ФЕДЕРАЛЬНОЕ АГЕНТСТВО МОРСКОГО И РЕЧНОГО ТРАНСПОРТА

ФЕДЕРАЛЬНОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ВОДНЫХ КОММУНИКАЦИЙ»

ЭКОНОМИКО-МАТЕМАТИЧЕСКИЕ МЕТОДЫ И МОДЕЛИ В УПРАВЛЕНИИ ВОДНЫМ ТРАНСПОРТОМ

Линейное программирование

«Рекомендовано УМО по образованию в области эксплуатации водного транспорта» в качестве учебного пособия для студентов (курсантов) высших учебных заведений, обучающихся по укрупненной группе специальностей 180000 «Морская техника»

Санкт-Петербург

2012

УДК 519.852

ББК 22.18

Рецензенты:
доктор экономических наук, профессор ЦНИИМФ
Л. Н. Буянова;
доктор технических наук, профессор СПГУВК
Д. П. Голоскоков;
доктор технических наук, профессор СПГУВК
А. В.Земсков

Экономико-математические методы и модели в управлении водным транспортом. Линейное программирование: учеб. пособие / В. А. Бабурин и др. / Под редакцией профессора В. А. Бабурина. — СПб.: СПГУВК, 2012. — 206 с.

ISBN 978-5-88789-363-1

В учебном пособии излагаются некоторые теоретические основы оптимальных решений при управлении экономическими системами, а также рекомендации по применению методов получения этих решений для решения практических задач.

Учебное пособие предназначено для студентов факультета экономики и финансов, изучающих «Экономико-математические методы и модели в управлении и экономике» (специальности 190701.65 «Организация перевозок и управление на транспорте», 080507.65 «Менеджмент организации», 080502.65 «Экономика и управление на предприятии водного транспорта», 080503.65 «Антикризисное управление»), «Математические методы в экономике» (специальности 080105.65 «Финансы и кредит», 080109.65 «Бухгалтерский учет, анализ и аудит»); «Математическую экономику» (специальность 080801.65 «Прикладная информатика в экономике») как по дневной, так и по вечерней, и по заочной формам обучения.

Пособие может оказаться полезным для аспирантов, занимающихся разработкой экономико-математических моделей и оптимальных параметров экономических систем.

УДК 519.852

ББК 22.18

ISBN 978-5-88789-363-1

СОДЕРЖАНИЕ

Введение

Применение экономико-математических методов в управлении экономическими системами можно разделить на две части:

- разработку математической или экономико-математической модели функционирования системы;
- определение наилучших по какому-либо критерию управляемых параметров системы.

Вторая из названных частей и является предметом теории оптимального управления экономическими системами, которая соответственно и рассматривается в данном учебном пособии.

В первой главе учебного пособия конкретизируется постановка задач оптимального управления, приводятся критерии и их классификация, необходимые сведения о характеристиках множеств и функций, которые наиболее существенны при поиске оптимальных решений. Эти общие положения необходимы для конкретного понимания и описания реальных задач оптимизации и анализа решений. Далее рассмотрены конкретные методы поиска оптимальных решений при разных математических моделях экономических систем, характерных, в том числе, и для водного транспорта.

Основная направленность учебного пособия состоит в том, чтобы изложить основные положения теории оптимального управления экономическими системами в том объеме, который необходим для самостоятельной работы обучающихся с целью получения практических результатов при поиске оптимальных решений, а также для самостоятельного более углубленного изучения тех или иных методов оптимизации. Конечно, наилучшие решения не даются просто. Поэтому процедуры поиска оптимальных решений для практических задач достаточно громоздки и требуют, как правило, применения средств вычислительной техники. В учебном

пособии на примере пакета QSB и электронных таблиц Excel достаточно подробно рассматриваются вопросы поиска оптимальных решений с применением средств вычислительной техники.

Следует отметить, что хотя все рассмотренные методы проиллюстрированы примерами, освоение методов может быть успешным только на базе самостоятельного решения задач оптимизации: для студентов — в рамках практических и лабораторных занятий, для аспирантов — при решении конкретных задач научного исследования.

Данное учебное пособие является вторым в серии «Экономико-математические методы и модели в управлении водным транспортом». Первое — «Системы массового обслуживания» было подготовлено практически тем же авторским коллективом в 2009 г. и пользуется большим спросом у студентов, изучающих соответствующие учебные дисциплины, а также у аспирантов.

Авторы надеются, что и настоящее издание также будет полезно курсантам, студентам и аспирантам вузов водного транспорта, для которых учебное пособие и написано.

Глава 1. Характеристика задач оптимального управления

1.1. Постановка и классификация задач оптимального управления

Любая экономическая система представляет собой совокупность элементов и их связей, предназначенную для преобразования материальных и трудовых ресурсов в другие ресурсы (материальные и/или духовные) или услуги (например транспортные услуги). Характеристикой экономической системы является совокупность ее параметров. Они делятся на неуправляемые и управляемые в том смысле, что последним можно придавать желаемые значения для получения требуемых показателей оценки работы системы. При этом в силу имеющихся связей между элементами системы показатели оценки работы системы являются зависимыми как от неуправляемых, так и от управляемых параметров. Разработка математической модели функционирования системы, по сути, сводится к построению математического описания показателей оценки ее работы. При оптимальном управлении системой один из показателей оценки работы системы выбирается в качестве цели (критерия) управления, максимальное (max) или минимальное (min) значение которого необходимо обеспечить выбором значений (оптимальных) управляемых параметров. Если другие показатели оценки работы системы при этом должны удовлетворять некоторым условиям, то эти условия образуют ограничения при поиске оптимальных значений управляемых параметров.

Математической моделью задачи оптимального управления называется описание множества допустимых решений (управлений) и критерия управления с использованием математических средств.

Прежде чем математически сформулировать задачу оптимального управления, обозначим через X вектор управления, компоненты которого $x_1, x_2, x_3, \ldots, x_n$ являются управляемыми параметрами системы, а через A — вектор неуправляемых параметров с компонентами $a_1, a_2, \ldots, a_l$. Тогда за-

дачу оптимального управления можно сформулировать следующим образом. Для заданной функции цели *z(A,X)* найти такое оптимальное управление X^*, при котором функция цели достигает экстремума, т.е.:

$$z\left(A, X^*\right) = \underset{X \in X_{доп}}{extr} \; z\left(A, X\right), \tag{1.1}$$

где *extr* соответствует *max* или *min* , а область допустимых управлений $X_{доп}$ определяется так, чтобы выполнялись ограничения:

$$G_j^*\left(A, X\right) \theta b_j \quad j = 1, 2, ..., m \,. \tag{1.2}$$

В последней формуле символом θ обозначена одна из операций отношения. Значение $z(A,X^*)$ является оптимальным значением функции цели, а формула (1.1) выражает критерий оптимальности.

В литературе и справочных системах программных средств по задачам оптимизации могут использоваться и другие формы записи задачи оптимального управления. Например, можно записать: *найти управления, при которых функция цели стремится к минимуму, т.е.*

$$z\left(A, X\right) \rightarrow \underset{X \in X_{доп}}{min}, \tag{1.3}$$

а область допустимых управлений $X_{доп}$ определяется выполнением ограничений:

$$G_j^*\left(A, X\right) = 0 \, при \; j = 1, 2, ..., h; \quad G_j^*\left(A, X\right) \le 0 \, при \; j = h+1, h+2, ..., m \,. \tag{1.4}$$

Приведенная формулировка является такой же общей, как и формулировка в виде (1.1) и (1.2). Действительно, т.к. для любой функции *f(X)* справедливо: $\max f\left(A, X\right) = \min\left(-f\left(A, X\right)\right)$, то всегда задачу на максимум умножением критерия на минус единицу можно свести к задаче на минимум, и наоборот. Значение ограничений b_j в формуле (1.2) можно рассматривать как дополнительные неуправляемые параметры системы и вклю-

чить их во множество A, а ограничения типа $\geq$ умножением на минус единицу всегда можно свести к ограничениям типа $\leq$.

Поскольку множество неуправляемых параметров в задаче поиска оптимального управления является постоянным, то часто при записи задачи оптимального управления его просто опускают. Поэтому более компактно математическая формулировка задачи оптимального управления будет иметь вид: *найти управления x, при которых функция цели стремится к минимуму, т.е.*:

$$z(x) \to \min_{X \in X_{доп}} \qquad (1.5)$$

и выполняются ограничения

$$G_j(X) = 0 \; при \; j = 1,2,...,h \quad G_j(X) \leq 0 \; при \; j = h+1, h+2,...,m \; . \qquad (1.6)$$

Словесное определение задачи можно сформулировать также в виде: *по критерию оптимальности (1.5) найти управления, удовлетворяющие ограничениям (1.6).*

Если рассматривается управление системой во времени, то управляемые параметры, очевидно, также будут изменяться во времени, т.е. будут функциями времени. При этом вместо вектора X необходимо рассматривать вектор-функцию $X(t)$ с компонентами $x_1(t), x_2(t), x_3(t), ..., x_n(t)$.

Показатели оценки работы системы в общем случае могут зависеть как от функций управления $x_1(t), x_2(t), x_3(t), ..., x_n(t)$, так и от производных *k-го* порядка по времени от этих функций $x_1^{(k)}(t), x_2^{(k)}(t), x_3^{(k)}(t), ..., x_n^{(k)}(t)$, $k=1,2,...$. Неуправляемые параметры, в свою очередь, также могут изменяться со временем. Поэтому показатели оценки работы системы будут зависеть не только от управлений и производных от них, но и от $A(t)$ или, в конечном счете, от времени t. Таким образом, функция цели и ограничения будут в этом случае функциями от функций и называются поэтому функционалами. Формулировка задачи оптимального управления примет в этом

случае вид: *найти функции управления X(t), при которых функционал цели стремится к минимуму, т.е.*:

$$z\left(X(t), X^{(1)}(t), X^{(2)}(t), ..., X^{(k)}(t), t\right) \to min, \qquad (1.7)$$

и выполняются ограничения

$$G_j\left(X(t), X^{(1)}(t), X^{(2)}(t), ..., X^{(k)}(t), t\right) = 0 \, при \; j = 1, 2, ..., h;$$

$$G_j\left(X(t), X^{(1)}(t), X^{(2)}(t), ..., X^{(k)}(t), t\right) \le 0 \, при \; j = h+1, h+2, ..., m. \qquad (1.8)$$

Вместо записи в виде (1.5) и (1.6) может использоваться также формулировка задачи оптимального управления в виде:

$$z(x) \to \min_{X \in X_{доп}}, \qquad (1.9)$$

$$при \; X_{доп} = \left\{ X : G_j(X) = 0 \, при \; j = 1, 2, ..., h; G_j(X) \le 0 \, при \; j = h+1, h+2, ..., m \right\}. \qquad (1.10)$$

Такая запись означает, что ставится задача поиска такого управления из множества допустимых управлений $X_{доп}$, при котором функция цели $z(X)$ достигает минимума, а множество допустимых управлений состоит из всех возможных управлений X, для которых выполняются заданные ограничения.

Аналогично вместо записи в виде (1.7) и (1.8) может использоваться также формулировка задачи оптимального управления в виде:

$$z\left(x(t), x^{(1)}(t), x^{(2)}(t), ..., x^{(k)}(t), t\right) \to \min_{x(t) \in X_{доп}}, \qquad (1.11)$$

$$при \; X_{доп} = \left\{ X : G_j\left(X(t), X^{(1)}(t), X^{(2)}(t), ..., X^{(k)}(t), t\right) = 0 \, при \; j = 1, 2, ..., h; \right.$$
$$\left. G_j\left(X(t), X^{(1)}(t), X^{(2)}(t), ..., X^{(k)}(t), t\right) \le 0 \, при \; j = h+1, h+2, ..., m \right\}. \qquad (1.12)$$

Запись в виде формул (1.11), (1.12) означает, что ставится задача поиска таких функций управления из множества допустимых функций управлений

$X_{доп}$, при которых функционал цели $z\left(X(t), X^{(1)}(t), X^{(2)}(t),..., X^{(k)}(t), t\right)$ достигает минимума, а множество допустимых функций управления состоит из всех возможных управлений *X(t)*, для которых выполняются заданные ограничения.

Формулировка задачи оптимального управления в виде формул (1.11) и (1.12) является наиболее общей, т.к. при отсутствии зависимости управлений и неуправляемых параметров системы от времени эти формулы принимают вид (1.9) и (1.10).

Таким образом, математическая формулировка задачи оптимального управления в общем виде может записываться в разных формах. Выбор конкретной формы определяется обычно автором постановки задачи.

Классификация задач математического программирования может производиться по разным признакам, которые связаны или с особенностями функции цели и ограничений, или с методами поиска оптимальных управлений.

Если управления не зависят от времени, а значит, не зависят от времени функция цели и ограничения, то задачи поиска оптимальных управлений в этом случае называют задачами *математического программирования*. Математическая формулировка таких задач в общем случае определяется, например, в виде формул (1.9) и (1.10).

Если управления зависят от времени, т.е. математическая формулировка задачи поиска оптимального управления задается, например, в виде формул (1.11) и (1.12), то задачи поиска оптимальных управлений в этом случае называют *вариационными задачами*. Различают вариационные задачи с *голономными* и *неголономными* связями. В первом случае ограничения зависят только от функций управления и времени, т.е. множество допустимых управлений определяется в виде:

$$X_{доп} = \left\{ X(t) : G_j\left(X(t), t\right) = 0 \, при \; j = 1, 2, ..., h; G_j\left(X(t), t\right) \le 0 \, при \; j = h+1, h+2, ..., m \right\},$$

иначе говоря, ограничения являются алгебраическими функциями от управлений и времени. При неголономных связях область допустимых управлений определяется системой дифференциальных уравнений и задается в виде:

$$X_{доп} = \left\{ X(t) : G_j\left(X(t), X^{(1)}(t), X^{(2)}(t), ..., X^{(k)}(t), t\right) = 0 \; при \; j = 1, 2, ..., m \right\}.$$

В зависимости от вида функции цели (или функционала в вариационных задачах) и ограничений в задачах оптимизации различают *линейные* и *нелинейные* задачи поиска оптимальных управлений, или короче — *линейные* и *нелинейные* задачи оптимизации.

В линейной задаче математического программирования функция цели и все ограничения должны быть линейными функциями управлений.

В линейной вариационной задаче функционал цели и ограничения должны быть линейными относительно функций управления, их производных и времени.

Если в сформулированной задаче оптимизации отсутствуют ограничения на управления, то такая задача называется *задачей безусловной оптимизации*, в противном случае — *задачей условной оптимизации.*

Вариационные задачи для экономических систем обычно слишком сложны за счет большой размерности вектор — функции управления, т.е. за счет большого количества управляемых параметров. Поэтому для оптимизации экономических систем наибольшее значение имеют задачи математического программирования, более детальную классификацию которых рассмотрим далее.

При оптимизации экономических систем управляемые параметры физически часто представляют либо ресурсы для производства продукции, либо объемы выпускаемой продукции, либо стоимости ресурсов или производимой продукции. Поэтому значения управляемых параметров, как правило, не могут быть отрицательными, т.е.:

$$x_i \geq 0 \qquad i = 1, 2, ..., n\,. \tag{1.13}$$

Эти ограничения обычно подразумеваются по умолчанию и в явном виде не приводятся при математической формулировке задачи оптимизации. Далее везде так и будем поступать, но если ограничения (1.13) отсутствуют, это следует специально указывать.

После сделанного замечания вернемся к классификации задач математического программирования. Основным признаком их классификации является вид функции цели и ограничений. Используя этот признак, можно выделить следующие виды математического программирования.

1. Линейное программирование: раздел математического программирования, который позволяет решать экстремальные задачи при наличии ограничений, причём искомые переменные и в целевой функции, и в ограничениях должны быть только в 1-й степени и не перемножаться друг на друга.

Методы линейного программирования широко используются в управлении экономикой, т.к. они достаточно просты и доступны. В самом общем виде модель задачи линейного программирования выглядит следующим образом:

$$1)\begin{cases} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n \gtrless b_1 \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n \gtrless b_2 \\ ..... \\ a_{m1}x_1 + a_{m2}x_2 + ... + a_{mn}x_n \gtrless b_m \end{cases} \tag{1.13а}$$

$$2)\, x_j \geq 0 \left(для \forall_j = 1 \div n\right)$$

$$3)\, Z = C_1x_1 + C_2x_2 + ... + C_nx_n \rightarrow max(min).$$

Здесь: X_i — искомая переменная;

b_i — численные значения ограничений задачи.

1) — ограничения задачи. Отражают условия, при которых данная задача должна решаться (производственные и ресурсные);

2) — ограничения на переменные (допустимые значения для экономических задач);

3) — целевая функция. Отражает основную цель решаемой задачи.

2. Целочисленное линейное программирование. Функция цели и ограничения также задаются в виде формул (1.13а), т.е. являются линейными, но хотя бы на одно искомое управление x_i накладывается требование целочисленности. Если все x_i могут принимать значения только целых чисел, то говорят о полностью целочисленной задаче линейного программирования, иначе — о частично целочисленной.

3. Дробно-линейное программирование. Ограничения в задачах этого типа являются линейными, а функция цели является дробно–линейной функцией вида:

$$z(x) = \frac{\sum_{i=1}^{n} \alpha_i x_i}{\sum_{i=1}^{n} \beta_i x_i} \to \begin{matrix} min \\ x \in X_{доп} \end{matrix}. \qquad (1.14)$$

4. Квадратичное программирование. Так называются задачи математического программирования, в которых область допустимых значений определяется линейными функциями ограничений, а функция цели — квадратичной функцией управляемых параметров, т.е. имеет вид:

$$z(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_{ij} x_i x_j \to \begin{matrix} min \\ x \in X_{доп} \end{matrix}.$$

5. Сепарабельное программирование. Функция цели для задач этого типа (нелинейная в общем случае) имеет вид:

$$z(x) = \sum_{i=1}^{n} \varphi_i(x_i) \to \begin{matrix} min \\ x \in X_{доп} \end{matrix}, \qquad (1.15)$$

а ограничения линейны.

Очевидно, что линейное программирование является частным случаем сепарабельного программирования.

В некоторых случаях существенные особенности функции цели или ограничений дают основание рассматривать соответствующую задачу как один из видов задач оптимизации. Примером такой задачи может служить *транспортная задача* линейного программирования с вектором управления $x=(x_{11},x_{12},...,x_{1m},x_{21},x_{22},...,x_{2m},...,x_{n1},x_{n2},...,x_{nm})$, математическая формулировка которой имеет вид:

$$z(x)=\sum_{i=1}^{n}\sum_{j=1}^{m}c_{ij}x_{ij}\rightarrow \min_{x\in X_{доп}}, \tag{1.16}$$

$$X_{доп}=\left\{\begin{array}{l} x: G_i(x)=\sum_{j=1}^{m}x_{ij}=S_i;\ i=1,2,...,n; \\ G_i(x)=\sum_{i=1}^{n}x_{ij}=D_i;\ j=1,2,...,m; \end{array}\right\}. \tag{1.17}$$

Если в формуле (1.17) все S_i и D_i равны 1, то приведенная математическая формулировка задачи оптимизации будет соответствовать *задаче о назначениях.*

Классификацию задач математического программирования по виду функции цели и ограничений на область допустимых управлений можно было бы продолжить, но именно отмеченные типы задач имеют наиболее развитые методы их решения и поэтому выделяются особо.

Задачи оптимального управления могут объединяться по методу их решения. Примером может служить метод *динамического программирования*, который предусматривает пошаговое решение задачи. При этом функция цели и ограничения на каждом шаге могут быть как линейными, так и нелинейными.

На основании наиболее общей формулировки задачи оптимального управления (формулы 1.11 и 1.12) логично ожидать, что методы поиска

оптимальных управлений существенно зависят от характеристик используемых функций и характеристик множества допустимых управлений. Так как характеристики функций и множеств подробно изучаются в курсах высшей математики, остановимся только на тех, которые наиболее существенны при решении задач оптимизации.

1.2. Множество допустимых решений и его характеристики

Каждому конкретному набору значений координат $x_1, x_2, x_3, \ldots, x_n$ (или управлений) будет соответствовать точка в n-мерном пространстве. Совокупность таких точек в n-мерном пространстве называется *множеством* или в нашем рассмотрении — *множеством управлений*.

Вектор — отрезок, имеющий определенную длину и направление, который соединяет две точки. Отрезок, соединяющий начало координат с конкретной точкой X с координатами $x_1, x_2, x_3, \ldots, x_n$, есть вектор X с компонентами $x_1, x_2, x_3, \ldots, x_n$. *Длина* или *модуль вектора* X обозначается $|X|$. Для двух различных точек $X^1 = (x_{11}, x_{21}, \ldots, x_{n1})$ и $X^2 = (x_{12}, x_{22}, \ldots, x_{n2})$ можно построить вектор из первой точки во вторую или из второй в первую. Первый из таких векторов является разностью $X^2 - X^1 = X^2 + (-X^1)$, а второй — разностью $X^1 - X^2 = X^1 + (-X^2)$ исходных векторов X^1, X^2. Очевидно, что должно выполняться условие $|X^1 - X^2| = |X^2 - X^1|$. Если из конца вектора X^1 построить вектор такой же длины и такого же направления, как вектор X^2, то вектор, соединяющий начало вектора X^1 (в нашем случае — начало координат) и конец построенного вектора, есть вектор суммы векторов X^1 и X^2 (рис. 1.1). Легко проверить, что $X^1 + X^2 = X^2 + X^1, (X^1 + X^2) + X^3 = X^1 + (X^2 + X^3), |X^1 + X^2| \le |X^1| + |X^2|$. Последнее условие известно как правило треугольника — сумма длин двух сторон треугольника больше длины третьей стороны.

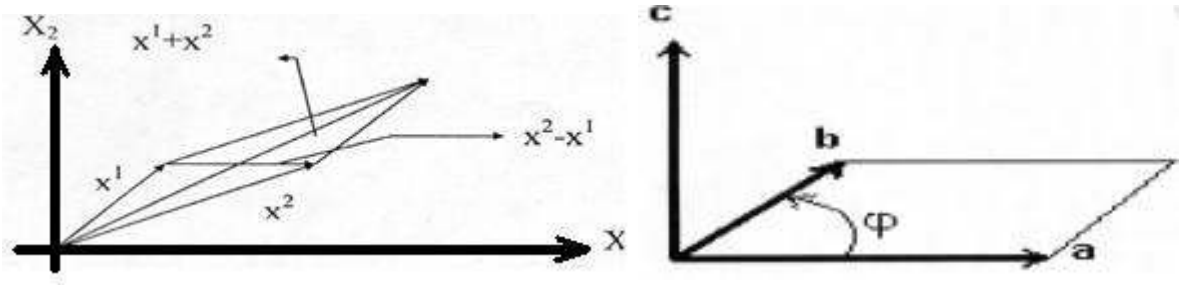

Рисунок 1.1. Действия над векторами:
а) сумма и разность векторов; б) векторное произведение векторов

Вектор, длина которого равна 1, называется *ортом*. При умножении орта на заданное число создается вектор, направление которого совпадает с направлением орта, а длина равна заданному числу. Если с каждой координатной осью x_i связать орт ς_i, то вектор X через орты и координаты x_i по осям определится формулой:

$$X = \sum_{i=1}^{n} x_i \cdot \varsigma_i , \qquad (1.18)$$

т.е. n-мерный вектор X равен сумме векторов, направление которых совпадает с направлением координатных осей, а длина каждого равна проекции вектора x на соответствующую ось. Собственно, запись $X = (x_1, x_2, x_3, \ldots, x_n)$ является более краткой записью суммы n векторов, заданных по координатным осям.

Скалярным произведением двух векторов a и b называется скаляр, т.е. число, значение которого равно $|a| \cdot |b| \cdot \cos\varphi$, где φ — угол между векторами a и b,приведенными к общему началу. В прямоугольной системе координат угол между любой парой ортов равен 90° и, следовательно, $\cos(90°) = 0$. Поэтому скалярное произведение векторов $X^1 = (x_{11}, x_{21}, \ldots, x_{n1})$ и $X^2 = (x_{12}, x_{22}, \ldots, x_{n2})$ в конечном счете определяется формулой:

$$\left(X^1, X^2\right) = X^1 \cdot X^2 = \sum_{i=1}^{n} x_{i1} \cdot x_{i2}. \qquad (1.19)$$

Из определения скалярного произведения непосредственно следует, что скалярное произведение вектора на самого себя определяет квадрат его длины. Поэтому:

$$X \cdot X = \left|X\right|^2 = \sum_{i=1}^{n} x_i^2; \left|X\right| = \sqrt{\sum_{i=1}^{n} x_i^2}. \qquad (1.20)$$

Кроме суммы, разности скалярного произведения для векторов, определено векторное произведение, которое для векторов *a* и *b* обозначается как $c = a*b$.

Длина вектора *c* равна $|a| \cdot |b| \cdot sin(\varphi)$ (т.е. равна площади параллелограмма, построенного на векторах *a* и *b*, как на сторонах), где φ — угол между векторами. Вектор *c* направлен перпендикулярно векторам *a* и *b* и в такую сторону, чтобы после совмещения начал всех трех векторов кратчайший поворот от *a* к *b* казался наблюдателю, смотрящему с конца вектора *c*, идущим против часовой стрелки (см. рис. 1.1).

Если вектора $P_1, P_2, ..., P_m$ удовлетворяют равенству:

$$K_1 \cdot P_1 + K_2 \cdot P_2 + K_3 \cdot P_3 + ... + K_m \cdot P_m = 0$$

только при $k_i=0, i=1, 2, ..., m,$ то они называются линейно-независимыми.

Все возможные наборы значений $x_1, x_2, x_3, ..., x_n$ образуют множество точек или просто множество *X*. Если вектор *X* с его компонентами $x_1, x_2, x_3, ..., x_n$ есть вектор управления, то возможное множество *X* есть множество допустимых управлений или решений. Если *X* есть множество точек, попадающих в шар радиуса *R* в *n*-мерном пространстве, то этот факт обозначают как $X \in R^n$. Если границы множества задаются особо, то записывают:

$$X = \left\{X : G_j\left(X\right) = 0 \, при \, j = 1,2,...,h; G_j\left(X\right) \le 0 \, при \, j = h+1, h+2, ..., m\right\},$$

что означает: множество X образуют те вектора X, для которых выполняется $G_j(X)=0 при\ j=1,2,...,h$ и $G_j(X)\le 0 при\ j=h+1,h+2,...,m$, где m — общее число ограничений на допустимые управления.

Классификация множества допустимых решений может проводиться по разным характеристикам множества.

Множество X называется *выпуклым*, если вместе с любыми двумя точками P и Q из этого множества ему принадлежат все точки соединяющего их отрезка. Более строгое определение: множество X в n-мерном пространстве называется выпуклым, если для любых двух точек $P, Q \in X$ выпуклая комбинация $k \cdot P+(1-k)Q$ при всех $0<k<1$ будет принадлежать множеству X.

Для двухмерного пространства при $X=(x_1, x_2)$ это означает, что все точки линии, соединяющей две точки, принадлежащие множеству X, будут также принадлежать этому множеству.

На рис. 1.2 приведены примеры выпуклого и невыпуклого множества X для двухмерного пространства.

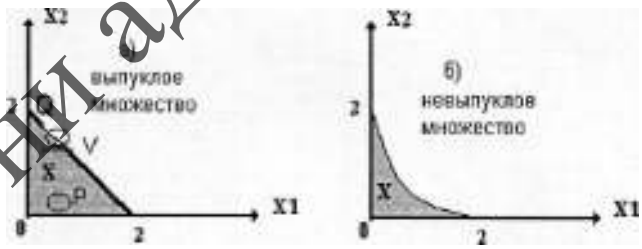

Рисунок 1.2. Примеры множеств

Точка $P \in X$ называется *внутренней точкой множества* X, если любая ε-окрестность точки P целиком содержится в X (точка P на рис. 1.2, а).

Точка называется *предельной точкой множества* X, если каждая ε-окрестность содержит точки, принадлежащие множеству X (точки P и V на рис. 1.2, а).

Предельная точка, не являющаяся внутренней точкой, является *граничной точкой множества X* (точка *V* на рис. 1.2, а).

Точка *Q* называется *крайней точкой* выпуклого множества *X* тогда и только тогда, когда не существует различных точек $X^1=(x_{11},x_{21},...,x_{n1})$ и $X^2=(x_{12},x_{22},...,x_{n2})$ из этого множества, для которых $Q=kX^1+(1-k)X^2$, $0<k<1$. Важным является то, что неравенство для *k* является строгим, ибо в противном случае крайнюю точку можно было бы представить в виде выпуклой комбинации ее самой и какой-нибудь другой точки. Для двухмерного множества это означает, что для крайней точки множества нельзя подобрать какие-либо две точки множества, на линии соединения которых находилась бы крайняя точка. Например, множество $X=\{X: x_1+x_2\le 2, x_1\ge 0, x_2\ge 0\}$ имеет три крайние точки: $X^1=(0,0)$, $X^2=(2,0)$, $X^3=(0,2)$ (см. рис. 1.2, а).

Открытое множество состоит только из внутренних точек. *Замкнутое множество* содержит все свои предельные точки. На рис. 1.2 приведены примеры замкнутых множеств.

Точка является *изолированной точкой* для множества *X*, если у нее есть окрестность, не содержащая точек множества *X*. *Дискретное множество* содержит только изолированные точки. Например, если бы для множеств на рис. 1.2 было бы наложено условие целочисленности управлений x_1, x_2, то они были бы уже дискретными.

Множество *X ограниченно*, если ограниченны значения Декартовых координат $x_1, x_2, x_3,...,x_n$ его точек. В этом случае множество *X* является и *замкнутым*.

Множество *X* является односвязным, если любую простую замкнутую поверхность из точек множества можно с помощью непрерывной деформации стянуть в точку, не выходя из множества *X*. Если множество *X* содержит изолированную точку или замкнутое множество, не принадле-

жащее X, то оно является *двухсвязным*. Если таких точек или замкнутых множеств в множестве X много, то оно является многосвязным.

На рис. 1.3 показано трехсвязное множество X. Множества A и B не входят в множество X. Если исключить множество B, то X станет двухсвязным множеством или двухсвязной областью значений X. Границы множества в общем случае могут иметь разную форму в зависимости от видов ограничений на управления. При линейных ограничениях множество допустимых управлений является симплексом. *Симплексом* в n-мерном пространстве называется выпуклый многогранник, имеющий $n+1$ вершину. Так, например, при $n = 2$ симплексом является треугольник, при $n = 3$ — тетраэдр и т.д.

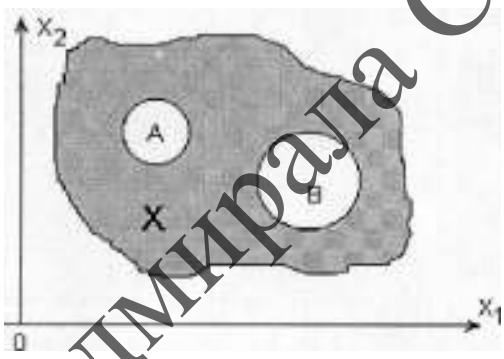

Рисунок 1.3. Трехсвязное множество X

В задачах оптимального управления множество допустимых управлений обычно является односвязным, выпуклым и в зависимости от ограничений — открытым или замкнутым (ограниченным).

Важной характеристикой множества является оценка близости двух произвольных точек множества. Фактически речь идет об оценке величины (длины) вектора, соединяющего две точки множества.

В общем случае эта оценка называется *нормой вектора*. Для вектора X нормой является скалярная функция вектора X (обозначается $\|X\|$), т.е. функция, дающая число. Эта функция от вектора должна удовлетворять следующим условиям:

1. $\|X\| > 0, для\ x > 0, \|X\| = 0, для\ x = 0$;
2. $\|X^1 + X^2\| \le \|X^1\| + \|X^2\|$;
3. $\|a \cdot X\| = |a| \cdot \|X\|$, где а — скаляр. (1.21)

Введенный ранее модуль вектора (или Евклидово расстояние) есть одна из возможных норм вектора X в R^n, которая определяется формулой (см. рис. 1.3):

$$\|X\| = \sqrt{(X \cdot X)} = |X| = \sqrt{\sum_{i=1}^{n} x_i^2}. \quad (1.22)$$

Примерами других норм вектора *X* могут служить:

$$\|X\| = \sum_{i=1}^{n} |x_i|, \quad (1.23)$$

$$\|X\| = max|x_i|, 1 \le i \le n. \quad (1.24)$$

1.3. Функция цели и ее характеристики

Величина $y = z(X)$ называется функцией от *X*, если для каждого значения *x* она принимает одно (*однозначная* функция) или несколько значений (*многозначная* функция). Это значение функции в общем случае может быть *скалярным* или *векторным*. Соответственно, множество значений функции образует *скалярное* или *векторное поле*. Например, если функция цели определяет доходы, расходы, прибыль, рентабельность и тому подобные скалярные величины, то множество ее возможных значений образует скалярное поле. Если функция цели определяет, например, скорость перемещения грузов, т.е. ее значение при каждом наборе управлений является вектором (в каком направлении и с какой величиной скорости перемещаются грузы), то множество ее значений является векторным полем. Обыч-

но для задач оптимизации на транспорте приходится иметь дело со скалярными функциями. Поэтому в дальнейшем по умолчанию будем подразумевать именно этот тип функций цели.

Величина X в общем случае может быть вектором с компонентами x_1, x_2, x_3, ..., x_n. Поэтому скалярная функция $z(X)$ имеет векторный аргумент и по сути является функцией n переменных. При $n=1$ функция $z(X)=z(x_1)$ является функцией одной переменной. График функции одной переменной в системе координат z, x_1 представляется линией, как это показано на рис.1.4. При $n=2$ график функции $z(X)$ в системе координат z, x_1, x_2 представляет поверхность (рис.1.5). График функции n переменных в системе координат z, x_1, x_2, x_3, ..., x_n также изображается поверхностью, которую при $n>2$ называют *гиперповерхностью*.

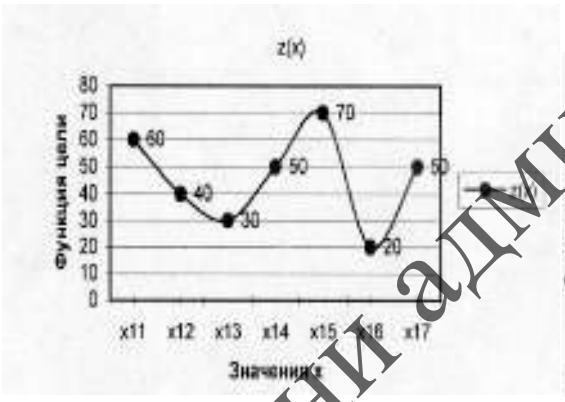

Рисунок 1.4. График функции одной переменной

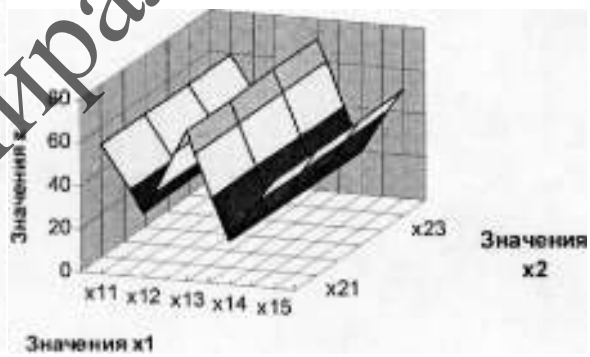

Рисунок 1.5. График функций двух переменных

В задачах оптимального управления часто рассматривают систему координат x_1, x_2, x_3, ..., x_n. Поскольку значения этих координат определяют состояния или фазы управляемой системы, то их называют *фазовыми координатами*, а множество значений фазовых координат — *фазовым пространством*. Каждому конкретному значению функции $z(X)=y_k=const$ в фазовом пространстве будет соответствовать своя кривая, называемая *фазовой траекторией* или *линией уровня* (поверхностью уровня при $n>2$), уравнение которой $y_k-z(X)=0$ или $z(X)=const$.

Область задания функции — это множество значений *X*, для которых определено принятое аналитическое выражение $z(X)$, т.е. принятая формула для записи функции.

Область допустимых управлений — это та часть области задания функции цели, точки которой удовлетворяют ограничениям задачи. В общем случае эта область называется *областью существования функции*, которая при отсутствии ограничений на аргументы совпадает с областью задания функции.

При поиске оптимальных управлений весьма важно, как функция себя ведет в области допустимых управлений. Характеристиками поведения функции могут быть следующие.

Функция *непрерывна* в области, если для любой точки

$$X^* = \left(x_1^*, x_2^*, ..., x_n^*\right) \text{ области } \lim_{X \to X^*} F(X) = F\left(X^*\right).$$

Для непрерывной функции повторный предел в точке X^* равен пределу функции в этой точке, т.е. $\lim_{x_1 \to x_1^*} \lim_{x_2 \to x_2^*} \lim_{x_3 \to x_3^*} \cdots \lim_{x_n \to x_n^*} F(X) = F\left(X^*\right)$ при любом порядке взятия пределов.

Функция $z(X)$ называется *выпуклой*, если для любых двух точек $X^1 = \left(x_{11}, x_{21}, ..., x_{n1}\right)$ и $X^2 = \left(x_{12}, x_{22}, ..., x_{n2}\right)$, принадлежащих R^n, и числа $0 \le p \le 1$ выполняется неравенство:

$$p \cdot z\left(X^1\right) + (1-p) \cdot z\left(X^2\right) \ge z\left(p \cdot X^1 + (1-p) \cdot X^2\right) \text{ — } \textit{выпукла вниз}$$

или

$$p \cdot z\left(X^1\right) + (1-p) \cdot z\left(X^2\right) \le z\left(p \cdot X^1 + (1-p) \cdot X^2\right) \text{ — } \textit{выпукла вверх}.$$

Иногда для упрощения выкладок или лучшего понимания понятия выпуклости принимают $p = 0{,}5$. Тогда:

$$\frac{z\left(X^1\right) + z\left(X^2\right)}{2} \ge z\left(\frac{X^1 + X^2}{2}\right) \text{ — } \textit{выпукла вниз,}$$

$$\frac{z\left(X^1\right)+z\left(X^2\right)}{2} \le z\left(\frac{X^1+X^2}{2}\right)$$ — *выпукла вверх*.

Приведенные соотношения геометрически означают, что гиперповерхность $z(X)$ выпукла вверх, если отрезок, соединяющий любые две точки $\left(X^1, z\left(X^1\right)\right)$, $\left(X^2, z\left(X^2\right)\right)$ лежит на этой гиперповерхности или ниже ее. При выпуклости вниз этот отрезок лежит на гиперповерхности или выше ее.

Например, функция одной переменной на рис. 1.4 в области задания функции $[x_{11}, x_{17}]$ не является выпуклой. Однако в областях задания $[x_{11}, x_{15}]$, $[x_{13}, x_{16}]$ или $[x_{15}, x_{17}]$ функция будет уже выпуклой.

Во многих случаях проверка функции на выпуклость облегчается благодаря следующему утверждению: *сумма выпуклых множеств на одном и том же множестве является выпуклой функцией*.

Функция $z(X)$ является *монотонной*, если для двух произвольных точек X^1 и X^2 таких, что $X^1 \le X^2$, выполняется:

$z\left(X^1\right) \le z\left(X^2\right)$ (монотонно возрастающая функция),

$z\left(X^1\right) \ge z\left(X^2\right)$ (монотонно убывающая функция).

Непрерывная функция, заданная в связной области, обладает свойствами, утверждаемыми следующими теоремами.

Прохождение через нуль (теорема Коши). Если функция в двух точках X^1 и X^2 области задания имеет разные знаки, то найдется по меньшей мере одна третья точка X^3 в этой области, в которой функция $z(X)$ обращается в нуль, т.е. $z\left(X^3\right)=0$, если $z\left(X^1\right)>0$ и $z\left(X^2\right)<0$.

Теорема о промежуточном значении. Если функция в двух точках X^1 и X^2 области задания имеет разные значения A и B, то каково бы ни было число C, лежащее между A и $B\left(A<C<B \text{ или } B<C<A\right)$, в области найдется по меньшей мере одна такая точка X^3, в которой $z\left(X^3\right)=C$.

Теорема о наибольшем и наименьшем значении (теорема Вейерштрасса). Если непрерывная функция $z(X)$ определена на непустом замкнутом и ограниченном множестве *X*, то по крайней мере по одному разу принимает на этом множестве наибольшее и наименьшее значение.

Теорема об ограниченности функции. Если функция задана в ограниченной замкнутой области, то она *ограниченна* в этой области, когда существуют два таких числа *m* и *M*, что для любой точки области $m \le z(X) \le M$.

Функция $z(X)$ называется *ограниченной сверху* на множестве *X*, если существует число *M* такое, что $z(X) \le M$ для любого $X \in X$. Функция $z(X)$ называется *ограниченной снизу* на множестве *X*, если существует число *m* такое, что $z(X) \ge m$ для любого $X \in X$. Функция $z(X)$ называется *неограниченной снизу* на множестве *X*, если существует последовательность точек $\{X^k\}$, для которой $\lim\limits_{k\to\infty} z(X^k) = -\infty$, т.е. функция неограниченно убывает.

1.4. Экстремальные значения функции и их определение

Напомним, что экстремальным *(extr)* может быть максимальное *(max)* или минимальное *(min)* значение функции. Рассмотрим определения, связанные с экстремальными значениями функции и конкретно — с минимальным значением, так как всегда от задачи на максимум можно перейти к задаче на минимум (вспомните, как это делается: см. стр. 8).

Точка X^* называется точкой *глобального минимума* $z(X)$ на множестве *X*, если $z(X^*) \le z(X)$ для всех $X \in X$. Значением функции для этой точки является величина $z(X^*) = z^* = \inf . z(X)$, которая называется *наименьшим значением* или *нижней гранью* функции $z(X)$ на множестве *X*. Иногда под нижней гранью функции понимают такое значение $z^* = \inf . z(X)$, при кото-

ром для любого сколь угодно малого $\varepsilon > 0$ найдется точка $X^{**} \in X$ такая, что $z\left(X^{**}\right) > z^* + \varepsilon$.

Точка X^* называется точкой *локального минимума* $z(X)$ на множестве X, если существует такое δ, что $z\left(X^*\right) \le z(X)$ для всех $X \in X$, для которых $\left|X - X^*\right| < \delta$. Это означает, что из всего множества допустимых управлений рассматривается только то множество управлений, которое попадает в шар радиуса δ с центром в точке X^*.

В зависимости от того, является точка X^* внутренней или граничной точкой множества допустимых управлений, различают *внутренний* или *граничный минимум* (максимум) функции в точке X^*.

Обратимся к рис. 1.4 для функции цели $z(X)$ от одной переменной $X = (x_1)$. Если множество допустимых управлений X определено значениями управлений в интервале $[x_{11}, x_{17}]$, то точка $X^* = (x_{16})$ будет точкой глобального минимума. При этом нижней гранью функции будет $z\left(X^*\right) = z^* = \inf . z(x) = 20$.

Если множество допустимых управлений X определено значениями управлений, например, в интервале $[x_{11}, x_{15}]$, то точка $X^* = (x_{13})$ будет уже точкой глобального минимума, т.к. вне этой области управления считаются просто недопустимыми. При этом нижней гранью функции на этом множестве будет $z\left(X^*\right) = z^* = \inf . z(X) = 30$.

Минимальное значение функции цели может достигаться и на границе области. Например, если множество допустимых управлений X определено значениями управлений в интервале $[x_{11}, x_{12}]$, то точка $X^* = (x_{12})$ будет точкой глобального минимума в этой области. При этом нижней гранью функции цели на этом множестве будет $z\left(X^*\right) = z^* = \inf . z(X) = 40$.

Функция $z(X)$ называется *унимодальной* на отрезке $a \le x \le b$ тогда и только тогда, когда она монотонна по обе стороны от единственной на этом интервале оптимальной точки X^*.

Понятно, что если в области допустимых управлений имеются глобальные и локальные точки минимума, то задача методов оптимального управления сводится к поиску управления X^*, соответствующего глобальному минимуму. К сожалению, нет аналитических методов, позволяющих сразу определить точку глобального минимума. Все существующие методы направлены на поиск минимумов в заданной области, из которых затем сравнением приходится выбирать глобальный. Разумеется, дело упрощается, если из физических соображений о сущности задачи удается выделить область управлений, в которой находится единственная точка или даже множество точек, в которых функция цели минимальна.

Важнейшее значение для поиска экстремальных значений функции цели имеет понятие градиента функции $z(X)$. Понятие градиента связано с понятием производной скалярного поля по некоторому заданному направлению, т.е. по направлению некоторого заданного вектора q. Эта производная является вектором и определяется формулой:

$$\frac{\partial z(X)}{\partial q} = \lim_{\varepsilon \to 0} \frac{z(X + \varepsilon \cdot q) - z(X)}{\varepsilon}.$$

Физически она определяет скорость изменения функции по направлению вектора q.

Если в качестве направления вектора q выбрать направление нормали (перпендикуляра при $n=1$) в точке x к поверхности уровня функции в сторону увеличения функции, то в этом направлении скорость изменения (увеличения) функции будет максимальна, а значит, и максимальна по модулю производная от скалярного поля, которая и называется *градиентом*

функции или *градиентом скалярного поля*. Градиент функции $z(X)$ обозначается как $grad\, z(X)$ или $\nabla z(X)$.

Если с каждой координатной осью x_i связать орт ς_i, то выражение для градиента в системе координат $x_1, x_2, x_3, ..., x_n$ примет вид:

$$\nabla z(X) = grad\, z(X) = \sum_{i=1}^{n} \frac{\partial z}{\partial x_i} \cdot \varsigma_i = \left(\frac{\partial z}{\partial x_1}, ..., \frac{\partial z}{\partial x_n} \right) \cdot \varsigma , \qquad (1.25)$$

где $\varsigma = (\varsigma_1, \varsigma_2, ..., \varsigma_n)^T$. Таким образом, *градиентом функции* $z(X)$ называется вектор $\nabla z(X)$, который равен сумме произведений, в каждом из которых производная скалярного поля по направлению *i-й* оси координат, имеющая величину $\partial z / \partial x_i$ и характеризующая скорость изменения функции при изменении *i-й* координаты, умножается на орт ς_i по той же *i-й* координате.

Стационарной точкой функции $z(X)$ называется точка X^*, в которой $\nabla z(X^*) = 0$. Это означает, что скорость изменения функции в стационарной точке по направлению нормали к поверхности уровня функции равна нулю, и в этом направлении значение функции не может увеличиваться или уменьшаться. Требование равенства нулю градиента означает, что его модуль $|grad\, z(X)| = \sqrt{\sum_{i=1}^{n} \frac{\partial z^2}{\partial x_i}}$, определяемый формулой (1.25), должен быть равен нулю. Отсюда следует, что для определения стационарной точки необходимо решить систему уравнений:

$$\frac{\partial z}{\partial x_i} = 0, i = 1, 2, ..., n . \qquad (1.26)$$

Рассмотрим пример. Пусть функция цели зависит от двух управляемых параметров x_1, x_2 и имеет вид: $z(X) = 3 \cdot x_1^3 + 5 \cdot x_1 \cdot x_2^4 + 6$. Тогда для определения стационарной точки X^* нужно найти x_1^*, x_2^* из системы уравнений:

$$\frac{\partial z}{\partial x_1} = 9 \cdot x_1^2 + 5 \cdot x_2^4 = 0,$$

$$\frac{\partial z}{\partial x_2} = 20 \cdot x_1 \cdot x_2^3 = 0.$$

Решением этой системы будет $x_1^* = x_2^* = 0$, т.е. начало координат будет стационарной точкой X^* для рассматриваемой целевой функции.

В общем случае функция может иметь не одну, а несколько стационарных точек, если система уравнений (1.26) имеет несколько решений.

Стационарная точка обычно соответствует *max* или *min* функции, но может соответствовать и точке перегиба, в которой функция не имеет ни *max*, ни *min*. Поэтому условие $\nabla z\left(X^*\right) = 0$ является необходимым, но недостаточным условием экстремума функции в точке X^*.

Для функции одной переменной на рис. 1.4 в области задания $[x_{11}, x_{17}]$ стационарными точками будут x_{13}, x_{15}, x_{16}. Если область задания $[x_{11}, x_{12}]$, то в ней функция вообще не имеет стационарных точек, хотя имеет максимальное (точка x_{11}) и минимальное (точка x_{12}) значение. Поэтому при решении задач оптимизации наличие стационарных точек не является обязательным.

При наличии стационарных точек в области допустимых управлений, естественно, возникает вопрос, в каких стационарных точках функция имеет максимум, в каких минимум, а какие являются точками перегиба. Решение этого вопроса связано с анализом производных второго и более высокого порядка в стационарных точках. Однако для функций многих переменных ограничиваются, как правило, анализом только вторых производных, который позволяет определить, имеется *max* или *min* в стационарной точке.

В основе этого анализа лежит формула для приращения функции Δz в окрестности точки x^*:

$$\Delta z = z\left(x_1^* + \Delta x_1, x_2^* + \Delta x_2, ..., x_n^* + \Delta x_n\right) - z\left(x_1^*, x_2^*, ..., x_n^*\right) = z\left(X^* + \Delta X\right) - z\left(X^*\right).$$

Если функция $z(X)$ в некоторой окрестности точки X^* имеет непрерывные частные производные до второго порядка включительно, то можно записать:

$$\Delta z\left(x^*\right) = \sum_{i=1}^{n}\left[\frac{\partial z}{\partial x_i}\right]_{x=x^*} \cdot \Delta x_i + \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\left[\frac{\partial^2 z}{\partial x_i \partial x_j}\right]_{x=x^*} \cdot \Delta x_i \cdot \Delta x_j + 0\left(\delta^2\right) \quad (1.27)$$

где $\delta = \sqrt{\Delta x_1^2 + \Delta x_2^2 + ... + \Delta x_n^2}$.

Формула (1.27) называется локальной формулой Тейлора. Слагаемое $0\left(\delta^2\right)$ обозначает, что остальные члены формулы имеют более высокий порядок малости и могут не учитываться. Первое слагаемое (1.27) в стационарной точке равно нулю. Второе слагаемое в (1.27) определяет половину дифференциала второго порядка от функции $z(X)$ в точке X^*, который определяется формулой:

$$d^2 z = \sum_{i=1}^{n}\sum_{j=1}^{n}\left[\frac{\partial^2 z}{\partial x_i \partial x_j}\right]_{x=x^*} \cdot \Delta x_1 \cdot \Delta x_j = \Delta X \cdot H\left(X^*\right) \cdot \Delta X \ , \quad (1.28)$$

где $\Delta X'$ обозначает вектор-столбец с теми же элементами $\Delta x_1, \Delta x_2, ..., \Delta x_n$, что и вектор строка ΔX, а матрица $H\left(X^*\right)$ — матрица Гессе.

Матрицей Гессе функции $z(X)$ в точке X^* называется матрица, составленная из вторых производных функции $z(X)$, вида:

$$H\left(X^*\right) = \left[\frac{\partial^2 z}{\partial x_i \cdot \partial x_j}\right]_{x=x^*} ; i, j = 1, 2, ..., n \ . \quad (1.29)$$

Формула (1.28) определяет многочлен относительно $\Delta x_1, \Delta x_2, ..., \Delta x_n$ и называется *квадратичной формой*. Если обозначить элементы матрицы Гессе как:

$$h_{ij} = \frac{\partial^2 z}{\partial x_i \cdot \partial x_j}, \qquad (1.30)$$

то легко заметить, что $h_{ij} = h_{ji}$, т.е. матрица Гессе является симметричной, а значит, симметрична связанная с ней квадратичная форма (1.28).

Рассмотрим пример. Пусть функция цели имеет рассмотренный ранее вид: $z(X) = 3 \cdot x_1^3 + 5 \cdot x_1 \cdot x_2^4 + 6$. Элементы матрицы Гессе для этой функции будут иметь вид:

$$h_{11} = 18 \cdot x_1;\ h_{12} = h_{21} = 20 \cdot x_2^3;\ h_{22} = 60 \cdot x_1 \cdot x_2^2.$$

Обозначим вектор-строку $\Delta X = (\Delta x_1, \Delta x_2)$, а вектор-столбец $\Delta X' = \begin{pmatrix} \Delta x_1 \\ \Delta x_2 \end{pmatrix}$.

Тогда по формуле (1.28) получим:

$$d^2 z = \Delta x \cdot H(x^*) \cdot \Delta x' = (\Delta x_1 \quad \Delta x_2) \cdot \begin{pmatrix} 18x_1 & 20x_2^3 \\ 20x_2^3 & 60x_1x_2^2 \end{pmatrix}_{x=x^*} \cdot \begin{pmatrix} \Delta x_1 \\ \Delta x_2 \end{pmatrix} =$$

$$= \left(\Delta x_1 \cdot 18 \cdot x_1 + \Delta x_2 \cdot 20 \cdot x_2^3 \quad \Delta x_1 \cdot 20 \cdot x_2^3 + \Delta x_2 \cdot 60 \cdot x_1 \cdot x_2^2\right)\Big|_{x=x^*} \cdot \begin{pmatrix} \Delta x_1 \\ \Delta x_2 \end{pmatrix} =$$

$$= \left(\Delta x_1 \cdot 18 \cdot x_1 + \Delta x_2 \cdot 20 \cdot x_2^3 + \Delta x_2 \cdot \Delta x_1 \cdot 20 \cdot x_2^3 + \Delta x_2^2 \cdot 60 \cdot x_1 \cdot x_2^2\right)\Big|_{x=x^*} =$$

$$= \left(18 \cdot x_1 \cdot \Delta x_1^2 + 40 \cdot x_2^3 \cdot \Delta x_1 \cdot \Delta x_2 + 60 \cdot x_1 \cdot x_2^2 \cdot \Delta x_2^2\right)\Big|_{x=x^*} = 0.$$

Последнее выражение показывает конкретный вид квадратичной формы в рассматриваемом примере, которая равна нулю, т.к. для рассматриваемой функции цели стационарная точка $X^* = 0$.

Симметричная квадратичная форма, а также соответствующая ей симметричная матрица называется *положительно определенной*, *отрицательно определенной*, *неотрицательно определенной* или *неположительно определенной*, если, соответственно, $\Delta X \cdot H(X^*) \cdot \Delta X' > 0$, $\Delta X \cdot H(X^*) \cdot \Delta X' < 0$, $\Delta X \cdot H(X^*) \cdot \Delta X' \geq 0$, $\Delta X \cdot H(X^*) \cdot \Delta X' \leq 0$ для каждого набора чисел $\Delta x_1, \Delta x_2, ..., \Delta x_n$, не все из которых равны нулю. Во всех остальных случаях

квадратичная форма является неопределенной или тождественно равной нулю.

Симметричная квадратичная форма, а также соответствующая ей симметричная матрица называется *положительно полуопределенной* или *отрицательно полуопределенной*, если, соответственно, она неположительная или неотрицательная, но $\Delta X \cdot H\left(X^*\right) \cdot \Delta X' = 0$ для некоторых наборов $\Delta x_1, \Delta x_2, ..., \Delta x_n$.

Приведенные определения важны потому, что характер стационарной точки определяется видом квадратичной формы, определяющей второй дифференциал в стационарной точке.

Если второй дифференциал в точке является *положительно определенной* квадратичной формой, то эта точка является точкой *минимума*.

Если второй дифференциал в стационарной точке является *отрицательно определенной* квадратичной формой, то эта точка является точкой *максимума*.

Эти условия являются достаточными условиями *min* или *max* в стационарной точке.

Если второй дифференциал в стационарной точке является *неопределенной* квадратичной формой, то в этой точке функция не имеет ни *max*, ни *min*.

Если второй дифференциал в стационарной точке является *полуопределенной* квадратичной формой, то характер значения функции в стационарной точке зависит от дифференциалов более высокого порядка. При этом функция должна быть достаточное число раз дифференцируемой.

Ввиду произвольного выбора наборов $\Delta x_1, \Delta x_2, ..., \Delta x_n$, определяющих векторы приращений $\Delta X'$ и ΔX, анализ квадратичной формы (1.28), по сути, сводится к анализу матрицы Гессе. Этот анализ основан на определении *собственных чисел* квадратной матрицы $n \times n$, которые являются кор-

нями алгебраического уравнения n-й степени (*характеристического уравнения* для матрицы), определяемого, с учетом обозначений (1.28–1.30), соотношением:

$$F_H(\lambda) = \det(H - \lambda \cdot E) = \det\left[h_{ij} - \lambda \cdot \delta_{ij}\right] = \begin{vmatrix} h_{11-\lambda} & h_{12} & \dots & h_{1n} \\ h_{21} & h_{22-\lambda} & \dots & h_{2n} \\ \dots & \dots & \dots & \dots \\ h_{n1} & h_{n2} & \dots & h_{nn-\lambda} \end{vmatrix} = 0 \text{,} \quad (1.31)$$

где E — единичная матрица размером $n \times n$, элементы которой δ_{ij} равны 1 при $i=j$ и нулю в других случаях.

Матрица $H(X)$ является *положительно определенной* (или *определенно-положительной*), что обозначается как $H(X) > 0$, если все ее собственные числа $\lambda_i, i = \overline{1,n}$ положительны.

Матрица $H(X)$ является отрицательно определенной (или определенно-отрицательной), если ее собственные числа $\lambda_i, i = \overline{1,n}$ отрицательны.

Достаточные условия min или *max* будут формулироваться в виде:

– если в стационарной точке матрица Гессе является *положительно определенной*, то эта точка является точкой *минимума*;

– если в стационарной точке матрица Гессе является *отрицательно определенной*, то эта точка является точкой *максимума*.

Если $n=1$, т.е. функция цели является функцией одной переменной x_1, то матрица Гессе $H(X^*)$ в стационарной точке $X^* = (x_1^*)$ состоит из одного элемента — второй производной от функции по ее аргументу, а второй дифференциал будет равен:

$$d^2 z = \left[\frac{\partial^2 z}{\partial x_1^2}\right]_{x=x^*} \cdot \Delta x_1^2 = H\left(x_1^*\right) \cdot \Delta x_1^2 .$$

На основании (1.31) получим:

$$F_H(\lambda)=\det(H-\lambda\cdot E)=\det\left[h_{11}-\lambda\cdot\delta_{11}\right]=\left|\frac{\partial^2 z}{\partial x_1^2}-\lambda\right|_{x_1=x_1^*}=0 \text{ или } \lambda_1=\left|\frac{\partial^2 z}{\partial x_1^2}\right|_{x_1=x_1^*} \quad (1.32)$$

Из только что приведенных выше определений для квадратичной формы и связанной с ней матрицы Гессе следует, что стационарная точка является точкой минимума, если единственное собственное число $\lambda_1>0$, а это значит, что $H(x^*)>0$. При $H(X^*)<0$ стационарная точка будет точкой максимума. Если $H(X^*)=0$, то необходимо исследовать производные более высокого порядка. Это, естественно, совпадает с известными из математического анализа результатами об условиях экстремума функции одной переменной: при равенстве нулю первой производной функция имеет *max* при отрицательном значении второй производной в той же точке и *min* — при положительном значении второй производной в той же точке.

Если *n* достаточно велико, то определить корни характеристического уравнения матрицы Гессе не так просто. Заметим, что в задачах оптимизации важны не сами значения собственных чисел, а прежде всего их знаки.

Для того чтобы матрица Гессе была положительно определенной, т.е. $H(X)>0$, необходимо и достаточно, чтобы каждый из диагональных (угловых) определителей (миноров):

$$h_{11},\begin{vmatrix}h_{11} & h_{12}\\ h_{21} & h_{22}\end{vmatrix},\begin{vmatrix}h_{11} & h_{12} & h_{13}\\ h_{21} & h_{22} & h_{23}\\ h_{31} & h_{32} & h_{33}\end{vmatrix},\ldots,\det\left[h_{ij}\right] \quad (1.33)$$

был положителен.

Таким образом, можно сформулировать следующее правило, известное как *критерий Сильвестра*.

Составляется определитель матрицы Гессе в точке X^* и вычисляются все его диагональные миноры. Точка X^* есть точка локального минимума, если все диагональные миноры положительны.

Для отрицательности матрицы Гессе, т.е. $H(X)<0$, нечетные угловые миноры, определяемые формулой (1.33), должны быть отрицательны, а четные — положительны.

Рассмотрим другие характеристики функции, связанные с градиентом и матрицей Гессе, которые важны при поиске точки экстремума функции.

Гладкая функция $z(X), X \in R^n$, *выпукла* тогда и только тогда, когда для любых X, X^* имеет место $z(X)-z(X^*)\geq\left((X-X^*)\cdot\Delta z(X^*)\right)$.

Дважды дифференцируемая $z(X), X \in R^n$, строго выпукла тогда и только тогда, когда $H(X)>0$ или $H(X)<0$. Это означает, что $z(X)$ имеет только один *min* или только один *max* для всех $X \in R^n$.

Функция $z(X)$ *называется сильновыпуклой*, если ее матрица Гессе $H(X)$ удовлетворяет условию $m\cdot\|Y\|^2 \leq <Y,H(X),X> \leq M\cdot\|Y\|^2$ для всех $X,Y \in R^n$, где $M\geq m\geq 0$ — некоторые числа. Обозначение $<Y,H(X),X>$, которое иногда используется в литературе, соответствует скалярному произведению $Y\cdot H(X)\cdot X$.

Глава 2. Линейное программирование

Задачи математического программирования, в которых целевая функция и ограничения на область допустимых решений (управлений) являются линейными относительно управлений, называются *задачами линейного программирования (ЗЛП)*. Для задач этого типа математический аппарат их решения наиболее разработан, легко поддается алгоритмизации и реализации на ЭВМ. Задачи линейного программирования имеют наибольшее распространение еще и потому, что математическое описание задач оптимизации экономических систем часто в наибольшей степени соответствует линейной модели их функционирования.

Например, если управляемыми переменными являются объемы выпускаемой продукции разной номенклатуры, то при фиксированной цене за единицу продукции каждого вида объем выручки за ее реализацию пропорционален объему продукции, расход ресурсов на производство продукции пропорционален ее объему, стоимость ресурсов при фиксированной цене за единицу ресурса пропорциональна объему ресурса и т.п.

Линейность целевой функции и ограничений означает пропорциональную зависимость их значений от значений управления. Когда такая пропорциональность имеет место, можно надеяться на успешное применение методов линейного программирования.

В качестве примера рассмотрим следующую оптимизационную задачу. Составим математическое описание задачи закрепления судов за линиями перевозок. Формулировка задачи состоит в следующем.

Имеется n типов судов, которые нужно закрепить за m линиями перевозок. Месячный объем перевозок (иначе — провозная способность) судна *j-го* типа ($j = 1, 2, \ldots, n$) на *i-й* линии ($i = 1, 2, \ldots, m$) равен a_{ij} единиц, а связанные с этим затраты — C_{ij} рублей. Определить число x_{ij} судов *j-го* типа, закрепленных за *i*-линией для обеспечения перевозки по этой линии

не менее b_i единиц груза при минимальных суммарных расходах на перевозки, если известно, что число судов каждого типа равно N_j, $j = 1...n$.

Математическая модель этой задачи:

$$\sum_{j=1}^{n} a_{ij} x_{ij} = b_i; i = 1,...,m;$$

$$\sum_{i=1}^{m} x_{ij} \le N_j; j = 1,...,n;$$

$$z(x) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} x_{ij} \to min;$$

$$x_{ij} \ge 0 \, целые \, числа; i = 1,2,...,m; \quad j = 1,2,...,n.$$

В приведенной формулировке $x_{ij} \ge \forall i, j$ записана задача целочисленного программирования. Если отбросить условия целочисленности, то получится задача линейного программирования.

Чтобы свести поставленную задачу к задаче линейного программирования, изменим ее постановку. Выберем в качестве управляемых параметров величины x_{ij}, которые будут теперь определять не количество судов, а долю навигационного периода, в течение которого судно *j-го* типа должно отработать на *i-й* линии. Если обозначим через T продолжительность периода работы линии, то величина $x_{ij} \cdot T$ будет определять продолжительность работы судна *j-го* типа на *i-й* линии. Величина же $N_j \cdot T$ будет определять общую максимальную продолжительность работы всех судов *j-го* типа.

С учетом новой постановки задачи математическая модель ее будет иметь вид:

$$z(x) = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} x_{ij} \cdot T \to min$$

при

$$\sum_{j=1}^{n} a_{ij} x_{ij} \cdot T \geq b_i; i = 1,...,m;$$

$$\sum_{i=1}^{m} x_{ij} \cdot T \leq N_j \cdot T; j = 1,...,n;$$

$$x_{ij} \geq 0 \, целые \, числа; i = 1,2,...,m; j = 1,2,...,n.$$

Это будет уже задача линейного программирования. В качестве значения *T* можно принять, например, единицу — одна навигация *(T = 1)*. Тогда целая часть x_{ij} будет определять, сколько судов *j-го* типа должно отработать за навигацию на *i-й* линии, а дробная часть — какую часть навигации должно отработать судно *j-го* типа на *i-й* линии.

2.1. Формы записи задач линейного программирования

При составлении математической модели оптимизируемой экономической системы ограничения на область допустимых решений (управлений) могут оказаться сформулированными в разных формах. Поэтому по виду формы записи ограничений говорят о разных формах записи ЗЛП. Как будет показано ниже, всегда можно перейти от одной формы к другой. Само определение форм неоднозначно у разных авторов, но важно понимать наличие таких форм, так как некоторые программные средства реализации ЗЛП требуют представления этих задач в определенной форме.

Общая (смешанная) ЗЛП

Общая форма ЗЛП характеризуется наличием ограничений типа равенств и неравенств. Целевая функция и ограничения суть линейные функции переменных $x_i, i = 1,2,..,n$:

$$z(x) = \sum_{j=1}^{n} c_j x_j \rightarrow max;$$

$$\sum_{j=1}^{n} a_{ij}x_j = b_i; i = 1,2,...,l; \sum_{j=1}^{n} a_{ij}x_j \le b_i; i = l+1, l+2,...,m;$$

$$x_j \ge 0, j = 1,2,...,n.$$

Основная (стандартная) ЗЛП

Задача характеризуется наличием ограничений только типа неравенств. Математическая формулировка ее имеет вид:

$$z(x) = \sum_{j=1}^{n} c_j x_j \to max; \sum_{j=1}^{n} a_{ij}x_j \le b_i; i = 1,...,m; X_j \ge 0; j = 1,...,n.$$

Примером такой задачи может служить задача о максимальной прибыли предприятия.

Для изготовления каждого из *n* видов продукции употребляется *m* видов сырья, причем расход *i-го* вида сырья на единицу *j-го* вида продукции составляет a_{ij} единиц. Прибыль на единицу продукции *j-го* вида составляет C_j рублей. Определить, сколько единиц $x_j\left(j = \overline{1,n}\right)$ продукции каждого вида следует изготовить предприятию при условии получения максимальной прибыли, если в его распоряжении имеется $b_i\left(i = \overline{1,m}\right)$ единиц сырья каждого вида. Математическая модель задачи имеет вид, приведенный выше в этом параграфе.

Каноническая ЗЛП

Задача характеризуется наличием ограничений только типа равенств. Примером такой задачи может служить задача о назначениях.

Имеются m механизмов для выполнения n работ. При этом каждый из механизмов должен выполнять одну и только одну работу. Производительность *j-го* механизма на *i-й* работе равна C_{ij}. Требуется распределить механизмы по работам из условий максимальной суммарной производительности. Математическая модель задачи:

$$z(x)=\sum_{i=1}^{n}\sum_{j=1}^{n}c_{ij}x_{ij}\to max;\sum_{j=1}^{n}x_{ij}=1;i=1,...,m;\sum_{i=1}^{n}x_{ij}=1;j=1,...,n.$$

Всегда имеется возможность произвести переход от одной формы записи ЗЛП к другой. Этот переход связан с добавлением или исключением переменных задачи. Рассмотрим в общем виде основную и соответствующую ей каноническую задачи.

Основная	Каноническая
$z(x)=\sum_{j=1}^{n}c_jx_j\to max$	$z(x)=\sum_{j=1}^{n}c_jx_j\to max$
$\sum_{j=1}^{n}a_{ij}x_j\le b_i;i=1,...,m;$	$\sum_{j=1}^{n}a_{ij}x_j+x_{n+i}=b_i;i=1,...,m;$
$x_j\ge 0;j=1,...,n;$	$x_j\ge 0;j=1,...,n,n+1,n+2,...,n+m;$
$b_i\ge 0;i=1,..,m.$	$b_i\ge 0;i=1,..,m.$

Как видно, преобразование основной задачи в каноническую свелось к добавлению в левой части каждого *i-го* ограничения типа $\le$ дополнительной переменной $x_{n+i}\ge 0$, которая превращает неравентсво в равенство. Так как вводимые переменные являются фиктивными, то они не должны изменять значение целевой функции и потому имеют в целевой функции нулевой коэффициент.

Рассмотрим числовой пример.

Общая ЗЛП	Каноническая ЗЛП
$z(x)=x_1-x_2\to max;$	$z(x)=x_1-x_2\to max;$
$x_1+x_2\ge 2;$	$x_1+x_2-x_3=2;$
$3\cdot x_1-5\cdot x_2=6;$	$3\cdot x_1-5\cdot x_2=6;$
$x_1,x_2\ge 0.$	$x_1,x_2,x_3\ge 0.$

Если теперь из приведенной в примере канонической формы исключить переменную x_3, то получим снова общую форму ЗЛП.

Напомним здесь также, что задачу на максимум всегда можно свести к эквивалентной задаче на минимум, так как $\max z(x) = \min\{-z(x)\}$. Поэтому вместо задачи с критерием оптимальности $z(x) \to max$ можно решать задачу с критерием оптимальности $f(x) = -z(x) \to min$.

2.2. Виды записи задач линейного программирования

Скалярный вид записи в ЗЛП:

$$z(x) = \sum_{j=1}^{n} c_j x_j \to max; \sum_{j=1}^{n} a_{ij} x_j \le b_i; i = 1,...,m; x_j \ge 0; j = 1,...,n.$$

Матричный вид записи ЗЛП:

$$z(x) = c \cdot x^T \to max; при\ A \cdot x^T \le b^T; x \ge 0,$$

где $c = (c_1 ... c_n)$ — строка коэффициентов целевой функции;

$x^T = (x_1 ... x_n)^T$ — столбец искомых переменных;

$A = [a_{ij}]$ — матрица размером $m \times n$;

$b^T = (b_1 ... b_m)^T$ — столбец правых частей ограничений.

Здесь и далее символ транспонирования ^{T} означает, что, например,

$$x^T = (x_1, x_2, \ldots, x_n)^T = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}.$$

Векторный вид записи в ЗЛП:

$$z(x) = \sum_{j=1}^{n} c_j x_j \to max\ при \sum_{j=1}^{n} P_j x_j \le P_0,$$

где $P_j = (a_{1j},...,a_{mj})^T, j = 1,...,n$ — векторы-столбцы из коэффициентов ограничений;

$P_0 = (b_1 ... b_m)^T$ — столбец правых частей ограничений;

$x_1, x_2, x_3, ..., x_n$ — управляемые переменные.

2.3. Двойственные задачи линейного программирования

К каждой задаче линейного программирования, которую будем называть исходной или прямой, может быть сопоставлена соответствующая ей двойственная задача линейного программирования. Пусть прямая задача линейного программирования сформулирована в виде:

$$z_{П}(x) = c \cdot x^T \to max;$$
$$A \cdot x^T \le b^T;$$
$$x \ge 0,$$

где $c = (c_1 ... c_n)$ — вектор-строка коэффициентов целевой функции;

$x^T = (x_1 ... x_n)^T$ — вектор-столбец исходных переменных;

$A = [a_{ij}]$ — матрица коэффициентов ограничений размером $m \times n$;

$b^T = (b_1 ... b_m)^T$ — вектор-столбец правых частей ограничений.

Тогда соответствующая ей двойственная задача линейного программирования запишется в виде:

$$z_{Д}(u) = b \cdot u^T \to min;$$
$$A^T u^T \ge c^T;$$
$$u \ge 0,$$

где A^T — транспонированная матрица коэффициентов ограничений исходной задачи,

$u^T = (u_1 ... u_m)^T$ — вектор-столбец двойственных переменных.

Пусть, например, прямая задача имеет следующий конкретный вид:

$$z_{П}(x) = 10 \cdot x_1 - 20 \cdot x_2 + 30 \cdot x_3 \to max;$$
$$3 \cdot x_1 + 4 \cdot x_2 - 5 \cdot x_3 \le 200;$$
$$6 \cdot x_1 - 7 \cdot x_2 + 8 \cdot x_3 \le 300;$$

$$x_1, x_2, x_3 \geq 0.$$

Тогда соответствующая ей двойственная задача запишется в виде:

$$z_Д(u) = 200 \cdot u_1 + 300 \cdot u_2 \to min;$$

$$3 \cdot u_1 + 6 \cdot u_2 \geq 10;$$

$$4 \cdot u_1 - 7 \cdot u_2 \geq -20;$$

$$-5 \cdot u_1 + 8 \cdot u_2 \geq 30;$$

$$u_1, u_2, u_3 \geq 0.$$

На основе общей формулировки и приведенного примера можно было бы сформулировать словесное описание правил построения двойственной задачи типа: число двойственных переменных равно числу ограничений в исходной задаче и т.д.

Теоремы связи прямой и двойственной ЗЛП.

Теорема 1.

Прямая и двойственная ЗЛП имеют оптимальные решения в том и только в том случае, если они имеют допустимые решения.

Теорема 2.

Допустимый вектор x^* оптимальный в том и только в том случае, если в двойственной задаче имеется такое допустимое решение u^* (оно является и оптимальным), при котором

$$x^* \cdot c^T = u^* \cdot b^T,$$

т.е. иначе:

$$z_n(x^*) = \sum_{i=1}^{n} c_i x_i^* = z_Д(u^*) = \sum_{j=1}^{m} b_j u_j^*.$$

Теорема 3.

В любой ЗЛП, имеющей единственное оптимальное решение x^*, каждому ограничению, для которого

$$\sum_{j=1}^{n} a_{ij} x_j^* = b_i; i = 1,...,m$$

соответствует $u^* > 0$.

Теорема 4.

В любой двойственной ЗЛП, имеющей единственное оптимальное решение u^*, каждому ограничению, для которого

$$\sum_{i=1}^{m} a_{ji} u_j^* = c_{ji}; j = 1,...,n$$

соответствует $x^* > 0$.

Использование понятия двойственной задачи линейного программирования с практической точки зрения позволяет уменьшить число искомых переменных при решении задачи.

С экономической точки зрения, если исходная задача связана с поиском оптимального объема производимой продукции при ограничениях на ресурсы, то соответствующая ей двойственная задача связана с поиском оптимальной условной (теневой) цены u_j каждого b_j-*го* ресурса при производстве продукции в объеме большем или равным заданному. Из последнего соотношения теоремы 2 следует, что оптимальные значения u_j^* можно рассматривать как коэффициенты при имеющихся ресурсах b_j, которые показывают, насколько изменится целевая функция при изменении соответствующего ресурса на единицу.

Понятие двойственной задачи имеет также огромное значение для теоретических исследований построения алгоритмов решения задач линейного программирования, на свойствах которых в конечном счете они и строятся.

2.4. Методы решения задач линейного программирования

Все множество методов решения задач линейного программирования можно разделить на следующие группы:

1. Графические (геометрические) методы. Они очень просты, имеют малую трудоемкость, большую наглядность, но могут быть использованы только тогда, когда число переменных равно двум (максимум трем для объемного изображения).
2. Точные (строгие) методы. Они позволяют получать оптимальное решение. Перечислим основные из них.

   Распределительный метод (метод потенциалов для транспортной задачи). Здесь все исходные данные и вспомогательные величины записываются в виде матрицы. Затем по определенному алгоритму производится перемещение исходного (начального) плана по строкам и столбцам матрицы, и в результате нескольких приближений (интераций) достигается оптимальное решение (план).

   Преимущества: простота и малая трудоемкость; возможность реализации на компьютере.

   Недостаток: при использовании этого метода необходимо, чтобы все величины, входящие в план, имели одну и ту же размерность.

   Метод разрешающих множителей. Он предназначен для составления планов наилучшего использования оборудования. Здесь оптимальный план находится с помощью специальных оценочных чисел (разрешающих множителей).

   Симплекс-метод. Наиболее универсальный метод решения задач линейного программирования, с его помощью можно решать многие экономические задачи.
3. Приближенные методы. Их существует более десяти. С их помощью можно получить оптимальное или близкое к оптимальному решение. Наиболее часто используются:

– метод индексов;
– метод простейших аппроксимаций;
– метод круговых разностей.

Методы линейного программирования позволяют решать широкий круг задач коммерческо-организационной деятельности, таких как:

– планирование товарооборота;
– прикрепление торговых предприятий к поставщикам;
– организация рациональных схем доставки товаров (транспортная задача);
– распределение работников по должностям;
– распределение ресурсов;
– планирование капиталовложений;
– замена оборудования;
– определение оптимального ассортимента товаров в условиях ограничения торговых площадей, —

и ряд других задач.

2.5. Геометрическая интерпретация задач линейного программирования

Для наглядности изложения предположим, что $n-r=2$, т.е. имеется две свободные переменные x_1 и x_2. Тогда формулу (1.13а) можно записать в виде:

$$\begin{aligned} x_3 &= \beta_3 + \alpha_{31}\cdot x_1 + \alpha_{32}\cdot x_2 \\ x_4 &= \beta_4 + \alpha_{41}\cdot x_1 + \alpha_{42}\cdot x_2 \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ x_n &= \beta_n + \alpha_{n1}\cdot x_1 + \alpha_{n2}\cdot x_2 \end{aligned} \qquad (2.1)$$

Теперь можно дать геометрическую интерпретацию задаче линейного программирования. Каждому уравнению из формулы (2.1) в плоскости

переменных x_1, x_2 при $x_{n-r+1}=0$, $x_{n-r+2}=0$, $x_n=0$ будет соответствовать прямая линия. Для точек множества (x_1,x_2) с одной стороны от этих линий будут значения $x_{n-r+i}>0$, а с другой стороны от этих линий будет $x_{n-r+i}<0$.

По условию постановки задачи допустимой областью значений x_1,x_2 будет та сторона от линии $x_{n-r+i}=0$, в которой $x_{n-r+i}>0$. В какой стороне от линии $x_{n-r+i}=0$ будет выполняться это условие, зависит от коэффициентов β_{n-r+i}, $\alpha_{n-r+i,1}$ и $\alpha_{n-r+i,2}$. Проще всего эту сторону определить по направлению градиента для x_{n-r+i} как функции x_1 и x_2. Для этого нужно в любой точке прямой $x_{n-r+i}=0$ построить вектор градиента для функции $x_{n-r+i}=f_{n-r+i}(x_1,x_2)$ (см.1.25):

$$\nabla f_{n-r+i}(x_1,x_2)=\frac{\partial f_{n-r+i}(x_1,x_2)}{\partial x_1}\cdot\varsigma_1+\frac{\partial f_{n-r+i}(x_1,x_2)}{\partial x_2}\cdot\varsigma_2,$$

где ς_1 и ς_2 — единичные орты по направлению, соответственно, осей x_1 и x_2 в выбранной точке на прямой $x_{n-r+i}=f_{n-r+i}(x_1,x_2)=0$. Область, в которую направлен вектор градиента, будет соответствовать области значений x_1,x_2, где $x_{n-r+i}>0$.

Допустимой областью решений будет та область, в которой значения всех переменных больше или равны нулю. Понятно, что эта область при двух свободных переменных x_1 и x_2 всегда может находиться только в первом квадранте $(x_1\ge 0, x_2\ge 0)$. Она может быть неограниченной или замкнутой, как это показано для примера на рис. 2.1 при двух свободных переменных и трех базисных.

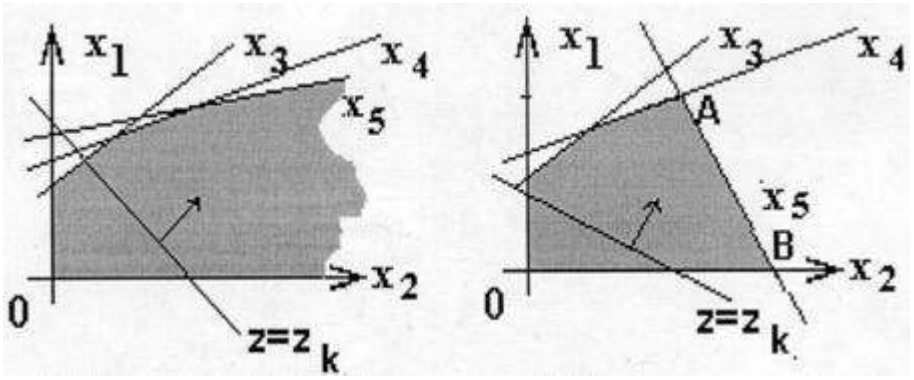

Рисунок 2.1. Области допустимых значений:

а) неограниченная область; б) ограниченная область

Область допустимых управлений может не существовать вовсе, если нет множества точек, соответствующих свободным переменным, для которых выполняются ограничения на управляемые переменные.

На рис. 2.1 показаны линии функции цели. Предположим, что решается задача на максимум функции цели. Чтобы увеличить значение z_k функции цели, нужно перемещать функцию цели параллельно самой себе в сторону градиента. Для варианта *а)* на рис. 2.1 такое перемещение можно делать бесконечно, т.к. при $z_k \to \infty$ найдутся точки (x_1, x_2), принадлежащие области допустимых решений. Для варианта *б)* на рис. 2.1 такое перемещение возможно только до точки пересечения линий $x_4 = 0, x_5 = 0$ (точка А на рисунке), т.к. при дальнейшем перемещении линии функции цели уже не будет точек (x_1, x_2), принадлежащих области допустимых решений. Если бы линия функции цели была бы параллельна линии ограничения $x_5 = 0$, то одно и то же наибольшее значение функции цели достигалось бы в любой точке на линии $x_5 = 0$ между точками А и В.

Если бы решалась задача на минимум функции цели, то линию функции цели $z(x_1, x_2) = z_k = const$ нужно перемещать в сторону, противоположную градиенту. В обоих случаях для рис.2.1 минимум функции цели будет достигаться в точке $x_1 = 0, x_2 = 0$.

Из геометрического представления задачи линейного программирования можно сделать следующие важные выводы.

Область допустимых управлений может быть, а может и не быть. Если она существует, то всегда является выпуклой, т.к. границы ее определяются прямыми линиями, соответствующими линейным ограничениям задачи. Выпуклый многогранник, соответствующий области допустимых решений, называется симплексом. Для каждой линии ограничений область допустимых управлений лежит по одну сторону от нее.

Если область является незамкнутой, то оптимальное решение может быть, а может и не быть. Оптимальное решение, если оно существует, всегда достигается на границе области. Это можно показать и математически, т.к. система уравнений (см. формулу (1.26)), определяющая стационарную точку, в этом случае не имеет решения.

Если линия функции цели совпадает с линией границы, то задача имеет бесчисленное множество решений для одного и того же оптимального значения функции цели.

Поскольку оптимальное значение функции цели достигается на границе, то решение задачи линейного программирования фактически сводится к поиску вершин многогранника ограничений (симплекса) и выбору той из них, в которой функция цели имеет оптимальное (наибольшее в задачах на *max* или наименьшее в задачах на *min*) значение. При большой размерности задачи, т.е. при большом количестве ограничений и переменных, полный перебор вершин представляется слишком трудоемким. Поэтому задача методов линейного программирования состоит в том, чтобы сделать этот перебор целенаправленным и избежать перебора всех вершин симплекса.

2.6. Симплекс-метод. Построение первоначальной симплекс-таблицы

В математическую постановку задачи входит построение ограничений и целевой функции.

Как и другие методы оптимизации, симплекс-метод основан на последовательном приближении к оптимальности. Процедура симплекс-метода включает 3 существенных элемента:

- указывается способ нахождения исходного (опорного) плана;
- устанавливается признак, дающий возможность проверить, является ли допустимый план оптимальным;
- формулируются правила, по которым неоптимальный план можно улучшить.

Решение задачи линейного программирования симплекс-методом получается не аналитическим путем, т.е. не с помощью формул, позволяющих вычислить оптимальный план через ограничения и целевую функцию, что здесь и невозможно, а решение получается алгоритмически, шаг за шагом — итерационно.

Особенность метода состоит в том, что составление первоначального плана основывается на понятии «базиса» — совокупности линейно независимых векторов.

Основные теоремы линейного программирования

Множество допустимых планов задач линейного программирования является выпуклым (выпуклое множество — множество, которое вместе с двумя $\forall$ точками содержит и отрезок, содержащий эти точки).

Если задача линейного программирования разрешима, то оптимальное значение целевой функции достигается, по крайней мере, в одной из вершин многогранника ограничений.

Первый подход к поиску базиса

В случае, если в ограничениях задачи левая часть меньше $(<)$либо меньше или равна $(\le)$ правой, используется первый подход к поиску базиса. Исходная система ограничений — неравенств преобразуется в систему равенств путем добавления свободных переменных, коэффициенты при которых равны единице. Пусть

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \le b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \le b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \le b_m \end{cases}.$$

В каждое из ограничений системы добавляют свою свободную переменную. Для наглядности расположим их по диагонали.
Тогда

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n + x_{n+1} = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n + 0 + x_{n+2} = b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n + 0 + 0 + \ldots + x_{n+m} = b_m \end{cases}.$$

С экономической точки зрения свободные переменные представляют собой неиспользованные ресурсы, следовательно, их «цена» в целевой функции равна нулю, т.е. они не учитываются в целевой функции.

Коэффициенты при свободных переменных образуют единичную матрицу, определитель которой равен единице. Это означает, что система невырожденная и векторы-столбцы, компонентами которых являются коэффициенты соответствующих свободных переменных, линейно независимы и образуют базис.

Симплекс-метод основан на разложении векторов по базису. Условия задачи можно записать в виде векторов.

$$\begin{bmatrix} a_{11} \\ a_{21} \\ a_{m1} \end{bmatrix} = P_1 \quad \begin{bmatrix} a_{12} \\ a_{22} \\ a_{m2} \end{bmatrix} = P_2 \quad \begin{bmatrix} a_{1n} \\ a_{2n} \\ a_{mn} \end{bmatrix} = P_n \bullet$$

$\begin{bmatrix} b_1 \\ b_2 \\ b_m \end{bmatrix} = P_0$ — вектор правых частей уравнений-ограничений (вектор решений).

Пример задачи оптимизации и ее решение симплекс-методом

Необходимо загрузить судно в порту 3-мя родами грузов: а, б, в. Грузоподъемность судна — *2000 т* (Q_p); грузовместимость *(W)* — *3080* $м^3$; удельный погрузочный объем грузов:

$W_а$=2,1 $м^3$/т;

$W_б$=1,5 $м^3$/т;

$W_в$=1,7 $м^3$/т.

Максимально возможное время погрузки $(T_n\ max)$= 36 час = 2160 мин.

Удельное время погрузки каждого рода груза:

$t_а$ = 0,8 мин/т;

$t_б$ = 0,9 мин/т;

$t_в$ = 1,3 мин/т.

Имеется в порту груза:

а = 900 т;

б = 1300 т;

в = не ограничено.

За перевозку *1т* груза взимается провозная плата:

$C_а$=8 у.е.;

$C_б$=7,5 у.е.;

$C_в$=7 у.е.

Необходимо составить оптимальный план загрузки судна на максимум дохода от перевозки.

Обозначим:

x_1 — количество груза *а*, загруженного в судно в оптимальном плане;

x_2 — количество груза б, загруженного в судно в оптимальном плане;

x_3 — количество груза в, загруженного в судно в оптимальном плане.

Целевая функция — доход от перевозок:

$$Z = 8x_1 + 7,5x_2 + 7x_3 \to max;$$

Ограничения
$$\begin{cases} 1)\, x_1 + x_2 + x_3 \le 2000m \\ 2)\, 2,1x_1 + 1,5x_2 + 1,7x_3 \le 3080м^3 \\ 3)\, 0,8x_1 + 0,9x_2 + 1,3x_3 \le 2160мин \\ 4)\, x_1 \le 900m \\ 5)\, x_2 \le 1300m \\ x_1, x_2, x_3 \ge 0 \end{cases}$$

Перейдем от системы неравенств к системе равенств, добавляя свободные переменные.

$$\begin{cases} x_1 + x_2 + x_3 + x_4 = 2000m \\ 2,1x_1 + 1,5x_2 + 1,7x_3 + x_5 = 3080м^3 \\ 0,8x_1 + 0,9x_2 + 1,3x_3 + x_6 = 2160мин \\ x_1 + x_7 = 900m \\ x_2 + x_8 = 1300m \end{cases}$$

Здесь x_4 — неиспользуемая грузоподъемность;

x_5 — неиспользуемая грузовместимость;

x_6 — неиспользуемое до максимально возможного время погрузки;

x_7 — недопогруженное количество груза *а*;

x_8 — недопогруженное количество груза *б*.

Представим условия (т.е. ограничения) в виде векторов.

$$P_1 = \begin{bmatrix} 1 \\ 2,1 \\ 0,8 \\ 1 \\ 0 \end{bmatrix} \; P_2 = \begin{bmatrix} 1 \\ 1,5 \\ 0,9 \\ 0 \\ 1 \end{bmatrix} \; P_3 = \begin{bmatrix} 1 \\ 1,7 \\ 1,3 \\ 0 \\ 0 \end{bmatrix} \; P_4 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$P_5 = \begin{bmatrix}0\\1\\0\\0\\0\end{bmatrix} \; P_6 = \begin{bmatrix}0\\0\\1\\0\\0\end{bmatrix} \; P_7 = \begin{bmatrix}0\\0\\0\\1\\0\end{bmatrix} \; P_8 = \begin{bmatrix}0\\0\\0\\0\\1\end{bmatrix} \; P_0 = \begin{bmatrix}2000\\3080\\2160\\900\\1300\end{bmatrix}$$

Векторы P_1, P_2, P_3, образованные из коэффициентов при реальных переменных, называются структурными.

Векторы P_4, P_5, P_6, P_7, P_8 называются линейно независимыми (свободными) векторами.

Данная задача решается относительно m линейно независимых базисных векторов. В данном случае это свободные векторы (образующиеся из коэффициентов при свободных переменных) $P_4 \div P_8$.

Алгоритм решения предусматривает построение симплекс-таблиц, которая для данного примера имеет следующий вид (табл. 2.1).

Таблица 2.1

Симплекс-таблица решения задачи линейного программирования

C_j / **C_i**			**8**	**7,5**	**7**	**0**	**0**	**0**	**0**	**0**
0	базис	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	**P_8**
0	P_4	2000	1	1	1	1	0	0	0	**0**
0	P_5	3080	2,1	1,5	1,7	0	1	0	0	**0**
0	P_6	2160	0,8	0,9	1,3	0	0	1	0	**0**
0	P_7	900	1	0	0	0	0	0	1	**0**
0	P_8	1300	0	1	0	0	0	0	0	**1**
Z_j			0	0	0	0	0	0	0	**0**
Z_j–C_j			**8**	**7,5**	**7**	**0**	**0**	**0**	**0**	**0**

Z_j — симплекс-оценка $Z_j = \sum_{i=1}^{m} c_i x_{ij}$;

Z_j–C_j — признак оптимальности в симплекс-методе.

Если задача решается на максимум и значения последней строки ≥ 0 по всем столбцам, то план является оптимальным.

Если задача решается на минимум и значения последней строки ≤ 0 по всем столбцам, то план так же является оптимальным.

Если при решении задач на максимум хотя бы у одного вектора значение $Z_j - C_j < 0$, то план не оптимален и требует улучшений.

В общем виде первоначальная симплекс-таблица выглядит так, как приведено на стр. 56 (см. табл. 2.2).

Таблица 2.2

Общий вид симплекс-таблицы

$\mathbf{C_j}$ / $\mathbf{C_i}$			$\mathbf{C_1}$		$\mathbf{C_j}$		$\mathbf{C_к}$		$\mathbf{C_n}$
	базис	P_0	P_1	…	P_j	…	P_k	…	P_n
$\mathbf{C_1}$	P_1	x_1	x_{11}	…	x_{1j}	…	x_{1k}	…	x_{1n}
…	…	…	…	…	…	…	…	…	…
$\mathbf{C_i}$	P_i	x_i	x_{i1}	…	x_{ij}	…	x_{ik}	…	x_{in}
…	…	…	…	…	…	…	…	…	…
$\mathbf{C_r}$	P_r	x_r	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…	…
$\mathbf{C_m}$	P_m	x_m	x_{mi}	…	x_{mj}	…	x_{mk}	…	x_{mn}
$\mathbf{Z_j}$			Z_1	…	Z_j	…	Z_k	…	Z_n
$\mathbf{Z_j – C_j}$			$Z_1 – C_1$	…	$Z_j – C_j$	…	$Z_k – C_k$	…	$Z_n – C_n$

Рассмотрим алгоритм перехода от одного допустимого плана к другому при решении ЗЛП симплекс-методом, который позволяет превратить неоптимальный план в оптимальный.

Алгоритм имеет следующие этапы.

Определяется вектор, который вводится в базис. Это вектор, который имеет максимальное нарушение признака оптимальности. Столбец, который соответствует вводимому в базис вектору, называется ключевым, его индекс обозначается буквой *k*. В приведенном примере *k = 1*.

Определяется вектор, который выводится из базиса. Это тот вектор, у которого имеет место следующее соотношение:

$$\theta = min\left\{\frac{x_i}{x_{ik}}\right\} для\ x_{ik} > 0,$$

где x_i — элементы вектора решений (столбца P_0);

x_{ik} — элементы ключевого столбца.

Строка, которая соответствует минимуму, т.е. выводимому из базиса вектору, называется ключевой строкой, ее индекс обозначается буквой *r*. Элемент таблицы, который находится на пересечении *k-го* столбца и *r-й* строки, называется генеральным элементом и обозначается x_{rk}.

Определяется новое значение элементов вектора решений:

$$P_0' = P_0 - \theta P_k,$$

$$x_i' = x_i - \theta x_{ik}.$$

Для ключевой строки значение $X_r = \theta$.

Определяется новое значение ключевой строки $x_{rj}' = \frac{x_{rj}}{x_{rk}}$.

Определяются новые значения всех остальных элементов симплекс-таблицы

$$x_{ij}' = x_{ij} - \frac{x_{rj} \cdot x_{ik}}{x_{rk}},$$

где x_{rj} — элемент данного вектора в ключевой строке;

x_{ik} — соответствующий элемент в ключевом столбце;

x_{rk} — генеральный элемент.

Пример расчета по алгоритму.

Выделить вектор с *max* нарушением признака оптимальности, это вектор P_1.

$$\theta = min\left\{\frac{2000}{1};\frac{3080}{2,1};\frac{2160}{0,8};\frac{900}{1}\right\} = 900.$$

Следовательно, ключевая строка будет для вектора P_7, и этот вектор выводится из базиса.

$$x_1^{'} = 2000 - 900 \cdot 1 = 1100;$$
$$x_2^{'} = 3080 - 900 \cdot 2,1 = 1190;$$
$$x_3^{'} = 2160 - 900 \cdot 0,8 = 1440;$$
$$x_4^{'} = \theta = 900;$$
$$x_5^{'} = 1300 - 900 \cdot 0 = 1300.$$

Правила, упрощающие заполнение симплекс-таблицы.

В новой симплекс-таблице значения элементов ключевого столбца будут равны *0*, а на месте генерального элемента будет стоять *1*.

Каждая строка, в ключевом столбце которой стоит *0*, переписывается без изменений.

Каждый столбец, в ключевой строке которого стоит *0*, также переписывается без изменений.

Таблица 2.3

Скорректированная симплекс-таблица

C_j / **i**	базис	P_0	**8** P_1	**7,5** P_2	**7** P_3	**0** P_4	**0** P_5	**0** P_6	**0** P_7	**0** **P_8**
0	P_4	1100	0	1	1	1	0	0	-1	**0**
0	P_5	1190	0	1,5	1,7	0	1	0	-2,1	**0**
0	P_6	1440	0	0,9	1,3	0	0	1	-0,8	**0**
8	P_1	900	1	0	0	0	0	0	1	**0**
0	P_8	1300	0	1	0	0	0	0	0	**1**
Z_j			8	0	0	0	0	0	8	**0**
Z_j–C_j			**0**	**-7,5**	**-7**	**0**	**0**	**0**	**8**	**0**

$$x'_{rj} = \frac{x_{rj}}{x_{rk}};$$

$$x'_{ij} = x_{ij} - \frac{x_{rj} \cdot x_{ik}}{x_{rk}};$$

$$x'_{47} = 0 - \frac{1 \cdot 1}{1} = -1;$$

$$x'_{57} = 0 - \frac{1 \cdot 2,1}{1} = -2,1;$$

$$x'_{67} = 0 - \frac{1 \cdot 0,8}{1} = -0,8;$$

$$Z_j = \sum_{i=1}^{m} C_i \cdot X_{ij} \ .$$

Поскольку в последней строке имеются нарушения, представленный в табл. 2.3 план не является оптимальным.

Рассчитаем целевую функцию для этого плана:

$$Z = 8 \cdot 900 = 7200 y.e.$$

Составим следующую симплекс-таблицу.

Максимальное нарушение — вектор P_2 — ключевой столбец, он будет вводиться в базис.

$$\theta = min\left\{\frac{1100}{1}; \frac{1190}{1,5}; \frac{1440}{0,9}; \frac{1300}{1}\right\} = 794.$$

Выводим P_5, вводим на его место в базис P_2.

$$x'_1 = 1100 - 794 \cdot 1 = 306;$$

$$x'_2 = 794;$$

$$x'_3 = 1440 - 794 \cdot 0,9 = 726;$$

$$x'_4 = 900 - 794 \cdot 0 = 900;$$

$$x'_5 = 1300 - 794 \cdot 1 = 506.$$

Таблица 2.4

Симплекс-тпблица улучшенного плана

C_j / i	базис	P_0	**8** P_1	**7,5** P_2	**7** P_3	**0** P_4	**0** P_5	**0** P_6	**0** P_7	**0** P_8
0	P_4	306	0	0	-0,13	1	-0,67	0	0,4	**0**
7,5	P_2	794	0	1	1,13	0	0,67	0	-1,4	**0**
0	P_6	726	0	0	0,28	0	-0,6	1	0,46	**0**
8	P_1	900	1	0	0	0	0	0	1	**0**
0	P_8	506	0	0	-1,13	0	-0,67	0	1,4	**1**
$\mathbf{Z_j}$			8	-7,5	8,5	0	5	0	-2,5	**0**
$\mathbf{Z_j–C_j}$			**0**	**0**	**1,5**	**0**	**5**	**0**	**-2,5**	**0**

$$x'_{43} = 1 - \frac{1,7 \cdot 1}{1,5} = -0,13; \quad x'_{63} = 0,28;$$

$$x'_{45} = 0 - \frac{1 \cdot 1}{1,5} = -0,67; \quad x'_{65} = -0,6;$$

$$x'_{47} = 0,4; \qquad x'_{67} = 0,46.$$

Поскольку есть отрицательное значение в последней строке, план (табл. 2.4) не оптимальный.

$$Z = 7,5 \cdot 794 + 3 \cdot 900 = 13155 y.e.$$

В итоге после построения еще нескольких симплекс-таблиц мы наконец получили оптимальный план.

Таблица 2.5

Симплекс-таблица оптимального плана

C_j / i	базис	P_0	**8** P_1	**7,5** P_2	**7** P_3	**0** P_4	**0** P_5	**0** P_6	**0** P_7	**0** P_8
0	P_4	35	0,24	0	0	1	0,59	0	0	**0,12**
7,5	P_2	300	0	1	0	0	0	0	0	**1**
0	P_6	125	0,81	0	0	0	0,76	1	0	**0,25**
7	P_3	665	1,24	0	1	0	0,59	0	0	**0,88**
0	P_7	900	1	0	0	0	0	0	1	**0**
$\mathbf{Z_j}$			8,67	7,5	7	0	4,12	0	0	**1,32**
$\mathbf{Z_j–C_j}$			**0,65**	**0**	**0**	**0**	**4,12**	**0**	**0**	**1,32**

В приведенном в табл. 2.4 плане нет нарушений признака оптимальности, все значения последней строки $Z_j – C_j$ больше или равны нулю.

$$Z = 1300 \cdot 7,5 + 665 \cdot 7 = 14406 у.е.$$

Проверяем ограничения для полученного оптимального плана:

$x_1 + x_2 + x_3 = 0 + 1300 + 665 = 1965 < 2000 т$,

$x_4 = 35$ — неиспользуемая грузоподъемность,

$1,5 \cdot 1300 + 1,7 \cdot 665 = 3080 м^3$,

$0,9 \cdot 1300 + 1,3 \cdot 665 < 2160 мин$,

$x_6 = 125 мин$ — сэкономленное время (до максимально возможного),

$0 < 900$ *т,*

$x_2 = 1300 т = 1300 т$.

В оптимальном плане в судно будет погружено *1300 т* груза *б* и *665 т* груза *в*. При этом значение целевой функции составит *14 406 у.е.* — максимально возможный доход от перевозок.

2.7. Модифицированный симплекс-метод (М-метод)

Рассмотрим второй подход к поиску базиса, это тот случай, когда условия задачи (все или часть из них) выражены равенствами.

Для решения таких задач применяется *М-метод* (метод вычисления с искусственным базисом).

Для составления начального допустимого плана в систему ограничений вводят искусственные переменные (векторы), которые имеют не «нулевую» цену, как свободные векторы, а большую *(М)* цену.

Значение *М* предполагается очень большим.

Образно говоря, искусственные векторы могут рассматриваться как свободные векторы, за использование которых взимается большой штраф.

При этом штраф столь велик, что использование этих векторов в конечном оптимальном решении не допускается.

Рассмотрим алгоритм *М-метода* на примере расстановки судов по линиям.

Общая постановка задачи *М-метода.*

Дано:

m — количество типов судов; i — индекс типа судна;

n — количество линий движения; j — индекс линии движения;

Φ_i — количество судов *i-го* типа;

Q_j — грузооборот *j-й* линии за планируемый период;

Z_{ij} — провозная способность *i-го* типа судна на *j-й* линии за планируемый период;

$Э_{ij}$ — расходы *i-го* типа судна на *j-й* линии за планируемый период.

Ограничения:

$\Phi_{ij} \geq 0$ — количество судов *i-го* типа, закрепленное за *j-й* линией, должно быть величиной неотрицательной;

$\sum_{i=1}^{m} Z_{ij}\Phi_{ij} = Q_j$ — на каждую линию необходимо поставить такое количество судов, чтобы выполнить запланированный грузооборот на *j-й* линии;

$\sum_{i=1}^{m} \Phi_{ij} \leq \Phi_i$ — необходимо использовать флота *i-го* типа не больше, чем имеется его в наличии.

Необходимо так расставить суда по линиям, чтобы выполнить заданный план перевозок с минимальными эксплуатационными расходами.

Пример.

Дано:

1) 2 типа грузовых самоходных судов *(m = 2),* *i = 1,2;*

2) 3 грузовые линии *(n = 3),* *j = 1,2,3;*

3) количество судов каждого типа

$\Phi_1 = 15, \Phi_2 = 20.$

Грузооборот на каждой линии, млн ткм

$Q_1 = 800, Q_2 = 1200, Q_3 = 900.$

Z_{ij} — провозная способность судна *i-го* типа на *j-й* линии за навигацию, млн ткм:

$$Z_{ij} = \begin{pmatrix} 50 & 70 & 100 \\ 90 & 130 & 120 \end{pmatrix}.$$

$Э_{ij}$ — эксплуатационные расходы судна *i-го* типа на *j-й* линии за навигацию, тыс. усл. единиц.

$$Z_{ij} = \begin{pmatrix} 65 & 70 & 180 \\ 80 & 90 & 110 \end{pmatrix}.$$

Таким образом, целевая функция:

$$Z = \sum_{i=1}^{2}\sum_{j=1}^{3} \Phi_{ij} Э_{ij} \to min.$$

Алгоритм решения для конкретной задачи.

Составление ограничений —

ограничения:

— $\Phi_{ij} \geq 0$;

— $\sum_{i=1}^{2} Z_{ij}\Phi_{ij} = Q_j \quad (j = 1,2,3)$

$$50\Phi_{11} + 90\Phi_{21} = 800$$

$$70\Phi_{12} + 130\Phi_{21} = 1200$$

$$100\Phi_{13} + 120\Phi_{23} = 900;$$

$$— \sum_{i=1}^{3} \Phi_{ij} \le \Phi_i \quad (i = 1,2)$$

$$\Phi_{11} + \Phi_{12} + \Phi_{13} \le 15$$

$$\Phi_{21} + \Phi_{22} + \Phi_{23} \le 20.$$

Целевая функция:

$$Z = 65 \cdot \Phi_{11} + 70 \cdot \Phi_{12} + 180 \cdot \Phi_{12} + 80 \cdot \Phi_{21} + 90 \cdot \Phi_{22} + 110 \cdot \Phi_{23} \to min$$

Так как при решении задач *М-методом* для получения начального базисного плана расстановки судов необходимо иметь единичную диагональную матрицу и систему равенств, то преобразуем систему ограничений и функционал цели к виду, обеспечивающему необходимое количество единичных векторов, т.е. в каждое из ограничительных уровней добавляем по одному вектору.

$$50\Phi_{11} + 90\Phi_{21} + P_1 + 0 + 0 + 0 + 0 = 800,$$

$$70\Phi_{12} = 130\Phi_{22} + 0 + P_2 + 0 + 0 + 0 = 1200,$$

$$100\Phi_{12} + 120\Phi_{23} + 0 + 0 + P_3 + 0 + 0 = 900,$$

$$\Phi_{11} + \Phi_{12} + \Phi_{12} + 0 + 0 + 0 + P_4 + 0 = 15,$$

$$\Phi_{21} + \Phi_{22} + \Phi_{23} + 0 + 0 + 0 + 0 + P_5 = 20.$$

Таким образом получим равенства и единичные векторы P_1, P_2, и P_3 — искусственные векторы, которые показывают невыполнение плана перевозок на линиях.

P_4 и P_5 — свободные векторы, которые показывают недоиспользованное количество флота.

Искусственные векторы отличаются от свободных тем, что они имеют очень большую цену *(М)*, которую можно рассматривать как штраф за

невыполнение плана перевозок. Свободные векторы, показывающие избыток судов, имеют нулевую цену.

После преобразования функционал цели будет иметь вид:

$$Z = 65\Phi_{11} + 70\Phi_{12} + 80\Phi_{13} + 80\Phi_{21} + 90\Phi_{22} + 110\Phi_{23} + MP_1 + \\ + MP_2 + MP_3 + 0P_4 + 0P_5 \to min.$$

Свободные векторы с нулевой ценой в функционал обычно не записываются.

После произведенных преобразований составляем симплексную таблицу и заполняем ее исходными данными.

Таблица 2.6

Симплекс-таблица

C_j	Базис	C_j	65	70	80	80	90	110	M	M	M	0	0
		P_0	Φ_{11}	Φ_{12}	Φ_{13}	Φ_{21}	Φ_{22}	Φ_{23}	P_1	P_2	P_3	P_4	P_5
M	P_1	1800	50	0	0	90	0	0	1	0	0	0	0
M	P_2	1200	0	70	0	0	130	0	0	1	0	0	0
M	P_3	900	0	0	100	0	0	120	0	0	1	0	0
0	P_4	15	1	1	1	0	0	0	0	0	0	1	0
0	P_5	20	0	0	0	1	1	1	0	0	0	0	1
Z_j			50 M	70 M	100 M	90 M	130 M	120 M	M	M	M	0	0
Z_j-C_j			50 M-65	70 M-70	100 M-100	90 M-80	130 M-90	120 M-110	M-M=0	0	0	0	0

В табл. 2.6 P_0 — вектор решений, который составляется из правых частей ограничений и численно выражает грузооборот и количество судов;

C_i — цена базисных векторов;

C_j — цена структурных, искусственных и свободных векторов.

Далее определяются Z_j и разности Z_j-C_j для каждого вектора.

Z_j — показывает суммарные эксплуатационные расходы по каждому вектору.

$$Z_{\text{j}} = \sum_{i=1}^{5} C_i \, X_{ij} \ ,$$

где X_{ij} — элемент *i-й* строки и *j-го* столбца-вектора.

Разность Z_j-C_j отражает оптимальность плана, т.е., иначе говоря, является условием оптимальности.

Если решение задачи осуществляется на минимум критерия, то оно будет оптимальным при условии

$$Z_j - C_j \le 0 \, .$$

Если же решается задача на максимум критерия, то для достижения оптимальности должно выполняться условие

$$Z_j - C_j \ge 0 \, .$$

Выбираем вектор, который будем вводить в базис. Он будет соответствовать наибольшему нарушению, т.е. наибольшей разности *Zj-Cj*.

Наибольшая разность равна *130 М — 90*, что соответствует вектору $Ф_{22}$. Обозначим этот вектор $P_к$.

Определим в базисе вектор, который следует вывести из базиса, т.е. заместить вектором $P_к$. Для этого производится деление элементов X_i из столбца P_o на соответствующие положительные элементы $X_{iк}$ вектора $P_к$. В базисе замещается тот вектор, для которого достигается минимум отношения

$$\theta = \min_i \left\{ \frac{X_i}{X_{ik}} \right\} для \ X_{ik} > 0 \, .$$

В данном случае

$$\theta = min \left\{ \frac{1200}{130}; \frac{20}{1} \right\} = \frac{1200}{130} = 9,2.$$

Следовательно, из базиса выводится вектор P_2, обозначаем его как P_r.

Элемент на пересечении векторов $P_к$ (ключевой столбец) и P_r (ключевая строка) называется генеральным элементом. Обозначаем его $X_{rк}$, в нашем случае $X_{rк}=130$.

Вектор *Рк (Ф22)* вводим в базис, а вектор *Pr (P2)* выводим. Для этого строим новую симплексную таблицу.

Таблица 2.7

Вторая симплекс-таблица

C_i	Базис	C_j	65	70	80	80	90	110	M	M	M	0	0
		P_0	$Ф_{11}$	$Ф_{12}$	$Ф_{13}$	$Ф_{21}$	$Ф_{22}$	$Ф_{23}$	P_1	P_2	P_3	P_4	P_5
M	P_1	800	50	0	0	90	0	0	1	0	0	0	0
90	P_2	9,2	0	0,54	0	0	1	0	0	0,01	0	0	0
M	P_3	900	0	0	100	0	0	120	0	0	1	0	0
0	P_4	15	1	1	1	0	0	0	0	0	0	1	0
0	P_5	10,8	0	-0,54	0	1	0	1	0	-0,01	0	0	1
Z_j			50 M	48	100 M	90 M	90	120 M	M	0,9	M	0	0
Z_j-C_j			50 M-65	48-70	100 M-80	90 M-80	0	120 M-110	0	0,9 M	0	0	0

В новой таблице (табл. 2.7) в первую очередь заполняется строка вводимого в базис вектора $Ф_{22}$. Для этого все элементы строки P_2 делятся на генеральный элемент *(130)* и заносятся в соответствующие клетки табл. 2.7. В ней в строке базиса проставляется цена вектора $Ф_{22}$, т.е. *90*. Затем определяются новые значения элементов вектора решений по формуле:

$$X_i^{/} = X_i - \theta x_{ik} .$$

Для вновь введенного в базис вектора *(P_{22})* это не определяется,

для остальных:

$$X'_1 = 800 - 9,2 \cdot 0 = 800,$$
$$X'_3 = 900 - 9,2 \cdot 0 = 900,$$
$$X'_4 = 15 - 9,2 \cdot 0 = 15,$$
$$X'_5 = 20 - 9,2 \cdot 1 = 10,8.$$

Все остальные элементы новой таблицы определяются по формуле:

$$x'_{ij} = x_{ij} - \frac{x_{ik} x_{rj}}{x_{rk}}.$$

Из этой формулы следуют правила, упрощающие заполнение симплекс-таблицы.

В новой матрице в столбце, соответствующем ключевому столбцу предыдущей матрицы, все элементы будут равны нулю и лишь на месте генерального элемента будет *1*.

Если в ключевом столбце какой-либо элемент $X_{ik} = 0$, то все элементы этой строки (где находится $X_{ik} = 0$) переписываются в новую матрицу без изменения.

В нашем случае без изменения перепишутся строки P_1, P_3.

Если в ключевой строке какой-либо элемент $X_{rj}=0$, то все элементы *j-го* столбца переписываются в новую матрицу без изменения. В нашем случае столбцы P_{11}, P_{13}, P_{21}, P_{23}, P_1, P_3, P_4, P_5.

Оставшиеся элементы табл. 2.7 определим по приведенной формуле:

$$X'_{52} = 0 - \frac{1 \cdot 70}{130} = -0,54 \quad X'_{58} = 0 - \frac{1 \cdot 1}{130} = -0,01$$

Определяем значения Z_j и $Z_j - C_j$ в табл. 2.7, т. е. ищем максимальное нарушение.

В нашем случае max нарушение в табл. 2.7 равно *120 M-110*, что соответствует вектору Φ_{23}.

Действия, изложенные выше, повторяются до тех пор, пока не будут устранены все нарушения плана, т.е. когда во всех случаях $Z_j – C_j$ будет меньше либо равно нулю.

Расшифровка полученного оптимального решения.

После построения еще нескольких симплекс-таблиц было получено оптимальное решение (таблица, где в последней строке все значения $Z_j - C_j \le 0$).

Приведем фрагмент этой таблицы:

C_1	базис	P_o
80	$Ф_{13}$	9
80	$Ф_{21}$	8,8889
90	$Ф_{22}$	9,2308
0	P_4	6
0	P_5	1,883

Таким образом, в оптимальном плане искомые переменные имеют следующие значения:

$Ф_{13} = 9$ — количество судов первого типа, работающих на третьей линии;

$Ф_{21} = 8,8889$ — количество судов второго типа, работающих на первой линии;

$Ф_{22} = 9,2308$ — количество судов второго типа, работающих на второй линии;

$P_4 = 6$ — неиспользованное количество судов первого типа;

$P_5 = 1,883$ — неиспользованное количество судов второго типа.

Для полученного оптимального плана целевая функция:

$$Z = 80 \cdot 9 + 80 \cdot 8,8889 + 90 \cdot 9,2308 = 2261,88\, y.e.$$

Это минимальное значение эксплуатационных расходов при выполнении заданного плана перевозок наличным флотом.

Проверим ограничения:

$\Phi_{ij} \geq 0$ — выполняется;

$90 \cdot 8{,}8889 = 800$ — выполняется грузооборот первой линии;

$130 \cdot 9{,}2308 = 1200$ — выполняется грузооборот второй линии;

$100 \cdot 9 = 900$ — выполняется грузооборот третьей линии;

$9 < 15$ — осталось не использовано 6 судов первого типа;

$8{,}8889 + 9{,}2308 < 20$ — осталось не использовано 1,8803 судна второго типа.

Дробные части в полученных значениях количества судов означают, что одно судно работает на данной линии не весь плановый период (не всю навигацию например), а часть планового периода.

2.8. Правила подготовки задач линейного программирования к решению

Задачи линейного программирования в реальных условиях могут иметь десятки или даже тысячи переменных и ограничений. Поэтому их решение требует применения средств вычислительной техники. В зависимости от возможностей программного обеспечения содержание подготовки ЗЛП к решению может быть разным.

Собственно подготовка ЗЛП к решению, выполняемая вручную или автоматически, состоит в том, чтобы можно было легко найти опорное решение, от которого затем симплекс-методом или модифицированным симплекс-методом находится оптимальное решение, если оно, конечно, есть.

При рассмотрении сущности симплекс-методов решения ЗЛП удалось просто найти опорное решение, т.к. при представлении задачи в канонической форме коэффициенты при базовых переменных образовывали единичную матрицу. Поэтому подготовка ЗЛП к решению сводится к такому представлению задачи, чтобы в ограничениях можно было выделить

единичную матрицу коэффициентов для переменных, принимаемых в качестве базисных.

Обычно последовательность подготовки ЗЛП к решению сводится к следующему:

1. Исходная задача, заданная в общей или основной форме, приводится к канонической путем введения дополнительных переменных. Каноническая задача будет в этом случае расширенной по отношению к исходной, т.к. переход к канонической задаче выполняется обычно введением дополнительных переменных. Коэффициенты в целевой функции для дополнительных переменных принимаются равными нулю.
2. Если в каких-то ограничениях правая (свободная) часть имеет отрицательные знаки, то они умножаются на минус единицу, чтобы столбец свободных членов в ограничениях стал положительным.
3. Если в левой части ограничений удается выделить переменные, коэффициенты при которых образуют единичную матрицу размером $m \times m$, где m — число ограничений, то эти переменные принимаются в качестве базисных. Все остальные принимаются свободными, и через них выражаются базисные переменные. Принимая свободные переменные равными нулю, легко получить значения базисных переменных и в конечном счете опорное решение, которое называют также *начальным базисным решением*. Подготовка ЗЛП на этом заканчивается.

   К сожалению, не всегда на третьем из рассмотренных этапов удается выделить переменные с единичной матрицей коэффициентов. В этом случае выполняется следующий этап.
4. Каноническая ЗЛП приводится к так называемой *М*-задаче.

   Чтобы лучше понять суть подготовки ЗЛП к решению, рассмотрим следующие примеры.

Пример 1.

Основная форма

$$z(x) = x_1 - x_2 \to max;$$

$$2 \cdot x_1 + 3 \cdot x_2 \geq 6;$$

$$x_1, x_2 \geq 0.$$

Каноническая форма

$$z(x) = x_1 - x_2 + 0 \cdot x_3 \to max;$$

$$2 \cdot x_1 + 3 \cdot x_2 + x_3 = 6;$$

$$x_1, x_2, x_3 \geq 0.$$

Начальное базисное решение

$$x_3 = 6, \;\; x_1 = x_2 = 0.$$

Конечно, каноническую задачу можно представить с ограничением

$$x_1 + 3/2 \cdot x_2 + 1/2 \cdot x_3 = 3$$

или

$$2/3 \cdot x_1 + x_2 + 1/2 \cdot x_3 = 2.$$

Соответственно, начальными базисными решениями были бы:

$$x_1 = 3, \quad x_2 = x_3 = 0 \text{ или } x_2 = 2, \quad x_1 = x_3 = 0.$$

Можно показать что при n переменных и m ограничениях $(m < n)$ число начальных базисных решений, а значит, число вершин симплекса равно $\frac{n!}{m!(n-m)!}$. Поэтому разные постановщики задачи могут принять разные начальные базисные решения.

Пример 2.

Основная задача

$$z(x) = x_1 - x_2 \to max;$$

$$2 \cdot x_1 + 3 \cdot x_2 \geq 6;$$

$$x_1 \leq 4;$$

$$x_2 \leq 3;$$

$$x_1, x_2 \geq 0.$$

Каноническая (расширенная) задача

$$z(x) = x_1 - x_2 + 0 \cdot x_3 + 0 \cdot x_4 + 0 \cdot x_5 \to max;$$

$$2 \cdot x_1 + 3 \cdot x_2 - 1 \cdot x_3 + 0 \cdot x_4 + 0 \cdot x_5 = 6;$$

$$1 \cdot x_1 + 0 \cdot x_2 + 0 \cdot x_3 + 1 \cdot x_4 + 0 \cdot x_5 = 4;$$

$$0 \cdot x_1 + 1 \cdot x_2 + 0 \cdot x_3 + 0 \cdot x_4 + 1 \cdot x_5 = 3;$$

$$x_1, x_2, x_3, x_4, x_5 \geq 0.$$

Как видно, матрица коэффициентов при переменных x_3, x_4, x_5 не является единичной. Сформулированная каноническая задача имеет m ограничений и n переменных. Можно, конечно, взять в качестве свободных любые n-m переменных и выразить через них все остальные, которые будут базисными. Но даже для приведенного простого примера этот путь представляется слишком сложным.

Идея сведения канонической задачи к *М-задаче* состоит в следующем. В ограничения вводятся еще дополнительные переменные так, чтобы можно было выделить переменные, для которых коэффициенты образуют единичную матрицу размером $m \times m$. Но эти введенные переменные искажают исходную постановку задачи, и в окончательном решении они должны принять нулевое значение. Чтобы это обеспечить, достаточно в целевой функции учесть новые введенные переменные с бесконечно большим отрицательным коэффициентом для задач на максимум и бесконечно большим положительным коэффициентом в задаче на минимум. Модуль этого коэффициента обозначают условно буквой *М*. Если любая из дополнительно введенных переменных будет отлична от нуля в конечном решении, то решение не будет оптимальным, т.к. по сравнению с остальными коэффициентами значение *М* считается бесконечно большим.

Для рассматриваемого примера достаточно ввести такую дополнительную переменную с коэффициентом единица в первое ограничение. Во всех остальных ограничениях эта переменная будет иметь нулевой коэффициент. В результате *М-задача* будет иметь вид:

$$z(x) = x_1 - x_2 + 0 \cdot x_3 + 0 \cdot x_4 + 0 \cdot x_5 - M \cdot x_6 \to max;$$

$$2 \cdot x_1 + 3 \cdot x_2 - 1 \cdot x_3 + 0 \cdot x_4 + 0 \cdot x_5 + 1 \cdot x_6 = 6;$$

$$1 \cdot x_1 + 0 \cdot x_2 + 0 \cdot x_3 + 1 \cdot x_4 + 0 \cdot x_5 + 0 \cdot x_6 = 4;$$

$$0 \cdot x_1 + 1 \cdot x_2 + 0 \cdot x_3 + 0 \cdot x_4 + 1 \cdot x_5 + 0 \cdot x_6 = 3;$$

$$x_1, x_2, x_3, x_4, x_5, x_6 \geq 0.$$

Теперь при переменных x_4, x_5, x_6 коэффициенты образуют единичную матрицу. Поэтому легко найти *начальное базисное решение*:

$$x_6 = 6,\ x_4 = 4,\ x_5 = 3,\ x_1 = x_2 = x_3 = 0.$$

Таким образом, исходную постановку ЗЛП введением дополнительных переменных всегда можно привести к каноническому виду. Если при этом нельзя выделить базисные переменные, матрица коэффициентов при которых является единичной, то введением еще дополнительных переменных каноническую задачу всегда можно свести к *М-задаче*, в которой выделение базисных переменных с единичной матрицей уже становится возможным.

Следует отметить, что описанная процедура подготовки ЗЛП к решению не является единственной. Ее недостаток — увеличение размерности задачи за счет введения дополнительных переменных, что в конечном счете увеличивает объем вычислительной работы. Поэтому для методов ручного счета имеются и другие процедуры поиска начального базисного решения.

2.9. Симплекс-таблицы и работа с ними

Симплекс-таблицы представляют собой инструмент унификации процедуры счета при решении ЗЛП обычно с использованием модифицированного симплекс-метода.

Следует отметить, что оформление их достаточно разнообразно. Например, в компьютерном курсе по линейному программированию используется форма симплекс-таблицы, приведенная в табл. 2.8.

Дадим описание этой таблицы по строкам и укажем порядок заполнения. В первой строке "$C_1, C_2, ..., C_{n+q}$", проставляются значения коэффициентов целевой функции при переменных канонической или *М-задачи*. Индексом q здесь обозначено число введенных дополнительных переменных.

Таблица 2.8

Симплекс-таблица. Вариант 1

				C_1	C_2	…	c_n	…	c_{n+q}	$Б_{pj}$
	c_j	$Б_п$	$Б_{pj}$	X_1	X_2	…	x_n	…	X_{n+q}	a_{jl}
1			b_1	a_{11}	a_{12}	…	a_{1n}	…	a_{1n+q}	
2			b_2							
…										
m			b_m	a_{m1}	a_{m2}	…			a_{mn+q}	

Во второй строке записываются названия столбцов для следующих строк таблицы.

Все остальные строки, кроме последней, число которых равно «*m*», заполняются в следующем порядке:

1. Заполняются столбцы $x_1 \ldots x_{n+q}$. В них заносятся коэффициенты ограничений задачи, стоящие при переменных x_i, образующие ее матрицу $A = (a_{ji})$, где j — номер ограничения (номер строки), а индекс i — номер столбца.
2. Определяются базисные переменные, коэффициенты при которых образуют единичную матрицу, и записываются в столбец $Б_П$.
3. В столбец $Б_{pj}$ заносятся значения правых частей ограничений $b_1, b_2, \ldots, b_m$, т.е. значения базисных переменных.
4. В столбец C_j заносятся коэффициенты целевой функции при базисных переменных, занесенных в столбец $Б_П$.
5. Для перехода к следующему базисному решению и определения переменной x_1, вводимой в базис, рассчитываются симплекс-разности $\Delta_i = C_i - \sum_{j=1}^{m} a_{ji} \cdot C_j^* = C_i - z_i$, $i = 1, \ldots, n+q$, занося их в последнюю строку

для столбца x_i. В этой формуле C_j^* выбираются из столбца C_j таблицы, т.е. коэффициенты при переменной в ограничениях умножаются на коэффициенты при базисных переменных в целевой функции. Если все симплекс-разности Δ_i отрицательны в задачах на *max* или положительны в задачах на *min*, то текущее решение задачи является оптимальным. Иначе можно улучшить решение. Переменной x_1, которую следует перевести из свободных в базисную, будет та, для которой симплекс-разность Δ_i положительна и максимальна в задачах на *max* и отрицательна и минимальна в задачах на *min*.

6. Для определения переменной x_k, выводимой из базиса, вычисляем симплекс-отношения $\frac{Б_{pj}}{a_{jl}}$, где a_{jl} — коэффициенты столбца, в котором стоит переменная x_1, вводимая в базис. Если все симплекс-отношения меньше нуля, то задача не имеет решения, т.к. функция цели может неограниченно возрастать в задачах на *max* и убывать в задачах на *min*. В противном случае в качестве выводимой из базиса переменной выбирается та, для которой симплекс-отношение $\frac{Б_{pj}}{a_{jl}}$ положительно и минимально.
7. Если из базиса выводится переменная x_k и на ее место вводится переменная x_1, то для определения следующего базисного решения пересчитывается заново текущая таблица, сохраняя форму, в следующем порядке.

   Первой в новой таблице заполняется строка, в которой произошли изменения по базису, т.е. *k-я* строка. Ее новые значения определяются по формуле:

$$b_k^* = b_k / a_{kl}; \qquad a_{ki}^* = a_{ki} / a_{kl}{}^c, \quad i = 1,\ldots,n+m,$$

где a_{kl} — генеральный элемент, лежащий в старой таблице на пересечении *k-й* строки и *1-го* столбца, b_k и a_{ki} — коэффициенты *k-й* строки старой таблицы. Поэтому $a*k_1$, т.е. новое значение на месте разрешающего элемента равно 1.

Затем вычисляются коэффициенты всех остальных строк новой таблицы по формулам:

$$a^*_{ji} = a_{ji} - a_{j_1} / a_{k_1}, \text{ для всех } i \text{ и } j \text{ при } j \neq k,$$

где a^*_{ji} — коэффициенты *j-й* строки новой таблицы;

a_{ji}, a_{j_1} — коэффициенты в *j-й строке* в *i-м* и *1-м* столбцах старой таблицы, соответственно.

Понятно, что $a^*_{ji} = a_{j_1} - a_{k_1} \cdot a_{j_1} / a_{k_1} = 0$, для всех j при $j \neq k$, т.е. новые значения коэффициентов в разрешающем столбце равны нулю, кроме нового значения $a^*_{k_1} = 1$ для генерального элемента.

Аналогичные вычисления выполняются с элементами свободного столбца:

$$b^*_j = b_j - b_k \cdot a_{j_1} / a_{k_1}, j \neq k.$$

В столбце *БП* записывается имя переменной x_k, которая стала теперь базисной взамен переменной x_1, переведенной в число свободных. В соответствующей строке столбца C_j записывается значение коэффициента при новой базисной переменной в целевой функции.

8. Для новой получившейся таблицы расчеты повторяются начиная с пятого пункта.

Рассмотрим следующий пример. Исходная формулировка ЗЛП имеет вид:

$$z(x) = 50 \cdot x_1 + 60 \cdot x_2 \to max;$$

$$2 \cdot x_1 + 3 \cdot x_2 \leq 180;$$

$$3 \cdot x_1 + 2 \cdot x_2 \le 150.$$

Приведем исходную задачу к каноническому виду:

$$z(x) = 50 \cdot x_1 + 60 \cdot x_2 + 0 \cdot s_1 + 0 \cdot s_2 \to max;$$

$$1) 2 \cdot x_1 + 3 \cdot x_2 + s_1 = 180 \text{ или } 1) 2 \cdot x_1 + 3 \cdot x_2 + 1 \cdot s_1 + 0 \cdot s_2 = 180;$$

$$2) 3 \cdot x_1 + 2 \cdot x_2 + s_2 = 150 \text{ или } 2) 3 \cdot x_1 + 2 \cdot x_2 + 0 \cdot s_1 + 1 \cdot s_2 = 150.$$

Используем приведенный выше алгоритм из 8 пунктов.

Как видно, коэффициенты при s_1 и s_2 образуют единичную матрицу, и потому их можно принять за базисные переменные. После выполнения пунктов $1 \div 6$ приведенного выше алгоритма начальная таблица для расчетов примет вид:

Таблица 2.9

Начальная симплекс-таблица. Вариант 1

				50	60	0	0	$Б_{pj}$
	c_j	$Б_п$	$Б_{pj}$	X_1	X_2	S_1	S_2	a_{jl}
1	0	S_1	180	2	3	1	0	60
2	0	S_2	150	3	2	0	1	75
	$\Delta i = ci - zi$			50	60	0	0	0

Симплекс-разности будут равны:

$$\Delta 1 = 50 - 2 \cdot 0 - 3 \cdot 0 = 50, \ \Delta 2 = 60 - 3 \cdot 0 - 2 \cdot 0 = 60,$$

$$\Delta 3 = 0 - 1 \cdot 0 - 0 \cdot 0 = 0, \ \Delta 4 = 0 - 0 \cdot 0 - 1 \cdot 0 = 0,$$

что и проставлено в последней строке симплекс-таблицы. В правой крайней клетке этой строки удобно записать значение целевой функции, которое равно сумме произведений значений базисных переменных на их коэффициенты в целевой функции, т.е. равно скалярному произведению $(c_j) T \cdot Б_{pj}$. В нашем примере значение целевой функции составит:

$$0 \cdot 180 + 0 \cdot 150 = 0.$$

Так как задача решается на *max*, то среди симплекс-разностей, больших нуля, выбираем наибольшую. Поэтому перевод переменной x_2 из свободной в базисную позволяет увеличить значение целевой функции на 60 единиц на каждую единицу увеличения x_2.

Далее вычисляем и заносим в последний столбец симплекс-отношения. Значения их будут равны:

$$Бр_1 / a_{12} = 180 / 3 = 60\ ,\ Бр_2 / a_{22} = 150 / 2 = 75\ .$$

Так как для базисной переменной s_1 симплексное отношение наименьшее, то, следовательно, из базисных в свободную нужно перевести переменную s_1.

Таким образом, определим номер $k = 1$ для разрешающей строки и номер $l = 2$ для разрешающего столбца.

Далее необходимо сделать пересчет таблицы согласно пункту 7 приведенного выше алгоритма. Новый вид таблицы после первой итерации будет иметь вид (табл. 2.10).

Таблица 2.10

Симплекс-таблица после первой итерации. Вариант 1

				50	60	0	0	$Б_{pj}$
	c_j	$Б_п$	$Б_{pj}$	X_1	X_2	S_1	S_2	a_{jl}
1	60	x_2	60	0,67	1	0,33	0	89,35
2	0	S_2	30	1,67	0	–0,67	1	17,96
	$\Delta i = ci - zi$			9,8	0	–19,8	0	3600

Покажем, как получились приведенные в табл. 2.10 значения. Первой в новой таблице заполняется строка, в которой произошли изменения по базису, т.е. 1-я строка в нашем примере. Поскольку переменная x_2 вводится в базис, а s_1 переводится в свободную, то в столбце $Б_П$ следует записать

x_2, а в столбце C_j записать коэффициент при x_2 в целевой функции, т.е. число 60. Для остальных элементов первой строки их новые значения определяются по формулам:

$$b*1 = b_1 / a_{12} = 180/3 = 60;$$

$$a*11 = a_{11} / a_{12} = 2/3 = 0,67;$$

$$a*12 = a_{12} / a_{12} = 3/3 = 1;$$

$$a*13 = a_{13} / a_{12} = 1/3 = 0,33;$$

$$a*14 = a_{14} / a_{12} = 0/3 = 0.$$

Затем вычисляются коэффициенты всех остальных строк новой таблицы по формулам:

$$b*2 = b_2 - b_1 \cdot a_{22} / a_{12} = 150 - 180 \cdot 2/3 = 30;$$

$$a*21 = a_{21} - a_{11} \cdot a_{22} / a_{12} = 3 - 2 \cdot 2/3 = 1,67;$$

$$a*22 = a_{22} - a_{12} \cdot a_{22} / a_{12} = 0;$$

$$a*23 = a_{23} - a_{13} \cdot a_{22} / a_{12} = 0 - 1 \cdot 2/3 = -0,67;$$

$$a*24 = a_{24} - a_{14} \cdot a_{22} / a_{12} = 1 - 0 \cdot 2/3 = 1.$$

Новое значение целевой функции составит:

$$60 \cdot 60 + 0 \cdot 30 = 3600.$$

Далее согласно пункту 8 приведенного выше алгоритма вычисляются симплекс-разности для табл. 2.10:

$$\Delta 1 = 50 - 0,67 \cdot 60 - 1,67 \cdot 0 = 9,8;$$

$$\Delta 2 = 60 - 1 \cdot 60 - 0 \cdot 0 = 0;$$

$$\Delta 3 = 0 - 0,33 \cdot 60 + 0,67 \cdot 0 = -19,8;$$

$$\Delta 4 = 0 - 0 \cdot 60 - 1 \cdot 0 = 0.$$

Эти значения следует занести в последнюю строку табл. 2.10. Из анализа симплекс-разностей следует, что перевод переменной x_1 из свободной в базисную позволит увеличить значение функции цели.

Остается определить разрешающую строку и, следовательно, базисную переменную, которую следует перевести в число свободных. Для этого вычисляем симплекс-отношения:

$$Б_{p1} / a_{11} = 60 / 0,67 = 89,55 ; \; Б_{p2} / a_{21} = 30 / 1,67 = 17,96 .$$

Минимальным является второе симплекс-отношение, и, следовательно, переменную s_2 необходимо из числа базисных перевести в свободные. Далее необходимо выполнить перерасчет элементов табл. 2.10, т.е. выполнить вторую итерацию. Вид таблицы после второй итерации приведен в табл. 2.11.

Таблица 2.11

Симплекс-таблица после второй итерации. Вариант 1

				50	60	0	0	$Б_{pj}$
	c_j	$Б_п$	$Б_{pj}$	X_1	X_2	S_1	S_2	a_{jl}
1	60	x_2	48	0	1	0,6	–0,4	
2	50	x_1	18	1	0	–0,4	0,6	
	$\Delta i = ci - zi$			0	0	–16	–6	3780

В табл. 2.11 показаны также значения симплекс-разностей после второй итерации. Так как все они ≤ 0, то решение $x_1 = 18$, $x_2 = 48$, $s_1 = s_2 = 0$, при котором значение целевой функции равно 3780, является оптимальным.

В ППП QSB для решения ЗЛП используются симплекс-таблицы несколько иного вида. При использовании пакета QSB пользователю нет необходимости приводить исходную задачу к канонической форме или *М-задаче*. Это преобразование выполняется программой пакета.

Предположим, что формулировка ЗЛП является той же самой, по которой составлена табл. 2.9. Поэтому после ввода данных о задаче без приведения ее к канонической форме начальная симплекс-таблица автоматически будет иметь вид, показанный в табл. 2.12.

Таблица 2.12

Симплекс-таблица. Вариант 2. Initial tableau

Basis	$C_{(j)}$	X_1 50	X_2 60	S_1 0	S_2 0	$B_{(j)}$	$B_{(j)}$ $A_{(j,1)}$
S_1	0	2	3	1	0	180	
S_2	0	3	2	0	1	150	
$C(i)-z(i)$ *Big M		50 0	60 0	0 0	0 0	0 0	

Из сравнения табл. 2.8 и 2.12 видно, что они содержат одинаковую информацию, только с другим размещением данных.

Рассмотри теперь следующую задачу:

найти *max* для $z(x) = 50x_1 + 60x_2$ при ограничениях:

$$2x_1 + 3x_2 \le 50;$$

$$3x_1 + 2x_2 \ge 2;$$

$$3x_1 + 5x_2 = 3.$$

После ввода условий задачи начальная таблица в этом случае будет иметь вид, показанный в табл. 2.13.

Таблица 2.13

Начальная симплекс-таблица. Вариант 2. Initial tableau

Basis	$C_{(j)}$	X_1 50	X_2 60	S_1 0	S_2 0	A_2 –M	A_3 –M	$B_{(j)}$	$B_{(j)}$ $A_{(j,1)}$
S_1	0	2	3	1	0	0	0	50	
A_2	–M	3	2	0	–1	1	0	2	
A_3	–M	3	5	0	0	0	1	3	
$C(i)-z(i)$ *Big M		50 6	60 7	0 0	0 –1	0 0	0 0	0 –5	

Как видно из табл. 2.13, исходная задача автоматически преобразовалась в *М–задачу*. Применяемый в пакете алгоритм модифицированного симплекс-метода для приведенной таблицы сводится к следующему.

Определяется максимальная положительная (в задачах на *min* минимальная отрицательная) симплекс-разность:

$$\Delta_i = C(i) - \sum_{j=1}^{m} a_{ji} \cdot C(j) = C(i) - \sum_{\substack{j-1 \\ C_j^{\neq \pm M}}}^{m} a_{ji} \cdot C(j) - \sum_{\substack{j-1 \\ C_j^{=\pm M}}}^{m} a_{ji} \cdot C(j) = C(i) - z(i) - \{g \ * Big\, M\}.$$

и записывается в последней строке.

Обозначение g_i в фигурных скобках соответствует множителю при бесконечно большом M, который записывается отдельно. Так, для первого столбца табл. 2.13 получим:

$$\Delta 1 = 50 - 2 \cdot 0 - 3 \cdot (-M) - 3 \cdot (-M) = 50 + \{6 * Big\, M\}.$$

Для второго столбца:

$$\Delta 2 = 60 - 3 \cdot 0 - 2 \cdot (-M) - 5 \cdot (-M) = 60 + \{7 * Big\, M\}.$$

Эти вычисления составляют первый шаг итерации и результаты его приведены в табл. 2.14.

Таблица 2.14

Первая итерация. Вариант 2. Iteration 1

Basis	$C_{(j)}$	X_1 50	X_2 60	S_1 0	S_2 0	A_2 –M	A_3 –M	$B_{(j)}$	$B_{(j)}$ $A_{(j,1)}$
S_1	0	2	3	1	0	0	0	50	16,67
A_2	–M	3	2	0	–1	1	0	2	1,00
A_3	–M	3	5	0	0	0	1	3	0,60
$C(i) - z(i)$		50	60	0	0	0	0	0	
*Big M		6	7	0	–1	0	0	–5	

Current objective function value (Max.) = 0 + (-5 Big M)

< Highlighted variable is the entering or leaving variable >

Как видно из табл. 2.14, для переменной x_2 симплекс-разность максимальна, и эта переменная должна быть переведена из свободной в базисную.

Для определения переменной, которая должна быть выведена из базиса, элементы столбца сводных членов $B(j)$ делятся на стоящие в тех же

строках элементы столбца выбранной переменной x_2 и результат записывается в последний столбец. Из значений последнего столбца определяется минимальное симплекс-отношение, а соответствующая строка определяет базисную переменную, которая должна быть переведена в свободную. В табл. 2.14 это строка с переменной A_3. На этом заканчивается второй шаг итерации.

Таким образом, выбраны разрешающий столбец (x_2) и разрешающая строка (A_3), а также разрешающий элемент на их пересечении. Далее необходимо выполнить пересчет таблицы для второй итерации (табл. 2.15).

Таблица 2.15

Вторая итерация. Вариант 2. Iteration 2

Basis	$C_{(j)}$	X_1	X_2	S_1	S_2	A_2	A_3	$B_{(j)}$	$B_{(j)}$ / $A_{(j,1)}$
		50	60	0	0	-M	-M		
S_1	0	0,20	0	1	0	0	-0,60	48,20	241,0
A_2	-M	1,80	0	0	-1	1	-0,40	0,80	0,44
X_2	60	0,60	1	0	0	0	0,20	0,60	1,00
$C(i)-z(i)$		14	0	0	0	0	-12	36	
*Big M		1,8	0	0	-1	0	-1,4	-0,8	

Current objective function value (Max.) = 36 + (-0,8 Big M)

< Highlighted variable is the entering or leaving variable >

Entering: X_1 Leaving: A_2

Значения элементов табл. 2.15 определяются следующим образом:

1. Разрешающий элемент становится равным 1, а все остальные элементы разрешающего столбца равны 0.
2. Элементы разрешающей строки равны частному от деления старого значения соответствующего элемента разрешающей строки на старое значение разрешающего элемента.
3. Все остальные элементы таблицы определяются по следующему правилу.

Обозначим k — номер разрешающей строки (в табл. 2.14 строка A_3, поэтому k = 3), 1 — номер разрешающего столбца (в табл. 2.14 для столбца x_2 – 1=2).

Пусть a_{ji} — старое значение элемента (в табл. 2.14) на пересечении *j-й* строки и *i-го* столбца. Тогда новое значение a^*_{ji} этого элемента (для табл. 2.15) определяется по формуле:

$a^*_{ji} = a_{ji} - a_{ki} \cdot a_{j1} / a_{kl}$ для всех i и j при $j \neq k$.

Например, для j, i=1 получим $a*11 = 2 - 3 \cdot 3 / 5 = 0,2$.

При пересчете таблицы справедливы следующие правила:

- если для некоторой строки n старое значение разрешающего столбца в строке n равно нулю, то эта строка сохраняет свои значения (кроме позиции в разрешающем столбце, в которой новое значение 0);
- если старое значение в разрешающей строке равно нулю, то соответствующий столбец сохраняет свои значения.

Эти правила вытекают из алгоритма преобразований симплекс-таблицы.

В столбце $C(j)$ разрешающей строки записывается коэффициент целевой функции для переменной (X_2), переводимой в базис (в приведенном примере 60).

После этого возможно выполнение второй итерации (вычисления последней строки с симплекс-разностями, вычисления последнего столбца с симплекс-отношениями, выбора разрешающего столбца, разрешающей строки, разрешающего элемента, выбора переменных для обмена и пересчета данных таблицы для следующей итерации).

Вычисления заканчиваются, как только на очередной итерации все симплекс-разности не больше нуля (в задаче на *min* они должны быть все не меньше нуля).

Если все значения базисных переменных больше нуля, то оптимальный план найден, иначе задача не имеет решения. Вид симплекс-таблицы к началу 3-й интерации показан в табл. 2.16.

Таблица 2.16

Третья итерация. Вариант 2. Iteration 3

Basis	$C_{(j)}$	X_1 50	X_2 60	S_1 0	S_2 0	A_2 –M	A_3 –M	$B_{(j)}$	$B_{(j)}/A_{(j,1)}$
S_1	0	0	–0,00	1,00	0,111	–0,111	–0,556	48,11	433,0
X_1	50	1	0	0	–0,556	0,556	–0,222	0,444	Inf.
X_2	60	0	1	0	0,333	–0,333	0,333	0,333	1,00
$C(i)-z(i)$		0	0	0	7,778	–7,788	–8,89	42,22	
*Big M		0	0	0	0	–1	–1	0	

Current objective function value (Max.) = 42,22222

< Highlighted variable is the entering or leaving variable >

Entering: S_2 Leaving: X_2

Таблица 2.17

Окончательная таблица. Вариант 2. Final tableau (Total iteration = 3)

Basis	$C_{(j)}$	X_1 50	X_2 60	S_1 0	S_2 0	A_2 -M	A_3 -M	$B_{(j)}$	$B_{(j)}/A_{(j,1)}$
S_1	0	0	-0,333	1,00	0	0	-0,667	48,11	0
X_1	50	1	1,667	0	0	0	0,333	1	0
S_2	0	0	3,00	0	1,00	-1,00	1,00	1	0
$C(i)-z(i)$		0	-23,3	0	0	0	-16,7	50,00	
*Big M		0	0	0	0	-1	-1	0	

Обозначение Inf. означает, что для переменной x_1 увеличение ее неограниченно. Как видно из таблицы, на третьей итерации нужно переменную X_2 перевести в свободные, а S_2 — в базисные. Выполнив третью итерацию, получим симплекс-таблицу, приведенную в табл. 2.17. Эта таблица является финальной, т.к. в задачах на *max* все симплекс-разности меньше или равны нулю, и потому дальнейшее улучшение решения невозможно.

Оптимальное решение будет: $S_1 = 48$, $X_1 = S_2 = 1$, $X_2 = A_2 = A_3 = 0$. При этом максимальное значение целевой функции равно 50.

Заметим, что симплекс-разности в итоговой таблице соответствуют значениям коэффициентов целевой функции на последней итерации. Так, на основании табл. 2.17 целевая функция на последней итерации будет иметь вид:

$$Z(x) = 0 \cdot X_1 - 23,3 \cdot X_2 + 0 \cdot S_1 + 0 \cdot S_2 + (0 - M) \cdot A_2 + (-16,7 - M) \cdot A_3 + 50.$$

Как видно, коэффициенты при базисных переменных равны нулю. Это правило является общим. Если равны нулю коэффициенты при свободных переменных, являющихся основными, то задача имеет множество решений, т.к. таким свободным переменным можно придавать любые значения, и значение целевой функции при этом не изменится. В этом случае после финальной таблицы выдается сообщение *with multiple solutions*.

2.10. Анализ оптимального решения

В методах решения ЗЛП значения всех коэффициентов целевой функции и ограничений считаются точно известными. В действительности их точное определение не всегда возможно. Особенно это касается коэффициентов c_i целевой функции, экономически обычно соответствующих стоимостям единицы продукции *i-го* вида, и правых частей ограничений b_j, экономически соответствующих располагаемым ресурсам *j-го* вида. Неточно могут быть известны и коэффициенты a_{ji}, экономически соответствующие нормативам расхода *j-го* ресурса при производстве единицы продукции *i-го* вида. Поэтому всегда необходимо выполнить анализ оптимального решения с целью выяснения влияния на оптимальное решение тех или иных данных задачи.

Анализ оптимального решения сводится к параметрическому анализу и анализу чувствительности оптимального решения.

При параметрическом анализе исследуется влияние коэффициентов a_{ji} на результаты оптимального решения путем многократного решения задачи оптимизации при разных значениях этих коэффициентов. Параметрический анализ называют также параметрическим программированием.

Анализ чувствительности оптимального решения сводится к ответу на вопрос: в каких диапазонах могут изменяться c_i и b_j, при которых сохраняется найденное оптимальное решение? При этом сохранение оптимального решения понимается в смысле сохранения набора базисных переменных, который соответствует полученному оптимальному решению.

Имея в виду геометрическую интерпретацию ЗЛП (см. п. 2.5), можно сказать, что задача чувствительности оптимального решения сводится к определению того, в каких пределах можно изменять наклон линии целевой функции и перемещать вершину симплекса, которая все равно остается вершиной оптимального решения. Конкретные значения базисных переменных и целевой функции, естественно, будут меняться при изменении коэффициентов c_i и b_j в постановке задачи.

Анализ влияния коэффициентов целевой функции

Чтобы понять, как определяются диапазоны изменения коэффициентов целевой функции, при которых сохраняется структура оптимального решения, рассмотрим следующий пример. Предположим, что ЗЛП имеет вид:

$$z(x) = C_1 \cdot x_1 + C_2 \cdot x_2 + C_3 \cdot x_3 + C_4 \cdot x_4 \to max,$$

где $C_1 = 4;\ C_2 = 5;\ C_3 = 9;\ C_4 = 11$,

при ограничениях:

$$\begin{aligned} &(1) 1 \cdot x_1 + 1 \cdot x_2 + 1 \cdot x_3 + 1 \cdot x_4 \le 15, \\ &(2) 7 \cdot x_1 + 5 \cdot x_2 + 3 \cdot x_3 + 2 \cdot x_4 \le 120, \\ &(3) 3 \cdot x_1 + 5 \cdot x_2 + 10 \cdot x_3 + 15 \cdot x_4 \le 100. \end{aligned} \qquad (2.2)$$

Последняя итерация симплекс-метода или модифицированного симплекс-метода для этой задачи будет иметь вид:

$$z(x) = 99,29 - 0,429 \cdot x_2 - 1,57 \cdot x_4 - 1,86 \cdot s_1 - 0,714 \cdot s_3 \to max$$

при ограничениях:

$$\begin{cases} (1)\, x_1 \to +7,143 = 1 \cdot x_1 + 0,714 \cdot x_2 + 0 \cdot x_3 - 0,714 \cdot x_4 + 1,429 \cdot s_1 + 0 \cdot s_2 - 0,143 \cdot s_3, \\ (2)\, s_2 \to +46,43 = 0 \cdot x_1 - 0,857 \cdot x_2 + 0 \cdot x_3 + 1,857 \cdot x_4 - 8,71 \cdot s_1 + 1 \cdot s_2 + 0,571 \cdot s_3, \\ (3)\, x_3 \to +7,857 = 0 \cdot x_1 + 0,286 \cdot x_2 + 1 \cdot x_3 + 1,714 \cdot x - 0,429 \cdot s_1 + 0 \cdot s_2 + 0,143 \cdot s_3. \end{cases} \quad (2.3)$$

Так как все коэффициенты в целевой функции на этой итерации отрицательны, то найдено оптимальное решение исходной ЗЛП:

- свободные переменные: $x^*_2 = x^*_4 = s*_1 = s^*_3 = 0$;
- базисные переменные: $x_1^* = 7,143; s_2^* = +46,43; x_3^* = +7,857$;
- значение целевой функции: $z(x) = 99,29$.

Начнем с оценки чувствительности оптимального решения по отношению к изменению заданных коэффициентов целевой функции при свободных переменных x_2 и x_4. Симплекс-разности для этих переменных будут:

$$\begin{cases} \Delta_2 = C_2 - C_1 \cdot 0,714 - 0 \cdot (-0,857) - C_3 \cdot 0,286 = 5 - 4 \cdot 0,714 - 9 \cdot 0,286 = -0,429, \\ \Delta_4 = C_4 - C_1 \cdot (-0,714) + 0 \cdot (1,857) - C_3 \cdot 1,714 = 11 + 4 \cdot 0,714 - 9 \cdot 1,714 = -1.57. \end{cases} \quad (2.4)$$

Предположим теперь, что заданные значения коэффициентов C_2 и C_4 стали, соответственно, $C_2 + \delta_2$ и $C_4 + \delta_4$. Новые значения симплекс-разностей будут тогда равны:

$$\Delta'_2 = -0,429 + \delta_2; \Delta'_4 = -1,57 + \delta_4.$$

Для того чтобы оптимальное решение осталось тем же (в смысле сохранения набора базисных и свободных переменных), необходимо, чтобы каждая из симплекс-разностей при любых изменениях δ_2 и δ_4 не оказалась больше нуля, т.е. должны соблюдаться условия:

$$\Delta'_2 = -0,429 + \delta_2 \le 0\,; \Delta'_4 = -1,57 + \delta_4 \le 0\,. \qquad (2.5)$$

Если δ_2 и δ_4 отрицательны, то условия (2.5) будут выполняться при любых их значениях $(-\infty \le \delta_2 \le 0$ и $-\infty \le \delta_4 \le 0)$.

Если δ_2 и δ_4 положительны, то на основании (2.5) должно выполняться: $\delta_2 \le 0,429;\ \delta_4 \le 1,57$. Поэтому найденное оптимальное решение сохранится, если для новых значений C_2^* и C_4^* будут выполнены условия:

$-\infty \le C_2^* \le C_2 + 0,429$; $-\infty \le C_4^* \le C_4 + 1,57$ или $-\infty \le C_2^* \le 5,429; -\infty \le C_4^* \le 12,57$.

Определим теперь диапазоны изменения коэффициентов C_1 и C_3 целевой функции при базисных переменных x_1 и x_3, в пределах которых оптимальное решение сохраняется. Для решения этой задачи необходимо составить выражения симплекс-разностей для всех свободных переменных. Для переменных x_2 и x_4 они по-прежнему будут определяться формулами (2.4), а для переменных s_1 и s_3 будут иметь вид:

$$\Delta_{s1} = 0 - C_1 \cdot 1,429 - 0 \cdot (-0,871) - C_3 \cdot (-0,429) = 0 - 4 \cdot 1,429 + 9 \cdot 0,429 = -1,86\,, \qquad (2.6)$$

$$\Delta_{s3} = 0 - C_1 \cdot (-0,143) + 0 \cdot (0,571) - C_3 \cdot 0,143 = 0 + 4 \cdot 0,143 - 9 \cdot 0,143 = -0,714\,.$$

Для определения диапазона изменения C_1 предположим, что исходное значение C_1 изменяется на некоторую величину δ_1. Формулы (2.4) и (2.6) примут вид:

$$\Delta'_2 = C_2 - (C_1 + \delta_1) \cdot 0,714 - C_3 \cdot 0,28 = 5 - (4 + \delta_1) \cdot 0,714 - 9 \cdot 0,286 = -0,429 - \delta_1 \cdot 0,714\,,$$

$$\Delta'_4 = C_4 - (C_1 + \delta_1) \cdot (-0,714) - C_3 \cdot 1,714 = 11 + (4 + \delta_1) \cdot 0,714 - 9 \cdot 1,714 = -1,57 + \delta_1 \cdot 0,714\,, \qquad (2.7)$$

$$\Delta'_{s1} = 0 - (C_1 + \delta_1) \cdot 1,429 - C_3 \cdot (-0,429) = 0 - 4 \cdot 1,429 + 9 \cdot 0,429 = -1,86 - \delta_1 \cdot 1,429\,,$$

$$\Delta'_{s3} = 0 - (C_1 + \delta_1) \cdot (-0,143) - C_3 \cdot 0,143 = 0 + 4 \cdot 0,143 - 9 \cdot 0,143 = -0,714 + \delta_1 \cdot 0,143\,.$$

Решение останется оптимальным при таких значения δ_1, при которых одновременно будут выполняться условия:

$$\Delta'_2 = -0,429 - \delta_1 \cdot 0,714 \le 0\,, \quad \Delta'_4 = -1,57 + \delta_1 \cdot 0,714 \le 0\,, \qquad (2.8)$$

$$\Delta'_{s1} = -1,86 - \delta_1 \cdot 1,429 \le 0\,, \quad \Delta'_{s3} = -0,714 + \delta_1 \cdot 0,143 \le 0\,.$$

Каждое из этих неравенств определяет диапазон изменения δ_1, при котором соответствующая симплекс-разность остается отрицательной:

$$\delta_1 \geq -0,429 / 0,714 = -0,601; \quad \delta_1 \leq 1,57 / 0,714 = 2,199;$$
$$\delta_1 \geq -1,86 / 1,429 = -1,302; \quad \delta_1 \leq 0,714 / 0,143 = 4,993. \qquad (2.9)$$

Как следует из (2.9), любое из приведенных условий будет выполнено, если

$$-0,601 \leq \delta_1 \leq 2,199.$$

Поэтому найденное оптимальное решение будет сохраняться, если новое значение коэффициента C_1^* будет находиться в диапазоне $C_1 - 0,601 \leq C_1^* \leq C_1 + 2,199$ или для нашего примера $3,399 \leq C_1^* \leq 6,199$.

Для определения диапазона изменения коэффициента C_3, при котором сохраняется оптимальное решение, формулы (2.6) нужно представить в виде:

$$\Delta_2' = C_2 - C_1 \cdot 0,714 - (C_3 + \delta_3) \cdot 0,286 = -0,429 - \delta_3 \cdot 0,286,$$
$$\Delta_4' = C_4 - C_1 \cdot (-0,714) - (C_3 + \delta_3) \cdot 1,714 = -1,57 - \delta_3 \cdot 1,714,$$
$$\Delta_{s1}' = 0 - C_1 \cdot 1,429 - (C_3 + \delta_3) \cdot (-0,429) = -1,86 + \delta_3 \cdot 0,429, \qquad (2.10)$$
$$\Delta_{s3}' = 0 - C_1 \cdot (-0,143) - (C_3 + \delta_3) \cdot 0,143 = -0,714 - \delta_3 \cdot 0,143.$$

Т.к. для сохранения оптимальности решения все симплекс-разности должны оставаться отрицательными, то на основании (2.10) можно составить условия, аналогичные условиям (2.8) и (2.9). В результате получим:

$$\delta_3 \geq -0,429 / 0,286 = -1,5;$$
$$\delta_3 \geq -1,57 / 1,714 = -0,916;$$
$$\delta_3 \leq 1,86 / 0,429 = 4,336;$$
$$\delta_1 \geq -0,714 / 0,143 = -4,993.$$

Поэтому найденное оптимальное решение будет сохраняться, если новое значение коэффициента C_3^* будет находиться в диапазоне $C_3 - 0,916 \le C_3^* \le C_3 + 4,336$ или для нашего примера $8,084 \le C_3^* \le 13,336$.

Отметим, что диапазон изменения каждого коэффициента целевой функции определяется в предположении, что остальные сохраняют свое значение. Можно было бы каждому коэффициенту C_i в выражениях (2.4), (2.6) для симплекс-разностей задать вариации δ_i и из условия сохранения отрицательности симплекс-разностей построить систему неравенств для учета совместного изменения коэффициентов целевой функций. Однако решить такую систему неравенств непросто. Поэтому обычно ограничиваются определением диапазона изменения конкретного коэффициента целевой функции при сохранении значений остальных.

Анализ влияния свободных членов ограничений

Для определения чувствительности оптимального решения при изменении свободных членов ограничений обратимся к исходной постановке задачи (2.2) и запишем ее ограничения в канонической форме в виде:

$$\begin{aligned}
&(1)\, 1 \cdot x_1 + 1 \cdot x_2 + 1 \cdot x_3 + 1 \cdot x_4 + 1 \cdot s_1 + 0 \cdot s_2 + 0 \cdot s_3 = 15 + \theta_1, \\
&(2)\, 7 \cdot x_1 + 5 \cdot x_2 + 3 \cdot x_3 + 2 \cdot x_4 + 0 \cdot s_1 + 1 \cdot s_2 + 0 \cdot s_3 = 120 + \theta_2, \\
&(3)\, 3 \cdot x_1 + 5 \cdot x_2 + 10 \cdot x_3 + 15 \cdot x_4 + 0 \cdot s_1 + 0 \cdot s_2 + 1 \cdot s_3 = 100 + \theta_3 .
\end{aligned} \qquad (2.11)$$

Здесь переменными θ_1, θ_2, θ_3 обозначены возможные отклонения свободных членов ограничений от их первоначального значения. Наличие таких отклонений равносильно тому, что при решении ЗЛП вместо переменных s_1, s_2 и s_3 фактически решается задача с переменными $s_1-\theta_1$, $s_2-\theta_2$ и $s_3-\theta_3$. Поэтому в ограничениях (2.3) на последней итерации вместо s_1, s_2 и s_3 нужно подставить, соответственно, $s_1-\theta_1$, $s_2-\theta_2$ и $s_3-\theta_3$. Тогда получим:

$$(1)\, x_1 \to +7,143 = 1 \cdot x_1 + 0,714 \cdot x_2 + 0 \cdot x_3 - 0,714 \cdot x_4 + 1,429 \cdot (s_1 - \theta_1) + 0 \cdot (s_2 - \theta_2) - 0,143 \cdot (s_3 - \theta_3),$$

$$(2)\, s_2 \to +46,43 = 0 \cdot x_1 - 0,857 \cdot x_2 + 0 \cdot x_3 + 1,857 \cdot x_4 - 8,71 \cdot (s_1 - \theta_1) + 1 \times (s_2 - q_2) + 0,571 \cdot (s_3 - \theta_3), \quad (2.12)$$

$$(3)\, x_3 \to +7,857 = 0 \cdot x_1 + 0,286 \cdot x_2 + 1 \cdot x_3 + 1,714 \cdot x_4 - 0,429 \cdot (s_1 - \theta_1) + 0 \cdot (s_2 - \theta_2) + 0,143 \cdot (s_3 - \theta_3).$$

Переменные θ_1, θ_2, θ_3 могут принимать некоторые произвольные значения, но такие, чтобы все переменные оптимального решения оставались ≥ 0. В рассматриваемом примере базисными переменными оптимального решения являются x_1, s_2 и x_3. Используя (2.12), выразим их через свободные переменные x_2, x_4, s_1 и s_3 и переменные θ_1, θ_2, θ_3, а затем примем свободные переменные равными нулю. В результате получим:

$$x_1 = 7,143 - 1,429 \cdot (0 - \theta_1) + 0,143 \cdot (0 - \theta_3) \geq 0,$$

$$s_2 = 46,43 + 8,71 \cdot (0 - \theta_1) + 1 \cdot \theta_2 - 0,571 \cdot (0 - \theta_3) \geq 0,$$

$$x_3 = 7,857 + 0,429 \cdot (0 - \theta_1) - 0,143 \cdot (0 - \theta_3) \geq 0,$$

или

$$x_1 = 7,143 + 1,429 \cdot \theta_1 - 0,143 \cdot \theta_3 \geq 0, \quad (2.13)$$

$$s_2 = 46,43 - 8,71 \cdot \theta_1 + 1 \cdot \theta_2 + 0,571 \cdot \theta_3 \geq 0,$$

$$x_3 = 7,857 - 0,429 \cdot \theta_1 + 0,143 \cdot \theta_3 \geq 0.$$

Формулы (2.13) определяют систему неравенств, которым должны удовлетворять возможные *одновременные* изменения свободных членов ограничений при сохранении оптимального решения.

В программных средствах, реализующих решение ЗЛП, может предусматриваться выполнение анализа чувствительности при изменении какого-либо одного свободного члена ограничений. Например, для определения диапазона изменения свободного члена в первом ограничении (2.11) следует в формуле (2.13) принять $\theta_2 = \theta_3 = 0$. Тогда получим:

$$7,143 + 1,429 \cdot \theta_1 \geq 0 \text{ или } \theta_1 \geq -7,143 / 1,429 = -5;$$

$$46,43 - 8,71 \cdot \theta_1 \geq 0 \text{ или } \theta_1 \leq 46,43 / 8,71 = 5,331;$$

$$7,857 - 0,429 \cdot \theta_1 \geq 0 \text{ или } \theta_1 \leq 7,857 / 0,429 = 18,315.$$

Поэтому свободный член b_1, равный в исходной постановке 15, может изменяться в диапазоне $b_1 - 5 \le b_1^* \le b_1 + 5,331$ или $10 \le b_1^* \le 20,331$. При этом найденное оптимальное решение сохранится (в смысле сохранения набора базисных и свободных переменных). Аналогично можно выполнить анализ чувствительности при изменении других свободных членов ограничений.

Таким образом, исходными данными для анализа чувствительности оптимального решения, т.е. сохранения оптимального решения в смысле сохранения набора базисных и свободных переменных при изменении коэффициентов целевой функции и свободных членов ограничений в исходной постановке задачи, являются результаты последней итерации в решении ЗЛП, на которой получено оптимальное решение.

Анализ использования ресурсов

Анализ использования ресурсов предполагает определение размера их остатков при найденном оптимальном решении и влияния изменения ресурса на величину целевой функции. Для ответа на второй вопрос можно было бы для исходной задачи составить двойственную ЗЛП.

$$15 \cdot u_1 + 120 \cdot u_2 + 100 \cdot u_3 \to min$$

при ограничениях:

$$u_1 + 7 \cdot u_2 + 3 \cdot u_3 - 1 \cdot v_1 = 4 .$$

$$u_1 + 5 \cdot u_2 + 7 \cdot u_3 - 1 \cdot v_1 = 5 .$$

$$u_1 + 3 \cdot u_2 + 10 \cdot u_3 - 1 \cdot v_1 = 9 .$$

$$u_1 + 2 \cdot u_2 + 15 \cdot u_3 - 1 \cdot v_1 = 9 .$$

После ее решения следует проанализировать чувствительность целевой функции к изменению правых частей ограничений (т.е. ресурсов), т.к. оптимальные значения u_j^* можно рассматривать как коэффициенты при имеющихся ресурсах b_j, которые показывают, насколько изменится целевая функция при изменении соответствующего ресурса b_j на единицу, о

чем говорилось при рассмотрении двойственной задачи линейного программирования.

Однако в действительности нет необходимости составлять и решать двойственную задачу, т.к. в силу связи исходной и двойственной ЗЛП оптимальные значения u^*_j двойственных переменных можно определить по результатам последней итерации в решении ЗЛП, на которой получено оптимальное решение (2.3).

При приведении исходной задачи к канонической форме введены дополнительные переменные s_j (s_1, s_2, s_3 в примере (2.2)), которые физически определяют остаток ресурса b_j при производстве продукции. Если в найденном оптимальном решении s^*_j равно нулю, то это означает, что ресурс *j-го* типа использован полностью при оптимальных объемах продукции. Иначе s^*_j определяет величину недоиспользованного ресурса *j-го* типа. В примере (2.3) $s^*_2 = +46,43$, а остальные дополнительные переменные равны нулю. Поэтому можно сказать, что при найденных оптимальных объемах производства продукции ресурс второго типа недоиспользован на 46,43 единицы при общем количестве 120 единиц.

Если посмотреть на выражение для целевой функции в последней итерации симплекс-метода (выражение для *z(x)* в (2.3)), то можно заметить, что дополнительные переменные имеют нулевой коэффициент, если в оптимальном решении значение их отлично от нуля (для переменной s_2 в примере). Дополнительные переменные, которые в оптимальном решении являются свободными и имеют нулевое значение, в целевой функции имеют отличные от нуля коэффициенты (коэффициенты в целевой функции при s_1, s_3 в примере (2.3)).

Изменение ресурса, имеющего резерв, не изменит значения целевой функции, т.к. коэффициент в целевой функции для соответствующей дополнительной переменной равен нулю. Конечно, это изменение возможно в пределах, при которых сохраняется структура оптимального решения

(проще говоря, сохраняется номенклатура выпускаемой продукции). Определение этих пределов изменения правых частей ограничений мы уже рассмотрели.

Изменение же ресурсов, по которым нет резервов, повлияет на величину выпускаемой продукции и, возможно, номенклатуру и, следовательно, значение целевой функции. Величина изменения целевой функции при изменении такого ресурса на единицу определяется коэффициентом при соответствующей ресурсу дополнительной переменной в целевой функции на последней итерации. Так, для примера (2.3) при увеличении ресурса первого типа на единицу целевая функция уменьшится на 1,86 единицы (коэффициент при s_1 равен –1,86).

Коэффициенты при дополнительных переменных в целевой функции на последней итерации как раз и являются оптимальными значениями двойственных переменных u^*_j соответствующей двойственной ЗЛП, т.к. они определяют изменение значения целевой функции при изменении ресурса на единицу.

При приведении двойственной задачи к канонической форме вводятся дополнительные двойственные переменные v_i (в примере v_1-v_4). С экономической точки зрения эти переменные определяют изменение стоимости единицы продукции. Так как их значения могут быть больше или равны нулю, то они в конечном счете определяют, на сколько уменьшится величина целевой функции при принудительном увеличении выпуска соответствующей продукции на единицу. Коэффициенты при x^*_i в оптимальном решении исходной задачи как раз и определяют оптимальные значения v^*_i двойственной задачи. Так, для рассматриваемого примера (5.16) $x^*_1 = 7,143$; $x^*_3 = +7,857$ и для этих переменных на последней итерации коэффициенты в целевой функции равны нулю, а для x^*_2=x^*_4=0 коэффициенты, соответственно, равны –0,429 и –1,57. Поэтому $v^*_1 = 0$; $v^*_2 = 0,429$; $v^*_3 = 0$;

$v_4^* = 1,57$. Это говорит, например, о том, что принудительный выпуск единицы продукции 2-го типа повлечет уменьшение значения функции цели на 0,429.

В заключение отметим, что получение оптимального решения исходной задачи является лишь первым этапом в информационном обеспечении принятия окончательного решения. Не менее, а возможно, и более важным этапом является анализ оптимального решения, который позволяет судить об устойчивости оптимального решения при вариациях параметров задачи резервов ресурсов, последствиях субъективного принятия решений.

2.11. Сценарий решения ЗЛП в пакете QSB

В главном меню пакета QSB имеется специальная работа для решения задач линейного программирования (LP). При выборе ее на экран выводится меню возможных функций при решении ЗЛП, которое имеет следующий вид:

Вы работаете с Linear Programming (LP)!

Возможны следующие функции для LP.

Для ознакомления с работой выберите 1-ю функцию меню

Номер Функция

1 — Общие сведения о задаче LP

2 — Ввод новой задачи

3 — Чтение ранее сохраненной задачи с диска

4 — Выдача на экран и/или печать исходных данных задачи

5 — Решение задачи

6 — Запись на диск исходных данных задачи

7 — Изменение постановки задачи

8 — Выдача на экран и/или печать результатов решения

9 — Переход к меню возможных методов оптимизации

0 — Выход из системы QSB

При выборе 1-й функции на экран выдаются в краткой форме на английском языке возможности пакета при решении задач линейного программирования. При этом указывается, что программа пакета решает ЗЛП с размерностью до 40 переменных (не включая искусственные, т.е. дополнительные переменные) и до 40 ограничений. Подготовка задачи к решению сводится к представлению ее в форме:

$$Max\ +50,0000x_1 + 60,0000x_2$$

Subject to (при ограничениях):

$$(1) + 2,00000x_1 + 3,00000x_2 \le +180,000\ ,$$

$$(2) + 3,00000x_1 + 2,00000x_2 \le +150,000\ ,$$

$$(3) + 5,00000x_1 + 3,00000x_2 = +3,00000\ .$$

Переменные считаются неотрицательными. Их имена по умолчанию — $X_1, X_2, ..., Xn$, но пользователь может задать и свои имена длиной не более четырех символов.

Как видно, формат ввода фактически совпадает с исходной формулировкой задачи, и пользователю не нужно самому делать каких-либо преобразований постановки задачи.

Если пакет вводит дополнительные переменные, то при приведении исходной задачи к канонической (расширенной) они получают имена S_1, S_2 и т.д. Если задача приводится к *M*-задаче, то имена дополнительных переменных будут A_1, A_2 и т.д. При этом индекс у дополнительных переменных соответствует номеру ограничения, в котором дополнительные переменные введены.

При решении ЗЛП, как это видно из состава функций, пользователь имеет возможность сохранить введенные данные задачи, выполнить их изменение, получить решение, вывести на экран или печать результаты решения и исходные данные задачи.

Ввод данных новой задачи выполняется работой:

2 — Ввод новой задачи

При этом на экран выдается предложение ознакомиться с принятыми соглашениями при вводе задачи, которые сводятся к следующему:

$100,\ 100.0,\ +100,\ +100.0,\ 1E2,\ 1.0E+2$ – эквивалентно;

-123, -1.23E2, -1.23E+2 — эквивалентно;

>=, >, => и Є эквивалентны; <=, <, =< и є эквивалентны.

После набора данного нажимайте клавишу ENTER.

В пределах экрана можно корректировать данные, используя клавиши

BACKSPACE и Enter для установки курсора в нужную позицию.

Если Вы уверены в правильности данных на экране, то нажимайте пробел.

При вводе данных задачи нажимайте Esc для перехода к предыдущей странице и/ для перехода к следующей странице.

При вводе данных задачи с пользователем проводится диалог:

Вас интересует задача на max(1) или min(2) критерия?
(введите 1 или 2) <1 >
How many variables are there in your problem? (Enter number є 40)
(Сколько переменных в Вашей задаче?) <2 >
How many constraints are there in your problem? (Enter number є 40)
(Сколько ограничений в Вашей задаче ?) <3 >
Do you want to use the default variable names (X1,X2,...,Xn) (Y/N)?
(Вы хотите использовать имена переменных по умолчанию
(X1,X2,..,Xn)?) <Y >

В символах < > приведены примеры конкретных значений запрашиваемых данных для сформулированной выше ЗЛП.

Рекомендуемый сценарий решения задачи состоит в следующем:

а) выполняется функция «Ввод новой задачи» или функция «Чтение ранее сохраненной задачи с диска». В качестве тестовой можно прочитать с диска задачу из файла LPTEST (в реальной установке пакета может использоваться и другое имя);

б) выполняется функция «Выдача на экран и/или печать исходных данных задачи»;

в) если данные задачи в чем-то не устраивают, то выполняется функция «Изменение постановки задачи ...» и вновь выполняется пункт б);

г) сформулированная задача сохраняется на диске функцией «Запись на диск исходных данных задачи»;

д) выполняется решение задачи функцией «Решение задачи»;

е) при успешном решении задачи выполняется функция «Выдача на экран и/или печать результатов решения» или повторяется работа с задачей, начиная с пункта б).

При выполнении функции «Решение задачи» пользователю предлагается выбрать режим решения:

Option

1 — Solve and display the initial tableau — решение и выдача на экран начальной таблицы,

2 — Solve and display the final tableau — решение и выдача на экран заключительной таблицы,

3 — Solve and display the initial and final tableaus — решение и выдача на экран начальной и заключительной таблицы,

4 — Solve and display every tableau — решение и выдача на экран каждой таблицы по ходу решения задачи,

5 — Solve without displaying any tableau — решение без выдачи таблиц,

6 — Возврат к меню функций.

Обычно целесообразно выбрать 4-й режим, который обеспечивает пошаговую выдачу данных. Если выполняется соотношение: N+N1+N2+N3·2 ≤ 9, где N — число переменных, N1 — число ограничений типа «є» (<,<=,=<), N2 — число ограничений типа «=», N3 — число ограничений типа «Є» (>, >=, =>)), то на каждом шаге выдается текущая симплекс-таблица. При этом подсвечиваются переменные, которые вводятся или выводятся из базисных, а после текущей таблицы выдается сообщение:

Current objective function value (Max.) = <значение>

< Highlighted variable is the entering or leaving variable >

Entering: <имя переменной$_1$> Leaving: <имя переменной$_2$>

Последней фразой явно указывается, что переменная с <имя переменной$_1$> вводится в число базисных, а переменная с <имя переменной$_2$> выводится из числа базисных и становится свободной.

Если задача не имеет решения, то выдается сообщение:

** No feasible solution!!!

После выдачи финальной таблицы на экран автоматически выдаются следующие функции (options), которые можно использовать при выдаче результатов решения:

Option Menu for Displaying and/or Printing the Final Solution to AMC
You have the following options available for displaying
or printing the final solution. If you want to print the
solution, make sure that the printer is ready.

Option:

1 — Вывод на экран только решения,

2 — Display the solution and sensitivity analysis — вывод на экран решения и анализа чувствительности,

3 — Display/print the solution — вывод на экран/ печать решения,

4 — Display/print the solution and sensitivity analysis — вывод на экран/ печать решения и анализа чувствительности,

5 — Возврат к меню функций.

При выборе функции

«1 — Вывод на экран только решения»

на экран будут выданы результаты решения в форме табл. 2.18.

Таблица 2.18

Результаты решения задачи

Summarized Results for lptest Page : 1							
Variables		Solution	Opportunity Cost	Variables		Solution	Opportunity Cost
No	Names			No	Names		
1	X1	0.0000	50.0000	4	S2	148	0.0000
2	X2	1.0000	0.0000	5	A3	0.0000	20.0000
3	S1	177.000	0.0000				

Maximum value of the OBJ = 60 Iters. = 2

В этой форме X1, X2 — переменные исходной задачи, S1, S2 и A3 — автоматически введенные пакетом дополнительные переменные при преобразовании исходной задачи к канонической форме и затем к М-задаче. В столбцах с именем Solution выдается полученное оптимальное значение управляемых переменных. В столбцах Opportunity Cost приводятся значения симплекс-разностей для итерации, на которой найдено оптимальное решение.

При использовании в меню выдачи результатов опции

«4 — Display the solution and sensitivity analysis»

на экран выводится сначала решение ЗЛП в приведенной выше форме, а затем результаты анализа чувствительности в виде двух таблиц:

Таблица 2.19

Sensitivity Analysis for OBJ Coefficients Page : 1							
C(j)	Min. C(j)	Original	Max. C(j)	C(j)	Min. C(j)	Original	Max. C(j)
C(1)	- Infinity	50.0000	100.0000	C(2)	30.0000	60.0000	+ Infinity

Таблица 2.20

Sensitivity Analysis for RHS Page : 1							
B(i)	Min. B(i)	Original	Max. B(i)	B(i)	Min. B(i)	Original	Max. B(i)
B(1)	3.0000	180.0000	100.0000	B(3)	0.0000	3.0000	180.0000
B(2)	2.0000	150.0000	+ Infinity				

Первая из приведенных таблиц определяет диапазон Min.C(j) – Max.C(j), в пределах которого могут изменяться значения коэффициентов целевой функции при сохранении оптимальности найденного решения. Так, в приведенной таблице определено, что коэффициент C(1) может изменяться от – до 100, а коэффициент C(2) — от 30 до +∞.

Вторая таблица (табл. 2.20) определяет диапазон Min.B(i) – Max.B(i) , в пределах которого могут изменяться значения правых частей ограничений при сохранении оптимальности найденного решения. Так, значение B(1)=180, заданное в исходной постановке задачи, может изменяться от 3 до 100. При этом найденное оптимальное решение сохранится.

Предположим теперь, что мы решили изменить постановку задачи и получить новое решение. В этом случае нужно выполнить следующую работу главного меню: «7 — Изменение постановки задачи».

На экран будет выдан список функций при выполнении этой работы в виде:

«Функции для изменений задачи АМС».

Option:

1 — Modify the objective function coefficients — изменение коэффициентов целевой функции,

2 — Modify one constraint — изменение заданного и последующих ограничений,

3 — Add one constraint — добавление одного ограничения,

4 — Delete one constraint — удаление одного ограничения,

5 — Add one variable — добавление одной переменной,

6 — Delete one variable — удаление одной переменной,

7 — Вывод на экран и/или печать исходных данных,

8 — Возврат к меню функций.

Как видно из состава функций изменения, пакет предоставляет широкие возможности для изменения постановки задачи. При выборе 1-й функции — изменение коэффициентов целевой функции можно будет изменить и другие данные задачи.

Окончание работы с пакетом производится выполнением работы главного меню: «0» — Выход из системы QSB.

Глава 3. Дискретное линейное программирование

Имеется достаточно много примеров оптимизационных задач, в которых все или некоторые переменные управления могут принимать значения из некоторого дискретного множества значений. Задачи такого типа называют задачами дискретного программирования.

Если множество управлений образуют их целочисленные значения, то соответствующие задачи называют задачами целочисленного программирования.

Если искомые управляемые переменные могут принимать лишь одно из двух возможных значений (Булевы переменные), то соответствующие задачи называют задачами булевского программирования.

С помощью Булевых переменных можно решать еще один класс задач дискретного программирования, в котором исходные управляемые переменные могут принимать значения лишь из фиксированного набора значений. В каждом наборе значений переменные могут быть целыми или дробными числами. Пусть, например, задача имеет две исходные управляемые переменные x_1 и x_2, а возможные наборы значений этих переменных: $1)\, x_1 = 3,5;\, x_2 = 7,8;\;\; 2)\, x_1 = 3;\, x_2 = 9,4;\, 3)\, x_1 = 6,5;\, x_2 = 8$. Другими словами, множество возможных исходных управлений состоит в данном примере из трех точек с указанными координатами. При решении таких задач вместо исходных управляемых переменных принимают Булевы управляемые переменные, число которых равно числу точек допустимого множества исходных управлений.

3.1. Целочисленное линейное программирование

Целочисленной задачей линейного программирования (ЦЗЛП) называется ЗЛП, в которой переменные полностью или частично подчиняются требованию целочисленности значений.

Примером ЦЗЛП может служить задача о закреплении судов за линиями перевозок.

Имеется *n* типов судов, которые нужно закрепить за *m* линиями перевозок. Месячный объем перевозок судна *j-го* типа *(j = 1, 2, ..., n*) по *i-й* линии *(i = 1, 2, ..., m)* равен a_{ij} единиц, а связанные с этим затраты — C_{ij} рублей. Определить число x_{ij} судов *j-го* типа, закрепленных за *i-й* линией, для обеспечения перевозки по этой линии не менее b_i единиц груза при минимальных суммарных расходах на перевозки, если известно, что число судов *j-го* типа равно N_j.

Математическая модель данной задачи:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}\,x_{ij} \to \min;$$

$$\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}\cdot x_{ij} \ge b_i;\ i=1,\ldots,m;\ \sum_{i=1}^{m} x_{ij} = N_j;\ j=1,\ldots,n;$$

$x_{ij} \ge 0$ и целые числа; $i=1,2,\ldots,m$; $j=1,2,\ldots,n$.

Как видно, математическая формулировка ЦЗЛП аналогична ЗЛП, только добавляется дополнительное требование целочисленности управляемых переменных. Поэтому естественно желание применить методы решения ЗЛП, например, симплекс-метод. Если получающееся оптимальное решение имеет целочисленные значения управляемых переменных, то задача является решенной. В противном случае нужно как-то учесть требование целочисленности. На первый взгляд, условие целочисленности можно учесть округлением до целого значения переменных, которые получились нецелочисленными. В действительности округление может существенно исказить результаты.

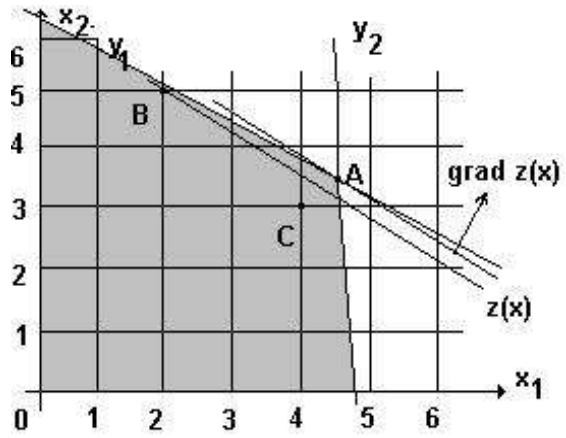

Рисунок 3.1.

Геометрическое представление ЦЗЛП

На рис. 3.1 приведен пример геометрического представления целочисленной задачи линейного программирования при $z(x) \to max$ с двумя ограничениями вида:

$$y_1 = a_{11} \cdot x_1 + a_{12} \cdot x_2 \geq 0;$$

$$y_2 = a_{21} \cdot x_1 + a_{22} \cdot x_2 \geq 0.$$

Точка *A* соответствует оптимальному решению рассматриваемой задачи как задачи линейного программирования. Из рисунка видно, что решение, соответствующее точке *A*, будет $y_1 = y_2 = 0$ и примерно $x_1 = 4,6$; $x_2 = 3,5$. Если округлить полученные значения (округлять в данном случае можно только в сторону уменьшения, чтобы не выйти из области допустимых решений), то получим предполагаемое оптимальное решение в точке *C(4,3)*. Но из рисунка также видно, что значение целевой функции будет больше в точке *B(2,5)*, чем в точке *C(4,3)*, соответствующей округленному значению оптимального решения. Поэтому оптимальное решение целочисленной задачи в общем случае нельзя получить из оптимального решения соответствующей ЗЛП и последующего округления нецелочисленных значений.

При решении ЦЗЛП используются два основных подхода к отысканию оптимального решения.

При первом из них, методы которого называются методами отсечения, вначале решается исходная задача как ЗЛП. Если найденное решение нецелочисленное, то вводится дополнительное ограничение, удовлетворяющее исходным ограничениям задачи, но отсекающее нецелочисленное решение. При этом дополнительное ограничение не должно отсекать точки с целочисленными координатами, первоначально принадлежащие области допустимых решений. Например, для рис. 3.1 таким дополнительным ограничением должно быть линейное ограничение, отсекающее точку *A* (например ограничение $x_1 \leq 4$). Одним из наиболее известных методов этого подхода является *метод Гомори*.

Второй подход к решению ЦЗЛП использует так называемые методы возврата. Один из наиболее распространенных методов этого подхода — *метод ветвей и границ.* В методах возврата также вначале решается исходная задача как ЗЛП. Затем формируется семейство связанных, но различных задач линейного программирования.

Метод Гомори

Этот метод предусматривает определенный порядок построения дополнительного ограничения, отсекающего вершину симплекса, если найденное в ней оптимальное решение не является целочисленным.

Алгоритм метода Гомори состоит в следующем:

1. Решается ЗЛП с использованием симплекс-метода. Если полученное решение целочисленно, расчет заканчивается. Если имеются нецелочисленные значения у компонентов оптимального решения, то переходим к шагу 2.

2. Выбирается нецелая базисная переменная x_k, которая имеет наибольшую дробную часть.

3. На основании последней симплекс-таблицы (решения ЗЛП), коэффициенты которой обозначим $a_{ji,}$ записывается уравнение, определяющее значение базисной переменной x_k:

$$x_k = x_k^* - \sum_{\substack{i=1 \\ i \neq k}}^{n} a_{ki} \cdot x_i, \tag{3.1}$$

где x_k^* — найденное нецелое оптимальное значение x_k (свободный член ограничения, из которого находится базисная переменная x_k).

Так как коэффициенты при остальных базисных переменных в строке для базисной переменной x_k равны нулю, то i фактически соответствует только свободным переменным, т.е. второе слагаемое образует небазисные (свободные) переменные, умноженные на коэффициенты при них в *k-й* строке таблицы.

4. Строится дополнительное ограничение:

$$\sum_{\substack{i=1 \\ i \neq k}}^{n} -\{a_{ki}\} \cdot x_i \leq -\{x_k^*\}, \tag{3.2}$$

где $\{a_{ki}\}$, $\{x_k^*\}$ — дробные части соответствующих чисел.

Вводится дополнительная переменная для преобразования (3.2) в равенство:

$$\sum_{\substack{i=1 \\ i \neq k}}^{n} -\{a_{ki}\} \cdot x_i + x_{n+1} = -\{x_k^*\}. \tag{3.3}$$

С учетом дополнительного ограничения вновь решается задача симплекс-методом. Построение дополнительных ограничений и расчет заканчиваются, когда требование целочисленности решения будет выполнено.

Поясним, почему дополнительное ограничение имеет вид (3.1) или (3.2). Симплекс-таблица, соответствующая оптимальному решению, фактически определяет вид ограничений ЗЛП для опорного плана, соответствующего вершине симплекса, в которой функция цели достигла экстре-

мума. Формула (3.1) определяет одно из этих ограничений. Запишем его в виде:

$$x_k + \sum_{\substack{i=1 \\ i \neq k}}^{n} a_{ki} \cdot x_i = x_k^* . \tag{3.4}$$

Если к системе ограничений добавить ограничение, являющееся линейной комбинацией имеющихся, то найденное решение также будет удовлетворять новой системе ограничений. Поэтому добавим к имеющимся ограничениям еще одно вида (3.4) (умножение на 1 имеющегося ограничения также является линейной комбинацией).

Попробуем теперь учесть требование целочисленности переменных, входящих в добавленное ограничение. Из (3.4) следует, что x_k получится целым, если будет выполнено неравенство:

$$x_k + \sum_{\substack{i=1 \\ i \neq k}}^{n} [a_{ki}] \cdot x_i \leq [x_k^*], \tag{3.5}$$

где $[a_{ki}]$, $[x_k^*]$ — целые части соответствующих чисел. Например $[-3,6] = -4$; $[2,7]=2$.

Представим теперь $a_{ki} = [a_{ki}] + \{a_{ki}\}; x_k = [x_k] + \{x_k\}$, где { } означает дробную часть числа, которая ≥0, но меньше 1. Например, $\{-3,6\} = 0,4$; $\{2,5\} = \{-2,5\} = 0,5$. Поэтому ограничение (3.6) можно представить в виде:

$$x_k + \sum_{\substack{i=1 \\ i \neq k}}^{n} ([a_{ki}] + \{a_{ki}\}) \cdot x_i = [x_k^*] + \{x_k^*\} . \tag{3.6}$$

Если теперь из введенного дополнительного ограничения (3.5) вычесть (3.6), то получим (3.2).

Рассмотрим следующий пример.

Найти $max\ z(x) = 0,7 \cdot x_1 + x_2$

при ограничениях:

$$(1)\ 0,65 \cdot x_1 + 1 \cdot x_2 \leq 6,5 , \tag{3.7}$$

$$(2)\ 17,5 \cdot x_1 + 1 \cdot x_2 \le 84\ ,$$

x_1, $x_2 \ge 0$ и должны быть целыми числами.

Решение этой задачи как задачи линейного программирования дает следующие результаты (они получены с помощью пакета QSB):

Таблица 3.1

Final tableau (Total iteration = 2)

Basis	C(j)	X_1	X_2	S_1	S_2	B(j)	B(j)
		0.7	1	0	0		A(j,l)
X_2	1	0	1	1.039	–0.039	3.510	0
X_1	0.7	1	0	–0.059	0.059	4.599	0
C(i) – z(i)		0	0	–0.997	–0.003	6.730	
*Big M		0	0	0	0	0	

(Max.) Optimal OBJ value = 6.729971

Полученное решение не является целочисленным. Выбираем переменную x_1, так как она имеет наибольшую дробную часть. Введенные пакетом дополнительные переменные s_1 и s_2 обозначим как x_3 и x_4. Теперь формируем новую ЗЛП.

Найти $max\ z(x) = 0,7 \cdot x_1 + x_2 + 0 \cdot x_3 + 0 \cdot x_4 + 0 \cdot x_5$

при ограничениях:

$$\begin{aligned} &(1)\ 0 \cdot x_1 + 1 \cdot x_2 + 1,039 \cdot x_3 - 0,039 \cdot x_4 = 3,510\ , \\ &(2)\ 1 \cdot x_1 + 0 \cdot x_2 - 0,059 \cdot x_3 + 0,059 \cdot x_4 = 4,599\ , \\ &(3)\ 0,059 \cdot x_3 - 0,059 \cdot x_4 + 1 \cdot x_5 = -0,599\ . \end{aligned} \qquad (3.8)$$

Дополнительное 3-е ограничение составлено на основе (3.8). Решаем теперь ЗЛП вида (3.8). Финальная симплекс-таблица для оптимального решения будет иметь вид:

Таблица 3.2

Final tableau (Total iteration = 3)

Basis	C(j)	X_1	X_2	X_3	X_4	X_5	A_1	A_2	A_3	B(i)	B(i) A(i,j)
		0.7	1	0	0	0	–M	–M	–M		
X_2	1	0	1	1	0	–0.661	1	0	0.66	3.906	0
X_1	0.7	1	0	–0.059	0.059	1	0	1	–1	4.000	0
X_4	0	0	0	–1	1	–16.9	0	0	16.9	10.15	
C(j) – z(j) *Big M		0 0	0 0	–1 0	0 0	–0.039 0		–1 –1	–0.7 –1	0.039 –1	6.706 0

(Max.) Optimal OBJ value = 6.705949

Как и следовало ожидать, оптимальное значение целевой функции стало меньше, т.к. в решенной ЗЛП отсечена вершина симплекса предыдущей ЗЛП, в которой было оптимальное, но нецелочисленное решение. Из симплекс-таблицы (табл. 3.2) $x_1 = 4$, т.е. равно целому значению, но дробным остается $x_2 = 3{,}906$. Поэтому следует по полученной симплекс-таблице составить ограничения, выбрать базисную переменную x_2, для которой по соответствующей строке таблицы составить дополнительное ограничение по формуле (3.3), и решить вновь сформулированную ЗЛП. При этом x_2 окажется целым числом, но x_1 может снова оказаться дробным. Поэтому итерации следует продолжить.

Таких итераций может оказаться достаточно много, но можно показать, что алгоритм Гомори конечен. При этом на каждой итерации добавляется еще одна переменная задачи, т.е. размерность ЗЛП возрастает. Все это определяет достаточные вычислительные трудности при реализации алгоритма Гомори даже при использовании ЭВМ.

Метод ветвей и границ

Этот метод применим для решения как полностью, так и частично целочисленных задач дискретного программирования. Идея метода основана на следующих очевидных рассуждениях.

Пусть задача оптимизации задана в виде:

$$z(x) = \sum_{i=1}^{n} c_i x_i \to \max;$$

$$\sum_{i=1}^{n} a_{ji} x_i \le b_j; \quad j = 1,...,m; \tag{3.9}$$

$$x_i \ge 0 \text{ и целые числа}, i = 1,2,...,q; \tag{3.10}$$

$$x_l \ge 0,\ l = q+1,\ldots,n. \tag{3.11}$$

Предположим также, что для каждой целочисленной переменной x_j можно задать верхнюю U_i и нижнюю L_i границы, в пределах которых безусловно содержится оптимальное решение, т.е.

$$L_i \le x_i \le U_i. \tag{3.12}$$

Обычно $L_i = 0$, но может без потери общности принимать любое положительное целое значение. Для любой переменной x_i примем, что I_i есть некоторое целое число, удовлетворяющее условию:

$$L_i \le I_i \le U_i - 1. \tag{3.13}$$

Тогда оптимальное решение x^* задачи (3.9), (3.11), (3.12) может и не быть целочисленным, но должно удовлетворять либо ограничению:

$$x_i^* \le I_i + 1, \tag{3.14}$$

либо ограничению:

$$x_i^* \ge I_i. \tag{3.15}$$

На основании этих соображений осуществляется ветвление задач. Ограничения (3.14) и (3.15) рассматриваются не для всех переменных, а только для одной x_k, например, имеющей наибольшую дробную часть. Ограничения (3.14) и (3.15) говорят о том, что вместо исходной задачи

(3.9) и (3.11) с ограничениями (3.12) нужно теперь рассматривать две задачи.

Исходная задача (3.9) и (3.11) с ограничениями $[x_k^*] + 1 \le x_k, x_k \le U_k$ и задача (3.12) при $i \neq k$;

Поэтому алгоритм метода ветвей и границ строится следующим образом:

1. Задаются границы L_i и U_i для целочисленных переменных задачи, которые определяют дополнительные ограничения типа (3.12) для исходной задачи. Получающаяся ЗЛП решается с использованием симплекс-метода. Если полученное решение $x^* = (x_1^* \dots x_n^*)$ целочисленно, то расчет заканчивается, в противном случае переходим к шагу 2.

2. Вычисляется значение $f(x^*) = z(x^*)$, которое является верхней границей оптимальных значений *z(x)* на целых решениях.

3. Из числа нецелочисленных координат по установленному правилу выбирается нецелочисленная переменная x_k, выделяется целая часть $[x_k^*]$ ее оптимального значения, и записываются два новых ограничения:

$$\left[x_k^*\right] + 1 \le x_k \le U_k;\ L_k \le x_k \le \left[x_k^*\right].$$

4. Решаются две задачи:

* ЗЛП 1 — представляющая собой сформулированную в пункте 1 задачу с дополнительными ограничениями $[x_k^*] + 1 \le x_k$. При этом исключается ограничение $L_k \le x_k$;

* ЗЛП 2 — представляющая собой сформулированную в пункте 1 задачу с дополнительными ограничениями $x_k \le [x_k^*]$. При этом исключается ограничение $x_k \le U_k$.

5. Из каждой задачи ЗЛП 1 и ЗЛП 2 вновь строится по две ЗЛП по правилам пункта 4. Ветвление задачи продолжается до тех пор, пока в одной из ветвей не будет найдено целое решение $X_н^* = (x_1^*, x_2^*, \dots, x_n^*)$. Получа-

ющееся при этом значение $f_н(X^*_н) = z(X^*_н)$ суть нижняя грань оптимального значения *z(x)* на целочисленных решениях.

6. Если значение $z(x^*)$ на нецелых решениях в оставшихся ветвях не превышает $f_н(X^*_н)$, расчет заканчивается, и *x** есть решение исходной задачи. Ветви, в которых это условие несправедливо, подлежат дальнейшему ветвлению.

Нецелая координата x_k оптимального решения *x** может выбираться в соответствии с одним из правил:

1. Это может быть координата с наименьшей или наибольшей дробной частью.

2. Это может быть координата с наименьшим или наибольшим индексом.

3. Приоритет координат может быть установлен лицом, решающим задачу.

Таким образом, алгоритм метода ветвей и границ достаточно прост, но требует многократного решения ЗЛП со все увеличивающимся количеством ограничений. Поэтому его реализация, как правило, требует применения средств вычислительной техники.

3.2. Сценарий решения целочисленной задачи в пакете QSB

В пакете QSB для решения целочисленных задач линейного программирования используется метод ветвей и границ. Сценарий работы пользователя при решении задач данного типа во многом похож на сценарии работы пользователя при решении задач других типов с помощью этого пакета.

При выборе в главном меню пакета задачи целочисленного линейного программирования на экран выдается меню функций при решении задач данного типа. Обычно при решении новой задачи сценарий работы пользователя включает выполнение функций ввода данных задачи, решения,

выдачи результатов решения, сохранения данных задачи на диске, изменение постановки задачи, повторное решение и т.д.

Рассмотрим сценарий работы пользователя при решении конкретной задачи:

$$z(x) = 3 \cdot x_1 + 3 \cdot x_2 + 13 \cdot x_3 \rightarrow max$$

при ограничениях:

$$\begin{aligned} &(1) - 3 \cdot x_1 + 6 \cdot x_2 + 7 \cdot x_3 \le 8 ; \\ &(2) 6 \cdot x_1 - 3 \cdot x_2 + 7 \cdot x_3 \le 8 ; \end{aligned} \qquad (3.16)$$

где $x_1, x_2 \ge 0$ и должны быть целыми числами.

Задача может быть сформулирована в любом виде, и никаких специальных представлений задачи при этом не требуется.

При выборе функции *Ввод новой задачи* сначала запрашивается имя задачи:

Введите имя задачи *aaa.*

Затем выдается справка о принятых в пакете соглашениях при вводе и предлагается ввести данные о задаче:

Вы решаете задачу на *max(1)* или *min(2)* <1 >.

How many variables are there in your problem<=(Enter number <= 20) <3 >.

(Как много переменных в вашей задаче<=(введи число <= 20)

How many constraints are there in your problem<= (Enter number <= 20) <2 >

(Как много ограничений в вашей задаче<=(введи число <= 20).

Do you want to use the default variable names (X1,X2,...,Xn) (Y/N) <Y>

(Будете использовать имена переменных по умолчанию).

Далее в диалоге запрашиваются характеристики переменных задачи.

Enter Integrality and Bounds for Variables.

Are all variables integer (Y/N)<=y.

Are all variables with 0-1 values (Y/N)<=N.

Если на этом шаге диалога ответить "Y", то все переменные будут принимать значения 0 или 1, т.е. будет решаться задача булевского программирования, но в рассматриваемом примере сформулирована задача целочисленного программирования, поэтому надо поставить ответ N.

Далее запрашивается, собираетесь ли вы определить границы изменения переменных:

Are you going to define bounds for variables (Y/N) <Y>.

При ответе "Y" предлагается определить для каждой переменной ее тип в столбце Integrality (I/C) (I — целочисленная переменная, C — любого типа), нижнюю (Lower bound) и верхнюю (Upper bound) границы. Определения границ задаются в форме:

Enter Integrality and Bounds for Variables

(Default values are continuous with lower bound 0 and upper bound 32000)

Var. no.	Var.	Integrality (I/C)	Lower bound	Upper bound
1	X1	<I>	<0 >	<5 >
2	X2	<I>	<0 >	<5 >
3	X3	<I>	<0 >	<5 >

Для рассматриваемого примера приняты приведенные границы. Границы переменных иногда могут быть известны из физического смысла задачи. В некоторых случаях их можно определить из анализа математической формулировки задачи. Однако при использовании пакета QSB можно не задавать границы переменных. В этом случае сначала границы принимаются в диапазоне от *0* до ∞ (т.е. верхняя граница считается неопределенной — *Infinity*). Фактически по умолчанию границы изменения переменных *0 — 32000*.

В методе ветвей и границ задание границ переменных — достаточно важный момент. При неудачном выборе границ задача вообще может не

иметь решения. Можно сделать несколько попыток, уменьшая верхнюю границу, начиная с *32000*.

После задания характеристик переменных при работе с пакетом запрашиваются данные о целевой функции и ограничениях. Формат ввода этих данных соответствует исходной формулировке задачи и для рассматриваемого примера имеет вид:

Entering the Coefficients of ILP Model Page 1

$$Max\ 3X_1 + 3X_2 + 13X_3$$

Subject to

$$(1) - 3X_1 + 6X_2 + 7X_3 \le 8,$$

$$(2)\, 6X_1 - 3X_2 + 7X_3 \le 8.$$

После ввода данных задачи вновь выводится меню функций. Обычно целесообразно выполнить функцию сохранения задачи и затем функцию:

5 — *Решение задачи.*

При этом выдается меню возможных режимов решения задачи, которыми могут быть:

1 — *Solve and display the first iteration* (решать и показывать 1-ю итерацию);

2 — *Solve and display each iteration* (решать и показывать каждую итерацию);

3 — *Solve without displaying any iteration* (решать без показа итераций);

4 — Change integer tolerance (default tolerance is .001) (изменить точность целого);

5 — Возврат к меню функций.

Четвертый режим позволяет задать, с какой точностью представления решение считается целочисленным. По умолчанию точность составляет 0,001. Это означает, что решение считается найденным, если оно отстоит от целого значения меньше, чем на 0,001. Поэтому если задать эту точ-

ность равной 1, то целочисленная задача превратится в обычную ЗЛП, т.к. любое дробное значение переменных будет отстоять от целого значения меньше, чем на 1.

Для рассматриваемого примера примем точность, задаваемую по умолчанию. Для того чтобы проследить реализацию метода ветвей и границ, выберем режим: 2 — *Solve and display each iteration.*

На первой итерации фактически решается исходная задача (3.16) с ограничениями:

$$0 \le x_i \le 5,\ i = 1,2,3\ , \qquad (3.17)$$

вытекающими из заданных границ для переменных. Обозначим эту задачу как *Задача а.* Результаты ее решения будут выведены в виде, приведенном в табл. 3.3. Оптимальное решение *(Solution)* будет: $x_1^* = 2,667$, $x_2^* = 2,667$, $x_3^* = 0$. Последняя строка этой таблицы говорит о том, что полученное решение не является целочисленным, а значение целевой функции равно $z(x^*) = 16$ и определяет верхнюю грань $f(x^*)$ для любого целочисленного решения.

Таблица 3.3

Задача а

Current Branch-and-Bound Solution--Iteration: 1 Page: 1					
Lw. bound	Variable	Up. bound	Variable	Solution	Obj. Fnctn
0 <=	X1	<= 5	X1	2.667	3.000
0 <=	X2	<= 5	X2	2.667	3.000
0 <=	X3	<= 5	X3	0.000	13.000
Noninteger solution with OBJ (Max.) = 16 ZL =-1E+20					

Теперь в соответствии с методом ветвей и границ из *Задачи а* автоматически порождаются две задачи линейного программирования. Так как переменные x_1 и x_2 имеют одинаковые дробные части, то выбирается первая из них — x_1. Порождаемыми задачами являются:

Задача аа: исходная задача (3.16) с ограничениями $[x_1^*] + 1 = 3 \le x_1 \le 5$, $0 \le x_2 \le 5, 0 \le x_3 \le 5$;

Задача аб: исходная задача (3.16) с ограничениями $0 \le x_1 \le [x_1^*] = 2$, $0 \le x_2 \le 5, 0 \le x_3 \le 5$.

Результаты решения этих двух задач приведены, соответственно, в таблицах 3.4 и 3.5. Сообщение в последней строке табл. 3.4 говорит о том, что при таких ограничениях *Задача аа* не имеет возможного решения. *Задача аб* имеет решение, но оно не является целочисленным. Поэтому из этой задачи порождается две новые задачи: *Задача аба* и *Задача абб*.

Для формирования ограничений этих задач на основании табл. 3.5 была выбрана нецелочисленная переменная X_3 и вместо прежних ее границ принято:

$[x_3^*] + 1 = 1 \le x_3 \le 5$ для *Задачи аба,*

$0 \le x_3 \le [x_3^*] = 0$, т.е. фактически принято $x_3 = 0$ для *Задачи абб.*

Таблица 3.4

Задача аа

Current Branch-and-Bound Solution--Iteration: 2 Page: 1					
Lw. bound	Variable	Up. bound	Variable	Solution	Obj. Fnctn
3 <=	X1	<= 5			
0 <=	X2	<= 5			
0 <=	X3	<= 5			
This branch has no feasible solution					

Таблица 3.5

Задача аб

Current Branch-and-Bound Solution--Iteration: 3 Page: 1					
Lw. bound	Variable	Up. bound	Variable	Solution	Obj. Fnctn
0 <=	X1	<= 2	X1	2.000	3.000
0 <=	X2	<= 5	X2	2.000	3.000
0 <=	X3	<= 5	X3	0.286	13.000
Noninteger solution with OBJ (Max.) = 15.71429 ZL =-1E+20					

Оставим на время ветвь с *Задачей абб* и займемся *Задачей аба*, результаты решения которой приведены в табл. 3.6.

Таблица 3.6

Задача аба

Current Branch-and-Bound Solution--Iteration: 4 Page: 1					
Lw. bound	Variable	Up. bound	Variable	Solution	Obj. Fnctn
0 <=	X1	<= 2	X1	0.333	3.000
0 <=	X2	<= 5	X2	0.333	3.000
1 <=	X3	<= 5	X3	1.000	13.000
Noninteger solutin with OBJ (Max.) = 15 ZL =-1E+20					

Данные табл. 3.6 говорят о том, что решение получилось нецелочисленным по переменным x_1 и x_2. Так, если дробные части их одинаковы, то выбираем переменную x_1 и порождаем две задачи:

Задачу абаа с границами (ограничениями) $[x_1^*] + 1 = 1 \le x_1 \le 2$,

Задачу абаб с границами (ограничениями) $0 \le x_1 \le [x_1^*] = 0$.

Первая из них, как видно из табл. 3.7, не имеет решения.

Таблица 3.7

Задача абаа

Current Branch-and-Bound Solution--Iteration: 5 Page: 1					
Lw. bound	Variable	Up. bound	Variable	Solution	Obj. Fnctn
1 <=	X1	<= 2			
0 <=	X2	<= 5			
1 <=	X3	<= 5			
This branch has no feasible solution					

Решение второй приведено в табл. 3.8, и оно оказалось нецелочисленным по переменной x_3.

Таблица 3.8

Задача абаб

Current Branch-and-Bound Solution--Iteration: 6 Page: 1					
Lw. bound	Variable	Up. bound	Variable	Solution	Obj. Fnctn
0 <=	X1	<= 0	X1	0.000	3.000
0 <=	X2	<= 5	X2	0.000	3.000
1 <=	X3	<= 5	X3	1.143	13.000
Noninteger solutin with OBJ (Max.) = 14.85714 ZL =-1E+20					

Поэтому на основании *Задачи абаб* порождается две новые задачи:

Задача абаба с границами (ограничениями) $[x_3^*] + 1 = 2 \le x_3 \le 5$,

Задача ababб с границами (ограничениями) $1 \le x_3 \le [x_3^*] = 1$.

Первая из них, как видно из табл. 3.9, не имеет решения.

Таблица 3.9

Задача абаба

Current Branch-and-Bound Solution--Iteration: 7 Page: 1					
Lw. bound	Variable	Up. bound	Variable	Solution	Obj. Fnctn
0 <=	X1	<= 0			
0 <=	X2	<= 5			
2 <=	X3	<= 5			
This branch has no feasible solution					

Решение второй приведено в табл. 3.10, и оно оказалось нецелочисленным по переменной x_2.

Таблица 3.10

Задача ababб

Current Branch-and-Bound Solution--Iteration: 8 Page: 1					
Lw. bound	Variable	Up. bound	Variable	Solution	Obj. Fnctn
0 <=	X1	<= 0	X1	0.000	3.000
0 <=	X2	<= 5	X2	0.167	3.000
1 <=	X3	<= 1	X3	1.000	13.000
Noninteger solutin with OBJ (Max.) = 13.5 ZL =-1E+20					

Поэтому на основании *Задачи ababб* порождается две новые задачи:

Задача ababба с границами (ограничениями) $[x_2^*] + 1 = 1 \le x_2 \le 5$,

Задача ababбб с границами (ограничениями) $0 \le x_2 \le [x_2^*] = 0$.

Первая из них, как видно из табл. 3.11, не имеет решения.

Таблица 3.11

Задача ababба

Current Branch-and-Bound Solution--Iteration: 9 Page: 1					
Lw. bound	Variable	Up. bound	Variable	Solution	Obj. Fnctn
0 <=	X1	<= 0			
1 <=	X2	<= 5			
1 <=	X3	<= 1			
This branch has no feasible solution					

Для второй, как видно из табл. 3.12, оптимальное решение получилось целочисленным.

Таблица 3. 12

Задача абаббб

Current Branch-and-Bound Solution--Iteration: 10 Page: 1					
Lw. bound	Variable	Up. bound	Variable	Solution	Obj. Fnctn
0 <=	X1	<= 0	X1	0.000	3.000
0 <=	X2	<= 0	X2	0.000	3.000
1 <=	X3	<= 1	X3	1.000	13.000
Integer feasible solution with OBJ (Max.) = 13 § ZL =-1E+20					

Полученное значение целевой функции $f_н\left(x_н^*\right) = z\left(x_н^*\right) = 13$ пока является нижней гранью целочисленных решений. В выводимых таблицах оно обозначается *ZL* и первоначально $ZL = -1E+20 = -10^{20}$, т.е. минимальному значению.

Порождение и анализ всех ветвей, начиная с *Задачи аба*, таким образом, закончен. Но осталась еще ветвь с *Задачей абб.* Оптимальное решение ее приведено в табл. 3.13.

Таблица 3.13

Задача абб

Current Branch-and-Bound Solution—Iteration: 11 Page: 1					
Lw. Bound	Variable	Up. Bound	Variable	Solution	Obj. Fnctn
0 <=	X1	<=2	X1	2.000	3.000
0 <=	X2	<= 5	X2	2.333	3.000
0 <=	X3	<= 0	X3	0.000	13.000
Current OBJ (Max.) = 13 <= ZL = 13					

Таблица 3.14

Результаты решения

Summary of Results for aa Page : 1							
Variable		Obj. Fnctn		Variable		Obj. Fnctn	
No.	Names	Solution	Coefficient	No.	Names	Solution	Coefficient
1	X1	0.000	3.000	3	X3	1.000	13.000
2	X2	0.000	3.000				
Maximum value of the OBJ = 13 Total iterations = 11							

Поскольку значение целевой функции *z(x)=13* удовлетворяет условию $f_н(X^*_н) \geq z(x)$, то для *Задачи абб* нет необходимости порождать новые задачи, т.к. в них значение целевой функции может быть только $\leq z(x) = 13$, а значит, меньше значения целевой функции для найденного целочисленного решения (напомним, что в нашем примере задача решается на max).

Поэтому найденное целочисленное решение является оптимальным. Оно выдается в виде, показанном в табл. 3.14.

Как следует из рассмотренного примера, для нахождения целочисленного оптимального решения пришлось выполнить 11 итераций, т.е. решить 11 задач линейного программирования. На рис. 4.2 показана схема ветвления задач линейного программирования в методе ветвей и границ, соответствующая рассмотренному примеру ЦЗЛП.

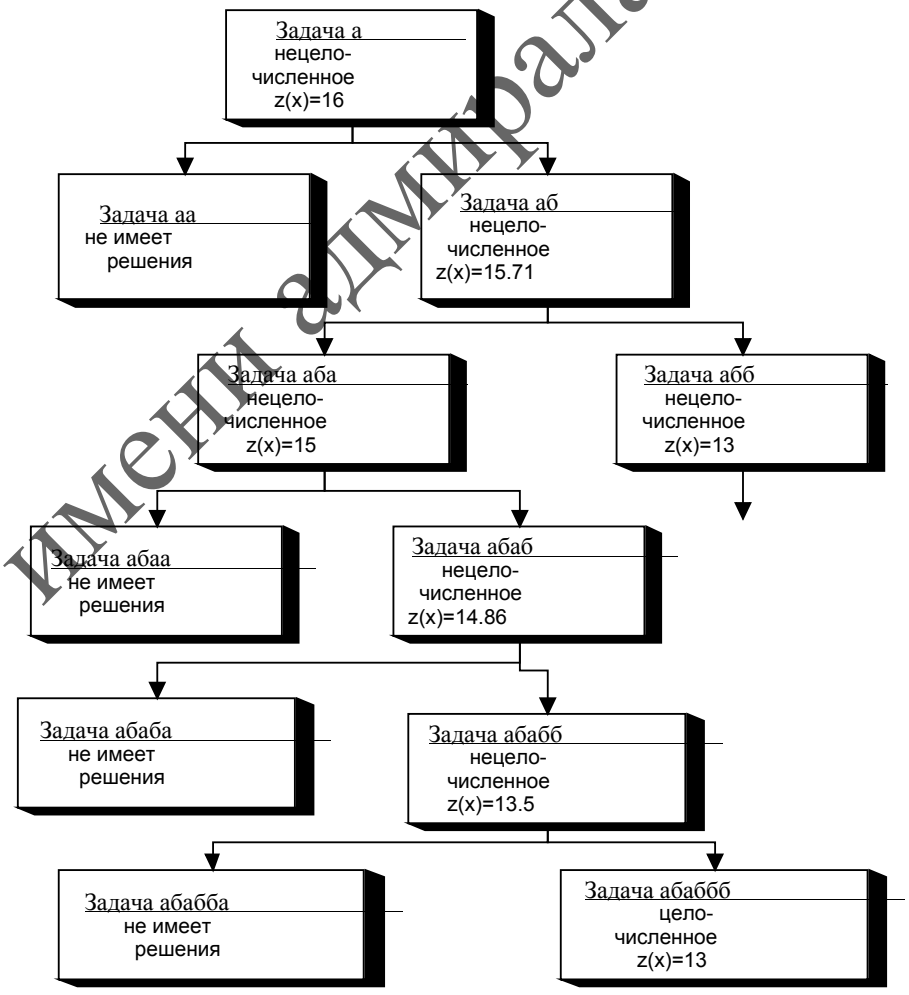

Рисунок 4.2. Схема ветвления в методе ветвей и границ

3.3. Булевское программирование

К задачам с Булевыми переменными обычно приводят распределительные задачи (задачи о назначениях) и задачи выбора вариантов.

Рассмотрим пример постановки распределительной задачи. Пусть требуется произвести назначение каждого из трех объектов O_1, O_2, O_3 на одну и только одну работу T_1, T_2, T_3 или T_4. Известна стоимость (расходы) C_{ij} выполнения каждым *i-м* объектом каждой *j-й* работы. Распределение должно быть таким, чтобы общая стоимость выполнения работ была минимальной.

В этой задаче *12* Булевых переменных x_{ij}, $i=1,2,3$; $j=1,2,3,4$. Если *i-й* объект назначается на *j-ю* работу, то $x_{ij}=1$, иначе –0. Математическая формулировка данной задачи будет иметь вид:

$$z(x)=\sum_{i=1}^{3}\sum_{j=1}^{4}c_{ij}\cdot x_{ij}\rightarrow min$$

при ограничениях:

$\sum_{i=1}^{3}x_{ij}=1$; $j=1,2,3,4$; $\sum_{j=1}^{4}x_{ij}=1$; $i=1,2,3$; все x_{ij} целые числа и ≥ 0.

В качестве примера задачи выбора вариантов рассмотрим следующий. Пусть, например, судоремонтное предприятие может выполнять пять видов ремонта. Прибыль от каждого вида ремонта известна. Известны расходы ресурсов при каждом виде ремонта, а также имеющееся в наличии количество ресурсов. Требуется выбрать такие виды ремонта, обеспеченные ресурсами, чтобы достигалась максимальная прибыль.

В этой задаче *5* Булевых переменных x_i, $i=1,2,3,4,5$. В качестве примера исходные данные для сформулированной задачи приведены в табл. 3.15.

Таблица 3.15

Исходные данные для задач с Булевыми переменными

Прибыль и ресурсы	Обозначение вариантов					Наличие
	X_1	X_2	X_3	X_4	X_5	
Прибыль	70	90	120	130	150	
ресурс 1	10.00	15.00	20.00	25.00	40.00	70.00
ресурс 2	20.00	10.00	15.00	30.00	30.00	80.00
ресурс 3	30.00	20.00	10.00	12.00	15.00	70.00

Обозначим через c_i прибыль при выборе *i-го* вида ремонта (из табл. 3.15 следует $c_1 = 70$, $c_2 = 90$ и т. д.). Обозначим через a_{ji} расходы *j-го* ресурса при выборе *i-го* варианта ремонта ($a_{11} = 10$, $a_{12} = 15$ и т. д.), а через b_j — имеющееся количество ресурса *j-го* типа ($b_1 = 70$, $b_2 = 80$ и т. д.).

С учетом принятых обозначений математическая модель задачи будет иметь вид:

$$\sum_{i=1}^{5} c_i \cdot x_i \to max$$

при ограничениях:

$$\sum_{i=1}^{5} a_{ji} \cdot x_i \le b_j;\ j = 1,2,3\ ;$$

все x_j — целые числа с нижней границей 0 и верхней границей *1*.

Использование Булевых переменных позволяет задавать ряд логических условий на выбираемые варианты. Рассмотрим несколько примеров ограничений такого типа, которые должны дополнять ограничения задачи.

Ограничение в виде $x_1 + x_3 = 1$ говорит о том, что в оптимальное решение должен входить или *1-й,* или *3-й* вариант.

Ограничение в виде $x_2 - x_5 = 0$ говорит о том, что если в оптимальное решение входит *2-й* вариант, то должен входить и *5-й* вариант, и наоборот.

Ограничение в виде $x_1 + x_2 - x_4 = 0$ говорит о том, что если в оптимальное решение входит *4-й* вариант, то должен входить или *1-й*, или *2-й* вариант.

Аналогично можно составить множество других условий. Все они основаны на том, что Булевы переменные могут принимать значения *0* или *1*.

Ранее было рассмотрено, что решение задач целочисленного линейного программирования, а значит и булевского программирования, в общем случае сводится к решению ряда задач линейного программирования. Если число Булевых переменных невелико, то может оказаться более простым применение комбинаторных методов решения, т.е. простого перебора возможных значений Булевых переменных и выбора такого из них, для которого функция цели принимает экстремальное значение. Поэтому, например, в пакете QSB имеется специальная работа для решения задач о назначениях.

3.4. Дискретное программирование

В предыдущих задачах поиска оптимального решения искомые переменные могли принимать любое, или целочисленное значение из заданного диапазона, или значения *0* или *1* (задачи с Булевыми переменными). Однако возможны постановки, когда переменная может принимать одно из ряда возможных значений, которые могут иметь произвольное значение. Например, переменная x_1 может принимать одно из значений *2.35, 3.75, 5.32*. Задачи в такой постановке называют задачами дискретного программирования.

Возьмем в качестве примера постановку задачи в следующей математической формулировке:

$$F_1 = c_1 \cdot x_1 + c_2 \cdot x_2 \to max \quad \text{— прибыль,}$$

$$-3 \cdot x_1 + 2 \cdot x_2 \le 12 \quad \text{— ограничение на ресурс } 1,$$

$$3 \cdot x_1 + 2 \cdot x_2 \le 60 \quad \text{— ограничение на ресурс } 2,$$

$$2 \cdot x_1 + x_2 \le 24 \quad \text{— ограничение на ресурс } 3$$

при $c_1=3$ и $c_2=2$, возможные значения $x_1 = \{2.35, 3.75, 5.32\}$, а $x_2 \geq 0$.

Введем дополнительные Булевы переменные d_1, d_2, d_3. Тогда можно записать дополнительное ограничение:

$$x_1 = 2,35 \cdot d_1 + 3,75 \cdot d_2 + 5,32 \cdot d_3$$

или

$$x_1 - 2,35 \cdot d_1 - 3,75 \cdot d_2 - 5,32 \cdot d_3 = 0.$$

При этом должно выполняться также еще одно ограничение или условие: $d_1 + d_2 + d_3 = 1$, т.е. одна и только одна из дополнительных Булевых переменных должна равняться 1. Таким образом, если к исходной постановке задачи добавить два приведенных ограничения, то будут удовлетворены все требования поставленной задачи дискретного программирования.

Глава 4. Решение задач линейного и целочисленного программирования в Excel

Решение линейных задач оптимизации в пакете QSB достаточно просто и наглядно, но требуется, чтобы ЗЛП имела не более *40* искомых переменных и не более *40* ограничений, а целочисленная задача — не более *20* искомых переменных и не более *40* ограничений. Эти ограничения отсутствуют при использовании версий электронных таблиц *Excel*, в которых имеется работа *Сервис — Поиск решения*. Достаточно подробно применение *Excel* при оптимизации решений рассмотрено в [1].

Рассмотрим применение *Excel* на примерах решения конкретных задач. Пусть задача линейного программирования сформулирована в следующей математической формулировке:

$$z(x) = c_1 \cdot x_1 + c_2 \cdot x_2 \to max \text{ — прибыль,}$$
$$-3 \cdot x_1 + 2 \cdot x_2 \le 12 \text{ — ограничение на ресурс } 1,$$
$$3 \cdot x_1 + 2 \cdot x_2 \le 60 \text{ — ограничение на ресурс } 2, \qquad (4.1)$$
$$2 \cdot x_1 + x_2 \le 24 \text{ — ограничение на ресурс } 3.$$

Сначала на отдельном листе *Excel* необходимо выполнить ввод исходных данных задачи в следующем формате (рис. 4.1).

	A	B	C	D	E	F
1	Исходные данные для линейных задач оптимизации					
2		Переменные				
3	Имя	X1	X2			
4	Значение					
5	Нижняя гр.					
6	Верхняя гр.					
7	Целочислен.					
8	Коэф. в целевой функции	1.00	2.00	Целевая функция:	Искать:	
9				0.00	max	
10		Ограничения				
11	Вид ограничения	Коэффициенты		Левая часть	Знак	Правая часть
12	"-3*x1+2*x2	-3.00	2.00	0.00	<=	12.00
13	"3*x1+2*x2	3.00	2.00	0.00	<=	60.00
14	"2*x1+x2	2.00	1.00	0.00	<=	24.00

Рисунок 4.1. Макет ввода данных для задачи оптимизации в Excel

Предлагаемый формат избыточен для исходной формулировки задачи, т.к. в ней не упоминаются верхние и нижние границы переменных, требование целочисленности. Поэтому строки *5, 6, 7* можно, конечно, исключить, но лучше их оставить, чтобы иметь общий трафарет ввода и для других постановок задачи.

Если в требуемой задаче не две, а большее число искомых переменных, то, очевидно, нужно добавить соответствующее число столбцов после столбца с именем *C*. Начиная со строки *12* можно записать требуемое число ограничений.

В трафарете ввода данных формульные зависимости записываются только для целевой функции (в примере в клетке D_9) и для левых частей ограничений (в примере в клетках D_{12}, D_{13}, D_{14}). При этом удобно воспользоваться функцией *СУММПРОИЗВ* (массив1; массив2; массив3; ...), которая перемножает соответствующие элементы заданных массивов и возвращает сумму произведений. Для рассматриваемого примера нужно задать:

=СУММПРОИЗВ($B$4:$C$4;$B$8:$C$8) для клетки D9

=СУММПРОИЗВ(B12:C12;$B$4:$C$4) для клетки D12

=СУММПРОИЗВ(B13:C13;$B$4:$C$4) для клетки D13

=СУММПРОИЗВ(B14:C14;$B$4:$C$4) для клетки D14

Далее работой *Сервис – Поиск решения* можно вызвать окно *Поиск решения*, показанное на рис. 4.2.

Из рис. 4.2 видно, что поля этого диалогового окна позволяют задать ячейку ($D$9) размещения целевой функции, диапазон ячеек ($B$4:$C$4), в которых будут формироваться значения искомых переменных.

С помощью радиокнопок можно задать тип критерия — максимальное или минимальное значение. Можно также найти решение, при котором целевая функция равна заданному значению.

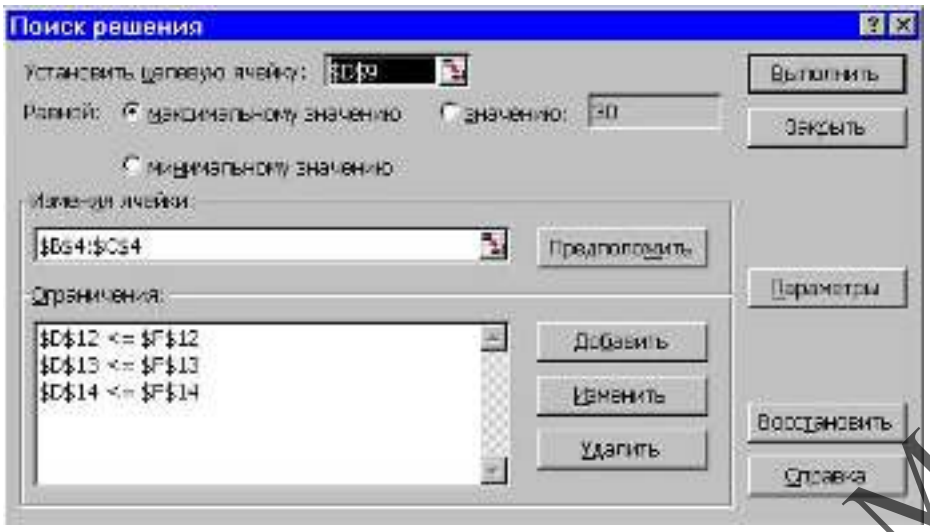

Рисунок 4.2. Окно задания данных для поиска решения

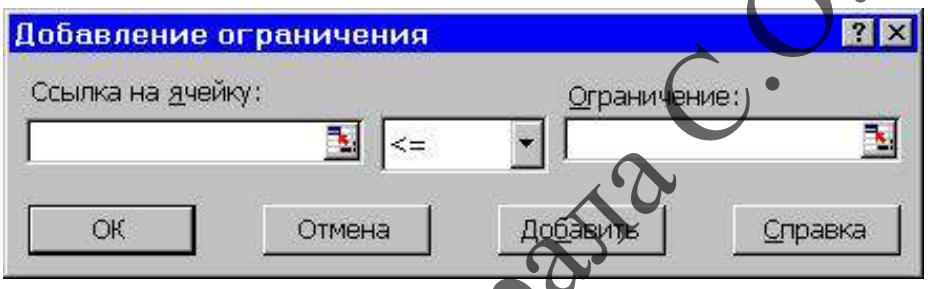

Рисунок 4.3. Добавление ограничений

В поле *Ограничения* выводятся введенные ограничения задачи. Если нажать кнопку *Добавить*, то выводится диалоговое окно, показанное на рис. 4.3, которое позволяет задать ссылку (имя) ячейки с формулой левой части ограничения, в среднем поле со списком выбрать знак ограничения, а в правом поле задать ссылку на ячейку со значением правой части ограничения. При нажатии кнопки *Добавить* в этом окне заданное ограничение добавляется к списку имеющихся.

С помощью диалогового окна, аналогичного окну на рис. 4.3, можно изменить имеющееся ограничение.

Если в окне на рис. 4.2 щелкнуть по кнопке *Параметры*, то на экран будет выведено окно *Параметры поиска решения*, показанное на рис. 4.4.

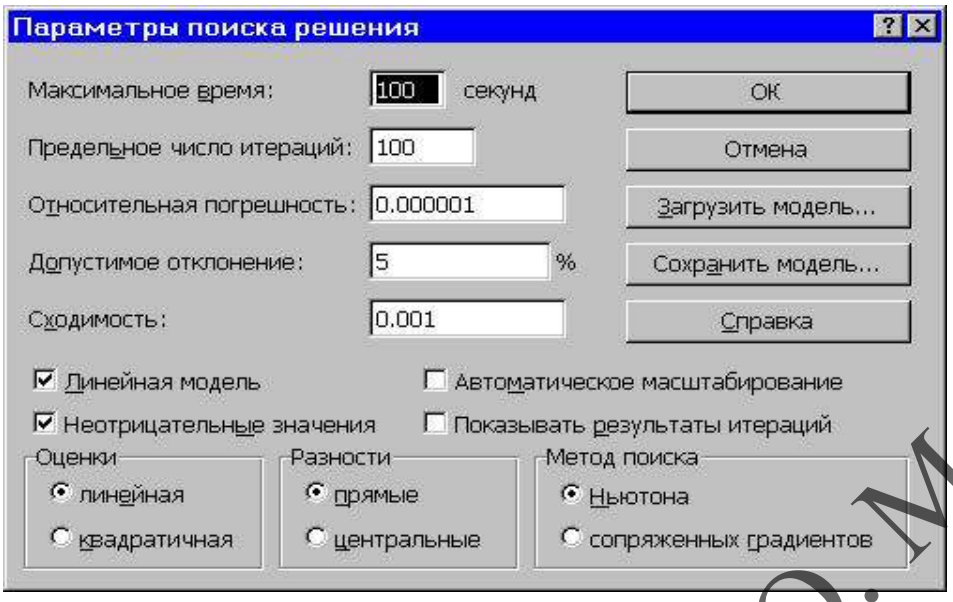

Рисунок 4.4. Окно параметров поиска

Подсказку о параметрах можно получить по кнопке *Справка*. Обычно для задачи линейного программирования достаточно установить флажки для параметров *Линейная модель* и *Неотрицательные значения*.

После закрытия окна *Параметры поиска решения* происходит возврат в окно *Поиск решения*. По кнопке *Выполнить* в окне на рис. 4.2 производится решение задачи.

По окончанию решения будет выведено окно *Результаты поиска решения*, показанное на рис. 4.5.

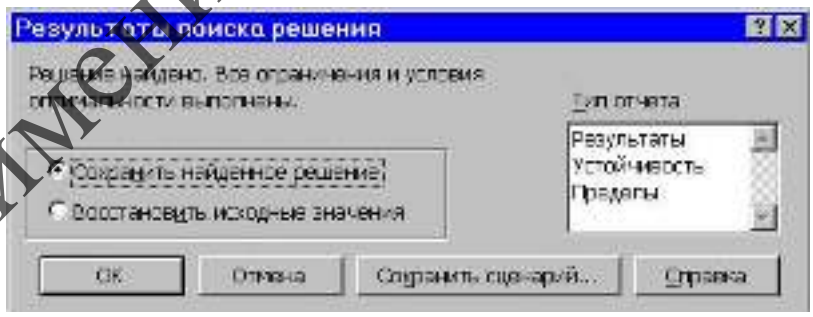

Рисунок 4.5. Окно по окончании решения

Если теперь сразу нажать кнопку *OK*, то на листе с постановкой задачи будут выведены найденные оптимальные значения переменных и значение целевой функции. Однако для более детального анализа задачи целесообразно в поле *Тип отчета* отметить щелчком левой кнопки мыши

все три типа отчетов и затем нажать кнопку *ОК*. Каждый из отчетов будет создан на отдельном листе с названиями ярлычков *Отчет по результатам, Отчет по устойчивости, Отчет по пределам*, которые размещаются перед листом с постановкой задачи.

Анализ полученных результатов решения является наиболее важной частью в решении задач оптимального управления. Выводимые в общем случае три отчета одинаковы для разных задач оптимизации.

Отчет по результатам. Для рассматриваемого примера вид этого отчета приведен на рис. 4.6.

Microsoft Excel 8.0 Отчет по результатам
Рабочий лист: [омс_5-7.xls]Постановка задачи
Отчет создан: 11.01.99 17:50:11
Целевая ячейка (Максимум)

Ячейка	Имя	Исходно	Результат
$D$9	Целевая функция:	0.00	32.57

Изменяемые ячейки

Ячейка	Имя	Исходно	Результат
$B$4	Значение X1	0.00	5.14
$C$4	Значение X2	0.00	13.71

Ограничения

Ячейка	Имя	Значение	формула	Статус	Разница
$D$12	"-3*'x1'+2*'x2'Левая часть	12.00	$D$12<=$F$12	связанное	0.00
$D$13	"3*x1+2*x2 Левая часть	42.86	$D$13<=$F$13	не связан.	17.14
$D$14	"2*x1+x2 Левая часть	24.00	$D$14<=$F$14	связанное	0.00

Рисунок 4.6. Отчет по результатам

Как видно, отчет состоит из трех достаточно очевидных таблиц. Комментария заслуживает только таблица *Ограничения*. В ней в столбце *Значение* выводятся фактически израсходованные ресурсы, соответствующие каждому из ограничений, а в столбце *Разница* — разность между имеющимися (правые части ограничений в постановке задачи) и фактически израсходованными ресурсами. Если эта разность равна нулю, т.е. ресурс используется полностью, то в столбце *Статус* выводится сообщение связанное, иначе — не связан.

Отчет по устойчивости. Для рассматриваемого примера вид этого отчета приведен на рис. 4.7.

Как видно, отчет состоит из двух таблиц. Таблица *Изменяемые ячейки* фактически содержит результаты анализа на устойчивость структуры оптимального решения при изменении коэффициентов целевой функции. Значения в столбцах *Допустимое увеличение* и *Допустимое уменьшение* показывают величину возможного отклонения принятого значения коэффициента в столбце *Целевой коэффициент*. Например, для коэффициента $C_1=1$ при переменной X_1 диапазон изменения будет определяться соотношением: $1-4 \le C_1 \le 1+3$, т.е. $-3 \le C_1 \le 4$. Для переменой X_2 возможное увеличение ее коэффициента в целевой функции составляет *1E+30*, т.е. 10^{30}, что соответствует ∞ в математическом обозначении.

Значения в столбце *Нормированная* (или редуцированная) *стоимость* показывают, на сколько изменится целевая функция при принудительном выпуске единицы продукции соответствующего типа.

Таблица *Ограничения* содержит результаты анализа на чувствительность значений правых частей ограничений, т.е. результаты анализа возможного изменения имеющихся ресурсов.

Microsoft Excel 8.0 Отчет по устойчивости
Рабочий лист: [омс_5-7.xls]Постановка задачи
Отчет создан: 11.01.99 17:50:14
Изменяемые ячейки

Ячейка	Имя	Результ. значение	Нормир. стоимость	Целевой коэффициент	Допустимое увеличение	Допустимое уменьшение
$B$4	Значение X1	5.14	0.00	1.00	3.00	4.00
$C$4	Значение X2	13.71	0.00	2.00	1E+30	1.50

Ограничения

Ячейка	Имя	Результ. значение	Теневая цена	Ограничение правая часть	Допустимое увеличение	Допустимое уменьшение
$D$12	"-3*x1+2*x2 Левая часть	12.00	0.43	12.00	36.00	48.00
$D$13	"3*x1+2*x2 Левая часть	42.86	0.00	60.00	1E+30	17.14
$D$14	"2*x1+x2 Левая часть	24.00	1.14	24.00	10.00	18.00

Рисунок 4.7. Отчет по устойчивости

Значения в столбцах *Допустимое увеличение* и *Допустимое уменьшение* показывают величину возможного отклонения принятого значения ресурса в столбце *Ограничение правая часть*. Например, для ресурса, задаваемого первым ограничением, начальное значение которого равно $b_1=12$, а допустимое увеличение и уменьшение составляют, соответственно, *36* и *48*, диапазон изменения будет $12-48 \le b_1 \le 12+36$, т.е. $-36 \le b_1 \le 48$.

В столбце *Теневая цена* выводится изменение значения целевой функции при изменении ресурса на единицу, т.е. условная цена стоимости единицы ресурса.

В столбце *Результ. значение* выводится фактически израсходованное количество ресурсов при оптимальном решении.

Отчет по пределам. Для рассматриваемого примера вид этого отчета приведен на рис. 4.8. Этот отчет показывает, в каких пределах может изменяться выпуск продукции, т.е. значения искомых переменных, вошедших в оптимальное решение, при сохранении структуры оптимального решения.

Как видно, отчет состоит из двух таблиц. Таблица *Целевое* просто выводит информацию о целевой функции.

Вторая таблица *Изменяемое* состоит из трех частей. В первой части приводятся найденные оптимальные значения переменных.

Во второй части в столбце *Нижний предел* выводится минимально возможное значение для каждой переменной, а в столбце *Целевой результат* — значение целевой функции, когда соответствующая переменная принимает минимальное значение при оптимальном значении остальных переменных.

Microsoft Excel 8.0 Отчет по пределам
Рабочий лист: [омс_5-7.xls]Постановка задачи
Отчет создан: 11.01.99 17:50:15

Целевое

Ячейка	Имя	значение
$D$9	Целевая функция:	32.57

Изменяемое

Ячейка	Имя	значение	Нижний предел	Целевое результат	Верхний предел	Целевое результат
$B$4	Значение X1	5.14	5.14	32.57	5.14	32.57
$C$4	Значение X2	13.71	0.00	5.14	13.71	32.57

Рисунок 4.8. Отчет по пределам

Третья часть таблицы аналогична второй, только в ней выводятся верхние пределы изменения переменных и для каждого из них значение целевой функции. В примере верхние значения совпадают с оптимальным решением, поэтому значение целевой функции везде одинаково и равно оптимальному значению.

Заметим, что информации, соответствующей отчету по пределам, в пакете QSB не выдается.

Вернемся снова к постановке задачи (4.1) и предположим, что для переменной x_1 задано условие целочисленности и известна нижняя и верхняя граница значений для этой переменной. В этом случае в формате ввода данных на рис. 4.1 нужно ввести в ячейки *B5, B6* значения нижней и верхней границы, а в клетке *B7* — задать текст *целое*. Задание значения упомянутых ячеек фактически нужно лишь для наглядного представления постановки задачи, т.к. в окне *Поиск решения* на рис. 4.2 нужно добавить дополнительные ограничения:

$B$4>=$B$5 x_1>=нижней границы,

$B$4<=$B$6 x_1<=верхней границы,

$B$4=целое x_1 целое.

При задании ограничений можно было бы и не делать ссылку на ячейку, например, $B$5, а просто непосредственно записать нужное значение.

При решении целочисленной задачи следует более внимательно отнестись к заданию параметров в окне на рис. 4.4. В первую очередь это касается параметра *Допустимое отклонение*, который служит для задания допуска на отклонение от оптимального целочисленного решения. При указании большего допуска поиск решения заканчивается быстрее. Например, если значение этого параметра *5%*, то решение считается целочисленным, если полученное в результате решения задачи линейного программирования оптимальное значение переменной x^* удовлетворяет условию $[x^*] - 0,05 \cdot x^* \le x^* \le [x^*] + 0,05 \cdot x^*$, где запись $[x^*]$ обозначает ближайшее целое для значения x^*. Другими словами, этот параметр определяет, с какой погрешностью нецелочисленное решение можно считать целочисленным.

Параметр *Относительная погрешность* (или точность) служит для задания точности, с которой определяется целевая функция. Поле должно содержать число из интервала от *0* (нуля) до *1*. Низкая точность соответствует введенному числу, содержащему меньшее количество десятичных знаков, чем число, используемое по умолчанию, например, *0,0001*. Высокая точность увеличит время, которое требуется для того, чтобы сошелся процесс оптимизации.

После задания исходных данных и параметров решение задачи производится по кнопке *Выполнить* в окне *Поиск решения*. По окончанию решения выдается окно *Результаты поиска решения*, показанное на рис. 4.5. Если решение найдено, то можно задать выдачу отчета *Результаты*. Другие виды отчетов для задач целочисленного программирования не выдаются.

Задачи булевского и дискретного программирования являются частным случаем целочисленного программирования. При этом искомые Булевы переменные могут принимать значения *0* или *1*, которые являются соответственно нижней и верхней границей каждой из них. В остальном решение таких задач ничем не отличается от решения задач целочисленного линейного программирования.

Таким образом, в *Excel* легко задать постановку всех видов задач линейной оптимизации. Формируемые результаты содержат не только значения искомых переменных и целевой функции, но и данные для анализа оптимального решения. *Excel* позволяет достаточно просто выполнить параметрический анализ задачи. Для этого достаточно для каждого нового значения интересуемого параметра задачи вновь выполнить решение и сохранить выполненный сценарий. Затем можно объединить результаты сценариев и получить сводные данные. Возможности *Excel* позволяют достаточно просто представить результаты решения и анализа оптимизационной задачи в графической форме и считать его удобным инструментом решения оптимизационных задач.

Глава 5. Транспортная задача линейного программирования

5.1. Постановка задачи

Рассмотрим следующую постановку задачи оптимального управления. В каждом из *n* пунктов отправления (*Sources*) имеется запас однородного груза в количестве S_i, $i = 1, 2, ..., n$. В *m* пунктах назначения (*Destinations*) известна потребность в этом грузе, и она составляет D_j, $j = 1, 2, ..., m$. Расходы на перевозку единицы груза (удельные расходы) из *i-го* пункта отправления в *j-й* пункт назначения известны и равны C_{ij}. Требуется определить количество груза x_{ij}, отправляемого из пункта отправления *i* в пункт назначения *j*, т.е. составить план перевозок $x = (x_{11}, x_{12}, ..., x_{ij}, ..., x_{nm})$, при котором общие расходы *z(x)* на перевозку будут минимальны.

Как следует из словесной постановки этой транспортной задачи, ее графически можно представить в виде сети, показанной на рис. 5.1.

Рисунок 5.1. Сеть транспортной задачи

Математическое описание этой задачи будет иметь вид:

$$z(x) = \sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij} \cdot x_{ij} \to min$$

при ограничениях:

$$\sum_{i=1}^{n} x_{ij} = D_j;\ j = 1,2,...,m\ ; \qquad (5.1)$$

$$\sum_{j=1}^{m} x_{ij} = S_i; i = 1,2,...,n \ . \qquad (5.2)$$

Первое ограничение говорит о том, что объемы перевозок из всех пунктов отправления в *j-й* пункт назначения должны в точности соответствовать потребности или заявке (*Demand*) в *j-м* пункте назначения. Второе ограничение требует, чтобы объем отправлений из *i-го* пункта отправления в точности соответствовал запасам (*Capacities* или *Supplies*) груза в нем. Чтобы ограничения (5.1) и (5.2) могли быть выполнены, должно, очевидно, выполняться соотношение:

$$\sum_{i=1}^{n} S_i = \sum_{j=1}^{m} D_j . \qquad (5.3)$$

Соотношение между количеством запасов груза в пунктах отправления и количеством заявок в пунктах назначения называется *балансовым условием*. Если условие (5.3) выполняется, то поставленная задача оптимизации называется транспортной задачей линейного программирования (ТЗЛП) с правильным балансом.

Если условие (5.3) для исходной постановки задачи не выполняется, то всегда можно ввести фиктивный (*Dummy*) пункт отправления или назначения. Для фиктивного пункта отправления объем фиктивного запаса груза S_{n+1} должен быть принят равным $\sum_{j=1}^{m} D_j - \sum_{i=1}^{n} S_i$, а для фиктивного пункта назначения объем фиктивной заявки D_{m+1} должен быть принят равным $\sum_{i=1}^{n} S_i - \sum_{j=1}^{m} D_j$. В первом случае $x_{n+1,j}$ будет определять объем неудовлетворенного спроса в *j-м* пункте назначения, а во втором величина $x_{i,m+1}$ будет определять объем невывезенного груза из *i-го* пункта отправления. Поскольку нет фактических перевозок из фиктивного пункта отправления или в фиктивный пункт назначения, то фиктивные перевозки не должны влиять на значение целевой функции. Поэтому коэффициенты стоимостей

фиктивных перевозок (соответствуют они доходам или расходам) принимаются равными нулю.

С учетом сказанного без ограничения общности можно считать, что в ТЗЛП балансовое условие всегда выполняется.

В приведенной постановке задачи известными являются расходы на перевозку, и потому ставится задача поиска min расходов. Если известны доходы от перевозок единицы груза по линии *(i,j)*, то следовало бы решать задачу на max доходов от перевозки. Поэтому в общем случае коэффициенты целевой функции могут задавать или расходы, или доходы *(Cost/Profit Coefficients)*, и, соответственно, задача решается или на *min*, или на *max*.

Существенной особенностью в приведенной постановке задачи является то, что речь идет о перевозках однородного груза. Поэтому модель поставленной задачи называют также *однопродуктовой моделью*. Запасы, имеющиеся в пунктах отправления, можно рассматривать как объемы производства определенной продукции предприятиями фирмы, расположенными в пунктах отправления. При этом заявки пунктов назначения можно рассматривать как требуемые объемы этой продукции, размещаемые на складах потребителей в пунктах назначения. Поэтому транспортную задачу можно рассматривать как задачу закрепления поставщиков к потребителям. В действительности часто предприятия являются «*многопродуктовыми*». Можно, конечно, разрабатывать отдельные планы перевозок по каждому виду продукции. Как правило, однако, экономически выгодно ограничить число поставщиков, снабжающих один склад. Поэтому фирмы при распределении продукции своих предприятий пользуются планами, включающими в явном виде всю номенклатуру продукции. Легко себе представить, что требуется немалая изобретательность, чтобы представить такую многопродуктовую модель в виде однопродуктовой или классической транспортной задачи.

Как видно из математической формулировки, транспортная задача является задачей линейного программирования, и ее можно было бы решать, например, симплекс-методом. Но она имеет существенную особенность ограничений, которая позволяет для нее использовать более простые методы решения.

Особенность ограничений ТЗЛП, во-первых, состоит в том, что все коэффициенты при переменных в ограничениях равны *1*. Во-вторых, в том, что не все $n+m$ уравнений ограничений являются линейно независимыми. Действительно, если сложить все уравнения (5.1) и все уравнения (5.2), то в силу (5.3) мы должны получить одно и то же. Поэтому условия (5.1) и (5.2) связаны одной линейной зависимостью, и только $n+m-1$, а не $n+m$ уравнений из (5.1) и (5.2) являются линейно независимыми. Это означает, что ранг системы уравнений (5.1), (5.2) равен $r=n+m-1$ и можно разрешить эти уравнения относительно $n+m-1$ базисных переменных, выразив их через оставшиеся свободные переменные. Количество свободных переменных при этом равно:

$$k = n \cdot m - (n + m - 1) = n \cdot m - m - (n - 1) = m \cdot (n - 1) - (n - 1) = (m - 1) \cdot (n - 1).$$

Поскольку оптимальное решение ЗЛП достигается в одной из вершин симплекса, в которой не менее k переменных должно обращаться в ноль, то в нашем случае по крайней мере k переменных из x_{ij} должно обратиться в ноль для оптимального плана перевозок. В любом опорном плане, соответствующем одной из вершин симплекса, не более r переменных должно быть отличными от нуля, а остальные должны быть равны нулю.

Указанные особенности и определяет класс ЗЛП, называемый транспортными задачами линейного программирования, хотя содержательная постановка задачи может быть совсем не связана с транспортным процессом.

Особенности ограничений ТЗЛП допускают пользоваться при их решении не симплексными таблицами, а более простыми. Существует не-

сколько модификаций методов решения ТЗЛП, с каждым из которых обычно связывают свой вид транспортной таблицы. Будем использовать вид табл. 5.1, применяемый в пакете QSB для решения ТЗЛП.

Таблица 5.1

Транспортная таблица

$A_n \setminus F_m$	F_1	…	F_m	Supplies	U(i)
A_1	C_{11}	…	C_{1m}	S_1	
	x_{11}		x_{1m}		
…				…	
A_n	C_{n1}	…	C_{nm}	S_n	
	x_{n1}		x_{nm}		
Demands	D_1	…	D_m	$\sum_{i=1}^{n} S_i = \sum_{j=1}^{m} D_j$	
V(j)					

В этой таблице $A_1,\ldots,A_n$ обозначают пункты отправления, а F_1, ..., F_m — пункты назначения. В верхней части каждой клетки *(i,j)* записываются стоимости перевозок C_{ij} единицы груза из пункта отправления A_i в пункт назначения F_j, а в нижней части — объемы перевозок x_{ij} между этими пунктами. В столбце *Supplies* (запасы) записываются объемы запасов груза для каждого пункта отправления A_i, а в строке *Demands* (заявки) — объемы заявок в каждом пункте назначения F_j.

Если в исходной постановке задачи балансовые условия (5.3) не выполняются, то добавляется либо строка с фиктивным пунктом отправления *Dummy*, либо столбец с фиктивным пунктом назначения *Dummy* с соответствующими значениями в столбце *Supplies* или строке *Demands*.

В столбце *U(i)* и в строке *V(j)* записываются потенциалы пунктов отправления и назначения, но о них мы поговорим позже при рассмотрении метода потенциалов при решении ТЗЛП.

Методы решения ТЗЛП можно разделить на две основные группы. В первой группе методов, которые наиболее часто используются, решение ТЗЛП, как любой задачи линейного программирования, состоит из двух

этапов: нахождения опорного плана и последующего улучшения его, пока не будет найден оптимальный. К этой группе относятся *распределительный метод* и *метод потенциалов*, которые далее будут рассмотрены подробно.

Во второй группе методов решения ТЗЛП сначала определяется некоторым образом наилучший план, возможно и не удовлетворяющий полностью заданным ограничениям, а затем постепенно корректируется так, чтобы оптимальный план соответствовал всем ограничениям. К этой группе методов относится, например, «*венгерский*» метод.

5.2. Нахождение опорного плана перевозок

В отличие от общего случая задачи линейного программирования, решение ТЗЛП всегда существует. Это вытекает из физических (смысловых) соображений постановки транспортной задачи. Очевидно, что в силу балансовых условий хоть какой-то допустимый план обязательно существует. Среди допустимых планов непременно найдется (возможно и не один) оптимальный, для которого функция цели максимальна (в задачах на *max*).

Могут быть использованы разные методы нахождения опорного плана. Их отличие принципиально состоит в том, — учитываются или нет стоимости перевозок при поиске опорного плана. Однако в любом случае используется та особенность ТЗЛП, что из n·m переменных $(m-1)\cdot(n-1)$ свободных переменных должны быть равны нулю для любого опорного плана. Если среди базисных переменных, число которых равно $n+m-1$, есть равные нулю, то такой план называется вырожденным.

Наиболее простой и часто применяемый метод нахождения опорного плана — метод «*северо-западного угла*» (*Nord West Corner — NWC*).

Рассмотрим сущность этого метода на конкретном примере.

Запасы груза в пунктах отправления (*Sources*):

A1: 100 A2: 200 A3: 400

Заявки на груз в пунктах назначения (*Destinations*):

F1: 200 F2:100 F3:150 F4:250

Стоимость перевозки единицы груза из пункта отправления в пункт назначения:

Из (*From*)	в (*To*)			
A1	F1: 5.000	F2: 4.000	F3: 5.000	F4: 6.000
A2	F1:3.000	F2: 3.000	F3: 6.000	F4: 6.000
A3	F1:2.000	F2: 5.000	F3: 7.000	F4: 8.000

Приведенные числовые данные стоимостей определяют значения коэффициентов C_{ij} целевой функции. Требуется найти оптимальный план перевозок, при котором общая стоимость перевозок минимальна, т.е.:

$$z(x) = \sum_{i=1}^{3}\sum_{j=1}^{4} c_{ij} \cdot x_{ij} \to min .$$

Приводить запись ограничений в виде (5.1), (5.2) не имеет смысла, т.к. все коэффициенты при переменных задачи равны *1*.

Приведенные исходные данные представим в форме транспортной табл. 5.2.

Таблица 5.2

Начальная транспортная таблица

$A_n \setminus F_m$	F_1	F_2	F_3	F_4	Supplies	U(i)
A_1	5	4	5	6	100	
A_2	3	3	6	6	200	
A_3	2	5	7	8	400	
Demands	200	100	150	250	700	
V(j)						

Таблица 5.3

Опорный план

$A_n \setminus F_m$	F_1	F_2	F_3	F_4	Supplies	U(i)
A_1	5	4	5	6		
	100				100	
A_2	3	3	6	6		
	100	100	0		200	
A_3	2	5	7	8		
			150	250	400	
Demands	200	100	150	250	700	
V(j)						

Перепишем табл. 5.2 (см. табл. 5.3) и будем заполнять ее клетки объемами перевозок, начиная с левой верхней клетки (1.1) («*северо-западного угла*» таблицы). При этом будем рассуждать следующим образом.

В пункте F_1 заявка равна *200* единицам. Максимально удовлетворим эту заявку запасами пункта A_1. Поэтому в клетку (1.1) задаем перевозку в объеме $x_{11} = 100$ единиц. Но остались неудовлетворенными еще *100* единиц заявки пункта F_1. Удовлетворим этот остаток заявки перевозкой из пункта A_2, т.е. запишем в клетку (2.1) $x_{21} = 100$ единиц. Мы полностью обеспечили перевозками заявку пункта F_1.

Теперь переходим к следующему пункту назначения F_2. Его заявку можно полностью удовлетворить остатком груза в пункте A_2. Поэтому в клетку (2.2) записываем $x_{22} = 100$ единиц.

Далее переходим к удовлетворению заявки пункта F_3. Т.к. запасы пунктов A_1 и A_2 полностью израсходованы, то удовлетворяем заявку пункта F_3 из запасов пункта A_3 — в клетку (3.3) записываем $x_{33} = 150$.

Заявку пункта F_4 мы можем полностью удовлетворить остатком запаса в пункте A_3, т.е. принимаем $x_{34} = 250$.

Таким образом, мы составили опорный план перевозок, удовлетворяющий условиям задачи, в котором значение клетки *(i,j)* определяет объем перевозки x_{ij} между пунктами A_i и F_j.

В любом опорном плане, соответствующем вершине симплекса, из общего числа переменных $n \cdot m = 3 \cdot 4 = 12$ базисными являются $n \cdot m - (m-1) \cdot (n-1) = 12 - 3 \cdot 2 = 6$. В нашем случае клеток с ненулевыми перевозками всего *5*. Поэтому составленный план является вырожденным, т.к. одна из базисных переменных равна нулю. Для методов поиска оптимального плана перевозок существенно определить все базисные переменные. Не нарушая условия задачи, мы можем считать базисной переменной любую незаполненную клетку, поставив в нее нулевой объем перевозки. В табл. 5.3 мы приняли за базисную переменную клетку (2.3), записав в ней явно *0*. Это значение называют значащим нулем. В общем случае значащих нулей, характеризующих степень вырожденности плана, может быть несколько.

Рассмотренный метод «*северо-западного угла*» для поиска опорного плана достаточно прост, но соответствующая ему вершина симплекса может далеко отстоять (в смысле количества промежуточных вершин) от вершины, соответствующей оптимальному решению. Очевидно, что можно получить опорный план, более близкий к оптимальному, если при его составлении учитывать стоимости перевозок из пунктов отправления в пункты назначения. Такой подход к поиску опорного плана называют поиском опорного плана методом «*минимального элемента*», когда опорный план строится с учетом удельных стоимостей перевозок. Конкретных алгоритмов этого метода может быть предложено достаточно много. Один из них может состоять, например, в следующем.

Просматривается транспортная таблица и выбирается первая клетка с минимальной стоимостью (в задаче на *min*). Для этой клетки назначается максимальный объем перевозки с учетом заявки и запаса в соответствующих клетке пункта назначения и пункта отправления. Вновь просматривается транспортная таблица и выбирается следующая клетка с минимальной стоимостью перевозки. Для нее назначается максимальный объем перевоз-

ки с учетом заявки и запаса в соответствующих клетках пункта назначения и пункта отправления, но также с учетом уже назначенных ранее перевозок. Такие действия продолжаются до тех пор, пока не будут удовлетворены все заявки на перевозки.

Очевидно, что опорный план по методу «*минимального элемента*» будет лучше, чем опорный план по методу NWC. В зависимости от того, какое решение принято в качестве опорного плана, число итераций при улучшении плана обычно может колебаться от $0{,}5 \cdot n$ *до* $2 \cdot n$.

5.3. Нахождение оптимального плана перевозок

Улучшение плана перевозок, естественно, связано с перераспределением объемов перевозок между пунктами отправления и назначения. Вернемся к табл. 5.3 и приведем ее еще раз в табл. 5.4. Для представленного в ней опорного плана значение целевой функции:

$$z(x) = 5 \cdot 100 + 3 \cdot 100 + 3 \cdot 100 + 7 \cdot 150 + 8 \cdot 250 = 4150 .$$

Попробуем улучшить этот план, перераспределив объемы перевозок. Предположим, что мы решили отменить перевозку *100* единиц груза из A_1 в F_1. Но для выполнения заявки пункта F_1 нам придется увеличить на такое же количество объем перевозки из какого-то другого пункта отправления (например A_2), а *100* единиц из пункта A_1 переадресовать в другой пункт назначения (например F_2). Теперь, чтобы перевозка в пункт F_2 соответствовала заявке, придется на *100* единиц уменьшить перевозку из других пунктов (например из A_2). Таким образом, мы построили замкнутый цикл переноса, в котором в каждой клетке, входящей в цикл, происходит поворот движения по циклу на *90* градусов. Если в клетке увеличивается объем перевозки, то удобно отметить ее плюсом, если уменьшается, то — минусом. В табл. 5.4 клетки рассмотренного цикла отмечены другим цветом и знаками «+» и «–».

Таблица 5.4

Цикл переноса

$A_n \setminus F_m$	F_1	F_2	F_3	F_4	Supplies	U(i)
A_1	5	4	5	6		
	$^{-}100$	$^{+}$			100	
A_2	3	3	6	6		
	$^{+}100$	$^{-}100$	0		200	
A_3	2	5	7	8		
			150	250	400	
Demands	200	100	150	250	700	
V(j)						

Поскольку оптимальное решение может достигаться в вершине симплекса, то начинать цикл переноса имеет смысл в одной из свободных клеток. Тогда все остальные клетки цикла окажутся базисными. Выбрав базисную клетку, принадлежащую циклу, со знаком «–» и минимальным объемом перевозки, можно этот объем переместить по циклу переноса, причем выбранная базисная клетка после переноса груза станет свободной. Таким образом осуществляется переход к другой вершине симплекса.

Перенос единицы груза по циклу изменит значение целевой функции на величину $n_{12} = C_{12} - C_{22} + C_{21} - C_{11} = 4 - 3 + 3 - 5 = -1$, т.е. уменьшит на единицу. Так как переносится *100* единиц, то значение функции цели уменьшится на *100* единиц.

Распределительный метод

В распределительном методе обычно ограничиваются поиском цикла переноса (иногда его называют циклом пересчета) с минимальной отрицательной ценой в задачах на *min* или максимальной положительной ценой в задачах на *max*. Поэтому для табл. 5.4 нужно определить цену цикла переноса для всех свободных клеток (1.2), (1.3), (1.4), (2.4), (3.1), (3.2) и затем выбрать цикл переноса с минимальной отрицательной ценой.

Если после переноса по выбранному циклу для всех свободных клеток цена циклов переноса ≥0 в задачах на *min* или ≤0 в задачах на *max*, то найденный план перевозок является оптимальным.

Рассмотрим теперь некоторые особенности, с которыми приходится встречаться при использовании распределительного метода. Обратимся к приведенному выше примеру ТЗЛП. Предположим сейчас, что мы пока не заботимся о поиске цикла переноса наилучшего в смысле цены, а просто выбрали отмеченный в табл. 5.4 цикл и сделали по нему перенос 100 единиц груза. Новое состояние транспортной таблицы будет иметь вид, показанный в табл. 5.5. В этой таблице в клетке (2.2) мы проставили значащий ноль, чтобы число базисных переменных, включая значащие нули, оставалось равным $n+m-1=3+4-1=6$. Теперь в табл. 5.5 оказалось два значащих нуля. Можно было бы проставить значащий нуль не в клетке (2.3), а в клетке (1.1). Тогда клетка (1.1) осталась бы базисной, а клетка (2.3) была бы свободной.

Вновь обратимся к табл. 5.4. Построим циклы для всех свободных клеток этой таблицы и вычислим их цены:

$$
\begin{aligned}
n_{12} &= C_{12} - C_{22} + C_{21} - C_{11} = 4 - 3 + 3 - 5 = -1, \\
n_{13} &= C_{13} - C_{23} + C_{21} - C_{11} = 5 - 6 + 3 - 5 = -3, \\
n_{14} &= C_{14} - C_{34} + C_{33} - C_{23} + C_{21} - C_{11} = 6 - 8 + 7 - 6 + 3 - 5 = -3, \\
n_{24} &= C_{24} - C_{34} + C_{33} - C_{23} = 6 - 8 + 7 - 6 = -1, \\
n_{31} &= C_{31} - C_{21} + C_{23} - C_{33} = 2 - 3 + 6 - 7 = -2, \\
n_{32} &= C_{32} - C_{22} + C_{23} - C_{33} = 5 - 3 + 6 - 7 = 1.
\end{aligned}
\qquad (5.4)
$$

Получилось два цикла переноса с одинаковой минимальной ценой, равной минус три. Для улучшения плана можно взять любой из этих циклов. Возьмем, например, цикл для клетки (1.3). Но в этом цикле в клетке со значащим нулем (клетка (2.3)) мы должны проставить знак «–». Поэтому фактически перенести по циклу мы можем ноль единиц груза. Казалось бы, это бесполезная операция. Но в действительности это не так, посколь-

ку при этом клетка (2.3) стала свободной, а клетка (1.3) — базисной, как это показано в табл. 5.6. Измененный набор базисных и свободных переменных будет соответствовать уже новой вершине симплекса и, соответственно, новому плану перевозки в смысле состава свободных и базисных переменных.

Теперь нужно снова построить циклы для свободных клеток табл. 5.6 и выбрать цикл с минимальной отрицательной ценой, по которому сделать максимально возможный перенос груза по циклу. Такая процедура продолжается до тех пор, пока для текущего плана цены циклов для свободных клеток не станут ≥0 (в задачах на *max* все цены должны быть ≤0).

Для закрепления навыков в освоении распределительного метода предлагаем самостоятельно довести решение рассматриваемого примера до получения оптимального плана. Для контроля в табл. 5.7 приведено состояние транспортной таблицы, соответствующей оптимальному плану.

Таблица 5.5

Состояние после переноса (вариант 1)

$A_n \setminus F_m$	F_1	F_2	F_3	F_4	Supplies	U(i)
A_1	5	4	5	6		
		100			100	
A_2	3	3	6	6		
	200	0	0		200	
A_3	2	5	7	8		
			150	250	400	
Demands	200	100	150	250	700	
V(j)						

Таблица 5.6

Состояние после переноса (вариант2)

$A_n \setminus F_m$	F_1	F_2	F_3	F_4	Supplies	U(i)
A_1	5	4	5	6		
	100		0		100	
A_2	3	3	6	6		
	100	100			200	
A_3	2	5	7	8		
			150	250	400	
Demands	200	100	150	250	700	
V(j)						

Таблица 5.7

Оптимальный план перевозки

$A_n \setminus F_m$	F_1	F_2	F_3	F_4	Supplies	U(i)
A_1	5	4	5	6		
			100		100	
A_2	3	3	6	6		
		100		100	200	
A_3	2	5	7	8		
	200		50	150	400	
Demands	200	100	150	250	700	
V(j)						

Недостатком распределительного метода является необходимость построения всех циклов переноса для свободных клеток, чтобы определить затем цену каждого из них. Этого недостатка лишен метод потенциалов, в котором цена каждого возможного цикла переноса определяется без их фактического построения.

5.4. Сценарий решения транспортной задачи в пакете QSB

В пакете QSB решение транспортной задачи линейного программирования (ТЗЛП) выполняется методом потенциалов. Решаемая задача ТЗЛП может иметь до 50 пунктов отправления или источников (sources) и такое же количество пунктов назначения (destination). Объемы (capacities) запасов и заявок могут быть только целыми значениями, а величины стоимостей перевозок, т.е. расходы или доходы (cost/profit) за перевозку единицы груза, могут быть вещественными числами.

Сценарий решения ТЗЛП в пакете QSB во многом похож на рассмотренный ранее сценарий решения в том же пакете задачи линейного программирования. Поэтому остановимся только на особенностях при решении ТЗЛП.

Для простоты изложения рассмотрим решение с помощью пакета следующей задачи. Имеется три пункта S1, S2 и S3 отправления однородного груза с запасами:

S1:35 S2:23 S3:20.

Заявки в трех пунктах D1, D2 и D3 составляют:

D1:20 D2:20 D3:43

Удельные расходы на перевозки единицы груза составляют

Из	в		
S1	D1:0.500	D2:5.300	D3:4.500
S2	D1:6.400	D2:4.500	D3:5.200
S3	D1:7.000	D2:3.000	D3:6.000

Нужно найти такой план перевозок, при котором суммарные расходы на перевозки минимальны.

Легко заметить, что в поставленной задаче не соблюдаются балансовые условия. Однако нет необходимости делать такую проверку, т.к. пакет автоматически ее осуществит и добавит в нашем случае фиктивный пункт отправления с запасом 5 единиц груза и с нулевыми удельными перевозками во все пункты назначения.

После запуска пакета и выбора работы для решения ТЗЛП на экран выдается главное меню возможных функций при решении ТЗЛП.

При выборе функции:

2 — Ввод новой задачи

с пользователем проводится диалог:

Вас интересует задача на max (1) или min (2) критерия? (введите 1 или 2)

How many sources are there in your problem? (Enter number є 50) <3>

How many destination are there in your problem? (Enter number є 50) <3>

Do you want to use the default names (S1,…Sn; D1,…,Dn) (Y/N)? <y>

Ввод исходных данных задачи trp (Capacities and Demands)

Sources:

S1:40 S2:23 S3:20

Destinations:

D1:20 D2:20 D3:43

Ввод исходных данных задачи trp (Cost/Profit Coefficients) Page 1

From To

S1 D1:10.00 D2:5.000 D3:4.000

S2 D1:6.000 D2:4.000 D3:5.000

S3 D1:7.000 D2:3.000 D3:6.000

Как видно, диалог этот достаточно прост и непосредственно соответствует использованному виду постановки задачи. После ввода данных обычно целесообразно выполнить функцию записи на диск исходных данных задачи.

Решение задачи начинается с выбора функции:

5 — Решение задачи…

При этом на экран выводится: *Таблица 5.8*

Функция при решении задач trptes When solving a problem, you can display every iteration of the MODI method if your problem scale is M < 5 and N < 6, where M is the number of sources, N is the number of destinations. Also you can use the NorthWest Corner Method (NWC) or Vogel's Approximation Method (VAM) to find the initial solution. Default method is NWC.
1----Solve and display the initial tableau (Решать и выводить начальную таблицу) 2----Solve and display each iteration (Решать и выводить каждую итерацию) 3----Solve and display the final tableau (Решать и выводить финальную таблицу) 4----Solve without displaying any iteration (Решать без показа итерации) 5----Use VAM for the initial solution (Использовать VAM-метод для начального решения) 6----Возврат к меню функций

Клавишами стрелок вверх / вниз или номером выберите нужную информацию и затем ENTER.

Первая часть выведенной информации говорит о том, что при числе пунктов отправления M<5 и пунктов назначения N<6 можно при выборе режима 2 получать на экране транспортную таблицу на каждой итерации решения задачи. Вторая часть представляет собой меню возможных режимов.

Таблица 5.9

Initial solution by NWC

SN/DN	D1	D2	D3	Supplies	U(i)
S1	10.50	5.300	4.500		0
	20.00	15.00		35.00	
S2	6.400	4.500	5.200		0
		5.00	18.00	23.00	
S3	7.000	3.000	6.000	20.00	
			20.00		
Dummy	0.000	0.000	0.000		0
				5	
Demands	20.00	20.00	43.00	83.00	
V(j)	0	0	0		

Minimum Value of OBJ=525.6
Любая клавиша для продолжения. Or `G' — No stop.

По умолчанию в пакете используется метод «северо-западного угла» (North West Corner — NWC) для поиска опорного плана или начального решения. Но с помощью режима 5 можно выбрать метод поиска опорного плана с учетом удельных расходов на перевозки (Vogel's Approximation Method — VAM). В этом случае число итераций при улучшении плана будет меньше.

В любом случае далее следует выбрать один из 4 режимов дальнейшего решения задачи. Предположим, что мы приняли метод NWC. Тогда при выборе функции:

2 Solve and display each iteration

на экран будет выдана табл. 5.9.

Она соответствует исходной постановке задачи, но в ней автоматически добавился фиктивный пункт отправления Dummy с запасами, которые делают балансовые условия справедливыми. Нажатие любой клавиши выводит на экран табл. 5.10.

В этой таблице пакетом проставлены платежи и отмечена свободная клетка (4,1) (она отмечена в таблице двумя звездочками), с которой связан цикл переноса с максимальной отрицательной ценой. Величина цены указана в конце таблицы фразой with e (4,1) = –4.5.

Теперь нужно перенести по циклу переноса (4,1), (1,1), (1,2), (2,2), (2,3), (3,4) максимально возможное количество груза. Из табл. 5.10 видно, что оно равно 5. Этот перенос выполняется пакетом по нажатию любой клавиши. Результаты выводятся в следующей выдаваемой таблице — табл. 5.11.

Таблица 5.10

Iteration 1

SN/DN	D1	D2	D3	Supplies	U(i)
S1	10.50	5.300	4.500		0
	20.00	15.00		35.00	
S2	6.400	4.500	5.200		–0.80
		5.00	18.00	23.00	
S3	7.000	3.000	6.000		0
			20.00	20.00	
Dummy	0.000	0.000	0.000		–6.00
	**		5	5	
Demands	20.00	20.00	43.00	83.00	
V(j)	10.500	5.300	6.000		

Minimum Value of OBJ=525.6 with e (4,1)= — 4.5
Любая клавиша для продолжения. Or `G' — No stop.

Таблица 5.11

Iteration 2

SN/DN	D1	D2	D3	Supplies	U(i)
S1	10.50	5.300	4.500		0
	15.00	20.00	**	35.00	
S2	6.400	4.500	5.200		–5.30
			23.00	23.00	
S3	7.000	3.000	6.000		–4.50
			20.00	20.00	
Dummy	0.000	0.000	0.000		–10.5
	5.000		0	5	
Demands	20.00	20.00	43.00	83.00	
V(j)	10.500	5.300	6.000		

Current Minimum Value of OBJ=525.6 with e (1,3)= — 6
Любая клавиша для продолжения. Or `G' — No stop.

Таблица 5.12

Final tableau (Total iteration=6)

SN/DN	D1	D2	D3	Supplies	U(i)
S1	10.50	5.300	4.500		0
			35.00	35.00	
S2	6.400	4.500	5.200		0.700
	15.00	0	8.00	23.00	
S3	7.000	3.000	6.000		–0.800
		20.00		20.00	
Dummy	0.000	0.000	0.000		–5.70
	5.00			5	
Demands	20.00	20.00	43.00	83.00	
V(j)	5.700	3.800	4.500		

Minimum Value of OBJ=525.6 with multiple optimal.
Оптимальное решение найдено.
Любая клавиша для продолжения

В этой таблице показаны также новые значения потенциалов и новая свободная клетка (1,3), цена цикла переноса для которой равна — 6.

Обратите внимание, что пакет автоматически проставил значащий ноль в клетке (4,3).

При нажатии любой клавиши будет выдана таблица для следующей итерации. Однако если нажать клавишу G, то дальнейшее преобразование транспортных таблиц будет происходить без вывода их на экран, а по окончании решения на экран будет выдана финальная таблица. Она приведена в табл. 5.12.

Как видно из этой таблицы, оптимальное решение достигнуто за 6 итераций. Оно получилось вырожденным, т.к. в клетке (2,2) стоит значащий ноль.

Из табл. 5.12 можно, конечно, выписать все перевозки из каждого пункта отправления в каждый пункт назначения. Но при нажатии любой клавиши выдается меню:

Функции

1---Вывод на экран только решения

2---Вывод и печать конечного решения

Возврат к меню функций

Первая функция выдает результаты решения в виде табл. 5.13.

Таблица 5.13

Результаты решения

Summary of Results for aaa Page:1							
From	To	Shipment	Unit cost	From	To	Shipment	Unit cost
S1	D1	0.0	10.50	S3	D1	0.0	7.000
S1	D1	0.0	5.300	S3	D2	20.0	3.000
S1	D3	35.0	4.500	S3	D3	5.0	0
S2	D1	15.0	6.400	Dummy	D1	5	0
S2	D2	0.0	4.500	Dummy	D2	0.0	0
S2	D3	8.0	5.200	Dummy	D3	0.0	0

Minimum Value of OBJ=355(multiple sols). Iterations=6
Любая клавиша для продолжения.

В пакете имеется также в главном меню функция выдачи результатов решения в том же виде.

Сравнения табл. 5.12 и 5.13 позволяет понять назначение данных в табл. 5.13.

Вернемся теперь снова к началу решения задачи запуском функции

5---Решение задачи…

В выдаваемом при этом меню выберем функцию

5---Use VAM for the initial solution

После чего выдается:

The initial solution method has been shifted to VAM. Please use option 1-4.

Теперь мы изменили поиск опорного плана с метода NWC на метод VAM. Далее выбираем функцию

2--- Solve and display each iteration

Пакетом будет выдана на экран табл. 5.14

Таблица 5.14

Initial solution by VAM

SN/DN	D1	D2	D3	Supplies	U(i)
S1	10.50	5.300	4.500		35.00
			35.00	35.00	
S2	6.400	4.500	5.200		15.00
	15.00		8.00	23.00	
S3	7.000	3.000	6.000		0.000
	0	20.00		20.00	
Dummy	0.000	0.000	0.000		5.000
	5.00			5	
Demands	20.00	20.00	43.00	83.00	
V(j)	15.00	20.00	8.00		

Minimum Value of OBJ=355.1
Любая клавиша для продолжения. Or `G' — No stop.

Обратим внимание, что данные табл. 5.14, за исключением платежей, полностью совпадают с данными табл. 5.12. Это говорит о том, что при учете стоимостей перевозок на каждой линии, что и учитывает метод VAM, составленный опорный план оказался и оптимальным. В общем случае этого может и не быть, но число последующих операций всегда будет меньше.

При нажатии любой клавиши на экран будет выдана такая же таблица, как табл. 5.12, только в ней будет указано, что число итераций равно 0. Таблица результатов будет такой же, как табл. 5.13, только первая строка под таблицей будет иметь вид:

Minimum value of OBJ = 355,1 (multiple sols.) Iterations = 0

К недостаткам пакета QSB при решении ТЗЛП следует отнести сравнительно небольшую размерность допустимых задач (50x50), а также отсутствие анализа чувствительности. Однако при необходимости анализа чувствительности с помощью пакета QSB транспортную задачу линейного программирования можно решить как обычную ЗЛП. Но так как ЗЛП при этом не должна иметь больше 40 переменных и 40 ограничений, то должны выполняться условия:

$n \cdot m \le 40$ — число переменных,

$n + m \le 40$ — число ограничений,

где n — число пунктов отправления, m — число пунктов назначения.

Первое из этих условий является более сильным и оно определяет предельные соотношения значений n и m.

5.5. Определение оптимальных грузопотоков

Классическая транспортная задача (5.1), (5.2) с правильным балансом (5.3) является одним из вариантов постановки задачи определения оптимальных грузопотоков.

Уже было видно, как просто добавлением фиктивного пункта отправления или назначения выполнить балансовые условия. Однако реальные постановки задачи определения оптимальных грузопотоков могут и более существенно отличаться от классической транспортной задачи. Тем не менее во многих случаях их можно привести к виду классической ТЗЛП. Рассмотрим некоторые из таких случаев:

1. Предположим, что в исходной постановке задачи количество запасов меньше количества заявок. Требуется определить оптимальные грузопотоки, которые обеспечивали бы пропорциональное удовлетворение каждой из заявок. В этом случае в формулировке ТЗЛП не выводится фиктивный пункт отправления, а для каждого пункта назначения принимается новая заявка, равная:

$$\overline{D}_j = D_j - \frac{\sum_{j=1}^{m} D_j - \sum_{i=1}^{n} S_i}{\sum_{j=1}^{m} D_j} \cdot D_j, j = 1,2,...,m \quad (5.13)$$

Легко заметить, что при таком новом составе заявок их сумма уменьшается, по сравнению с суммой старых заявок, на величину нехватки запасов в первоначальной постановке задачи. Поэтому условие баланса:

$$\sum_{i=1}^{n} S_i = \sum_{j=1}^{m} \overline{D_j} = \sum_{j=1}^{m} D_j - (\sum_{j=1}^{m} D_j - \sum_{i=1}^{n} S_i) \equiv \sum_{i=1}^{n} S_i$$

при новых значениях заявок будет уже выполняться.

2. Предположим, что в исходной постановке задачи количество запасов опять меньше количества заявок. При этом с учетом особенностей задачи заявки каждого пункта назначения могут быть ранжированы по приоритетам p_j их удовлетворения. Если пункт назначения имеет наименьшее значение приоритета $p_{min} = min\ (p_j)$, то его заявка должна удовлетворяться полностью.

Такую постановку исходной задачи можно свести к ТЗЛП, если заявки пунктов назначения переопределить следующим образом:

$$\overline{D}_j = D_j - \frac{\sum_{j=1}^{m} D_j - \sum_{i=1}^{n} S_i}{\sum_{j=1}^{m} \theta_j} \cdot \theta_j,$$

где $\theta_j = p_j - \min(p_j), j = 1, 2, ..., m.$

Однако приоритеты заявок пунктов назначения следует задавать так, чтобы все $\overline{D_j} \geq 0$.

3. При формулировке реальных задач определения оптимальных грузопотоков могут действовать запрещения перевозок по определенным линиям из *i*–го пункта отправления в *j*–й пункт назначения. Эти особенности можно учесть, изменяя стоимость перевозок по линиям. В задачах на min значения C_{ij} для соответствующих линий следует принять равными некоторому достаточно большому числу (например, в 1000 раз больше, чем самое большое значение из всех C_{ij}), а в задачах на max — наоборот некоторому достаточно малому числу (обычно достаточно принять отрицательное значение, если все $C_{ij} \geq 0$).

Варьируя стоимостями перевозок при решении задачи, можно достаточно гибко перераспределять грузопотоки, отдавая предпочтение тем или другим из них по некоторым причинам, возможно, даже не экономического характера. Конечно, реальная стоимость перевозок будет при этом меняться в соответствии с исходными стоимостями C_{ij} и полученным оптимальным планом перевозок при измененных стоимостях.

5.6. Обобщенная транспортная задача

Простота решения классической транспортной задачи стимулирует попытки сведения к ней оптимизационных задач, которые в своей первоначальной постановке таковыми не являются. При организации перевозок важно в конечном счете распределить грузопотоки и распределить транспортные средства по линиям перевозок, но при этом желательно учитывать

производительность погрузоразгрузочных средств в пунктах отправления и назначения, производительность транспортных средств и другие подобные показатели. Задачу, учитывающую подобные показатели, называют обобщенной транспортной задачей.

Не стоит здесь детально останавливаться на возможных постановках обобщенной транспортной задачи и особенностях их решений, но в качестве примера вернемся снова к постановке транспортной задачи, математическое описание которой определено формулами (7.1) — (7.3). В ней переменные x_{ij} определяли объемы перевозок по линии (i, j), из i–го пункта отправления в j–й пункт назначения.

Предположим теперь, что эти объемы перевозок должны быть выполнены транспортными средствами, производительность которых на разных линиях различна и составляет p_{ij} на линии (i, j). Тогда $y_{ij} = x_{ij} / p_{ij}$ определяет количество транспортных средств, необходимых для перевозки по линии (i, j). Отсюда $x_{ij} = p_{ij}\, y_{ij}$. После подстановки таких значений x_{ij} в (7.1) и (7.2) получим:

$$z(x) = \sum_{i=1}^{n} \sum_{j=1}^{m} c_{ij} \cdot p_{ij} \cdot y_{ij} \to \min$$

при ограничениях:

$$\sum_{i=1}^{n} p_{ij} \cdot y_{ij} = D_j; j = 1,2,..., m;$$

$$\sum_{j=1}^{m} p_{ij} \cdot y_{ij} = S_i; i = 1,2,..., n. \qquad (5.14)$$

Для сформулированной задачи (7.14), очевидно, выполняются и балансовые условия (7.3). Значение $\bar{c}_{ij} = c_{ij} \cdot p_{ij}$ следует рассматривать как удельные расходы при перевозке единицы груза транспортным средством на линии (i, j). Решение такой обобщенной транспортной задачи относительно новых переменных y_{ij}, конечно, сложнее, чем классической транспортной задачи, в которой коэффициенты при переменных в ограничениях

равны единице. При поиске опорного плана и переносах по циклам пересчета значения y_{ij} придется назначать с учетом p_{ij}.

5.7. Решение транспортной задачи по критерию времени

В рассмотренных ранее постановках транспортной задачи в качестве критериям оптимальности использовался минимум расходов, и потому в качестве стоимости перевозок c_{ij} принимались удельные расходы на перевозку единицы груза. Но стоимость перевозки c_{ij} по линии (i, j) в общем случае может иметь самый разный физический смысл. Например, она может задавать удельные доходы от перевозки, и тогда должна решаться задача на max. В качестве c_{ij} может приниматься, например, расстояние L_{ij} перевозки по линии (i, j). Тогда $L_{ij} \cdot x_{ij}$ будет определять объем транспортной работы на линии (i, j), измеряемый, например, в тонно-километрах, а задача оптимизации должна решаться на min транспортной работы.

Одним из возможных значений c_{ij} может быть время t_{ij} на перевозку по линии (i, j). Если предположить, что имеется достаточное количество транспортных средств для перевозки любого количества груза x_{ij} по линии (i, j), то t_{ij} не будет зависеть от объема перевозок по этой линии. Если в качестве критерия выбрать минимальное время T окончания всех перевозок, то можно сформулировать следующую задачу оптимизации.

При заданных продолжительностях t_{ij} перевозок по линии (i, j) найти такой план перевозок x_{ij} грузов из i-го пункта отправления в j-й пункт назначения, при котором время T завершения всех перевозок минимально. При этом в каждом i-м пункте отправления запас груза равен S_i, $i = 1, 2, \ldots, n$, а в каждом j-м пункте назначения заявка равна D_j, $j = 1, 2, \ldots, m$ единицам груза.

Такая задача называется транспортной задачей по критерию времени. Так как все перевозки заканчиваются в момент окончания самой про-

должительной перевозки, то время T равно максимальному t_{ij} из всех тех линий, по которым перевозки x_{ij} отличны от нуля. Математически поставленная задача запишется в виде:

$$z(x) = T = \max_{x_{ij}>0} t_{ij} \to \min$$

при

$$\sum_{i=1}^{n} x_{ij} = D_j;\ j = 1,2,...,m;$$
$$\sum_{j=1}^{m} x_{ij} = S_i;\ \mathrm{i} = 1,2,...,n. \qquad (5.15)$$

В формулах (5.15) запись $x_{ij}>0$ означает, что max выбирается только для тех значений t_{ij}, которые соответствуют отличным от нуля перевозкам. Поставленная задача не является задачей линейного программирования, т.к. критерий T не является линейной функцией x_{ij}. Ее решение можно свести к решению нескольких задач линейного программирования, но можно использовать расчетный «метод запрещенных клеток» [2].

Сначала составляется начальная таблица согласно конкретной задаче, например, так, как это показано в табл. 5.15. Здесь $t_{11} = 10$; $t_{12} = 5$ и т.д.

Можно было бы составить опорный план, например, методом северо-западного угла. Но при этом получится, что продолжительность перевозок составит не менее $T = 10$ за счет клетки (1,1). Попытаемся этого избежать, «запретив» себе ставить отличные от нуля перевозки в клетки, в которых $t_{ij} \geq 10$. В нашем примере таких клеток 2: (1,1) и (4.1). В любом случае число оставшихся клеток не должно быть меньше числа базисных переменных $(n+m-1)=(4+4-1)=7$.

В оставшихся запрещенных клетках составляем допустимый план, используя технику того же метода северо-западного угла, — сначала пытаемся максимально удовлетворить заявки первого пункта назначения, затем второго и т.д.

Составленный план перевозок показан в табл. 5.16. Для него максимальное время перевозки соответствует $t_{33} = 8$ для клетки (3,3). Поэтому к ранее запрещенным добавим еще свободные клетки, в которых $t_{ij} \geq 8$. Таких клеток в примере две: (3,1) и (4,3). Теперь попытаемся улучшить план, переведя базисную переменную x_{33}, которой соответствует клетка (3,3), в свободную. Для этого достаточно построить цикл переноса для свободной клетки (3,2). В этот цикл войдут клетки $(3,2)^{+}$, $(1,2)^{-}$, $(1,3)^{+}$ и $(3,3)^{-}$. Символы вверху обозначают: «$^{+}$» — увеличение; «$^{-}$» — уменьшение перевозок в соответствующих клетках. По этому циклу можно перенести 5 единиц груза. Ставшую свободной клетку (3,3) также делаем запрещенной.

Таблица 5.15

Начальная таблица

ПО/ПН	D1	D2	D3	D4	Запасы
S1	10	5	5	4	
					35
S2	6	4	5	5	
					25
S3	7	3	8	6	
					15
S4	11	4	9	3	
					28
Заявки	20	20	20	43	103

Таблица 5.16

Первая итерация

ПО/ПН	D1	D2	D3	D4	Запасы
S1	10	5	5	4	
		20	10		30
S2	6	4	5	5	
	20	5			25
S3	7	3	8	6	
		*	5	15	20
S4	11	4	9	3	
				28	28
Заявки	20	25	15	43	103

Таблица 5.17

Вторая итерация

ПО/ПН	D1	D2	D3	D4	Запасы
S1	10	5	5	4	
		15	15		30
S2	6	4	5	5	
	20	5			25
S3	8	3	8	6	
		5		15	20
S4	11	4	9	3	
				28	28
Заявки	20	25	15	43	103

Таблица 5.18

Третья итерация

ПО/ПН	D1	D2	D3	D4	Запасы
S1	10	5	5	4	
		15	15		30
S2	6	4	5	5	
	20	5			25
S3	8	3	8	6	
		5			20
S4	11	4	9	3	
				28	28
Заявки	20	25	15	43	103

Новый план показан в табл. 5.17. Для него продолжительность перевозок составляет $T = t_{34} = 6$. В оставшихся свободных клетках нет $t_{ij} \geq 6$. Поэтому нет и дополнительных запрещенных клеток. Однако попытаемся улучшить план, сделав клетку (3,4) свободной. Из таблицы видно, что только цикл переноса для свободной клетки (1,4) позволяет переместить по циклу 15 единиц с освобождением клетки (3,4). Новый план перевозок представлен в табл. 5.18. В этом плане продолжительность перевозок составляет 5 единиц, и она определяется клетками (2,1), (1,3). Единственная свободная клетка, для которой время перевозки меньше 5, — это клетка (4,2). Но по циклу, связанному с ней, можно перенести только 0 объема перевозок, сделав свободной клетку (1,2), в которой время перевозки равно 5. Поэтому план, соответствующий табл. 5.18, является оптимальным по критерию минимума времени завершения всех перевозок.

Глава 6. Динамическое программирование

Термином динамическое программирование определяют пошаговый метод решения оптимизационных задач. При этом решение задачи представляют в виде последовательности шагов, на каждом из которых нужно выбрать такие значения управляемых параметров, чтобы по совокупности всех шагов получить оптимальное (максимальное или минимальное в зависимости от постановки задачи) значение целевой функции. Если решение задачи происходит во времени, то разделение на шаги обычно естественно и достаточно просто. Однако даже при отсутствии времени как параметра оптимизационной задачи ее всегда можно представить в виде последовательности шагов по одному из параметров системы, правда, сделать это не всегда просто.

6.1. Общая постановка задачи динамического программирования

Представим, что процесс функционирования оптимизируемой системы можно разбить на этапы (шаги). На каждом v-м этапе, $v = 1, 2, \ldots, k$, управляемые параметры системы могут принимать значения $x_{1v}, x_{2v}, \ldots, x_{nv}$. Совокупность этих управляемых параметров на каждом v-м этапе образует вектор управления $u_v = (x_{1v}, x_{2v}, \ldots, x_{nv})$, а вектор $U = (u_1, u_2, \ldots, u_k)$ — вектор управления системой.

Если обозначить через S_{v-1} состояние системы к началу v-го этапа, то под действием управления u_v система в конце v-го этапа перейдет в состояние:

$$S_v = \varphi_v(S_{v-1}, u_v). \qquad (6.1)$$

Функция φ_v, которая называется функцией перехода, на каждом этапе может иметь свой вид и отражает функционирование оптимизируемой системы под действием управления на каждом этапе.

При v =1 состояние S_0 определяет начальное состояние системы, а при $v = k$ состояние S_k — соответствует конечному состоянию системы. Любое из S_v определяется совокупностью значений неуправляемых параметров системы. В пространстве неуправляемых параметров S_v соответствует некоторой точке. Число точек множества, образующего пространство возможных состояний, может быть в общем случае конечным или бесконечным. Если оно бесконечно, то оно может быть дискретным или непрерывным.

На каждом v-м этапе в зависимости от того, в каком состоянии S_{v-1} находилась система и какое было выбрано управление u_v, функция цели z_v примет на v-м этапе некоторое значение z_v, которое является функцией от состояния в начале v-го шага и управления на этом шаге, т.е.:

$$z_v = w_v(S_{v-1}, u_v). \tag{6.2}$$

Требуется найти такое оптимальное управление $U=(u_1,u_2, \ldots,u_k)$, при котором функция цели системы:

$$W = \sum_{v=1}^{k} z_v \to \max. \tag{6.3}$$

В задачах на минимум $W \to \min$.

Формулы (6.1)–(6.3) полностью определяют поставленную оптимизационную задачу, и в принципе она может решаться любым из возможных методов. Метод динамического программирования предусматривает поиск оптимальных управлений по этапам или шагам, когда на каждом шаге выбирается оптимальное управление. Однако метод динамического программирования обеспечивает выбор оптимального управления на каждом шаге не независимо от управлений на других шагах. Напротив, этот выбор делается с учетом предыдущих и будущих шагов.

Учет управлений на предыдущих шагах определяется состоянием S_{v-1}, в котором система окажется в начале v-го шага. Учет последующих шагов может выразиться в том, что мы принимаем на v-м шаге такое управление

u_v, которое обеспечит оптимальное значение W_v функции цели на всех шагах, начиная с v-го. При этом значение W_v равно значению w_v (S_{v-1}, u_v) функции цели на v-м шаге в сумме с оптимальным значением W_{v+1} функции цели на всех последующих шагах. Разумеется, u_v должно принадлежать области допустимых управлений $U_{доп.v}$ на v-м шаге. Таким образом, эти очевидные рассуждения мы можем математически записать в виде:

$$W_v = \max_{u_v \in U_{доп.v}} \{w_v(S_{v-1}, u_v) + W_{v+1}(S_v)\};\ v = 1,2,...,\ k. \quad (6.4)$$

С учетом формулы (6.1) формула (6.4) примет вид:

$$W_v = \max_{u_v \in U_{доп.v}} \{w_v(S_{v-1}, u_v) + W_{v+1}(\varphi_v(S_{v-1}, u_v))\};\ v = 1,2,...,\ k. \quad (6.5)$$

Формула (6.5) определяет *основное функциональное уравнение динамического программирования*. Заметим, что в наших рассуждениях ничего не изменится, если функция цели системы будет не аддитивной функцией выигрышей на каждом из этапов, как это определено формулой (6.3), а мультипликативной функцией, т.е.:

$$W = \prod_{v=1}^{k} z_v \to \max. \quad (6.6)$$

В этом случае основное функциональное уравнение динамического программирования будет иметь вид:

$$W_v = \max_{u_v \in U_{доп.v}} \{w_v(S_{v-1}, u_v) \cdot W_{v+1}(\varphi_v(S_{v-1}, u_v))\};\ v = 1,2,...,\ k. \quad (6.7)$$

6.2. Геометрическая интерпретация и алгоритм решения

Для лучшего понимания сущности метода динамического программирования рассмотрим его геометрическую интерпретацию. Предположим, что оптимизируемая система имеет дискретное множество состояний (рис. 6.1) и вначале находится в одном из возможных состояний S_{0i}, i = 1, 2, …, n. Если система находится, например, в состоянии S_{01}, то под действием того или иного допустимого управления u_1 в конце первого шага она

окажется в одном из возможных состояний S_{1i} в соответствии с функцией перехода на этом шаге, т.е. $S_{1i}=\varphi_1(S_{01}, u_1)$. Если система находится в состоянии S_{02}, то под действием управления u_1 она окажется в конце первого шага в общем случае в другом состоянии из множества возможных состояний S_{1i}.

В зависимости от начального состояния S_{0i} и выбранного управления u_1 получится выигрыш на этом шаге $w_1(S_{0i}, u_1)$. Если система окажется в некотором состоянии S_{1i} к началу второго шага, то под действием некоторого допустимого управления u_2 в конце второго шага она окажется в одном из возможных состояний S_{2i} в соответствии с функцией перехода на этом шаге, т.е. $S_{2i}=\varphi_2(S_{1i}, u_2)$. Выигрыш на втором шаге составит $w_2(S_{1i}, u_2)$, а общий выигрыш составит $w_1(S_{0i}, u_1)+w_2(S_{1i}, u_2)$, если функция цели системы будет аддитивной функцией выигрышей на каждом из этапов.

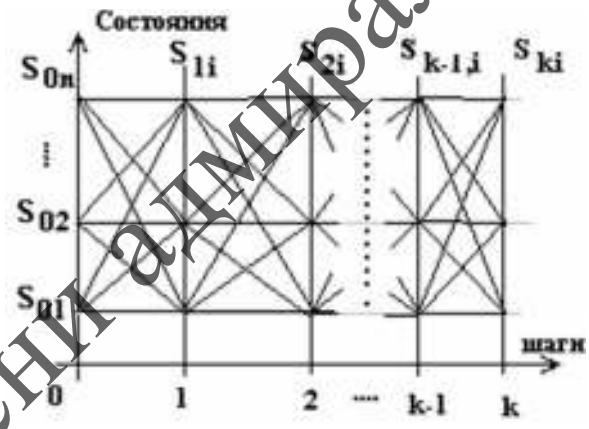

Рисунок 6.1. Схема переходов системы

На последнем k-м шаге система под действием управления $u_k \in U_{доп.k}$ из некоторого состояния $S_{k-1,i}$ перейдет в некоторое состояние $S_{k,i} = \varphi_k(S_{k-1,i}, u_k)$, а выигрыш на этом шаге составит $w_k(S_{k-1,i}, u_k)$. Собственно, нас не очень интересует конечное состояние $S_{k,i}$. Важно выбрать такие управления на каждом шаге, чтобы суммарный выигрыш на всех шагах был максимален.

В соответствии с принципом динамического программирования, управление на каждом шаге нужно планировать с оглядкой на то, к чему это может привести в будущем. Но из всех шагов только последний можно планировать без оглядки на будущее, т.к. больше шагов нет. Правда, мы пока не знаем, в каком состоянии окажется система в начале *k*-го шага. Но мы можем найти оптимальное управление $\bar{u}_k$ как функцию состояния $S_{k-1,i}$, т.е. определить такое

$$\overline{u_k} = \chi(S_{k-1,i}), \tag{6.8}$$

чтобы $\overline{W}_k(S_{k-1,i}) = \max\limits_{\bar{u}_k \in U_{доп.k}} \{w_k(S_{k-1,i}, \bar{u}_k)\}; i = 1,2,..., n.$ (6.9)

Управление $\bar{u}_k$ и $\bar{W}_k(S_{k-1,i})$ называются, соответственно, *условным оптимальным управлением* и *условным оптимальным выигрышем на k-м шаге*. Условность состоит в том, что мы пока не знаем, в каком конкретном состоянии окажется система к началу *k*-го шага. Но нам теперь известно, какое нужно применить управление и какой будет оптимальный выигрыш, если будет известно конкретное значение из множества возможных состояний $S_{k-1,i}$.

Зная условный оптимальный выигрыш $\bar{W}_k(S_{k-1,i})$ на *k*-м шаге, мы можем теперь определить условный оптимальный выигрыш $\bar{W}_{k-1}(S_{k-2,i})$ на *k*–1-м шаге, если выберем такое условное оптимальное управление

$$\bar{u}_{k-1} = \chi(S_{k-2,i}), \tag{6.10}$$

чтобы в соответствии с основным функциональным уравнением (6.5) динамического программирования выполнялось соотношение:

$$\overline{W}_{k-1}(S_{k-2,i}) = \max\limits_{\bar{u}_{k-1} \in U_{доп.k-1}} \{w_{k-1}(S_{k-2,i}, \bar{u}_{k-1}) + \overline{W}_k(S_{k-1})\}; i = 1,2,..., n. \tag{6.10а}$$

Но в соответствии с функцией перехода $S_{k-1,I} = \varphi_{k-1}(S_{k-2,i}, \bar{u}_{k-1})$ на *k*–1-м шаге мы можем записать:

$$\overline{W}_{k-1}(S_{k-2,i}) = \max\limits_{\bar{u}_{k-1} \in U_{доп.k-1}} \{w_{k-1}(S_{k-2,i}, \bar{u}_{k-1}) + \overline{W}_k(\varphi_{k-1}(S_{k-2}, \bar{u}_{k-1}))\}; i = 1,2,..., n. \tag{6.11}$$

Продолжая аналогичным образом, мы можем найти условное оптимальное управление на *k*–2, *k*–3, *k*–4 и т.д. и, наконец, на первом шаге:

$$\overline{u}_1 = \chi(S_{0,i}), \tag{6.12}$$

которое должно удовлетворять соотношению:

$$\overline{W_1}(S_{0,i}) = \max_{\overline{u}_1 \in U_{доп.1}} \{w_1(S_{0,i}, \overline{u}_1) + \overline{W}_2(\varphi_1(S_{0,1}, \overline{u}_1))\}; i = 1,2,..., n. \tag{6.13}$$

Если теперь с учетом конкретной постановки задачи задать конкретное значение состояния системы S_0^* в начальный момент, то по формуле (6.12) можно вычислить уже не условное, а действительно оптимальное управление на первом шаге:

$$u_1^* = \chi(S_0^*). \tag{6.14}$$

Зная это управление, можно теперь определить действительное состояние в конце 1-го шага:

$$S_1^* = \varphi_1(S_0^*, u_1^*). \tag{6.15}$$

Аналогично будем получать:

$$u_2^* = \chi(S_1^*); S_2^* = \varphi_2(S_1^*, u_2^*);$$

$$u_3^* = \chi(S_2^*); S_3^* = \varphi_3(S_2^*, u_3^*);$$

……

$$u_{k-1}^* = \chi(S_{k-2}^*); S_{k-1}^* = \varphi_{k-1}(S_{k-2}^*, u_{k-1}^*);$$

$$u_k^* = \chi(S_{k-1}^*); S_k^* = \varphi_k(S_{k-1}^*, u_k^*). \tag{6.16}$$

Таким образом, при решении задачи динамического программирования многошаговый процесс «проходится» дважды:

- первый раз (первый проход) — от конца к началу, в результате чего определяются условные оптимальные управления и условные оптимальные выигрыши на каждом шаге по формулам (6.8)–(6.13);

– второй раз (второй проход) — от начала к концу, в результате чего после выбора конкретного значения начального состояния по формулам (6.14) — (6.16) определяются оптимальные (не условные) управления на каждом шаге.

При втором проходе определяются по необходимости и действительные состояния системы в конце каждого шага, а по формуле:

$$W_j(S^*_{j-1}) = \overline{W}_j(S_{j-1}{}^*) \qquad (6.17)$$

можно получить значение целевой функции за все шаги, начиная с *j*-го, при оптимальном управлении $U_k^*=(u_j^*, u_{j+1}{}^*, \ldots, u_k^*)$. Понятно, что нас в первую очередь интересует $W_1(S_0{}^*)$, определяющее оптимальный выигрыш всей системы.

Как видно, логическая схема вычислений в методе динамического программирования достаточно проста. Метод не накладывает ограничений на вид функций цели и переходов на каждом шаге. Они могут быть линейными или нелинейными. Множество состояний может быть дискретным или непрерывным. Функция цели системы может быть аддитивной или мультипликативной относительно функций цели на отдельных шагах. Задача может решаться на max или на min.

Сложности в применении метода динамического программирования обусловлены тремя причинами.

Первая из них состоит в том, что не всегда просто представить функционирование оптимизируемой системы состоящим из отдельных шагов.

Вторая сложность связана с проблемой отыскания условных оптимальных управлений на каждом шаге по формулам типа (6.9), (6.11), (6.13). Фактически речь идет о решении на каждом шаге вариационной задачи, т.к. на каждом шаге требуется найти условное оптимальное управление как функцию состояний, оптимизирующую критерий функционирования си-

стемы за все оставшиеся шаги, начиная с текущего, который является функционалом относительно условного оптимального управления.

Третья сложность связана с проблемой размерности. Размерность задачи, с одной стороны, определяется количеством шагов, а с другой — количеством состояний, если состояния системы образуют конечное множество. В этом случае функции выигрыша и функции переходов на каждом шаге являются дискретными функциями, определение которых на множестве состояний системы достаточно громоздко.

6.3. Задача об оптимальной загрузке судна

Рассмотрим следующую постановку задачи. На судно с заданной грузовместимостью Q должны быть погружены комплекты груза трех типов (например контейнеры трех типов). Для комплекта груза i-го типа, i = 1, 2, 3, известны его объем v_i и стоимость перевозки c_i. Требуется подобрать такой набор $x = (x_1, x_2, x_3)$ комплектов груза каждого типа, чтобы общая стоимость от перевозки была максимальна. Математическую формулировку поставленной задачи можно записать в виде:

$$W(x) = \sum_{i=1}^{3} c_i \cdot x_i \to \max \tag{6.18}$$

при ограничениях:

$$\sum_{i=1}^{3} v_i \cdot x_i \le Q; \tag{6.19}$$

$x_i \ge 0$ и должны быть целыми числами.

Кроме ограничения (6.19), можно было бы включить ограничения на имеющееся количество комплектов груза каждого типа, но мы не будем этого делать с целью обеспечения более наглядного представления о применении метода динамического программирования.

Как видно, поставленная задача является задачей целочисленного линейного программирования, но попробуем решить ее методом динами-

ческого программирования. Для применения этого метода нужно прежде всего определить, что характеризует состояния рассматриваемой системы (или задачи), на какие шаги или этапы можно разделить функционирование системы (решение задачи). Управляемыми параметрами в поставленной задаче являются значения x_i, т.е. число погруженных контейнеров каждого типа. Те или иные значения управляемых параметров изменяют оставшийся свободный объем на судне для погрузки контейнеров. Потому естественно имеющимся свободным объемом на судне характеризовать состояние анализируемой системы. Обозначим его через S. В начальный момент $S = S_0 = Q$. В качестве этапов решения задачи можно принять:

- 1-й этап — погрузка контейнеров 1-го типа;
- 2-й этап — погрузка контейнеров 2-го типа;
- 3-й этап — погрузка контейнеров 3-го типа.

Это, конечно, не означает, что в действительности погрузка должна происходить именно в такой последовательности. Выделение этапов — необходимый момент в возможности применения метода динамического программирования.

Управлением на первом этапе, очевидно, будет $u_1 = x_1$, на втором — $u_2 = x_2$ и на третьем — $u_3 = x_3$.

Функции переходов и допустимые управления на каждом этапе будут иметь вид:

$$S_1 = \varphi_1(S_0, u_1) = S_0 - v_1 \cdot u_1, \quad U_{доп.1} = 0 \div \left[\frac{S_0}{v_1}\right]; \tag{6.20}$$

$$S_2 = \varphi_2(S_1, u_2) = S_1 - v_2 \cdot u_2, \quad U_{доп.2} = 0 \div \left[\frac{S_1}{v_2}\right]; \tag{6.21}$$

$$S_3 = \varphi_3(S_2, u_3) = S_2 - v_3 \cdot u_3, \quad U_{доп.3} = 0 \div \left[\frac{S_2}{v_3}\right], \tag{6.22}$$

где запись [ ] означает целую часть от значения в квадратных скобках. Области допустимых управлений фактически учитывают ограничения (6.19) в исходной постановке задачи.

Вид функций выигрыша, т.е. функций критерия на каждом v-ом этапе будет:

$$w_v(S_{v-1}, u_v) = c_v \cdot u_v = c_v \cdot x_v, \quad v = 1,2,3. \qquad (6.23)$$

В данном примере постановки задачи функции выигрыша не зависят от состояния в начале этапа. Если в постановке задачи предусмотреть, например, штрафы за недоиспользованную грузовместимость, то функция выигрыша могла бы иметь, например, вид $w_v(S_{v-1}, u_v) = c_v \cdot u_v - g_v \cdot S_{v-1}$, где g_v — заданный коэффициент. Очевидно, что выигрыш за все этапы составит:

$$W(x) = \sum_{v=1}^{3} w_v = \sum_{v=1}^{3} c_v \cdot u_v = \sum_{i=1}^{3} c_v \cdot x_v \,. \qquad (6.24)$$

Теперь все определено для решения задачи методом динамического программирования, и можно начать первый проход по этапам, начиная с конца, для определения условных оптимальных управлений.

Для третьего, последнего, этапа:

$$\overline{W}_3(S_2) = \max_{u_3 \in U_{доп.3}} \{w_3(S_2, \bar{u}_3)\} = \max_{u_3 \in U_{доп.3}} \{c_3 \cdot \bar{u}_3\} \,. \qquad (6.25)$$

Очевидно, что максимальное значение $\overline{W}_3(S_2)$ будет при максимальном значении u_3. Поэтому на основании (6.22) условное оптимальное управление на 3-м этапе будет равно:

$$\bar{u}_3 = [\frac{S_2}{v_3}]. \qquad (6.26)$$

Следовательно: $\overline{W}_3(S_2) = c_3 \cdot [\frac{S_2}{v_3}]. \qquad (6.27)$

С учетом (6.22): $S_3 = \varphi_3(S_2, \bar{u}_3) = S_2 - v_3 \cdot \bar{u}_3 = S_2 - v_3 \cdot [\frac{S_2}{v_3}]. \qquad (6.28)$

Определение S_3 носит, вообще говоря, информативный характер и позволяет просто иметь информацию о количестве неиспользованной грузовместимости судна.

Для второго этапа с учетом (6.10а), а также (6.23) и (6.27) будем иметь:

$$\overline{W}_2(S_1) = \max_{\bar{u}_2 \in U_{доп.2}} \{w_2(S_1, \bar{u}_2) + \overline{W}_3(S_2)\} = \max_{\bar{u}_3 \in U_{доп.2}} \{c_2 \cdot \bar{u}_2 + c_3 \cdot [\frac{S_2}{v_3}]\}. \quad (6.29)$$

С учетом (6.21) мы можем записать:

$$S_2 = \varphi_2(S_1, \bar{u}_2) = S_1 - v_2 \cdot \bar{u}_2. \quad (6.30)$$

Поэтому условное оптимальное управление на втором этапе должно быть определено из условия:

$$\overline{W}_2(S_1) = \max_{\bar{u}_2 \in U_{доп.2}} \{c_2 \cdot \bar{u}_2 + c_3 \cdot [\frac{S_1 - v_2 \cdot \bar{u}_2}{v_3}]\}. \quad (6.31)$$

Выражение для $\overline{W}_2(S_1)$ не является дифференцируемой функцией относительно $\bar{u}_2$. Поэтому построить на основе условия (6.31) зависимость $\bar{u}_2(S_1)$ можно только численным методом. При этом следует иметь в виду, что возможными значения S_1 на основании (6.20) могут быть:

$$\begin{aligned}
&S_{10} = \varphi_1(S_0, u_1) = S_0 - v_1 \cdot u_1 = S_0 \ при\ u_1 = u_{10} = 0,\\
&S_{11} = S_0 - v_1 \cdot 1 \quad при\ u_1 = u_{11} = 1,\\
&S_{12} = S_0 - v_1 \cdot 2 \quad при\ u_1 = u_{12} = 2,\\
&\qquad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\\
&S_{1n} = S_0 - v_1 \cdot n \quad при\ u_1 = u_{1n} = n = [\frac{S_0}{v_1}].
\end{aligned} \quad (6.32)$$

Для нахождения зависимости $\bar{u}_2(S_1)$ по условию (6.31) необходимо для каждого значения $S_{1j}, j = 0,1,2,...,n,$ последовательно задавать значения $\bar{u}_{2i} = i \quad при\ i = 0,1,2,...,m,\ где\ m = [\frac{S_{1j}}{v_2}],\ пока\ S_2 = \varphi_2(S_1, \bar{u}_2) = S_{1j} - v_2 \cdot \bar{u}_2 \geq 0,$ и выбирать такое значение $\bar{u}_{2i}^*$ для каждого S_{1j} при котором $\overline{W}_2(S_{1j})$ максимально. В результате мы получим $\bar{u}_2(S_1)$, удовлетворяющую условию

(6.31), как таблично заданную функцию, определенную на $n+1$-й точках парами значений $(S_{1j} \to \bar{u}_{2i}^{*})$.

Пары значений $(S_{1j} \to \overline{W}_2(S_{1j}))$будут определять таблично заданную функцию для условного оптимального выигрыша $\overline{W}_2(S_1)$ за второй и третий этап.

После определения $\bar{u}_2(S_1)$ мы можем определить состояние S_2, в которое на этапе 2 перейдет система при условии оптимального управления из состояния S_1:

$$S_2 = \varphi_2(S_1, \bar{u}_2) = S_1 - v_2 \cdot \bar{u}_2(S_1). \qquad (6.33)$$

Функция $\varphi_2(S_1, \bar{u}_2)$ будет также табличной функцией, определенной на множестве точек S_{1j}, j=0,1,2,…,n, согласно (6.32), т.е. будет определена парами значений $(S_{1j} \to S_{2j})$.

Теперь переходим к первому этапу. В соответствии с основным функциональным уравнением динамического программирования и (6.20) мы можем записать:

$$\overline{W}_1(S_0) = \max_{\bar{u}_1 \in U_{доп.1}} \{w_1(S_0, \bar{u}_1) + \overline{W}_2(S_1)\} = \max_{\bar{u}_1 \in U_{доп.1}} \{c_1 \cdot \bar{u}_1 + \overline{W}_2(S_0 - v_1 \cdot \bar{u}_1)\}. \qquad (6.34)$$

Так как функция $\overline{W}_2(S_1)$ выше была определена таблично парами значений $(S_{1j} \to \overline{W}_2(S_{1j}))$, то определение $\bar{u}_1$, удовлетворяющее условию (6.34), придется определять численно. Для этого при наборе значений (6.32) нужно выбрать такое условно оптимальное управление на первом этапе $\bar{u}_1(S_0) = \bar{u}_{1i}^{*}$, при котором выполняется (6.34). Значение $\overline{W}_1(S_0) = \overline{W}_1^{*}(S_0)$ будет определять при этом условный оптимальный выигрыш за все три этапа загрузки судна.

На этом первый проход в решении задачи методом динамического программирования заканчивается, и начинается второй проход.

Для первого этапа прямого хода по таблично определенной функции $u_1(S_0)$ определяем оптимальное управление $u_1 = u_{1i} = u_1(S_0^*)$. Выигрыш за все этапы погрузки, т.е. оптимальное значение целевой функции составит $W = \overline{W}_1^*(S_0^*)$. Зная u_1, можем определить по одной из формул (6.32) конкретное значение $S_{1i}^* = S_0^* - v_1 \cdot u_{1i}$ к концу первого этапа.

Для второго этапа для найденного S_{1i}^* по таблично определенной функции для $\bar{u}_2(S_1)$ определяем оптимальное управление $u_2 = u_{2i} = \bar{u}_2(S_{1i}^*)$, а затем состояние к концу второго этапа $S_{2i}^* = \varphi_2(S_{1i}^*, u_{2i}) = S_{1i}^* - v_2 \cdot u_{2i}$.

Для третьего этапа для найденного S_{1i}^* определяем оптимальное управление по (6.26), которое будет

$$u_3 = \left[\frac{S_{2i}^*}{v_3} \right].$$

Рассмотрим следующий числовой пример. Пусть для сформулированной вначале этого параграфа задачи дано:

$$Q = 135, v_1 = 20, v_2 = 30, v_3 = 40, c_1 = 100, c_2 = 140, c_3 = 170.$$

Решим теперь эту же задачу методом динамического программирования, используя полученные ранее соотношения в общем виде. По условию задачи

$$S_0 = S_0^* = Q = 135.$$

Начинаем обратный проход с 3-го этапа. На основании (6.26) условное оптимальное управление на 3-м шаге будет:

$$\bar{u}_3 = [\frac{S_2}{40}]. \qquad (6.35)$$

Для второго шага, чтобы определить условное оптимальное управление на основе (6.31), нам придется сначала определить возможные значения S_1 на основе соотношений (6.32). При этом $n = [135/20] = 6$. Сами значения S_1 в зависимости от u_1 представим в виде табл. 6.1.

Таблица 6.1

Зависимость S_1 b u_1

S_1		u_1	
Обозначение	Значение	Обозначение	Значение
S_{10}	135	u_{10}	0
S_{11}	115	u_{11}	1
S_{12}	95	u_{12}	2
S_{13}	75	u_{13}	3
S_{14}	55	u_{14}	4
S_{15}	35	u_{15}	5
S_{16}	15	u_{16}	6

Теперь можно определить условное оптимальное управление $\bar{u}_2(S_1)$ из условия (6.31). Для этого при каждом значении S_1 и табл. 6.1 нужно последовательно задавать:

$u_{2i} = i$ при $i = 0, 1, 2, \ldots, m$, где $m = [\frac{S_{1j}}{v_2}]$, пока $S_2 = \varphi_2(S_1, u_2) = S_{1j} - v_2 \cdot u_2 \geq 0$.

Все вычисления удобно представить в виде табл. 6.2. В этой таблице столбцы 6 и 7 определяют таблично заданные функции, соответственно, для условного оптимального выигрыша $\bar{W}_2(S_1)$ за два последних этапа и условное оптимальное управление $\bar{u}_2(S_1)$ на втором этапе в зависимости от свободной грузовместимости судна S_1 в конце первого этапа (столбец 2).

Далее переходим к первому этапу загрузки судна. Поскольку в постановке задачи конкретно задано начальное состояние $S_0 = Q = 135$, то нет необходимости находить условное оптимальное управление как функцию возможных значений S_0. Поэтому по (6.34) искомое $\bar{u}_1(S_0) = u_1(135)$ одновременно будет и оптимальным управлением u_1 на первом этапе. Формула (6.34) в нашем случае будет иметь вид:

$$W_1(135) = \max_{u_1 \in U_{доп.1}} \{c_1 \cdot u_1 + \bar{W}_2(S_1)\} = \max_{u_1 \in U_{доп.1}} \{c_1 \cdot u_1 + \bar{W}_2(135 - 20 \cdot u_1)\}.$$

Для определения по этой формуле оптимального управления u1 придется перебрать все возможные значения u1≤Uдоп.1=[135/20]=6 и для каждого из них, воспользовавшись значением $\overline{W}2(135-20\cdot u1)$, из табл. 6.2 выбрать такое u1*, при котором функция цели W1(135) будет максимальна. Все вычисления удобно представить в виде табл. 6.3.

Как видно из табл. 6.3, оптимальное управление на первом этапе $u_1^*=5$, т.е. должно быть погружено 5 комплектов груза первого типа, а значение целевой функции за все этапы равно $W^*=640$. При $u_1^*=5$ остаток свободной грузовместимости для последующих этапов $S_1^*=135–20\cdot5=35$.

Таблица 6.2

Расчет условного оптимального управления на этапе 2

S_1		u_2		$\overline{W}_2(S_1)=\{c_2\cdot u_2+c_3\cdot[\frac{S_1-v_2\cdot u_2}{v_3}]\}$	$\max \overline{W}_2(S_1)$	Условное оптимальное управление $\overline{u}_2$
1	2	3	4	5	6	7
S_{10}	135	$\overline{u}_{20}$	0	510	590	3
			1	480		
			2	450		
			3	590		
			4	560		
S_{11}	115	$\overline{u}_{21}$	0	340	480	1
			1	480		
			2	450		
			3	420		
S_{12}	95	$\overline{u}_{21}$	0	340	420	3
			1	310		
			2	280		
			3	420		
S_{13}	75	$\overline{u}_{23}$	0	170	310	1
			1	310		
			2	280		
S_{14}	55	$\overline{u}_{24}$	0	170	170	0
			1	140		
S_{15}	35	$\overline{u}_{25}$	0	0	140	1
			1	140		

Таблица 6.3

Расчет оптимального управления на первом этапе

S_0		u_1		$W_1(S_0)=\{c_1\cdot u_1+\overline{W}_2(S_0-v_1\cdot u_2)\}$	max $W_1(S_0)$	Оптимальное управление u_1^*
1	2	3	4	5	6	7
Q	135	u_{10}	0	100·0+590=590	640	5
		u_{11}	1	100·1+480=580		
		u_{12}	2	100·2+420=620		
		u_{13}	3	100·3+310=610		
		u_{14}	4	100·4+170=570		
		u_{15}	**5**	**100·5+140=640**		
		u_{16}	6	100·6+0=600		

Теперь можно перейти ко второму этапу прямого хода в решении задачи методом динамического программирования для определения оптимального управления u_2^* и остатка свободной грузовместимости S_2^* на конец этого этапа. Для этого по таблице 6.2 при $S_1 = S_1^* = 35$ определяем, что $u_2^* = 1$, т.е. должен быть погружен один комплект второго типа. Поэтому $S_2^* = S_1^* - v_2 \cdot u_2^* = 35 - 30 = 5$.

На третьем этапе прямого хода согласно (6.35) оптимальное управление:

$$u_3^* = \left[\frac{S_2^*}{40}\right] = 0.$$

Таким образом, методом динамического программирования мы нашли оптимальное управление $U^* = (u_1^*, u_2^*, u_3^*) = (5,1,0)$ и значение целевой функции оптимизируемой системы $W_1^* = 640$.

Решение задачи (6.18), (6.19) при заданных конкретных значениях можно выполнить как решение задачи целочисленного линейного программирования с применением, например, пакета QSB. Очевидно, что мак-

симальное число комплектов груза *i*-го типа не может быть больше Q / v_i. Поэтому данные о границах переменных можно задать в виде:

Integrality and Bounds for Variables

(Default values are continuous with lower bound 0 and upper bound 32000)

Var.no. Var. Integrality (I/C) Lower bound Upper bound

1 X1 <I> <0 > <7 >

2 X2 <I> <0 > <5 >

3 X3 <I> <0 > <4 >

Решение задачи дает следующие результаты:

Таблица 6.4

Результаты целочисленной задачи линейного программирования

Summary of Results for aa Page:1							
Variable		Obj.Fnctn		Variable		Obj.Fnctn	
No.	Names	Solution	Coeffi-cient	No.	Names	Solution	Coeffi-cient
1	X1	5.000	100.000	3	X3	0.000	170.000
2	X2	1.000	140.000				
Maximum value of the OBJ=640 Total iterations=43							

Как видно, $X1=u_1^*=5$, $X2=u_2^*=1$, $X3=u_3^*=0$, оптимальное значение целевой функции одно и то же и равно 640, т.е. результаты решения рассматриваемой оптимизационной задачи как задачи целочисленного линейного программирования совпадают с результатами решения методом динамического программирования, что, конечно, и должно быть.

Решение рассмотренной задачи оптимальной загрузки судна методом динамического программирования может показаться достаточно громоздким, но стоит заметить, что решение той же задачи как ЦЗЛП методом ветвей и границ потребовало 43 итерации, а значит, решения 43 задач линейного программирования типа (6.18), (6.19).

Фактически рассмотренная задача является частным случаем более общей задачи, известной в литературе как «задача о ранце» (problem knap-

sack). Для этой задачи требуется задание емкости ранца (Capacity of Knapsack), количества груза каждого типа (Number of Items), объема (Space Required) единицы груза каждого типа и веса или стоимость (Weight/Value) единицы груза каждого типа. С каждым типом груза связывается один шаг (Type of Item (Stage)) в поиске оптимальной (в смыслах max веса или стоимости) загрузки ранца. Для решения этого и еще двух типов задач в пакете QSB имеется специальная работа — *Динамическое программирование*.

6.4. Решение задач распределения ресурсов

При поиске оптимального решения экономических задач метод динамического программирования наиболее часто используется для задач распределения ресурсов. Такими ресурсами могут быть, например, денежные средства, сырье, трудовые ресурсы и т.д., которые необходимо распределить, например, между предприятиями, отраслями промышленности, видами транспорта или этапами при выполнении некоторых проектов. Распределение может производиться на определенный планируемый период времени или на определенные отрезки времени, составляющие планируемый период.

Рассмотренная выше задача оптимальной загрузки судна фактически также является задачей распределения ресурса, которым является свободная грузовместимость судна.

Рассмотрим несколько характерных постановок задач распределения ресурсов.

Распределение ресурсов на весь планируемый период

Постановка задач этого вида в достаточно общем случае формулируется следующим образом. Имеется объем средств в количестве S_0, который на планируемый период необходимо распределить между k предприятиями. Если в ν-е предприятие вкладывается x_ν, $\nu = 1, 2, \ldots, k$, средств, то до-

ход предприятия составляет $z_\nu(x_\nu)$. Требуется найти такие значения x_ν, при которых:

$$W = \sum_{\nu=1}^{k} z_\nu \to \max \quad \textit{и выполняется ограничение} \quad \sum_{\nu=1}^{k} x_\nu = S_0.$$

При решении такой оптимизационной задачи методом динамического программирования в качестве этапов можно принять, что 1-м этапом является выделение средств первому предприятию, 2-м — второму и т.д. Состояние системы в начале и конце каждого ν-го этапа логично характеризовать количеством средств на начало и конец этапа ($S_{\nu-1}$ и S_ν).

Для каждого ν-го этапа должны быть определены функции переходов $S_\nu = \varphi_\nu(S_{\nu-1}, u_\nu)$ и функции выигрыша $z_\nu = w_\nu(S_{\nu-1}, u_\nu)$, где $u_\nu = x_\nu$ — управление на ν-м этапе. Таким образом, функция z_ν определяет доход от вложения средств в ν-е предприятие. Очевидно, что $x_\nu = u_\nu \le U_{доп.\ \nu} = S_{\nu-1}$ и должно $S_k = 0$, т.к. средства должны быть распределены все.

Таким образом сформулированная задача распределения ресурсов ничем не отличается от общей постановки задач динамического программирования (6.1)–(6.3). Ее решение необходимо строить на основе уравнения (6.5), и оно было рассмотрено в общем виде и на примере загрузки судна.

В отличие от задачи загрузки судна комплектами груза разных типов, в общей постановке задачи распределения ресурсов нет ограничений на целочисленность управлений и состояний. Поэтому при дифференцируемости функций $z_\nu = w_\nu(S_{\nu-1}, u_\nu)$ и $S_\nu = \varphi_\nu(S_{\nu-1}, u_\nu)$ не придется прибегать к методам численного дифференцирования. Однако поиск условных оптимальных управлений на ν-м этапе на основе (6.11) из условия:

$$\frac{\partial}{\partial \bar{u}_\nu}(\overline{W}_\nu(S_{\nu-1})) = \frac{\partial}{\partial \bar{u}_\nu}\left\{w_\nu(S_{\nu-1}, \bar{u}_\nu) + \overline{W}_{\nu+1}(\varphi_\nu(S_\nu, \bar{u}_\nu))\right\} = 0$$

при ограничении $\bar{u}_v \le U_{доп.v} = S_v$ может оказаться достаточно сложной задачей нелинейной оптимизации.

В качестве примера рассмотрим следующую задачу распределения ресурсов. Инвестор располагает финансовыми средствами в объеме *S*. На планируемый период все или часть средств можно распределить между двумя предприятиями или положить в банк с известной процентной ставкой за планируемый период. При инвестировании *i*-го предприятия в объеме u_i оно увеличивает выпуск продукции, выручка от реализации которой составляет $\lambda_i(u_i)$, но при этом расходы на подготовку и выпуск продукции составляют $\mu_i(u_i)$. Очевидно, что в банк можно положить только $u_3 = S - u_1 - u_2$ средств. Выручка составит $\lambda_3(u_3) = \lambda_3(S - u_1 - u_2)$, *а* $\mu_3(u_3) \equiv 0$.

Требуется найти такое распределение средств, чтобы общая прибыль за планируемый период была максимальна, т.е. чтобы

$$W(S) = \max_{u_1, u_2} \left\{ \sum_{i=1}^{3} w_i(u_i) \right\} \qquad (6.36)$$

при ограничении

$$u_1 - u_2 \le S\,, \qquad (6.37)$$

где $w_1(u_1) = \lambda_1(u_1) - \mu_1(u_1), w_2(u_2) = \lambda_2(u_2) - \mu_2(u_2),$
$w_3(u_3) = w_3(S - u_1 - u_2) = \lambda_3(S - u_1 - u_2).$

Сформулированная задача (6.36), (6.37) может быть как линейной, так и нелинейной в зависимости от вида функций прибыли $w_i(u_i)$. Предположим для конкретности, что $\lambda_i(u_i) = a_i \cdot u_i$, а $\mu_i(u_i) = b_i \cdot u_i^2$, но $b_3 \equiv 0$, т.к. при размещении денег в банке расходов никаких нет. При таком предположении задача (6.36), (6.37) является нелинейной задачей оптимального управления.

Попытаемся решить поставленную задачу методом динамического программирования. Этапами в данном случае будут:

1–й этап — выделение средств первому предприятию,

2–й этап — выделение средств второму предприятию,

3–й этап — выделение средств для внесения в банк.

Состояние каждого этапа естественно характеризовать объемом нераспределенных средств. Понятно, что в начале первого этапа $S_0 = S$.

Характеристику каждого этапа можно представить в виде таблицы:

Этап	Управление	Функция перехода	Функция выигрыша	Область допустимых управлений
1	u_1	$S_1=S_0-u_1$	$w_1(u_1)=a_1\cdot u_1-b_1\cdot u_1^2$	$0\le u_1\le S_0$
2	u_2	$S_2=S_0-u_2$	$w_2(u_2)=a_2\cdot u_2-b_2\cdot u_2^2$	$0\le u_2\le S_1$
3	$u_3=S-u_1-u_2$	$S_3=S_2-u_3=0$	$w_3(u_3)=a_3\cdot u_3$	$u_3=S_2$

6.5. Сценарий решения задач динамического программирования в пакете QSB

В главном меню пакета имеется работа *Динамическое программирование*. После запуска этой работы и, значит, соответствующей программы на экран выдается главное меню выбранной задачи. Работой *1 — Общие сведения о задаче DP* на английском языке выдаются общие сведения о программе, суть которых состоит в следующем.

Эта программа содержит три общеизвестных алгоритма динамического программирования: задача **stagecoach** (почтовая карета, или задача о дилижансе), задача **knapsack** (задача о ранце), и задача управления производством и запасами продукции (**a production and inventory control problem**). Шаги или этапы в этой программе называются стадиями.

В общем случае задача может иметь до 20 шагов (стадий) и до 900 состояний на всех шагах. Процедура решения использует обратную рекурсию из последней стадии к первой.

Данные, требуемые для **задачи stagecoach** — число стадий, число состояний для каждой стадии и расстояния (цены) между состояниями на

последовательных стадиях. DP допускает 20 стадий (шагов) и до 50 состояний для каждой стадии. В частных случаях можно решать задачи stagecoach с 2800 ветвями (дугами).

Для **задачи knapsack** требуется задание емкости ранца (Capacity of Knapsack), количества груза каждого типа (Number of Items), которое должно быть <=50, объема (Space Required) единицы груза каждого типа и вес или стоимость (Weight/Value) единицы груза каждого типа. С каждым типом груза (Type of Item) связывается один шаг или стадия (Stage) в поиске оптимальной (в смысле max веса или стоимости) загрузки ранца. С учетом ограничений программного обеспечения емкость ранца, умноженная на количество типов груза, не может быть больше 900. Иначе программа зависает и требуется ее перезапуск.

Данные, требуемые для **задачи управления производством**, — заявка, объем хранения (объем склада) и стоимость единицы хранения, объем продукции и единица ее стоимости, а также стоимость начала производства продукции для каждой стадии. Задача может иметь до 20 шагов и до 900 состояний (единиц) запасов и объема продукции на всех шагах.

Функция перехода из состояния в состояние на каждом шаге и функция стоимости принимаются в этой программе линейными.

После ввода данных можно выбрать режим решения и отображения результатов.

Функциональной клавишей F8 можно отпечатать изображение экрана, но текущая установка пакета этого не допускает.

При вводе данных своей задачи с пользователем проводится диалог, в процессе которого вводятся требуемые для задачи данные.

Сначала выводится:

Задайте имя Вашей задачи набором не более 6 символов? ааа

Затем:

Your DP program can analyze problems with up to 20 stages and with up to900 states in total. Specifically, the program can solve problems with up to 2800 branches (arcs) for stagecoach problems.

(В общем случае задача может иметь до 20 шагов (стадий) и до 900 состояний на всех шагах. В частных случаях можно решать задачи stage-coach с 2800 ветвями (дугами)).

How many stages (є 20) are there in your problem?

(Сколько шагов (стадий <20) в вашей задаче?)

Далее предлагается выбрать тип задачи (алгоритма):

Select one of the following algorithms:

1 -- Stagecoach Problem

2 -- Knapsack Problem

3 -- Production and Inventory Control Problem

4 -- Возврат к меню функций

Enter the option number?

После выбора алгоритма начинается диалог по вводу данных, кото-рый зависит от типа решаемой задачи.

Исходные данные можно изменить работой в главном меню:

7 ---- Изменение постановки задачи…

Для начального ознакомления с работой пакета целесообразно работой:

3 ---- Чтение ранее сохраненной задачи с диска

загрузить данные одного из типов решаемых задач из файла:

DPTEST1 — задача о дилижансе,

DPTEST2 — задача о ранце,

DPTEST3 — задача управления производством.

При конкретной установке пакета имена этих файлов могут быть и изменены.

Диалог при загрузке и последующей выдаче введенных данных име-ет вид:

Чтение данных задачи с диска (текстовый пример: Prim<номер задачи>)

Введите <имя файла>для Вашего файла (например, Lab1).

Введите <имя файла> (ввод <имя диска>: — выводит текущий директорий)? DPTEST1

При успешной загрузке далее можно выполнить в главном меню работу:

4 ---- Выдача на экран и / или печать исходных задачи

Аналогично можно загрузить и выдать на экран тестовые примеры из файлов DPTEST2, DPTEST3.

Для лучшего понимания работы пакета рассмотрим по одному конкретному примеру для задачи каждого типа.

Задачи stagecoach (задачи о дилижансе)

Постановка этой задачи предусматривает поиск оптимального маршрута из исходного (входного) узла 1 (см. рис. 8.2) в конечный (выходной) узел 8 через промежуточные узлы в соответствии с заданными дугами между узлами. Для каждой дуги задана ее длина (или стоимость переезда из связанных дугой узлов). Критерием оптимизации является минимум общей длины маршрута из входного узла в выходной (или минимум стоимости переезда из 1 в 8).

Все промежуточные пункты, через которые может проходить маршрут, распределены по шагам (этапам). На первом шаге можно попасть в пункты 2, 3, 4, на втором -- в пункты 5, 6, 7, и т.д. Выбираемые дуги (Arc Decision) на каждом из этапов являются управлениями в данной задаче.

Данные, соответствующие рис. 6.2, сохранены в файле DPTEST1.

После загрузки примера задачи из файла DPTEST1 и выполнения работы 4 главного меню на экран будет выведено:

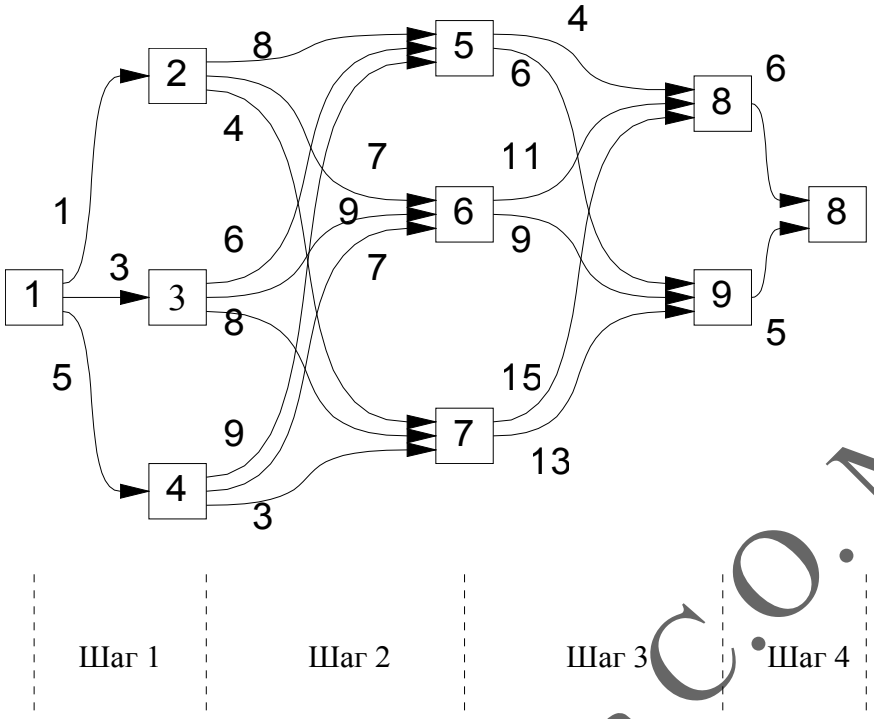

Рисунок 6.2. Шаги и состояние в задаче stagecoach

Ввод исходных данных задачи Stage — Stagecoach Problem

(input node– входной узел, output node– выходной узел)

Stage 1:

From input node 1 to output node 2: Distance/Cost 1

From input node 1 to output node 3: Distance/Cost 3

From input node 1 to output node 4: Distance/Cost 5

Stage 2:

From input node 2 to output node 5: Distance/Cost 8

From input node 2 to output node 6: Distance/Cost 7

From input node 2 to output node 7: Distance/Cost 4

From input node 3 to output node 5: Distance/Cost 6

From input node 3 to output node 6: Distance/Cost 9

From input node 3 to output node 7: Distance/Cost 8

From input node 4 to output node 5: Distance/Cost 9

From input node 4 to output node 6: Distance/Cost 7

From input node 4 to output node 7: Distance/Cost 3

Stage 3:

From input node 5 to output node 8: Distance/Cost 4

From input node 5 to output node 9: Distance/Cost 6

From input node 6 to output node 8: Distance/Cost 11

From input node 6 to output node 9: Distance/Cost 9

From input node 7 to output node 8: Distance/Cost 15

From input node 7 to output node 9: Distance/Cost 13

Stage 4:

From input node 8 to output node 10 : Distance/Cost 6

From input node 9 to output node 10 : Distance/Cost 5

Теперь можно приступать к решению задачи, выполнив работу 5 главного меню. Если выбрать режим выдачи каждого шага, то на экране появится:

Detailed Steps for the Stagecoach Problem:

Stage 4:

Input Node	*Arc Decision*	*Output Node*	*Distance to Destination*
8	*8 – 10*	*10*	*6*
9	*9 – 10*	*10*	*5*

Stage 3:

Input Node	*Arc Decision*	*Output Node*	*Distance to Destination*
5	*5 – 8*	*8*	*10*
6	*6 – 9*	*9*	*14*
7	*7 – 9*	*9*	*18*

Из рисунка 6.2 видно, что из пятого узла можно попасть в 10-й либо через 8-й, либо через 9-й узел. Программа для 5–го узла выбирает именно условное оптимальное управление -- дугу 5 – 8, т.к. при таком управлении из узла 5 в узел 10 путь будет кратчайшим. Аналогично можно проанализировать и другие выдаваемые данные.

Stage 2:

Input Node	*Arc Decision*	*Output Node*	*Distance to Destination*
2	*2 – 5*	*5*	*18*
3	*3 – 5*	*5*	*16*
4	*4 – 5*	*5*	*19*

Stage 1:

Input Node	*Arc Decision*	*Output Node*	*Distance to Destination*
1	*1 – 2*	*2*	*19*

Можно заметить, что эти данные соответствуют обратному ходу в решении задачи методом динамического программирования. Так как в начале первого шага исходное состояние всего одно, то условное оптимальное управление на первом шаге одновременно является и безусловным оптимальным управлением на первом шаге. Поэтому далее выдается:

Оптимальное решение найдено. Любая клавиша для продолжения.

The Final Shortest Routes for Stage

(Окончательные кратчайшие пути для шага)

Stage	*Arc Decision*	*Distance to Destination*
4	*1 – 2*	*19*
3	*2 – 5*	*18*
2	*5 – 8*	*10*
1	*8 – 10*	*6*

Таким образом, для рассматриваемого примера оптимальный маршрут будет проходить через узлы 1–2–5–8–10.

Задача knapsack (задача о ранце)

Пример постановки задачи состоит в следующем. Пусть имеется 4 типа груза (*Type of Item)*. Для груза каждого типа известно его количество (*Number of Items*). Известны также занимаемый объем (*Space Required*) и вес (или стоимость) (*Weight/Value*) единицы груза каждого типа. Известна емкость (объем) ранца (*Capacity of Knapsack*). Требуется подобрать такую

композицию грузов для ранца, чтобы общий вес (или стоимость) был максимален.

После загрузки примера задачи из файла DPTEST2 и выполнения работы 4 главного меню на экран будет выведено:

Ввод исходных данных задачи dp2 -- Knapsack Problem

Capacity of Knapsack = 10

Type of Item (Stage)	*Number of Items*	*Space Required*	*Weight/Value*
1	*3*	*1*	*10*
2	*6*	*2*	*15*
3	*7*	*3*	*20*
4	*2*	*1*	*20*

Шаг (*Stage*) в этой задаче связывается с типом груза. Поэтому управлением на первом шаге является количество груза 1–го типа, на втором — количество груза 2–го типа, и т.д.

Для каждого шага состояниями является емкость ранца в начале и конце шага. При обратном ходе решения задачи в начале каждого шага емкость ранца равна 10, т.к. на предыдущих шагах погрузка грузов соответствующих типов могла бы и отсутствовать. Поэтому в начале каждого шага рассматривается 11 возможных начальных (*Input State*) состояний: 0 — ранец заполнен, 1 — в ранце одна единица свободного объема, 2 — в ранце две единицы свободного объема, и т.д.

Условное оптимальное управление $u(i)$, которое в выводимых данных обозначается как *Decision*, определяет количество груза, которое можно загрузить на текущем шаге.

Остающаяся емкость ранца (*Output State*) получается уменьшением емкости в начале шага на объем погруженного груза на этом шаге.

Maximum Return Value — условный оптимальный выигрыш за все шаги, начиная с текущего. Для 4–го шага он равен условному оптимально-

му управлению, умноженному на стоимость единицы груза, которая соответствует этому шагу.

Для получения решения с высвечиванием каждого шага необходимо выполнить работу 5 главного меню, а затем выбрать режим:

1 ---- Solve and display each step

На экране будут появляться результаты решения в виде:

Stage (Type of Item) 4:

Input State	*Decision*	*Output State*	*Maximum Return Value*
0	*0*	*0*	*0*
1	*1*	*0*	*20*
2	*2*	*0*	*40*
3	*2*	*1*	*40*
4	*2*	*2*	*40*
5	*2*	*3*	*40*
6	*2*	*4*	*40*
7	*2*	*5*	*40*
8	*2*	*6*	*40*
9	*2*	*7*	*40*
10	*2*	*8*	*40*

Stage (Type of Item) 3:

Input State	*Decision*	*Output State*	*Maximum Return Value*
0	*0*	*0*	*0*
1	*0*	*1*	*20*
2	*0*	*2*	*40*
3	*0*	*3*	*40*
4	*0*	*4*	*40*
5	*1*	*2*	*60*
6	*1*	*3*	*60*
7	*1*	*4*	*60*

8	*2*	*2*	*80*
9	*2*	*3*	*80*
10	*2*	*4*	*80*

Stage (Type of Item) 2:

Input State	*Decision*	*Output State*	*Maximum Return Value*
0	*0*	*0*	*0*
1	*0*	*1*	*20*
2	*0*	*2*	*40*
3	*0*	*3*	*40*
4	*1*	*2*	*55*
5	*0*	*5*	*60*
6	*2*	*2*	*70*
7	*1*	*5*	*75*
8	*3*	*2*	*85*
9	*2*	*5*	*90*
10	*4*	*2*	*100*

Stage (Type of Item) 1:

Input State	*Decision*	*Output State*	*Maximum Return Value*
10	*2*	*8*	*105*

The Final Solution for dp2

Stage	*Decision*	*Return*	*Capacity Left*
1	*2*	*20*	*8*
2	*3*	*45*	*2*
3	*0*	*0*	*2*
4	*2*	*40*	*0*

Total Return = 105

В окончательном (финальном) решении выводимые данные означают:

Stage — номер шага, который соответствует принятому типу груза.

Decision — оптимальное управление на шаге, т.е. количество груза соответствующего типа, которое должно быть погружено.

Return — оптимальный выигрыш на текущем шаге.

Capacity Left — состояние в конце шага, т.е. оставшаяся емкость ранца в конце шага.

Total Return — оптимальный выигрыш за все шаги при оптимальном управлении.

Таким образом, для рассматриваемого примера при оптимальном решении следует погрузить 2 единицы груза 1-го типа, вес (стоимость) которых составляет 20 единиц, а остающаяся емкость ранца равна 8.

Груз 2-го типа должен быть погружен в количестве 3 единиц, вес (стоимость) которых составляет 45 единиц, а остающаяся емкость ранца станет равной 2.

Груз 3-го типа не грузится в ранец.

Груз 4-го типа должен быть погружен в количестве 2 единиц, вес (стоимость) которых составляет 40 единиц, а остающаяся емкость ранца станет равной 0.

6.6. Задача управления производством и запасами продукции

(A production and inventory control problem)

Рассмотрим в качестве примера следующую конкретную постановку этой задачи. Планируется выпуск продукции для четырех периодов, например, поквартально в течение года.

Известна величина заявки (demand ≤50), т.е. требуемое количество продукции в каждом периоде.

Известна производственная мощность выпуска продукции, т.е. максимальный объем производства (production capacity) продукции в каждом периоде. Если продукция выпускается в каком-то периоде, то известна стоимость начала производства продукции (setup cost), связанная с подго-

товкой к выпуску продукции, а также стоимость производства единицы продукции (unit production cost) в каждом периоде.

Излишки продукции в каждом периоде могут сохраняться на складе, причем максимальный объем запасов (storage capacity), т.е. емкость склада, для каждого периода известна. Известна также стоимость хранения единицы продукции (unit holding cost) в каждом периоде.

Требуется так спланировать выпуск продукции по периодам, чтобы при безусловном выполнении заявки в каждом периоде общие расходы на производство и хранение продукции были минимальны.

Каждый шаг в этой задаче соответствует одному периоду. При вводе данных сначала вводится начальное количество имеющейся продукции (units of initial inventory). Затем в диалоге вводятся указанные данные для каждого шага, начиная с первого.

После ввода данных их можно выдать на экран. Например, в тестовой задаче DPTEST3 они имеют вид:

Ввод исходных данных задачи dp3 -- Production/Inventory Problem

Initial Inventory = 3

Period (Stage)	*Demand Capacity*	*Production Capacity*	*Storage Cost*	*Setup Unit*	*Production Cost*	*Holding Unit Cost*
1	*5*	*20*	*15*	*10*	*5*	*1*
2	*5*	*20*	*15*	*10*	*5*	*1*
3	*5*	*20*	*15*	*10*	*5*	*1*
4	*5*	*20*	*15*	*10*	*5*	*1*

В данном примере все параметры задачи мы приняли одинаковыми на каждом шаге, но можно было бы задать их и разными.

Теперь работой 5 главного меню задачи можно выполнить решение, например, с высвечиванием каждого шага, если выбрать *1-- Solve and display each step*. Результаты решения будут выдаваться в виде:

Stage (Period) 4:

Input Inventory	*Production*	*Output Inventory*	*Minimum Total Cost*
0	*5*	*0*	*35*
1	*4*	*0*	*30*
2	*3*	*0*	*25*
3	*2*	*0*	*20*
4	*1*	*0*	*15*
5	*0*	*0*	*0*
6	*0*	*1*	*1*
7	*0*	*2*	*2*
8	*0*	*3*	*3*
9	*0*	*4*	*4*
10	*0*	*5*	*5*
11	*0*	*6*	*6*
12	*0*	*7*	*7*
13	*0*	*8*	*8*
14	*0*	*9*	*9*
15	*0*	*10*	*10*

Таким образом, решение начинается с последнего периода (шага). В выводимых данных значение *Input Inventory* определяет количество имеющейся продукции на начало периода, т.е. начальное состояние для шага. Значение *Production* определяет условное оптимальное управление на шаге — количество выпускаемой продукции на данном шаге. Величина *Output Inventory* определяет конечное состояние на шаге, т.е. количество продукции, оставшейся на шаге после удовлетворения заявки. Значение *Minimum Total Cost* определяет условные оптимальные затраты на производство и хранение продукции на шаге.

Так, первая строка значений говорит о следующем. Если к началу 4-го периода остаток продукции от 3-го периода равен нулю, то нужно для

удовлетворения заявки выпустить 5 единиц продукции. При этом затраты составят 35 единиц, т.к. 5*5=25 единиц будет затрачено на производство пяти единиц продукции и 10 на организацию выпуска продукции в данном периоде.

Общее количество состояний в начале шага равно 16, т.к. емкость склада равна 15 и потому с предыдущего периода не может остаться больше 15 единиц продукции.

Stage (Period) 3:

Input Inventory	*Production*	*Output Inventory*	*Minimum Total Cost*
0	*10*	*5*	*65*
1	*9*	*5*	*60*
2	*8*	*5*	*55*
3	*7*	*5*	*50*
4	*6*	*5*	*45*
5	*0*	*0*	*35*
6	*0*	*1*	*31*
7	*0*	*2*	*27*
8	*0*	*3*	*23*
9	*0*	*4*	*19*
10	*0*	*5*	*5*
11	*0*	*6*	*7*
12	*0*	*7*	*9*
13	*0*	*8*	*11*
14	*0*	*9*	*13*
15	*0*	*10*	*15*

Посмотрим теперь, как формируются условные оптимальные расходы (*Minimum Total Cost*) для 3-го периода. Возьмем для примера первую строку, в которой начальное состояние (*Input Inventory*) равно нулю. Понятно, что для удовлетворения спроса можно произвести только 5 единиц

продукции. В этом случае остаток на конец 3-го шага будет равен нулю. Затраты составят 10+5*5=35 единиц. Остаток на конец 3-го периода, а значит, и на начало 4-го периода будет 0. Но из данных на 4-м шаге мы видим, что тогда расходы на нем составят 35 единиц, а общие расходы за 3–й и 4-й периоды составят 35+35=70.

Предположим теперь, что мы решили выпустить на 3-м шаге 6 единиц продукции. Тогда к концу 3-го шага останется на хранение одна единица продукции (следовательно и на начало 4-го периода), расходы на 3-м шаге составят 10+6*5+1*1=41, а общие расходы на 3-м и 4-м шаге составят 41+30=71.

В действительности мы можем выпускать на 3-м шаге не только 5 или 6 единиц продукции, а любое количество в пределах мощности производства (от 5 до 20 единиц продукции). Если мы выберем такое количество производства продукции, которое с учетом последствий на 4-м шаге приведет к минимальным затратам, то эти затраты будут условным оптимальным значением затрат, а выбранный объем производства продукции — условным оптимальным управлением. Первая строка и показывает, что условным оптимальным управлением будет производство 10 единиц продукции, если к началу 3-го шага запас был равен нулю. При этом условное оптимальное значение целевой функции составит 65 единиц, а остаток (*Output Inventory*) на конец этого периода или начало следующего составит 5 единиц продукции.

Аналогично можно разобрать другие строки выводимых данных.

После расчетов для 3–го периода выдаются данные для следующих периодов в обратном ходе решения задачи.

Stage (Period) 2:

Input Inventory	*Production*	*Output Inventory*	*Minimum Total Cost*
0	*5*	*0*	*100*
1	*4*	*0*	*95*

2	*3*	*0*	*90*
3	*2*	*0*	*85*
4	*1*	*0*	*80*
5	*0*	*0*	*65*
6	*0*	*1*	*61*
7	*0*	*2*	*57*
8	*0*	*3*	*53*
9	*0*	*4*	*49*
10	*0*	*5*	*40*
11	*0*	*6*	*37*
12	*0*	*7*	*34*
13	*0*	*8*	*31*
14	*0*	*9*	*28*
15	*0*	*10*	*15*

Stage (Period) 1:

Input Inventory	*Production*	*Output Inventory*	*Minimum Total Cost*
3	*7*	*5*	*115*

Так как начальное состояние известно, то теперь могут быть найдены оптимальные управления на каждом периоде, сведения о которых выдаются в виде:

The Final Solution for dp3

Period (Stage)	*Beginning Inventory*	*Production*	*Setup Cost*	*Production Cost*	*Ending Inventory*	*Holding Cost*	*Total Cost*
1	*3*	*7*	*10*	*35*	*5*	*5*	*50*
2	*5*	*0*	*0*	*0*	*0*	*0*	*0*
3	*0*	*10*	*10*	*50*	*5*	*5*	*65*
4	*5*	*0*	*0*	*0*	*0*	*0*	*0*

Total Cost = 115

Значения в первой строке говорят о следующем. К началу 1–го периода запас продукции (*Beginning Inventory*) составлял 3 единицы. Оптимальное управление на первом шаге — выпуск 7 единиц продукции. При затратах на начало производства (*Setup Cost)* в 3 единицы затраты на само производство (*Production Cost*) составили 35 единиц, а затраты на хранение (*Holding Cost*) — 5 единиц. Поэтому общие оптимальные расходы (*Total Cost*) в первом периоде составили 50 единиц. При этом остаток продукции на конец периода (*Ending Inventory*) после удовлетворения заявки составил 5 единиц продукции.

Аналогично можно раскрыть значение других строк выведенных итоговых данных (*The Final Solution)*.

Общие оптимальные расходы за все периоды (*Total Cost*) составят 115 единиц.

Заключение

Во втором выпуске учебного пособия «Экономико-математические методы и модели в управлении водным транспортом. Линейное программирование» изложены известные методы линейного программирования, наиболее часто используемые для решения практических задач оптимального управления в экономических системах. Постановка и решение некоторых задач показаны на примерах, характерных для реальных ситуаций эксплуатационной деятельности водного транспорта. Показаны возможности использования ППП ЛП, в частности QSB. Из-за ограниченности объема учебного пособия в нем не рассмотрены другие пакеты и оболочки.

Линейным программированием не исчерпываются методы выбора оптимальных управленческих решений. Поэтому в планах авторского коллектива — подготовка следующего выпуска по тематике «Экономико-математические методы и модели в управлении водным транспортом»: нелинейное программирование и другие методы исследования операций как методы оптимизации управления в производственных системах.

Авторы с благодарностью примут дельные замечания и предложения по структуре и содержанию этого и предполагаемых изданий, стилю изложения.

Библиографический список

1. Балашевич В. А. Математические методы в управлении производством. — Минск, «Высшэйская школа», 1976.
2. Бережная Е. В., Бережной В. И. Математические методы моделирования экономических систем. — М.: Финансы и статистика, 2006.
3. Атлас Б. А., Бутов А. С., Волков Н. И., Голоскоков П. Г., Ступин О. К. Экономическая кибернетика на водном транспорте. Учебник. — М.: Транспорт. 1978.
4. Венцтель Е. С. Исследование операций. — «Советское радио», М., 1972.
5. Высшая математика для экономистов. Учебник для вузов/ Под редакцией Кремера Н. Ш. 2-е изд.перераб. и дополн. — М.: ЮНИТИ, 2004.
6. Интриллигатор М. Математические методы оптимизации и экономическая теория/Пер. с англ. Жукова Г.И., Кельман Ф.Я. — М.: «Айрис-Пресс», 2002.
7. Контарович Л. В., Булавский В. А., Звягина Р. А., Яковлева М. А. Численные методы линейного программирования. Специальные задачи. — М.: Наука, 1977.
8. Косоруков О. А., Мищенко А. В. Исследование операций: Учебник. — М.: Экзамен, 2003.
9. Куринский Б. Поиск оптимальных решений средствами Excel 7.0 в примерах. — СПб.: «BHV — Санкт-Петербург», 1997.
10. Математика для менеджеров и экономистов. Учебник для вузов. — СПб.: Издательство Михайлова В. А., 2002.
11. Саламанов О. Н. Математическая экономика с применением MathCAD и Excel. — СПб.: БХВ — Петербург, 2003.

12. Экономико-математические методы и прикладные модели: Учебное пособие для вузов/ Под редакцией Федосеева В. В. — М.: ЮНИТИ, 2006.

Учебное издание

Бабурин *Валерий Александрович*

Полянская *Татьяна Ивановна*

Полянский *Владимир Михайлович*

Шилкина *Ирина Дмитриевна*

ЭКОНОМИКО-МАТЕМАТИЧЕСКИЕ МЕТОДЫ И МОДЕЛИ В УПРАВЛЕНИИ ВОДНЫМ ТРАНСПОРТОМ

ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ

Учебное пособие

Компьютерный набор Е. В. Циценко

Художественное оформление С. В. Курбатов

Корректор *Сыворотка Н. И.*

Подписано в печать 07.09.12 Сдано в производство 07.09.12
Формат 60×84 1/16 Усл.-печ. л. 12,03. Уч.-изд. л. 10,35.
Тираж 173 экз. Заказ № 121

Санкт-Петербургский государственный университет водных коммуникаций
198035, Санкт-Петербург, ул. Двинская, 5/7

Отпечатано в типографии ФБОУ ВПО СПГУВК
198035, Санкт-Петербург, Межевой канал, 2