Московский государственный технический
университетим. Н.Э. Баумана

Р.С. Исмагилов,
А.В. Калинкин,
В.В. Станцо

ГРАФЫ

Издательство МГТУ им. Н.Э. Баумана
1998

Московский государственный технический университет
им. Н.Э.Баумана

Р.С.Исмагилов, А.В.Калинкин, В.В.Станцо

ГРАФЫ

Рекомендовано редсоветом МГТУ им. Н.Э. Баумана
в качестве учебного пособия по курсу
"Дискретная математика"

Под редакцией Р.С.Исмагилова

Издательство МГТУ им. Н.Э.Баумана
1999

УДК 519.17
ББК 22.174
И87

Рецензенты: А.П. Савин, Э.Р. Смольяков

И87 Исмагилов Р.С., Калинкин А.В., Станцо В.В. Графы: Учебное пособие по курсу "Дискретная математика" /Под ред. Р.С.Исмагилова. М.: Изд-во МГТУ им. Н.Э.Баумана, 1999. 40 с.

ISBN 5-7038-1375-1

В учебном пособии приведены основные определения и краткие теоретические сведения по теории графов. Рассмотрены алгоритмы решения основных задач оптимизации на графах. Разобраны примеры, даны условия типового расчета.

Для студентов факультетов ИБМ, РК, ФН.

Ил. 31. Табл. 3. Библиогр. 5 назв.

УДК 519.17
ББК 22.174

Редакция заказной литературы

Раис Сальманович Исмагилов
Александр Вячеславович Калинкин
Виталий Владимирович Станцо

ГРАФЫ

Заведующая редакцией *Н.Г.Ковалевская*
Редактор *С.А.Филиппова*
Корректор *Л.И.Малютина*

ISBN 5-7038-1375-1

Издательская лицензия No 020523 от 25.04.97.

Подписано в печать 29.10.98. Формат 60×84/16. Бумага тип. No 2. Печ. л. 2,5. Усл. печ. л. 2,32. Уч.-изд. л. 2,27. Тираж 200 экз. Изд. No 136. Заказ No 299

Издательство МГТУ им. Н.Э.Баумана.
107005, Москва, 2-я Бауманская, 5.

Граф — наглядный геометрический объект, представляющий собой набор точек, соединенных отрезками. При кажущейся простоте объекта он является основой содержательной математической теории. С графами связан ряд оптимизационных задач, имеющих прикладное значение. Алгоритмы решения некоторых задач такого типа рассматриваются в настоящем учебном пособии.

1. ОСНОВНЫЕ ПОНЯТИЯ ТЕОРИИ ГРАФОВ

1.1. Графы [1]. Пусть V — конечное непустое множество. Его подмножество, состоящее из элементов u, v, $u \neq v$, записывают в виде $\{u, v\}$. Оно называется также *неупорядоченной парой*. Пусть X — некоторый (возможно, пустой) набор неупорядоченных пар. В этом случае говорят, что задан *граф* $G = (V, X)$. Элементы множества V называются *вершинами* графа, а элементы набора X — *ребрами* графа. Граф удобно изображать геометрически: вершины — в виде точек на плоскости, а ребра — в виде дуг, соединяющих соответствующие точки. Изображения вершин выделяют так, чтобы они отличались от точек пересечения дуг.

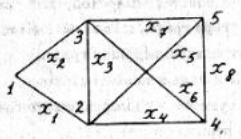


Рис. 1

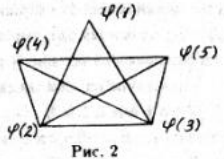


Рис. 2

П р и м е р 1.1. Пусть $V = \{1,2,3,4,5\}$, $X = \{x_1 = \{1,2\}, x_2 = \{1,3\}, x_3 = \{2,3\}, x_4 = \{2,4\}, x_5 = \{2,5\}, x_6 = \{3,4\}, x_7 = \{3,5\}, x_8 = \{4,5\}\}$. Одно из возможных изображений графа $G = (V, X)$ приведено на рис. 1.

Если $x = \{u, v\}$ — ребро графа, то вершины u и v называются *концами ребра* x. Если вершина v является концом ребра x, то говорят, что v и x *инцидентны*. Вершины u и v графа G называются *смежными*, если существует ребро, соединяющее их, т. е. $\{u, v\} \in X$. *Степенью*

вершины v (обозначение: $\deg(v)$) называется число ребер графа, инцидентных вершине v. Вершина графа, имеющая степень 0, называется *изолированной*, а вершина, имеющая степень 1, — *висячей*.

Теорема 1.1. $\sum_{v \in V} \deg(v) = 2m$, где $m = |X|$.

Графы $G_1 = (V_1, X_1)$ и $G_2 = (V_2, X_2)$ называются *изоморфными*, если существует взаимно однозначное отображение $\varphi : V_1 \to V_2$, сохраняющее смежность, т. е. $\{u, v\} \in X_1 \Leftrightarrow \{\varphi(u), \varphi(v)\} \in X_2$.

Пример 1.2. Графы, изображенные на рис. 1 и 2, изоморфны.

Последовательность $v_1 v_2 v_3 \dots v_k v_{k+1}$ $(k \geqslant 1)$, в которой $\{v_i, v_{i+1}\} \in X$ для любого $i = 1, \dots, k$, называется *путем, соединяющим вершины* v_1 и v_{k+1}. Число ребер k называется *длиной* пути. Путь, в котором все ребра разные, называется *цепью*. Цепь, соединяющая вершины u и v, называется (u, v)-*цепью*. Если все вершины пути различны (кроме, может быть, первой и последней), то цепь — *простая*. Путь, в котором $v_1 = v_{k+1}$, называется *замкнутым*. Замкнутая цепь называется *циклом*, замкнутая простая цепь — *простым циклом*.

Граф называется *связным*, если для любых его вершин существует путь, соединяющий эти вершины. Цепь является *эйлеровой*, если она проходит по одному разу через каждое ребро графа, и *гамильтоновой*, если она проходит по одному разу через каждую вершину графа.

Аналогично определяются *эйлеровы* и *гамильтоновы циклы*.

Теорема 1.2. В связном графе существует эйлеров цикл тогда и только тогда, когда степени всех его вершин четны.

Пример 1.3. На рис. 1: 1 2 3 1 — простой цикл длиной 3; 1 2 3 4 5 — простая цепь длиной 4, соединяющая вершины 1 и 5; 2 3 4 2 5 — цепь длиной 4, соединяющая вершины 2 и 5 (эта цепь не является простой, так как дважды проходит через вершину 2); 1 2 1 — замкнутый путь длиной 2, не являющийся цепью; 4 3 1 2 5 3 2 4 5 — эйлерова цепь; 1 2 4 5 3 1 — гамильтонов цикл.

Деревом называется связный граф, не содержащий циклов.

Теорема 1.3. Следующие утверждения равносильны: а) G — дерево, б) G — связный граф и $m = n - 1$, где $m = |X|$, $n = |V|$, в) G —

граф без циклов и $m = n - 1$, г) любые две несовпадающие вершины графа G соединяет единственная простая цепь.

Подграфом графа G называется граф, все вершины и ребра которого содержатся среди вершин и ребер графа G. Подграф является *собственным*, если он отличен от самого графа. *Остовным подграфом* называется подграф, содержащий все вершины графа. *Остовом* называют остовный подграф, являющийся деревом.

Пример 1.4. Остовами графа, изображенного на рис. 1, являются, например, деревья с множествами ребер $X_1 = \{x_1, x_2, x_4, x_7\}$, $X_2 = \{x_1, x_2, x_4, x_5\}$ (рис 3). Всего этот граф имеет 40 остовов.

Пусть $G(V, X)$ — граф, $f : X \to R^+$ — вещественнозначная функция, ставящая в соответствие каждому ребру x положительное число $f(x)$ — вес ребра x. Пару (G, f) называют *взвешенным графом*. Под весом любого подграфа взвешенного графа понимают сумму весов ребер подграфа.

Расстоянием между вершинами u и v связного графа G называется минимальная длина простой цепи, соединяющей u и v, — $d(u, v)$. Для произвольных вершин u, v, w связного графа G выполняются аксиомы метрики: а) $d(u, v) \geqslant 0$, причем $d(u, v) = 0 \Leftrightarrow u = v$; б) $d(u, v) = d(v, u)$; в) $d(u, w) \leqslant d(u, v) + d(v, w)$ — неравенство треугольника.

Диаметром связного графа G называется величина $D(G) = \max_{u,v \in V} d(u, v)$. Величина $R(G) = \min_{u \in V} \max_{v \in V} d(u, v)$ называется *радиусом* графа G (из неравенства треугольника следует, что $D(G) \leqslant 2R(G)$). Вершина $u_c \in G$ такая, что $R(G) = \max_{v \in V} d(u_c, v)$, называется *центром* графа G.

Пример 1.5. Для графа на рис. 1 $D(G) = 2$, $R(G) = 1$, центрами являются вершины 2 и 3.

1.2. Ориентированные графы (орграфы). *Орграф* $G = (V, \vec{X})$ задается непустым конечным множеством V и набором $\vec{X}$ упорядоченных пар (u, v), где $u \in V$, $v \in V$, $u \neq v$; (пары (u, v) и (v, u) считаются различными). V — это множество *вершин*, $\vec{X}$ — множество *ребер* орграфа. Вершина u называется *началом*, а вершина v — *концом* ребра

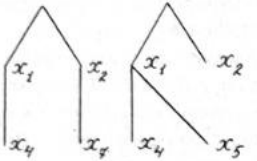


Рис. 3

Рис. 4

(u,v). При геометрическом изображении орграфа дуга, изображающая ребро (u,v), снабжается стрелкой, идущей от u к v.

Путем в орграфе называется цепочка вершин $v_1v_2v_3\dots v_kv_{k+1}$, в которой $(v_i,v_{i+1})\in\vec{X}$ для любого $i=1,\dots,k$. Вершина v_1 называется *началом* пути, вершина v_{k+1} —*концом* пути, а число ребер k — *длиной* пути. Вершина v называется *достижимой* из вершины u, если существует путь с началом в u и концом в v. *Цепь, простая цепь, цикл, простой цикл* определяются так же, как для неориентированных графов. Простой цикл в орграфе также называется *контуром*. Контур называется *гамильтоновым*, если проходит через каждую вершину графа и притом ровно один раз. *Взвешенным орграфом* называется пара (G,f), где $f:\vec{X}\to R^+$ — вещественнозначная функция.

Пример 1.6. На рис. 4 изображен взвешенный орграф $G=(V,\vec{X})$, $V=\{1,2,3,4\}$, $\vec{X}=\{x_1=(1,2),\ x_2=(4,1),\ x_3=(4,3),\ x_4=(3,2),\ x_5=(3,1),\ x_6=(2,4)\}$ (числа около дуг задают веса ребер). 4 3 2 — простая цепь, 1 2 4 1 — контур, 1 2 4 3 1 — единственный гамильтонов контур (его вес равен 25). Вес всего графа равен 36.

Заменяя каждую упорядоченную пару (u,v) из набора $\vec{X}$ неупорядоченной парой $\{u,v\}$, состоящей из тех же элементов, получаем *граф* $G=(V,X)$, *ассоциированный с орграфом* $G=(V,\vec{X})$.

1.3. Способы задания графов и орграфов на ЭВМ. Рассмотрим неориентированный граф $G=(V,X)$. Для каждой вершины $v\in V$ определен список вершин, смежных с ней. Обозначим его Γ_v. Множество $\{\Gamma_{v_i},\ i=1,\dots,n\}$ называется *списком смежности* графа G.

Пример 1.7. Список смежности для графа, изображенного на рис. 1: $\Gamma_1=\{2,3\}$, $\Gamma_2=\{1,3,4,5\}$, $\Gamma_3=\{1,2,4,5\}$, $\Gamma_4=\{2,3,5\}$, $\Gamma_5=\{2,3,4\}$.

Пусть $G=(V,X)$, $|V|=n$, $|X|=m$, — граф, в котором вершинам и ребрам присвоены номера. *Матрицей инцидентности* графа G называется $m\times n$-матрица $I(G)=[a_{ij}]_{i,j=1}^{m,n}$, где $a_{ij}=1$, если вершина v_i инцидентна ребру x_j; $a_{ij}=0$, если вершина v_i не инцидентна ребру x_j. *Матрицей весов* взвешенного графа (G,f) называется квадратная симметричная $n\times n$-матрица $W(G)=[w_{ij}]_{i,j=1}^{m,n}$, где $w_{ij}=f(\{v_i,v_j\})$, если $\{v_i,v_j\}\in X$; $w_{ij}=\infty$, если $\{v_i,v_j\}\notin X$.

Для ориентированного графа $G=(V,\vec{X})$ вводятся множества $\Gamma_v^+=\{u|u\in V,(u,v)\in\vec{X}\}$, $\Gamma_v^-=\{u|u\in V,(v,u)\in\vec{X}\}$.

Пример 1.8. Для орграфа, изображенного на рис. 4, список смежности: $\Gamma_1^+=\{3,4\}$, $\Gamma_1^-=\{2\}$, $\Gamma_2^+=\{1,3\}$, $\Gamma_2^-=\{4\}$, $\Gamma_3^+=\{4\}$, $\Gamma_3^-=\{1,2\}$, $\Gamma_4^+=\{2\}$, $\Gamma_4^-=\{1,3\}$

Матрицей смежности орграфа $G(V,\vec{X})$, $|V|=n$, $|\vec{X}|=m$, называется квадратная $n\times n$-матрица $S(G)=[s_{ij}]_{i,j=1}^{m,n}$, где $s_{ij}=1$, если $(v_i,v_j)\in\vec{X}$; $s_{ij}=0$, если $(v_i,v_j)\notin\vec{X}$. *Матрицей весов* взвешенного орграфа (G,f) называется квадратная $n\times n$-матрица $W(G)=[w_{ij}]_{i,j=1}^{m,n}$, где $w_{ij}=f(v_i,v_j)$, если $(v_i,v_j)\in\vec{X}$; $w_{ij}=\infty$, если $(v_i,v_j)\notin\vec{X}$ (здесь $f(v_i,v_j)$ обозначает вес ребра $(v_i,v_j)\in\vec{X}$).

Пример 1.9. Для взвешенного орграфа, изображенного на рис. 4,

$$S(G)=\begin{bmatrix}0&1&0&0\\0&0&0&1\\1&1&0&0\\1&0&1&0\end{bmatrix},\qquad W(G)=\begin{bmatrix}\infty&7&\infty&\infty\\\infty&\infty&\infty&8\\1&5&\infty&\infty\\6&\infty&9&\infty\end{bmatrix}.$$

2. ЗАДАЧА ПОИСКА ОСТОВА МИНИМАЛЬНОГО ВЕСА

Задача поиска во взвешенном связном графе остова минимального веса может иметь следующую интерпретацию. Исходный граф G есть проектируемая система дорог (ребра графа), связывающих города некоторой области (вершины графа), вес ребра — стоимость строительства дороги, соединяющей два города. Требуется построить систему дорог с минимальной стоимостью, чтобы из любого города можно было

проехать в любой другой город (искомый остовный подграф — связный). Искомый подграф — дерево, так как если он содержит цикл, то можно удалить ребро, входящее в цикл, — требования задачи останутся выполненными. Опишем один из алгоритмов решения (Р. Прим, 1957 г. [2]).

Дан связный граф $G = (V, X)$, $|V| = n$, и весовая функция $f(x)$, $x \in X$. Алгоритм состоит из $n-1$ шага. На каждом шаге строится дерево D_i, $i = 1, \ldots, n-1$. Дерево D_{n-1} является остовом минимального веса.

1. Выбираем ребро x_1 с минимальным весом из множества всех ребер X и строим дерево D_1, полагая его состоящим из ребра x_1 и двух инцидентных ребру x_1 вершин.

2. Если дерево D_i порядка $|D_i| = i + 1$ уже построено, то среди ребер, соединяющих вершины этого дерева с вершинами графа G, не входящими в D_i, выбираем ребро x_{i+1} с минимальным весом. Строим дерево D_{i+1}, присоединяя к D_i ребро x_{i+1} вместе с его не входящим в D_i концом.

3. Если $i = n-1$, то остов минимального веса D_{n-1} построен, конец алгоритма. Иначе переходим к п. 2.

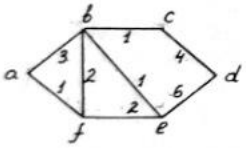


Рис. 5

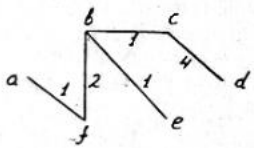


Рис. 6

Пример 2.1. Рассмотрим взвешенный граф, изображенный на рис. 5. На первом шаге полагаем $x_1 = \{b, c\}$. Второй шаг дает $x_2 = \{b, e\}$. Далее $x_3 = \{b, f\}$, $x_4 = \{f, a\}$, $x_5 = \{c, d\}$. Ребра $\{b, c\}$, $\{b, e\}$, $\{b, f\}$, $\{f, a\}$, $\{c, d\}$ составляют остов с минимальным весом 9 (рис. 6).

3. ЗАДАЧА ПОИСКА ДЕРЕВА КРАТЧАЙШИХ ПУТЕЙ

Пусть (G, f) — взвешенный связный граф. Вес $f(x)$ ребра x интерпретируем как расстояние между вершинами, смежными данному ребру. Для заданной начальной вершины s и конечной вершины t необходимо найти простую цепь, соединяющую s и t, с минимальным весом, или *кратчайший* (s, t)*-путь*. Опишем один из алгоритмов решения задачи (Е. Дейкстра, 1957 г. [2]).

Дан связный граф $G(V, X)$, $|V| = n$, и весовая функция $f(x)$. На каждой итерации алгоритма любая вершина v графа G имеет неотрицательную метку $l(v)$, которая может быть временной или постоянной. Перед началом первой итерации вершина s имеет постоянную метку $l(s)+ = 0+$, а метки всех остальных вершин равны ∞ и являются временными. Постоянная метка $l(v)+$ — найденный вес кратчайшего (s, v)-пути. Временная метка $l(v)$ — вес кратчайшего (s, v)-пути, проходящего через вершины с постоянными метками. На каждой итерации алгоритма временная метка одной из вершин переходит в постоянную. С каждой вершиной v графа G (кроме s) связывается также метка $\theta(v)$, которая является номером вершины, предшествующей v в (s, v)-пути, имеющем минимальный вес среди всех (s, v)-путей, проходящих через вершины, получивших к данному моменту постоянные метки. После получения вершиной v постоянной метки с помощью θ-меток указывается последовательность вершин, составляющая кратчайший (s, v)-путь.

1. Положим $l(s)+ = 0+$, т. е. будем считать эту метку постоянной, и $l(v) = \infty$ для всех $v \in V$, $v \neq s$, считая эти метки временными. Примем $u = s$.

2. Для всех $v \in \Gamma(u)$ с временными метками выполним действия: если $l(v) > l(u) + f(x)$, где $x = \{u, v\}$ — ребро, то $l(v) = l(u) + f(x)$ и $\theta(v) = u$. Иначе $l(v)$ и $\theta(v)$ не изменяем.

3. Пусть V' — множество вершин с временными метками l. Найдем вершину v^* такую, что $l(v^*) = \min\limits_{v \in V'} l(v)$. Считаем метку $l(v^*)$ постоянной меткой вершины v^*; $l(v^*)+$.

4. Положим $u = v^*$. Если $u = t$, то переходим к п. 5, иначе переходим к п. 2.

5. По θ-меткам указываем кратчайший (s,t)-путь. Конец алгоритма.

Пункт 4 можно модифицировать так, чтобы алгоритм заканчивал работу после получения всеми вершинами графа G постоянных меток. Алгоритм определит остовное дерево D кратчайших путей из вершины s. Единственный (s,t)-путь в дереве D будет кратчайшим (s,t)-путем в графе G.

Пример 3.1. Рассмотрим взвешенный граф, изображенный на рис. 7. На рис. 8 изображено соответствующее дерево D. Работа алгоритма представлена в таблице. В строках выписаны значения меток $l(v)$, $\theta(v)$ на i-й итерации для каждой из вершин (кроме получивших постоянную метку). В последнем столбце указана текущая вершина u для i-й итерации.

i	s	q	r	p	z	t	u
0	0+	∞	∞	∞	∞	∞	$u=s$
		$5;s$	$2;s$	$5;s$	∞	$12;s$	
1		$3;r$	$2+;s$	$4;r$	∞	$12;s$	$u=r$
2		$3+;r$		$4;r$	$5;q$	$12;s$	$u=q$
3				$4+;r$	$5;q$	$12;s$	$u=p$
4					$5+;q$	$7;z$	$u=z$
5						$7+;z$	$u=t$

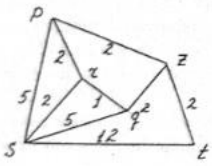


Рис. 7

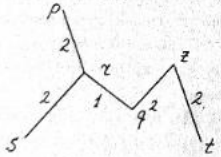


Рис. 8

4. АЛГОРИТМ ДОСТИЖИМОСТИ

Пусть дан орграф $G=(V,\vec{X})$, в котором фиксирована вершина $\alpha \in V$. Обозначим через V^α множество всех вершин, достижимых из α; его называют *множеством достижимости из вершины* α. Нижеследующий алгоритм достижимости позволяет описать V^α, и, кроме того, для любой вершины $a \neq \alpha$, входящей в V^α, найти путь из α в a.

1. Нулевой шаг алгоритма. Вершине α присваиваем метку 0.

2. Пусть выполнен k-й шаг алгоритма; при этом некоторые вершины получили метки из набора чисел $\{0,1,2,\ldots,k\}$.

3. $k+1$-й шаг алгоритма. Находим все вершины $v \in V$, не имеющие метки и такие, что в $\vec{X}$ имеется ребро (u,v), где вершина u имеет метку k. Указанные вершины v помечаем числом $k+1$. Работа алгоритма завершается, когда этот процесс не приводит к присвоению новых меток. Искомое множество V^α состоит из всех вершин, имеющих метки.

Пусть V_k^α — множество вершин с меткой k, $\vec{X}_k^\alpha$ — множество ребер вида (u,v), где $u \in V_{k-1}^\alpha$, $v \in V_k^\alpha$; для ребра (u,v) указанного вида назовем вершину u предшественницей вершины v. Для вершины $v \in V_k^\alpha$, последовательно отыскивая предшественниц, получаем такие вершины $v=v_k, v_{k-1}, \ldots, v_0=\alpha$, что $v_i \in V_i^\alpha$, $(v_{i-1},v_i) \in \vec{X}_i^\alpha$. Это дает искомый путь $v_0, v_1, \ldots, v_k$ из α в v. Граф $G^\alpha=(V^\alpha,\vec{X}^\alpha)$, где $\vec{X}^\alpha=\bigcup\limits_k\{\vec{X}_k^\alpha\}$, называют *слоистым подграфом* графа G, а множества V_k^α — его *уровнями*.

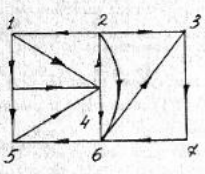


Рис. 9

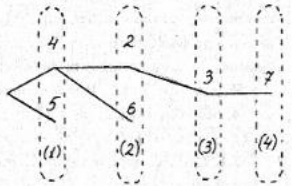


Рис. 10

Пример 4.1. Найдем множество достижимости из вершины α и построим слоистый орграф G^α для орграфа G, изображенного на рис. 9. (Вершины, отличные от α, перенумерованы.) Искомый граф изображен на рис. 10.

5. ЗАДАЧА О МАКСИМАЛЬНОМ ПОТОКЕ В СЕТИ

Начнем с наглядного описания задачи. Пусть имеется несколько пунктов и сеть дорог (каждая дорога соединяет два пункта). Предположим, что в одном из пунктов производят некоторую продукцию, а в другом — потребляют ее. Требуется организовать поток перевозок продукции от производителя к потребителю, т. е. для каждой из дорог указать количество перевозимой по ней продукции (за единицу времени). Задача состоит в том, чтобы количество продукции, доставляемой от производителя к потребителю, было наибольшим. Имеются два ограничения: во-первых, на каждой из дорог одностороннее движение, и, во-вторых, каждая из них имеет фиксированную пропускную способность. Ниже излагается математическая постановка задачи и метод ее решения (алгоритм Форда—Фалкерсона, 1962 г. [3]).

Сетью называется взвешенный ориентированный граф с двумя выделенными вершинами. Назовем *псевдопутем* в орграфе любую цепочку ребер $x_1, x_2, \ldots, x_m$, такую, что для любого $i = 1, 2, \ldots, m$ ребра x_i, x_{i+1} различны и имеют общую вершину. При движении по этой цепочке (движение начинается с ребра x_1 и заканчивается на ребре x_m) некоторые ребра x, будут пройдены согласно ориентации ребра, а некоторые — против ориентации. Ребра первого типа называются *прямыми*, а ребра второго типа — *обратными*. Если все ребра псевдопути прямые, то этот псевдопуть является путем.

Пример 5.1. В ориентированном графе, изображенном на рис. 4, ребра $x_1 = (3,2)$, $x_2 = (4,3)$, $x_3 = (4,1)$ составляют псевдопуть x_1, x_2, x_3, причем ребра x_1, x_2 — обратные, ребро x_3 — прямое.

5.1. Постановка задачи. Дана ориентированная сеть $G = (V, \vec{X})$ с двумя выделенными вершинами α и ω, называемыми источником и стоком, $\alpha \neq \omega$ (предполагаем, что ω достижима из α). Каждому ребру $x = (u, v)$ поставлено в соответствие положительное число $c(u, v)$, называемое *пропускной способностью ребра* (u, v). *Потоком* называется функция $f(x) = f(u, v)$, $(u, v) \in \vec{X}$, удовлетворяющая следующим условиям: а) $0 \leqslant f(u, v) \leqslant c(u, v)$ для любого ребра $(u, v) \in \vec{X}$; б) для любой

вершины $a \in V$, $a \neq \alpha$, $a \neq \omega$, справедливо равенство

$$\sum_{u \in V} f(u, a) = \sum_{v \in V} f(a, v). \qquad (1)$$

Будем обозначать поток $\{f(u, v), (u, v) \in \vec{X}\}$, или просто f.

Суммарной величиной потока называется число

$$M(f) = \sum_{v \in V} f(\alpha, v) - \sum_{u \in V} f(u, \alpha). \qquad (2)$$

Поясним связь этих понятий с приведенным выше наглядным описанием задачи. Число $f(u, v)$ — количество продукции, перевозимой из вершины u в вершину v по ребру (u, v). Это число (неотрицательное) не превосходит пропускной способности ребра $c(u, v)$. Равенство (1) можно выразить так: для любой вершины a, $a \neq \alpha$, $a \neq \omega$, количество продукции, поступающей в a, равно количеству продукции, вывозимой из a. Суммарная величина $M(f)$, заданная формулой (2), — количество продукции, вывозимой из источника (от производителя) α.

Будем говорить, что поток f' лучше потока f, если $M(f') > M(f)$. Основная задача: последовательно улучшая поток, найти максимальное значение величины $M(f)$, или, кратко, $M(f) \to \max$. Поток f, на котором достигается этот максимум, называется оптимальным.

5.2. Способы улучшения потока. Пусть дан поток $\{f(x) = f(u, v), (u, v) \in \vec{X}\}$. Ребро $(u, v) \in \vec{X}$ называется: а) *пустым*, если $f(u, v) = 0$; б) *положительным*, если $0 < f(u, v) \leqslant c(u, v)$; в) *насыщенным*, если $f(u, v) = c(u, v)$.

Первый способ улучшения. Допустим, что в орграфе $G = (V, \vec{X})$ нашелся путь из α в ω, состоящий только из ненасыщенных ребер, т. е. обладающий следующим свойством: $f(u, v) < c(u, v)$ для любого ребра (u, v) данного пути из α в ω. Рассмотрим величины $c(u, v) - f(u, v)$ для всех ребер (u, v) указанного пути; пусть d — наименьшая из них, т. е. $d = \min\{c(u, v) - f(u, v)\}$, где (u, v) — ребра данного пути. Ясно, что $d > 0$. Определим функцию $\{f_1 = f_1(u, v), (u, v) \in \vec{X}\}$ равенствами

$$f_1(u, v) = \begin{cases} f(u, v) + d, & \text{если ребро } (u, v) \text{ входит в данный путь,} \\ f(u, v), & \text{в противном случае.} \end{cases}$$

Очевидно, функция f_1 удовлетворяет условиям определения потока, причем $M(f_1) = M(f) + d$. Путь из α в ω с условием $d > 0$ называется *улучшающим путем*.

Второй способ улучшения. Допустим, что в орграфе $G = (V, \vec{X})$ нашелся псевдопуть из α в ω, обладающий следующим свойством: все прямые ребра псевдопути — ненасыщенные, а все обратные ребра — положительные, т. е.

$$\begin{cases} f(u',v') < c(u',v'), & \text{если } (u',v') \text{ — прямое ребро псевдопути,} \\ f(u'',v'') > 0, & \text{если } (u'',v'') \text{ — обратное ребро псевдопути.} \end{cases}$$

Рассмотрим числа $c(u',v') - f(u',v')$ для всех прямых ребер (u',v') этого псевдопути, и числа $f(u'',v'')$ для всех обратных ребер (u'',v''). Пусть d — наименьшее из этих чисел,

$$d = \min \begin{cases} c(u',v') - f(u',v'), & \text{если } (u',v') \text{ — прямое ребро псевдопути,} \\ f(u'',v''), & \text{если } (u'',v'') \text{ — обратное ребро псевдопути.} \end{cases}$$

Из предположений о псевдопути следует, что $d > 0$. Определим функцию $\{f_1 = f_1(u,v), (u,v) \in \vec{X}\}$ равенствами

$$f_1(u,v) = \begin{cases} f(u,v) + d, & \text{если } (u,v) \text{ — прямое ребро псевдопути,} \\ f(u,v) - d, & \text{если } (u,v) \text{ — обратное ребро псевдопути,} \\ f(u,v), & \text{если ребро } (u,v) \text{ не входит в псевдопуть.} \end{cases}$$

Очевидно, функция $f_1 = \{f_1(u,v), (u,v) \in \vec{X}\}$ удовлетворяет условиям определения потока, причем $M(f_1) = M(f) + d$. Псевдопуть из α в ω, использованный в этой конструкции, называется *улучшающим псевдопутем*.

Первый способ является частным случаем второго способа. Опишем метод нахождения улучшающих путей и псевдопутей из α в ω.

Метод вспомогательных сетей. Для нахождения улучшающих путей удаляем в рассматриваемом орграфе все насыщенные ребра. В полученной сети N_f^0 ищем путь из α в ω, используя алгоритм достижимости. Если такой путь найден, переносим его на исходную сеть; это и есть искомый улучшающий путь. Если же в сети N_f^0 вершина ω не достижима из вершины α, то улучшающих путей нет.

Для нахождения улучшающих псевдопутей строим вспомогательную сеть N_f^1: для любого положительного ребра (u,v) строим противоположное ребро (v,u) (изображаем его пунктирной дугой), после чего

все насыщенные ребра удаляем. В полученной сети ищем путь из α в ω, используя алгоритм достижимости, и переносим его на исходную сеть. В результате получим искомый псевдопуть (при упомянутом переносе обозначенные пунктиром ребра перейдут в обратные ребра псевдопути, а обозначенные сплошной линией — в прямые ребра псевдопути). Если в сети N_f^1 вершина ω не достижима из α, то улучшающих псевдопутей нет.

5.3. Алгоритм Форда—Фалкерсона.

Теорема 5.1 [3]. Пусть в данной сети для потока f не существует улучшающих псевдопутей из α в ω. Тогда поток f оптимален.

Другими словами, если поток в сети нельзя улучшить описанными в п. 5.2 способами, то его нельзя улучшить никакими иными способами. Теорема 5.1 приводит к следующему алгоритму решения задачи о максимальном потоке.

Выбираем произвольный поток f (например, нулевой поток) и последовательно применяем первый способ улучшения потока. Если для очередного потока f' нет улучшающих путей, то используем второй способ улучшения. Работа алгоритма заканчивается, когда для очередного потока f'' нет улучшающих псевдопутей из α в ω, т. е. во вспомогательной сети $N_{f''}^1$ вершина ω не достижима из вершины α. Полученный поток f'' оптимален.

Пример 5.2. Решим задачу о максимальном потоке для сети, изображенной на рис. 11 (числа около ребер обозначают пропускные способности $c(u,v)$).

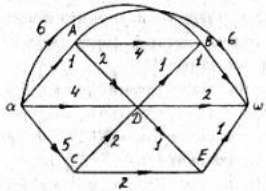


Рис. 11

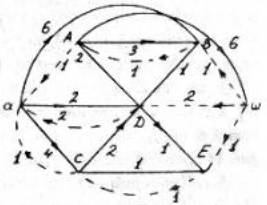


Рис. 12

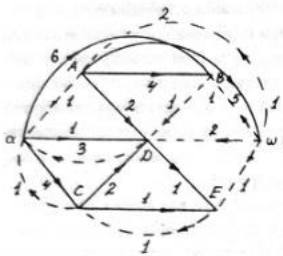


Рис. 13

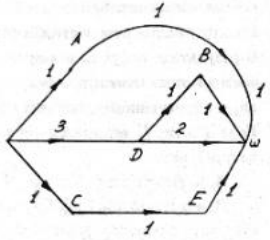


Рис. 14

Первый шаг. Строим поток f_0, полагая $f_0(\alpha,C)=1$, $f_0(C,E)=1$, $f_0(E,\omega)=1$, на остальных ребрах $f_0=0$. Затем улучшаем его, прибавив к потоку 1 на ребрах (α,A), (A,B), (B,ω), и, далее, прибавив к потоку 2 на ребрах (α,D) и (D,ω). Получили поток f_1, $M(f_1)=4$.

Построив сеть $N^0_{f_1}$ (т. е. удалив все насыщенные ребра), убеждаемся в том, что в ней вершина ω не достижима из вершины α.

Второй шаг. Строим сеть $N^1_{f_1}$, вводя изображенные пунктиром ребра, как описано в п. 5.2, см. рис. 12 (числа около прямых ребер обозначают величины $c(u,v)-f_1(u,v)$, числа около обратных ребер — величины $f_1(u,v)$).

В сети $N^1_{f_1}$ имеется путь из α в ω вида $(\alpha,D),(D,B),(B,A),(A,\omega)$. Взяв в исходной сети псевдопуть с теми же вершинами, добавив 1 к значениям потока f_1 на прямых дугах, и вычтя 1 из значений потока f_1 на обратном ребре (B,A), получаем новый поток f_2. Построенная по нему сеть $N^1_{f_2}$ имеет вид, изображенный на рис. 13.

Применив алгоритм достижимости, убеждаемся в том, что в сети $N^1_{f_2}$ вершина ω не достижима из вершины α. Следовательно, построенный поток f_2 максимален и его суммарная величина $M(f_2)=5$, см. рис. 14 (указаны значения $f_2(u,v)$, отличные от нуля).

5.4. Критерий оптимальности потока. Рассматривается сеть $G=(V,\vec{X})$ с выделенными вершинами α, ω и с заданными пропускными способностями $\{c(u,v),(u,v)\in\vec{X}\}$. *Разрезом* сети называется набор ребер, обладающий следующим свойством: при удалении из множества

$\vec{X}$ этого набора ребер в новом орграфе не существует пути из α в ω. *Пропускной способностью (весом)* разреза называется сумма пропускных способностей всех ребер разреза.

Теорема 5.2 [3]. Максимальное значение суммарной величины потока в сети совпадает с минимальным значением пропускной способности разреза сети.

Иногда критерий оптимальности формулируют кратко: максимальный поток равен весу минимального разреза. Теорема 5.2 позволяет в случае небольшого числа $|\vec{X}|$ ребер в сети $G=(V,\vec{X})$ найти значение $\max M(f)$ перебором всех возможных разрезов сети.

Пример 5.3. Разрезом для сети, изображенной на рис. 11, является, например, набор ребер (α,A), (α,B), (α,C), (α,D) (пропускная способность разреза равна 16). Разрез минимального веса 5 составляют ребра (α,A), (B,ω), (D,ω), (E,ω).

6. ЗАДАЧА О ЛАДЕЙНОМ НАБОРЕ

Набор ребер графа $G=(V,X)$ назовем *паросочетанием*, если любые два ребра из этого набора не имеют общих вершин. Набор вершин графа называется *независимым*, если любые две вершины из этого набора несмежны [4]. *Опорой* графа называется такой набор вершин, что любое ребро инцидентно хотя бы одной вершине этого набора. *Максимальное паросочетание, максимальное независимое множество, минимальная опора* определяются очевидным образом.

Теорема 6.1. Множество $E\subset V$ независимо тогда и только тогда, когда его дополнение $E'=V\backslash E$ является опорой.

Граф называется *двудольным*, если множество его вершин разбито на непересекающиеся подмножества U и V, такие, что любое ребро графа имеет вид $\{u,v\}$, $u\in U$, $v\in V$. Двудольный граф записывают в виде $G=(U,V;X)$.

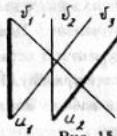


Рис. 15

C	v_1	v_2	v_3	v_4
u_1	0	1	0	1
u_2	1	0	1	0
u_3	0	1	1	1

Рис. 16

Пример 6.1. На рис. 15 изображен двудольный граф и паросочетание, состоящее из ребер $\{u_1, v_1\}$, $\{u_2, v_4\}$. Вершины u_1, v_2, u_3, v_4 образуют независимое множество. Вершины u_2, v_1, v_3 образуют опору.

6.1. Формулировки задач А, Б, В. Дан двудольный граф $G = (U, V; X)$ без изолированных вершин. Рассмотрим следующие задачи: А) найти в графе G максимальное паросочетание; Б) найти минимальную опору; В) найти максимальное независимое множество. Из теоремы 6.1 следует, что задачи Б) и В) эквивалентны.

Матричная формулировка задач А, Б, В. Пусть в двудольном графе $U = \{u_1, \ldots, u_m\}$, $V = \{v_1, \ldots, v_n\}$. Графу $G = (U, V; X)$ поставим в соответствие *матрицу смежности двудольного графа* $C = [c_{ik}]_{i,k=1}^{m,n}$, где $c_{ik} = 0$, если вершины u_i, v_k смежны, $c_{ik} = 1$ в противном случае. Вершинам из U соответствуют строки матрицы C, вершинам из V — ее столбцы, ребрам — нулевые элементы матрицы C.

Пример 6.2. Граф на рис. 15 соответствует матрице на рис. 16.

Столбцы и строки матрицы назовем *линиями*; при данных наборах строк и столбцов назовем *подматрицей* совокупность элементов матрицы, стоящих на пересечении строк и столбцов; сумма числа указанных строк и числа указанных столбцов называется *суммарным размером подматрицы*. Задачи А, Б, В могут быть переформулированы: А) найти максимальный набор нулевых элементов матрицы C, стоящих в разных строчках и разных столбцах матрицы; Б) найти минимальный набор линий, содержащий все нулевые элементы; В) найти подматрицу с максимальным суммарным размером, не содержащую нулей.

Набор нулевых элементов, стоящих в разных строках и разных столбцах матрицы C, назовем *ладейным набором* (шахматные ладьи, расставленные на этих местах, не бьют друг друга). В задаче А) требуется найти *максимальный ладейный набор*.

6.2. Решение задачи А. Выбор начального ладейного набора. В первой строке матрицы возьмем левый нулевой элемент. Удалим из матрицы строку и столбец, содержащие этот элемент. В оставшейся матрице снова выберем нулевой элемент по тому же правилу. Продолжая этот процесс, получим начальный ладейный набор. Этот метод называется *методом северо-западного угла*.

Улучшение данного ладейного набора. Пусть дан ладейный набор. Вошедшие в него нулевые элементы матрицы C, а также содержащие их строки и столбцы назовем *насыщенными*, остальные нулевые элементы и содержащие их строки и столбцы — *пустыми*. Опишем алгоритм меток, который либо заменяет данный набор набором с бо́льшим числом элементов, либо показывает, что данный набор максимален.

1. Отмечаем числом 1 все пустые строки.

2. Пусть совершен k-й шаг алгоритма, k нечетно. Возьмем все строки с меткой k, все находящиеся в них пустые элементы (если такие существуют) и все содержащие эти элементы неотмеченные столбцы (если они есть). Если таких элементов или таких столбцов нет, то ладейный набор максимален, конец алгоритма. Если такие элементы и столбцы имеются, отметим эти столбцы числом $k+1$. Переходим к п. 3.

3. Пусть совершен k-й шаг алгоритма, k четно. Рассмотрим все столбцы с меткой k. Возможны следующие случаи.

Случай 3.1. Все эти столбцы насыщенные. Рассмотрим все находящиеся в них насыщенные элементы и неотмеченные строки, содержащие эти элементы. Если таких строк нет, то построенный ладейный набор максимален, конец алгоритма. Если такие строки имеются, отметим их числом $k+1$. Переходим к п. 2.

Случай 3.2. Существует пустой столбец с четной меткой k. Назовем элемент матрицы, стоящий на пересечении строки с меткой i и столбца с меткой j, элементом типа (i, j). Последовательно строим в матрице цепочку элементов типа $(k-1, k)$, $(k-1, k-2)$, $(k-3, k-2)$, $(k-3, k-4)$, …, $(1, 2)$. В этой цепочке пустые и насыщенные элементы чередуются, и пустых элементов на один больше, чем насыщенных. Изменим ладейный набор, объявив все насыщенные элементы этой цепочки пустыми, а все пустые — насыщенными. Этим ладейный набор улучшен. Удаляем все метки. Переходим к п. 1.

Решение задачи Б. Решим задачу А описанным выше методом. Рассмотрим метки строк и столбцов, полученные на заключительном шаге алгоритма. Искомую минимальную опору образуют неотмеченные строки и отмеченные столбцы.

Р е ш е н и е з а д а ч и В. Рассмотрим заключительные метки строк и столбцов, полученные при решении задачи А. Подматрица, стоящая на пересечении отмеченных строк и неотмеченных столбцов, является подматрицей с максимальным суммарным размером без нулевых элементов.

П р и м е р 6.3. Рассмотрим задачу о нахождении максимального ладейного набора для матрицы, изображенной на рис. 17.

	v_1 v_2 v_3 v_4 v_5 v_6
u_1	0 0 0 0 0 0
u_2	0 1 0 2 3 1
u_3	0 2 0 1 1 1
u_4	0 0 0 0 0 0
u_5	0 0 0 0 0 0
u_6	0 1 1 2 1 1

Рис. 17

	2 4 2 4 4 4
3	0 0 0 0 0 0
3	0 1 0 2 3 1
1	0 2 0 1 1 1
	0 0 0 0 0 0
	0 0 0 0 0 0
1	0 1 1 2 1 1

Рис. 18

	2 4
	0 0 0 0 0 0
5	0 * 0 * * *
3	0 * 0 * * *
	0 0 0 0 0 0
	0 0 0 0 0 0
1	0 * 1 * * *

Рис. 19

Методом северо-западного угла находим первоначальный ладейный набор; составляющие его нулевые элементы отмечены на рис. 17. На рис. 18 указаны метки строк и столбцов, полученные с использованием алгоритма меток. Пятый и шестой столбцы, получившие метку 4, пусты. Следовательно, ладейный набор можно улучшить. Для этого строим цепочку элементов типа $(3,4)$, $(3,2)$, $(1,2)$; на рис. 18 эти элементы соединены пунктирной линией. Теперь улучшаем ладейный набор, объявив два пустых элемента этой цепочки насыщенными, а один насыщенный элемент — пустым. Полученный ладейный набор изображен на рис. 19. На нем указаны метки строк и столбцов, полученные применением к новому ладейному набору алгоритма меток. Алгоритм закончился нечетной меткой 5, т. е. ладейный набор на рис. 19 оптимален. Звездочками отмечены элементы матрицы, стоящие на пересечении отмеченных строк и неотмеченных столбцов; они отличны от нуля и образуют подматрицу с максимальным суммарным порядком 7 без нулей. Строки с номерами 1, 4, 5 и столбцы с номерами 1, 4 образуют минимальную опору.

7. ЗАДАЧА О НАЗНАЧЕНИЯХ

7.1. Постановка задачи. Имеется n персон, которых следует назначить на n должностей (каждого на одну должность). Назначение i-го лица на k-ю должность приносит затраты $c_{ik} \geqslant 0$. Нужно произвести назначения так, чтобы суммарные затраты были минимальны. Из чисел c_{ik} составим *матрицу затрат* C, а каждый набор назначений опишем *матрицей полных назначений* X (в любой строке и любом столбце матрицы полных назначений есть ровно одна единица):

$$C = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1n} \\ c_{21} & c_{22} & \dots & c_{2n} \\ \dots & \dots & \dots & \dots \\ c_{n1} & c_{n2} & \dots & c_{nn} \end{bmatrix}, \quad X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{22} & \dots & x_{2n} \\ \dots & \dots & \dots & \dots \\ x_{n1} & x_{n2} & \dots & x_{nn} \end{bmatrix},$$

где

$$x_{ik} = \begin{cases} 1, & \text{если } i\text{-е лицо назначено на } k\text{-ю должность,} \\ 0 & \text{в противном случае.} \end{cases}$$

Приходим к задаче оптимизации *функции суммарных затрат*:

$$L(C) = \sum_{i=1}^{n}\sum_{k=1}^{n} c_{ik}x_{ik} \to \min, \ \sum_{k=1}^{n} x_{ik} = 1, \ \sum_{i=1}^{n} x_{ik} = 1, \ \forall i, k, \ x_{ik} \in \{0,1\}. \tag{1}$$

Можно поставить аналогичную задачу, понимая под $c_{ik} \geqslant 0$ прибыль, связанную с назначением i-го лица на k-ю должность. В этом случае цель — максимизировать сумму прибыли:

$$\sum_{i=1}^{n}\sum_{k=1}^{n} c_{ik}x_{ik} \to \max, \ \sum_{k=1}^{n} x_{ik} = 1, \ \sum_{i=1}^{n} x_{ik} = 1, \ \forall i, k, \ x_{ik} \in \{0,1\}. \tag{2}$$

Покажем, как свести задачу (2) к задаче (1). Введем величину $S = \max\limits_{i,k} c_{ik}$ и рассмотрим матрицу $\widehat{C}$ с элементами $\widehat{c}_{ik} = S - c_{ik}$. Поскольку для любой матрицы полных назначений X выполняется равенство

$$\sum_{i=1}^{n}\sum_{k=1}^{n} \widehat{c}_{ik}x_{ik} = S\sum_{i=1}^{n}\sum_{k=1}^{n} x_{ik} - \sum_{i=1}^{n}\sum_{k=1}^{n} c_{ik}x_{ik} = Sn - \sum_{i=1}^{n}\sum_{k=1}^{n} c_{ik}x_{ik},$$

задача (2) с матрицей полезности C эквивалентна задаче (1) с матрицей затрат $\widehat{C}$.

7.2. Частичное назначение с нулевыми затратами. Предположим, что матрица C в задаче (1) содержит некоторое количество

нулевых элементов c_{ik}. Рассмотрим задачу о частичном назначении кандидатов на должности, не требующем никаких затрат (при этом, разумеется, какие-то должности могут остаться вакантными, но мы постараемся минимизировать их число). Эта задача, по существу, представляет собой переформулировку задачи о ладьях.

Напомним, что ладейным набором называется любой (не обязательно максимальный) набор нулевых элементов матрицы, находящихся в разных строках и разных столбцах. Каждый ладейный набор элементов матрицы C определяет *матрицу частичных назначений*,

$$x_{ik} = \begin{cases} 1, & \text{если элемент } c_{ik} = 0 \text{ входит в ладейный набор,} \\ 0 & \text{в противном случае.} \end{cases}$$

Максимальный ладейный набор может быть найден методами, описанными в п. 6.2. Если соответствующая матрица частичных назначений содержит n единиц (т. е. является матрицей полных назначений), то задача (1) оказывается решенной: вакантных должностей не осталось, а нулевые затраты минимальны. Из сказанного ясно, что чем больше нулей в матрице C, тем больше шансов решить задачу о назначениях "ладейным" методом. В случае, когда нулей в матрице C недостаточно, можно увеличить их количество, преобразовав исходную задачу (1) к эквивалентной задаче, с новой матрицей $\tilde{C}$.

7.3. Венгерские преобразования матрицы (Б. Эгервари, 1931 г. [3]). Матрица называется *неотрицательной*, если все ее элементы неотрицательны. Пусть C. $\tilde{C}$ — неотрицательные $n \times n$-матрицы. Говорят, что $\tilde{C}$ получена из C *венгерским преобразованием*, если существуют произвольные числа α_i, β_k, $i,k = 1,\ldots,n$, такие, что $\tilde{c}_{ik} = c_{ik} + \alpha_i + \beta_k$ для $\forall\, i,k$.

Удобно записывать числа α_i справа от матрицы C, а числа β_k — под матрицей C (рис. 20). Для получения матрицы $\tilde{C}$ из матрицы C следует прибавить α_i ко всем элементам матрицы C, стоящим в той же строке, а β_k — ко всем элементам, стоящим в том же столбце.

Легко показать, что функции затрат $L(C)$ и $L(\tilde{C})$ задачи (1) связаны соотношением $L(\tilde{C}) = L(C) + H$, где $H = \sum\limits_{i=1}^{n} \alpha_i + \sum\limits_{k=1}^{n} \beta_k$ не зависит от матрицы X. Отсюда вытекает эквивалентность задачи (1) для C

и $\tilde{C}$. Опишем два типа уменьшающих венгерских преобразований, для которых $H < 0$.

$$\begin{array}{ccc} \begin{bmatrix} c_{11} & \ldots & c_{1q} \\ \vdots & \vdots & \vdots \\ c_{p1} & \ldots & c_{pq} \\ \vdots & \vdots & \vdots \\ c_{n1} & \ldots & c_{nq} \end{bmatrix} & \begin{matrix} \\ \\ \\ \\ \\ \end{matrix} & \begin{bmatrix} c_{1,q+1} & \ldots & c_{1n} \\ \vdots & \vdots & \vdots \\ c_{p,q+1} & \ldots & c_{pn} \\ \vdots & \vdots & \vdots \\ c_{n,q+1} & \ldots & c_{nn} \end{bmatrix} \\ \begin{matrix} \alpha_1 \\ \vdots \\ \alpha_p \\ \vdots \\ \alpha_n \end{matrix} & & \begin{matrix} -d \\ \vdots \\ -d \\ \vdots \\ 0 \end{matrix} \\ \begin{matrix} \beta_1 & \ldots & \beta_q \end{matrix} & & \begin{matrix} 0 & \ldots & 0 \end{matrix} \\ \text{Рис. 20} & & \begin{matrix} +d & \ldots & +d \end{matrix} \\ & & \text{Рис. 21} \end{array}$$

1. В каждой строке матрицы C находим наименьший элемент и вычитаем его из всех элементов этой строки; в полученной матрице делаем такое же преобразование со столбцами. Это *преобразование приведения* — уменьшающее, если в исходной матрице есть хотя бы одна не содержащая нулевых элементов строка или столбец.

2. Предположим, что в матрице C содержится подматрица D размерами $p \times q$ ($p + q > n$), все элементы которой положительны. Пусть d — наименьший элемент этой подматрицы. Вычтем его из элементов тех строк матрицы C, которые пересекают подматрицу D. Далее прибавим d к элементам тех столбцов матрицы C, которые не пересекают подматрицу D (рис. 21; подматрица D расположена в левом верхнем углу матрицы C). Полученная матрица $\tilde{C}$ неотрицательна, и значение

$$H = -pqd + (n-p)(n-q)d = (n-p-q)nd < 0.$$

7.4. Алгоритм решения задачи о назначениях.

1. Применяем к матрице C венгерское преобразование 1-го типа.

2. Для полученной матрицы $\tilde{C}$ решаем задачу о ладьях и задачу о нахождении подматрицы D максимального суммарного размера без нулей, как описано в п. 6.2.

3. Если найден ладейный набор из n элементов, то задача о назначениях решена. Конец алгоритма. В противном случае применяем венгерское преобразование 2-го типа, используя матрицу D. Переходим к п. 2.

Пример 7.1. Решить задачу о назначениях для матрицы затрат C (рис. 22).

$$C = \begin{bmatrix} 1 & 4 & 1 & 1 & 1 & 1 \\ 2 & 6 & 2 & 4 & 5 & 3 \\ 1 & 6 & 1 & 2 & 2 & 2 \\ 3 & 6 & 3 & 3 & 3 & 3 \\ 1 & 4 & 1 & 1 & 1 & 1 \\ 0 & 4 & 1 & 2 & 1 & 1 \\ 0 & -3 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{matrix} -1 \\ -2 \\ -1 \\ -3 \\ -1 \\ 0 \end{matrix}$$

C	v_1	v_2	v_3	v_4	v_5	v_6
u_1	1	4	1	1	1	1
u_2	2	6	2	4	5	3
u_3	1	6	1	2	2	2
u_4	3	6	3	3	3	3
u_5	1	4	1	1	1	1
u_6	0	4	1	2	1	1

Рис. 22

Сделаем венгерское преобразование 1-го типа, из элементов каждой строки вычтем минимальный элемент этой строки. Далее из второго столбца вычтем 3; *константа приведения* равна 11. Матрица $\widetilde{C}$ совпадает с матрицей из примера 6.3 и содержит максимальную положительную подматрицу D. Применяя венгерское преобразование 2-го типа с $d = 1$, получим

$$\widetilde{C} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 3 & 1 \\ 0 & 2 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 2 & 1 & 1 \end{bmatrix} \begin{matrix} -1 \\ -1 \\ \\ \\ -1 \end{matrix} \to \begin{bmatrix} 1 & \underline{0} & 1 & 0 & 0 & 0 \\ \underline{0} & 0 & 0 & 1 & 2 & 1 \\ 0 & 1 & \underline{0} & 0 & 0 & 0 \\ 1 & 0 & 1 & \underline{0} & 0 & 0 \\ 1 & 0 & 1 & 0 & \underline{0} & 0 \\ 0 & 0 & 1 & 1 & 0 & \underline{0} \end{bmatrix}.$$

$$+1 \quad +1$$

В новой матрице очевиден максимальный ладейный набор, определяющий полное назначение без затрат. Выбор в исходной матрице C элементов, стоящих на тех же местах, дает оптимальное назначение с суммой затрат $4+2+1+3+1+1=12$. Соответствующая матрица с выделенными элементами изображена на рис. 22.

Замечание. Решение задачи о назначениях для матрицы C интерпретируется с помощью двудольных графов. Пусть c_{ik} обозначает вес ребра двудольного графа $G = (U, V; X)$, соединяющего вершину u_i с вершиной v_k (см. рис. 22). Выделенные элементы матрицы на рис. 22 дают в G *максимальное паросочетание с минимальным весом*.

8. МЕТОД ВЕТВЕЙ И ГРАНИЦ. ЗАДАЧА О КОММИВОЯЖЕРЕ

8.1. Метод ветвей и границ. Метод ветвей и границ позволяет решить задачу оптимизации $f(s) \to \min$, $s \in S$, последовательно сводя ее к задачам минимизации функций на меньших множествах (как правило, S — конечное множество). На этом пути возникает дерево задач (каждая задача "разветвляется" на две более простые задачи).

Предположения, при которых работает метод, таковы.

Во-первых, предполагается, что можно указать оценку снизу h для функции $f(s)$ ($f(s) \geqslant h$, $s \in S$). Полагая $\varphi(s) = f(s) - h$, приходим к эквивалентной задаче $\varphi(s) \to \min$, $s \in S$; эту задачу назовем *приведенной*. Предположим, что для всех подзадач, возникающих при применении метода, также имеется подобная оценка снизу; эти подзадачи будут заменяться приведенными. Будем считать, что оценка снизу обладает следующим свойством: если подмножество, принадлежащее S, состоит из одного элемента s^*, то $h = f(s^*)$. Тем самым приведенная задача, рассматриваемая на одноэлементном множестве, является тривиальной.

Во-вторых, предполагается, что для каждой приведенной задачи дан набор свойств элементов $s \in S$; назовем эти свойства различающими. Выбрав различающее свойство $\mathcal{P}$, разбиваем множество S на подмножества $S(\mathcal{P}) = \{s | s \in S,\ s$ обладает свойством $\mathcal{P}\}$ и $S(\overline{\mathcal{P}}) = \{s | s \in S,\ s$ обладает противоположным свойством $\overline{\mathcal{P}}\}$. Таким образом, задача $\varphi(s) \to \min$, $s \in S$, "разветвляется", после приведения, на подзадачи $\varphi(s) \to \min$, $s \in S(\mathcal{P})$ и $\varphi(s) \to \min$, $s \in S(\overline{\mathcal{P}})$.

Теперь опишем алгоритм построения дерева задач.

А. Изображаем корень дерева, отмечаем упомянутую оценку h, а в стороне от конструируемого дерева записываем приведенную задачу $\varphi(s) \to \min$, $s \in S$.

Пусть уже построена некоторая часть дерева. Каждой его вершине u поставлена в соответствие оценка h_u (это оценка снизу функции f на некотором подмножестве $S_u \subset S$) и приведенная задача $\varphi_u(s) \to \min$, $s \in S_u$; здесь $\varphi_u(s) = f(s) - h_u$ при $s \in S_u$. Вершина u отмечена числом

h_u, а соответствующая приведенная задача сформулирована. Кроме того, каждое ребро дерева отмечено одним из различающих свойств $\mathcal{P}$ (либо его отрицанием $\overline{\mathcal{P}}$).

Б. Отбираем все висячие вершины u, для которых оценка h_u минимальна. Возможны два случая.

Случай 1. Для каждой отобранной вершины u множество S_u содержит более одного элемента. В этом случае фиксируем какую-либо вершину u из указанных и осуществляем следующие действия.

В. Выбираем произвольное различающее свойство $\mathcal{P}$ для задачи $\varphi_u(s) \to \min$, $s \in S_u$, его отрицание $\overline{\mathcal{P}}$ и находим оценку снизу для задачи $\varphi_u(s) \to \min$, $s \in S_u$, где s обладает свойством $\overline{\mathcal{P}}$. Выбираем то $\mathcal{P}$, для которого эта оценка максимальна; если имеется несколько таких $\mathcal{P}$, то выбираем любое из них.

Г. Вводим две новые вершины дерева v, w, проводим ребра (u, v), (u, w) и отмечаем их символами $\overline{\mathcal{P}}$ и $\mathcal{P}$ соответственно; здесь $\mathcal{P}$ — свойство, выбранное в п. В.

Д (работа с вершиной v). Вводим множество $S_v = \{s | s \in S_u,$ s обладает свойством $\overline{\mathcal{P}}\}$. К числу h_u прибавляем найденную в п. В оценку снизу для функции $\varphi_u(s)$, $s \in S_v$ (эту оценку обозначим через h_v^0). Сумму $h_u + h_v^0$ записываем рядом с вершиной v. Вершине v ставим в соответствие приведенную задачу $\varphi_v(s) \to \min$, $s \in S_v$, где $\varphi_v(s) = \varphi_u(s) - h_v^0$, $s \in S_v$.

Е (работа с вершиной w). Вводим множество $S_w = \{s | s \in S_u,$ s обладает свойством $\mathcal{P}\}$. Находим оценку снизу для функции $\varphi_u(s)$, $s \in S_w$; эту оценку обозначим через h_w^0. Сумму $h_u + h_w^0$ записываем рядом с вершиной w. Вершине w ставим в соответствие приведенную задачу $\varphi_w(s) \to \min$, $s \in S_w$. Переходим к п. Б.

Случай 2. Существует висячая вершина u, для которой множество S_u состоит из единственного элемента. Тогда элемент u есть точка минимума функции $f(s)$, $s \in S$, число h_u — значение минимума. Конец алгоритма.

8.2. Задача о коммивояжере. Коммивояжер должен объехать несколько городов и, посетив каждый из них один раз, вернуться в город, из которого он выехал. Известна стоимость проезда из любого

города в любой другой (стоимость может быть равной ∞, — это означает, что проезд невозможен). Требуется найти маршрут коммивояжера, суммарная стоимость которого минимальна.

Постановка задачи. Дан орграф с множеством вершин $V = \{v_1, \ldots, v_n\}$ и множеством дуг $\vec{X} = \{(v_i, v_j)\}_{i,j=1}^{n,n}$. Дугам графа (v_i, v_k) приписан вес a_{ik}, $0 \leqslant a_{ik} \leqslant \infty$. Запись $a_{ik} = \infty$ означает отсутствие дуги (v_i, v_k). Матрицу $A = [a_{ik}]_{i,k=1}^{n,n}$, составленную из элементов a_{ik}, назовем матрицей весов орграфа $G = (V, \vec{X})$. Требуется найти гамильтонов контур $v_{i_1} v_{i_2} \ldots v_{i_n} v_{i_1}$ с наименьшим весом $a_{i_1 i_2} + \ldots + a_{i_n i_1}$. Задача оптимизации записывается в следующем виде: $f_A(s) \to \min$, $s \in S$, где S — конечное множество всех гамильтоновых контуров, $f_A(s)$ — вес контура s, подсчитанный с использованием матрицы весов A.

Чтобы применить метод ветвей и границ к задаче о коммивояжере, поясним, как определяются приведенная задача, оценка снизу и различающие признаки.

Матрицу весов назовем *приведенной*, если в любой ее строке и любом столбце имеется нуль. Соответствующую задачу о коммивояжере также назовем *приведенной*.

Можно показать, что вес любого гамильтонова контура не меньше константы приведения (определение см. п. 7.3). Таким образом, в качестве оценки снизу в рассматриваемой задаче можно взять константу приведения.

Рассмотрим приведенную матрицу $A_{\text{пр}}$ и соответствующую ей задачу о коммивояжере. Каждому нулевому элементу $a_{ik} = 0$ поставим в соответствие следующее различительное свойство $\mathcal{P}_{ik}$: гамильтонов контур проходит через ребро (v_i, v_k).

Построим дерево задач, следуя алгоритму метода ветвей и границ. Задачи, соответствующие вершинам дерева, также будут задачами о коммивояжере со следующей модификацией: в них требуется отыскать гамильтонов контур с наименьшим весом, содержащий некоторые ребра исходного графа и не содержащий некоторых других его ребер. Эти модифицированные задачи также задаются матрицами весов.

При рассмотрении гамильтоновых циклов, содержащих ребро (v_i, v_k), следует вычеркнуть i-ю строку и k-й столбец из A и решить задачу оптимизации с получившейся матрицей. Если рассматриваются гамильтоновы циклы, не содержащие данное ребро (v_i, v_k), то достаточно в матрице весов A заменить элемент a_{ik} на ∞ и рассмотреть соответствующую задачу.

Итак, вместо самих модифицированных задач о коммивояжере, поставленных в соответствие вершинам дерева, достаточно рассмотреть измененные матрицы весов.

Алгоритм метода ветвей и границ для задачи о коммивояжере (Д. Литтл, 1963 г. [3]). Исходные данные — матрица весов A.

А. По матрице A находим приведенную матрицу $A_{\text{пр}}$ и константу приведения h. Изображаем кружком корень Λ дерева, пишем в нем число h, а в стороне — матрицу $A_{\text{пр}}$.

Пусть уже построена часть дерева. В каждой его вершине u написана оценка h_u, а в стороне — приведенная матрица весов $A_{\text{пр}u}$.

Б. Отбираем все висячие вершины с минимальным значением оценки h_u. Возможны два случая.

Случай 1. Для каждой из отобранных вершин u соответствующая матрица $A_{\text{пр}u}$ имеет порядок больше единицы. Выбираем одну из этих вершин.

В. Для каждого нулевого элемента a_{lm} матрицы $A_{\text{пр}u}$ находим сумму наименьших элементов l-й строки и m-го столбца, отличных от a_{lm}. Эту сумму назовем *степенью элемента* $a_{lm} = 0$. Выбираем элемент a_{ik} с наибольшей степенью. Вводим различающее свойство $\mathcal{P}_{ik}$: гамильтонов контур проходит через ребро (v_i, v_k).

Г. Вводим две новые вершины дерева v, w, проводим ребра (u, v), (u, w) и отмечаем их свойствами $\overline{\mathcal{P}}_{ik}$, $\mathcal{P}_{ik}$ соответственно.

Д (работа с вершиной v). В кружок, обозначающий вершину v, вписываем оценку h_v, равную сумме числа h_u с найденной в п. В максимальной степенью нулевого элемента a_{ik}. Вершине v ставим в соответствие матрицу $A_{\text{пр}v}$, которую строим так: в матрице $A_{\text{пр}u}$ заменяем a_{ik} на ∞, затем приводим получениую матрицу. Матрицу $A_{\text{пр}v}$

записываем в стороне от дерева и отмечаем ее *кодом вершины* u, — цепочкой свойств, написанных около ребер, ведущих от корня дерева Λ к вершине u.

Е (работа с вершиной w). Рассмотрим код вершины w и выпишем из него свойства $\mathcal{P}_{i_1,k_1}, \mathcal{P}_{i_2,k_2}, \ldots, \mathcal{P}_{i_r,k_r}, \mathcal{P}_{i,k}$ (т. е. свойства без знака отрицания). В графе возьмем все ребра (v_l, v_m) такие, что они образуют цикл длиной строго меньше n с некоторыми из ребер (v_{i_1}, v_{k_1}), (v_{i_2}, v_{k_2}), $\ldots$, (v_{i_r}, v_{k_r}), (v_i, v_k). В матрице $A_{\text{пр}u}$ заменяем соответствующие элементы a_{lm} на ∞. Затем удаляем из матрицы i-ю строку и k-й столбец. Для полученной матрицы находим приведенную матрицу $A_{\text{пр}w}$ и константу приведения h^0_w. В кружок, обозначающий вершину w, вписываем оценку $h_w = h^0_w + h_u$. Записываем матрицу $A_{\text{пр}w}$ и отмечаем ее кодом вершины w. Переходим к п. Б.

Случай 2. Среди вершин, отобранных в п. Б, имеется u, для которой матрица $A_{\text{пр}u}$ имеет единственный элемент (равный нулю). Выписываем из кода вершины u свойства без знаков отрицаний $\mathcal{P}_{i_1,k_1}, \ldots, \mathcal{P}_{i,k}$. Ребра $(v_{i_1,k_1}), \ldots, (v_{i,k})$, расположенные в нужной последовательности, образуют искомый гамильтонов контур, а величина h_u есть его вес. Конец алгоритма.

Замечание. Для нахождения в графе гамильтонова контура с максимальным весом используется прием, описанный в п. 7.1.

Пример 8.1. Найдем гамильтонов цикл с наименьшим весом во взвешенном орграфе, изображенном на рис. 23. На рис. 24 изображена матрица весов орграфа.

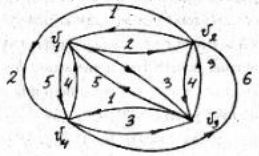


Рис. 23

A	v_1	v_2	v_3	v_4	
v_1	∞	2	3	5	-2
v_2	1	∞	4	2	-1
v_3	5	3	∞	1	-1
v_4	4	6	3	∞	-3
	0	0	0	0	

Рис. 24

$A_{пр}(\Lambda)$	v_1	v_2	v_3	v_4
v_1	∞	$\underline{0}^3$	1	3
v_2	0^2	∞	3	1
v_3	4	2	∞	0^3
v_4	1	3	0^2	∞

Рис. 25

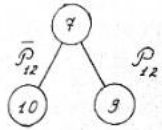


Рис. 26

Приведем матрицу A, получим матрицу $A_{пр}(\Lambda)$, изображенную на рис. 25, и константу приведения, равную 7. Изображаем корень дерева, рис. 26. Запись 0^k означает, что степень этого нулевого элемента матрицы равна k.

В матрице $A_{пр}(\Lambda)$ заменяем отмеченный нуль на ∞ и, приводя полученную матрицу, находим матрицу $A_{пр}(\overline{\mathcal{P}}_{12})$ (рис. 27). В матрице $A_{пр}(\Lambda)$ вычеркиваем строку и столбец с отмеченным нулем. Поскольку добавлением к ребру (v_1, v_2) ребра (v_2, v_1) получается контур $v_1v_2v_1$, содержащий не все вершины графа, то заменяем элемент с индексом (v_2, v_1) на ∞. Приводим полученную матрицу, находим матрицу $A_{пр}(\mathcal{P}_{12})$ (рис. 28) и константу приведения, равную 2.

$A_{пр}(\overline{\mathcal{P}}_{12})$	v_1	v_2	v_3	v_4
v_1	∞	∞	0^2	2
v_2	$\underline{0}^2$	∞	3	1
v_3	4	0^1	∞	0^1
v_4	1	1	0^1	∞

Рис. 27

$A_{пр}(\mathcal{P}_{12})$	v_1	v_3	v_4
v_2	∞	2	0^2
v_3	3	∞	$\underline{0}^3$
v_4	0^3	0^2	∞

$h^0 = 2$

Рис. 28

На рис. 29 показано дерево задач, построенное при решении данной задачи о коммивояжере. Каждой вершине дерева поставлена в соответствие приведенная матрица, отмеченная кодом этой вершины (рис. 30). В каждой матрице отмечен ее нулевой элемент с наибольшей степенью. Ниже каждой матрицы указано значение константы приведения h^0 и указаны элементы, которые заменены на ∞ с целью исключить циклы длины строго меньше 4 при наложенных условиях $\mathcal{P}_{ik}$.

Алгоритм закончился на вершине дерева с кодом $\overline{\mathcal{P}}_{12} \cap \mathcal{P}_{21} \cap \mathcal{P}_{13} \cap \mathcal{P}_{34} \cap \mathcal{P}_{42}$. Искомый гамильтонов контур обладает свойством $\mathcal{P}_{21} \cap \mathcal{P}_{13} \cap \mathcal{P}_{34} \cap \mathcal{P}_{42}$, т. е. имеет вид $v_2v_1v_3v_4v_2$, его вес равен 11 (рис. 31).

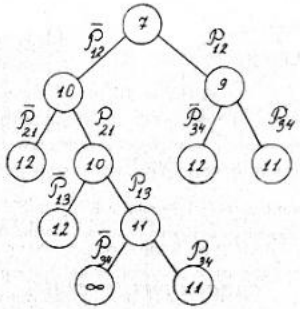


Рис. 29

$A_{пр}(\mathcal{P}_{12} \cap \overline{\mathcal{P}}_{34})$	v_1	v_2	v_3
v_2	∞	2	0
v_3	0	∞	∞
v_4	0	0	∞

$A_{пр}(\mathcal{P}_{12} \cap \mathcal{P}_{34})$	v_1	v_3
v_2	∞	0
v_4	0	∞

$h^0 = 2$, $a_{43} = \infty$

$A_{пр}(\overline{\mathcal{P}}_{12} \cap \overline{\mathcal{P}}_{21})$	v_1	v_2	v_3	v_4
v_1	∞	∞	0	2
v_2	∞	∞	2	0
v_3	3	0	∞	0
v_4	0	1	0	∞

$A_{пр}(\overline{\mathcal{P}}_{12} \cap \mathcal{P}_{21})$	v_2	v_3	v_4
v_1	∞	$\underline{0}^2$	2
v_3	0^1	∞	0^2
v_4	1	0^1	∞

$h^0 = 0$

$A_{пр}(\ldots \cap \overline{\mathcal{P}}_{13})$	v_2	v_3	v_4
v_1	∞	∞	0
v_3	0	∞	0
v_4	1	0	∞

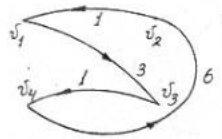
$A_{пр}(\ldots \cap \mathcal{P}_{13})$	v_2	v_4
v_3	∞	$\underline{0}^\infty$
v_4	0^∞	∞

$h^0 = 1$, $a_{32} = \infty$

$A_{пр}(\ldots \cap \mathcal{P}_{34})$	v_2
v_4	0

$h^0 = 0$

Рис. 30



A	v_1	v_2	v_3	v_4
v_1	∞	2	3	5
v_2	1	∞	4	2
v_3	5	3	∞	1
v_4	4	6	3	∞

Рис. 31

Задачу оптимизации $f(s) \to \min$, $s \in S$, где S — конечное множество, число элементов которого невелико, можно решить методом перебора [5].

СПИСОК ЛИТЕРАТУРЫ

1. Гаврилов Г.П., Сапоженко А.А. Задачи и упражнения по курсу дискретной математики. 2-е изд. М.: Наука, 1992. 406 с.

2. Лекции по теории графов /В.А. Емеличев, О.И. Мельников, В.И. Сарванов, Р.И. Тышкевич. М.: Наука, 1990. 384 с.

3. Кофман А. Введение в прикладную комбинаторику /Пер. с франц. В.П. Мякишева и В.Е. Тараканова. Под ред. Б.А. Севастьянова. М.: Наука, 1975. 480 с.

4. Смольяков Э.Р. Введение в теорию графов. М.: Изд-во МГТУ им. Н.Э. Баумана, 1992. 32 с.

5. Исмагилов Р.С., Калинкин А.В., Станцо В.В. Комбинаторика и булевы функции. М.: Изд-во МГТУ им. Н.Э. Баумана, 1998. 40 с.

ТИПОВОЙ РАСЧЕТ ПО ТЕОРИИ ГРАФОВ

На рисунках приведены 30 вариантов ориентированного графа $G(V, \vec{X})$ (запись вида $G(n, m)$ обозначает граф с n вершинами и m ребрами). Соответствующий ассоциированный граф обозначается $G(V, X)$. Матрицы весов графов, рассматриваемых в задачах 8, 9, приведены на рис. a–m.

1. Для графа $G(V, X)$ выписать, перенумеровав вершины: а) множество вершин V и множество ребер X; б) степени вершин; в) списки смежности; г) матрицу инцидентности; д) матрицу весов. Выписать матрицу смежности для графа $G(V, \vec{X})$.

2. Найти диаметр $D(G)$, радиус $R(G)$, количество центров $Z(G)$ для графа $G(V, X)$. Указать вершины, являющиеся центрами графа.

3. Найти остов с минимальным весом для графа $G(V, X)$.

4. Найти дерево кратчайших путей из вершины α для графа $G(V, X)$.

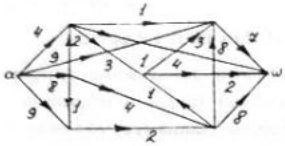
5. Найти максимальный поток в сети $G(V, \vec{X})$, с выделенными вершинами α и ω. Указать разрез с минимальным весом.

6. Найти в графе $G(V, X)$ эйлерову цепь. Если такой цепи не существует, то в графе $G(V, X)$ удалить наименьшее число ребер таким образом, чтобы в новом графе эйлерову цепь можно было указать (записать соответствующий путь).

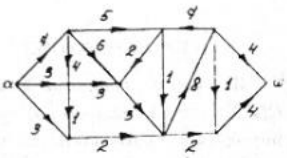
7. Найти в графе $G(V, X)$ гамильтонов цикл. В графе $G(V, \vec{X})$ изменить ориентацию наименьшего числа ребер таким образом, чтобы в новом графе можно было указать гамильтонов контур (записать соответствующий путь).

8. Решить задачу о назначениях по заданной матрице весов двудольного графа (табл. П1). Дать графическую интерпретацию ответа.

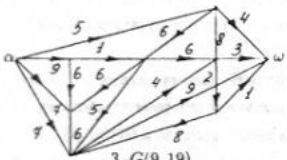
9. Найти гамильтонов контур с минимальным (максимальным) весом для ориентированного графа, заданного матрицей весов (табл. П2). Дать графическую интерпретацию ответа.



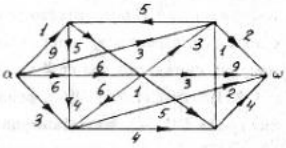
1. $G(9,19)$



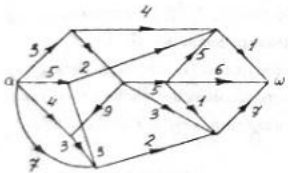
2. $G(10,18)$



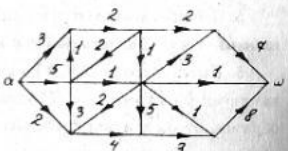
3. $G(9,19)$



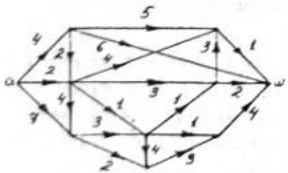
4. $G(9,20)$



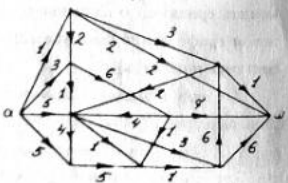
5. $G(10,18)$



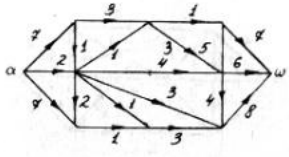
6. $G(10,19)$



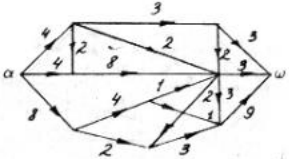
7. $G(10,20)$



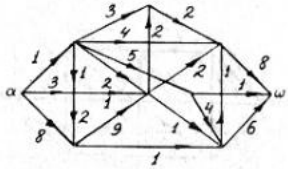
8. $G(10,21)$



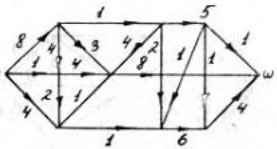
9. $G(10,19)$



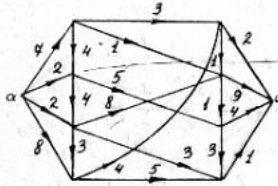
10. $G(10,18)$



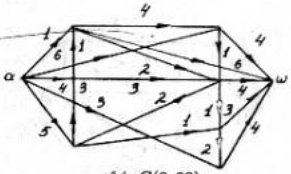
11. $G(10,21)$



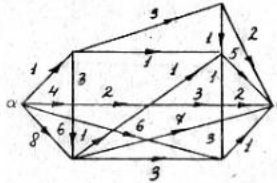
12. $G(10,19)$



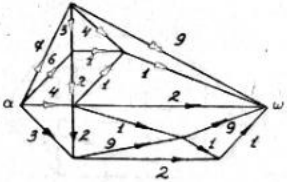
13. $G(10,21)$



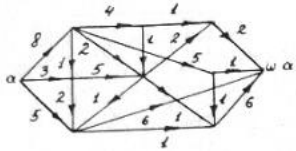
14. $G(9,20)$



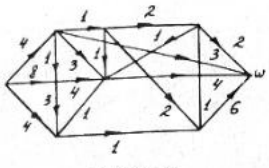
15. $G(10,21)$



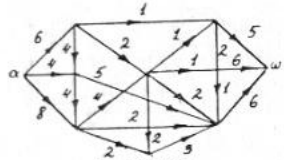
16. $G(9,19)$



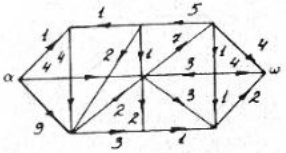
17. $G(10,20)$



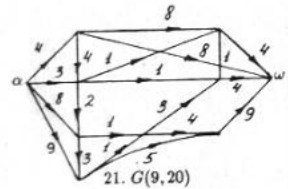
18. $G(9,19)$



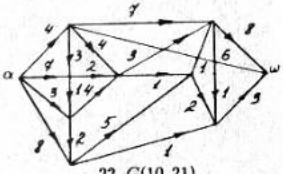
19. $G(10,21)$



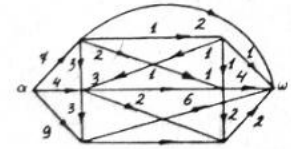
20. $G(10,20)$



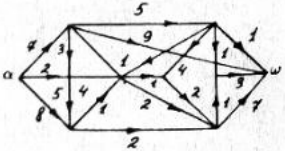
21. $G(9,20)$



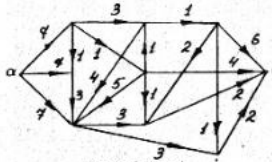
22. $G(10,21)$



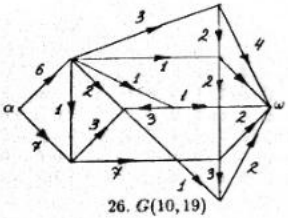
23. $G(9,20)$



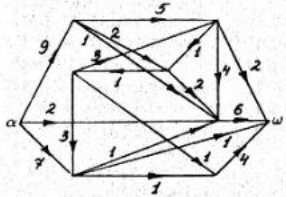
24. $G(10,21)$



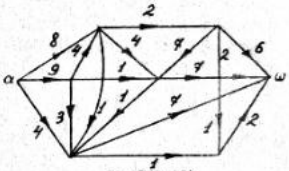
25. $G(10,20)$



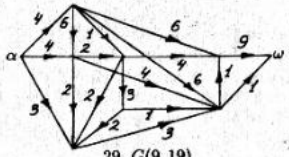
26. $G(10,19)$



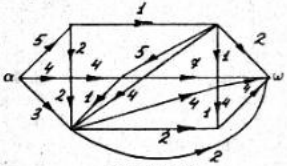
27. $G(9,19)$



28. $G(9,19)$



29. $G(9,19)$



30. $G(9,19)$

∞	0	5	4	0	6
5	∞	3	0	1	6
3	2	∞	0	3	2
2	5	0	∞	6	0
0	4	2	0	∞	2
0	1	6	4	6	∞

а

∞	3	4	5	1	8	7	
10	∞	1	5	6	7	8	
3	4	∞	6	7	8	1	
5	6	9	∞	10	11	12	
1	2	8	9	∞	3	4	
5	6	8	9	1	∞	3	
3	1	2	1	2	2	∞	

б

∞	5	2	7	9
3	∞	8	6	2
7	3	∞	9	9
8	8	11	∞	5
3	6	4	6	∞

в

∞	5	7	9	0	2	3
0	∞	4	3	2	5	0
6	0	∞	7	1	3	8
9	0	4	∞	5	0	2
3	5	0	2	∞	1	7
2	6	3	4	9	∞	0
5	2	9	0	4	3	∞

г

3	0	7	4	0	11	9	2
0	8	5	6	0	7	0	3
4	5	0	3	6	9	0	4
0	3	6	8	5	0	3	0
7	0	11	0	7	4	0	5
9	2	0	9	6	0	3	0
2	0	2	7	0	8	5	6
1	7	3	0	4	9	0	2

д

4	28	16	6	0	0	0	0
3	2	5	7	0	3	0	0
4	13	4	0	18	9	0	0
28	0	∞	6	3	9	0	0
0	5	2	6	∞	17	0	0
19	4	∞	5	33	∞	0	0
0	0	2	8	7	0	0	0
13	31	∞	0	3	4	0	0

е

3	7	9	4	∞
1	5	∞	0	8
7	∞	3	∞	2
6	3	∞	0	0
2	8	∞	1	∞

ж

3	6	10	4	7	11
4	2	3	1	7	9
3	2	6	3	4	10
8	7	10	9	6	7
9	8	9	5	6	5
2	8	7	5	3	8

з

2	3	1	∞	7	10	5
7	∞	6	8	5	5	7
5	6	9	7	∞	3	4
9	5	10	2	10	∞	∞
8	5	8	1	10	10	10
0	0	0	0	0	0	0
0	0	0	0	0	0	0

и

∞	24	18	22	31	19
15	∞	19	27	26	32
22	23	∞	23	16	29
24	31	18	∞	19	13
23	18	34	20	∞	31
24	12	17	15	10	∞

к

3	3	2	0	8	0
10	1	8	9	1	0
10	3	6	8	4	0
7		8	4	6	9	0
1		9	4	3	8	0
9		2	3	6	1	0

л

4		9	1	8	6	5
23	6	2		3		3
7		2	0		21	3
4		8	7		6	28
4		9	1	8	3	2

м

2	0	0	0	0
0	7	9	8	6
0	1	3	2	2
0	8	7	6	4
0	7	6	8	3

н

∞	3	7	2	∞	11
8	∞	0	∞	4	3
6	0	∞	7	∞	2
6	∞	13	∞	5	∞
3	3	3	4	∞	5
8	6	∞	2	2	∞

л

2	1	3	6	9	8
3	4	2	8	0	0
5	2	11	6	4	9
5	3	0	0	3	14
0	0	0	0	0	0
0	0	0	0	0	0

м

∞	1	3	4	7	9	1
5	∞	6	7	1	3	4
5	8	∞	3	4	2	1
2	3	8	∞	9	1	5
3	7	7	8	∞	1	2
4	5	8	9	1	∞	9
8	3	2	4	5	6	∞

р

∞	5	6	1	2	3	4
4	∞	5	6	1	2	3
7	6	∞	5	8	9	10
11	1	5	∞	6	8	9
1	3	4	5	∞	1	2
4	5	8	9	10	∞	11
3	4	5	6	8	9	∞

с

∞	6	1	5	2	4	3
1	∞	6	5	2	5	1
3	8	∞	6	7	1	4
4	5	6	∞	1	2	∞
8	1	3	4	∞	5	3
3	2	1	3	2	∞	1
1	2	3	3	2	1	∞

т

Таблица П1

Вар.	Найти назначение	Рис.	Заменить элементы матрицы
1	максимальное	*м*	$v_{23} = 20,\ v_{42} = 25$
2	максимальное	*б*	
3	минимальное	*а*	
4	мнимальное	*е*	$v_{32} = 10,\ v_{41} = 19,\ v_{64} = 7$
5	максимальное	*ж*	$v_{42} = 7$
6	максимальное	*о*	
7	минимальное	*м*	
8	максимальное	*л*	
9	минимальное	*ж*	
10	минимальное	*м*	$v_{51} = 2,\ v_{52} = 2,\ v_{53} = 2,\ v_{54} = 2$
11	минимальное	*и*	
12	максимальное	*а*	$v_{36} = 6,\ v_{46} = 6,\ v_{56} = 6,\ v_{61} = 6$
13	минимальное	*п*	
14	максимальное	*г*	$v_{42} = 7,\ v_{54} = 1,\ v_{55} = 2,\ v_{67} = 4$
15	минимальное	*к*	
16	максимальное	*к*	$v_{21} = 9,\ v_{22} = 9,\ v_{24} = 9,\ v_{25} = 9$
17	минимальное	*б*	$v_{13} = 5,\ v_{14} = 5,\ v_{35} = 1$
18	максимальное	*ж*	$v_{13} = 7,\ v_{35} = 8,\ v_{45} = 8,\ v_{54} = 8$
19	минимальное	*к*	$v_{25} = 3,\ v_{23} = 3,\ v_{34} = 3,\ v_{43} = 6$
20	максимальное	*д*	
21	минимальное	*о*	
22	максимальное	*м*	$v_{12} = 7,\ v_{34} = 8,\ v_{51} = 4,\ v_{52} = 7$
23	минимальное	*з*	
24	максимальное	*д*	$v_{26} = 11,\ v_{48} = 11,\ v_{73} = 5,\ v_{77} = 9$
25	минимальное	*к*	
26	минимальное	*и*	$v_{23} = 1,\ v_{33} = 1,\ v_{43} = 1,\ v_{53} = 1$
27	минимальное	*л*	$v_{52} = 1,\ v_{53} = 1,\ v_{54} = 1,\ v_{55} = 1$
28	максимальное	*з*	$v_{14} = 11,\ v_{24} = 5,\ v_{43} = 8,\ v_{54} = 2$
29	минимальное	*е*	
30	минимальное	*ж*	$v_{14} = 2,\ v_{41} = 0,\ v_{42} = 0$

Таблица П2

Вар.	Найти контур	Рис.	Заменить элементы матрицы
1	минимальный	*к*	$v_{23} = 20,\ v_{42} = 25,\ v_{61} = 12,\ v_{56} = 30$
2	максимальный	*т*	
3	максимальный	*о*	$v_{15} = 13,\ v_{25} = 5,\ v_{51} = 4,\ v_{12} = 7$
4	максимальный	*б*	
5	минимальный	*р*	$v_{13} = 8,\ v_{23} = 6,\ v_{51} = 8$
6	максимальный	*р*	
7	минимальный	*т*	$v_{52} = 7,\ v_{45} = 4,\ v_{27} = 5$
8	максимальный	*г*	$v_{21} = 1,\ v_{31} = 4,\ v_{32} = 4,\ v_{42} = 5$
9	минимальный	*р*	
10	максимальный	*д*	$v_{12} = 8,\ v_{52} = 4,\ v_{25} = 7,\ v_{27} = 8$
11	минимальный	*о*	
12	максимальный	*о*	$v_{46} = 3,\ v_{35} = 9,\ v_{63} = 4,\ v_{42} = 3$
13	минимальный	*с*	$v_{13} = 6,\ v_{32} = 1,\ v_{51} = 4,\ v_{62} = 7$
14	максимальный	*о*	$v_{12} = 6,\ v_{31} = 6,\ v_{52} = 5,\ v_{53} = 7$
15	минимальный	*к*	$v_{13} = 20,\ v_{35} = 18,\ v_{53} = 20,\ v_{64} = 18$
16	максимальный	*р*	$v_{14} = 1,\ v_{36} = 8,\ v_{54} = 1,\ v_{65} = 5$
17	минимальный	*к*	
18	максимальный	*а*	$v_{16} = 6,\ v_{31} = 4,\ v_{51} = 3$
19	минимальный	*о*	$v_{42} = 3,\ v_{24} = 3,\ v_{63} = 5,\ v_{46} = 10$
20	максимальный	*с*	$v_{12} = 6,\ v_{32} = 1,\ v_{54} = 3,\ v_{61} = 4$
21	минимальный	*э*	
22	максимальный	*в*	$v_{12} = 8,\ v_{34} = 9,\ v_{51} = 6,\ v_{52} = 8$
23	минимальный	*г*	$v_{14} = 3,\ v_{32} = 1,\ v_{67} = 2,\ v_{74} = 3$
24	максимальный	*с*	
25	максимальный	*э*	
26	максимальный	*д*	$v_{15} = 1,\ v_{36} = 8,\ v_{53} = 12,\ v_{63} = 3$
27	минимальный	*б*	
28	максимальный	*о*	$v_{15} = 2,\ v_{35} = 4,\ v_{51} = 1,\ v_{52} = 4$
29	минимальный	*с*	
30	минимальный	*к*	$v_{26} = 27,\ v_{42} = 30,\ v_{31} = 17,\ v_{61} = 27$

ОГЛАВЛЕНИЕ