МИНИСТЕРСТВО ОБЩЕГО И ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ РФ

САМАРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра информатики и вычислительной математики

ЛАБОРАТОРНЫЙ ПРАКТИКУМ ПО МЕТОДАМ ИССЛЕДОВАНИЯ ОПЕРАЦИЙ

Издательство "Самарский университет"
1997

Настоящий лабораторный практикум является продолжением "Лабораторного практикума по методам оптимизации" В нем собраны математические методы, применяемые при решении задач по курсу "Исследования операций" В каждой лабораторной работе приведены постановки задач, кратко изложены элементы теории их решения, алгоритмы решения, варианты индивидуальных заданий, примеры решения задач.

Большинство методов оптимизации и исследования операций предполагает применение вычислительной техники. В лабораторных работах, которые в своих алгоритмах содержат решение вспомогательных задач линейного программирования, может быть использована обучающая программа GAUSS.EXE, которая работает на любых IBM совместимых персональных компьютерах.

Лабораторный практикум предназначен для студентов дневного и вечернего отделений ВУЗов, изучающих курс "Исследование операций"

Составители: кандидат физико-математических наук, доцент А.Г.Коваленко, кандидат физико-математических наук, доцент И.А.Власова, ассистент И.А.Шведова.

Рецензент кандидат техн. наук, доцент Самарского технического университета Л.Н.Смирнова

ЛАБОРАТОРНАЯ РАБОТА № 1

Многокритериальная задача выпуклого программирования. Поиск подходящих направлений

Рассматривается задача

$$\varphi_i(x) \to min, \qquad i \in M, \tag{1}$$
$$\varphi_i(x) \le 0, \qquad i \in I, \tag{2}$$

где $x=(x(1),x(2),...,x(n))^T$ - вектор n-мерного эвклидова пространства R^n, $\varphi_i(x)$ - выпуклые дифференцируемые функции, $i \in M \& I$.

Задача означает, что требуется:

- либо определить множество оптимальных точек при заданном предпочтении (в данной лабораторной работе по Слейтеру);
- либо определить,что множество оптимальных точек при заданном предпочтении пусто;
- либо убедиться, что множество допустимых решений, определяемых ограничениями (2), пусто;

Обозначим $X = \{ x \in R^n \ \varphi_i(x) \le 0, \ i \in I \}$. Пусть $x \in X$, через $I(x)$ обозначим множество $\{ i \in I \mid \varphi_i(x)=0 \}$. Будем говорить, что множество X удовлетворяет условиям регулярности $R2$ по Слейтеру, если существует точка $z \in X$ такая, что $\varphi_i(z)<0$ для всех $i \in$I.

Будем говорить, что направление $s \in R^n$ - подходящее по Слейтеру в точке $y \in X$, если для достаточно малого $\lambda>0$ справедливы неравенства

$$\varphi_i(y+\lambda \times s) \le 0, \qquad i \in I, \tag{3}$$
$$\varphi_i(y+\lambda \times s) < \varphi_i(y), \qquad i \in M. \tag{4}$$

Множество всех подходящих направлений в точке $y \in X$ образуют конус, который будем обозначать через $K(y)$.

Теорема 1

Пусть множество X удовлетворяет условию регулярности $R2$.Для того. чтобы точка $y \in X$ была точкой оптимума по Слейтеру, необходимо и достаточно, чтобы конус $K(y)=\varnothing$.

Для поиска подходящих направлений рассмотрим следующую задачу:

Задача ZS(y).

$$\max \sigma,$$
$$\langle \varphi'_i(y), s \rangle + \sigma \le 0, \quad i \in I(y) \cup M,$$
$$|s| \le 1, \qquad (5)$$
$$\sigma \ge 0.$$

Задача $ZS(y)$ не является задачей линейного программирования из-за нелинейности ограничения (5).Обычно это ограничение заменяют следующим

$$-1 \le s(j) \le 1, \quad j \in [1..n]. \qquad (6)$$

Теорема 2

Пусть множество X удовлетворяет условию регулярности R2. Для того, чтобы точка $y \in X$ была точкой оптимума по Слейтеру, необходимо и достаточно, чтобы максимальное значение σ в задаче $ZS(y)$ было равно нулю.

Заметим, что если в этой задаче $\sigma > 0$, то получившееся соответствующее значение s дает подходящее направление в точке y.

Задание.

Для задачи

$$\varphi_1(x) = (x_1 - a_1)^2 + (x_2 - b_1)^2 \to min,$$
$$\varphi_2(x) = (x_1 - a_2)^2 + (x_2 - b_2)^2 \to min,$$
$$\varphi_3(x) = (x_1 - a_3)^2 + (x_2 - b_3)^2 \to min,$$
$$\varphi_4(x) = (x_1 - a_4)^2 + (x_2 - b_4)^2 \to min.$$

проверить на оптимальность следующие точки:

$$y=(y_1,y_2)^T, \qquad z=(z_1,z_2)^T.$$

Если точка не является оптимальной, то для нее указать подходящее направление.

Варианты заданий взять в следующей таблице:

номер варианта	a_1	b_1	a_2	b_2	a_3	b_3	a_4	b_4	y_1	y_2	z_1	z_2
1	0	0	0	3	3	0	3	3	1	1	1	5
2	0	0	0	4	4	0	4	4	1	1	1	5
3	0	0	0	5	5	0	5	5	1	1	1	6
4	0	0	0	6	6	0	6	6	1	1	1	7
5	1	1	1	6	6	1	6	6	2	2	2	7
6	2	2	2	6	6	2	6	6	3	3	3	7
7	3	3	3	6	6	3	6	6	4	4	4	7
8	2	2	3	6	6	3	6	6	4	4	4	7
9	1	1	3	6	6	3	6	6	4	4	4	7
10	0	0	3	6	6	3	6	6	4	4	4	7
11	0	0	2	5	6	3	6	6	4	4	4	7
12	0	0	2	5	6	3	5	6	4	4	4	7
13	0	0	2	5	6	3	6	5	4	4	4	7
14	0	0	2	5	6	3	5	4	4	4	4	7
15	0	0	2	5	6	3	4	3	1	1	4	7
16	0	0	2	5	6	3	4	3	1	1	4	6
17	0	0	2	5	6	3	4	3	1	1	4	5
18	1	1	2	5	6	3	4	3	1	1	4	5
19	0	1	2	5	6	3	4	3	1	1	4	5
20	1	2	2	5	6	3	4	3	1	1	4	5
21	0	2	2	5	6	3	4	3	1	1	4	5
22	2	0	2	5	6	3	4	3	1	1	4	5
23	2	1	2	5	6	3	4	3	3	1	4	5
24	2	1	2	5	6	3	4	3	3	1	5	6
25	2	1	2	5	6	3	4	3	4	1	5	6

Пример

Для задачи

$$\varphi_1(x) = x_1^2 + x_2^2 \to min,$$
$$\varphi_2(x) = x_1^2 + (x_2-4)^2 \to min,$$
$$\varphi_3(x) = (x_1-4)^2 + x_2^2 \to min,$$
$$\varphi_4(x) = (x_1-4)^2 + (x_2-4)^2 \to min.$$

проверить на оптимальность точку $y=(2,2)^T$

Решение.

В этой задаче множество $I=\varnothing$, поэтому задача $ZS(y)$ примет вид:

$$
\begin{aligned}
&\textit{Задача } ZS(y).\\
&\max \sigma,\\
&\langle \varphi_i'(y), s\rangle + \sigma \le 0 \ , \ i \in [1..4], \qquad (7)\\
&-1 \le s(j) \le 1 \ , \ \ j \in [1..2],\\
&\sigma \ge 0.
\end{aligned}
$$

Подставляя в (7) значения градиентов $\varphi_i'(y)$,получим

$$
\begin{aligned}
&\max \sigma,\\
&4s(1)+4s(2)+\sigma \le 0,\\
&4s(1)-4s(2)+\sigma \le 0,\\
&-4s(1)+4s(2)+\sigma \le 0,\\
&-4s(1)-4s(2)+\sigma \le 0,\\
&-1 \le s(1) \le 1,\\
&-1 \le s(2) \le 1.
\end{aligned}
$$

Проведя замену переменных $s(j) = s'(j)-1, \ j \in [1..2]$, получим

$$
\begin{aligned}
&\max \sigma,\\
&4s'(1)+4s'(2)+\sigma \le 8,\\
&4s'(1)-4s'(2)+\sigma \le 0,\\
&-4s'(1)+4s'(2)+\sigma \le 0,\\
&-4s'(1)-4s'(2)+\sigma \le -8,\\
&s'(1) \le 2,\\
&s'(2) \le 2,\\
&s'(1) \ge 0,\\
&s'(2) \ge 0,\\
&\sigma \ge 0.
\end{aligned}
$$

Решая эту задачу симплекс-методом, получаем, что максимальное значение $\sigma=0$. Это означает, что точка $y=(2,2)$ оптимальна по Слейтеру.

ЛАБОРАТОРНАЯ РАБОТА № 2

Многокритериальная задача выпуклого программирования. Теорема Куна-Таккера

Рассматривается задача

$$\varphi_i(x) \to min, \qquad i \in M, \tag{1}$$
$$\varphi_i(x) \le 0, \qquad i \in I \tag{2}$$

где $x=(x(1),x(2),...,x(n))^T$ - вектор n-мерного эвклидова пространства R^n, $\varphi_i(x)$ - выпуклые дифференцируемые функции, $i \in M \& I$.

Задача означает , что требуется:

- либо определить множество оптимальных точек при заданном предпочтении (в данной лабораторной работе по Слейтеру).
- либо определить.что множество оптимальных точек при заданном предпочтении пусто,
- либо убедиться, что множество допустимых решений, определяемых ограничениями (2), пусто.

Обозначим $X = \{ x \in R^n , \varphi_i(x) \le 0, \ i \in I \}$. Пусть $x \in X$, через $I(x)$ обозначим множество $\{ i \in I \mid \varphi_i(x)=0 \}$. Будем говорить, что множество X удовлетворяет условиям регулярности $R2$ по Слейтеру, если существует точка $z \in X$ такая, что $\varphi_i(z)<0$ для всех $i \in$ I.

Теорема.

Пусть множество X удовлетворяет условию регулярности $R2$. Для того,чтобы точка $y \in X$ была точкой оптимума по Слейтеру, необходимо и достаточно существование таких $u(i)$, $i \in (M \ \& \ I(y))$, для которых справедлива система

$$0_n = \Sigma \{ u(i) \times \varphi'_I(y) \mid i \in (M \ \& I(y)) \}, \tag{3}$$
$$1 = \Sigma \{ u(i) \mid i \in (M \ \& I(y)) \}, \tag{4}$$
$$u(i) \ge 0 , \ i \in (M \ \& I(y)), \tag{5}$$

где 0_n - n-мерный нуль-вектор в R^n.

Задание.

Для задачи

$$\varphi_1(x) = (x_1 - a_1)^2 + (x_2 - b_1)^2 \to min,$$
$$\varphi_2(x) = (x_1 - a_2)^2 + (x - b_2)^2 \to min,$$
$$\varphi_3(x) = (x_1 - a_3)^2 + (x_2\ b_3)^2 \to min,$$
$$\varphi_4(x) = (x_1 - a_4)^2 + (x_2 - b_4)^2 \to min.$$

проверить на оптимальность следующие точки

$$y=(y_1,y_2)^T, \qquad z=(z_1,z_2)^T.$$

Варианты заданий взять в лабораторной работе "Многокритериальная задача выпуклого программирования. Поиск подходящих направлении".

Пример.

Для задачи

$$\varphi_1(x) = x_1^2 + x_2^2 \to min,$$
$$\varphi_2(x) = x_1^2 + (x_2-4)^2 \to min,$$
$$\varphi_3(x) = (x_1-4)^2 + x_2^2 \to min,$$
$$\varphi_4(x) = (x_1-4)^2 + (x_2-4)^2 \to min.$$

проверить на оптимальность точку $y=(2,2)^T$.

Решение.

Составим для этой задачи систему вида (3) - (5) . Для этого, подставив в нее значения градиентов $\varphi'_i(y)$, получим

$$\begin{aligned} 4u(1) + 4u(2) - 4u(3) - 4u(4) &= 0,\\ 4u(1) - 4u(2) + 4u(3) - 4u(4) &= 0,\\ u(1) + u(2) + u(3) + u(4) &= 1,\\ u(1) &\geq 0,\\ u(2) &\geq 0,\\ u(3) &\geq 0,\\ u(4) &\geq 0. \end{aligned} \qquad (6)$$

Нетрудно видеть, что эта система разрешима и имеет следующие решения:

$(0.5\ ,\ 0.0\ ,\ 0.0\ ,\ 0.5\ ),$
$(0.0\ ,\ 0.5\ ,\ 0.5\ ,\ 0.0\ ),$
$(0.25,\ 0.25,\ 0.25,\ 0.25).$

В общем случае проверить ее разрешимость можно симплекс-методом. Воспользуемся методом исскуственного базиса. Введем исскуственные переменные $z(j),\ j=1,2,3$ и построим задачу:

$$
\begin{array}{lllll}
F= & & z(1)+z(2)+z(3) \to min, \\
4u(1)+4u(2) & 4u(3)-4u(4)+z(1) & & & =0, \\
4u(1)-4u(2)+4u(3)-4u(4) & & +z(2) & & =0, \\
u(1)+u(2)+u(3)+u(4) & & & +z(3) & =1, \\
u(1) & & & & \geq 0, \\
& u(2) & & & \geq 0, \\
& & u(3) & & \geq 0, \\
& & & u(4) & \geq 0, \\
& & & z(1) & \geq 0, \\
& & & z(2) & \geq 0, \\
& & & z(3) & \geq 0.
\end{array}
$$

Если в этой задаче $F=0$, т.е. $z(1)=0,\ z(2)=0,\ z(3)=0,$ то система (6) разрешима. если $F>0,$ то система (6) не имеет решения.

ЛАБОРАТОРНАЯ РАБОТА № 3

Метод Гомори для решения задач целочисленного линейного программирования

Рассмотрим задачу целочисленного линейного программирования в виде

$$
\begin{array}{ll}
\langle c,x\rangle \to max & \\
Ax = b, & (1) \\
x \geq 0 & \\
x_i \text{ - целые}, \quad i\in\{1,2,3,\ldots,n\}=[1..n]. &
\end{array}
$$

Метод Гомори предполагает следующую последовательность действий для решения задачи (1):

1. Решаем задачу (1) без условий целочисленности решения (например,одним из алгоритмов симплекс-метода).
2. Если оптимальное решение X целочисленное (т.е. x_i целые для всех $i \in [1..n]$), то алгоритм прекращает работу, иначе переходим к пункту 3.
3. Если задача решалась симплекс-методом, значение базисных переменных x_i записано в столбце b симплекс-таблицы, внебазисные переменные равны нулю. Среди элементов столбца b ищем нецелочисленные. Если таких несколько,то обычно берется такое i_0, что $i_0 = argmax\{b(i) | \ i \in [1..m]\}$, составляется дополнительное ограничение (отсечение Гомори):

$$\sum \{-frac\,(a(i_0, j)) \times x(i) | j \in [,..n]\} \le -frac\,(b(i_0)),$$

где $frac(a)$ - дробная часть числа a. Приводим это ограничение к каноническому виду, приписываем к последней строке симплекс - таблицы и решаем двойственным симплекс-методом. Получив оптимальное решение,переходим к выполнению пунта 2.

Пример.

$$\begin{aligned} & x_1 + x_2 \to max, \\ & 2x_1 + x_2 \le 2, \\ & x_1 + 2x_2 \le 2, \\ & x_i \ge 0, x_i - \text{целые}, \ i=1,2. \end{aligned}$$

Занесем данные в симплекс-таблицу:

X_b	b	X_1	X_2	Y_1	Y_2
Y_1	2	2	1	1	0
Y_2	2	1	2	0	1
d	0	-1	-1	0	0

Решив эту задачу без условия целочисленности, получим симплекс-таблицу.

X_b	b	X_1	X_2	Y_1	Y_2
X_1	0.67	1	0	0.67	-0.33
X_2	0.67	0	1	- 0.33	0.67
d	1.33	0	0	0.33	0 33

Из этой таблицы получаем $X_1=0.67$, $X_2=0.67$, т.е. решение не целочисленное. строим дополнительное ограничение по первой строке (можно и по второй) : $-0.67Y_1 - 0.67Y_2 \leq -0.67$.
Приводим его к каноническому виду $-0.67Y_1 - 0.67Y_2 + Y_3 = -0.67$
и приписываем к симплекс-таблице :

X_b	b	X_1	X_2	Y_1	Y_2	Y_3
X_1	0.67	1.0	0	0.67	- 0.33	0
X_2	0.67	0	1	-0.33	+0.67	0
Y_3	-0.67	0	0	-0.67	-0.67	1
d	1.33	0	0	0.33	+0.33	0

Решаем двойственным симплекс-методом :

X_b	b	X_1	X_2	Y_1	Y_2	Y_3
X_1	0	1.0	0	0	-1	1
X_2	1	0	1	0	1	0
Y_1	1	0	0	1	1	1
d	1.33	0	0	0	0	0.5

Получим целочисленное решение $X_1= 0$, $X_2=1$, $Y_1=1$, $Y_2=0$, $Y_3=0$, Y_1, Y_2, Y_3 - дополнительные переменные. Ответ: $X_1= 0$, $X_2=1$.

Задание

Решить задачу из лабораторной работы N13 "Лабораторного практикума по методам оптимизации", добавить к условиям задачи условия целочисленности. Исходной таблицей для метода Гомори взять последнюю симплекс-таблицу лабораторных работ 14-15. Если в ней получено целочисленное решение, то задание скорректировать у преподавателя.

ЛАБОРАТОРНАЯ РАБОТА № 4

Метод ветвей и границ (алгоритм Ленд и Дойга) для решения задач целочисленного линейного программирования.

Рассмотрим задачу целочисленного линейного программирования в виде

$$\langle c,x\rangle \rightarrow max, \quad (1)$$
$$A\,x = b, \quad (2)$$
$$x \geq 0, \quad (3)$$
$$x_i \text{ - целые}, \quad i \in [1..n] \quad (4)$$

Для ее решения применим метод ветвей и границ.

Общая схема метода ветвей и границ

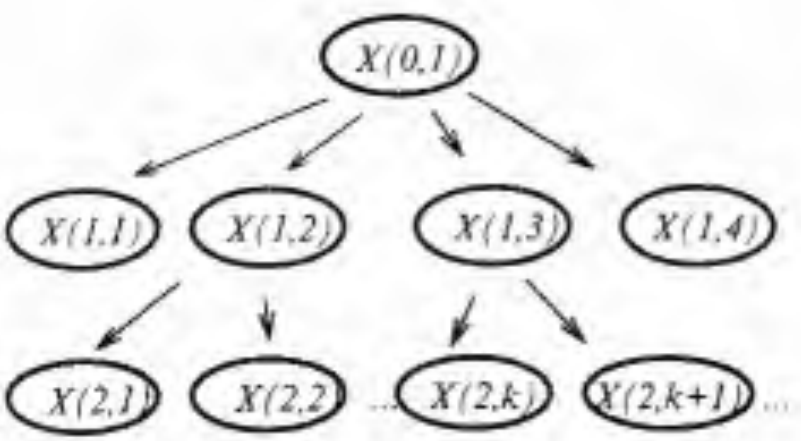
Рассматривается задача дискретного программирования

$\varphi(x) \rightarrow max,$

$x \in X$, X - *конечное множество*

Для ее решения методом ветвей и границ достаточно:

1. Уметь множество X (переобозначим его через $X(0,1)$) разбивать на некоторые подмножества $X(1,1), X(1,2), X(1,3), \ldots$. Каждое из этих подмножеств, в свою очередь, может быть разбито на подмножества $X(2,1), X(2,2), X(2,3), \ldots$ и так далее , вплоть до получения одноэлементных подмножеств. Индексы (i,j) у подмножеств $X(i,j)$ означают следующее: i - номер уровня разбиения, j - порядковый номер в уровне. В результате такого разбиения получается дерево подмножеств:



2. На каждом из таких подмножеств $X(i,j)$ уметь строить верхние оценки максимального значения функционала, т.е. определять значение $\xi(i,j)$ такое, что

$$\xi(i,j) \geq \max\{\varphi(x) \mid x \in X(i,j)\}.$$

Алгоритм метода ветвей и границ

0 - шаг. Полагаем $x'=\varnothing$, $Fx'=M$, где M достаточно малое число. В дальнейшем x' будем называть рекордом, Fx' - рекордным значением функционала. Строим верхнюю оценку $\xi(0,1)$ функционала $\varphi(x)$ на подмножестве $X(0,1)$, полагаем $V=\xi(0,1)$ (V - наименьшая из всех нижних оценок на всех подмножествах). Если в результате этого построения получаем некоторое $x \in X(0,1)$, то $x':=x, Fx':=\varphi(x)$, в противном случае x' и Fx' оставляем без изменения. Если $Fx'=V$,то $x^*=x'$ искомое решение и Fx' оптимальное значение функционала, в противном случае переходим к следующему шагу.

k-ый шаг. Среди всех подмножеств, не подвергнутых разбиению (на дереве разбиения они концевые), ищем подмножество $X(r,t)$, у которого $\xi(r,t)=V$. Разбиваем его на подмножества $X(r+1,m+1)$, $X(r+1,m+2)$, $X(r+1,m+3)$,..., где m номер последнего подмножества на $(r+1)$-ом уровне. Для каждого из этих подмножеств строим оценки $\xi(r+1,m+1)$, $\xi(r+1,m+2)$, $\xi(r+1,m+3)$,.... Обычно эти оценки находятся пересчетом (уточнением) из $\xi(r,t)$,то есть имеем :

$$\xi(r,t) \geq \xi(r+1,t'),\ t'=m+1,m+2,m+3,...$$

Если в результате этих построений получаем некоторое $x \in X(r,t)$ и $\varphi(x)>Fx'$,то $x':=x, Fx':=\varphi(x)$. Отбраковываем все подмножества $X(r,t)$, для которых $\xi(r,t) \leq Fx'$. Если в результате оказались отбракованными все подмножества, то x' и Fx' дают искомое решение и алгоритм заканчивает работу. В противном случае полагаем $V:=\max \xi(r,t)$, где максимум берется по всем неотбракованным и не подвергнутым разбиению подмножествам, и переходим к следующему шагу.

Замечание 1. Нетрудно видеть, что в работе алгоритма участвуют только подмножества, не подвергнутые разбиению (концевые вершины дерева разбиения), которые можно организовывать в виде списка. Если подмножество подвергается разбиению, то оно исключается из списка и заменяется своим разбиением.

Замечание 2. Если в исходной задаче функционал минимизируется, то строятся нижние оценки $\xi(r,t)$, т.е. $\xi(r,t) \le min\{\varphi(x) \mid x \in X(r,t)\}$, и M достаточно большое число.

Алгоритм Ленд-Дойга

Рассмотрим алгоритм метода ветвей и границ для задачи (1-4). Множество $X(0,1)$ есть множество решений задачи. Оценку $\xi(0,1)$ полагаем равной максимуму функционала (1) при ограничениях (2,3), т.е. исходная задача без условий целочисленности. Если при построении $\xi(0,1)$ получилось нецелочисленное решение, то множество $X(0,1)$ разбивается на два подмножества $X(1,1)$ и $X(1,2)$ следующим образом. Пусть в оптимальном решении задачи (1-3) $x_j = b(j)$, где $b(j)$ не целое, тогда множество $X(1,1)$ есть множество решений системы

$$A\,x = b, \qquad (5)$$
$$x \ge 0, \qquad (6)$$
$$x(j) \le b(j). \qquad (7)$$
$$x(i) \text{ - целые, } i \in [1..n]. \qquad (8)$$

Множество $X(1,2)$ есть множество решений системы

$$A\,x = b \qquad (9)$$
$$x \ge 0, \qquad (10)$$
$$x(j) \ge b(j)+1, \qquad (11)$$
$$x(i) \text{ - целые, } i \in [1..n]. \qquad (12)$$

Оценки $\xi(1,1)$, $\xi(1,2)$ получаются максимизацией функционала (1) при ограничениях (5-7) и (9-12). Заметим, что для построения этих оценок целесообразно к последней симплекс-таблице задачи (1-3) добавлять соответственно ограничения (7) и (11) и решать задачи двойственным симплекс-методом.

Дальнейшие разбиения и построение оценок проводятся аналогично.

Задание.

Решить задачу из лабораторной работы N13 "Лабораторного практикума по методам оптимизации", добавив к условиям задачи условия целочисленности переменных. Если при решении задачи (1-3) решение оказалось целочисленным , то задание скорректировать у преподавателя.

Пример.

$$x(1) + x(2) \rightarrow max \quad (13)$$
$$2x(1) + 11x(2) \leq 38 \quad (14)$$
$$x(1) + x(2) \leq 7 \quad (15)$$
$$4x(1) \quad 5x(2) \leq 5 \quad (16)$$
$$x(1) \geq 0 \quad (17)$$
$$x(2) \geq 0 \quad (18)$$
$$x(1), x(2) - \text{целые}. \quad (19)$$

0-**ой шаг**. Множество $X(0,1)$ состоит из всех решений задачи *(13-19)*. Для получения оценки $\xi(0,1)$ решаем задачу *(13-18)*.После решения этой задачи симплекс-методом последняя симплекс-таблица будет иметь вид:

x_b	b	$x(1)$	$x(2)$	$y(1)$	$y(2)$	$y(3)$
$y(1)$	1.00	0.00	0.00	1.00	-6.00	1.00
$x(2)$	2.56	0.00	1.00	0.00	0.44	-0.11
$x(1)$	4.44	1.00	0.00	0.00	0.65	0.11
d	7.00	0.00	0.00	0.00	1.00	0.00

Оценка $\xi(0,1)=7$.Решение $x(1)=4.44$, $x(2)=2.56$ не является целочисленным. Дерево разбиения примет вид

$x(1)=4.44$

$\xi(0,1)=7$

$x(2)=2.56$

1-**ый шаг.** Разбиваем множество $X(0,1)$ на подмножества $X(1,1), X(1,2)$. В качестве переменной, по которой проводим разбиение берем переменную $x(1)$, которая имеет нецелочисленное значение. Можно взять и переменную x(2).

Для подмножества $X(1,1)$ дополнительное ограничение будет иметь вид $x(1) \leq 4$.

Добавляем его к последней симплекс-таблице подмножества X(0,1), получим:

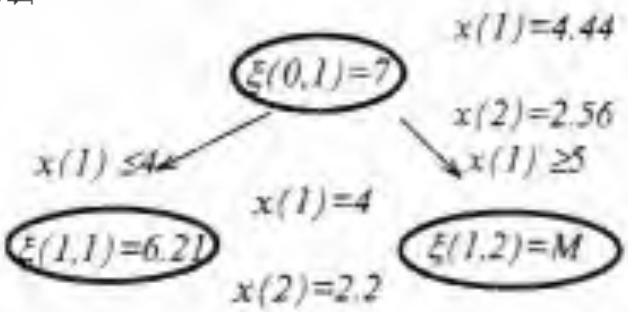
x_b	b	$x(1)$	$x(2)$	$y(1)$	$y(2)$	$y(3)$	$y(4)$
$y(1)$	1.00	0.00	0.00	1.00	-6.00	1.00	0.00
$x(2)$	2.56	0.00	1.00	0.00	0.44	-0.11	0.00
$x(1)$	4.44	1.00	0.00	0.00	0.65	0.11	0.00
$y(4)$	-0.44	0.00	0.00	0.00	-0.56	-0.11	1.00
d	7.00	0.00	0.00	0.00	1.00	0.00	0.00

Решив эту задачу двойственным симплекс-методом, получим таблицу:

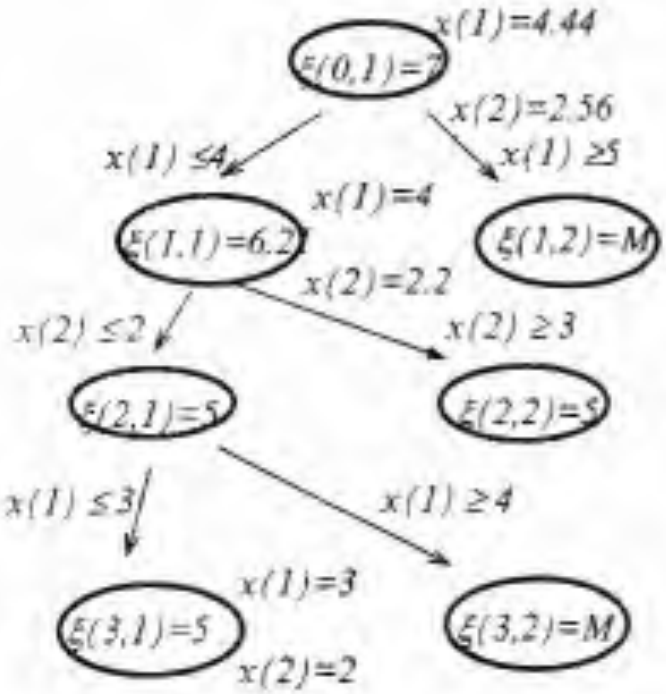
x_b	b	$x(1)$	$x(2)$	$y(1)$	$y(2)$	$y(3)$	$y(4)$
$y(1)$	5.79	0.00	0.00	1.00	0.00	2.20	-10.8
$x(2)$	2.20	0.00	1.00	0.00	0.00	-0.20	0.80
$x(1)$	4.00	1.00	0.00	0.00	0.00	0.00	1.00
$y(2)$	0.80	0.00	0.00	0.00	1.00	0.20	-1.80
d	6.20	0.00	0.00	0.00	0.00	-0.20	1.80

$x(1)=4$, $x(2)=2.2$, $\xi(1,1)=6.2$.

Для подмножества $X(1,2)$ дополнительное ограничение будет иметь вид $x(1)\geq 5$. Добавляем его к последней симплекс-таблице подмножества $X(0,1)$, получим, что задача не имеет решения, т.е. подмножество $X(1,2)=\varnothing$, $\xi(1,2)=M$ (M - достаточно большое число, $M\to\infty$).Дерево разбиения принимает вид:



На последующих шагах дальнейшему разбиению будет подвергаться подмножество $X(1,1)$. Полное дерево разбиения будет иметь вид



На подмножестве $X(3,1)$ получаем целочисленное решение $x(1)=3$, $x(2)=2$ со значением функционала, равным 5, которое принимаем за рекордное. Все подмножества $X(r,t)$, у которых значения $\xi(r,t) \geq 5$, отбраковываем, тогда $x(1)=3$, $x(2)=2$ становится оптимальным решением.

ЛАБОРАТОРНАЯ РАБОТА № 5

Задача оптимизации многошаговых процессов, задача о ранце

Задача оптимизации многошаговых процессов имеет вид

$$\Sigma \{f^{\circ}_i(x(i-1), u(i)) \mid i \in [1..n]\} \rightarrow max \qquad (1)$$

при ограничениях

$$x(0)=\alpha(0), \qquad (2)$$

$$x(i)=f_i(x(i-1), u(i)), \quad i \in [1..n], \qquad (3)$$

$$x(i) \in X(i), \quad i \in [1..n], \qquad (4)$$

$$u(i) \in U(i), \quad i \in [1..n], \qquad (5)$$

где $X(i)$, $U(i)$, $i \in [1..n]$, - конечные множества.
Положим $X(0)=\{\alpha(0)\}$.

Для этой задачи справедливо следующее функциональное соотношение Беллмана:

$$W_n(x)=0, \quad x \in X(n),$$

$$W_i(x)=max\{f^{\circ}_i(x,u)+W_i((f_i(x,u)) \mid u \in U_i(x)\}, \qquad (6)$$

$$x \in X(i), \quad i \in [0..n-1],$$

где $U_i(x)=\{u \in U(i) \mid f_i(x,u) \in X(i+1)\}$.

Для того, чтобы решение $x^*(i)$, $i \in [0..n]$, $u^*(i)$, $i \in [1..n]$, удовлетворяющее ограничениям (2 - 5) было оптимальным, необходимо и достаточно, чтобы

$$f^{\circ}_i(x^*(i),u^*(i+1))+W_i(f_i(x^*(i),u^*(i+1))) = max\{f^{\circ}_i(x^*(i),u)+ W_i(f_i(x^*(i),u)) \mid u \in U_i(x^*(i))\}.$$

Исходя из сказанного выше, получаем следующий алгоритм решения задачи *(1-5)*:

Алгоритм.

Первый этап:

for $x \in X(n)$ *do* $W_n(x) := 0;$
for $i := n-1$ *downto* 0 *do*
for $x \in X(i)$ *do*
begin
$W_i(x) = max\{f^0_i(x,u) + W_i(f_i(x,u)) \mid u \in U_i(x)\};$
$u_i(x) := argmax\{f^0_i(x,u) + W_i(f_i(x,u)) \mid u \in U_i(x)\}$
end;

Второй этап:

$x^*(0) := \alpha(0);$
for $i := 1$ *to* n *do*
begin
$u^*(i) := u_i(x^*(i-1));$
$x^*(i) := f_i(x^*(i-1), u^*(i))$
end;

Полученное в результате работы алгоритма решение $x^*(i)$, $u^*(i)$ $i \in [1..n]$, будет оптимальным для задачи *(1-5)*. Оптимальное значение функционала будет равно $W_0(\alpha(0))$.

Задача о ранце имеет вид

$$\Sigma\{c(i) \times u(i) \mid i \in [1..n]\} \longrightarrow max \qquad (7)$$

при ограничениях

$$\Sigma\{a(i) \times u(i) \mid i \in [1..n]\} \le b \qquad (8)$$

где $u(i)$- целые числа, $a(i)$ также рассматриваем целыми, $i \in [1..n]$.

Введем переменные $x(0)=0$, $x(i) = \Sigma\{a(k) \times u(k): k \in [1..i]\}$. Из задачи (7) - (8) получим эквивалентную задачу (9) - (13) :

$$\Sigma\{c(i) \times u(i) \mid i \in [1..n]\} \longrightarrow max, \qquad (9)$$
$$x(0) = 0, \qquad (10)$$
$$x(i) = x(i-1) + a(i) \times u(i), \quad i \in [1..n], \qquad (11)$$
$$x(i) \in X(i) = \{0,1,2,...,b\}, \; i \in [1..n], \qquad (12)$$
$$u(i) \in U(i) = \{0,1,2,...,b\}, \; i \in [1..n]: \qquad (13)$$

Задача (9) - (13) имеет такой же вид, как и задача (1) - (5),поэтому для решения задачи (8) - (12) возможно применение алгоритма, описанного выше.

Задание.

Решить задачу о ранце при $n=5$, $b=8$, значения $c(i)$, $a(i)$, $i \in [1..n]$, взять из следующей таблицы в соответствии с номером варианта.

N	C	A
1	2, 4, 4, 3 ,1.	2, 5, 3, 3 ,4
2	3, 4, 5, 2 , 2	2 ,3, 4, 5 ,1
3	4 ,5, 2, 2 .3	3 ,4 ,5, 1, 2
4	5, 2, 2, 3, 4	4 ,5, 1, 2 ,3
5	2, 2, 3, 4 ,5	5, 1, 2, 3 .4
6	2 ,3 ,4, 5 ,2	1, 2, 3, 4 ,5
7	2 ,4 ,3, 5 ,1	2, 5, 1, 3 ,4
8	4, 3, 5, 1, 2,	5 ,1 ,3, 4 ,2
9	3, 5. 1, 2 ,4	1, 3, 4. 2 ,5
10	4 .5, 1, 2, 3	3 ,4, 2, 5. 1
11	5, 1, 2, 3 ,4	4, 2, 5, 1 ,3
12	5 ,1 ,3, 2 ,4	4 ,2 ,5, 3, 1
13	1, 3, 2. 4 ,5	2 ,5, 3, 1 ,4
14	3, 2, 4. 5, 1	5 ,3, 1, 4 ,2
15	2. 4, 5, 1, 3	3 ,1, 4, 2 ,5
16	4 .5 .1, 3 .2	1, 4, 2, 5 ,3
17	5 ,1, 3, 2 ,4	4 ,2 .5, 3, 1
18	1, 3, 2, 4 ,5	2 ,5, 3, 1 ,4
19	3 ,2, 4, 5 ,1	5 ,3, 1, 4 ,2
20	2 ,4, 5. 1, 3	3 ,1, 4, 2 ,5
21	4, 5, 3, 1 ,2	1, 4. 5, 2 ,3
22	5 ,3, 1, 2 .4	4 ,5 ,2, 3 ,1
23	3 ,1. 2, 4 ,5	5 ,2, 3, 1 ,4
24	1, 2, 4, 5 ,3	2, 3, 1, 4 ,5
25	2, 4, 5, 3, 1	3, 1, 4, 5 ,2
26	4, 5, 3, 1 ,2	1, 4, 5, 2 ,3

Результаты представить в виде таблицы

x	i=0		i=1		i=2		i=3		i=4		i=5	
	W_0	u_0	W_1	u_1	W_2	u_2	W_3	u_3	W_4	u_4	W_5	u_5
0												
1												
2												
3												
4												
5												
6												
7												
8												

А также выписать оптимальное значение функционала и значения переменных, на которых он достигается.

Пример.

Решить задачу:

$$u_1+2u_2 \to max$$
$$2u_1+u_2 \le 3$$
$$u_1, u_2 - \text{целые}.$$

В этой задаче n–2, b–3, поэтому таблица с результатами вычислений будет иметь вид:

x	$i=0$		$i=1$		$i=2$
	W_0	u_1	W_1	u_2	W_2
0	6	0	6	3	0
1			4	2	0
2			2	1	0
3			0	0	0

Запишем соотношение Беллмана:

$$W_i(x) = max\ \{(C_{i+1}u+W_{i+1}(x+A_{i+1}u))\ \ u \in U_{i+1}(x),$$
$$U_{i+1}(x) = \{0,1, \dots, [(b-x)/A_{i+1}]\},$$
$$u_{i+1}(x) = argmax\{ C_{i+1}u+W_{i+1}(x+A_{i+1}u) \mid u \in U_i(x)\}.$$

Первый этап:

$W_2(x)=0,\ x \in \{0,1,2,3\}$ - (смотри последний столбец предыдущей таблицы)

$W_1(x)= max[\ 2u+W_2(x+u)]\ ,\ x \in \{0,1,2,3\},\ u \in U_2(x),\ U_2(0)=\{0,1,2,3\},$
$W_1(0)=max[0,2,4,6]=6,\ u_2(0)=3,\ U_2(1)=\{0,1,2\},$
$W_1(1)= max[0,2,4]=4,\ u_2(1)=2,\ U_2(2)= \{0,1\},$
$W_1(2)= max[0,2]=2,\ u_2(2)=1,\ U_2(3)=\{0\},$
$W_1(3)=max[0]=0,\ u_2(3)=0,$
$W_0(x)= max[\ u+W_1(x+2u)],\ u \in U_1(x_i),\ U_1(0)=\{0,1\}$
$W_0(0)= max[u+\ W_1(0+2u)]=max[6,5]=6,\ u \in \{0,1\},\ u_1(0)=0.$

Второй этап:

$x_0^*=0,\ u_1^*=u_1(x_0^*)=u_1(0)=0,\ x_1^*=x_0+A_1u_1^*=0;$
$u_2^*=u_2(x_1^*)=u_2(0)=3,\ x_2=x_1^*+A_2u_2^*=3.$

Итак, значение функционала составит $W_0(0)=6$, при этом
$u_1=0,\ u_2=3;\ x_0=0,\ x_1=0,\ x_2=3.$

ЛАБОРАТОРНАЯ РАБОТА № 6

Метод ветвей и границ для решения задачи о коммивояжере

Пусть имеется $(n+1)$ городов $A(0),A(1),A(2),A(3),...,A(n)$. Между всеми парами городов известно расстояние $c(i,j) \geq 0$. Коммивояжер, который находится в городе $A(0)$, должен обойти все города, побывав в каждом ровно один раз и вернуться в город $A(0)$ так, чтобы длина пройденного пути была наименьшей.

Путь коммивояжера $(i(0),i(1),i(2),...,i(n),i(0))$ образует замкнутый маршрут. Его длина

$$l(i(0),i(1),i(2),...,i(n),i(0))=c(i(0),i(1))+c(i(1),i(2)) + c(i(2),i(3)) + ...+c(i(n),i(0)).$$

Не трудно видеть, что число таких маршрутов равно $n!$. Для поиска минимального (оптимального) маршрута применим метод ветвей и границ. Общая схема этого метода приведена в лабораторной работе "Метод ветвей и границ (алгоритм Ленд и Дойга) для решения задач целочисленного линейного программирования". Дадим описание этого метода для задачи о коммивояжере.

Алгоритм

Задание множества $X(0,1)$ и вычисление оценки $\xi(0,1)$

Множество $X(0,1)$ есть множество всех возможных маршрутов коммивояжера. Оценку $\xi(0,1)$ получаем, используя процедуру приведения матрицы $C= (c(i,j)),\ i\in[0..n],\ j \in [0..n]$, следующим образом.

Для всех $i\in[0..n],\ h(i):= min\{c(i,j) \mid j\in[0..n]\}$,

$c'(i,j):= c(i,j) - h(i),\ i \in [0..n],\ j \in [0..n]$.

Для всех $j\in[0..n],\ H(j):= min\{c'(i,j) \mid i\in[0..n]\}$,

$c''(i,j):= c'(i,j) \quad H(i),\ i \in [0..n],\ j \in [0..n]$.

Переменные $h(i),H(j)$ называют коэффициентами приведения. Матрицу $C''= (c''(i,j)),\ i\in[0..n],\ j\in[0..n]$, называют приведенной матрицей.

Полагаем $\xi(0,1)=\Sigma\{h(i) \mid i\in[0..n]\} +\Sigma\{H(j) \mid j\in[0..n]\}$.

Разбиение множества $X(r,t)$ на подмножества

Пусть построено подмножество $X(r,t)$ и нижняя оценка функционала $\xi(r,t)$ на нем. Этому подмножеству соответствует приведенная матрица $C''(r,t)$. Подмножество разбиваем на два: $X(r+1,m+1)$, $X(r+1,m+2)$, каждому из которых будут соответствовать свои оценки $\xi(r+1,m+1)$,

$\xi(r+1,m+2)$ и приведенные матрицы $C''(r+1,m+1)$, $C''(r+1,m+2)$. Подмножество $X(r+1,m+1)$ получается из $X(r,t)$ добавлением условия: из пункта p непосредственно идти в пункт q. Подмножество $X(r+1,m+2)$ получается добавлением условия: из пункта p непосредственно в пункт q идти запрещается. Пару (p,q) выбирают таким образом, чтобы с наибольшей вероятностью подмножество $X(r+1,m+1)$ содержало оптимальный цикл, а $X(r+1,m+2)$ не содержало его. Для этого пару (p,q) выбирают таким образом, чтобы $c(p,q)=0$, при этом, чтобы величина $min\ \{c(p,j)j\neq q\} + min\ \{c(i,q)\ i\neq p\}$была максимальной, где $c(.,.)$ берутся из матрицы $C''(r,t)$

Преобразование матриц расстояний, пересчет оценок

Матрицу $C''(r+1,m+2)$ строим следующим образом. Полагаем $C(r+1,m+2)=C(r,t)$, в ней $c(p,q):=M$, где M - достаточно большое число, принимаемое за бесконечность. Матрица $C''(r+1,m+2)$ получается из $C(r+1,m+2)$ процедурой приведения. Для получения $\xi(r+1,m+2)$ складываем $\xi(r,t)$ с полученными коэффициентами приведения. Матрицу $C''(r+1,m+1)$ строим следующим образом. Полагаем $C(r+1,m+1)=C(r,t)$, и в ней вычеркиваем строку p и столбец q . Налагаем условие малых циклов. Для этого восстанавливаем части маршрута, образованные непосредственными переходами в $X(r+1,m+1)$. Если добавление любой пары (i,j) к этим маршрутам образует малый цикл, то в матрице $C(r+1,m+1)$ $c(i,j):=M$. Матрица $C''(r+1,m+1)$ получается из $C(r+1,m+1)$ процедурой приведения. Для получения $\xi(r+1,m+2)$ складываем $\xi(r,t)$ с полученными коэффициентами приведения.

Задание.

Построить матрицу расстояний при $n=5$, значения $c(i,j)$ выбрать произвольно и решить задачу коммивояжера.

Пример.

Матрица расстояний имеет вид

$C(0,1)$

i \ j	0	1	2	3	4	5	h(i)
0	M	4	10	13	4	8	4
1	2	M	9	7	6	7	2
2	8	5	M	5	5	9	5
3	5	8	5	M	7	10	5
4	6	4	4	9	M	4	4
5	5	1	4	8	3	M	1
H(j)							

Шаг *0.*

Приводим матрицу *C(0,1)*. Находим *h(i)* и вычитаем из *c(i,j)*.

C'(0,1)

i \ j	0	1	2	3	4	5	h(i)
0	M	0	6	9	0	4	
1	0	M	7	5	4	5	
2	3	0	M	0	0	4	
3	0	3	0	M	2	5	
4	2	0	0	5	M	0	
5	4	0	3	7	2	M	
H(j)	0	0	0	0	0	0	

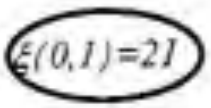
Величины *H(i)* записаны в последней строке этой таблицы, вычитаем их из *c'(i,j)*, получим следующую таблицу

C"(0,1)

i \ j	0	1	2	3	4	5	h(i)
0	M	0	6	9	0	4	
1	0	M	7	5	4	5	
2	3	0	M	0	0	4	
3	0	3	0	M	2	5	
4	2	0	0	5	M	0	
5	4	0	3	7	2	M	
H(j)							

Оценка $\xi(0,1)=21$.

Строим дерево разбиения



Шаг 1.

Разбиваем множество *X(0,1)* на подмножества *X(1,1),X(1,2)*. В качестве пары *(p,q)* берем *(2,3)*. Для нее $c''(2,3)=0$ и $(\min\{c(2,j)|j\neq 3\} + \min\{c(i,3)|\ i\neq 2\})$ максимально. Для подмножества *X(1,1)* из пункта *2* идти в пункт *3*. Матрица расстояний будет иметь вид:

C(1,1)

i \ *j*	*0*	*1*	*2*	*4*	*5*	*h(I)*
0	*M*	*0*	*6*	*0*	*4*	
1	*0*	*M*	*7*	*4*	*5*	
3	*0*	*3*	*0*	*2*	*5*	
4	*2*	*0*	*0*	*M*	*0*	
5	*4*	*0*	*3*	*2*	*M*	
H(j)						

Для запрета малых циклов запрещаем переход из пункта *3* в пункт *2*, получаем

C(1,1)

i \ *j*	*0*	*1*	*2*	*4*	*5*	*h(i)*
0	*M*	*0*	*6*	*0*	*4*	
1	*0*	*M*	*7*	*4*	*5*	
3	*0*	*3*	*M*	*2*	*5*	
4	*2*	*0*	*0*	*M*	*0*	
5	*4*	*0*	*3*	*2*	*M*	
H(j)						

Все коэффициенты приведения равны *0*, поэтому $C''(1,1)=C(1,1)$, $\xi(1,1)=\xi(0,1)=21$.

Для подмножества *X(1,2)* из пункта *2* непосредственно идти в пункт *3* запрещается. Матрица расстояний будет иметь вид

C(1,2)

i \ *j*	*0*	*1*	*2*	*3*	*4*	*5*	*h(i)*
0	*M*	*0*	*6*	*9*	*0*	*4*	*0*
1	*0*	*M*	*7*	*5*	*4*	*5*	*0*
2	*3*	*0*	*M*	*M*	*0*	*4*	*0*
3	*0*	*3*	*0*	*M*	*2*	*5*	*0*
4	*2*	*0*	*0*	*5*	*M*	*0*	*0*
5	*4*	*0*	*3*	*7*	*2*	*M*	*0*
H(j)	*0*	*0*	*0*	*5*	*0*	*0*	

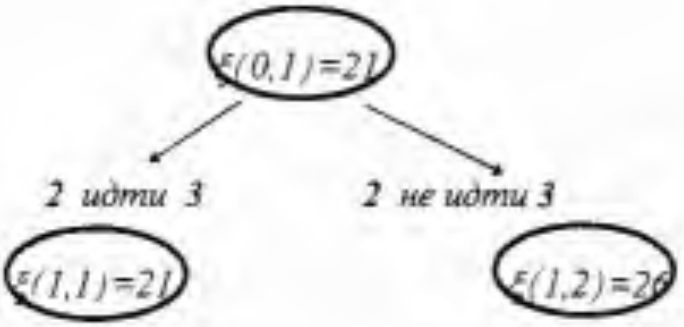
После приведения получим

$C''(1,2)$

i \ j	0	1	2	3	4	5	h(i)
0	M	0	6	4	0	4	
1	0	M	7	0	4	5	
2	2	2	M	M	0	4	
3	0	3	0	M	2	5	
4	2	0	0	0	M	0	
5	4	0	3	2	2	M	
H(j)							

$\xi(1,2)=21+5=26.$

Достраиваем дерево разбиения



Шаг 2.

Минимальная оценка $\xi(1,1)$, поэтому разбиваем подмножество$X(1,1)$. В качестве пары (p,q), относительно которой проводим ветвление,используем пару $(4,5)$. Для подмножества $X(2,1)$

$C(2,1)$

i \ j	0	1	2	4	h(i)
0	M	0	6	0	
1	0	M	7	4	
3	0	3	M	2	
5	4	0	3	M	
H(j)			3		

В этой матрице с целью запрета малых циклов запрещен переход $5\rightarrow 4$. $H(2)=3$, поэтому $\xi(2,1)=21+3=24$.

C''(2,1)

i \ j	0	1	2	4	h(i)
0	M	0	3	0	
1	0	M	4	4	
3	0	3	M	2	
5	4	0	0	M	
H(j)					

Для подмножества $X(2,2)$

C(2,2)

i \ j	0	1	2	4	5	h(i)
0	M	0	6	0	4	
1	0	M	7	4	5	
3	0	3	M	2	5	
4	2	0	0	M	M	
5	4	0	3	2	M	
H(i)					4	

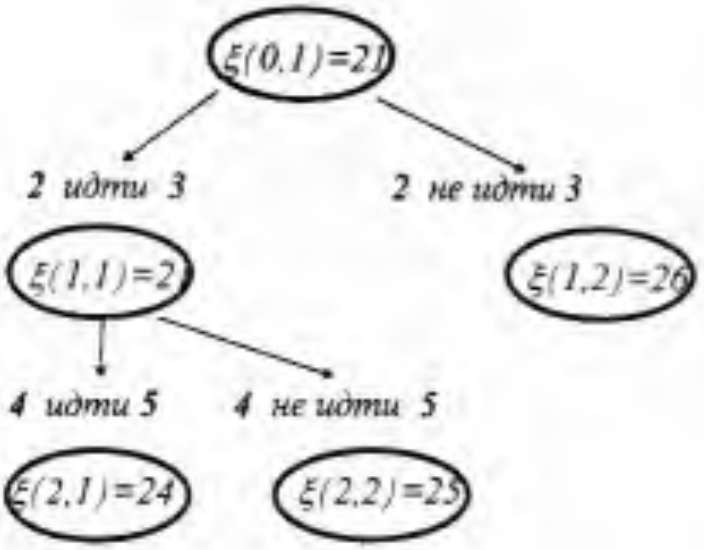
После приведения получим

C''(2,2)

i \ j	0	1	2	4	5	h(i)
0	M	0	6	0	0	
1	0	M	7	4	1	
3	0	3	M	2	1	
4	2	0	0	M	M	
5	4	0	3	2	M	
H(j)						

$\xi(2,2)=21+4=25.$

Достраиваем дерево разбиения



Шаг 3.

Минимальная оценка $\xi(2,1)$, поэтому разбиваем подмножество $X(2,1)$. В качестве пары (p,q), относительно которой проводим ветвление, используем пару $(1,0)$.

Подмножество $X(3,1)$:

$C''(3,1)$

i \ j	1	2	4	h(i)
0	M	3	0	
3	1	M	0	
5	0	0	M	
H(j)				

$\xi(3,1)=26.$

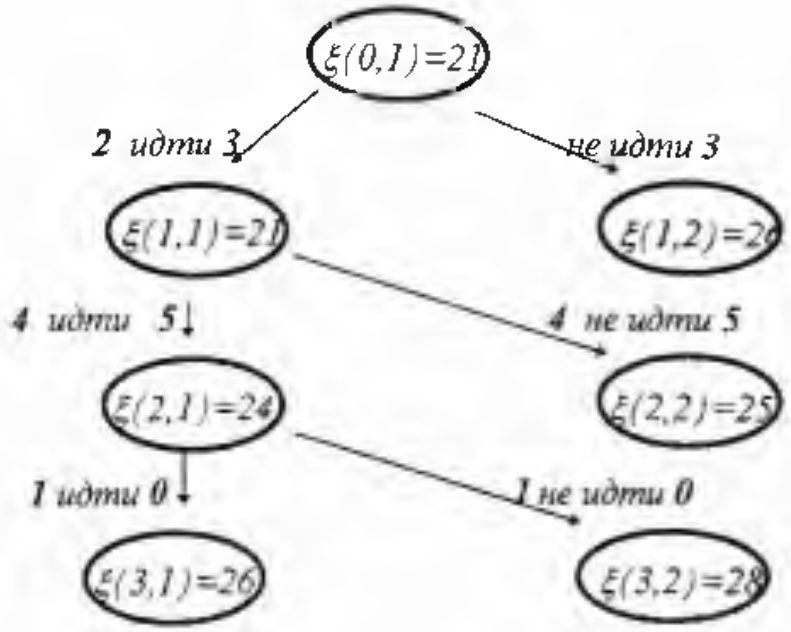
Подмножество $X(3,2)$:

$C''(3,2)$

I \ j	0	1	2	4	h(i)
0	M	0	3	0	
1	M	M	0	0	
3	0	3	M	2	
5	4	0	0	M	
H(j)					

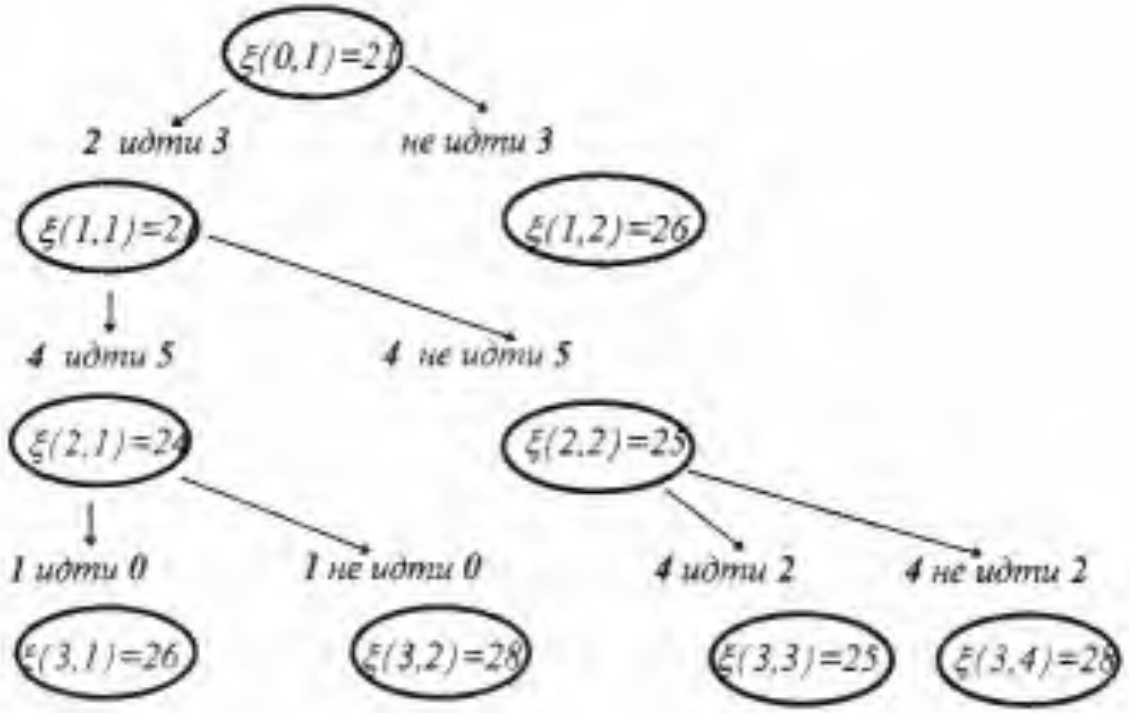
$\xi(3,2)=28.$

Дерево разбиения примет вид

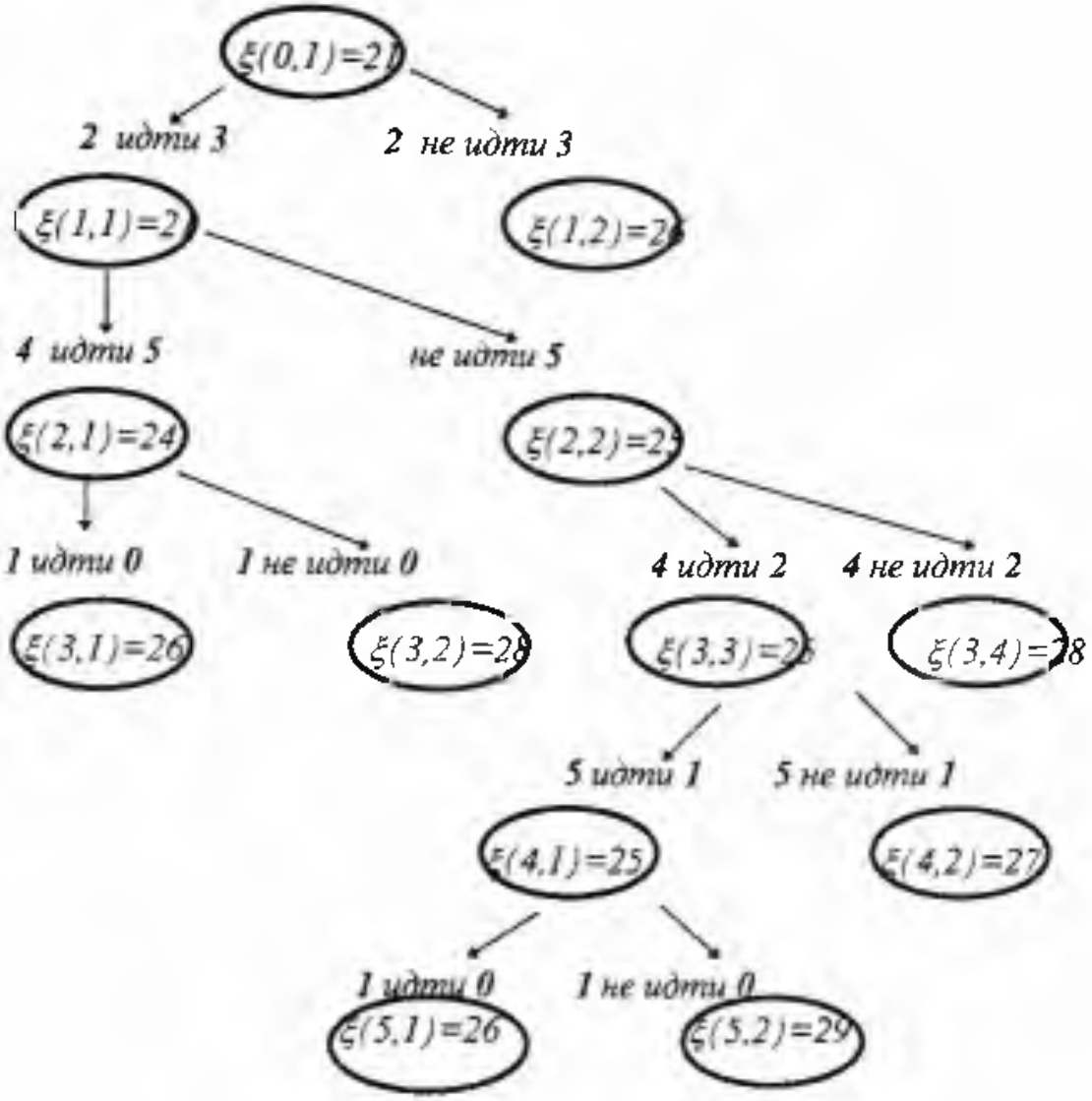


Шаг 4.

Минимальная оценка $\xi(2,2)$, поэтому разбиваем подмножество $X(2,2)$. В качестве пары (p,q), относительно которой проводим ветвление, используем пару $(4,2)$. После его выполнения дерево разбиения примет вид:



На последующих шагах получим



Для подмножества $X(5,1)$ будем иметь:

$C''(3,2)$

i \ j	4	5	h(i)
0	0	M	
3	M	0	
H(j)			

т.е. остаются переходы из *0* в *4*, из *3* в *5*. Этот переход и переходы, выбранные при движении по дереву от вершины для подмножества $X(5,1)$ до вершины подмножества $X(0,1)$, получаем цикл $0 \to 4 \to 2 \to 3 \to 5 \to 1 \to 0$, длина которого

$l=26$. Так как все $\xi(r,t)$ для концевых вершин дерева ≥ 26, то этот цикл оптимальный.

Если требуется найти все оптимальные решения,то работу алгоритма необходимо продолжить, т.к. на подмножестве $X(3,1)$ оценка $\xi(3,1)=26$ и на этом подмножестве могут быть оптимальные решения.

ЛАБОРАТОРНАЯ РАБОТА № 7

Транспортная задача в сетевой постановке

Пусть $G=<E,V,H>$ связный ориентированный граф, где E множество вершин, V множество ориентированных дуг, H - отображение $H:V\to E\times E$, $H(v)=(h1(v),h2(v))$. Вершину $h1(v)$ будем называть началом дуги v, $h2(v)$ - ее концом. $V^+(i)$ множество дуг, входящих в вершину i, $V^-(i)$ множество дуг, выходящих из вершины i. Все вершины графа делятся на пункты производства (источники) и пункты потребления (стоки) потока. Для каждой вершины $i\in E$ известна величина, задающая объем производства или потребления i-ой вершины, причем если $b(i)<0$, то вершина i пункт производства (источник), если $b(i)>0$, то i-ая вершина пункт потребления (сток) потока, если $b(i)=0$, то вершина i - промежуточная вершина.

Для каждой дуги $v\in V$ задана величина $c(v)$, характеризующая стоимость перевоза единицы груза по данной дуге. Задача состоит в том, чтобы определить количество перевозимого груза $x(v)$ по каждой дуге $v\in V$ при условии, что весь поток из пунктов производства будет вывезен, потребности всех пунктов потребления будут удовлетворены и суммарная стоимость транспорта потока будет наименьшей.

Приведем формализованную постановку данной задачи:

$$L(x)=\sum\{c(v)\times x(v) \mid v\in V\}\to min \qquad (1)$$

при ограничениях

$$\sum\{x(v) \mid v\in V^+(i)\}-\sum\{x(v) \mid v\in V^-(i)\}=b(i),\ i\in E, \qquad (2)$$

$$x(v)\geq 0\ ,\quad v\in V. \qquad (3)$$

Здесь (1) минимизируемый функционал (суммарная стоимость перевоза потока). (2) - уравнение материального баланса, т.е. суммарный поток, входящий в вершину, минус суммарный поток, выходящий из вершины, должен быть равен объему потока, производимому (потребляемому) в i-ой

пункте. Соотношение (3) означает, что транспортируемый поток движется только в направлении дуг графа G.

Эта задача разрешима, если $\sum \{ b(i) | i \in E \} = 0$. Легко видеть, что она является задачей линейного программирования и может быть решена алгоритмом симплекс-метода. Однако сетевая структура задачи позволяет разработать более эффективные методы ее решения. Таковым является метод потенциалов для решения транспортных задач.

Метод потенциалов решения данной задачи предполагает, что на исходном графе некоторым способом определено начальное исходное базисное дерево $G'=<E,V',H>$, (его дуги будем называть базисными), по которому осуществляется перевозка груза таким образом, что для $v \in V' \; x(v) \geq 0$, и для всех внебазисных дуг v, т.е. для $v \in V | V' \; x(v)=0$. Для определения величин $x(v) \in V'$ можно воспользоваться следующим алгоритмом:

Алгоритм 1.

```
begin
  while ( |E| >1 ) do
  begin
    for (i∈E) do
    if ( |V'+(i)| + |V'-(i)| = 1 ) then {т.е.вершина i концевая на
                                          дереве G}
    begin
      if ( |V'+(i)| = 1 ) then
      for (v∈V'+(i)) do
      begin
        x(v):=b(i);
        b(h1(v)):=b(h1(v))+x(v);
      end;
      if ( |V'-(i)| = 1 ) then
      for v∈V'-(i) do
      begin
        x(v) := -b(i);
        b(h2(v)):=b(h2(v))-x(v);
      end;
      удаляем из графа G'=<E,V',H> вершину i и
      инцидентную ей дугу.
    end;
  end;
end.
```

Замечание.

В результате работы алгоритма может получиться недопустимое решение: т.е. для некоторых $v \in V$ $x(v)<0$. В этом случае следует сменить исходное базисное дерево. Однако это не гарантирует, что на нем будет получено допустимое решение. Далее будет описан алгоритм, позволяющий либо определить базисное дерево с допустимым решением, либо доказать, что его не существует.

Будем считать, что исходное базисное дерево G' найдено и для $v \in V'$ определены $x(v)>0$. Для того, чтобы определить, является ли полученное решение оптимальным, воспользуемся следующим критерием оптимальности:

КРИТЕРИЙ ОПТИМАЛЬНОСТИ:

Пусть $x(v)$, $v \in V$, - такое решение задачи *(1 - 3)*, что для $v \in V \setminus V'$ $x(v)=0$ и для $v \in V'$ $x(v)>0$. Это решение оптимально тогда и только тогда, когда существуют числа $u(i)$, $i \in E$, называемые потенциалами, такие, что

$$u(h2(v))=u(h1(v))+c(v), \quad v \in V' \quad (4)$$

$$u(h2(v)) \le u(h1(v))+c(v), \quad v \in V \setminus V'. \quad (5)$$

Для определения потенциалов применяется следующий алгоритм:

Алгоритм 2.

```
begin
   для произвольной (только одной) вершины i∈E
   u(i):= 0;
   M1:
   найти дугу v∈V', для которой известен
   потенциал только одной из ее вершин,
   если такой дуги нет, то конец работы
   алгоритма, иначе
   begin
      определяем неизвестный потенциал
      вершины дуги v из уравнения
      u(h2(v))=u(h1(v))+c(v);
      goto M1
   end;
end;
```

Для проверки критерия оптимальности найденного решения используется алгоритм:

Алгоритм 3.

1. Среди всех дуг $v \in V \setminus V'$ ищем дугу v_0 такую, что $u(h2(v_0)) > u(h1(v_0)) + c(v_0)$;
2. Если такой дуги нет, то исходная задача (1-3) решена, иначе следует выполнить алгоритм 4 перехода к новому базисному дереву.

Алгоритм 4.

$V' := V' \cup \{v_0\}$, где v_0 дуга, найденная в алгоритме 3. Теперь граф $G' = <E, V', H>$ содержит ровно один цикл $G'' = <E'', V'', H>$, причем $v_0 \in V'$. На подграфе G'' задаем направление обхода, совпадающее с направлением дуги v_0 и пропускаем по подграфу G'' в направлении обхода дополнительный поток, величиной θ. Каждой дуге $v \in V''$ приписываем символ '$+\theta$', если направление дуги v совпадает с направлением обхода, и символ '$-\theta$' в противном случае. Находим $\theta = min\ x(v)$ среди $v \in V''$, которым приписан символ '$-\theta$'. Полагаем $x(v_0) := \theta$; Для всех $v \in V'' \setminus \{v_0\}$:

- $x(v) := x(v) + \theta$, если дуге v приписан знак $+\theta$;
- $x(v) := x(v) - \theta$, если дуге v приписан знак $-\theta$.

Из множества $V'' \setminus \{v_0\}$ исключаем дугу, для которой $x(v) = 0$. Если таких дуг несколько, то удаляем только одну так, чтобы не нарушилась связность графа $G' = <E, V', H>$.

Для полученного решения заново вычисляем Алгоритмом 2 потенциалы и анализируем его на оптимальность алгоритмом 3 и т.д. до тех пор, пока не будет найдено оптимальное решение.

Пример.

Рассмотрим граф, изображенный на рис. 1.

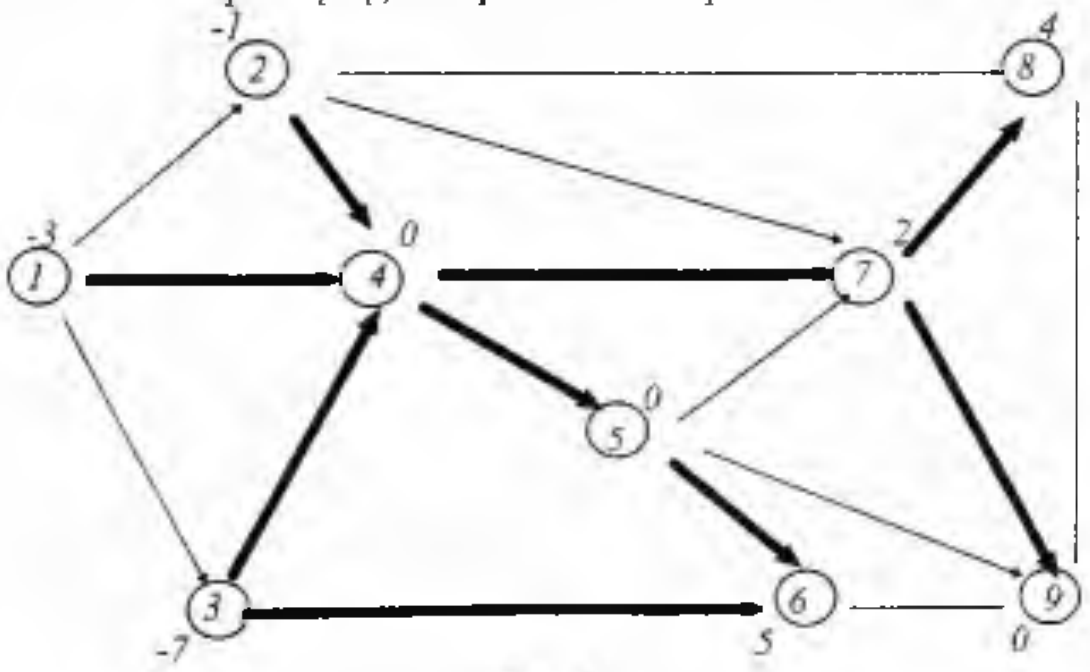


Рис. 1.

Вершины графа обозначены цифрами от 1 до 9, обведенными в кружочек. Рядом с каждой вершиной проставлены мощности вершин:

b_1=-3; b_2=-1; b_3=-7; b_4=0; b_5=0; b_6=5; b_7=2; b_8=4; b_9=0.

Стоимость перевозок задана таблицей 1.

Таблица 1

c_{14}	c_{15}	c_{16}	c_{24}	c_{25}	c_{26}	c_{34}	c_{35}	c_{36}	c_{45}	c_{46}	c_{57}	c_{58}	c_{59}	c_{67}	c_{68}	c_{69}	c_{78}
1	2	1	6	1	1	1	2	2	4	1	3	1	7	6	1	3	2

За исходный базис примем дерево, дуги которого выделены жирной линией (см. рис. 1).

1 шаг.

Определим объем перевозок для исходного базисного дерева.

x_{14} =3; x_{24}=1; x_{34}=7; x_{78}=4; x_{56}=5;

x_{45}=5 ; x_{47}=2+0+4=6 ; x_{79}=0.

Общая стоимость перевозок составит

$$f(X) = c_{14} \times x_{14} + c_{24} \times x_{24} + c_{34} \times x_{34} + c_{78} \times X_{78} + c_{56} \times x_{56} + c_{45} \times x_{45} + c_{47} \times x_{47} + c_{79} \times x_{79} = 52$$

Определим потенциалы вершин:

u_9=0;

$u_7 = u_9 - c_{79}$ = -3;

$u_8 = u_7 + c_{78}$ =-2;

$u_4 = u_7 - c_{47}$ =-7;

$u_5 = u_4 + c_{45}$ =-6;

$u_6 = u_5 + c_{56}$ =-5;

$u_3 = u_4 - c_{34}$ =-8;

$u_1 = u_4 - c_{14}$ =-9;

$u_2 = u_4 - c_{24}$ =-8;

Проверим полученное решение на оптимальность:

На дуге (2,7) $u_7 > u_2 + c_{27}$ следовательно, дугу (2,7) вводим в базис. Возникший цикл (см. рис.2) обходим в направлении дуги (2,7).

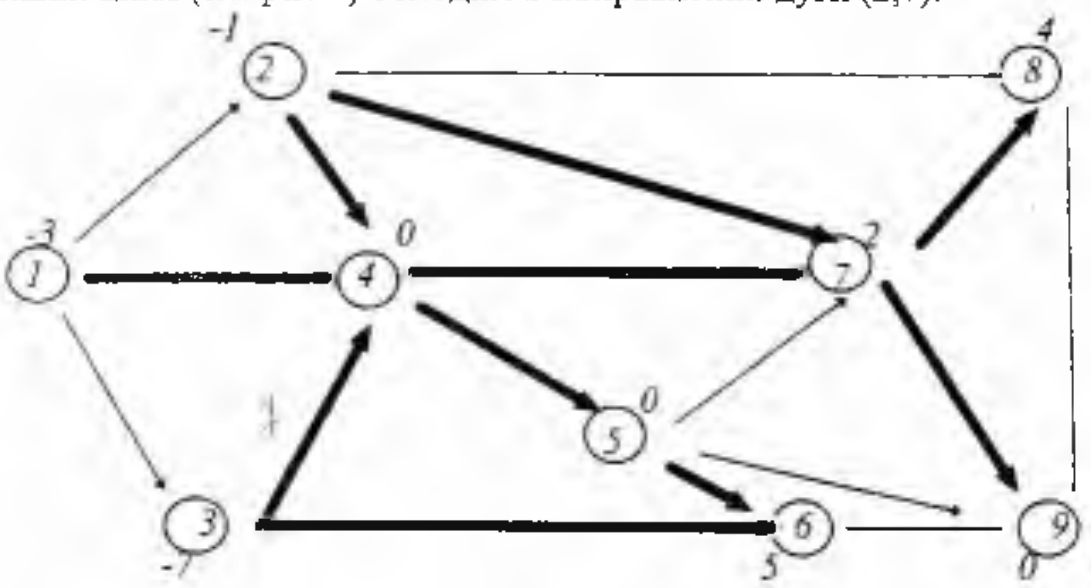


Рис. 2

$x_{27} = \theta$; $x_{24} = 1 - \theta$; $x_{47} = 6 - \theta$;

$\theta = min\ (1,6) = 1$; $x_{24} = 0$ дугу (2,4) выводим из базиса.

2 шаг.

Пересчитаем объем перевозок для нового базиса:

$x_{14} = 3$, $x_{27} = 1$; $x_{34} = 7$; $x_{78} = 4$,

$x_{56} = 5$; $x_{45} = 5$; $x_{47} = 6 - 1 = 5$, $x_{79} = 0$;

$f(X) = 48$.

Определим потенциалы вершин.

$u_1 = -9$;

$u_2 = -8$;

$u_3 = -8$;

$u_4 = -7$;

$u_5 = -6$;

$u_6 = -5$;

$u_7 = -3$;

$u_8 = -2$;

$u_9 = 0$.

Проверим полученное решение на оптимальность:

На дуге (3.6) $u_6 > u_3 + c_{36}$ Следовательно, дугу (3,6) вводим в базис.

Возникший цикл обходим в направлении дуги (3,6) . см. рис.3.

$x_{36} = \theta$; $x_{34} = 7 - \theta$; $x_{45} = 5 - \theta$; $x_{56} = 5 - \theta$;

$\theta = min\ (7,5,5) = 5$ $x_{56} = 0$ дугу (5,6) выводим из базиса.

3 шаг.

Пересчитаем объем перевозок для нового базиса:

$x_{14} = 3$; $x_{27} = 1$; $x_{34} = 7-5 = 2$;

$x_{78} = 4$; $x_{36} = 5$; $x_{45} = 5-5 = 0$, $x_{47} = 5$; $x_{79} = 0$

Определим потенциалы вершин.

$u_1 = -9$;

$u_2 = -8$;

$u_3 = -8$;

$u_4 = -7$;

$u_5 = -6$;

$u_6 = -6$;

$u_7 = -3$;

$u_8 = -2$;

$u_9 = 0$.

Полученное решение оптимально.

Дерево перевозок изображено на рис.3.

Стоимость перевозок составит:

$L(x) = x_{14}c_{14} + x_{27}c_{27} + x_{34}c_{34} + x_{78}c_{78} + x_{36}c_{36} + x_{45}c_{45} + x_{47}c_{47} + x_{79}c_{79} - 43$

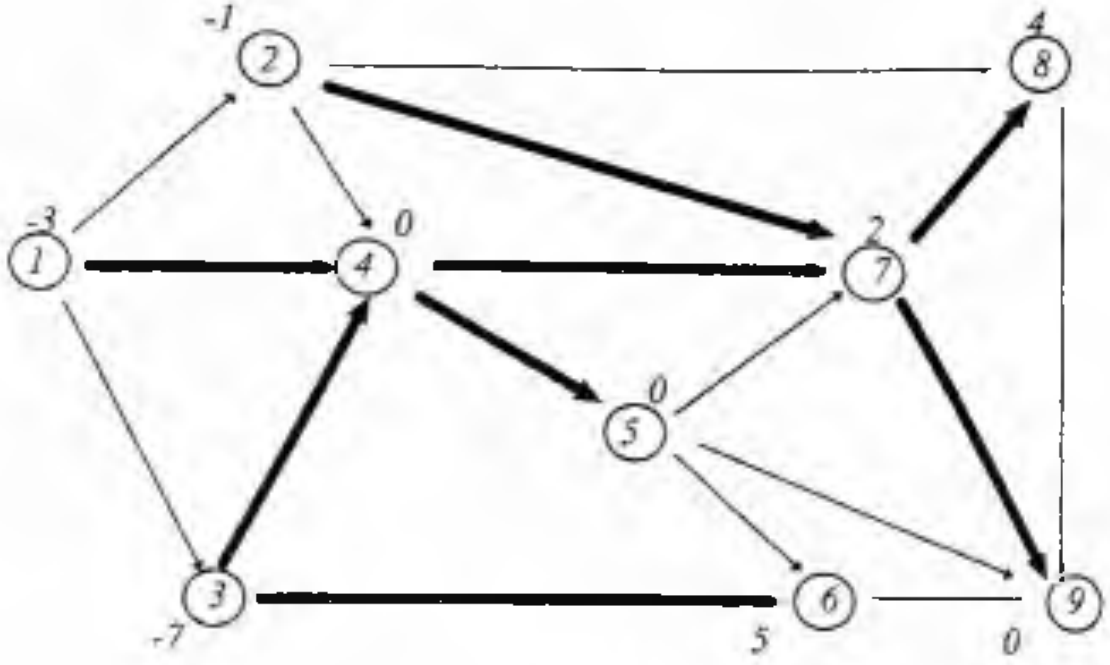


Рис. 3

Задания.

Определить объемы перевозок минимальной стоимости для графа, изображенного на рис.1 (см. пример), мощности вершин и исходное дерево заданы также, как в примере 1, стоимости перевозок задаются таблицей 3, где λ, ε- определяются из таблицы 2.

Таблица 2

№ варианта	λ	ε
1	1	0
2	1	1
3	1	2
4	1	3
5	2	0
6	2	1
7	2	2
8	2	3
9	3	0
10	3	1
11	3	2
12	3	3

Таблица 3

Дуга	Ст-ть перевозки
C_{12}	1+2λ+3ε
C_{14}	2+3λ-ε
C_{13}	1+2λ+3ε
C_{28}	6+2λ+3ε
C_{27}	1+5λ-2ε
C_{24}	1+2λ+3ε
C_{34}	1+2λ-ε
C_{35}	2+λ+3ε
C_{36}	2+2λ-ε
C_{47}	4+λ+3ε
C_{45}	1+2λ+3ε
C_{57}	3+2λ+3ε

Продолжение табл 2. и табл 3

№ варианта	λ	ε
13	4	0
14	4	1
15	4	2
16	4	3
17	5	0
18	5	1
19	5	2
20	5	3
21	6	0
22	6	1
23	6	2
24	6	3
25	7	0
26	7	1
27	7	2
28	7	3

Дуга	Ст-ть перевозки
C_{56}	$1+2\lambda+\varepsilon$
C_{59}	$7+2\lambda+3\varepsilon$
C_{69}	$6+2\lambda-2\varepsilon$
C_{78}	$1+2\lambda+3\varepsilon$
C_{79}	$3+2\lambda+3\varepsilon$
C_{89}	$2+3\lambda-\varepsilon$

ПРИЛОЖЕНИЕ

Алгоритм поиска исходного базисного дерева

Для поиска исходного базисного дерева применяют описанный выше метод для следующей 'искусственной' задачи.

Строим граф $G=<E,V,H>$, в котором $E=E \cup \{i_0\}$, где i_0 - дополнительная вершина, $V=V \cup V1 \cup V2$, где

$V1$ - множество дополнительных дуг, направленных от вершин - пунктов производства к дополнительной вершине i_0,

$V2$ - множество дополнительных дуг, направляемых от i_0 к промежуточным вершинам и вершинам-пунктам потребления.

Для $v \in V$ полагаем $c(v)=0$;

для $v \in V1$ полагаем $c(v)=1$;

для $v \in V2$ полагаем $c(v)=0$.

Полагаем $b(i_0)=0$.

В качестве исходного базисного дерева берется подграф $\tilde{T}=<E, V1 \cup V2, H>$. Если в результате решения этой задачи оказалось, что оптимальное значение функционала строго больше нуля, то исходная задача

(1 -3) решения не имеет, в противном случае будет получено базисное дерево исходной задачи.

ЛАБОРАТОРНАЯ РАБОТА № 8

Транспортная задача в матричной постановке

Имеется n пунктов производства и m пунктов потребления некоторого однородного продукта. Заданы объемы производства $a[i]$, $i \in [1..n]$, в каждом пункте производства и размеры спроса $b[j]$, $j \in [1..m]$ в каждом пункте потребления. Известны также транспортные расходы $c[i,j]$, связанные с перевозом единицы продукта из пункта i в пункт j. Требуется определить объемы перевозок $x[i,j]$ из i-го пункта в j-ый для всех $i \in [1..n]$, $j \in [1..n]$, при минимальных общих транспортных издержках, при этом весь груз из пунктов производства должен быть вывезен, и весь спрос в пунктах потребления должен быть удовлетворен.

Дадим математическую постановку транспортной задачи:

$$L(x) = \Sigma \{ c[i,j] \times x[i,j] \mid i \in [1..n],\ j \in [1..n] \} \rightarrow min$$

при ограничениях

$$\Sigma \{ x[i,j] \mid j \in [1..m] \} = a[i] \text{ для всех } i \in [1..n], \quad (1)$$

$$\Sigma \{ x[i,j] \mid i \in [1..n] \} = b[j] \text{ для всех } j \in [1..m], \quad (2)$$

$$x[i,j] \geq 0, \quad \text{для всех } i \in [1..n],\ j \in [1..m]. \quad (3)$$

Транспортная задача имеет решение тогда и только тогда, когда выполняется соотношение :

$$\Sigma \{ a[i] \mid i \in [1..n] \} = \Sigma \{ b[j] \mid j \in [1..m] \}, \quad (4)$$

т.е. объем производства должен быть равен объему потребления. В реальных задачах условие (4) часто нарушается и имеет место соотношение

$$\Sigma \{ a[i] \mid i \in [1..n] \} \geq \Sigma \{ b[j] \mid j \in [1..m] \}, \quad (5)$$

В этом случае необходимо ввести в рассмотрение фиктивного (внешнего) потребителя $j1$ с

$$b[j1] = \Sigma \{ a[i] \mid i \in [1..n] \} - \Sigma \{ b[j] \mid j \in [1..m] \}.$$

Для всех $i \in [1..n]$ $c[i,j1]$ полагают равным достаточно большому числу. Аналогично, если

$$\Sigma \{ a[i] \mid i \in [1..n] \} \leq \Sigma \{ b[j] \mid j \in [1..m] \}, \quad (6)$$

то вводится дополнительный (внешний) пункт производства $i1$.

Транспортные задачи, в которых выполняется соотношение (4) называют замкнутыми, в случаях (5-6) открытыми.

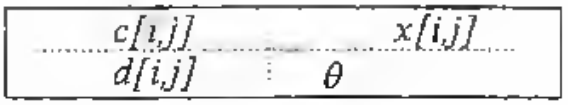
Транспортную задачу решают методом потенциалов, который является частным случаем метода потенциалов для транспортной задачи в сетевой постановке. Общая схема его такова:

1. Определяют исходное базисное решение.
2. По критерию оптимальности определяют, является ли найденное решение оптимальным.
3. Если критерий оптимальности не выполняется, то переходим к новому базису и возвращаемся к пункту 2, иначе переходим к пункту 4.
4. Конец алгоритма.

Опишем алгоритм решения транспортной задачи более детально. Запишем ее условия в таблицу следующего вида:

	j	*1*	...	*j*	...	*m*	
i	*b[j]* / *a[i]*	*b[1]*	...	*b[j]*	...	*b[m]*	*b[j]* / *u[j]*
1	*a[1]*	...	...	...	...	...	*u[1]*
2	*a[2]*	...	...	...	...	...	*u[2]*
...	...	...	...	...	...	...	...
i	*a[i]*	...	...	...	...	...	*u[i]*
...	...	...	...	...	...	...	...
n	*a[n]*	...	...	...	...	...	*u[n]*
	a[i] / *v[j]*	*v[1]*		*v[j]*		*v[m]*	*u[j]* / *v[j]*

Будем называть клеткой *[i,j]* клетку, находящуюся на пересечении *i*- той строки и *j*-го столбца. Каждую клетку *[i,j]* будем заполнять следующей информацией:



Описание позиций $d[i,j]$ и θ будет приведено ниже.

1. Определение исходного базисного решения .

Для определения исходного базиса можно использовать алгоритм северо-западного угла.

Алгоритм северо-западного угла

1. $k:=1;\ p:=1;$ (клетка $[k,p]$ находится в северо-западном углу рассматриваемой таблицы
2. $x[k,p]:=min\ (\ a[k],b[p]\ );$
   $a[k]:=a[k]-x[k,p];$
   $b[k]:=b[k]-x[k,p].$
3. Если $a[k]=0$, то вычеркиваем из таблицы k -ю строку, в противном случае (в этом случае $b[p]=0$) вычеркиваем p - ый столбец. Если одновременно $a[k] = b[k] = 0$, то вычеркиваем либо только столбец, либо только строку.
4. Если в таблице все столбцы и все строки вычеркнуты, то алгоритм заканчивает работу, в противном случае в качестве k и p берем значения северо-западного угла получившейся таблицы и переходим к выполнению 2.

Клетки с заполненной информацией о $x[i,j]$ являются базисными, все остальные внебазисные, для них $x[i,j]=0$ (в целях наглядности для внебазисных переменных значения $x[i,j]=0$ не заносятся). Среди базисных переменных возможно, что некоторые $x[i,j]=0$. В этом случае базис вырожденный.

Чтобы убедиться в том, что исходный базис найден верно, проверьте:

1. Количество базисных клеток (переменных) должно быть равно $n+m-1$.
2. Должны выполняться ограничения (1 - 3).

Для проверки найденного решения на оптимальность необходимо вычислить потенциалы всех пунктов и производства и потребления. Под потенциалами будем понимать числа $u[i]$, $i \in [1..n]$, и $v[j]$, определяемые по формуле:

$$v[j]=u[i]+c[i,j],$$

где клетка $[i,j]$ базисная клетка.

Алгоритм вычисления потенциалов

В данном алгоритме введен механизм пометок. Строка или столбец помечены знаком '+', если соответствующий потенциал найден, знаком '-' в противном случае.

В начале все строки и столбцы таблицы помечаем символом '-'

1 -**ый шаг**. *u[1]:=0* , *1*-ую строку помечаем символом '+'.

k - **ый шаг**.

а) Для каждой строки *i*, помеченной символом '+', выполнить:
для каждого $j \in [1..m]$ такого, что клетка *[i,j]* базисная и *j* - ый столбец помечен символом '-' *v[j] :=u[i]+c[i,j]*, столбец помечаем символом '+'.

b) Для каждого столбца *j* , помеченного символом '+', выполнить:
для каждой строки $i \in [1..n]$ такой, что клетка *[i,j]* базисная и *i* - ая строка помечена символом '-' *u[i]:=v[j]-c[i,j]*, строку помечаем символом '-'.

k - **шаг** выполняем до тех пор, пока все строки и столбцы не станут помеченными символами '+'.

Проверка критерия оптимальности

Для того, чтобы базисное решение $x[i,j], i \in [1..n], j \in [1..m]$ было оптимальным, необходимо и достаточно , чтобы для всех $i \in [1..n], j \in [1..m]$ было справедливо неравенство:

$$d[i,j]=v[j]-u[i]-c[i,j] \le 0.$$

Таким образом, для проверки найденного решения на оптимальность достаточно выполнить следующий алгоритм:

1. Для всех $i \in [1..n], j \in [1..m]$ положить:
$d[i,j]=v[j]-u[i]-c[i,j]$.
2. Найти такие *[k,p]*, что
$d[k,p] = max\{ d[i,j] \mid i \in [1..n], j \in [1..m] \}$.
3. Если $d[k,p] \le 0$, то полученное базисное решение является оптимальным и транспортная задача решена, иначе необходимо перейти к новому базису.

Алгоритм перехода к новому базису

Вводим в базис клетку *[k,p]*. Для определения клетки *[q,l]* выводимой из базиса, воспользуемся следующим алгоритмом:

1. Строим последовательность клеток
$П=\{[i1, j1], [i2, j2], \ldots [i,j], \ldots\}$ такую, что
а) $[k,p] \in П$;

b) все остальные клетки базисные;

c) последовательность $П$ образует цикл (смотри транспортную задачу в сетевой постановке).

2. Задаем направление обхода этого цикла, совпадающего с направлением $k \to p$. В клетку $[k,p]$ заносим символ '+'. Для остальных клеток $[i,j] \in П$, если направление $i \to j$ совпадает с направлением обхода цикла, то заносим в позицию θ символ '+', в противном случае заносим '-'.

3. Определяем клетку $[q,l]$, для которой $x[q,l]=min\ x[i,j]$, где $[i,j]$ - пробегает все клетки из $П$, помеченные символом '-'.

Полагаем $Q=x[q,l]$.Из базиса выводится клетка $[q,l]$.

4. Для всех $[i,j] \in П$ полагаем $x[i,j]=x[i,j]'*'Q$, где '*' - символ, которым помечена клетка $[i,j]$.

5. Перейти к выполнению проверки критерия оптимальности.

Замечание 1.

Иногда минимум достигается на нескольких клетках одновременно. В этом случае в качестве клетки $[q,l]$ выбирается произвольно только одна из них.

Замечание 2.

При заполнении таблицы для нового базиса значения $x[i,j]$ вносятся только в базисные клетки.

Пример.

Решить транспортную задачу методом потенциалов, значения $a[i]$, $b[j]$,$c[i,j]$ $(i \in [1..4], j \in [1..3])$, заданы в таблице 1.

Таблица 1

	j / Bj / Ai	1	2	3
I		1	7	3
1	2	1	2	2
2	4	2	3	1
3	1	1	4	3
4	5	3	5	1

Решение.

Транспортная задача имеет решение тогда и только тогда, когда выполняется соотношение :

$$\sum_{i=1}^{4} a[i] = \sum_{j=1}^{3} b[j]$$

В данном случае, $\sum_{i=1}^{4} a[i] > \sum_{j=1}^{3} b[j]$, поэтому необходимо ввести в рассмотрение фиктивного (внешнего) потребителя (№4) :

$b[4] = \sum_{i=1}^{4} a[i] - \sum_{j=1}^{3} b[j] = 12-11=1$

Положим для всех $i \in [1..4]$ $c[i,4]=100$

1. Определим исходное базисное решение методом северо-западного угла.

Таблица 7.

	j	1	2	3	4	
i	B_j / A_i	1	7	3	1	Ui
1		1 1	2 1	2	100	0
		0 -	0 +	-4	-3	
2		2	3 4	1	100	-1
		0	0	-2	-2	
3		1	4 1	3	100	-2
		2 +	0 -	-3	-1	
4		3	5 1	1 3	100 1	-3
		1	0	0	0	
	V_j	1	2	-2	97	U_i / V_j

Стоимость перевозок составит: $L(x) = 127$.

По критерию оптимальности определим, является ли найденное решение наилучшим. $d[3,1] = max\{ d[i,j] \mid i \in [1..4], j \in [1..4] \} = 2 \geq 0$, значит полученное базисное решение не является оптимальным и необходимо перейти к новому базису.

Строим цикл Π, который будет состоять из клеток $x[3,1]$, $x[1,1]$, $x[1,2]$, $x[3,2]$, в позициях θ которых записываются соответственно '+', '-', '+', '-'. Среди значений $x[i,j]$, в клетках которых записаны '-', находим минимальное $Q=min\ x[i,j]$. Оно равно 1, и достигается на клетке $[1,1]$. Т.о. клетку $[3,1]$ вводим в базис, а клетку $[1,1]$ выводим из базиса.

2. Пересчитаем значения $x[i,j]$ для нового базиса.

Таблица 3.

	J	*1*	*2*	*3*	*4*	
I	*Bj* / *Ai*	*1*	*6*	*3*	*2*	*Ui*
1	*2*	*1*	*2* *2*	*2*	*100*	*0*
		-3	*0*	*-4*	*-3*	
2	*4*	*2*	*3* *4*	*1*	*100*	*-1*
		-2	*0*	*-2*	*-2*	
3	*1*	*1* *1*	*4* *0*	*3*	*100*	*-2*
		0	*0*	*-3*	*-1*	
4	*5*	*3*	*5* *1*	*1* *3*	*100* *1*	*-3*
		-1	*0*	*0*	*0*	
	Vj	*-1*	*2*	*-2*	*97*	*Ui* / *Vi*

Стоимость перевозок составит: $L(x)=125$.

По критерию оптимальности определим, является ли найденное решение наилучшим. $max\{ d[i,j] |\ i \in [1..4],\ j \in [1..4] \} \le 0$, значит полученное базисное решение является оптимальным .

Стоимость перевозок без учета фиктивного потребителя составит L(x)=25.

Задания.

Решить транспортную задачу методом потенциалов, значения *a[i] (i∈[1..4])* и *b[j] (j∈[1..3])* заданы в таблице 4, значения *c[i,j]* задать произвольным образом.

Таблица 4

№ варианта	***A[I]***				***B[j]***		
	1	2	3	4	1	2	3
1	5	10	7	3	10	10	5
2	5	8	7	3	8	10	3
3	3	10	7	3	7	10	4
4	5	5	7	3	10	5	5
5	5	5	9	3	10	7	6
6	5	5	9	6	10	7	7
7	5	6	9	5	10	9	7
8	5	9	9	5	12	9	7
9	8	9	9	5	12	9	10
10	11	9	9	5	12	9	12
11	10	10	9	5	12	10	13

Продолжение табл.4

№ варианта	A[I]				B[j]		
12	14	10	9	5	12	13	13
13	14	10	9	7	14	13	13
14	14	10	9	8	14	14	13
15	6	10	9	9	14	8	14
16	8	10	7	9	12	8	14
17	8	6	7	9	12	8	10
18	8	6	7	6	12	8	7
19	8	6	12	6	12	13	7
20	6	6	7	6	12	8	7
21	8	10	7	6	10	8	7
22	8	13	7	6	16	11	7
23	8	13	9	6	16	11	9
24	8	15	9	6	16	13	9
25	12	15	9	4	16	17	9

ЛАБОРАТОРНАЯ РАБОТА № 9

Задача о максимальном потоке в сети

Пусть задан ориентированный граф $G=<E,V,H>$, в котором направление каждой дуги $v \in V$ означает направление движения потока (например, поток автомобилей), пропускная способность каждой дуги равна $d(v)$. На множестве вершин E выделены две вершины t и s. Вершина t является источником потока, s - стоком. Требуется определить максимальный поток, который может быть пропущен из вершины t в s

Обозначим через $x(v)$ величину потока, движущегося по дуге v. Очевидно, что

$$0 \le x(v) \le d(v) \ , \ v \in V \qquad (1)$$

В каждой вершине $i \in E \setminus \{t,s\}$ объем потока входящего равен объему потока выходящего, т.е. справедливо равенство

$$\Sigma\{x(v) | i \in V^{-}(i)\} = \Sigma\{x(v) | i \in V^{+}(i)\}$$

т.е.

$$\Sigma\{x(v) | i \in V^{-}(i)\} - \Sigma\{x(v) | i \in V^{+}(i)\} = 0. \qquad (2)$$

Для вершины t

$$\Sigma\{x(v) \mid i \in V^{+}(i)\} - \Sigma\{x(v) \mid i \in V^{-}(i)\} = -Q, \quad (3)$$

для вершины s

$$\Sigma\{x(v) \mid i \in V^{+}(i)\} - \Sigma\{x(v) \mid i \in V^{-}(i)\} = Q. \quad (4)$$

Величина Q является величиной потока, который выходит из вершины t и входит в вершину s.

Требуется определить

$$Q \to max \quad (5)$$

при ограничениях (1-4).

Величины Q, $x(v)$, $v \in V$, удовлетворяющие ограничениям (1-4) будем называть потоком в сети, и если они максимизируют величину Q, то максимальным потоком. Нетрудно видеть, что значения $Q=0$, $x(v)=0$ $v \in V$, являются потоком в сети.

Задача (1-5) является задачей линейного программирования, и ее можно решить алгоритмами симплекс-метода.

Разобьем множество вершин на две непересекающиеся части E1 и E2 таким образом, чтобы $t \in E1$, $s \in E2$. Разрезом $V(E1,E2)$, разделяющим t и s, будем называть множество $V(E1,E2) \subset V$ такое, что для каждой дуги $v \in V(E1,E2)$ либо $h1(v) \in E1$ и $h2(v) \in E2$, либо $h1(v) \in E2$ и $h2(v) \in E1$.

Разобьем множество $V(E1,E2)$ на две части $V(E1,E2,+)$, $V(E1,E2,-)$ следующим образом:

$$V(E1,E2,+) = \{v \in V(E1,E2) \mid h1(v) \in E1 \text{ и } h2(v) \in E2\}$$

$$V(E1,E2,-) = \{v \in V(E1,E2) \mid h2(v) \in E1 \text{ и } h1(v) \in E2\}$$

Пропускной способностью разреза будем называть величину

$$Q(E1,E2) = \Sigma\{d(v) \mid v \in V(E1,E2,+)\},$$

потоком через разрез величину

$$X(E1,E2) = \Sigma\{x(v) \mid v \in V(E1,E2,+)\} - \Sigma\{x(v) \mid v \in V(E1,E2,-)\}.$$

Справедлива следующая

Теорема 1. (О максимальном потоке и минимальном разрезе).

В любой сети величина максимального потока из источника t в сток s равна минимальной пропускной способности $Q(E1,E2)$ среди всех разрезов $V(E1,E2)$, разделяющих вершины t и s.

Заметим, что в максимальном потоке

$$x(v) = d(v), \; v \in V(E1,E2,+),$$
$$x(v) = 0, \; v \in V(E1,E2,-).$$

Пусть Q $x(v)$, $v \in V$, - некоторый поток в сети, последовательность

$$t = i(0), v(1), i(1), v(2), i(2), \ldots, v(k), i(k) = s,$$

является цепью, соединяющих вершины t и s. Зададим на этой цепи направление движения от вершины t к s. Дуга $v(j)$ из этой цепи называется прямой, если ее направление совпадает с направлением движения от t к s, и

обратной в противном случае. Эту цепь будем называть путем увеличения потока, если для прямых дуг v цепи $x(v) < d(v)$ и для обратных $x(v) > 0$. По этой цепи можно пропустить дополнительный поток q из t к s величиной $q = min\ (q1,q2)$, где $q1=min\ (d(v) - x(v))$, минимум берется по всем прямым дугам цепи, $q1=min\ (x(v))$, минимум берется по всем обратным дугам цепи.

Теорема 2.

Поток Q, $x(v)$, $v \in V$ максимальный тогда и только тогда, когда не существует пути увеличения потока.

Предлагаемый алгоритм решения задачи о максимальном потоке в сети основан на поиске пути увеличения потока из t в s, который в свою очередь основан на процессе расстановки пометок вершин. Будем говорить, что

- вершина i помечена пометкой $[g(i),+v(i)]$, если до нее дошел некоторый дополнительный поток величиной $q(i)>0$, а также известна прямая дуга $v(i)$, через которую поступил этот поток, либо помечена пометкой $[g(i),-v(i)]$, если до нее дошел некоторый дополнительный поток величиной $q(i)>0$, а также известна обратная дуга $v(i)$, через которую поступил этот поток;
- вершина i просмотрена, если помечены все соседние с ней вершины.

Если помечена вершина s, то найден путь увеличения потока величиной q, который пропускается по этому пути. Для описания алгоритма нам понадобится также массив SPW, в который помещаются номера помеченных вершин в порядке их пометки. $C1$ - номер в массиве SPW просматриваемой вершины, $C2$ - номер последней помеченной вершины в этом массиве.

Замечание.

Массив SPW можно организовать как список ссылок на помеченные вершины, как очередь или стек, то C1- ссылка на место просматриваемой вершины, C2 - место последней помеченной вершины.

Алгоритм 1.

```
begin
 for v∈V do x(v):=0; {обнуление начального потока}
 M1:
 g(t):=∞ {достаточно большое число};
 SPW(1):=t; C1:=1;C2:=1;{вершина t помечена и внесена в массив SPW}
 for i ∈ E\{t} do g(i):=0;{все вершины из E\{t} не помечены}
 M2:
 i:=SPW(C1); {i-номер просматриваемой вершины}
 for v ∈ V(i) do {пометка по прямым дугам}
```

```
begin j:=h2(v);
  if g(j)=0 then {т.е. вершина j не помечена}
  if (d(v) -x(v)) >0 then {по дуге можно пропустить дополнительный поток
                           в прямом направлении}.
  begin
    g(j):=min (g(i),d(v) -x(v));
    v(j):=v;
    C2:= C2+1; SPW(C2):=j;
    if j=s then {дополнительный поток дошел до вершины s}
    goto M3;
  end;
end;
for v ∈ V+(i) do {пометка по обратным дугам}
begin j:=h1(v);
  if g(j)=0 then {т.е. вершина j не помечена}
 if x(v) >0 then {по дуге можно пропустить дополнительный поток в
                  обратном направлении}.
  begin
    g(j):=min (g(i),x(v));
    v(j):=-v;
    C2:= C2+1; SPW(C2):=j;
    if j=s then {дополнительный поток дошел до вершины s}
    goto M3;
  end;
end;
if C1=C2 then stop; {все помеченные вершины просмотрены, но до вершины
                     s дополнительный поток не дошел, т.е. задача решена,
                     найден максимальный поток}
if C1≠C2 then
begin
  C1:=C1+1;
  goto M2; {переход на просмотр следующей помеченной вершины}
end;
M3:
begin {по найденному пути из t в s пропускается дополнительный поток}
  i=s;
  while i ≠ t do
  begin
    v:=abs (v(i));
    if v(i)>0 then
    begin
      x(v):=x(v)+g(s); {поток прошел по прямой дуге}
```

```
        i:=h1(v);
      end
      else
      begin
        x(v):=x(v)-g(s); {поток прошел по обратной дуге}
        i:=h2(s)
      end;
    end;
    goto M1; {переход на поиск нового пути увеличения потока}
  end;
end.
```

Для поиска разреза с минимальной пропускной способностью применим следующий алгоритм.

Алгоритм 2.

```
begin
  E1:=∅; E2:=∅;
  for i∈E do
  если g(i)>0 then {до вершины i дошел дополнительный поток}
  E1:=E1∪{i}
  else E2:=E2∪{i};
   {E(1) - множество вершин, к которым существует цепь увеличения потока
          из вершины i, E(2) - множество вершин, к которым такой цепи нет}
  V(E1,E2,+):=∅; V(E1,E2,-):=∅;
  for v∈V do
  begin
    if ((h1(v) ∈ E1 ) and (h2(v) ∈ E2 )) then
    V(E1,E2,+):=V(E1,E2,+) ∪ {v};
    if ((h1(v) ∈ E2) and (h2(v) ∈ E1 )) then
    V(E1,E2,-):=V(E1,E2,-) ∪ {v};
  end;
  V(E1,E2):=V(E1,E2,+) ∪V(E1,E2,-);
end.
```

Множество дуг $V(E1,E2)$ составляет разрез с минимальной пропускной способностью.

Пример.

На графе G, приведенном на рис.1., найти максимальный поток, который может быть пропущен из вершины 1 в вершину 9 и разрез с минимальной

пропускной способностью. В качестве пропускных способностей $d(v)$ возьмем значения из таблицы 1.

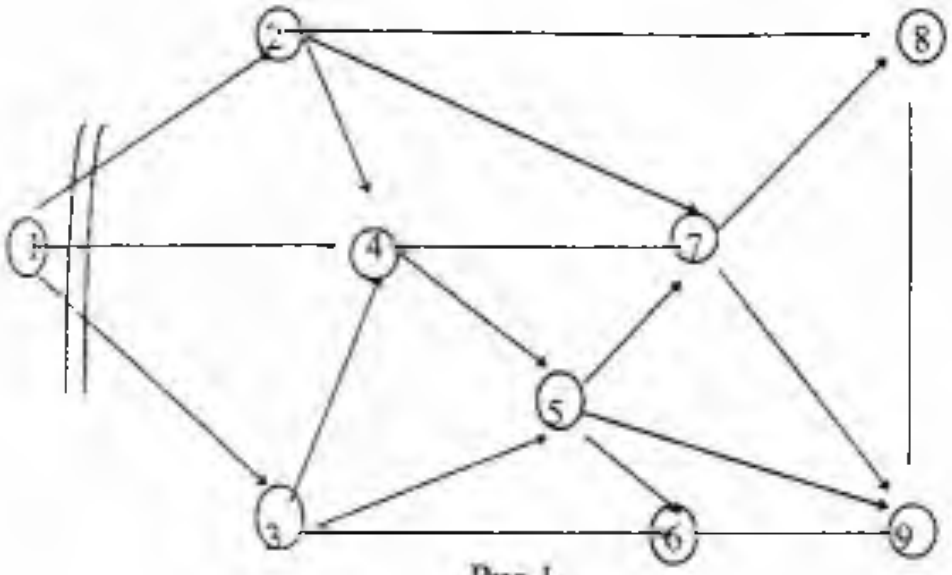


Рис.1.

Таблица 1

d_{12}	d_{14}	d_{13}	d_{28}	d_{27}	d_{24}	d_{34}	d_{35}	d_{36}	d_{47}	d_{45}	d_{57}	d_{58}	d_{59}	d_{67}	d_{68}	d_{79}	d_{89}
1	2	1	6	1	1	1	2	2	4	1	3	1	7	6	1	3	2

Решение.

Потоки по всем дугам полагаем равными нулю, т.е.

$X_{12}=0$; $X_{14}=0$; $X_{13}=0$; $X_{28}=0$; $X_{27}=0$; $X_{24}=0$;
$X_{34}=0$; $X_{35}=0$; $X_{36}=0$; $X_{47}=0$, $X_{45}=0$; $X_{57}=0$;
$X_{59}=0$; $X_{56}=0$; $X_{69}=0$; $X_{78}=0$; $X_{79}=0$; $X_{89}=0$;

Отыскание увеличивающего пути.

Помечаем вершину 1 пометками q(1)=1000 , v(1)= - Все остальные вершины не помечены, для них q(i)=0, v(i)= - .

Номер вершины i	q(i)	v(i)
1:	1000	

Из вершины 1 помечаем вершины 2, 3, 4. Покажем как это делается на примере вершины 2. Поток, поступивший в вершину 1 равен 1000, по дуге (1,2) $d_{12}=1$, $X_{12}=0$, поэтому дополнительный поток, который может поступить в

вершину 2 q(2)=min(q(1), d_{12}- X_{12})=min(1000,1-0)=1. Вершина 2 помечается по прямой дуге (1,2), поэтому v(i)=(1,2). После аналогичной пометки вершин 3. 4 получаем

Номер вершины i	q(i)	v(i)
1:	1000	-
2:	1	(1,2)
3:	1	(1,3)
4:	2	(1,4)

Затемнена просмотренная вершина, все остальные вершины не просмотрены. Из вершины 2 помечаем вершины 7 и 8

Номер вершины i	q(i)	v(i)
1:	1000	-
2:	1	(1,2)
3:	1	(1,3)
4:	2	(1,4)
7:	2	(4,7)
8:	1	(2,8)

Далее из вершины 3 помечаем 5 и 6

Номер вершины i	q(i)	v(i)
1:	1000	
2:	1	(1,2)
3:	1	(1,3)
4:	2	(1,4)
7:	2	(4,7)
8:	1	(2,8)
5:	1	(3,5)
6:	1	(3,6)

Аналогично вершина 4 .
После просмотра вершины 7 становится помеченной вершина 9.

Номер вершины i	q(i)	v(i)
1:	1000	-
2:	1	(1,2)
3:	1	(1,3)
4:	2	(1,4)
7:	2	(4,7)
8:	1	(2,8)
5:	1	(3,5)
6:	1	(3,6)
9:	2	(7,9)

Эта вершина является стоком, в нее пришел дополнительный поток q(9)=2. Восстанавливаем путь по которому пришел этот поток, используя значения v(i). Этот путь выглядит следующим образом

1, (1,4), 4, (4,7), 7, (7,9), 9.

По дугам этого пути увеличиваем потоки на величину q(9)=2. В результате получаем

$X_{12}=0$; $X_{14}=2$; $X_{13}=0$; $X_{28}=0$; $X_{27}=0$; $X_{24}=0$;
$X_{34}=0$; $X_{35}=0$; $X_{36}=0$; $X_{47}=2$, $X_{45}=0$; $X_{57}=0$;
$X_{59}=0$; $X_{56}=0$; $X_{69}=0$; $X_{78}=0$; $X_{79}=2$; $X_{89}=0$.

После аналогичного отыскания увеличивающих путей дважды получим

$X_{12}=1$; $X_{14}=2$; $X_{13}=1$; $X_{28}=0$; $X_{27}=0$; $X_{24}=1$;
$X_{34}=1$; $X_{35}=0$; $X_{36}=0$; $X_{47}=3$, $X_{45}=1$; $X_{57}=0$;
$X_{59}=0$; $X_{56}=1$; $X_{69}=1$; $X_{78}=0$; $X_{79}=3$; $X_{89}=0$.

Начинаем заново поиск увеличивающего пути из вершины t.
Помечаем вершину 1 пометками q(1)=1000 , v(1)= - . Все остальные вершины не помечены, для них q(i)=0, v(i)= - .

Из вершины 1 невозможно пометить по выходящим из нее дугам ни одной вершины

Номер вершины i	q(i)	v(i)
1:	1000	-

Все помеченные вершины просмотрены, поэтому максимальный поток найден. Его величина равна $Q=X_{69}+X_{79}+X_{89}=4$. Разрез с минимальной пропускной

способностью $V(E1,E2)=\{(1,2), (1,3),(1,4)\}$изображен на рис.1 двойной линией. $E1=\{1\}$. $E2=\{2,3,4,5,6,7,8,9\}$.

Задание.

На графе G, приведенном в лабораторной работе "Транспортная задача в сетевой постановке", найти максимальный поток, который может быть пропущен из вершины t в s и разрез с минимальной пропускной способностью, в качестве пропускных способностей $d(v)$ взять значения $C(v)$ из той же лабораторной работы.

ЛАБОРАТОРНАЯ РАБОТА № 10

Решение задачи о нахождении кратчайших расстояний на графе между двумя вершинами алгоритмами Беллмана-Калабы, Форда, Дейкстры

Пусть задан ориентированный граф $G=<E,V,H>$, в котором для каждой дуги $v \in V$ задана длина $c(v)$. На множестве вершин E выделены две вершины t и s. Требуется среди всех путей

$$t=i(0),v(1),i(1),v(2),i(2),...,v(k),i(k)=s,$$

соединяющих вершины t и s, где

$$h1(v(j))=i(j-1),\ h2(v(j))=i(j)\ ,\ j\in[1..k],$$

с длиной

$$l=\Sigma\{c(v(j))\mid j\in[1..k]\},$$

определить путь, длина которого минимальна .

Обозначим через $W(i)$ длину кратчайшего пути от вершины i до вершины s. Согласно принципа оптимальности Беллмана

$$W(s)=0,$$
$$W(i)=\min\{c(v)+W(h2(v))\mid v\in V^{-}(i)\},\ i\in E\setminus\{s\}. \qquad (1)$$

Значение $W(t)$ будет длиной кратчайшего пути от вершины t до вершины s. Для кратчайшего пути

$$t=i^{-}(0),v^{-}(1),i^{-}(1),v^{-}(2),i^{-}(2),...,v^{-}(k),i^{-}(k)=s,$$

справедливо равенство

$$W(i^{-}(j-1))=c(v^{-}(j))+W(i^{-}(j))=\min\{c(v)+W(h2(v))\mid v\in V^{-}(i^{-}(j-1))\},$$
$$j\in[1..k]. \qquad (2)$$

Алгоритм Беллмана-Калабы.

Первый этап.

0-шаг (задание начального приближения).

$W^0(s):=0$;

Для всех $i \in E\setminus\{s\}$ $W^0(i):=M$; (M - достаточно большое число, например большее, чем длина самого длинного пути).

j-ый шаг. ($j=1,2,3,\ldots$)

$W^j(s):=0$,

$$W^j(i) := \min \{ c(v) + W^{j-1}(h2(v)) \mid v \in V^-(i) \},\ i \in E\setminus\{s\}. \quad (4)$$

Если после выполнения двух, следующих друг за другом, итераций j и $(j+1)$

$$W^j(i)=W^{j+1}(i) \qquad \text{для всех } i \in E,$$

то полагаем

$$W(i)=W^{j+1}(i) \qquad \text{для всех } i \in E,$$

Если $W(t) \geq M$, то задача не имеет решения (нет путей между вершинами t и s), в противном случае переходим к выполнению второго этапа.

Второй этап.

0-шаг $i:=t$.

j-ый шаг ($j=1,2,3,\ldots$)

$$\underline{v}(j):=\arg\min \{ c(v) + W(h2(v)) \mid v \in V^-(\underline{i}(j-1)) \}, \quad (5)$$
$$\underline{i}(j):=h2(\underline{v}(j)).$$

Если $\underline{i}(j)=s$, то алгоритм заканчивает работу, в противном случае переходим к $(j+1)$ шагу.

Этот алгоритм можно модифицировать следующим образом. На первом этапе на каждом j-ом шаге при вычислении $W(i,j)$ по соотношению (4) полагаем

$$v(i)=\arg\min \{ c(v) + W(h2(v)) \mid v \in V^-(i) \},$$

тогда на втором этапе соотношение (5) можно заменить следующим соотношением

$$\underline{v}(j):=v(\underline{i}(j-1)).$$

Задание 1.

На графе G, приведенном в лабораторной работе "Транспортная задача в сетевой постановке", найти кратчайшее расстояние и путь между вершинами t и s, в качестве длин дуг взять значения C из той же лабораторной работы.

Результаты работы представить в виде следующей таблицы, а также выписать кратчайший путь и его длину.

Таблица 1

N шага						
i	0	1	2	3	4	5
	W(i,0)	*W(i,1)*	*W(i,2)*	*W(i,3)*	*W(i,4)*	*W(i,5)*
1=t						
2						
3						
4						
5						
6						
7						
8						
9=s						

Пример.

На графе *G*, приведенном в примере к лабораторной работе "Транспортная задача в сетевой постановке", найдем кратчайшее расстояние и путь между вершинами *t=1* и *s=9*, в качестве длин дуг возьмем значения *C* из того же примера, выпишем кратчайший путь и его длину.

N=1,

W(9,1)=0,

W(8,1)=min(2+W(9,0))=2,

W(7,1)=min(1+W(8,0),3+W(9,0))=3,

W(6,1)=min(6+W(9,0))=6,

W(5,1)=min(3+W(7,0),7+W(9,0),1+W(6,0))=7,

W(4,1)=min(4+W(7,0),1+W(5,0))=M,

W(3,1)=min(1+W(4,0),2+W(5,0),2+W(6,0))=M,

W(2,1)=min(6+W(8,0),1+W(7,0))=M,

W(1,1)=min(1+W(2,0),2+W(4,0),1+W(3,0))=M.

N шага										
i	0		1		2		3		4	
	W(i,0)	*V*	*W(i,1)*	*V*	*W(i,2)*	*V*	*W(i,3)*	*V*	*W(i,4)*	*V*
1=t	*M*		*M*	*2*	*M*	*2*	*5*	*2*	*5*	*2*
2	*M*		*M*	*8*	*4*	*7*	*4*	*7*	*4*	*7*
3	*M*		*M*	*4*	*8*	*4*	*8*	*4*	*8*	*4*
4	*M*		*M*	*7*	*7*	*7*	*7*	*7*	*7*	*7*
5	*M*		*7*	*9*	*6*	*7*	*6*	*7*	*6*	*7*
6	*M*		*6*	*9*	*6*	*9*	*6*	*9*	*6*	*9*
7	*M*		*3*	*9*	*3*	*8*	*3*	*8*	*3*	*8*
8	*M*		*2*	*9*	*2*	*9*	*2*	*9*	*2*	*9*
9=s	*0*		*0*		*0*		*0*		*0*	

Кратчайший путь : 1 → 2→7→8→9.
Длина кратчайшего пути составит:
l=1+1+1+2=5.

Алгоритм Форда для отыскания кратчайших расстояний на графе

Пусть $W(i)$ - верхние оценки кратчайших расстояний от вершины i до вершины s. Если для любой дуги $v \in V$ выполняется

$$W(h1(v)) \le c(v) + W(h2(v)),$$

то оценки $W(i)$ дают кратчайшие расстояния от вершины i до вершины s. Заметим, что если некоторая дуга $v \in V$ содержится в кратчайшем пути из некоторой вершины i (например из вершины $h1(v)$) в вершину s, то для нее

$$W(h1(v)) = c(v) + W(h2(v)).$$

Если существует дуга $v \in V$, для которой

$$W(h1(v)) > c(v) + W(h2(v)),$$

то оценки $W(i)$ не дают кратчайшие расстояния от вершин i до вершины s, так как оценку $W(h1(v))$ можно улучшить, положив

$$W(h1(v)) = c(v) + W(h2(v)).$$

На этом свойстве основан алгоритм Форда

Первый этап.

0-**шаг** (задание начального приближения).

$W(s):=0;$

Для всех $i \in E \setminus \{s\}$ $W(i):=M;$

j-**ый шаг**. (j=1,2,3,...)

Среди всех дуг $v \in V$ ищем дугу, для которой

$$W(h2(v)) < M \quad W(h1(v)) > c(v) + W(h2(v)).$$

Если такой дуги не существует, то переходим к выполнению второго этапа, в противном случае

$$W(h1(v)) := c(v) + W(h2(v))$$

и переходим к выполнению следующего шага.

Второй этап.

Если W(t)<M , то выполняем второй этап алгоритма Беллмана-Калабы. в противном случае задача решения не имеет.

Задание 2.

Решить предыдущую задачу алгоритмом Форда. Результаты представить в виде:

Таблица 2

i	Значения W(i)
t	
1	
2	
3	
4	
5	
6	
7	
8	
9	
s	

а также выписать кратчайший путь и его длину.
В каждой строке предыдущей таблицы выписывается последовательность значений W, получаемых на первом этапе.

Алгоритм Дейкстры отыскания кратчайших расстояний на графе

Алгоритм Дейкстры применяется для случая, когда $c(v)>0$ В нем каждая вершина может быть:

1. непомеченной,
2. помеченной временной пометкой,
3. помеченной постоянной пометкой.

Вершина i непомечена, если не найдено ни одного пути ведущего из вершины t в вершину i. Помечена временной пометкой, если из вершины t найден путь и величина $W(i)$ есть верхняя оценка кратчайшего расстояния от t до i, и на последующих итерация может быть уточнена вплоть до кратчайшего расстояния от t до i. Помечена постоянной пометкой, если $W(i)$ из верхней оценки стала кратчайшим расстоянием от t до i. Алгоритм первого этапа заканчивает работу, если $W(s)$ стало равным кратчайшему расстоянию от t до s.

Первый этап.

0-**шаг.** $W(t):=0$; вершину t считаем помеченной временной пометкой, для всех $i \in E \setminus \{t\}$ $W(i):=M$, вершины $i \in E \setminus \{t\}$ не помечены.

k-**ый шаг**. Среди всех временно помеченных вершин i ищем вершину j с наименьшим значением W. Вершину j помечаем постоянной пометкой. Если

$j=s$, то переходим к выполнению второго этапа, в противном случае просматриваем вершину j. Для этого для всех $v \in V^{-}(j)$ выполнить:

1.определяем i=h2(v),

2.если вершина i не помечена постоянной пометкой, то

$$W(i):=min(\ W(i), c(v)+W(j)\ ),$$

эта вершина помечается временной пометкой. Переходим к выполнению $(k+1)$ шага.

Второй этап. (восстановление кратчайшего пути)

0-**шаг.** $\underline{i}(0):=s;$

k-**ый шаг**. Среди $v \in V^{+}(\underline{i}(k-1))$ ищем $\underline{v}(k-1)$, для которой

$W(h1(\underline{v}(k)))=c(\underline{v}(k))+W(\underline{i}(k-1)),$

$\underline{i}(k):=h1(\underline{v}(k)).$

Если $\underline{i}(k)=t$, то алгоритм заканчивает работу, в противном случае переходим к выполнению $(k+1)$-го шага.

В результате выполнения второго этапа получим

$$t=i(k),\ \underline{v}(k-1),\underline{i}(k-1),\underline{v}(k-2),\ldots,\underline{v}(0),\underline{i}(0)=s,$$

являющееся решением задачи.

Задание 3

Решить предыдущую задачу алгоритмом Дейкстры. Результаты представить в виде таблицы 2 и выписать кратчайший путь.

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. **Моисеев Н.Н**. Математические задачи системного анализа . М.: Наука, 1988.

2. **Гермейер Ю.Б**. Введение в теорию исследования операций. М.: Наука, 1971.

3. **Зайченко Ю.П**. Исследование операций. Киев: Віща школа,1975.

4. **Коваленко А.Г**. Элементы выпуклого векторного программирования: Учебное пособие. Самара, 1990, С.83.

5. **Коваленко А.Г., Власова И.А. Цветков Ю.Д.** Лабораторный практикум по методам оптимизации. Куйбышев: СамГУ, 1991. С.59.

6. **Подиновский В.В., Ногин В.Д.** Парето-оптимальные решения. М.: Наука, 1971.

7. **Дубов Ю.А**. и др. Многокритериальные модели формирования и выбора вариантов систем. М.: Наука, 1986.

8. **Романовский И.В**. Алгоритмы решения экстремальных задач. М.: Наука, 1977.

9. **Коваленко А.Г**. Алгоритмы решения некоторых задач оптимизации многошаговых процессов. Куйбышев, 1985. С.72.

10. **Тетерев А.Г.** Динамическое программирование в курсе исследования операций. Куйбышев, 1983.

11. Исследование операций / Под ред. Дж.Моудера, С.Элмаграби. М.: Мир. 1981. Т.1,2.

12. **Корбут А.А**., Финкельштейн Ю.Ю. Дискретное программирование. М.: Наука, 1971.

13. **Райфа Г**. Анализ решений. М.: Мир, 1977.

14. **Хедли Дж**. Нелинейное и динамическое программирование. М.: Мир, 1967.

15. **Ху Т**. Целочисленное программирование и потоки в сетях. М.: Мир, 1971.

16. **Танаев В.С., Шкурба В.В**. Введение в теорию расписаний. М.: Наука, 1975.

17. **Беллман Р., Дрейфус С**. Прикладные задачи динамического программирования. М.: Мир, 1965.

Приложение 1. **Инструкция по применению программы Gauss.exe.**

Программа Gauss.exe предназначена для решения всевозможных задач, в которых необходимо выполнять преобразование матриц методом полного исключения Гаусса - Жордана. К числу таких задач относятся задачи :

- решение систем линейных уравнений,
- вычисление обратных матриц,
- решения задач линейного программирования прямым, модифицированным, двойственным симплекс-методом,
- решения задач квадратичного программирования,
- решения задач дробно- линейного программирования,
- решения задач целочисленного линейного программирования (как методами отсечения Данцига, Гомори, так и методом ветвей и границ),
- решение задач нелинейного программирования методом секущих гиперплоскостей
- и т.д.

Программа достаточно проста в обращении, неприхотлива к типу и мощности ПЭВМ (любой IBM - совместимый компьютер). Разработана под операционную систему DOS. При решении систем линейных уравнений размерность матрицы задачи 100×100.

Для начала работы запускается программа Gauss.exe, появляется заставка, от которой избавляемся нажатием на любую клавишу. Программа запрашивает о наличии исходных данных. Если их нет, то отвечаете 'n', в противном случае 'y' Если данных нет, то запрашивается размерность матрицы исходных данных. Не расстраивайтесь, если ошиблись - число строк и столбцов можно добавить до требуемого количества. Если у вас данные есть, то будет запрошено имя файла с данными. Здесь желательно не ошибиться, так как в противном случае произойдет сбой программы. Данные у вас могут образоваться в случае, если вы раньше решали какую-то задачу и записали ее в файл. Файл с данными размещается в той же директории, что программа gauss.exe.

После выполнения всех этих операций перед вами появляется таблица (точнее симплекс-таблица), которая либо заполнена нулями, если у вас данных не было, либо данными, считанными из указанного вами файла. Одна из клеток выделена. Если вы начнете нажимать на какие-нибудь цифры, то число, ранее стоявшее в этой клетке, пропадет и на его месте будет набираться новое число. По окончании набора нажмите 'Enter'. Нажимая на стрелки клавиатуры, можно выделить любую клетку таблицы и занести в нее требуемую информацию.

Ниже таблицы выписаны подсказки для ваших действий.

Для расчетов по формулам полного исключения Гаусса - Жордана подведите выделенное окно на разрешающий элемент и нажмите "пробел" это основная операция, выполняемая программой. В этом случае все элементы строки таблицы (разрешающей строки) с выделенным элементом делятся на выделенное число (на выделенном месте получается 1), все остальные строки (включая строку d) преобразуются таким образом, чтобы в разрешающем столбце (столбце, где стоит разрешающий элемент) получились нули. Для этого разрешающая строка умножается на число, стоящее на пересечении разрешающего столбца и преобразуемой строки, и вычитается из преобразуемой строки.

Для введения дополнительного столбца нажмите "w" - в таблице добавляется новый столбец. Эта операция применяется, например, для введения дополнительных переменных при приведении задачи линейного программирования к каноническому виду, или для введения искусственных переменных при решении систем линейных уравнений и неравенств, частным случаем которых является поиск исходного базиса в задаче линейного программирования. Если все столбцы не помещаются на экран, то к ним можно переместиться, нажимая на стрелки.

Для введения дополнительного столбца и строки нажмите "v" - вводятся в таблицу дополнительные строка и столбец. Эта операция применяется при присоединении к задаче линейного программирования дополнительных неравенств, которые так же приводятся к каноническому виду. Такие приемы применяются при решении задач целочисленного линейного программирования (как методами отсечения Данцига, Гомори, так и методом ветвей и границ), решение задач нелинейного программирования методом секущих гиперплоскостей,

Для записи данных с экрана в файл - "s"- данные из таблицы помещаются в файл, имя которого указывается по запросу. В последствии эти данные вы заново можете внести в таблицу. Эта операция часто применяется при решении задач целочисленного линейного программирования методом ветвей и границ.

Для записи таблицы с экрана в файл - "S" - таблица, в том виде, как она выглядит на экране, помещается в файл **fff.fff**. В последствии эти таблицы не могут быть вызваны на экран ПЭВМ, но их можно распечатать на бумажный носитель в любом текстовом редакторе.

Для перехода к новым задачам - "n" - в этом случае в программу вводятся данные таким же образом, как и при начале работы с программой. Эта операция часто применяется при решении задач целочисленного линейного программирования методом ветвей и границ.

Для окончания работы Esc - конец работы с программой.

Рассмотрим несколько примеров.

Пример 1

Решить систему уравнений

$$2x1 + x2 = 2$$
$$x1 + 2x2 = 2$$

После запуска программы Gauss.exe задать число строк равное 2 и число столбцов также равное 2. Появится таблица

нс	b	x1	x2
1	0	0	0
2	0	0	0
d	0	0	0

Заносим в нее данные. В столбец b свободные члены, в столбец *x1* коэффициенты при *x1*, в столбец *x2* коэффициенты при *x2*. Строку d оставляем без изменения. Получим таблицу

нс	b	x1	x2
1	2	2	1
2	2	1	2
d	0	0	0

Подводим стрелками выделение на пересечение первой строки и столбца *x1* и нажимаем клавишу "пробел" (в предыдущей таблице оно уже выделено). На экране появится преобразованная таблица

нс	b	x1	x2
1	1	1	0.50
2	1	0	1.50
d	0	0	0

Подводим выделение на пересечение второй строки и столбца *x2* и нажимаем клавишу "пробел" (в предыдущей таблице оно уже выделено).

нс	b	x1	x2
1	0.67	1	0
2	0.67	0	1
d	0	0	0

Для получения ответа в столбце *x1* находим 1, на пересечении строки, содержащей эту единицу и столбца b получаем ответ *x1*=0.67. Аналогично *x2*=0.67. Данные, записанные в этой таблице, эквивалентны системе

$$x1 + 0\,x2 = 0.67$$
$$0\,x1 + x2 = 0.67$$

Пример 2

Найти обратную матрицу для матрицы

$$\begin{pmatrix} 4 & 1 \\ 1 & 2 \end{pmatrix}$$

После запуска программы Gauss.exe задать число строк, равное 2 и число столбцов, также равное 2. Появится таблица, которую заполним элементами этой матрицы

нс	b	x1	x2
1	0	4	1
2	0	1	2
d	0	0	0

Добавим два столбца, в которые занесем единичную матрицу, получим

нс	b	x1	x2	x3	x4
1	0	4	1	1	0
2	0	1	2	0	1
d	0	0	0	0	0

Разрешающими элементами сделаем последовательно пересечение столбца x1 и строки 1,затем пересечение столбца x2 и строки 2 (на этих местах нажмем пробелом), получим

нс	b	x1	x2	x3	x4
1	0	1	0	0.29	-0.14
2	0	0	1	-0.14	0.57
d	0	0	0	0	0

На месте, где была исходная матрица, получилась единичная, на месте единичной обратная. Попробуйте на месте обратной вновь получить единичную, т.е. разрешающими элементами сделайте последовательно пересечение столбца x3 и строки 1, затем пересечение столбца x4 и строки 2.

СОДЕРЖАНИЕ