, когда его противник выбирает первую, вторую или третью чистую стра­тегию. Тогда значениям минимального гарантированного дохо­да первого игрока соответствует нижняя огибающая всех трех прямых. Согласно принципу максимина, оптимальному выбору игрока I будет соответствовать наивысшая точка, лежащая на данной огибающей, отмеченная на рисунке как (*x*1\*, *z\**). Зная ее, можно определить оптимальную смешанную стратегию перво­го игрока *х*\* = (*x*1\*, 1-*x*2\*) и цену игры, равную *z\**.

Исходя из отношения двойственности, которым, как было ус­тановлено в п. 6.1.5, связаны задачи обоих игроков, по оптималь­ной стратегии первого участника *х*\* однозначно определяется оптимальная стратегия его противника *у\**. Поскольку *у\** явля­ется результатом решения задачи линейного программирова­ния, то он обладает всеми свойствами допустимого базисного плана, т. е. в случае 2 х *п* игры имеет не более чем две ненулевых компоненты и не менее чем (*п*-2) нулевых. Номера ненулевых элементов *у\** определяются номерами линий, пересечение кото­рых определило оптимальную стратегию первого игрока. Дей­ствительно, игрок II знает оптимальную стратегию соперника, и применение им стратегий, соответствующих прямым, проходя­щим выше точки (*х*1\*, *z*\*), только увеличило бы его проигрыш.

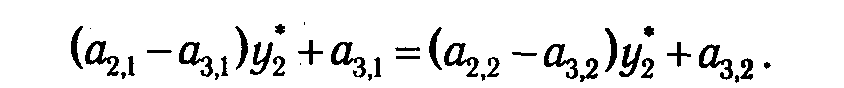
В рассматриваемом примере это линии *z*2 и *z*3, и, следова­тельно, в своей оптимальной стратегии второй игрок должен с ненулевыми вероятностями применять вторую и третью чистые стратегии (*у*2>0, *у*3 >0). На основе этого, а также учитывая условие нормировки



можем выразить: *y*3 = l – *y*2 тогда оптимальное значение *y*2\* мо­жет быть найдено из условия



или



В результате получаем оптимальную стратегию игрока II *у\*=* (0, *у*2\*, *у*3\*).

Очевидно, что поиск решения в игре *т* х 2 осуществляется аналогичным образом с точностью до наоборот: строятся гра­фики ожидаемого проигрыша игрока II, находится их верхняя огибающая и т. д.

Безусловно, графический способ в силу ограниченности круга задач, к которым он может быть применен, имеет скорее теоретическое, чем практическое значение. Однако он хорошо иллюстрирует содержательную сторону процесса поиска реше­ния в игре.

## 6.2. ТЕОРИЯ ОПТИМАЛЬНОГО УПРАВЛЕНИЯ

**6.2.1. Постановка и классификация задач теории оп­тимального управления.** В подавляющем большинстве рас­смотренных нами задач факторы, связанные с изменением изу­чаемых объектов и систем в течение времени, выносились за скобки. Возможно, при выполнении определенных предпосы­лок такой подход является конструктивным и правомерным. Однако очевидно и то, что это допустимо далеко не всегда. Су­ществует обширный класс задач, в которых необходимо найти оптимальные действия объекта, учитывающие динамику его состояний во времени и пространстве. Методы их решения со­ставляют предмет математической теории оптимального управ­ления.

В весьма общем виде задача оптимального управления мо­жет быть сформулирована следующим образом:

*Имеется некоторый объект, состояние которого харак­теризуется двумя видами параметров — параметрами состояния и параметрами управления, причем в зависи­мости от выбора последних процесс управления объек­том протекает тем или иным образом. Качество про­цесса управления оценивается с помощью некоторого функционала\*, на основе чего ставится задача: найти такую последовательность значений управляющих па­раметров, для которой данный функционал принимает экстремальное значение.*

\* *Функционалом* называется числовая функция, аргументами кото­рой, как правило, служат другие функции.

С формальной точки зрения многие проблемы оптимального управления могут быть сведены к задачам линейного или нели­нейного программирования большой размерности, так как каж­дой точке пространства состояний соответствует свой вектор неизвестных переменных. Все же, как правило, движение в дан­ном направлении без учета специфики соответствующих задач не приводит к рациональным и эффективным алгоритмам их ре­шения. Поэтому методы решения задач оптимального управле­ния традиционно связаны с другим математическим аппаратом, берущим свое начало от вариационного исчисления и теории интегральных уравнений. Следует также заметить, что опять-таки в силу исторических причин теория оптимального управ­ления была ориентирована на физические и технические при­ложения, и ее применение для решения экономических задач носит в определенном смысле вторичный характер. В то же вре­мя в целом ряде случаев модели исследования, применяющие аппарат теории оптимального управления, могут привести к содержательным и интересным результатам.

К сказанному выше необходимо добавить замечание о тес­ной связи, существующей между методами, применяемыми для решения задач оптимального управления, и динамическим про­граммированием. В одних случаях они могут использоваться на альтернативной основе, а в других довольно удачно дополнять друг друга.

Существуют различные подходы к классификации задач оп­тимального управления. Прежде всего, их можно классифици­ровать в зависимости от объекта управления:

* *задачи управления с* сосредоточенными параметрами;
* *задачи управления объектами с* распределенными парамет­рами.

Примером первых является управление самолетом как еди­ным целым, а вторых — управление непрерывным технологи­ческим процессом.

В зависимости от типа исходов, к которым приводят приме­няемые управления, выделяют *детерминированные* и *стоха­стические* задачи. В последнем случае результатом управле­ния является множество исходов, описываемых вероятностями их наступления.

По характеру изменения управляемой системы во времени различают задачи:

* с дискретно *изменяющимся временем*;
* с непрерывно *изменяющимся временем*.

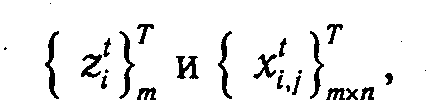
Аналогично классифицируются задачи управления объекта­ми с дискретным или непрерывным множеством возможных состояний. Задачи управления системами, в которых время и со­стояния меняются дискретно, получили название задач управле­ния *конечными автоматами*. Наконец, при определенных ус­ловиях могут ставиться задачи управления смешанными системами.

Многие модели управляемых систем основаны на аппарате дифференциальных уравнений как в обыкновенных, так и в час­тных производных. При исследовании систем с распределенны­ми параметрами, в зависимости от вида используемых диффе­ренциальных уравнений в частных производных, выделяют такие типы задач оптимального управления, как параболиче­ские, эллиптические или гиперболические.

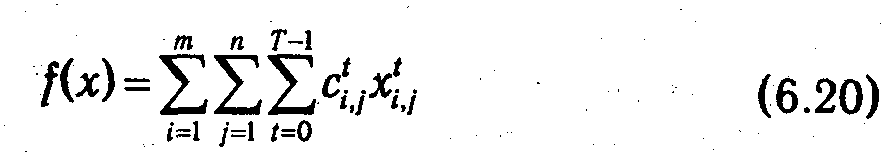
Рассмотрим два простейших примера задач управления эко­номическими объектами.

**Задача распределения ресурсов.** Имеется *т* складов с номерами *i* (*i*∊1:*m*), предназначенных для хранения однородно­го продукта. В дискретные моменты времени *t*∊0:(*T*-l) проис­ходит его распределение между объектами-потребителями (клиентами) с номерами *j*, *j*∊1:*n*. Пополнение запаса в пунктах хранения продукта в *t*-й момент времени определяется величи­нами *ait*, *i*∊1:*m*, а потребности клиентов в нем равняются *bjt*, *j*∊1:*n*. Обозначим через *cti,j* — затраты на доставку единицы продукта из *i*-го склада *j*-му потребителю в момент времени *t.* Также предполагается, что продукт, поступивший на склад в момент *t*, может быть использован, начиная со следующего мо­мента (*t*+l). Для сформулированной модели ставится задача найти такой план распределения ресурсов {*хti,j*}*Tm*x*n* , который минимизирует суммарные расходы на доставку потребителям продукции со складов в течение полного периода функциониро­вания системы.

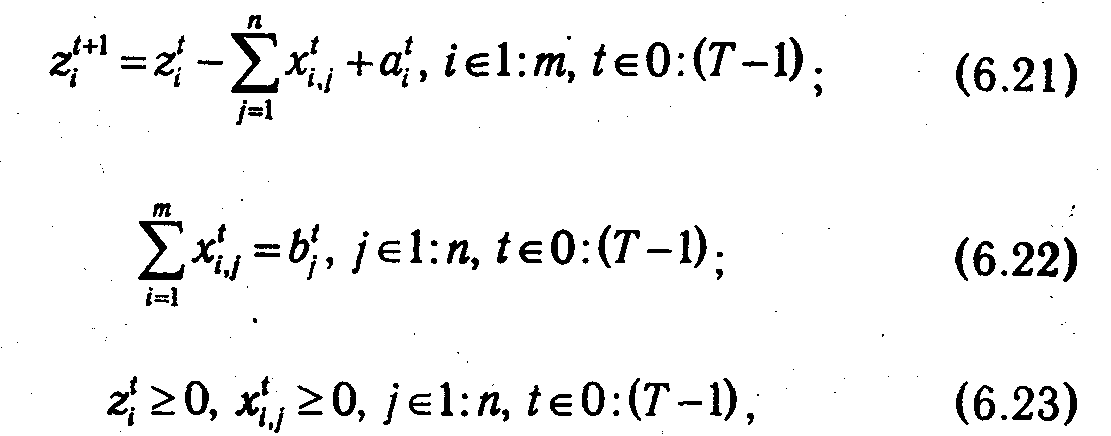
Обозначив через *хti,j* количество продукта, поставляемое *j*-му клиенту с *i*-го склада в *t*-й момент времени, а через *zti* — общее количество продукта на *i*-м складе, описанную выше про­блему можно представить как задачу нахождения таких сово­купностей переменных



которые обращают в минимум функцию



при условиях

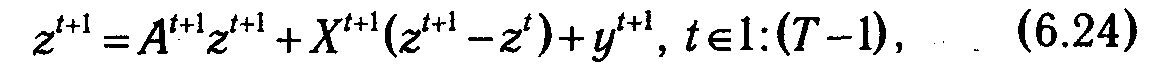


где объемы начальных запасов продукта на складах *z*0*i* = *ži* . пред­полагаются заданными.

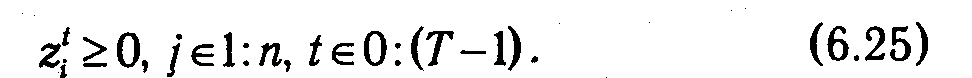
Задачу (6.20)-(6.23) называют *динамической транспорт­ной задачей линейного программирования*. С точки зрения приведенный выше терминологии независимые переменные *хti,j* представляют собой *параметры управления* системой, а зави­сящие от них переменные *zti* — совокупность *параметров состояния* системы в каждый момент времени *t.* Ограничения *zti* ≥ 0 гарантируют, что в любой момент времени с любого скла­да не может быть вывезен объем продукта, превышающий его фактическое количество, а ограничения (6.21) задают правила изменения этого количества при переходе от одного периода к другому. Ограничения данного вида, которые задают условия на значения параметров состояния системы, принято называть *фазовыми.*

Отметим также, что условие (6.21) служит простейшим при­мером фазовых ограничений, поскольку связываются значения параметров состояния для двух смежных периодов *t* и *t*+l. В общем случае может устанавливаться зависимость для груп­пы параметров, принадлежащих нескольким, возможно не­смежным, этапам. Такая потребность может возникнуть, на­пример, при учете в моделях фактора запаздывания поставок.

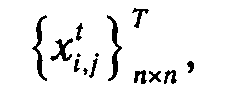
**Простейшая динамическая модель макроэкономики.** Представим экономику некоторого региона как совокупность *п* отраслей (*j*∊1:*п*), валовой продукт которых в денежном вы­ражении на некоторый момент *t* может быть представлен в виде вектора *zt*=(*zt*1 , *zt*2 ,..., *ztn*), где *t*∊0:(*Т*-1). Обозначим через *At* матрицу прямых затрат, элементы которой *ati,j* , отражают затра­ты продукции *i*-й отрасли (в денежном выражении) на изготов­ление единицы продукции *j*-й отрасли в *t*-й момент времени. Если *Xt* = ║*xti,j*║*n*x*m* — матрица, задающая удельные нормы продукции *i*-й отрасли, идущей на расширение производства в *j*-й отрасли, а *уt* = (*уt*1, *уt*2 , ..., *уtn*) — вектор объемов продукции от­раслей потребления, идущей на потребление, то условие рас­ширенного воспроизводства можно записать как



где *z*0 = *ž* — исходный запас продукции отраслей предполагает­ся заданным и



В рассматриваемой модели величины *zt* являются парамет­рами состояния системы, а *Xt* — управляющими параметрами. На ее базе могут быть поставлены различные задачи, типичным представителем которых является задача оптимального вывода экономики на момент *Т* к некоторому заданному состоянию *z*\*. Данная задача сводится к отысканию последовательности управляющих параметров

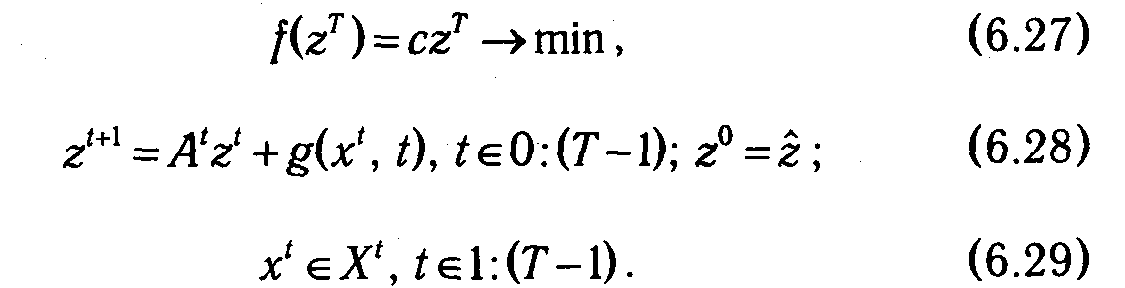


удовлетворяющих условиям (6.24)-(6.25) и минимизирующих функцию



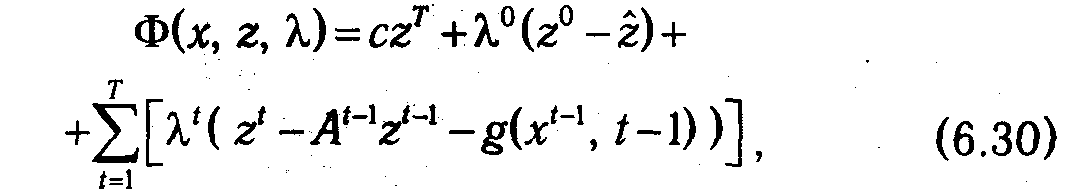
**6.2.2. Простейшая задача оптимального управления.** Один из приемов, применяемых для решения экстремальных задач, состоит в выделении некоторой проблемы, допускающей относительно несложное решение, к которой в дальнейшем могут быть сведены остальные задачи.

Рассмотрим так называемую *простейшую задачу управле­ния*. Она имеет вид

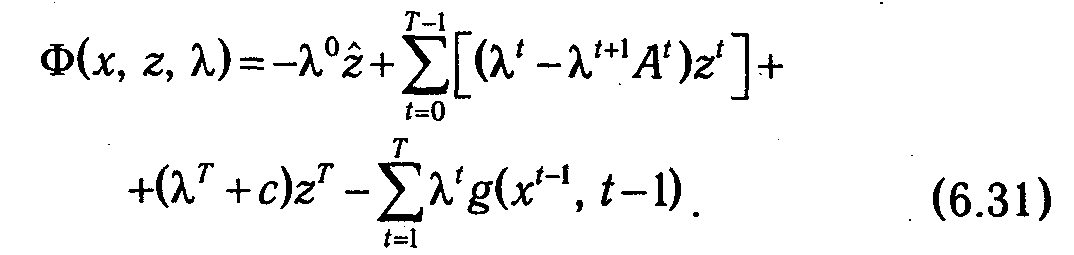


Специфика условий задачи (6.27)-(6.29) состоит в том, что функции качества управления (6.27) и ограничения (6.28) яв­ляются линейными относительно *zt*, в то же время функция *g*(*t*, *хt*), входящая в (6.28), может быть произвольной. Послед­нее свойство делает задачу нелинейной даже при *t*=1, т. е. в статическом варианте.

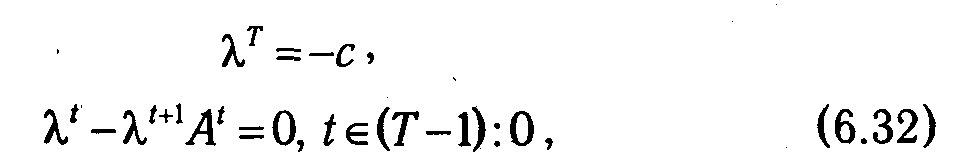
Общая идея решения задачи (6.27)-(6.29) сводится к ее «расщеплению» на подзадачи для каждого отдельно взятого момента времени, в предположении, что они успешно разреши­мы. Построим для задачи (6.27)-(6.29) функцию Лагранжа



где λ*t* — вектора множителей Лагранжа (*t*∊0:*Т*). Ограничения (6.29), носящие общий характер, в функцию (6.30) в данном случае не включены. Запишем ее в несколько иной форме

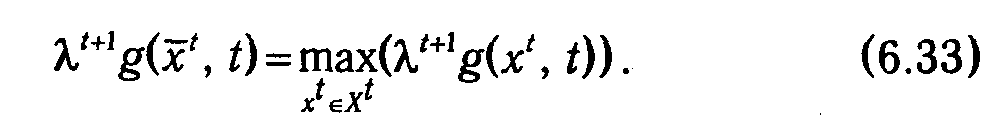


Необходимые условия экстремума функции Ф*(х, z,* λ) по со­вокупности векторов *zt* задаются системой уравнений



которая называется *системой для сопряженных перемен­ных*. Как можно заметить, процесс нахождения параметров λ*t* в системе (6.32) осуществляется рекуррентным образом в об­ратном порядке.

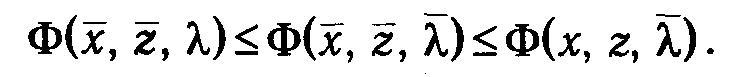
Необходимые условия экстремума функции Лагранжа по переменным λ*t* будут эквивалентны ограничениям (6.28), и, наконец, условия ее экстремума по совокупности векторов *хt*∊*Хt*, *t*∊1:(*Т*-1) должны быть найдены как результат реше­ния задачи



Таким образом, задача поиска оптимального управления сво­дится к поиску управлений, подозрительных на оптимальность, т. е. таких, для которых выполняется необходимое условие оп­тимальности. Это, свою очередь, сводится к нахождению таких *t*, *t*, *t*, удовлетворяющих системе условий (6.28), (6.32), (6.33), которая называется *дискретным принципом максиму­ма Понтрягина.*

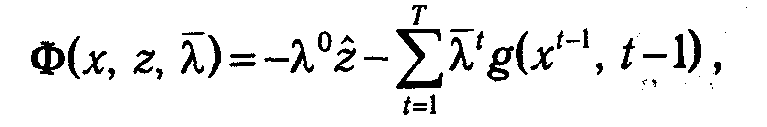
Справедлива теорема.

|  |
| --- |
| ***Теорема 6.2.*** *Совокупность векторов* *t*, *t*, *t*, *удов­летворяющих системе* (6.28), (6.32), (6.33), *образует седловую точку функции* Ф(*х, z,* λ) (6.30), *т. е. при лю­бых допустимых х, z,* λ *выполняются неравенства* |

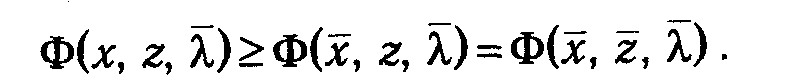


**Доказательство.**

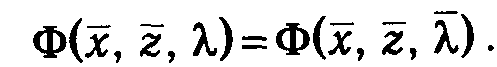
Пусть *t*, *t*, *t*, удовлетворяют системе (6.28), (6.32), (6.33). Тогда из (6.31) и (6.32) следует, что



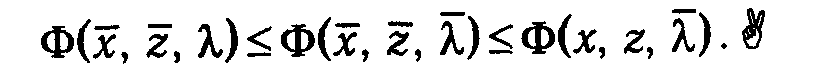
и поскольку *t* удовлетворяет (6.33), то



С другой стороны, в силу (6.28) из (6.30) следует, что при любом векторе *t*



Следовательно,



Применяя теорему (6.2), а также положения теории нели­нейного программирования, касающиеся связи между решени­ем экстремальной задачи и существованием седловой точки (см. п. 2.2.2), приходим к выводу о том, что векторы *t*, *t* явля­ются решением простейшей задачи оптимального управления (6.27)-(6.29).

В результате мы получили логически простую схему реше­ния данной задачи: из соотношений (6.32) определяются сопря­женные переменные *t*, затем в ходе решения задачи (6.33) на­ходятся управления *t*  и далее из (6.28) — оптимальная траектория состояний *t*,.

Предложенный метод относится к фундаментальным резуль­татам теории оптимального управления и, как уже это упомина­лось выше, имеет важное значение для решения многих более сложных задач, которые, так или иначе, сводятся к простей­шей. В то же время очевидны и пределы его эффективного ис­пользования, которые целиком зависят от возможности реше­ния задачи (6.33).

## КЛЮЧЕВЫЕ ПОНЯТИЯ

* Игра, игрок, стратегия.
* Игры с нулевой суммой.
* Матричные игры.
* Антагонистические игры.
* Принципы максимина и минимакcа.
* Седловая точка игры.
* Цена игры.
* Смешанная стратегия.
* Основная теорема матричных игр.
* Динамическая транспортная задача.
* Простейшая динамическая модель макроэкономики.
* Простейшая задача оптимального управления.
* Дискретный принцип максимума Понтрягина.

## КОНТРОЛЬНЫЕ ВОПРОСЫ

6.1. Кратко сформулируйте предмет теории игр как научной дисциплины.

6.2. Какой смысл вкладывается в понятие «игра»?

6.3. Для описания каких экономических ситуаций может быть применен аппарат теории игр?

6.4. Какая игра называется антагонистической?

6.5. Чем однозначно определяются матричные игры?

6.6. В чем заключаются принципы максимина и минимакcа?

6.7. При каких условиях можно говорить о том, что игра име­ет седловую точку?

6.8. Приведите примеры игр, которые имеют седловую точку и в которых она отсутствует.

6.9. Какие подходы существуют к определению оптимальных стратегий?

6.10. Что называют «ценой игры»?

6.11. Дайте определение понятию «смешанная стратегия».

## СПИСОК ЛИТЕРАТУРЫ

1. *Абрамов Л. М., Капустин В. Ф.* Математическое про­граммирование. Л.,1981.

2. *Ашманов С. А.* Линейное программирование: Учеб. посо­бие. М., 1981.

3. *Ашманов С. А., Тихонов А. В.* Теория оптимизации в зада­чах и упражнениях. М., 1991.

4. *Беллман Р.* Динамическое программирование. М., 1960.

5. *Беллман Р., Дрейфус С.* Прикладные задачи динамичес­кого программирования. М., 1965.

6. *Гавурин М. К., Малоземов В. Н.* Экстремальные задачи с линейными ограничениями. Л., 1984.

7. *Гасс С.* Линейное программирование (методы и приложе­ния). М., 1961.

8. *Гейл Д*. Теория линейных экономических моделей М., 1963.

9*. Гилл Ф., Мюррей У., Райт М.* Практическая оптимиза­ция / Пер. с англ. М., 1985.

10. *Давыдов Э. Г.* Исследование операций: Учеб. пособие для студентов вузов. М., 1990.

11. *Данциг Дж.* Линейное программирование, его обобще­ния и применения. М.,1966.

12. *Еремин И. И., Астафьев Н. Н.* Введение в теорию линей­ного и выпуклого программирования. М., 1976.

13. *Ермольев Ю.М., Ляшко И.И., Михалевич В.С., Тюптя В.И.* Математические методы исследования операций: Учеб. пособие для вузов. Киев, 1979.

14. *Зайченко Ю. П.* Исследование операций, 2-е изд. Киев, 1979.

15. *Зангвилл У. И.* Нелинейное программирование. Единый подход. М., 1973.

16*. Зойтендейк Г.* Методы возможных направлений. М., 1963.

17. *Карлин С.* Математические методы в теории игр, про­граммировании и экономике. М., 1964.

18. *Карманов В. Г.* Математическое программирование: Учеб. пособие. М., 1986.

19. *Корбут А.А., Финкелыитейн Ю. Ю.* Дискретное про­граммирование. М., 1968.

20. *Кофман А., Анри-Лабордер А.* Методы и модели иссле­дования операций. М., 1977.

21. *Кюнце Г.П., Крелле В.* Нелинейное программирование. М.,1965.

22. *Ляшенко И.Н., Карагодова Е.А., Черникова Н.В., Шор Н.3.* Линейное и нелинейное программирование. Киев, 1975.

23. *Мак-Кинси Дж.* Введение в теорию игр. М., 1960.

24. *Мухачева Э. А., Рубинштейн Г. Ш.* Математическое программирование. Новосибирск, 1977.

25. *Нейман Дж., Моргенштерн О.* Теория игр и экономи­ческое поведение. М, 1970.

26*. Оре О.* Теория графов. М., 1968.

27. *Таха X.* Введение в исследование операций/ Пер. с англ. М.,1985.

28. *Фиакко А., Мак-Кормик Г.* Нелинейное программирова­ние. Методы последовательной безусловной минимизации. М.,1972.

29. *Хедли Дж.* Нелинейное и динамическое программирова­ние. М., 1967.

30*. Юдин Д.Б., Гольштейн Е.Г.* Линейное программирова­ние (теория, методы и приложения). М., 1969.

31*. Юдин Д.Б., Гольштейн Е.Г.* Линейное программирова­ние. Теория и конечные методы. М., 1963.

32. *Lapin L.* Quantitative methods for business decisions with cases. Fourth edition. HBJ, 1988.

33. *Liitle I.D.C., Murty K.G„ Sweeney D.W., Karel C.* An al­gorithm for traveling for the traveling salesman problem. — Operation Research, 1963, vol.11, No. 6, p. 972-989/ Русск. пер.: *Литл Дж., Мурти К., Суини Д., Керел К.* Алгоритм для решения задачи о коммивояжере. — В кн.: Экономика и мате­матические методы, 1965, т. 1, № 1, с. 94-107.

**Содержание**

ПРЕДИСЛОВИЕ 2

ВВЕДЕНИЕ 3

ГЛАВА 1. ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ 8

1.1. ПОСТАНОВКА ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ 9

1.2. ОСНОВНЫЕ СВОЙСТВА ЗЛП И ЕЕ ПЕРВАЯ ГЕОМЕТРИЧЕСКАЯ ИНТЕРПРЕТАЦИЯ 11

1.3. БАЗИСНЫЕ РЕШЕНИЯ И ВТОРАЯ ГЕОМЕТРИЧЕСКАЯ ИНТЕРПРЕТАЦИЯ ЗЛП 15

1.4. СИМПЛЕКС-МЕТОД 17

1.5. МОДИФИЦИРОВАННЫЙ СИМПЛЕКС-МЕТОД 26

1.6. ТЕОРИЯ ДВОЙСТВЕННОСТИ В ЛИНЕЙНОМ ПРОГРАММИРОВАНИИ 30

1.7. ДВОЙСТВЕННЫЙ СИМПЛЕКС-МЕТОД 37

КЛЮЧЕВЫЕ ПОНЯТИЯ 42

КОНТРОЛЬНЫЕ ВОПРОСЫ 43

ГЛАВА 2. НЕЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ 44

2.1. МЕТОДЫ РЕШЕНИЯ ЗАДАЧ НЕЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ 44

2.2. ДВОЙСТВЕННОСТЬ В НЕЛИНЕЙНОМ ПРОГРАММИРОВАНИИ 55

КЛЮЧЕВЫЕ ПОНЯТИЯ 59

КОНТРОЛЬНЫЕ ВОПРОСЫ 59

ГЛАВА 3. ТРАНСПОРТНЫЕ И СЕТЕВЫЕ ЗАДАЧИ 60

3.1. ТРАНСПОРТНАЯ ЗАДАЧА И МЕТОДЫ ЕЕ РЕШЕНИЯ 60

3.2. СЕТЕВЫЕ ЗАДАЧИ 66

КЛЮЧЕВЫЕ ПОНЯТИЯ 73

КОНТРОЛЬНЫЕ ВОПРОСЫ 73

ГЛАВА 4. ДИСКРЕТНОЕ ПРОГРАММИРОВАНИЕ 74

4.1. ТИПЫ ЗАДАЧ ДИСКРЕТНОГО ПРОГРАММИРОВАНИЯ 74

4.2. МЕТОД ГОМОРИ 78

4.3. МЕТОД ВЕТВЕЙ И ГРАНИЦ 81

КЛЮЧЕВЫЕ ПОНЯТИЯ 86

КОНТРОЛЬНЫЕ ВОПРОСЫ 86

ГЛАВА 5. ДИНАМИЧЕСКОЕ ПРОГРАММИРОВАНИЕ 86

5.1. ОБЩАЯ СХЕМА МЕТОДОВ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ 86

5.2. ПРИМЕРЫ ЗАДАЧ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ 93

КЛЮЧЕВЫЕ ПОНЯТИЯ 101

КОНТРОЛЬНЫЕ ВОПРОСЫ 101

ГЛАВА 6. КРАТКИЙ ОБЗОР ДРУГИХ РАЗДЕЛОВ ИССЛЕДОВАНИЯ ОПЕРАЦИЙ 101

6.1. ТЕОРИЯ ИГР 101

6.2. ТЕОРИЯ ОПТИМАЛЬНОГО УПРАВЛЕНИЯ 108

КЛЮЧЕВЫЕ ПОНЯТИЯ 112

КОНТРОЛЬНЫЕ ВОПРОСЫ 112

СПИСОК ЛИТЕРАТУРЫ 112