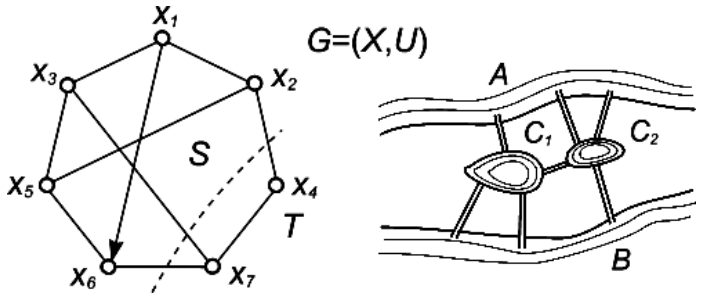
ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

А.М. ГОСТИН, В.П. КОРЯЧКО

ДИСКРЕТНАЯ МАТЕМАТИКА. ТЕОРИЯ ГРАФОВ



Рязань 2006

ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ
РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

А.М. ГОСТИН, В.П. КОРЯЧКО

ДИСКРЕТНАЯ МАТЕМАТИКА. ТЕОРИЯ ГРАФОВ

Учебное пособие

Допущено Учебно-методическим объединением вузов по университетскому политехническому образованию в качестве учебного пособия для студентов высших учебных заведений, обучающихся по направлению 230100 «Информатика и вычислительная техника», специальности 230104 «Системы автоматизированного проектирования»

Рязань 2006

УДК 519.17

Дискретная математика. Теория графов: Учеб. пособие / А.М. Гостин, В.П. Корячко. Рязан. гос. радиотехн. ун-т, 2006. 80 с.

ISBN 5-7722-0252-9

Содержит материалы учебного курса «Дискретная математика», включающие описание алгоритмов прикладных задач теории графов.

Предназначено для студентов, изучающих курс «Дискретная математика».

Табл. 14. Ил. 56. Библиогр.: 7 назв.

Теория графов, дискретная математика

Печатается по решению редакционно-издательского совета Рязанского государственного радиотехнического университета.

Рецензент: кафедра САПР ВС Рязанского государственного радиотехнического университета (зам. зав. кафедрой, д-р техн. наук, профессор С.В. Скворцов)

Гостин Алексей Михайлович

Корячко Вячеслав Петрович

Дискретная математика. Теория графов

Редактор Н.А. Орлова

Корректор С.В. Макушина

Подписано в печать 29.12.06. Формат бумаги 60 × 84 1/16.

Бумага газетная. Печать трафаретная. Усл. печ. 5,0.

Уч.-изд. 5,0. Тираж 200 экз. Заказ

Рязанский государственный радиотехнический университет.

390005, Рязань, ул. Гагарина 59/1.

Редакционно-издательский центр РГРТУ.

ISBN 5-7722-0252-9

ОГЛАВЛЕНИЕ

1. СПОСОБЫ ЗАДАНИЯ ГРАФОВ

1.1. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

Граф G задается множеством точек или вершин $X=\{x_1, x_2, ..., x_n\}$ и множеством линий $U=\{u_1, u_2, ..., u_m\}$, соединяющих между собой все или часть этих точек. Таким образом, граф G полностью задается (и обозначается) парой $G=(X, U)$.

Если ребра из множества U ориентированы, что обычно показывается стрелкой $\vec{U}$, то они называются *дугами,* и граф с такими ребрами называется *ориентированным графом,* который обозначается $\vec{G} = (X,\vec{U})$ (рис. 1.1). Если ребра не имеют ориентации, то такой граф называется *неориентированным*: $G=(X, U)$ (рис. 1.2).

Когда дуга обозначается упорядоченной парой, состоящей из начальной и конечной вершин (т. е. двумя концевыми вершинами дуги), ее направление предполагается заданным от первой вершины ко второй. Так, например, на рис. 1.1 обозначение (x_1, x_2) относится к дуге $\vec{u}_1$, а (x_2, x_1) - к дуге $\vec{u}_2$.

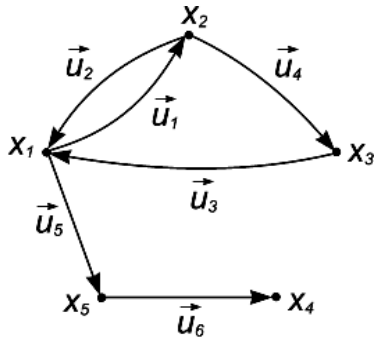


Рис. 1.1

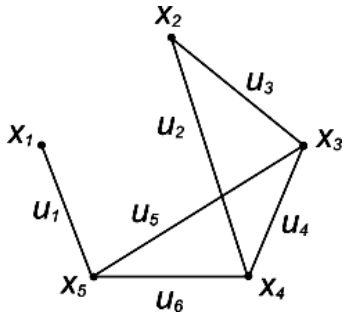


Рис. 1.2

Другое употребляемое описание ориентированного графа $\vec{G}$ состоит в задании множества вершин X и *соответствия* Γ, которое показывает, как между собой связаны вершины. Соответствие Γ называется *отображением* множества X на себя (*транзитивным замыканием*), а граф в этом случае обозначается парой $\vec{G} = (X,\Gamma)$ [1].

Соответствие (*образ*) одной вершины обозначается $\Gamma(x_i)$ и представляет собой множество вершин, соединенных дугами с вершиной x_i. Например, для графа G на рис. 1.1 имеем $\Gamma(x_1)=\{x_2,x_5\}$, т. е. вершины x_2

и x_5 являются конечными вершинами дуг, у которых начальной вершиной является x_1. Для остальных вершин графа справедливо:

$\Gamma(x_2)=\{x_1,x_3\}$, $\Gamma(x_3)=\{x_1\}$, $\Gamma(x_4)=\varnothing$ - пустое множество, $\Gamma(x_5)=\{x_4\}$.

Неориентированный граф эквивалентен ориентированному графу, который получается из исходного графа путем замены каждого неориентированного ребра двумя противоположно направленными дугами, соединяющими те же самые вершины. Так, например, для графа, приведенного на рис. 1.2, имеем $\Gamma(x_5)=\{x_1,x_3,x_4\}$, $\Gamma(x_1)=\{x_5\}$ и т. д.

Если соответствие $\Gamma(x_i)$ представляет собой множество таких вершин $x_i \in X$, для которых в графе G существует дуга $u(x_i,x_j)$, то $\Gamma^{-1}(x_i)$ представляет множество вершин x_k, для которых в графе G существует дуга $u(x_k,x_i)$. Отношение $\Gamma^{-1}(x_i)$ называется *обратным соответствием* или *прообразом* вершины x_i.

Для графа, изображенного на рис. 1.1, имеем $\Gamma^{-1}(x_1)=\{x_2,x_3\}$, $\Gamma^{-1}(x_2)=\{x_1\}$ и т. д.

Вполне очевидно, что для неориентированного графа $\Gamma^{-1}(x_i)=\Gamma(x_i)$ для всех $x_i \in X$.

Когда отображение Γ действует не на одну вершину, а на множество вершин $X_q=\{x_1,x_2,\ldots,x_q\}$, то под $\Gamma(X_q)$ понимают объединение отображений

$\Gamma(x_1)\cup\Gamma(x_2)\cup\ldots\cup\Gamma(x_q)$,

т. е. $\Gamma(X_q)$ является множеством таких вершин $x_i \in X$, что для каждой из них существует дуга $u(x_i,x_j)$ в G, где $x_i \in X$. Для графа, приведенного на рис. 1.1, $\Gamma(\{x_2,x_5\})=\{x_1,x_3,x_4\}$ и $\Gamma(\{x_1,x_3\})=\{x_2,x_5,x_1\}$.

Отображение $\Gamma(\Gamma(x_i))$ записывается как $\Gamma^2(x_i)$. Аналогично "тройное" отображение $\Gamma(\Gamma(\Gamma(x_i)))$ записывается как $\Gamma^3(x_i)$ и т. д.

Для графа, показанного на рис. 1.1, имеем:

$\Gamma^2(x_1)=\Gamma(\Gamma(x_1))=\Gamma(\{x_2,x_5\})=\{x_1,x_3,x_4\}$,
$\Gamma^3(x_1)=\Gamma(\Gamma^2(x_1))=\Gamma(\{x_1,x_3,x_4\})=\{x_2,x_5,x_1\}$ и т. д.

Аналогично понимаются обозначения $\Gamma^{-2}(x_i)$, $\Gamma^{-3}(x_i)$ и т. д.

Две вершины x_i и x_j называются *смежными*, если какая-нибудь из двух дуг $\vec{u}(x_i,x_j)$ и $\vec{u}(x_j,x_i)$ или обе одновременно присутствуют в графе.

Так, например, на рис. 1.3 вершины x_1 и x_3 являются смежными, в то время как вершины x_1 и x_4 не являются смежными.

Дуги $\vec{u}_i$, $\vec{u}_j$, имеющие общие концевые вершины, называются *инцидентными*. Например, на рис. 1.3 дуги $\vec{u}_1$ и $\vec{u}_2$ являются инцидентными, а дуги $\vec{u}_1$ и $\vec{u}_6$ – нет.

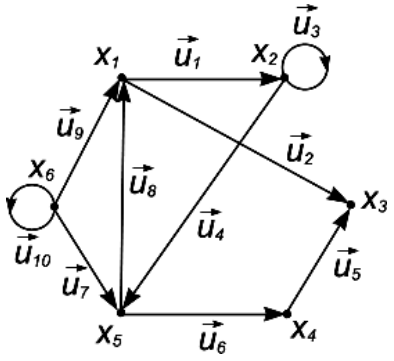


Рис. 1.3

В ориентированном графе число дуг, которые имеют вершину x_i своей начальной вершиной, называется *полустепенью исхода* вершины $d^+(x_i)$ и аналогично число дуг, которые имеют x_i своей конечной вершиной, называется *полустепенью захода* вершины $d^-(x_i)$.

На рис. 1.3 полустепень исхода вершины $d^+(x_6)$ равна $|\Gamma(x_6)|=2$ и полустепень захода вершины $d^-(x_6)$ равна $|\Gamma^{-1}(x_6)|=1$.

Очевидно, что сумма полустепеней захода всех вершин графа, а также сумма полустепеней исхода всех вершин равны общему числу дуг графа G, т. е.

$$\sum_{i=1}^{n} d^+(x_i) = \sum_{i=1}^{n} d^-(x_i) = m,$$

где n - число вершин, а m - число дуг графа G.

Для неориентированного графа $G=(X,\Gamma)$ степень вершины x_i определяется с помощью соотношения $d(x_i)=|\Gamma(x_i)|$.

Петлей называется дуга, начальная и конечная вершины которой совпадают. На рис. 1.3, например, дуги $\vec{u}_3$ и $\vec{u}_{10}$ являются петлями.

Граф $G=(X, U)$ называют *полным*, если для любой пары вершин $x_i, x_j \in X$ существует ребро $u(x_i, x_j) \in U$, т. е. для каждой пары вершин графа G должна существовать, по крайней мере, одна дуга, соединяющая их.

Полный неориентированный граф, построенный на n вершинах, обозначается через K_n. Например, граф, изображенный на рис. 1.4, является полным графом K_4.

Граф $G=(X, U)$ называют *пустым*, если он не содержит ни одной дуги, т.е. $U = \varnothing$, $|U| = 0$ (рис. 1.5).

Мультиграфом называют граф, в котором существует хотя бы одна пара вершин x_i, $x_j \in X$, связанных более чем одним ребром (рис. 1.6).

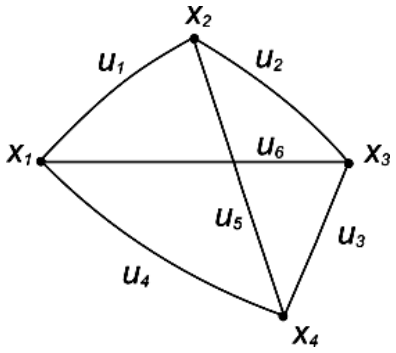


Рис. 1.4

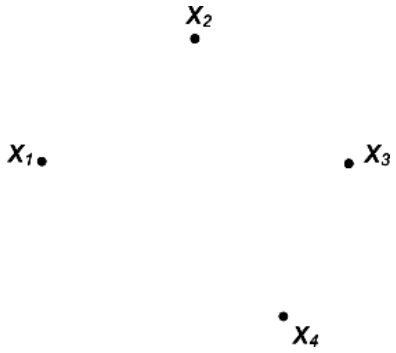


Рис. 1.5

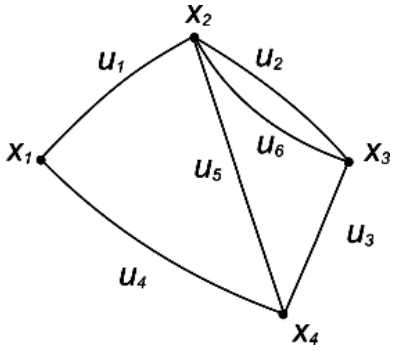


Рис. 1.6

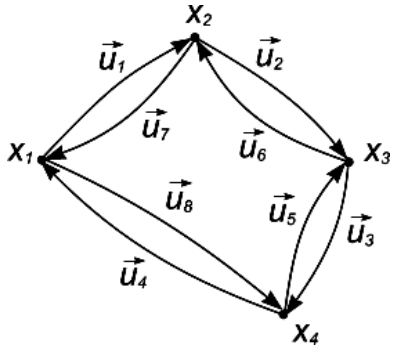


Рис. 1.7

Граф $\vec{G} = (X, \vec{U})$ называется *симметрическим*, если для любой дуги $\vec{u}(x_i, x_j) \in \vec{U}$ существует также противоположно ориентированная дуга $\vec{u}(x_j, x_i) \in \vec{U}$ (рис. 1.7).

Подграфом G_S графа $G=(X,U)$ называется граф (X_S, U_S), образуемый из некоторого подмножества вершин и дуг графа G, т.е. $X_S \subset X$ и $U_S \subset U$.

Остовым подграфом G_p графа $G=(X,U)$ называется граф (X, U_p), для которого $U_p \subset U$. Таким образом, остовый подграф имеет то же самое множество вершин, что и граф G, но множество дуг подграфа G_p является подмножеством множества дуг исходного графа.

На рис. 1.8 показан подграф графа K_4, изображенного на рис. 1.4, а на рис. 1.9 показан его остовой подграф.

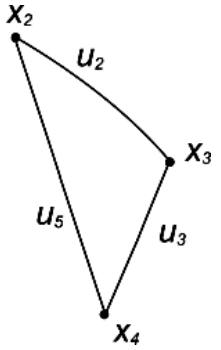


Рис. 1.8

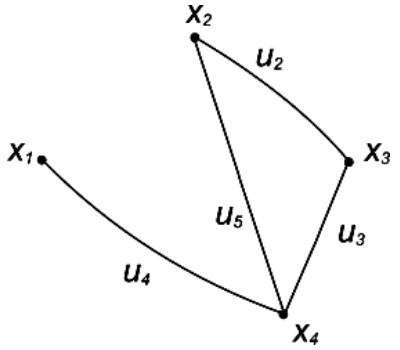


Рис. 1.9

Если с дугами графа G сопоставляются (приписываются) числа - дуге $u(x_i,x_j)$ ставится в соответствие некоторое число c_k, называемое *весом* (*длиной*, или *стоимостью*) дуги. Такой граф $G=(X,U,W)$ называется *взвешенным* (рис. 1.10). В этом случае W представляет собой вектор весов $W=||c_k||_m$, где m – число дуг графа. Если веса c_i приписываются вершинам x_i графа, то получается граф $G=(X,U,V)$ *с взвешенными вершинами* (рис. 1.11), а V представляет собой вектор весов $V=||c_i||_n$, где n – число вершин графа.

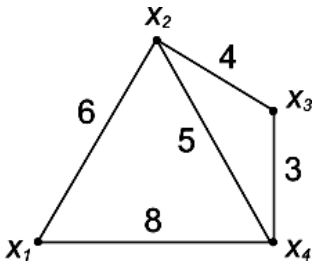


Рис. 1.10

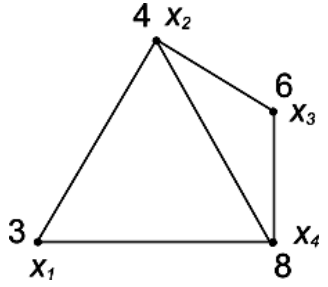


Рис. 1.11

1.2. МАТРИЧНЫЕ СПОСОБЫ ПРЕДСТАВЛЕНИЯ ГРАФОВ

В большинстве алгоритмов решения прикладных задач графы представляются с помощью матриц. *Матрица смежности* используется для представления ориентированных, неориентированных и взвешенных графов (с положительными и отрицательными весами дуг). Мультиграфы можно представить только с помощью *матрицы инцидентности*. Ориентированные графы с отрицательными весами дуг представлять с помощью матрицы инцидентности нельзя.

1.2.1. МАТРИЦА СМЕЖНОСТИ

Пусть дан граф $\vec{G} = (X, \vec{U})$, его *матрица смежности* обозначается через $S=||s_{ij}||_{n\times n}$, где n – число вершин графа, и определяется следующим образом:

s_{ij}=1, если в G существует дуга $\vec{u}(x_i, x_j)$;

s_{ij}=0, если в G нет дуги $\vec{u}(x_i, x_j)$.

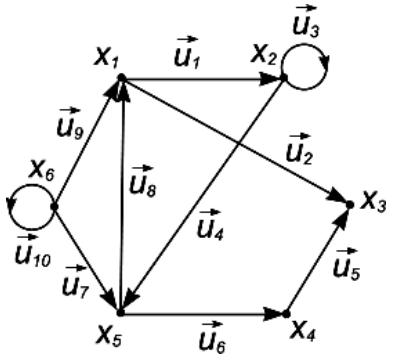


Рис. 1.12

Таким образом, матрица смежности графа, изображенного на рис. 1.12, имеет вид:

S=	x_1	x_2	x_3	x_4	x_5	x_6
x_1	0	1	1	0	0	0
x_2	0	1	0	0	1	0
x_3	0	0	0	0	0	0
x_4	0	0	1	0	0	0
x_5	1	0	0	1	0	0
x_6	1	0	0	0	1	1

Матрица смежности полностью определяет структуру графа. Например, сумма всех элементов строки x_i матрицы дает полустепень исхода вершины x_i, а сумма элементов столбца x_i - полустепень захода вершины x_i. Множество индексов ненулевых элементов строки x_i есть соответствие $\Gamma(x_i)$, а множество индексов ненулевых элементов столбца x_i представляет обратное соответствие $\Gamma^{-1}(x_i)$.

Петли на графе представляют собой ненулевые элементы главной диагонали матрицы, например s_{22} и s_{66} соответствуют дугам $\vec{u}_3$, $\vec{u}_{10}$ для графа, изображенного на рис. 1.12.

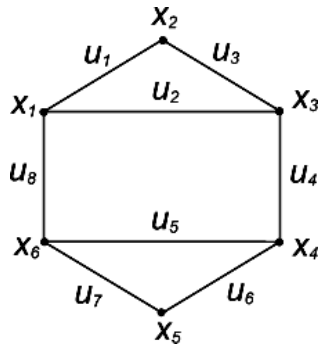


Рис. 1.13

В случае неориентированного графа матрица смежности является симметричной относительно главной диагонали (рис. 1.13).

S=	x_1	x_2	x_3	x_4	x_5	x_6
x_1	0	1	1	0	0	1
x_2	1	0	1	0	0	0
x_3	1	1	0	1	0	0
x_4	0	0	1	0	1	1
x_5	0	0	0	1	0	1
x_6	1	0	0	1	1	0

Если граф взвешенный, элементы матрицы смежности будут определяться как:

s_{ij}= c_{ij}, если в G существует дуга $\vec{u}(x_i, x_j)$, а c_{ij} – ее вес;

s_{ij}=0, если в G нет дуги $\vec{u}(x_i, x_j)$.

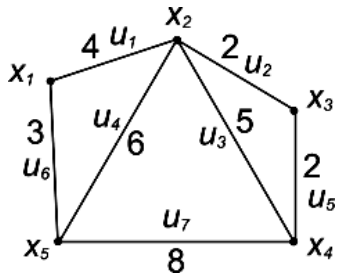


Рис. 1.14

Таким образом, матрица смежности графа, изображенного на рис. 1.14, будет иметь следующий вид:

S=	x_1	x_2	x_3	x_4	x_5
x_1	0	4	0	0	3
x_2	4	0	2	5	6
x_3	0	2	0	2	0
x_4	0	5	2	0	8
x_5	3	6	0	8	0

1.2.2. МАТРИЦА ИНЦИДЕНТНОСТИ

Пусть дан граф $\vec{G} = (X, \vec{U})$ с n вершинами и m дугами. *Матрица инцидентности* графа G обозначается через $B=||b_{ij}||_{n\times m}$ и определяется следующим образом:

b_{ij}=1, если x_i является начальной вершиной дуги $\vec{u}_j$;

b_{ij}=-1, если x_i является конечной вершиной дуги $\vec{u}_j$;

b_{ij}=0, если x_i не является концевой вершиной дуги $\vec{u}_j$.

Для графа, приведенного на рис. 1.12, матрица инцидентности имеет вид:

B=	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8	u_9	u_{10}
x_1	1	1	0	0	0	0	0	-1	-1	0
x_2	-1	0	±1	1	0	0	0	0	0	0
x_3	0	-1	0	0	0	0	0	0	0	0
x_4	0	0	0	0	-1	-1	0	0	0	0
x_5	0	0	0	-1	1	1	-1	1	0	0
x_6	0	0	0	0	0	0	1	0	1	±1

Поскольку каждая дуга инцидентна двум различным вершинам (за исключением случая, когда дуга образует петлю), то каждый столбец содержит один элемент, равный 1, и один, равный -1. Петля в матрице инцидентности не имеет адекватного математического представления (в программной реализации допустимо задание одного элемента b_{ij}=1).

Если G является неориентированным графом, то его матрица инцидентности определяется следующим образом:

b_{ij}=1, если x_i является концевой вершиной дуги $\vec{u}_j$;

b_{ij}=0, если x_i не является концевой вершиной дуги $\vec{u}_j$.

Для графа, приведенного на рис. 1.13, матрица инцидентности имеет вид:

B=	u_1	u_2	u_3	u_4	u_5	u_6	u_7	u_8
x_1	1	1	0	0	0	0	0	1
x_2	1	0	1	0	0	0	0	0
x_3	0	1	1	1	0	0	0	0
x_4	0	0	0	1	1	1	0	0
x_5	0	0	0	0	0	1	1	0
x_6	0	0	0	0	1	0	1	1

Матрица инцидентности *взвешенного* неориентированного графа будет содержать веса дуг:

b_{ij}= c_j, если x_i является концевой вершиной дуги $\vec{u}_j$, где c_j – вес дуги;

b_{ij}=0, если x_i не является концевой вершиной дуги $\vec{u}_j$.

Для графа, приведенного на рис. 1.14, матрица инцидентности имеет вид:

B=	u_1	u_2	u_3	u_4	u_5	u_6	u_7
x_1	4	0	0	0	0	3	0
x_2	4	2	5	6	0	0	0
x_3	0	2	0	0	2	0	0
x_4	0	0	5	0	2	0	8
x_5	0	0	0	6	0	3	8
x_6	0	0	0	0	0	0	0

Взвешенный ориентированный граф будет определяться с помощью матрицы инцидентности следующим образом:

b_{ij}= c_j, если x_i является начальной вершиной дуги $\vec{u}_j$, где c_j – вес дуги;

b_{ij}= - c_j, если x_i является конечной вершиной дуги $\vec{u}_j$, где c_j – вес дуги;

b_{ij}= 0, если x_i не является концевой вершиной дуги $\vec{u}_j$.

1.3. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Перечислите основные способы представления графов.

2. Покажите на примере прямое и обратное соответствия для заданной вершины.

3. Чему равна сумма степеней всех вершин неориентированного графа?

4. В чем отличия матричного представления ориентированных и неориентированных графов?

5. В чем особенности представления графа матрицей смежности?

6. В чем особенности представления графа матрицей инцидентности?

7. По заданному преподавателем изображению графа построить матрицы смежности и инцидентности.

8. По заданной преподавателем матрице смежности (инцидентности) изобразить граф.

2. ПОИСК КРАТЧАЙШИХ ПУТЕЙ

2.1. ПУТИ И МАРШРУТЫ

Путем (или *ориентированным маршрутом*) ориентированного графа $\vec{G} = (X, \vec{U})$ называется последовательность дуг, в которой конечная вершина всякой дуги, отличной от последней, является начальной вершиной следующей.

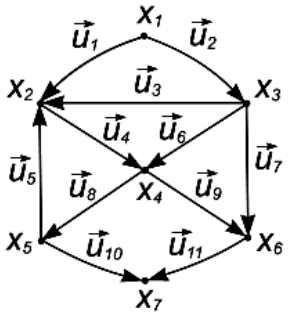


Рис. 2.1

Так, на рис. 2.1 последовательности дуг

$\mu_1 = \{\vec{u}_6, \vec{u}_8, \vec{u}_5, \vec{u}_4, \vec{u}_9\}$,

$\mu_2 = \{\vec{u}_2, \vec{u}_6, \vec{u}_8, \vec{u}_{10}\}$,

$\mu_3 = \{\vec{u}_1, \vec{u}_4, \vec{u}_8, \vec{u}_5, \vec{u}_4, \vec{u}_9, \vec{u}_{11}\}$

являются путями.

Ориентированной цепью (или, короче, *орцепью*) называется такой путь, в котором каждая дуга используется не больше одного раза. Так, например, приведенные выше пути μ_1 и μ_2 являются орцепями, а путь μ_3 не является таковым, поскольку дуга $\vec{u}_4$ в нем используется дважды.

Простой орцепью называется такой путь, в котором каждая вершина используется не более одного раза. Например, путь μ_2 является простой орцепью, а пути μ_1 и μ_3 - нет.

Маршрутом называется путь в неориентированном графе. Таким образом, маршрут есть последовательность ребер (u_1, u_2 ,..., u_q), в которой каждое ребро u_i связано с ребрами u_{i-1} и u_{i+1} своими двумя концевыми вершинами ($1<i<q$).

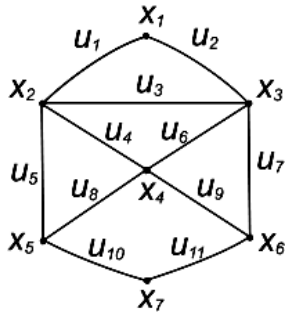


Рис. 2.2

Последовательности ребер

μ_4={u_2, u_6, u_8, u_{10}},

μ_5={u_6, u_4, u_5, u_8, u_9},

μ_6={u_1, u_4, u_8, u_5, u_4, u_9}

в графе, изображенном на рис. 2.2, являются маршрутами.

Точно так же, как мы определили орцепи и простые орцепи, можно определить цепи и простые цепи. Так, например, маршрут μ_4 есть простая цепь, маршрут μ_5 - цепь, но не простая, а маршрут μ_6 не является цепью.

Путь или маршрут можно представить также последовательностью вершин, например, μ_1={x_3, x_4, x_5, x_2, x_4, x_6}, и такое представление часто оказывается более полезным в тех случаях, когда осуществляется поиск простых орцепей или простых цепей.

2.2. ВЕС И ДЛИНА ПУТИ

При рассмотрении пути μ, представленного последовательностью дуг (u_1, u_2 ,..., u_q), за его *вес* (*длину*, *стоимость*) принимается число $l(\mu)$, равное сумме весов всех дуг, входящих в путь μ, т. е.

$$l(\mu) = \sum_{(x_i, x_j) \in \mu} c(x_i, x_j). \qquad (2.1)$$

Характеристики "длина", "стоимость", "цена" и "вес" имеют эквивалентный смысл применительно к дугам. Выбор того или иного наименования может быть произвольным и осуществляться в зависимости от контекста задачи.

Если граф не взвешенный, то *длиной* (или *мощностью*) пути называется количество дуг, входящих в него:

$$l(\mu) = \sum_{(x_i, x_j) \in \mu} u(x_i, x_j). \qquad (2.2)$$

2.3. ЗАДАЧА О КРАТЧАЙШЕМ ПУТИ

Пусть дан взвешенный граф $G=(X,U,W)$, где $W=||c_k||_m$ – множество весов дуг. *Задача о кратчайшем пути* состоит в нахождении кратчайшего пути от заданной начальной вершины $s \in X$ до заданной конечной вершины $t \in X$ при условии, что такой путь существует, т. е. при условии $t \in R(s)$. Здесь $R(s)$ – множество вершин, достижимых из вершины s. Единственное ограничение состоит в том, чтобы в графе $G=(X,U)$ не было циклов с суммарным отрицательным весом. Если такой цикл F все же существует и x_i - некоторая его вершина, то, двигаясь от s к x_i, обходя затем F достаточно большое число раз и попадая наконец в t, мы получаем путь со сколь угодно малым ($\to -\infty$) весом. Таким образом, в этом случае кратчайшего пути не существует. Отсюда также вытекает, что неориентированные дуги графа G не могут иметь отрицательные веса.

Нужно отметить, что почти все методы, позволяющие решить задачу о кратчайшем $\mu(s, t)$-пути, вычисляют в процессе решения и кратчайшие пути ко всем вершинам s к x_i ($s, x_i \in X$). Таким образом, они позволяют решить задачу с небольшими дополнительными вычислительными затратами.

Если в графе G дуга $u(x_i, x_j)$ отсутствует, то ее вес полагается равным ∞ (бесконечности).

Наиболее эффективный алгоритм решения задачи о кратчайшем $\mu(s, t)$-пути первоначально дал Дейкстра. В общем случае этот метод основан на приписывании вершинам временных пометок, причем пометка вершины дает верхнюю границу длины пути от s к этой вершине. Эти пометки (их величины) постепенно уменьшаются с помощью некоторой итерационной процедуры, и на каждом шаге итерации одна из временных пометок становится постоянной. Последнее указывает на то, что пометка уже не является верхней границей, а дает точную длину кратчайшего пути от s к рассматриваемой вершине.

Использование алгоритма Дейкстры для поиска кратчайших путей позволяет уменьшить количество вычислений. Число операций при использовании алгоритма Дейкстры в общем случае приближается к $O(n^3)$, где n – количество вершин графа.

2.4. АЛГОРИТМ ДЕЙКСТРЫ

Пусть $l(x_i)$ – пометка вершины x_i.

Шаг 1. Присвоение начальных значений. Положить $l^*(s)=0$ и считать эту пометку постоянной. Положить $l(x_i)=\infty$ (бесконечность) для всех $x_i \neq s$ и считать эти пометки временными. Положить $p=s$.

Шаг 2. Обновление пометок. Для всех вершин $x_i \in \Gamma(p)$, имеющих временные пометки, изменить пометки в соответствии с выражением:

$$l(x_i) = \min\{l(x_i), l(p) + c(p, x_i)\}. \quad (2.3)$$

Шаг 3. Превращение пометки в постоянную. Среди всех вершин с временными пометками найти такую, для которой $l^*(x_i) = \min\{l(x_i)\}$.

Шаг 4. Считать пометку вершины x_i^* постоянной и положить $p = x_i^*$.

Шаг 5. Если $p=t$, то $l^*(p)$ является длиной кратчайшего пути. Конец. Если $p \neq t$, перейти к шагу 2.

Как только длина кратчайшего пути от s до t будет найдена [она будет постоянной пометкой вершины $l^*(t)$], сами пути можно получить с помощью рекурсивной процедуры. Так, для вершины x_k, непосредственно предшествующей вершине t в кратчайшем пути от s к t, будет соблюдаться отношение: $l^*(x_k) + c(x_k, t) = l^*(t)$.

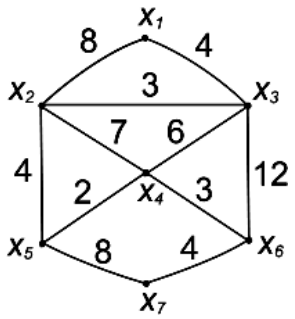
Таким образом, для любой вершины x_i можно найти предшествующую вершину x_k, принадлежащую пути $\mu(s, t)$, для которой справедливо:

$$l^*(x_k) + c(x_k, x_i) = l^*(x_i). \quad (2.4)$$

Если существуют несколько кратчайших путей от s к t, то данное соотношение будет выполняться для нескольких вершин. В этом случае выбор пути может быть произвольным.

2.5. ПРИМЕР

Рассмотрим неориентированный взвешенный граф $G=(X,U,W)$, изображенный на рис. 2.3. Матрица смежности S данного графа приведена ниже. Требуется найти кратчайший путь от вершины x_1 к вершине x_7 графа. Для решения задачи воспользуемся алгоритмом Дейкстры. Постоянные пометки будем снабжать знаком *, остальные пометки будут рассматриваться как временные.



$S=$

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
x_1	∞	**8**	**4**	∞	∞	∞	∞
x_2	**8**	∞	**3**	**7**	**4**	∞	∞
x_3	**4**	**3**	∞	**6**	∞	**12**	∞
x_4	∞	**7**	**6**	∞	**2**	**3**	∞
x_5	∞	**4**	∞	**2**	∞	∞	**8**
x_6	∞	∞	**12**	**3**	∞	∞	**4**
x_7	∞	∞	∞	∞	**8**	**4**	∞

Рис. 2.3

Шаг 1. $l^*(x_1)=0^*$; $\forall\, x_i \neq x_1$: $l(x_i)=\infty$; $p=x_1$.

Первая итерация

Шаг 2. $\Gamma(p)=\Gamma(x_1)=\{x_2, x_3\}$ - все пометки временные. По формуле (2.3) получаем оценки для смежных вершин:

$l(x_2) = \min\{\infty, 0^*+8\} = 8$; $l(x_3) = \min\{\infty, 0^*+4\} = 4$.

Шаг 3. Определяем постоянную пометку:

$l^*(x_3) = \min\{l(x_2), l(x_3)\} = \min\{8, 4\} = 4$.

Шаг 4. Вершина x_3^* получает постоянную пометку, $p=x_3^*$.

Шаг 5. Не все вершины имеют постоянные пометки, поэтому переходим к шагу 2.

Вторая итерация

Шаг 2. $\Gamma(p)=\Gamma(x_3)=\{x_2, x_4, x_6\}$ – все пометки временные. Из (2.3) получаем $l(x_2) = \min\{8, 4^*+3\} = 7$, аналогично $l(x_4) = \min\{\infty, 4^*+6\} = 10$, $l(x_6) = \min\{\infty, 4^*+12\} = 16$.

Шаг 3. $l^*(x_2) = \min\{l(x_2), l(x_4), l(x_6)\} = \min\{7,10,16\} = 7$.

Шаг 4. Вершина x_2^* получает постоянную пометку, $p=x_2^*$.

Шаг 5. Переход к шагу 2.

Для наглядности сведем полученные пометки в табл. 2.1. Продолжая этот процесс, получаем окончательную картину расстановки пометок, изображенную на рис. 2.4.

Таблица 2.1

№ п/п	p	$\Gamma(p)$	Оценка длины пути $l(x_i)$						
			$l(x_1)$	$l(x_2)$	$l(x_3)$	$l(x_4)$	$l(x_5)$	$l(x_6)$	$l(x_7)$
1	x_1	–	0*	∞	∞	∞	∞	∞	∞
2	x_1	x_2, x_3	0*	8	4*	∞	∞	∞	∞
3	x_3	x_2, x_4, x_6	0*	7*	4*	10	∞	16	∞
4	x_2	x_4, x_5	0*	7*	4*	10*	11	16	∞
5	x_4	x_5, x_6	0*	7*	4*	10*	11*	13	∞
6	x_5	x_7	0*	7*	4*	10*	11*	13*	19
7	x_6	x_7	0*	7*	4*	10*	11*	13*	17*

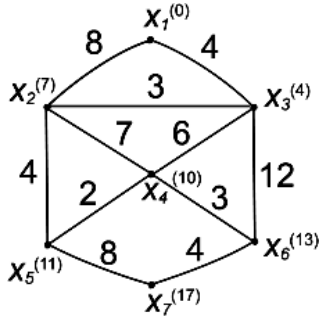


Рис. 2.4

Для нахождения кратчайшего пути между вершинами x_1 и x_7 нужно последовательно использовать соотношение (2.4). Таким образом, полагая $x_i=x_7$, находим вершину x_k, непосредственно предшествующую x_7 в кратчайшем пути от x_1 к x_7. Вершина x_k является прообразом вершины x_7: $x_k \in \Gamma^{-1}(x_7)$, и их оценки должны удовлетворять соотношению (2.4):

$$l^*(x_k) + c(x_k, x_7) = l^*(x_7) = 17^*.$$

Единственной вершиной, удовлетворяющей этому условию, является вершина x_6 $(13^*+4 = 17^*)$.

Далее, беря $x_i=x_6$ и снова применяя (2.4), получаем вершину x_4, непосредственно предшествующую x_6 в кратчайшем пути от x_1 к x_6: $l^*(x_4)+c(x_4, x_6) = 10^*+3 = 13^*$.

Для вершины x_4 единственной вершиной, удовлетворяющей (2.4), является x_3 и т.д.

В результате получается кратчайший путь $\mu(x_1, x_7)=\{x_1, x_3, x_4, x_6, x_7\}$, длина которого минимальна и равна $l(\mu)=17$.

2.6. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определение пути, маршрута, цепи.
2. Как определяется длина пути не взвешенного графа?
3. Формулировка задачи нахождения кратчайшего пути на графе.
4. Ограничения применимости алгоритма Дейкстры для нахождения кратчайшего пути на графе.
5. Как определяются постоянные пометки в алгоритме Дейкстры и что они выражают?
6. По заданному преподавателем графу найти с помощью алгоритма Дейкстры кратчайший путь между двумя вершинами.

3. ЗАДАЧА КОММИВОЯЖЕРА

3.1. ЦИКЛЫ НА ГРАФАХ

Маршрут (u_1, u_2,..., u_q) называется *циклом*, если в нем начальная вершина дуги u_1 совпадает с конечной вершиной дуги u_q.

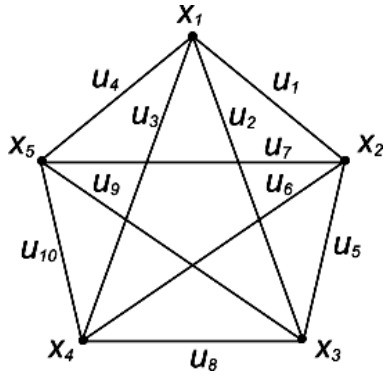


Рис. 3.1

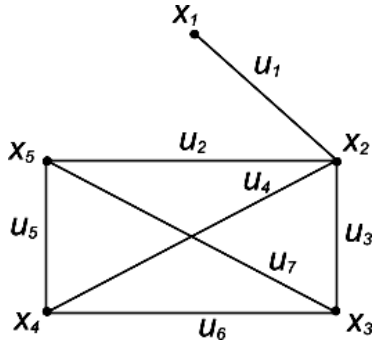


Рис. 3.2

Так, например, в графе, приведенном на рис. 3.1, последовательности дуг:

$\mu_1=\{u_1, u_5, u_2\}$, $\mu_2=\{u_1, u_5, u_2, u_3, u_{10}, u_4\}$, $\mu_3=\{u_1, u_5, u_8, u_{10}, u_4\}$, $\mu_4=\{u_1, u_5, u_8, u_{10}, u_4, u_2, u_9, u_7, u_6, u_3\}$ – являются циклами.

Цикл называется *эйлеровым*, если он содержит все ребра графа ровно один раз. Свое наименование цикл получил за знаменитую задачу о Кенигсбергских мостах, решенную великим математиком Л. Эйлером в 1736 году и положившую начало теории графов.

Теорема (Эйлера). Граф содержит эйлеров цикл только тогда, когда все его вершины имеют четную степень.

Например, граф, изображенный на рис. 3.1, будет содержать Эйлеров цикл (μ_4), а граф, изображенный на рис. 3.2, – нет.

Цикл называется *простым*, если каждая вершина, принадлежащая циклу, используется только один раз (за исключением начальной и конечной вершин, которые совпадают).

Например, пути μ_1 и μ_3 являются простыми циклами, а путь μ_2 не является простым циклом, так как вершина x_1 используется в нем дважды.

Простой цикл, проходящий через все вершины графа, имеет особое значение и называется *гамильтоновым* циклом. Конечно, не все графы обладают гамильтоновыми циклами. Так, например, путь μ_3 является гамильтоновым циклом графа, приведенного на рис. 3.1, а граф на рис. 3.2 не имеет гамильтонова цикла. В отличие от эйлерового цикла для гамильтонова цикла неизвестен простой критерий его существования в произвольном графе.

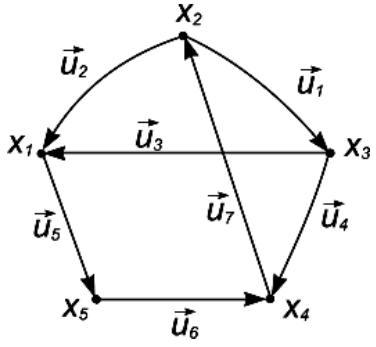


Рис. 3.3

Контуром называется цикл в ориентированном графе. Таким образом, контур является путем (x_1, x_2,..., x_q), в котором совпадают начальная и конечная вершины, т. е. в котором $x_1 = x_q$.

На рис. 3.3 маршруты

$\mu_4 = \{\vec{u}_1, \vec{u}_4, \vec{u}_7\}$,

$\mu_5 = \{\vec{u}_2, \vec{u}_5, \vec{u}_6, \vec{u}_7\}$ и

$\mu_6 = \{\vec{u}_1, \vec{u}_3, \vec{u}_5, \vec{u}_6, \vec{u}_7\}$

являются контурами.

Цикломатическим числом графа $\nu(G)$ называется число независимых простых циклов, содержащихся в графе. Оно вычисляется по формуле:

$$\nu(G) = m - n + 1. \qquad (3.1)$$

Например, количество независимых контуров электрической цепи определяется цикломатическим числом.

3.2. ПОСТАНОВКА ЗАДАЧИ КОММИВОЯЖЕРА

Задача коммивояжера, имеющая более чем 200-летнюю историю и которой занимались многие знаменитые ученые, начиная с Л.Эйлера и В.Гамильтона и кончая выдающимися математиками современности Д.Данцигом и Р.Белманом, имеет следующую формулировку. Имеется n городов, расстояния между которыми известны. Коммивояжер должен посетить все n городов по одному разу, вернувшись в тот, с которого начал. Требуется найти такой маршрут движения, при котором суммарное пройденное расстояние будет минимальным.

Очевидно, что задача коммивояжера – это задача отыскания кратчайшего гамильтонова цикла в полном графе или кратчайшего гамильтонова контура в ориентированном графе.

Можно предложить следующую простую схему решения задачи коммивояжера: сгенерировать все $n!$ возможных перестановок вершин полного графа, подсчитать для каждой перестановки длину маршрута и выбрать из них кратчайший. Очевидно, такое вычисление потребует $O(n!)$ шагов. Таким образом, решение задачи коммивояжера описанным методом полного перебора относится к числу NP-полных задач и даже для сравнительно небольших n оказывается практически неосуществимым.

Сформулируем математическую модель задачи коммивояжера. Пусть имеется n городов и задана матрица $C = ||c_{ij}||_{n\times n}$ расстояний между городами. Очевидно, что $c_{ii} = 0$, и, возможно, что $c_{ij} \neq c_{ji}$.

Введем переменные

$$x_{ij} = \begin{cases} 1, \text{ если дуга (i, j) включена в маршрут;} \\ 0 - \text{в противном случае,} \end{cases}$$

где $i, j = 1,2,\ldots, n$.

Задача поиска гамильтонова контура минимальной длины сводится к определению таких x_{ij}, обеспечивающих минимум целевой функции

$$F(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij} x_{ij} \rightarrow min \qquad (3.2)$$

при следующих ограничениях:

$$\sum_{j=1}^{n} x_{ij} = 1, \qquad i = \overline{1,n}, \qquad (3.3)$$

и

$$\sum_{i=1}^{n} x_{ij} = 1, \qquad j = \overline{1,n}. \qquad (3.4)$$

Ограничения (3.3) обеспечивают требования въезда в каждый j-й город ровно 1 раз (вхождения в j-ю вершину графа ровно одной дуги). Ограничения (3.4) обеспечивают требования выезда из каждого i-го города ровно 1 раз (выхода из i-й вершины графа ровно одной дуги). Задача (3.2) – (3.4) представляет собой задачу дискретного программирования с псевдобулевыми переменными. Одним из эффективных методов решения этой задачи является метод ветвей и границ.

3.3. МЕТОД ВЕТВЕЙ И ГРАНИЦ

Впервые идея *метода ветвей и границ* была предложена в 1960 г. Лэндом и Дойгом применительно к решению задач целочисленного линейного программирования. Однако эта работа не оказала заметного влияния на развитие техники решения задач дискретного программирования. Второе рождение метода ветвей и границ связано с работой Литтла, Мурти, Суини и Кэрел, опубликованной в 1963 г.

Основная идея метода довольно проста. Вначале строится некоторая *оценка снизу* ξ_0 длины маршрута для множества всех гамильтоновых контуров G_0. После этого множество всех гамильтоновых маршрутов разбивается на два непересекающихся подмножества. Первое подмножество состоит из гамильтоновых контуров, включающих некоторую дугу $u(x_i, x_j)$, а другое подмножество состоит из контуров, не включающих эту дугу. Обозначим первое подмножество как $G_1=[i, j]$, второе – $G_2=[\overline{i, j}]$. Очевидно, что $G_1 \cup G_2 = G_0$, $G_1 \cap G_2 = \varnothing$.

Другими словами, ветвление основано на следующем элементарном соображении: переезд из любого города i в j может либо принадлежать оптимальному циклу, либо не принадлежать ему.

Нижняя граница множества гамильтоновых контуров характеризует наименьшую неизбежную оценку маршрута. Для каждого из подмножеств по тому же правилу, что и для первоначального множества гамильтоновых

маршрутов, определяется нижняя граница ξ_i. Каждая новая нижняя граница оказывается не меньше нижней границы, определенной для всего множества $\xi_i \geq \xi_0$. Сравнивая нижние границы, можно отделить подмножество гамильтоновых путей, внутри которого с большей вероятностью содержится оптимальный маршрут. Это подмножество по аналогичному правилу разбивают еще на два, вновь находят измененные нижние границы и так поступают до тех пор, пока не останется единственный контур.

Получив некоторый гамильтонов контур, просматривают другие листья дерева и, если нижние границы подмножеств, соответствующие листьям, окажутся меньше, чем длина найденного контура, продолжают ветвление по тому же правилу, пока не получат маршрут с меньшей длиной или не убедятся, что среди этих подмножеств не может содержаться подобного маршрута. Те же ветви, для которых нижняя оценка окажется больше длины полученного маршрута, исключают из рассмотрения.

Основное достоинство метода состоит в указании способа вычисления нижней границы для соответствующих подмножеств и в указании дуги $u(x_i, x_j)$, включение в маршрут или исключение которой разбивает изучаемое множество гамильтоновых путей на подмножества. Остановимся вкратце на этих двух вопросах.

Идея получения оценок связана с тем, что если решить задачу коммивояжера с некоторой матрицей расстояний, а затем из какой-нибудь строки или столбца этой матрицы вычесть произвольное положительное число, то решение задачи коммивояжера с этой измененной матрицей расстояний совпадет с прежним решением, а длина маршрута изменится на это же самое число. Если эту операцию проделать и для других строк и столбцов, то длина маршрута будет отличаться на сумму всех чисел, вычитаемых из строк и столбцов.

Опираясь на эту теорему (доказанную Егервари), можно осуществить так называемое приведение матрицы, состоящее в следующем. В каждой строке матрицы находят минимальный элемент h_i и вычитают его из всех элементов этой строки. В результате по крайней мере один из элементов строки будет равен нулю. Так поступают со всеми строками. Полученная матрица называется *приведенной по строкам*.

После этого аналогичным образом осуществляется приведение полученной матрицы по столбцам, в результате чего находятся минимальные элементы столбцов h_j.

Приведенная по строкам и столбцам матрица содержит, по крайней мере, один нуль в каждой строке и каждом столбце. Так как длина L_1

оптимального маршрута в задаче с приведенной матрицей отличается от длины L маршрута в задаче с неприведенной матрицей на сумму констант приведения $h = \sum_n h_i + \sum_n h_j$, то

$$L = L_1 + h. \qquad (3.5)$$

В приведенной матрице все элементы неотрицательны, поэтому $L_1 \geq 0$, а сумма констант приведения h может служить нижней границей длины гамильтонова маршрута ξ_0.

Если какая-либо дуга $u(x_i, x_j)$ запрещается, то соответствующий элемент c_{ij} в исходной (а следовательно, и в приведенной) матрице расстояний заменяется на ∞. В результате такой замены появляется возможность провести дополнительное приведение матрицы и улучшение оценки.

Априорное включение дуги $u(x_i, x_j)$ в маршрут автоматически ведет к сокращению размеров матрицы. Так как коммивояжер выходит из x_i города только один раз, то i-я строка не будет содержать других дуг, входящих в маршрут, следовательно, она не подлежит дальнейшему рассмотрению и вычеркивается из матрицы. Аналогично, коммивояжер может войти в x_j город только один раз, поэтому из матрицы вычеркивается и j-й столбец.

Поскольку включение этой дуги представляет важный элемент алгоритма, то на нем следует остановиться подробнее.

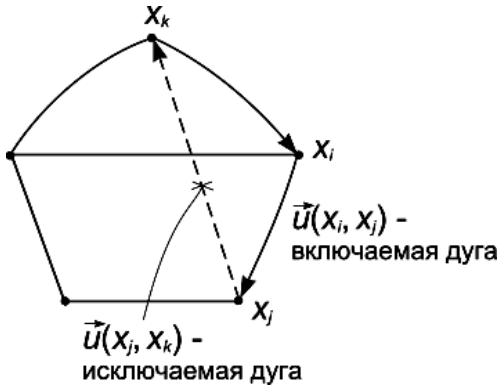


Рис. 3.4

Включение дуги $u(x_i, x_j)$ на очередной итерации приводит к образованию связного пути, соединяющего некоторые вершины x_k и x_j. Допустимые маршруты должны проходить через все вершины, поэтому запрещается включение в маршрут дуги $u(x_j, x_k)$, если образуемый замкнутый контур проходит только через часть вершин графа (рис. 3.4).

В простейшем случае включение дуги $u(x_i, x_j)$ означает исключение обратной дуги $u(x_j, x_i)$, т.е. $c_{ji} = \infty$.

Сокращение размеров матрицы и исключение элемента c_{ji} позволяет провести дополнительное приведение матрицы и тем самым улучшить нижнюю оценку длины любого гамильтонова маршрута, содержащегося в подмножестве $G_1=[i, j]$.

Рассмотрим теперь вопрос о выборе дуги $u(x_i, x_j)$, играющей столь важную роль при разбиении множества гамильтоновых путей на подмножества.

Наиболее вероятно, что в оптимальный маршрут входят дуги, для которых в приведенной матрице $c_{ij} = 0$. Невключение в маршрут именно этих дуг резко увеличивает нижнюю оценку. Как уже отмечалось, исключая дугу $u(x_i, x_j)$, мы должны положить $c_{ij} = \infty$ и, осуществляя приведение вновь полученной матрицы, улучшить оценку. Константами приведения являются минимальный элемент в i-й строке h_i и такой же в j-м столбце h_j. Поэтому наиболее резкое изменение оценки произойдет в том случае, когда выбирается элемент матрицы расстояний, для которого:

А) $c_{ij} = 0$;

Б) после замены $c_{ij} = 0$ на $c_{ij} = \infty$ сумма минимальных элементов в i-й строке и в j-м столбце имеет наибольшее значение.

Формально алгоритм метода ветвей и границ выглядит следующим образом.

В начале любой итерации **t** известна верхняя оценка $\overline{F}_t(x)$ оптимального значения целевой функции. Имеется список задач (маршрутов), в котором некоторое подмножество значений c_{ij} изменено и принято равным ∞, а также подмножество $\{x_{ij}\}=\{x_{ij} \mid x_{ij}=1\}$.

На итерации 1 основной список включает две задачи: в одной из них c_{ij} изменено на ∞, а в другой – соответствующая переменная $x_{ij}=1$, а $c_{ij}= \infty$.

На итерации **t** выполняются следующие шаги.

Шаг 1. Прекратить вычисления, если основной список пуст. В противном случае выбрать одну задачу и вычеркнуть ее из основного списка.

Шаг 2. Определить нижнюю оценку целевой функции для любого цикла, порождаемого выбранной задачей. Если нижняя оценка больше или равна $\underline{F}_t(x)$, то принять $\underline{F}_{t+1}(x) = \underline{F}_t(x)$ и вернуться к шагу 1. В противном случае перейти к шагу 3.

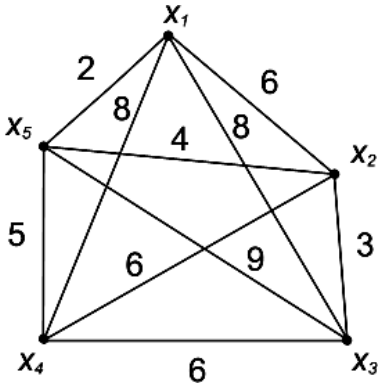
Шаг 3. Если текущее решение определяет цикл, то зафиксировать его, принять равным соответствующему значению целевой функции и вернуться к шагу 1. В противном случае – перейти к шагу 4.

Шаг 4. Выбрать переменную x_{hk}, не входящую в текущее решение, такую, что $c_{hk} < \infty$ при условии, что $x_{hk}=1$ не приводит к образованию подцикла на переменных, уже вошедших в решение. При таком выборе внести в основной список две задачи. Каждую из этих задач принять идентичной задаче, выбранной на шаге 1, за исключением лишь того, что в одну из них ввести изменение $c_{hk} =\infty$, а в другую – условие $x_{hk} =1$ и $c_{hk} = \infty$. Принять $\underline{F}_{t+1}(x) = \underline{F}_t(x)$ и вернуться к шагу 1.

3.4. ПРИМЕР

Для пояснения работы алгоритма рассмотрим следующий пример. Пусть дан граф, показанный на рис. 3.5. Исходная матрица расстояний C задана в виде таблицы.

Элементы главной диагонали матрицы $c_{ii} = \infty$, поскольку в графе отсутствуют петли.



$C=$

	x_1	x_2	x_3	x_4	x_5	h_i
x_1	∞	**6**	**8**	**8**	**2**	*2*
x_2	**6**	∞	**3**	**6**	**4**	*3*
x_3	**8**	**3**	∞	**6**	**9**	*3*
x_4	**8**	**6**	**6**	∞	**5**	*5*
x_5	**2**	**4**	**9**	**5**	∞	*2*

Рис. 3.5

Для получения нижней границы множества осуществим приведение матрицы по строкам. Для этой цели в каждой строке выбираем минимальный элемент h_i и вычитаем его из всех элементов строки. В данном случае константами приведения являются числа: 2, 3, 3, 5, 2. После приведения матрица расстояний C'_0 приобретает следующий вид:

$C_0'=$	x_1	x_2	x_3	x_4	x_5
x_1	∞	4	6	6	0
x_2	3	∞	0	3	1
x_3	5	0	∞	3	6
x_4	3	1	1	∞	0
x_5	0	2	7	3	∞
h_j	0	0	0	3	0

Сумма констант приведения дает некоторую нижнюю границу длин всех гамильтоновых путей $\sum_n h_i = 15$. Эта оценка может быть улучшена за счет приведения матрицы по столбцам. Минимальные элементы столбцов образуют константы приведения по столбцам h_j. В результате получим $\xi_0 = \sum_n h_i + \sum_n h_j = 15+3 = 18$, а матрица расстояний, приведенная по строкам и столбцам, приобретает вид матрицы C_0'':

$C_0''=$	x_1	x_2	x_3	x_4	x_5
x_1	∞	4	6	3	$0^{(3)}$
x_2	3	∞	$0^{(1)}$	$0^{(0)}$	1
x_3	5	$0^{(1)}$	∞	$0^{(0)}$	6
x_4	3	1	1	∞	$0^{(1)}$
x_5	$0^{(3)}$	2	7	$0^{(0)}$	∞

В соответствии с предписанием мы должны просмотреть все нулевые элементы этой матрицы и выбрать тот, для которого сумма констант приведения матрицы, получаемой в результате замены $c_{ij} = 0$ на $c_{ij} = \infty$, была бы максимальна. Поэтому подсчитываем сумму этих констант приведения для всех нулевых элементов.

Начнем с элемента c_{15}. Если положить $c_{15} = \infty$, то константа приведения по строкам была бы равна 3, а по столбцам – 0. Сумма констант приведения равна 3. Поступая аналогичным образом, устанавливаем, что для c_{23} сумма равна 1 + 1 = 2, для c_{32} – 0 + 2 = 2, для c_{34} – 0 + 0 = 0, для c_{45} – 1 + 0 = 1, для c_{51} – 0 + 3 = 3, для c_{54} – 0 + 0 = 0. Максимальная из этих сумм соответствует дуге $u(x_1, x_5)$. Разбиваем множество G_0 на два подмножества G_1=[1, 5] и G_2=[$\overline{1,5}$].

Исключение дуги $u(x_1, x_5)$ из числа дуг, входящих в искомый маршрут, достигается заменой этого элемента знаком ∞. В результате

такой замены появляется возможность осуществить дополнительное приведение матрицы путем вычитания из 1-й строки 3. В результате такого приведения матрица расстояний для множества G_2=[$\overline{1,5}$] примет вид C_2, а нижняя граница длин гамильтоновых контуров увеличится до $\xi_2 = \xi_0 + h = 18+3=21$.

$C_2=$	x_1	x_2	x_3	x_4	x_5
x_1	∞	1	3	0	∞
x_2	3	∞	0	0	1
x_3	5	0	∞	0	6
x_4	3	1	1	∞	0
x_5	0	2	7	0	∞

Включение дуги $u(x_1, x_5)$ в состав искомого маршрута ведет к автоматическому исключению из маршрута дуг $u(x_1, x_2)$, $u(x_1, x_3)$, $u(x_1, x_4)$, выходящих из 1-го пункта (т.е. всей 1-й строки), а также дуг $u(x_2, x_5)$, $u(x_3, x_5)$, $u(x_4, x_5)$, входящих в 5-й город (т.е. всего 5-го столбца). Кроме того, из числа дуг, входящих в маршрут, должна быть исключена дуга $u(x_5, x_1)$ с тем, чтобы запретить образование замкнутого контура $\{x_1, x_5, x_1\}$. Для улучшения оценки должны быть использованы числа матрицы, которая получается после удаления 1-й строки и 5-го столбца и замены величины c_{51} знаком ∞, т.е.

$C_1=$	x_1	x_2	x_3	x_4	h_i
x_2	3	∞	0	0	0
x_3	5	0	∞	0	0
x_4	3	1	1	∞	1
x_5	∞	2	7	0	0

Эта матрица также допускает дополнительное приведение по 4-й строке и соответствующее улучшение оценки на 1 единицу. После приведения матрица расстояний имеет вид C_1':

$C'_1 =$

	x_1	x_2	x_3	x_4
x_2	**3**	∞	**0**	**0**
x_3	**5**	**0**	∞	**0**
x_4	**2**	**0**	**0**	∞
x_5	∞	**2**	**7**	**0**
h_j	*2*	*0*	*0*	*0*

После приведения матрицы по 1-му столбцу нижняя граница снова увеличивается на 2 единицы и в результате становится равной $\xi_1 = \xi_0 + h = 18+3 = 21$. Приведенная матрица расстояний C''_1 для множества G_1=[1,5] имеет вид:

$C''_1 =$

	x_1	x_2	x_3	x_4
x_2	**1**	∞	$\mathbf{0}^{(0)}$	$\mathbf{0}^{(0)}$
x_3	**3**	$\mathbf{0}^{(1)}$	∞	$\mathbf{0}^{(0)}$
x_4	$\mathbf{0}^{(1)}$	$\mathbf{0}^{(0)}$	$\mathbf{0}^{(0)}$	∞
x_5	∞	**2**	**7**	$\mathbf{0}^{(2)}$

Таким образом, мы начали процесс образования дерева принятия решений (рис. 3.6).

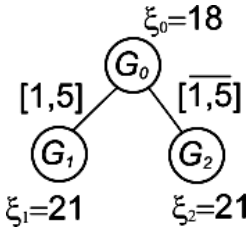


Рис. 3.6

Процесс дальнейшего ветвления рекомендуется вести так, чтобы обрезать большинство ветвей с нижней границей, превышающей минимальную оценку. В нашем случае нижние границы множеств G_1 и G_2 совпадают, поэтому для ветвления можно выбрать любое из них. Однако размерность матрицы C_1 меньше размерности матрицы C_2, поэтому в первую очередь мы разобьем на подмножества множество G_1=[1,5].

Множество G_1=[1,5] имеет в качестве нижней границы ξ_1=21 и допускает для построения кольцевого маршрута использование дуг, длины которых приведены в таблице C''_1.

В соответствии с предписанием алгоритма отыскиваем дугу, исключение которой максимально бы увеличило нижнюю оценку. Для этого выполняем те же действия, какие мы проделали при выборе дуги $u(x_1, x_5)$, т.е. просматриваем нулевые элементы, предварительно помеченные суммой констант приведения, образующихся после замены $c_{ij} = 0$ на $c_{ij} = \infty$. Максимальное значение суммы констант приведения соответствует элементу c_{54}. Поэтому будем улучшать нижнюю границу множеств G_3 ={[1,5], [5,4]} и G_4 ={[1,5], [$\overline{5,4}$]}.

Исключение дуги $u(x_5, x_4)$ приводит к изменению оценки на 2 единицы. Поэтому нижняя граница множества G_4 ={[1,5], [$\overline{5,4}$]} равна ξ_4 = 21+2=23. Соответствующая матрица C_4 приведена ниже:

$C_4 =$

	x_1	x_2	x_3	x_4
x_2	**1**	∞	**0**	**0**
x_3	**3**	**0**	∞	**0**
x_4	**0**	**0**	**0**	∞
x_5	∞	**0**	**5**	∞

Включение дуги $u(x_5, x_4)$ связано с исключением 5-й строки и 4-го столбца. Кроме того, для исключения неполного цикла $\{x_1, x_5, x_4, x_1\}$ должна быть запрещена дуга $u(x_4, x_1)$, как показано на рис. 3.7.

Для дальнейшего улучшения оценок должна быть использована приведенная таблица C'_3, изменяющая нижнюю границу множества G_3 ={[1,5], [5,4]} на 1, т.е. ξ_3=21+1 =22:

$C'_3 =$

	x_1	x_2	x_3
x_2	$\mathbf{0}^{(2)}$	∞	$\mathbf{0}^{(0)}$
x_3	**2**	$\mathbf{0}^{(2)}$	∞
x_4	∞	$\mathbf{0}^{(0)}$	$\mathbf{0}^{(0)}$

Соответственно, дерево решений примет следующий вид (рис. 3.8).

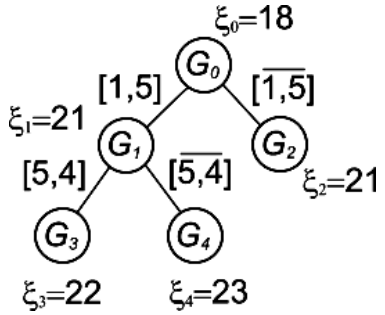


Рис. 3.8

Из определения констант приведения матрицы C'_3 следует, что максимальное увеличение нижней границы решения произойдет, если мы выберем дугу $u(x_2, x_1)$. В этом случае множество решений G_3 ={[1,5], [5,4]} разбивается еще на два подмножества G_5 ={[1,5], [5,4], [2,1]} и G_6 ={[1,5], [5,4], [$\overline{2,1}$]}. Оценка для последнего множества ξ_6=22+2 =24.

Для определения оценки подмножества G_5 необходимо вычеркнуть 2-ю строку и 1-й столбец матрицы C_3, а также запретить дугу $u(x_4, x_2)$, образующую неполный цикл $\{x_2, x_1, x_5, x_4, x_2\}$ (рис. 3.9).

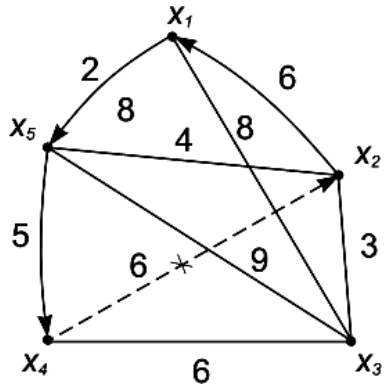


Рис. 3.9

Таким образом, матрица C_5 для множества G_5 ={[1,5], [5,4], [2,1]} будет иметь следующий вид:

$$C_5 = \begin{array}{c|cc} & x_2 & x_3 \\ x_3 & \mathbf{0}^{(\infty)} & \infty \\ x_4 & \infty & \mathbf{0}^{(\infty)} \end{array}$$

Последняя матрица C_5 прямо указывает на то, что в искомый маршрут надо включить дуги $u(x_3, x_2)$ и $u(x_4, x_3)$, не меняя при этом нижней границы оценки, которая становится точной оценкой длины маршрута.

Таким образом, дальнейшее развитие дерева привело к образованию определенного гамильтонова маршрута длиной 22 единицы (рис. 3.10).

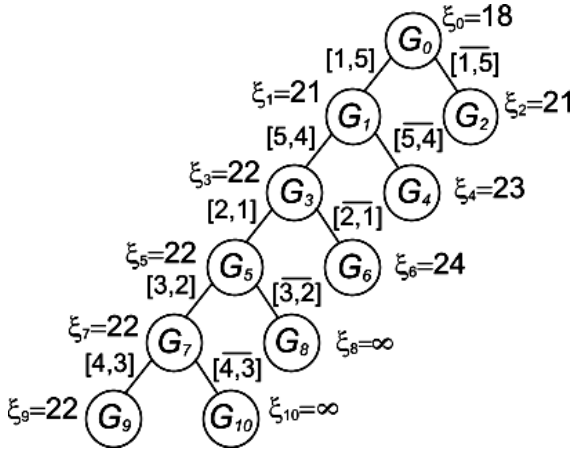


Рис. 3.10

Сопоставляя длину найденного маршрута ξ_9 с нижней границей множеств G_4 ={[1,5], [$\overline{5,4}$]} и G_6 ={[1,5], [5,4], [$\overline{2,1}$]}, не рассмотренных до конца, мы видим, что рассмотрение их не имеет смысла, так как их оценки больше длины найденного маршрута ξ_9.

Однако может оказаться, что среди гамильтоновых контуров подмножества G_2=[$\overline{1,5}$] с нижней границей ξ_2=21 также находится оптимальное решение. Поэтому необходимо вернуться и рассмотреть матрицу C_2:

$$C_2 = \begin{array}{c|ccccc} & x_1 & x_2 & x_3 & x_4 & x_5 \\ x_1 & \infty & \mathbf{1} & \mathbf{3} & \mathbf{0}^{(1)} & \infty \\ x_2 & \mathbf{3} & \infty & \mathbf{0}^{(1)} & \mathbf{0}^{(0)} & \mathbf{1} \\ x_3 & \mathbf{5} & \mathbf{0}^{(1)} & \infty & \mathbf{0}^{(0)} & \mathbf{6} \\ x_4 & \mathbf{3} & \mathbf{1} & \mathbf{1} & \infty & \mathbf{0}^{(2)} \\ x_5 & \mathbf{0}^{(3)} & \mathbf{2} & \mathbf{7} & \mathbf{0}^{(0)} & \infty \end{array}$$

Следуя алгоритму, мы должны рассмотреть подмножества G_{11}={[$\overline{1,5}$], [5,1]} и G_{12}={[$\overline{1,5}$], [$\overline{5,1}$]}. Нижняя граница множества G_{12} больше оценки ξ_9 и равна ξ_{12}=21+3=24, в силу чего это подмножество может в дальнейшем не рассматриваться. Что касается подмножества G_{11}, то

включение дуги $u(x_5, x_1)$ требует удаления 5-й строки и 1-го столбца в матрице C_2. Обратная дуга $u(x_1, x_5)$ уже была нами запрещена: $c_{15} = \infty$.

В результате преобразований получаем матрицу C_{11}, которая не допускает улучшения оценки – нижняя граница множества G_{11}={[$\overline{1,5}$], [5,1]} при этом также равна ξ_{11}=21:

C_{11}=

	x_2	x_3	x_4	x_5
x_1	**1**	**3**	**0**$^{(1)}$	∞
x_2	∞	**0**$^{(1)}$	**0**$^{(0)}$	**1**
x_3	**0**$^{(1)}$	∞	**0**$^{(0)}$	**6**
x_4	**1**	**1**	∞	**0**$^{(2)}$

Рассмотрим дальнейшее развитие дерева принятия решения. Анализируя таблицу C_{11} аналогичным образом, легко определить, что в маршрут должна входить дуга $u(x_4, x_5)$, имеющая максимальное значение суммы констант приведения. Для исключения неполного цикла дуге $u(x_1, x_4)$ присвоим значение c_{14}=∞. В результате получаем матрицу C_{13}, отражающую множество решений G_{13}={[$\overline{1,5}$], [5,1], [4,5]}:

C_{13}=

	x_2	x_3	x_4	h_i
x_1	**1**	**3**	∞	*1*
x_2	∞	**0**	**0**	*0*
x_3	**0**	∞	**0**	*0*

Матрица C_{13} приводится по 1-й строке на 1:

C'_{13} =

	x_2	x_3	x_4
x_1	**0**$^{(2)}$	**2**	∞
x_2	∞	**0**$^{(2)}$	**0**$^{(0)}$
x_3	**0**$^{(0)}$	∞	**0**$^{(0)}$

Оценка нижней границы множества G_{14}={[$\overline{1,5}$], [5,1], [$\overline{4,5}$]}, не включающего дугу $u(x_4, x_5)$, будет равна $\xi_{14}=\xi_{11}+2=23$. Следовательно, его можно дальше не рассматривать. Нижняя граница множества G_{13}={[$\overline{1,5}$], [5,1], [4,5]} увеличивается на приведенную единицу $\xi_{13}=\xi_{11}+1=22$.

Из приведенной матрицы C'_{13} следует ветвление множества решений G_{13} по дуге $u(x_1, x_2)$. При этом необходимо исключить дугу $u(x_2, x_4)$, образующую неполный цикл $\{x_4, x_5, x_1, x_2, x_4\}$. Полученная матрица C_{15} не приводится и оценка не улучшается:

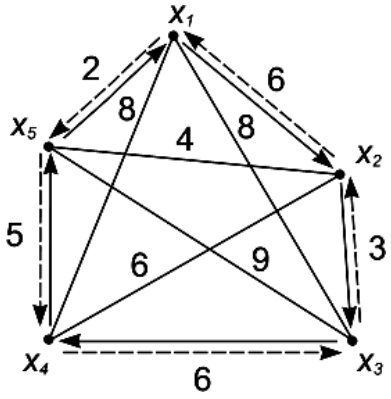


Рис. 3.11

C_{15}=

	x_3	x_4
x_2	**0**$^{(\infty)}$	∞
x_3	∞	**0**$^{(\infty)}$

Таким образом, в маршрут включаются еще две дуги: $u(x_2, x_3)$ и $u(x_3, x_4)$, образующие кратчайший гамильтонов цикл $\{x_1, x_2, x_3, x_4, x_5, x_1\}$.

Задача имеет два симметричных решения: это найденный ранее цикл G_9: $\{x_1, x_5, x_4, x_3, x_2, x_1\}$ и цикл G_{17}: $\{x_1, x_2, x_3, x_4, x_5, x_1\}$, направленный в другую сторону (рис. 3.11).

Полное дерево решений данного примера показано на рис. 3.12.

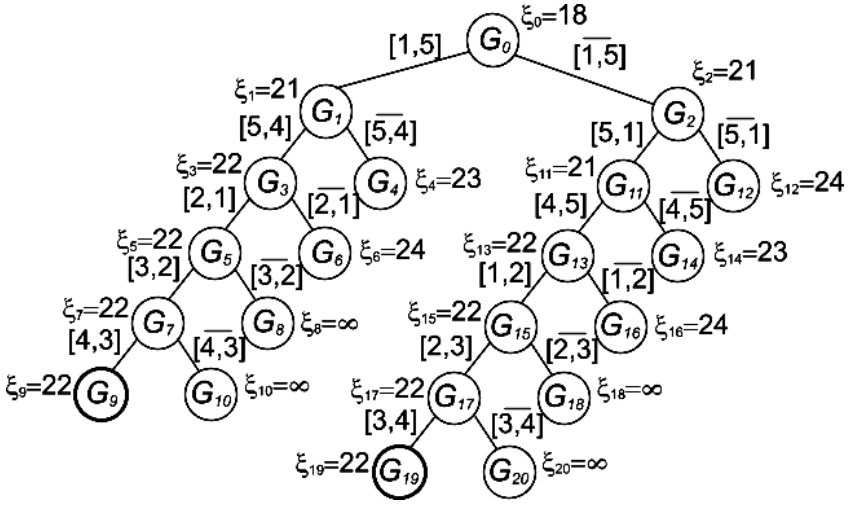


Рис. 3.12

3.5. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определение цикла, контура.
2. Какой цикл называется эйлеровым? Критерий существования эйлерова цикла.
3. Какой цикл называется гамильтоновым?
4. Постановка задачи коммивояжера.
5. Для чего осуществляется приведение матрицы расстояний в задаче коммивояжера?
6. В чем сущность метода ветвей и границ?
7. Какие элементы составляют множество решений G_0?
8. Как определяется нижняя граница длины гамильтонова цикла для каждого из разбиваемых подмножеств решений?
9. Как формируется матрица расстояний для каждого из разбиваемых подмножеств решений?
10. По заданному преподавателем графу найти с помощью метода ветвей и границ кратчайший гамильтонов цикл. Построить дерево решений.

4. ТРАНСПОРТНАЯ ЗАДАЧА

4.1. ДВУДОЛЬНЫЕ ГРАФЫ

Граф $G=(X, U)$ называется *двудольным* (графом Кенига), если его множество вершин X разбивается на два непересекающихся подмножества $X_1 \bigcap X_2 = \varnothing$ так, что начало любой дуги $u(x_i, x_j) \in U$ инцидентно вершинам первого подмножества $x_i \in X_1$, а конец – вершинам второго подмножества $x_j \in X_2$.

Любые две вершины, принадлежащие одному и тому же подмножеству X_i, являются несмежными. Множества вершин X_1 и X_2 двудольного графа G называются *долями*. Двудольный граф $G=(X, U)$ называется *полным*, если он содержит все ребра, соединяющие множества X_1 и X_2.

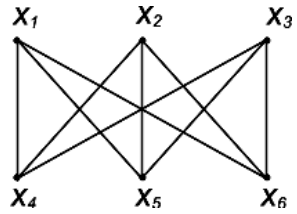


Рис. 4.1.

Полный двудольный граф обозначается как $K_{n,m}$, где $n=|X_1|$, $m=|X_2|$ – число вершин в каждой доли.

Например, на рис. 4.1 показан полный граф $K_{3,3}$.

С двудольными графами связан ряд задач, таких как *задача о назначениях, паросочетания, транспортная задача*.

Транспортная задача сводится к построению взвешенного двудольного графа минимальной стоимости.

4.2. МОДЕЛЬ ТРАНСПОРТНОЙ ЗАДАЧИ

Транспортная задача является одной из самых распространенных специальных задач линейного программирования.

Первая строгая постановка задачи принадлежит Хичкоку, и поэтому в зарубежной литературе ее называют проблемой Хичкока (1941 г.). Первый точный метод решения задачи разработан советскими учеными Л.В. Канторовичем и М.К. Гавуриным.

Пример 1. В пунктах $P_1, P_2, \ldots, P_l$ имеется однородный груз в количествах $a_1, a_2, \ldots, a_l$. Его необходимо перевезти в пункты $Q_1, Q_2, \ldots, Q_r$ в количествах $b_1, b_2, \ldots, b_r$ так, чтобы общая стоимость перевозок была минимальна (рис. 4.2). При этом предполагается, что суммарный запас всех перевозок равен суммарному объему потребления:

$$\sum_i a_i = \sum_j b_j \,. \qquad (4.1)$$

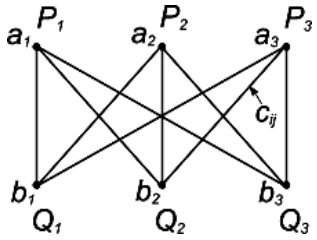


Рис. 4.2

Обозначим через x_{ij} – количество груза, перевозимого из пункта P_i в Q_j, а через c_{ij} – стоимость перевозки единицы этого груза.

В задаче имеются следующие ограничения.

1. Количество груза, отправляемого из пункта P_i, должно быть равно имеющимся запасам a_i:

$$\sum_j x_{ij} = a_i, \qquad i = \overline{1,I}. \qquad (4.2)$$

2. Количество груза, поставляемого в пункт Q_j, должно быть равно имеющимся потребностям b_j:

$$\sum_i x_{ij} = b_j, \qquad j = \overline{1,r}. \qquad (4.3)$$

Целевая функция определяет полную стоимость перевозки всего груза:

$$F(x) = \sum_i \sum_j c_{ij} x_{ij} \to min. \qquad (4.4)$$

Пример 2. Имеются источники энергии P_1, P_2,…, P_I (газа, нефти, электроэнергии) и потребители Q_1, Q_2,…, Q_r с соответствующими возможностями a_i и потребностями b_j. Требуется спроектировать минимальную сеть передачи энергии от источников к потребителям.

Пример 3. Имеются клиентские терминалы Q_1, Q_2, …, Q_r с соответствующими потребностями b_j и вычислительный центр коллективного пользования с серверами P_1, P_2, …, P_I и системными ресурсами a_i. Спроектировать терминальную сеть по критерию минимизации используемых системных ресурсов.

Специфика формальной постановки транспортной задачи:

- ограничения заданы в виде линейных уравнений;

- каждая неизвестная входит лишь в 2 уравнения;
- коэффициенты при неизвестных – единицы.

Для ее решения созданы специальные методы, значительно менее громоздкие, чем симплекс-метод.

4.3. РАСПРЕДЕЛИТЕЛЬНЫЙ МЕТОД

Простой *распределительный метод* решения транспортной задачи (Канторович, Гавурин, 1949 г.) состоит в определении оптимального размещения ресурсов путем итерационных циклических перестановок. По Канторовичу он называется *методом потенциалов*. Данциг несколько позже и независимо от него также открыл этот метод, который назвал *stepping-stone*.

Исходные данные удобно представить в виде табл. 4.1.

Таблица 4.1

Отправитель	Получатель			
	Q_1	Q_2	Q_3	Запас a_i
P_1	x_{11} 1	x_{12} 3	x_{13} 4	12
P_2	x_{21} 2	x_{22} 5	x_{23} 3	8
P_3	x_{31} 6	x_{32} 7	x_{33} 5	10
Потребность b_j	6	9	15	$\sum = 30$

В клетках таблицы расположены 9 неизвестных x_{ij}, обозначающих объем перевозок (в тоннах или единицах продукции) от i-го отправителя к j-му получателю. В тех же клетках в правом верхнем углу указаны стоимости (или расстояния) перевозки между пунктами.

Дадим задаче математическую формулировку:

$$\begin{aligned} min\ F(x) = {} & x_{11} + 3x_{12} + 4x_{13} + \\ & + 2x_{21} + 5x_{22} + 3x_{23} + \\ & + 6x_{31} + 7x_{32} + 5x_{33}, \end{aligned}$$

где $F(x)$ – общая стоимость перевозок в руб. или общий пробег груза в тыс.·км.

Ограничения в данной задаче связаны с тем, что из каждого пункта отправляется и в каждый пункт доставляется строго определенное количество груза. Например, из пункта P_1 можно отправлять груз трем получателям, но общее количество отправляемого груза – 12.

Математически это условие выражается уравнением:

$$x_{11} + x_{12} + x_{13} = 12.$$

Для отправителей P_2 и P_3 соответственно имеем:

$$x_{21} + x_{22} + x_{23} = 8;$$

$$x_{31} + x_{32} + x_{33} = 10.$$

С другой стороны, в пункт назначения Q_1 груз может поступать из 3-х различных мест отправки, но в общем количестве 6. Математически это можно выразить так:

$$x_{11} + x_{21} + x_{31} = 6.$$

Соответственно, для получателей Q_2 и Q_3 имеем:

$$x_{12} + x_{22} + x_{32} = 9;$$

$$x_{13} + x_{23} + x_{33} = 15.$$

Однако эти ограничения не являются независимыми друг от друга, поскольку

$$6 + 9 + 15 = 12 + 8 + 10 = 30.$$

Окончательно имеем 6 ограничений и $n - m = 9 - 6 = 3$ свободных переменных, причем независимых ограничений только 5.

Распределительный метод решения этой задачи принципиально очень прост и может быть легко усвоен и применен на практике. Этот метод получения оптимального размещения ресурсов аналогичен симплекс-методу: вначале принимается некоторый начальный вариант перевозок, удовлетворяющий вышеуказанным ограничениям, затем последовательно производится улучшение его до получения оптимального значения целевой функции.

Для составления первоначального, исходного плана перевозок удобно пользоваться *правилом северо-западного угла* [5]. Математически доказано, что оптимальное значение целевой функции в задаче линейного программирования располагается на вершине многогранника решений, образуемого из граничных условий. Такое решение достигается, если часть переменных целевой функции обращается в ноль, образуя при этом совместную систему линейных уравнений. Другими словами, мы должны так заполнить таблицу перевозок, чтобы она содержала, как минимум, $(m - 1)(n - 1)$ пустых клеток с учетом заданных ограничений, а также иметь ввиду это обстоятельство в ходе дальнейших перестановок.

В соответствии с правилом северо-западного угла в левый верхний угол таблицы заносится максимально допустимое значение ресурса, соответствующее граничным условиям ($x_{11} = min\{a_1, b_1\}$). Далее таблица

последовательно заполняется остаточными ресурсами слева направо и сверху вниз. При этом ресурс размещается в пустых соседних клетках, каждая из которых получает максимально допустимое значение в соответствии с ограничениями (4.2) и (4.3). Заполнение осуществляется до распределения всех остатков груза в таблице.

Количество заполненных клеток составит $m + n - 1$, где n – количество получателей, а m – количество поставщиков.

Рассмотрим клетку P_1Q_1 (табл. 4.2). Максимальное число, которое может в ней находиться – 6. Заполняем эту клетку цифрой 6 и переходим к следующей клетке этой строки. В ней тоже может находиться не более 6 т. Заполнив две клетки, мы использовали полностью возможности отправителя P_1. При этом в пункт Q_2 будет доставляться 6 т при потребности 9 т. Недостающие 3 т могут поступать из следующего места отправки – пункта P_2. Поставим в клетке P_2Q_2 цифру 3. Аналогично заполним остальные клетки таблицы.

Таблица 4.2

Отправитель	Получатель			
	Q_1	Q_2	Q_3	Запас a_i
P_1	**6** 1	**6** 3	4	12
P_2	2	**3** 5	**5** 3	8
P_3	6	7	**10** 5	10
Потребность b_j	6	9	15	$\sum = 30$

Полученный вариант, возможно, не является оптимальным, но он удовлетворяет всем требуемым ограничениям задачи:

$$F_0(x) = 6*1 + 6*3 + 3*5 + 5*3 + 10*5 = 104.$$

На втором этапе расчета предстоит определить, можно ли улучшить исходящую программу, чтобы уменьшить значение целевой функции $F(x)$.

В исходном варианте заполнены 5 клеток ($m + n - 1 = 3 + 3 - 1 = 5$), а четыре клетки при этом остались незаполненными.

Улучшение перевозок может заключаться в использовании отдельных незанятых клеток вместо заполненных. Поэтому необходимо проверить каждую свободную клетку с точки зрения целесообразности ее включения в программу перевозок.

Допустим, что значение пустой клетки P_1Q_3 увеличится на единицу (поставим в ней знак «+»). Тогда, с учетом того, что отправитель P_1 в сумме должен перевезти 12 т, это увеличение может произойти лишь за счет другого получателя (например, Q_2). Для этого поставим в соседней занятой клетке P_1Q_2 знак «–». Кроме того, получатель Q_3 имеет потребность в сумме 15 т груза, поэтому необходимо уменьшить на единицу значение перевозимого к нему груза от другого отправителя (например, P_2). Для этого поставим в соседней занятой клетке P_2Q_3 знак «–». Но в этом случае на единицу также уменьшатся суммарный запас отправителя P_2 и общая потребность получателя Q_2. Следовательно, чтобы решение удовлетворяло заданным ограничениям, необходимо также увеличить на единицу значение клетки P_2Q_2 (см. табл. 4.3).

Таблица 4.3

Отправитель	Получатель			
	Q_1	Q_2	Q_3	Запас a_i
P_1	**6** 1	**6** –3	(3) +4	12
P_2	2	**3** +5	**5** –3	8
P_3	6	7	**10** 5	10
Потребность b_j	6	9	15	$\sum = 30$

Заполняя одну из незаполненных клеток, необходимо одновременно изменить значения, по меньшей мере, в трех заполненных клетках. Эту связь между клетками можно установить путем построения так называемых *многоугольников замены*. Многоугольник строится так, что одна из его вершин находится в пустой клетке, а остальные – в заполненных; при этом все углы многоугольника должны быть прямыми. Клетки, входящие в вершины многоугольника, образуют *цепочки замены* с чередующимися знаками. Например, в табл. 4.3 показан многоугольник замены, состоящий из клеток: (P_1Q_3, P_2Q_3, P_2Q_2, P_1Q_2). При правильном заполнении первоначального варианта перевозок каждой свободной клетке будет соответствовать один такой многоугольник.

При размещении единицы ресурса в пустую клетку целевая функция изменится на величину стоимостей перевозок клеток, составляющих цепочку замены. Положительные клетки цепочки будут увеличивать значение функции, а отрицательные – уменьшать. Эта суммарная величина называется *потенциалом* пустой клетки. Например, для многоугольника, показанного на табл. 4.3, имеем: $\varphi_{13} = +4 - 3 + 5 - 3 = 3$.

Определим многоугольник замены для пустой клетки P_2Q_1. Он будет формироваться из соседних занятых клеток (P_2Q_1, P_2Q_2, P_1Q_2, P_1Q_1) (табл. 4.4). При размещении единицы ресурса в клетке P_2Q_1 необходимо одновременно уменьшить на единицу значения в клетках P_2Q_2 и P_1Q_1, а также увеличить значение в клетке P_1Q_2. Потенциал клетки P_2Q_1 будет определяться суммой: $\varphi_{21} = +2 - 5 + 3 - 1 = -1$.

Таблица 4.4

Отправитель	Получатель			
	Q_1	Q_2	Q_3	Запас a_i
P_1	**6** –1	**6** +3	(3) 4	12
P_2	(–1) +2	**3** –5	**5** 3	8
P_3	6	7	**10** 5	10
Потребность b_j	6	9	15	$\sum = 30$

Свободной клетке P_3Q_1 соответствует более сложный многоугольник, вершины которого лежат в шести клетках (P_3Q_1, P_3Q_3, P_2Q_3, P_2Q_2, P_1Q_2, P_1Q_1). При этом ресурсы в клетках P_3Q_1, P_2Q_3 и P_1Q_2 будут увеличиваться, а в клетках P_3Q_3, P_2Q_2 и P_1Q_1 – уменьшаться (табл. 4.5).

Таблица 4.5

Отправитель	Получатель			
	Q_1	Q_2	Q_3	Запас a_i
P_1	**6** –1	**6** +3	(3) 4	12
P_2	(–1) 2	**3** –5	**5** +3	8
P_3	(1) +6	7	**10** –5	10
Потребность b_j	6	9	15	$\sum = 30$

Таким образом, потенциал клетки P_3Q_1 будет определяться суммой: $\varphi_{31} = +6 - 5 + 3 - 5 + 3 - 1 = 1$.

И, наконец, для пустой клетки P_3Q_2 имеем многоугольник из четырех соседних клеток (P_3Q_2, P_3Q_3, P_2Q_3, P_2Q_2). Соответственно, значение потенциала клетки P_3Q_2 определится из суммы: $\varphi_{32} = +7 - 5 + 3 - 5 = 0$.

Таблица 4.6

Отправитель	Получатель			
	Q_1	Q_2	Q_3	Запас a_i
P_1	**6** 1	**6** 3	(3) 4	12
P_2	(–1) 2	**3** –5	**5** +	8
P_3	(1) 6	(0) +7	**10** –	10
Потребность b_j	6	9	15	$\sum = 30$

Из полученной табл. 4.6 следует, что при размещении единицы ресурса в свободную клетку P_1Q_3 значение целевой функции увеличится на 3, а при размещении в клетку P_3Q_1 – на единицу. Размещение ресурса в клетку P_3Q_2 не изменит значение целевой функции, и лишь размещение ресурса в клетку P_2Q_1 будет уменьшать функцию на единицу. Следовательно, мы выберем клетку P_2Q_1, что приведет к уменьшению общего пробега на 1 км для каждой перевозимой тонны груза.

В итоге установлено, что исходный вариант распределения не является оптимальным и может быть улучшен.

На третьем этапе производится улучшение начального распределения, а именно: перемещение грузов в пределах многоугольника замены перспективной клетки P_2Q_1 (табл. 4.7). Для этого в клетку P_2Q_1 размещается ***наименьшее*** из чисел, находящихся в клетках многоугольника замены, которые будут ***уменьшаться*** в ходе распределения. В данном случае это число 3 (клетка P_2Q_2). На эту же величину необходимо увеличить значение в клетке P_1Q_2 и уменьшить значение в клетке P_1Q_1.

В результате получим новое распределение, которое изменяет значение целевой функции на величину произведения потенциала перспективной клетки и размещаемого в нее ресурса (табл. 4.7):

$$F_1(x) = F_0 + \varphi_{21} * x_{21} = 104 - 1*3 = 101.$$

Таблица 4.7

Отправитель	Получатель			
	Q_1	Q_2	Q_3	Запас a_i
P_1	**3** 1	**9** 3	4	12
P_2	**3** 2	5	**5** 3	8
P_3	6	7	**10** 5	10
Потребность b_j	6	9	15	$\sum = 30$

Таким образом, использование данной свободной клетки приводит к уменьшению общего пробега на 1 км в расчете на 1 т груза, а для 3 т выигрыш составит 3 км.

Однако и новый вариант может оказаться неоптимальным. Для проверки возможности его улучшения надо снова построить многоугольники замены и подсчитать изменение суммарной стоимости перевозок при распределении ресурсов в свободные клетки, т.е. определить их потенциалы.

Заметим, что для клетки P_3Q_2 многоугольник имеет вид (табл. 4.8).

Таблица 4.8

Отправитель	Получатель			
	Q_1	Q_2	Q_3	Запас a_i
P_1	**3** +1	**9** –3	4	12
P_2	**3** –2	5	**5** +3	8
P_3	6	+7	**10** –5	10
Потребность b_j	6	9	15	$\sum = 30$

Этот многоугольник состоит из 6 клеток (P_3Q_2, P_1Q_2, P_1Q_1, P_2Q_1, P_2Q_3, P_3Q_3).

Потенциал клетки P_3Q_2 составляет: $\varphi_{32} = +7 - 3 + 1 - 2 + 3 - 5 = 1$.

Потенциалы остальных пустых клеток образуют прямоугольники и квадрат. Например, для клетки P_1Q_3 прямоугольник будет состоять из

клеток (P_1Q_3, P_2Q_3, P_2Q_1, P_1Q_1), а ее потенциал $\varphi_{13} = +4-3+2-1=2$; для клетки P_3Q_1 мы имеем прямоугольник (P_3Q_1, P_3Q_3, P_2Q_3, P_2Q_1) с потенциалом $\varphi_{31} = +6-5+3-2=2$; для клетки P_2Q_2 многоугольник представляет собой квадрат (P_2Q_2, P_1Q_2, P_1Q_1, P_2Q_1) с потенциалом $\varphi_{22} = +5-3+1-2=1$.

Таким образом, все потенциалы пустых клеток являются положительными числами (табл. 4.9).

Таблица 4.9

Отправитель	Получатель			
	Q_1	Q_2	Q_3	Запас a_i
P_1	**3** +1	**9** –3	(2) 4	12
P_2	**3** –2	(1) 5	**5** +3	8
P_3	(2) 6	(1) +7	**10** –5	10
Потребность b_j	6	9	15	$\sum = 30$

Это означает, что распределение перевозок, представленное в табл. 4.9, улучшено быть не может и является оптимальным.

Оптимальный вариант получен всего лишь с помощью одной итерации, при этом уменьшение общего пробега оказалось незначительным. В большей степени это объясняется тем, что сам исходный вариант близок к оптимальному. В ходе каждой итерации происходит переход к очередной соседней вершине многогранника решений, приближаясь к оптимальному результату.

Итерационный процесс будет осуществляться до тех пор, пока все пустые клетки таблицы не будут иметь положительные потенциалы, не улучшающие результат последнего полученного распределения.

4.4. ОБОБЩЕНИЕ ТРАНСПОРТНОЙ ЗАДАЧИ

Как было показано в п. 4.3, транспортная задача задается в виде:

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

при ограничениях:

$$\sum_{j=1}^{n} x_{ij} = a_i, \qquad i = \overline{1,m};$$

$$\sum_{i=1}^{m} x_{ij} = b_j, \qquad j = \overline{1,n}; \qquad x_{ij} \geq 0.$$

Всего задача включает $m \cdot n$ переменных и $m + n$ ограничений, из которых m ограничений связано с запасами отправителей и n – с потребностями получателей. Если первую группу ограничений просуммировать по i, а вторую – по j, то в левых частях полученных двух уравнений будет находиться одна и та же величина $\sum_i \sum_j x_{ij}$.

Следовательно, должны быть равны и правые части, а именно

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j .$$

Таким образом, система ограничивающих уравнений транспортной задачи является совместной лишь в том случае, когда общий запас груза отправителей в точности равен суммарной потребности получателей.

Отсюда вытекает еще одно важное следствие. Из $m + n$ уравнений в системе ограничений транспортной задачи одно (любое) уравнение можно отбросить, так как оно вытекает из остальных $m + n - 1$ уравнений. Действительно, если определены, например, наличие груза у всех отправителей и потребность всех получателей, кроме одного, то спрос последнего легко устанавливается как разница между общим запасом и общей потребностью других получателей.

Поскольку модель транспортной задачи содержит $m + n - 1$ независимых уравнений, любой ее невырожденный опорный план включает $m + n - 1$ переменных с положительным значением. Поэтому из $m \cdot n$ возможных маршрутов перевозок в оптимальном плане транспортной задачи загружается не более $m + n - 1$ маршрутов.

Невырожденным планом называется такой, который при m ограничениях и n неизвестных всегда содержит m положительных переменных, а остальные $n - m$ переменных в базис не входят и равняются нулю. Однако возможно равенство нулю одной или нескольких переменных, не входящих в базис. Такой план называется ***вырожденным***.

В линейном программировании доказывается, что любая транспортная задача имеет оптимальный план. При этом если исходные величины a_i и b_j являются целыми числами, то и все переменные в оптимальном плане

будут целыми величинами. Свойство целочисленности оказывается практически важным, например, при планировании поставок неделимых грузов.

Рассмотренная модель транспортной задачи, в которой выполняется условие (4.1), называется ***закрытой*** моделью.

В экономических расчетах немалую роль играют и так называемые ***открытые*** модели, в которых указанное равенство не соблюдается. При этом возможны два случая: или спрос превышает предложение (дефицит), или, наоборот, запас у поставщиков больше потребности получателей (профицит).

Если запасы грузов у поставщиков больше потребности получателей, то условия открытой транспортной задачи имеют вид:

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

при ограничениях:

$$\sum_{j=1}^{n} x_{ij} \le a_i, \qquad i = \overline{1,m};$$

$$\sum_{i=1}^{m} x_{ij} = b_j, \qquad j = \overline{1,n}; \qquad x_{ij} \ge 0.$$

Поскольку не весь имеющийся груз будет направляться получателям, первая группа ограничивающих условий имеет форму неравенств, которые, как известно, можно преобразовать в уравнения с помощью введения дополнительных неотрицательных переменных.

Тогда вместо неравенств будем иметь систему уравнений:

$$\sum_{j=1}^{n+1} x_{ij} = a_i, \qquad i = \overline{1,m};$$

или в развернутом виде

$x_{11} + x_{12} + \ldots + x_{1n} + x_{1,n+1} = a_1;$

……

$x_{m1} + x_{m2} + \ldots + x_{mn} + x_{m,n+1} = a_m,$

где $x_{1,n+1}$; $x_{2,n+1}$; …; $x_{m,n+1}$ – дополнительные переменные, обозначающие неиспользуемую для перевозок часть запасов.

Сумма этих дополнительных переменных должна быть равна разнице между общим запасом и общей потребностью:

$$\sum_{i=1}^{m} x_{i,n+1} = \sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j = b_{усл}.$$

Таким образом, в данном случае в открытую транспортную задачу как бы включается ***условный потребитель***, которому в качестве спроса приписывается разница между наличием груза и фактической потребностью в нем. Дополнительные переменные входят в целевую функцию с нулевыми коэффициентами.

Модель открытой транспортной задачи с включением условного потребителя представлена в табл. 4.10.

Таблица 4.10

Отправитель	Получатель					
	Q_1	Q_2	…	Q_n	$Q_{усл.}$	Запас
P_1	x_{11} c_{11}	x_{12} c_{12}	…	x_{1n} c_{1n}	$x_{1,n+1}$ 0	a_1
P_2	x_{21} c_{21}	x_{22} c_{22}	…	x_{2n} c_{2n}	$x_{2,n+1}$ 0	a_2
…	…	…	…	…	…	…
P_m	x_{m1} c_m	x_{m2} c_m	…	x_{mn} c_{mn}	$x_{m,n+1}$ 0	a_m
Потребность b_j	b_1	b_2	…	b_n	$b_{усл.}$	

Введением ***условного получателя*** открытая модель преобразуется в закрытую и решается затем как обычная транспортная задача.

При решении открытой транспортной задачи с включением условного потребителя в оптимальном плане основные переменные покажут рациональные маршруты и объемы перевозок на них, а дополнительные переменные – неперевозимый остаток запасов.

Другой вариант открытой транспортной задачи возникает тогда, когда спрос получателей оказывается выше возможностей поставщиков. В этом случае, очевидно, потребности некоторых получателей не будут удовлетворены, и задача формулируется следующим образом:

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

при ограничениях:

$$\sum_{j=1}^{n} x_{ij} = a_i, \qquad i = \overline{1,m};$$

$$\sum_{i=1}^{m} x_{ij} \le b_j, \qquad j = \overline{1,n}; \qquad x_{ij} \ge 0.$$

Очевидно, что здесь дополнительные переменные должны вводиться во вторую группу ограничений. Это равнозначно включению в модель условного отправителя, у которого наличие груза равно разнице между общей потребностью и общим фактическим запасом:

$$\sum_{j=1}^{n} x_{m+1,j} = \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i = a_{усл}.$$

Вместе с условным поставщиком расширенная модель оказывается закрытой моделью, и к ней применяется один из общих методов решения транспортной задачи. Часть спроса, не обеспечиваемая наличными запасами, в оптимальном плане будет отражена в строке условного отправителя.

Модель открытой транспортной задачи с включением условного потребителя представлена в табл. 4.11.

Таблица 4.11

Отправитель	Получатель				
	Q_1	Q_2	…	Q_n	Запас a_i
P_1	x_{11} c_1	x_{12} c_1	…	x_{1n} c_1	a_1
P_2	x_{21} c_2	x_{22} c_2	…	x_{2n} c_2	a_2
…	…	…	…	…	…
P_m	x_{m1} c_m	x_{m2} c_m	…	x_{mn} c_m	a_m
$P_{усл.}$	$x_{m+1,1}$ 0	$x_{m+1,2}$ 0		$x_{m+1,n}$ 0	$a_{усл}$
Потребность b_j	b_1	b_2	…	b_n	

При решении транспортной задачи распределительным методом процесс вычислений распадается на следующие основные этапы.

1. Составляется первоначальное распределение перевозок. Чаще всего для этого применяется правило северо-западного угла. Обязательно при

этом соблюдать непрерывность последовательного распределения ресурсов. Если ни в одну из двух соседних клеток нечего ставить, т.е. возможности исчерпаны, то в любой из них проставляется *0* и процесс продолжается дальше.

2. Для каждой свободной клетки составляется многоугольник замены, остальные вершины которого лежат в загруженных клетках. В вершинах многоугольника расставляются соответствующие показатели критерия оптимальности (расстояния или затраты) с чередующимися знаками (начиная с «+» для свободной клетки). Отрицательная алгебраическая сумма этих показателей гарантирует улучшение плана. Если исходный план составлялся по правилу северо-западного угла с включением в необходимых случаях нулевых поставок, то всегда для каждой свободной клетки можно составить только один замкнутый многоугольник замены.

3. После определения наиболее перспективной из свободных клеток осуществляется переход к новому варианту перевозок. В данную свободную клетку записывается наименьший из грузов, стоящих в отрицательных клетках многоугольника замены. В результате ранее свободная клетка становится загруженной, а отрицательная клетка, ранее содержащая минимальное значение, превращается в свободную. Если в пределах данного многоугольника одинаковый минимальный груз имели несколько отрицательных клеток, то освобождаться может лишь одна из них, а остальные должны считаться занятыми (с нулевыми поставками), так как иначе на следующем шаге нельзя будет построить многоугольники для всех свободных клеток.

4. Вновь полученный вариант плана проверяется на оптимальность, для чего составляются многоугольники и вычисляются потенциалы для каждой свободной клетки. Если есть отрицательные потенциалы, то план перевозок можно улучшить, если все потенциалы положительные, то получен единственно возможный оптимальный вариант. Наличие наряду с положительными нулевых потенциалов говорит о возможности построения множества оптимальных планов данной задачи.

При решении больших и средних по размеру матриц транспортной задачи распределительным методом чрезмерно громоздкими и трудоемкими являются процесс построения многоугольников замены и вычисление потенциалов всех свободных клеток на каждой итерации расчета. Так, например, если $m = 10$, $n = 10$, то таблица будет содержать 100 клеток, из которых на каждом шаге $m + n - 1 = 19$ будут загружены, а для 81 свободной клетки придется строить многоугольники, принимающие зачастую довольно сложные формы.

Указанный недостаток преодолевается с помощью *модифицированного распределительного метода*.

4.5. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Математическая постановка транспортной задачи.
2. Как определяется исходный план перевозок?
3. Что выражает потенциал пустой ячейки в таблице распределения?
4. Как строятся многоугольники замены?
5. Чем характеризуется открытый тип транспортной задачи?
6. По заданным преподавателем исходным данным составить оптимальный план перевозок с помощью распределительного метода.

5. ПОТОКИ В СЕТЯХ

5.1. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

Одной из важных задач теории графов является задача определения максимального потока, протекающего от некоторой вершины s (*источника*) графа $G=(X,U)$ к некоторой конечной вершине t (*стоку*). При этом каждой дуге $u(x_i, x_j)\in U$ приписана некоторая *пропускная способность* $c_{ij} > 0$, которая определяет наибольшее значение потока φ_{ij}, который может протекать по данной дуге.

Метод решения *задачи о максимальном потоке* ($s \to t$) был предложен Фордом и Фалкерсоном, и их техника расстановки пометок составляет основу других алгоритмов решения многочисленных задач, являющихся простыми обобщениями или расширениями указанной задачи.

Граф $G=(X,U)$, имеющий источник s, сток t и пропускные способности дуг c_{ij}, называется *сетью*.

Таким образом, значение потока φ_{ij} дуги $u(x_i, x_j)$ должно быть меньше или равно ее пропускной способности:

$$\varphi_{ij} \le c_{ij}, \quad \forall\, u(x_i, x_j) \in U. \tag{5.1}$$

Суммарный поток, входящий в вершину x_i, равен суммарному потоку, выходящему из нее (кроме вершин источника s и стока t):

$$\sum_{x_j \in \Gamma(x_i)} \varphi_{ij} = \sum_{x_k \in \Gamma^{-1}(x_i)} \varphi_{ki}, \qquad \forall\, x_i \ne s,t. \tag{5.2}$$

Для источника s и стока t значение потока совпадает по модулю и равно суммарному потоку в сети $\varphi(G)$:

$$\sum_{x_j \in \Gamma(s)} \varphi_{sj} = -\sum_{x_k \in \Gamma^{-1}(x_t)} \varphi_{kt} = \varphi(G). \tag{5.3}$$

Разрезом S/T называется разбиение вершин графа $G=(X,U)$ на два непересекающихся подмножества $S \cap T = \varnothing$, включающих исток и сток: $s \in S$ и $t \in T$.

Дуга принадлежит разрезу $u(x_i, x_j) \in S/T$, если ее концы принадлежат разным подмножествам. Например: $x_i \in S$, $x_j \in T$.

Дуга $u(x_i, x_j)$ является насыщенной, если значение потока, проходящего через эту дугу, равно ее пропускной способности: $\varphi_{ij} = c_{ij}$.

Пропускной способностью разреза S/T называется величина $\varphi(S/T)$, равная суммарной пропускной способности всех дуг, принадлежащих ему:

$$\varphi(S/T) = \sum_{u(x_i,x_j)\in S/T} c_{ij}. \tag{5.4}$$

Теорема Форда-Фалкерсона (о максимальном потоке). Величина максимального потока в сети ($s \to t$) равна пропускной способности его минимального разреза $\varphi(S/T)$:

$$\varphi_{max}(G)=\varphi_{min}(S/T). \tag{5.5}$$

5.2. АЛГОРИТМ РАССТАНОВКИ ПОМЕТОК

Алгоритм расстановки пометок, предложенный Фордом и Фалкерсоном, начинает работу с произвольного допустимого потока (обычно принимается $\varphi(G) = 0$), затем величина потока увеличивается с помощью систематического поиска всех возможных цепей потока от s к t. Поиск цепи осуществляется с помощью расстановки пометок в вершинах графа. При нахождении одной из таких цепей поток вдоль нее увеличивается до максимального значения, равного пропускной способности одной из дуг, принадлежащих этой цепи. Далее итерационный процесс повторяется до тех пор, пока можно найти хотя бы одну цепь, увеличивающую значение потока.

Вершина может находится в одном из трех состояний:

а) вершине приписана пометка и она просмотрена;

б) пометка приписана, но вершина не просмотрена;

в) вершина не помечена.

Пометка вершины x_i состоит из двух частей и имеет вид: $[x_j, \varphi(x_i)]$ или $[-x_k, \varphi(x_i)]$. Первая часть пометки x_j означает, что поток в вершину x_i может прийти из вершины x_j. Пометка $-x_k$ означает, что поток должен

уменьшаться. Вторая часть пометки $\varphi(x_i)$ означает величину потока, пришедшего в вершину x_i.

Вершина x_i считается просмотренной, если помечены все смежные с ней вершины $x_j \in \Gamma(x_i)$ и $x_k \in \Gamma^{-1}(x_i)$.

Работа алгоритма Форда-Фалкерсона осуществляется в два этапа [6].

Этап 1. Расстановка пометок.

В первой итерации считается, что поток в сети равен нулю: $\varphi(G) = 0$.

1. В источнике s находится резервуар бесконечной емкости $\varphi(s)=\infty$, следовательно, начальная вершина получает пометку $[s, \infty]$. Источник становится текущей вершиной: $p=s$.

2. Помечаются вершины, смежные с текущей:

а) каждая непомеченная вершина $x_j \in \Gamma(p)$, для которой выполняется условие $\varphi_{pj}<c_{pj}$ (дуга ненасыщена), получает пометку $[p, \varphi(x_j)]$, рассчитываемую по формуле:

$$\varphi(x_j)=\min\{\varphi(p), c_{pj} - \varphi_{pj}\}; \quad (5.6)$$

б) каждая непомеченная вершина $x_k \in \Gamma^{-1}(p)$, для которой выполняется условие $\varphi_{kp}>0$, получает пометку $[-p, \varphi(x_k)]$, рассчитываемую по формуле:

$$\varphi(x_k)=\min\{\varphi(p), \varphi_{kp}\}. \quad (5.7)$$

После расстановки пометок смежным вершинам текущая вершина p считается просмотренной.

3. Если текущая вершина является стоком ($p = t$), то следует конец итерации и переход к п.4. Если не все вершины просмотрены, переходим к следующей помеченной вершине $p = x_{i+1}$ и повторяем для нее п.2.

Если все помеченные вершины просмотрены, а вершина стока по-прежнему не имеет пометки, то – *конец алгоритма*.

Этап 2. Увеличение потока в сети.

После расстановки пометок определяется *аугментальная цепь*, по которой поток получает распространение от истока s к стоку t.

4. Назначение текущей вершины стоку $p= t$.

5. Изменение значения потока инцидентной дуги:

а) если пометка текущей вершины имеет вид $[x_i, \varphi(p)]$, то увеличить поток вдоль дуги $u(x_i, p)$ на величину $\varphi(t)$:

$$\varphi_{ip} := \varphi_{ip} + \varphi(t); \quad (5.8)$$

б) если пометка текущей вершины имеет вид $[-x_i, \varphi(p)]$, то уменьшить поток вдоль дуги $u(p, x_i)$ на величину $\varphi(t)$:

$$\varphi_{pi} := \varphi_{pi} - \varphi(t). \quad (5.9)$$

6. Переходим к предыдущей вершине $p = x_i$. Если вершина не является источником $p \neq s$, то повторить п.5. Иначе конец второго этапа. Пометки вершин стираются, переход к п.1.

В результате применения второго этапа алгоритма дуги, составляющие аугментальную цепь от источника до стока, получат приращение значения потока. Таким образом, суммарный поток сети будет увеличен на величину пометки вершины стока:

$$\varphi(G) := \varphi(G) + \varphi(t). \quad (5.10)$$

В результате последней итерации помеченные вершины образуют множество S, а непомеченные вершины – множество T минимального разреза $S/T(G)$. Этот разрез проходит через насыщенные дуги, суммарная пропускная способность которых, по теореме Форда-Фалкерсона, равна максимальной пропускной способности сети.

Задача решена.

5.3. ПРИМЕР

Рассмотрим граф, изображенный на рис. 5.1, и возьмем в качестве источника вершину x_1, а в качестве стока – вершину x_8. Пропускные способности дуг указаны на рисунке. Требуется найти максимальный поток от вершины x_1 к x_8.

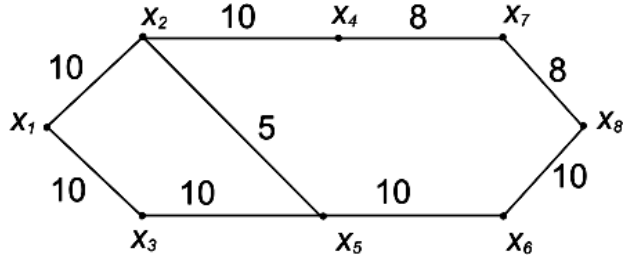


Рис. 5.1

Алгоритм работает следующим образом.

1. Первая итерация. Припишем вершине x_1 пометку $[x_1, \infty]$ и определим ее текущей.

2. Множество смежных вершин $\Gamma(x_1) = \{x_2, x_3\}$. В соответствии с выражением (5.6) вершина x_2 помечается пометкой $[x_1, \min\{\infty, 10 - 0\}]$ или $[x_1, 10]$. Вершине x_3 приписывается пометка $[x_1, \min\{\infty, 10 - 0\}]$ или $[x_1, 10]$. В результате вершина x_1 помечена и просмотрена, вершины x_2 и x_3 – помечены.

3. Переходим к следующей вершине $p = x_2$ и повторяем для нее п.2.

$\Gamma(x_2) = \{x_1, x_4, x_5\}$. Вершина x_1 уже просмотрена нами и помечена. Для вершины x_4 пометка будет $[x_2, \min\{10, 10 - 0\}]$ или $[x_2, 10]$. Вершина x_5 получает пометку $[x_2, \min\{10, 5 - 0\}]$ или $[x_2, 5]$.

Переходим к вершине x_3: $\Gamma(x_3) = \{x_1, x_5\}$. Вершина x_1 просмотрена, а вершина x_5 уже была помечена из вершины x_2.

Переходим к вершине x_4: $\Gamma(x_4) = \{x_2, x_7\}$. Вершина x_2 просмотрена. Для вершины x_7 пометка будет $[x_4, \min\{10, 8 - 0\}]$ или $[x_4, 8]$.

Переходим к вершине x_5: $\Gamma(x_5) = \{x_3, x_6\}$. Вершина x_3 просмотрена. Для вершины x_6 пометка будет $[x_5, \min\{5, 10 - 0\}]$ или $[x_5, 5]$.

Переходим к вершине x_6: $\Gamma(x_6) = \{x_5, x_8\}$. Вершина x_5 просмотрена. Для вершины x_8 пометка будет $[x_6, \min\{5, 10 - 0\}]$ или $[x_6, 5]$. Сток достигнут.

В результате применения первого этапа алгоритма, все вершины графа получают пометки, как показано на рис. 5.2. Поток в сети увеличился на 5 единиц: $\varphi(G) := 0 + 5 = 5$. Переходим ко второму этапу.

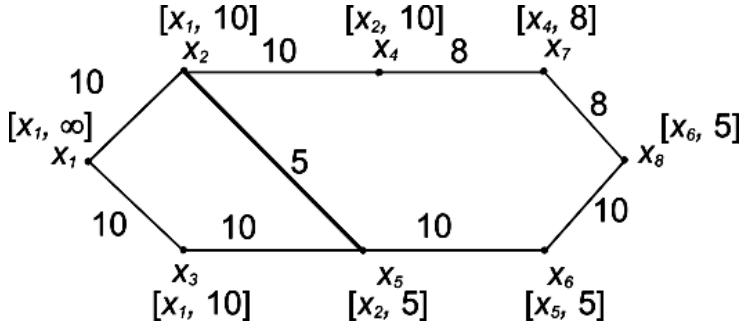


Рис. 5.2

4. Назначение текущей вершины стоку $p=x_8$.

5. Вершина x_8 имеет пометку $[x_6, 5]$, следовательно, поток пришел из вершины x_6. В соответствии с выражением (5.8), поток вдоль дуги $u(x_6, x_8)$ примет значение $\varphi_{6,8} := 0 + 5 = 5$.

6. Переходим к предыдущей вершине $p=x_6$. В вершину x_6 поток пришел из вершины x_5, поэтому поток вдоль дуги $u(x_5, x_6)$ примет значение $\varphi_{5,6} := 0 + 5 = 5$.

Аналогично, поток вдоль дуги $u(x_2, x_5)$ примет значение $\varphi_{2,5} := 0 + 5 = 5$. Так как величина потока вдоль этой дуги равна ее пропускной способности $\varphi_{2,5} = c_{2,5}$, дуга $u(x_2, x_5)$ станет насыщенной.

Наконец, поток вдоль дуги $u(x_1, x_2)$ примет значение $\varphi_{1,2} := 0 + 5 = 5$.

Аугментальная цепь определена: $\mu(s\text{-}t) = \{x_1, x_2, x_5, x_6, x_8\}$.

Полученные значения потока отметим на дугах аугментальной цепи в квадратных скобках. Граф примет вид, показанный на рис. 5.3.

Стрелками показано направление потока в аугментальной цепи.

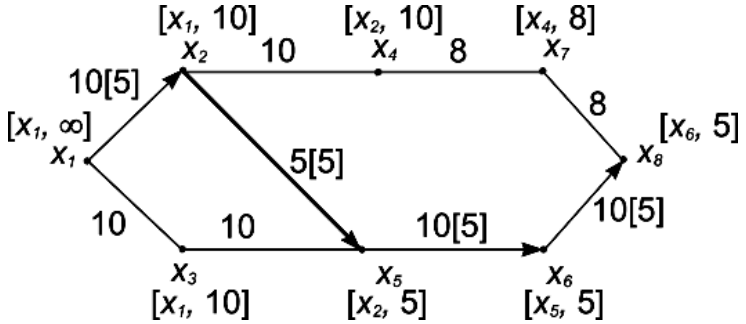


Рис. 5.3

Вторая итерация. Старые пометки вершин удаляются.

Просматриваем вершину x_1. Смежная с ней вершина x_2 получит пометку $[x_1, \min\{\infty, 10 - 5\}]$ или $[x_1, 5]$. Вершина x_3 получит пометку $[x_1, \min\{\infty, 10 - 0\}]$ или $[x_1, 10]$.

Просматриваем вершину x_2. Вершина x_4 получит пометку $[x_2, \min\{5, 10 - 0\}]$ или $[x_2, 5]$. Вершина x_5 является недостижимой, так как дуга $u(x_2, x_5)$ насыщена. Она пропускается.

Просматриваем вершину x_3. Вершина x_5 получит пометку $[x_3, \min\{10, 10 - 0\}]$ или $[x_3, 10]$.

Просматриваем вершину x_4. Вершина x_7 получит пометку $[x_4, \min\{5, 8 - 0\}]$ или $[x_4, 5]$.

В ходе просмотра x_5, вершина x_6 получит пометку $[x_5, \min\{10, 10 - 5\}]$ или $[x_5, 5]$.

Наконец, вершина x_8 будет помечена из вершины x_6: $[x_6, \min\{5, 10 - 5\}]$ или $[x_6, 5]$. Поток в сети увеличился еще на 5 единиц: $\varphi(G)$: = 5 + 5 = 10.

Второй этап. Принимаем текущую вершину за сток. Вершина x_8 была помечена из x_6, следовательно, поток по дуге $u(x_6, x_8)$ увеличится на $\varphi(t)$=5 и примет значение $\varphi_{6,8}$:= 5 + 5 = 10. Дуга $u(x_6, x_8)$ станет насыщенной, так как $\varphi_{6,8} = c_{6,8}$.

Инцидентная с ней дуга $u(x_5, x_6)$, входящая в аугментальную цепь, также станет насыщенной: $\varphi_{5,6} := \varphi_{5,6} + \varphi(t) = 5 + 5 = 10$.

Перемещаясь дальше к истоку, мы получаем аугментальную цепь, состоящую из вершин $\mu(s\text{-}t)=\{x_1, x_3, x_5, x_6, x_8\}$. Дуги $u(x_1, x_3)$ и $u(x_3, x_5)$ получают приращение потока $\varphi(t)$=5. Граф примет вид, показанный на рис. 5.4.

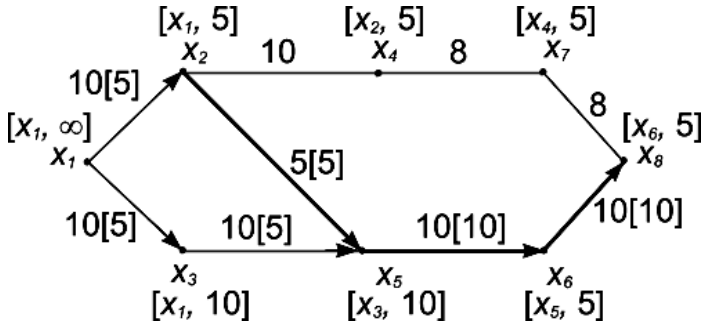


Рис. 5.4

Третья итерация. Старые пометки вершин удаляются.

Просматриваем вершину x_1. Вершина x_2 получит пометку $[x_1, \min\{\infty, 10 - 5\}]$ или $[x_1, 5]$. Вершина x_3 получит пометку $[x_1, \min\{\infty, 10 - 5\}]$ или $[x_1, 5]$.

Просматриваем вершину x_2. Вершина x_4 получит пометку $[x_2, \min\{5, 10 - 0\}]$ или $[x_2, 5]$.

Просматриваем вершину x_3. Вершина x_5 получит пометку $[x_3, \min\{5, 10 - 5\}]$ или $[x_3, 5]$.

Вершина x_7 получит пометку $[x_4, \min\{5, 8 - 0\}]$ из вершины x_4 или $[x_4, 5]$.

Вершина x_6 недостижима и не получает пометки.

Вершина x_8 получит пометку $[x_7, \min\{5, 8 - 0\}]$ из вершины x_7 или $[x_7, 5]$.

В результате первого этапа алгоритма получаем приращение потока на величину $\varphi(t)$=5.

Суммарный поток в сети становится равным 15:

$\varphi(G) := \varphi(G) + \varphi(t) = 10 + 5 = 15$.

В ходе второго этапа дуга $u(x_1, x_2)$ становится насыщенной и определяется очередная аугментальная цепь $\mu(s\text{-}t) = \{x_1, x_2, x_4, x_7, x_8\}$.

Граф примет вид, показанный на рис. 5.5.

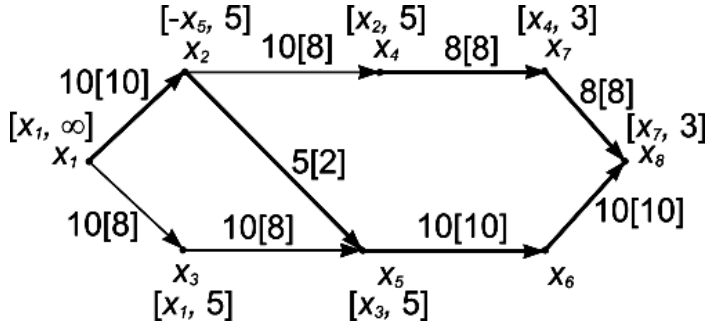


Рис. 5.5

Четвертая итерация. Старые пометки вершин удаляются.

Просматриваем вершину x_1. Вершина x_2 недостижима и пропускается. Вершина x_3 получает пометку $[x_1, \min\{\infty, 10 - 5\}]$ или $[x_1, 5]$.

Просматриваем вершину x_3. Вершина x_5 получит пометку $[x_3, \min\{5, 10 - 5\}]$ или $[x_3, 5]$.

Вершина x_4 недоступна, пропускаем ее. Просматриваем вершину x_5. Она имеет две смежные вершины x_2 и x_6. Вершина x_6 недоступна, а из вершины x_2 приходит поток мощностью $\varphi_{2,5} = 5$.

В соответствии с (5.7), вершина x_2 получает пометку $[-x_5, \min\{5, 5\}]$ или $[-x_5, 5]$.

Далее, вершина x_4 получает пометку $[x_2, \min\{5, 10 - 5\}]$ или $[x_2, 5]$.

Вершина x_7 получает пометку $[x_4, \min\{5, 8 - 5\}]$ или $[x_4, 3]$.

Вершина x_8 получает пометку $[x_7, \min\{5, 8 - 5\}]$ или $[x_7, 3]$.

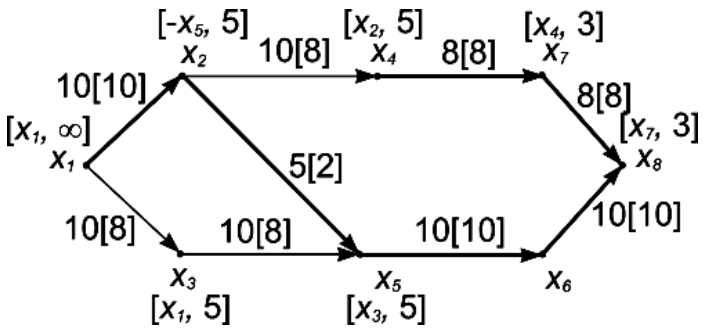


Рис. 5.6

Суммарный поток увеличивается еще на 3 единицы (рис. 5.6).

Очередная аугментальная цепь состоит из вершин $\mu(s\text{-}t)=\{x_1, x_3, x_5, x_2, x_4, x_7, x_8\}$. Из второго этапа находим, что поток вдоль дуги $u(x_2, x_5)$ будет уменьшен на 3 и составит, в соответствии с (5.9), $\varphi_{2,5} = 5 - 3 = 2$.

Последняя итерация.

Вершина x_3 получает пометку $[x_1, \min\{\infty, 10 - 8\}]$ или $[x_1, 2]$.

Вершина x_5 получает пометку $[x_3, \min\{2, 10 - 8\}]$ или $[x_3, 2]$.

Из вершины x_5 помечается вершина x_2: $[-x_5, \min\{2, 2\}]$ или $[-x_5, 2]$.

Из вершины x_2 помечается вершина x_4: $[x_2, \min\{2, 10 - 8\}]$ или $[x_2, 2]$.

Остальные вершины недостижимы. Конец алгоритма.

В результате последней итерации помеченные вершины образуют множество $S = \{x_1, x_2, x_3, x_4, x_5\}$, а непомеченные – множество $T = \{x_6, x_7, x_8\}$ минимального разреза графа $S/T(G)$.

Пропускная способность разреза равна суммарной пропускной способности насыщенных дуг $u(x_4, x_7)$ и $u(x_5, x_6)$, принадлежащих ему, и равна пропускной способности сети: $\varphi(S/T)=18$ (рис. 5.7).

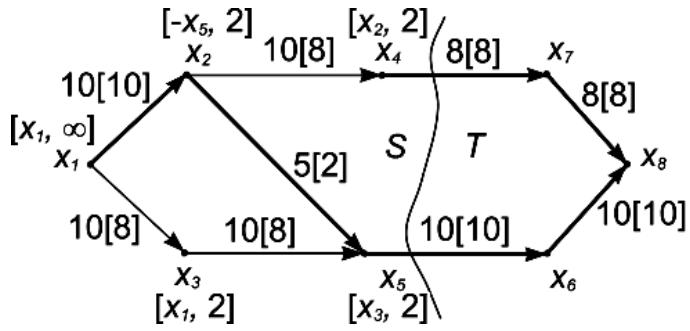


Рис. 5.7

5.4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что называется сетью? Разрезом? Как определяется пропускная способность разреза?
2. Сформулируйте теорему Форда-Фалкерсона.
3. Какая дуга называется насыщенной?
4. Как определяется значение потока на дугах графа?
5. По заданным преподавателем исходным данным найти максимальную пропускную способность сети с помощью алгоритма расстановки пометок.

6. ОСТОВЫЕ ДЕРЕВЬЯ

6.1. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

Одним из наиболее важных понятий теории графов является *дерево*.

Неориентированным деревом называется связанный граф, не имеющий циклов.

Остовым подграфом (*суграфом*) графа G является подграф G_p, содержащий все вершины исходного графа.

Если $G=(X,U)$ - неориентированный граф с n вершинами, то связный суграф G_p, не имеющий циклов, называется *остовым деревом* (*остовом*) *графа* G.

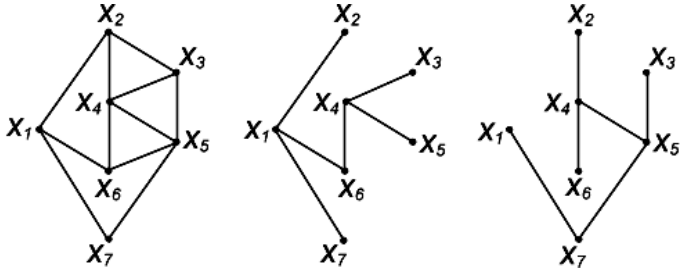
Для остового дерева справедливо соотношение:

$$G_p=(X_p,U_p) \subseteq G, \quad \text{где } X_p = X,\ U_p \subseteq U\ . \qquad (6.1)$$

Легко доказать, что остовое дерево имеет следующие свойства:

1) остовое дерево графа с n вершинами имеет $n-1$ ребро ($|X_p|=|U_p|-1$);
2) существует единственный путь, соединяющий любые две вершины остова графа:
   $\forall x_i, x_j \in X_p\ (i \neq j) \rightarrow \exists!\ \mu(x_i, x_j)$.

Например, если G - граф, показанный на рис. 6.1,а, то графы на рис. 6.1,б,в являются остовами графа G. Из сформулированных выше определений вытекает, что остов графа G можно рассматривать как минимальный связанный остовый подграф графа G.



а б в

Рис. 6.1

Понятие дерева как математического объекта было впервые предложено Кирхгофом в связи с определением фундаментальных циклов, применяемых при анализе электрических цепей. Приблизительно десятью годами позже Кэли вновь (независимо от Кирхгофа) ввел понятие дерева и получил большую часть первых результатов в области исследования свойств деревьев. Большую известность получила его знаменитая теорема.

Теорема Кэли. На графе с n вершинами можно построить n^{n-2} остовых деревьев.

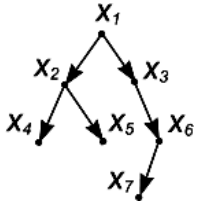


Рис. 6.2

Ориентированное дерево представляет собой ориентированный граф без циклов, в котором полустепень захода каждой вершины, за исключением одной (вершины r), равна единице, а полустепень захода вершины r (называемой корнем этого дерева) равна нулю.

На рис. 6.2 показан граф, который является ориентированным деревом с корнем в вершине x_1. Из приведенного определения следует, что ориентированное

дерево с n вершинами имеет $n-1$ дуг и связно.

Неориентированное дерево можно преобразовать в ориентированное: надо взять его произвольную вершину в качестве корня и ребрам приписать такую ориентацию, чтобы каждая вершина соединялась с корнем (только одной) простой цепью.

"Генеалогическое дерево", в котором вершины соответствуют лицам мужского пола, а дуги ориентированы от родителей к детям, представляет собой хорошо известный пример ориентированного дерева. Корень в этом дереве соответствует "основателю" рода (лицу, родившемуся раньше остальных).

6.2. КРАТЧАЙШИЙ ОСТОВ ГРАФА

В учебном пособии рассматривается метод прямого построения *кратчайших остовых деревьев* во взвешенном графе (в котором веса приписаны дугам). Кратчайшее остовое дерево графа находит применение при прокладке дорог (газопроводов, линий электропередач и т. д.), когда необходимо связать n точек некоторой сетью так, чтобы общая длина "линий связи" была минимальной. Если точки лежат на евклидовой плоскости, то их можно считать вершинами полного графа G с весами дуг, равными соответствующим "прямолинейным" расстояниям между концевыми точками дуг. Если "разветвление" дорог допускается только в заданных n точках, кратчайшее остовое дерево графа G будет как раз требуемой сетью дорог, имеющей наименьший вес.

Рассмотрим взвешенный связный неориентированный граф $G=(X,U)$; вес ребра (x_i,x_j) обозначим c_{ij}. Из большого числа остовов графа нужно найти один, у которого сумма весов ребер наименьшая. Такая задача возникает, например, в том случае, когда вершины являются клеммами электрической сети, которые должны быть соединены друг с другом с помощью проводов наименьшей общей длины (для уменьшения уровня наводок). Другой пример: вершины представляют города, которые нужно связать сетью трубопроводов; тогда наименьшая общая длина труб, которая должна быть использована для строительства (при условии, что вне городов "разветвления" трубопроводов не допускаются), определяется кратчайшим остовом соответствующего графа.

Задача построения кратчайшего остова графа является одной из немногих задач теории графов, которые можно считать полностью решенными.

6.3. АЛГОРИТМ ПРИМА-КРАСКАЛА

Этот алгоритм порождает остовое дерево посредством разрастания только одного поддерева, например X_p, содержащего больше одной вершины. Поддерево постепенно разрастается за счет присоединения ребер (x_i,x_j), где $x_i \in X_p$ и $x_j \notin X_p$; причем добавляемое ребро должно иметь

наименьший вес c_{ij}. Процесс продолжается до тех пор, пока число ребер в U_p не станет равным $n-1$. Тогда поддерево $G_p=(X_p,U_p)$ будет требуемым остовым деревом. Впервые такая операция была предложена Примом и Краскалом (с разницей в способе построения дерева), поэтому данный алгоритм получил название Прима-Краскала.

Алгоритм начинает работу с включения в поддерево начальной вершины. Поскольку остовое дерево включает все вершины графа G, то выбор начальной вершины не имеет принципиального значения. Будем каждой очередной вершине присваивать пометку $\beta(x_i)=1$, если вершина x_i принадлежит поддереву X_p, и $\beta(x_j)=0$ – в противном случае.

Алгоритм имеет вид:

Шаг 1. Пусть $X_p=\{x_1\}$, где x_1 - начальная вершина, и $U_p=\varnothing$ (U_p является множеством ребер, входящих в остовое дерево). Вершине x_1 – присвоить пометку $\beta(x_1)=1$. Каждой вершине $x_i \notin X_p$ присвоить $\beta(x_i)=0$.

Шаг 2. Из всех вершин $x_j \in \Gamma(X_p)$, для которых $\beta(x_j)=0$, найти вершину x_j^* такую, что

$$c(x_i, x_j^*) = \min_{x_j \in \Gamma(X_p)} \{c(x_i, x_j)\}, \quad \text{где } x_i \in X_p \text{ и } x_j \notin X_p. \qquad (6.2)$$

Шаг 3. Обновить данные: $X_p = X_p \cup \{x_j^*\}$; $U_p = U_p \cup u\{x_i, x_j^*\}$. Присвоить $\beta(x_j^*)=1$.

Шаг 4. Если $|X_p|=n$, то *конец*. Ребра в U_p образуют кратчайший остов графа. Если $|X_p|<n$, то перейти к шагу 2.

6.4. ПРИМЕР

Для примера рассмотрим граф, изображенный на рис. 6.3. Найдем для него кратчайшее остовое дерево, используя для этой цели рассмотренный выше алгоритм Прима-Краскала. Обозначим множество смежных вершин, не входящих в порожденное поддерево, как $\Gamma^*(X_p)$. Таким образом, для всех вершин, входящих в это множество, оценка $\beta(x_j)=0$, $\forall x_j \in \Gamma^*(X_p)$. Вектор B представляет собой множество оценок $\beta(x_i)$ для всех вершин графа G: $\forall x_i \in X$. Длина порожденного поддерева обозначена как L.

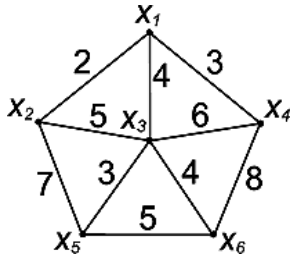


Рис. 6.3

Итерация 1: $X_p=\{x_1\}$; $A_p=\varnothing$;
$\Gamma^*(X_p)=\{x_2, x_3, x_4\}$; $c(x_1, x_2^*)=2$;
$B=\{1,1,0,0,0,0\}$; $L=2$.

Итерация 2: $X_p=\{x_1, x_2\}$; $A_p=\{(x_1, x_2)\}$;
$\Gamma^*(X_p)=\{x_3, x_4, x_5\}$; $c(x_1,x_4^*)=3$;
$B=\{1,1,0,1,0,0\}$; $L=2+3=5$.

Итерация 3: $X_p=\{x_1, x_2, x_4\}$;
$A_p=\{(x_1, x_2); (x_1, x_4)\}$; $\Gamma^*(X_p)=\{x_3, x_5, x_6\}$;
$c(x_1,x_3^*)=4$; $B=\{1,1,1,1,0,0\}$; $L=5+4=9$.

Итерация 4: $X_p=\{x_1, x_2, x_3, x_4\}$;
$A_p=\{(x_1, x_2); (x_1, x_4); (x_1,x_3)\}$; $\Gamma^*(X_p)=\{x_5, x_6\}$; $c(x_3, x_5^*)=2$;
$B=\{1,1,1,1,1,0\}$; $L=9+3=12$.

Итерация 5: $X_p=\{x_1, x_2, x_3, x_4,x_5\}$; $A_p=\{(x_1, x_2); (x_1, x_4); (x_1,x_3); (x_3,x_5)\}$;
$\Gamma^*(X_p)=\{x_6\}$; $c(x_3,x_6^*)=4$; $B=\{1,1,1,1,1,1\}$; $L=12+4=16$.

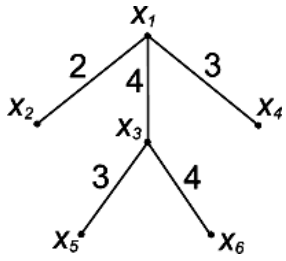


Рис. 6.4

Задача решена. Полученные множества вершин X_p и ребер A_p составляют кратчайшее остовое дерево:

$X_p=\{x_1, x_2, x_3, x_4,x_5,x_6\}$;

$A_p=\{(x_1, x_2); (x_1, x_4); (x_1,x_3); (x_3,x_5); (x_3,x_6)\}$.

Суммарная длина кратчайшего остового дерева $L=16$.

Результат решения задачи представлен на рис. 6.4.

6.5. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определение дерева; ориентированного дерева.
2. Какое дерево называется остовым?
3. Свойства остовых деревьев. Теорема Кэли.
4. Что называется корнем дерева?
5. Как преобразовать неориентированное дерево в ориентированное?
6. Сколько ребер содержит остовое дерево графа?
7. По заданному преподавателем графу определить кратчайший остов графа с помощью алгоритма Прима-Краскала.

7. РАСКРАСКИ

7.1. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

Разнообразные задачи, возникающие при планировании производства, составлении графиков осмотра, хранении и транспортировке товаров и т.д., могут быть представлены часто как задачи теории графов, тесно связанные с так называемой "задачей раскраски". Графы, рассматриваемые в данной лабораторной работе, являются неориентированными и не имеют петель.

Граф G называют *r-хроматическим*, если его вершины могут быть *раскрашены* с использованием r цветов (красок) так, что не найдется двух смежных вершин одного цвета. Наименьшее число r, такое, что граф G является *r-хроматическим*, называется *хроматическим числом* графа G и обозначается $\gamma(G)$. Задача нахождения хроматического числа графа называется *задачей о раскраске* (или *задачей раскраски*) графа. Соответствующая этому числу раскраска вершин разбивает множество вершин графа на r подмножеств, каждое из которых содержит вершины одного цвета. Эти множества являются независимыми, поскольку в пределах одного множества нет двух смежных вершин.

Задача нахождения хроматического числа произвольного графа явилась предметом многих исследований в конце XIX и в XX столетиях. По этому вопросу получено много интересных результатов.

Хроматическое число графа нельзя найти, зная только числа вершин и ребер графа. Недостаточно также знать степень каждой вершины, чтобы вычислить хроматическое число графа. При известных величинах n (число вершин), m (число ребер) и $deg(x_1),...,deg(x_n)$ (степени вершин графа) можно получить только верхнюю и нижнюю оценки для хроматического числа графа.

Пример раскраски графа приведен на рис. 7.1. Цифрами 1, 2 и 3 обозначены цвета вершин.

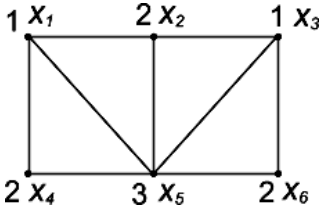


Рис. 7.1

Максимальное число независимых вершин графа $\varepsilon_0(G)$, равное мощности наибольшего множества попарно несмежных вершин, совпадает также с мощностью наибольшего множества вершин в G, которые могут быть окрашены в один цвет, следовательно:

$$\gamma(G) \geq \left\lceil \frac{n}{\varepsilon_0(G)} \right\rceil, \qquad (7.1)$$

где n - число вершин графа G, а $\lceil x \rceil$ обозначает наибольшее целое число, не превосходящее x.

Еще одна нижняя оценка для $\gamma(G)$ может быть получена следующим образом:

$$\gamma(G) \geq \frac{n^2}{n^2 - 2m}. \qquad (7.2)$$

Верхняя оценка хроматического числа может быть вычислена по формуле:

$$\gamma(G) \leq 1 + \max_{x_i \in X} \left[d(x_i) + 1 \right]. \qquad (7.3)$$

Применение оценок для хроматического числа значительно сужает границы решения. Для определения оценки хроматического числа также могут использоваться другие топологические характеристики графа, например свойство *планарности*.

Теорема о пяти красках. Каждый планарный граф можно раскрасить с помощью пяти цветов так, что любые две смежные вершины будут окрашены в разные цвета, т. е. если граф G - планарный, то $\gamma(G) \leq 5$.

Гипотеза о четырех красках (*недоказанная*). Каждый планарный граф можно раскрасить с помощью четырех цветов так, что любые две смежные вершины будут окрашены в разные цвета, т. е. $\gamma(G) \leq 4$, если граф G - планарный.

В 1852 г. о гипотезе четырех красок говорилось в переписке Огюста де Моргана с сэром Вильямом Гамильтоном. С тех пор эта "теорема" стала, наряду с теоремой Ферма, одной из самых знаменитых нерешенных задач в математике.

Полный граф G_n всегда раскрашивается в n цветов, равных количеству его вершин.

7.2. ЭВРИСТИЧЕСКИЕ АЛГОРИТМЫ РАСКРАШИВАНИЯ

Точные методы раскраски графа сложны для программной реализации. Однако существует много эвристических процедур раскрашивания, позволяющих находить хорошие приближения для определения хроматического числа графа. Такие процедуры также могут с успехом использоваться при раскраске графов с большим числом вершин, где применение точных методов неоправданно ввиду высокой трудоемкости вычислений.

Из эвристических процедур раскраски следует отметить последовательные методы, основанные на упорядочивании множества вершин [1].

В одном из простейших методов вершины вначале располагаются в порядке убывания их степеней. Первая вершина окрашивается в цвет 1; затем список вершин просматривается по убыванию степеней и в цвет 1 окрашивается каждая вершина, которая не является смежной с вершинами, окрашенными в тот же цвет. Потом возвращаемся к первой в списке неокрашенной вершине, окрашиваем ее в цвет 2 и снова просматриваем список вершин сверху вниз, окрашивая в цвет 2 любую неокрашенную вершину, которая не соединена ребром с другой, уже окрашенной в цвет 2 вершиной. Аналогично действуем с цветами 3, 4 и т. д., пока не будут окрашены все вершины. Число использованных цветов будет тогда приближенным значением хроматического числа графа.

Эвристический алгоритм раскраски вершин графа имеет следующий вид.

Шаг 1. Сортировать вершины графа по степеням убывания:
$deg(x_i) \geq deg(x_j)$, $\forall x_i, x_j \in G$; установить текущий цвет $p=1; i=1$.

Шаг 2. Выбрать очередную нераскрашенную вершину из списка и назначить ей новый цвет: $col(x_i)=p$; $X_p=\{x_i\}$.

Шаг 3. $i=i+1$. Выбрать очередную нераскрашенную вершину x_i и проверить условие смежности: $x_i \bigcap \Gamma(X_p) = \varnothing$, где X_p – множество вершин, уже раскрашенных в цвет p. Если вершина x_i не является смежной с данными вершинами, то также присвоить ей цвет p: $col(x_i)=p$.

Шаг 4. Повторить шаг 3, пока не будет достигнут конец списка ($i=n$).

Шаг 5. Если все вершины графа раскрашены, то – *конец алгоритма*. Иначе $p=p+1; i=1$. Повторить шаг 2.

7.3. ПРИМЕР

Раскрасим граф G, изображенный на рис. 7.2. Промежуточные данные для решения задачи будем записывать в табл. 7.1. Отсортируем вершины графа по убыванию их степеней. В результате получается вектор отсортированных вершин $X^*=\{x_1, x_5, x_6, x_4, x_2, x_3\}$.

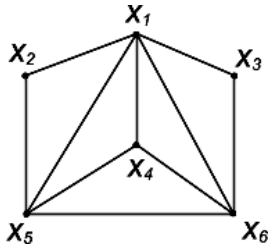


Рис. 7.2

Соответствующие данным вершинам степени образуют второй вектор:

$D=\{5, 4, 4, 3, 2, 2\}$. В первой строке таблицы запишем вектор X^*; во второй – вектор D. Последующие строки отражают содержание вектора раскраски $col(X^*)$.

Таблица 7.1

Номера вершин X^*	x_1	x_5	x_6	x_4	x_2	x_3
Степени вершин D	5	4	4	3	2	2
$p=1$	1	-	-	-	-	-
$p=2$	1	2	-	-	-	2
$p=3$	1	2	3	-	3	2
$p=4$	1	2	3	4	3	2

Таким образом, данный граф можно раскрасить не менее чем в четыре цвета, т.е. $\gamma(G)=4$.

7.4. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Сформулируйте задачу раскраски графа.
2. Какой граф называется r-хроматическим?
3. Что называется хроматическим числом графа?
4. Нижняя и верхняя оценка хроматического числа.
5. Во сколько цветов можно раскрасить планарный граф?
6. Во сколько цветов можно раскрасить полный граф?
7. Эвристический алгоритм раскраски графа.
8. Произвести раскраску графа, заданного преподавателем.

8. УСТОЙЧИВОСТЬ, ПОКРЫТИЯ, ПАРОСОЧЕТАНИЯ

8.1. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

В любом графе можно выделить совокупность некоторых множеств, объединенных по какому-то признаку, например непересекающихся подмножеств вершин. Аналогично граф можно разбить на подмножества ребер таким образом, чтобы ребра одного подмножества были попарно не смежны.

Можно ввести два инварианта графа для попарно несмежных вершин и попарно несмежных ребер.

Инвариантом графа G называется число, связанное с G, которое принимает одно и то же значение на любом графе, изоморфном G.

Два графа *изоморфны* ($G \cong H$), если между их множествами вершин существует взаимно однозначное соответствие, сохраняющее смежность.

Считается, что ребро $u(x_i, x_j)$ графа G *покрывает вершины* x_i и x_j. Аналогично можно рассматривать каждую вершину как *покрытие* всех инцидентных с ней *ребер*. С этой точки зрения определяются два инварианта графа G: минимальное число вершин, которые покрывают все ребра, и минимальное число ребер, покрывающих все вершины. Два других инварианта – это наибольшее число несмежных ребер и наибольшее число несмежных вершин. Эти четыре числа, связанные с произвольным графом, удовлетворяют некоторым соотношениям, и их рассмотрение приводит к изучению особых вершин и ребер, называемых критическими.

Ребро и вершина *покрывают* друг друга, если они инцидентны.

Множество вершин P графа $G=(X,U)$ называется *вершинным покрытием*, если оно покрывает все ребра графа.

Множество ребер Q графа $G=(X,U)$, покрывающих все вершины, называется *реберным покрытием* графа.

Минимальная мощность вершинного покрытия называется *числом вершинного покрытия* графа:

$$\pi_0(G) = \min |P|. \qquad (8.1)$$

Минимальная мощность реберного покрытия называется *числом реберного покрытия* графа:

$$\pi_1(G) = \min |Q|. \qquad (8.2)$$

Если учесть, что любая вершина графа покрывает сама себя и две смежные вершины покрывают друг друга, то такое множество вершин S,

покрывающее все вершины графа $G=(X,U)$, называется *внешне устойчивым* (*доминирующим*):

$$S \subset X, \quad S \cup \Gamma(S) = X. \qquad (8.3)$$

Минимальная мощность доминирующего множества вершин называется *вершинным числом внешней устойчивости* (*вершинного доминирования*) графа:

$$\beta_0(G) = \min |S|. \qquad (8.4)$$

Аналогично, если любое ребро графа покрывает себя и два инцидентных ребра покрывают друг друга, то такое множество ребер R, покрывающее все ребра графа $G=(X,U)$, называется *внешне устойчивым* (*доминирующим*). Соответственно, минимальная мощность множества ребер, покрывающих все ребра графа, называется *реберным числом внешней устойчивости* (*реберного доминирования*) графа:

$$\beta_1(G) = \min |R|. \qquad (8.5)$$

Множество вершин H графа $G=(X,U)$ называется *внутренне устойчивым* (*независимым*), если все его вершины попарно не смежны:

$$H \subset X, \quad H \cap \Gamma(H) = \varnothing. \qquad (8.6)$$

Внутренне устойчивое множество вершин называется *пустым подграфом*, если при добавлении хотя бы одной вершины, не принадлежащей этому множеству, образуется хотя бы одно ребро (дуга).

Максимальная мощность независимого множества вершин называется *вершинным числом внутренней устойчивости* (*вершинной независимости*) графа:

$$\varepsilon_0(G) = \max |H| \quad . \qquad (8.7)$$

Множество ребер W графа $G=(X,U)$ называется *внутренне устойчивым* (*независимым*), если все его ребра попарно не инцидентны. Соответственно, максимальное число попарно не инцидентных ребер графа $G=(X,U)$ называется *реберным числом внутренней устойчивости* (*реберной независимости*) графа:

$$\varepsilon_1(G) = \max |W| \quad . \qquad (8.8)$$

Граф с p вершинами и q ребрами называется (p, q)–графом.

$(1, 0)$–граф называется *тривиальным*.

Теорема. Для любого нетривиального графа $G=(X,U)$

$$\varepsilon_0(G) + \pi_0(G) = \varepsilon_1(G) + \pi_1(G) = |X|. \qquad (8.9)$$

Вычисление рассмотренных инвариантов графа требуется при решении многих практических задач.

Пример. Вычислим инварианты для графа Петерсена (рис. 8.1).

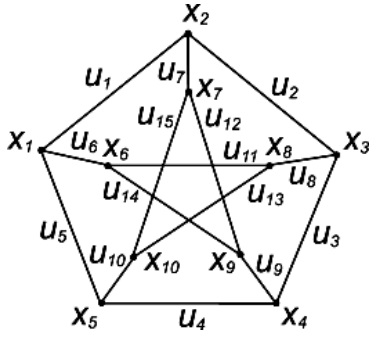


Рис. 8.1

$\varepsilon_0(G) = |\{x1, x3, x9, x10\}| = 4;$

$\varepsilon_1(G) = |\{u1, u4, u8, u14, u15\}| = 5;$

$\pi_0(G) = |\{x1, x3, x5, x7, x8, x9\}| = 6;$

$\pi_1(G) = |\{u6, u7, u8, u9, u10\}| = 5;$

$\beta_0(G) = |\{x1, x4, x10\}| = 3;$

$\beta_1(G) = |\{u1, u3, u13, u14\}| = 4.$

Множество ребер графа, в котором никакая пара ребер не смежна, называется *паросочетанием* графа.

Множество ребер паросочетания, в котором число ребер равно $\varepsilon_1(G)$, называется *наибольшим паросочетанием* графа.

Для двудольных графов справедлива следующая теорема о паросочетании.

Теорема Кенига. Для двудольного графа $K_{n,m}$ число ребер в наибольшем паросочетании равно числу вершинного покрытия:

$$\varepsilon_1(G) = \pi_0(G). \qquad (8.10)$$

Задача о восьми ферзях. Можно ли на шахматной доске расставить восемь ферзей так, чтобы ни один из них не находился под ударом какого-либо другого? Эта известная задача сводится к нахождению *наибольшего внутренне устойчивого множества* в симметрическом графе с 64 вершинами. Всего существует 92 решения.

Задача о часовых. В тюрьме города N около каждой камеры должен быть поставлен часовой. Однако часовой, стоящий у камеры x_0, видит также, что происходит в камерах x_1, x_2, x_3, x_4 (по коридорам). Какое наименьшее количество часовых требуется для наблюдения за всеми камерами? Необходимо найти число $\beta_0(G)$ *внешней устойчивости* графа.

Задача о пяти ферзях. Сколько ферзей достаточно расставить на шахматной доске так, чтобы каждая клетка доски находилась под ударом хотя бы одного из них? Эта задача сводится к отысканию *наименьшего внешне устойчивого множества* в графе с 64 вершинами, у которого дуга $u(x_i, x_j) \in U$ тогда и только тогда, когда вершины x_i и x_j расположены на одной и той же горизонтали, вертикали или диагонали. Число внешней устойчивости $\beta_0(G)=5$ для ферзей, $\beta_0(G)=8$ для ладей, $\beta_0(G)=12$ для коней, $\beta_0(G)=8$ для слонов.

8.2. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что называется инвариантом графа?
2. Дайте определение вершинного (реберного) покрытия.
3. Как определяется число вершинного (реберного) покрытия?
4. Как определяется вершинное (реберное) число внешней устойчивости графа?
5. Какое множество вершин графа является внутренне устойчивым?
6. Как определяется вершинное (реберное) число внутренней устойчивости графа?
7. Что называется паросочетанием графа?
8. По заданному преподавателем графу определите его числовые характеристики.

9. ПЛАНАРНЫЕ И ПЛОСКИЕ ГРАФЫ

9.1. ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

Топология исследует свойства графов, инвариантные относительно гомеоморфных преобразований.

Два графа ***гомеоморфны***, если они ***изоморфны*** с точностью до вершин степени 2. Другими словами, два графа гомеоморфны, если они преобразуются до графов, изоморфных друг другу, заменой некоторых ребер цепями соответствующей длины.

Граф $G=(X, U)$ называется ***плоским***, если его ребра, расположенные на плоскости, могут иметь общие точки только в инцидентных им вершинах, т.е. не пересекаются (рис. 9.1).

Граф, изоморфный плоскому и расположенный на плоскости с пересечением ребер, называется ***планарным*** (рис. 9.2).

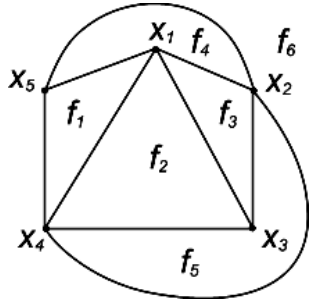


Рис. 9.1

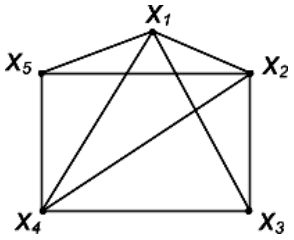


Рис. 9.2

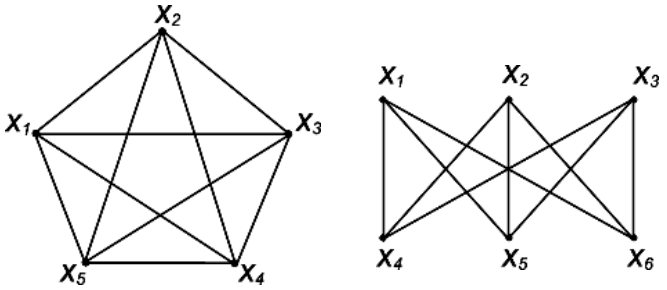
Область плоскости, ограниченная ребрами плоского графа, внутри которой нет ни вершин, ни ребер, называют ***гранью***. Ребра грани образуют простой цикл. Заметим, что плоский граф имеет всегда одну бесконечную грань, не ограниченную ребрами (f_6 на рис. 9.1).

Существует формула Эйлера, позволяющая установить связь между числом вершин n, ребер m и граней f плоского графа:

$$n - m + f = 2. \qquad (9.1)$$

Эту формулу Эйлер получил в 1736 г., и затем к топологическим аспектам теории графов не возвращались в течение почти двух столетий. В 1927 г. Л. Понтрягин доказал (но не опубликовал) критерий планарности, который независимо от него был открыт и опубликован в 1930 г. польским математиком К. Куратовским.

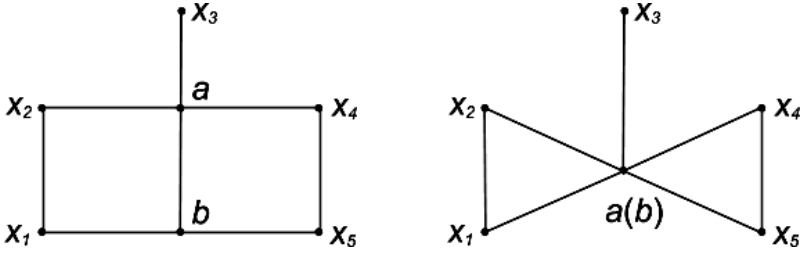
Теорема Понтрягина – Куратовского. Граф планарен тогда и только тогда, когда он не содержит подграфов, гомеоморфных полному графу K_5 и полному двудольному графу $K_{3,3}$ (рис. 9.3,а,б).



а б

Рис. 9.3

Основываясь на критерии Понтрягина, можно получить еще один критерий планарности, введя понятие ***элементарного стягивания***, состоящего в следующем. При стягивании любого ребра графа оно исключается, а обе вершины a и b, инцидентные ему, отождествляются (рис. 9.3,а,б). Полученная при этом вершина инцидентна тем же ребрам, что и a, b (кроме исключенного ребра).



а б

Рис. 9.4

Теорема Харари. Граф планарен тогда и только тогда, когда он не содержит подграфов, стягиваемых к K_5 и $K_{3,3}$.

Теорема Харари является двойственной формой теоремы Понтрягина-Куратовского.

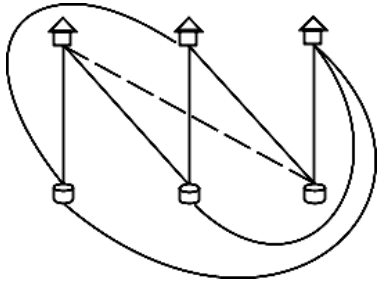
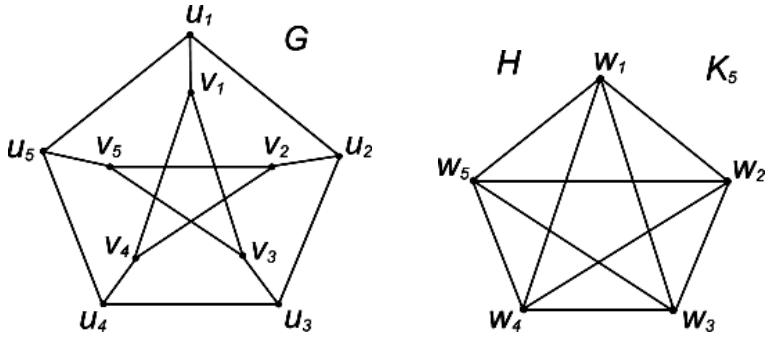


Рис. 9.5

Задача о трех домах и трех колодцах. Можно ли проложить от каждого дома к каждому колодцу тропинки так, чтобы никакие две из них не пересекались? Всегда можно нарисовать 8 линий, а девятая обязательно пересечет хотя бы одну из этих вершин. На рис. 9.5 представлен граф $K_{3,3}$.

Рассмотрим стягивание графа Петерсена (рис. 9.6, а). Граф G называется стягиваемым к графу H, если H можно получить из G с помощью некоторой последовательности элементарных стягиваний. Так, граф Петерсена стягивается к K_5 в результате стягивания в новую вершину w_i любого из пяти ребер (u_i, v_i), соединяющих пятиугольник с пентаграммой (рис. 9.6, б).



а б

Рис. 9.6

Поскольку каждая вершина графа Петерсена имеет степень $d(x)=3$, у него нет подграфов, гомеоморфных K_5. На рис. 9.7 показан один из его подграфов, гомеоморфных $K_{3,3}$.

Минимальное число ребер, которое нужно удалить из графа, чтобы он стал планарным, называется *числом планарности* и обозначается $\Theta(G)$.

Для полного графа K_n с $n \geq 4$ вершин $\Theta(G)$ определяется как

$$\Theta(K_n) = \frac{(n-3)(n-4)}{2}. \qquad (9.2)$$

Например, для графа K_5, $\Theta(K_5) = 1$.

Толщиной графа G называется наименьшее число планарных графов, в результате объединения которых получается исходный граф G. Толщина планарного графа равна 1.

Нижняя оценка толщины $t(G)$ определяется неравенством

$$t(G) \geq 1 + \left] \frac{\sum_{i=1}^{n} s_i}{6(n-2)} \right[, \qquad (9.3)$$

где] [– большее целое число, $|X| = n$, S_i – степень i-й вершины.

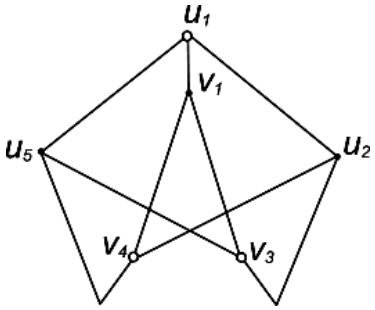


Рис. 9.7

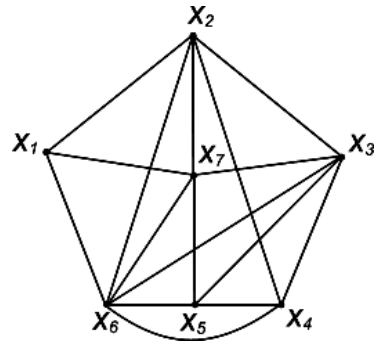
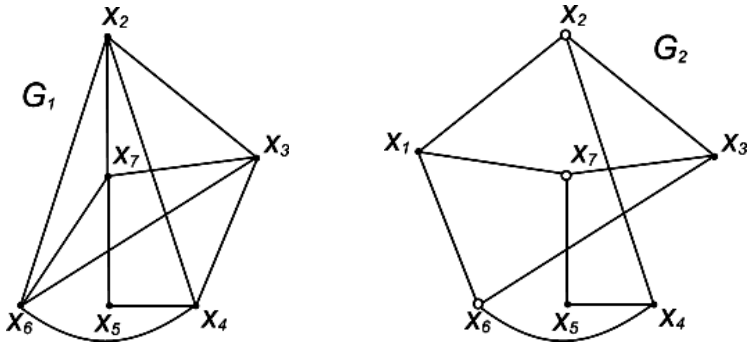


Рис. 9.8

Рассмотрим граф G на рис. 9.8. Определим, является ли он планарным, и если нет, то выясним, сколько в нем слоев и какие ребра следует удалить, чтобы граф стал планарным.

Согласно критерию Понтрягина, этот граф является непланарным, поскольку имеет подграфы G_1 и G_2, гомеоморфные K_5 и $K_{3,3}$ (рис. 9.9,а,б). Толщина графа G не меньше двух:

$$t(G) \geq 1 + \left] \frac{32-2}{6(7-2)} \right[= 2; \qquad t(G) \geq 2.$$



а б

Рис. 9.9

Чтобы определить, какие ребра необходимо удалить для преобразования графа в планарный, выделим все запрещенные фигуры и построим двумерную таблицу, каждая строка которой взаимно однозначно соответствует запрещенной фигуре Q_l, столбец – ребру ρ_j (табл. 9.1). Тогда покрытие строк столбцами этой таблицы определит, какие ребра необходимо удалить для приведения графа к планарному виду.

Минимальное покрытие будет соответствовать минимальному решению, так как удаление любого ребра выводит запрещенную фигуру из класса подграфов, гомеоморфных K_5 или $K_{3,3}$.

Для рассматриваемого графа эта таблица имеет следующий вид.

Таблица 9.1

Q_i	ρ_j						
	(x_1, x_2)	(x_2, x_6)	(x_2, x_3)	(x_1, x_6)	(x_6, x_7)	(x_1, x_7)	(x_2, x_4)
$G_1 \to Q_1$		1	1		1		1
$G_2 \to Q_2$	1		1	1		1	1
Q_i	ρ_j						
	(x_2, x_7)	(x_3, x_6)	(x_3, x_7)	(x_3, x_4)	(x_4, x_6)	(x_4, x_5)	(x_5, x_7)
$G_1 \to Q_1$	1	1	1	1	1	1	1
$G_2 \to Q_2$		1	1		1	1	1

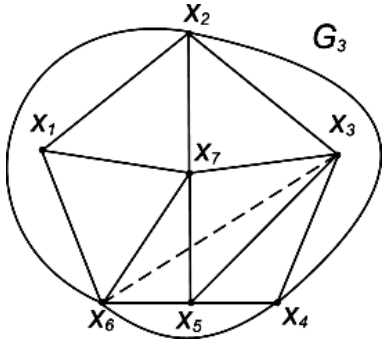


Рис. 9.10

Минимальное покрытие содержит одно из ребер (x_2, x_3), (x_2, x_4), (x_3, x_6), (x_3, x_7), (x_4, x_6), (x_4, x_5) или (x_5, x_7).

После удаления, например, ребра (x_3, x_6) получим планарный граф G_3 (рис. 9.10).

Соединение, которое соответствует удаленному ребру (показано пунктирной линией), реализуется на второй плоскости. Толщина графа $t(G)=2$.

9.2. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие графы называются гомеоморфными?
2. Дайте определение плоского (планарного) графа.
3. Сформулируйте критерии планарности графов.
4. Какие фигуры являются запрещенными при определении планарности графа?
5. Что называется числом планарности графа?
6. Как определяется толщина графа?
7. По заданному преподавателем графу определите, является ли он планарным. Найдите его толщину.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Кристофидес Н. Теория графов. Алгоритмический подход. М.: Мир, 1978. 432 с.
2. Ф.Харари Теория графов. М.: КомКнига, 2006. 302 с.
3. О.Оре Графы и их применение. М.: КомКнига, 2006. 172 с.
4. Новиков Ф.А. Дискретная математика для программистов: Учебник. С.Пб.:Изд. «Питер», 2004. 364 с.
5. Горбатов В.А., Горбатов А.В., Горбатова М.В. Дискретная математика: Учебник для вузов. М.: АСТ «Астрель», 2006. 447 с.
6. Бондарев В.М., Рублинецкий В.И., Качко Е.Г. Основы программирования / Рублинецкий В.И. Введение в мир алгоритмов. Харьков: Фолио; Ростов-на-Дону: Феникс, 1997. 368 с.
7. Скворцов С.В., Хрюкин В.И. Экстремальные пути на графах: Методические указания к практическим занятиям. Рязань: РГРТА, 1995.