М. Н. Кирсанов

ГРАФЫ В MAPLE

Задачи, алгоритмы, программы

Пособие по дискретной математике для студентов университетов

МОСКВА
ФИЗМАТЛИТ
2007

УДК 519.17+681.3.06
ББК 22.213
К 435

Кирсанов М. Н. Графы в Maple. Задачи, алгоритмы, К 435 программы. — М.: Издательство ФИЗМАТЛИТ, 2007. — 175 с. — ISBN 5-7046-1168-0.

Изложены решения задач теории графов. Даны описания основных алгоритмов на графах и тексты более 30 программ. Приведены алгоритмы теории искусственного интеллекта (муравьиный алгоритм и метод отжига) для решения задачи коммивояжера. Предметно-именной указатель на 500 терминов и имен может служить справочником по теории графов и командам Maple.

Книга предназначена как для очного, так и для дистанционного обучения.

Для студентов и преподавателей университетов и технических вузов.

УДК 531.3
ББК 22.213

ISBN 5-7046-1168-0

СОДЕРЖАНИЕ

Предисловие

Beauty is the wonder of wonders. It is only the shallow people who do not judge by appearances.
Oscar Wilde [1]

Теория графов — один из разделов современной математики, имеющий большое прикладное значение. Проблемы оптимизации тепловых, газовых и электрических сетей, вопросы совершенствования алгоритмов и создание новых химических соединений связаны с фундаментальными свойствами таких абстрактных математических объектов, как графы. Долгое время задачи теории графов решались вручную, с появлением компьютеров появилась возможность написания специальных программ на алгоритмических языках. Позднее появились пакеты аналитических вычислений Mathematica, MATLAB [9], Mathcad [26] и Maple [8], позволяющие выполнять аналитические символьные преобразования. Для решения задач, объектами которых являются графы, эти пакеты просто незаменимы. В системе Maple есть еще и специализированная библиотека `networks`, составленная из операторов для работы с графами. Таких операторов немного — всего 66, и число это в Maple не меняется от версии к версии [2]. Появилась проблема — описать пакет `networks` и дать примеры решения задач с его помощью. Решению этой проблемы и посвящена данная книга.

В первой части книги даны сведения из теории и решения некоторых основных задач теории графов. Во второй части содержатся готовые программы, разработанные автором как с привлечением операторов и команд пакета `networks`, так и решенные независимым образом на языке Maple. В связи с упоминанием языка представления программ следует уделить немного внимания библиографическому обзору. Дело в том, что во многих книгах и учебниках по дискретной математике даны тексты программ. Однако появилась тенденция печатать их на некотором мертвом условном языке, похожем на Pascal. К сожалению, очень часто такие программы плохо читаются, так как некоторые «команды» понятны только для посвященных (обычно

[1] «Красота — чудо из чудес. Только поверхностные люди не судят по внешности.» (Оскар Уайльд).

[2] На 2006 год последняя версия — Maple 10.

это автор книги). Приятным исключением являются книга Зубова В.С., Шевченко И.В. [11] и отчасти книга Иванова Б.Н. [13]. Книг с программами по дискретной математике на языке Maple нет, и, вероятно, это издание является первым.

В книге описаны почти все команды и процедуры пакета теории графов `networks`. Исключение составляют только вероятностные задачи на графах. Кроме того, добавлены некоторые новые собственные процедуры. В частности, введена процедура рисования орграфа. При выборе стиля программирования автор добивался ясности изложения алгоритма, иногда даже в ущерб краткости и скорости работы программы. Совершенствование программ предлагается читателям, тем более, что данная книга — учебник, а учиться лучше всего, программируя самому и убеждаясь, что «моя программа лучше!». Желаем читателям успехов в этом!

Автор благодарит студентов МЭИ(ТУ) Александрова В.А., Ульянова Р.В. и Сутяпова А.В. за программы 15, 16, 19, 20, 28 и 32 сборника.

Архив всех текстов программ из книги можно взять на сайте автора http://vuz.exponenta.ru.

Автор будет благодарен всем, кто пришлет свои замечания о книге по адресу mpei2004@yandex.ru.

Глава 1

НЕОРИЕНТИРОВАННЫЕ ГРАФЫ

Основные определения. *Граф* $G(V, E)$ — совокупность двух множеств: вершин V и ребер E, между которыми определено отношение инцидентности. Каждое ребро e из E инцидентно ровно двум вершинам, v' и v'', которые оно соединяет. При этом вершина v' и ребро e называются *инцидентными* друг другу, а вершины v' и v'' называются *смежными*. Часто пишут v', v'' из G и e из G. Принято обозначение n для числа вершин графа (мощность множества V): $|V(G)| = n$, и m для числа его ребер: $|E(G)| = m$. Говорят, что граф G есть (n, m) граф, где n — *порядок* графа, m — *размер* графа.

Если все ребра (v_1, v_2) графа неориентированные, т.е. пары вершин, определяющие элементы множества E, неупорядочены, то такой граф называется неориентированным графом, или *неографом*.

Маршрут — последовательность ребер, в которой каждые два соседних ребра имеют общую вершину.

Маршрут в неографе, в котором все ребра разные, — *цепь*.

Граф *связен*, если любые две вершины соединены хотя бы одним маршрутом. Число ребер маршрута определяет его длину.

Цепь в графе называется *полуэйлеровой* (эйлеровой), если она содержит все ребра и все вершины графа.

Ребра, инцидентные одной паре вершин, называются параллельными или *кратными*.

Граф с кратными ребрами называется *мультиграфом*.

Ребро (v, v) называется *петлей* (концевые вершины совпадают). Граф, содержащий петли (и кратные ребра), называется *псевдографом*.

Степень $\deg(v)$ вершины — число ребер, инцидентных v. *В неографе сумма степеней всех вершин равна удвоенному числу ребер* (лемма о рукопожатиях):

$$\sum_{i=1}^{n} \deg(v_i) = 2m.$$

Петля дает вклад, равный 2, в степень вершины.

Степенная последовательность — последовательность степеней всех вершин графа, записанная в определенном порядке (по возрастанию или убыванию).

Матрица смежности графа — квадратная матрица A порядка n, где элемент a_{ij} равен числу ребер, соединяющих вершины i и j.

С графом связывают также матрицу *инцидентности* I. Число строк этой матрицы равно числу вершин, число столбцов — числу ребер; $i_{ve} = 1$, если вершина v инцидентна ребру e; в противном случае $i_{ve} = 0$. В каждом столбце матрицы инцидентности простого графа (без петель и без кратных ребер) содержится по две единицы. Число единиц в строке равно степени соответствующей вершины.

Граф $H(V_1, E_1)$ называется *подграфом* графа $G(V, E)$, если $V_1 \subseteq V$, $E_1 \subseteq E$. Если $V_1 = V$, то подграф называется *остовным*.

Компонента связности графа — максимальный по включению вершин и ребер связный подграф.

Ранг графа $\nu^* = n - k$, где k — число компонент связности [1].

Дерево — связный граф, содержащий $n - 1$ ребро [2].

1.1. Радиус и диаметр графа. Эйлерова цепь

Вычисление расстояний и определение маршрутов в графе являются одной из наиболее очевидных и практичных задач на графе. С одной из таких задач началась теория графов [3].

Введем некоторые необходимые определения.

Эксцентриситет вершины графа — расстояние до максимально удаленной от нее вершины.

Радиус графа — минимальный эксцентриситет вершин, а *диаметр* графа — максимальный эксцентриситет вершин.

Центр графа образуют вершины, у которых эксцентриситет равен радиусу. Центр графа может состоять из одной, нескольких или всех вершин графа.

Периферийные вершины имеют эксцентриситет, равный диаметру.

Простая цепь с длиной, равной диаметру графа, называется *диаметральной*.

Теорема 1. *В связном графе диаметр не больше ранга его матрицы смежности.*

Теорема 2 (Жордана [4])**.** *Каждое дерево имеет центр, состоящий из одной или двух смежных вершин.*

---

[1] Величина $\nu^* = n - k$ называется также коциклическим рангом графа [10]. Часто ранг графа связывается с рангом матрицы смежности ($\operatorname{rank} G = \operatorname{rank} A$). Будем придерживаться терминологии, принятой в Maple: $\operatorname{rank} G = \nu^* = n - k$.

[2] Подробнее см. с. 41.

[3] Задача Л. Эйлера о кенигсбергских мостах (1736 г.) [30].

[4] Jordan C.

Теорема 3. *Если диаметр дерева четный, то дерево имеет единственный центр и все диаметральные цепи проходят через него, если диаметр нечетный, то центров два и все диаметральные цепи содержат ребро, их соединяющее.*

Очевидно практическое значение центра графа. Если, например, речь идет о графе дорог с вершинами-городами, то в математическом центре целесообразно размещать административный центр, складские помещения и т.п. [1] В данной задаче расстояние оценивается в числе ребер. Этот же алгоритм можно применять и для взвешенного графа, где расстояния — это веса ребер. В качестве веса можно брать евклидовы расстояния (для *евклидовых* графов с вершинами, являющимися точками на плоскости или в пространстве), время или стоимость передвижения между пунктами. Для несвязных графов в указанном смысле центр не определяется.

Для проверки существования эйлеровой цепи используется известная теорема.

Теорема 4 (Эйлера [2])**.** *Мультиграф обладает эйлеровой цепью тогда и только тогда, когда он связен и число вершин нечетной степени равно* 0 *или* 2.

Вершины нечетной степени в этой теореме, очевидно, являются началом и концом цепи. Если таких вершин нет, то эйлерова цепь становится *эйлеровым циклом*. Граф, обладающий эйлеровым циклом, называется *эйлеровым*. Заметим, что в задаче о кенигсбергских мостах разыскивался именно эйлеров цикл — по условию требовалось пройти по всем семи мостам города Кенигсберга [3] через реку Преголь по одному разу и вернуться к исходной точке.

Наряду с эйлеровыми циклами [30] представляют интерес гамильтоновы циклы — простые циклы, проходящие через все вершины графа [4].

Задача. Найти радиус r, диаметр d и центр графа (рис. 1.1). Проверить наличие эйлеровой цепи.

---

[1] *Кирсанов М.Н.* Математический центр московского метро //Exponenta Pro. Математика в приложениях. 2004. №4(4).

[2] Euler L.

[3] Сейчас это г. Калининград.

[4] Подробнее см. с. 84.

а *б* *в*

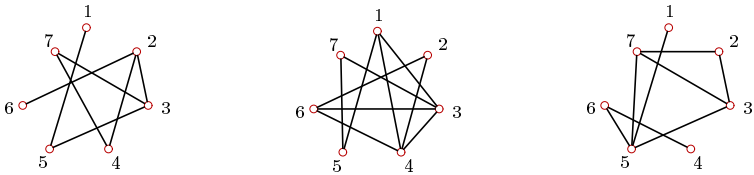
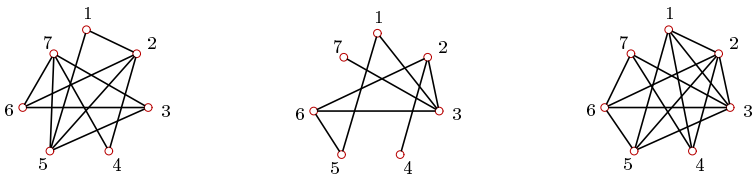


Рис. 1.1

г *д* *е*



ж *з* *и*

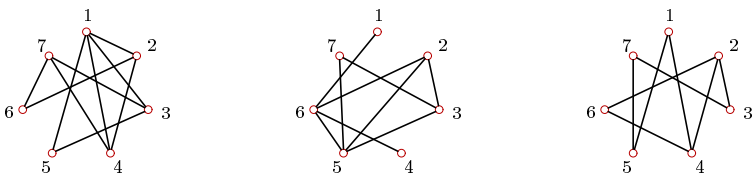


Рис. 1.1 (продолжение)

Ответы

	r	d	Центр	Эйлерова цепь
а	2	4	3	Нет
б	2	3	1, 3, 4	Есть
в	2	4	5	Нет
г	2	2	1, 2, 3, 4, 5, 6, 7	Есть
д	2	3	2, 3, 6	Нет
е	2	2	1, 2, 3, 4, 5, 6, 7	Нет
ж	2	3	1, 2, 3, 4, 7	Нет
з	2	3	2, 5, 6	Нет
и	3	3	1, 2, 3, 4, 5, 6, 7	Есть

Пример. Дан неограф (рис. 1.2). Найти радиус, диаметр и центр графа. Проверить наличие эйлеровой цепи.

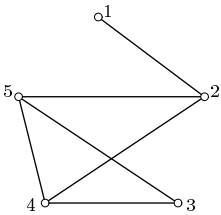


Рис. 1.2

Решение. Радиус и диаметр (способ 1). Находя цепи наименьшей длины, вычислим расстояния между вершинами. Результаты занесем в матрицу расстояний. Для неографа эта матрица симметричная.

Для сокращения вычислений находим элементы половины матрицы, заполняя другую половину из условия симметрии:

$$S = \begin{vmatrix} 0 & 1 & 3 & 2 & 2 \\ 1 & 0 & 2 & 1 & 1 \\ 3 & 2 & 0 & 1 & 1 \\ 2 & 1 & 1 & 0 & 1 \\ 2 & 1 & 1 & 1 & 0 \end{vmatrix}. \qquad (1.1)$$

Вычисляем эксцентриситет ε каждой вершины. Эту величину можно определять как максимальный элемент соответствующего столбца (или строки) матрицы расстояний. Получаем

№	ε
1	3
2	2
3	3
4	2
5	2

.

Радиус графа r — минимальный эксцентриситет вершин. В данном случае $r = 2$. Такой эксцентриситет имеют вершины № 2, № 4 и № 5. Эти вершины образуют центр графа. Диаметр графа d — максимальный эксцентриситет вершин. В данном случае $d = 3$. Такой эксцентриситет имеют вершины № 1 и № 3; это периферия графа. В исследованном графе вершины оказались либо центральными, либо периферийными. В графах большего порядка существуют и другие вершины.

Эффективный алгоритм Уоршолла и Флойда для определения кратчайших путей между всеми парами вершин графа, реализованный в Maple, приведен на с. 99.

Радиус и диаметр (способ 2). Эксцентриситеты вершин небольшого графа легко вычислять непосредственным подсчетом по рисунку. Однако не всегда граф задан своим рисунком. Кроме того, граф может иметь большой размер. Поэтому необходим другой способ решения задачи. Известна следующая теорема.

Теорема 5. *Пусть $A = \{\alpha_{ij}\}$ — матрица смежности графа G без петель и $A^k = \{\beta_{ij}\}$, где $k \in N$. Тогда β_{ij} равно числу маршрутов длины k от вершины v_i к вершине v_j* [1].

Решение задач теории графов с помощью различных преобразований матрицы смежности называют алгебраическим методом. Как правило, алгебраические методы годятся для графов небольшого порядка и особенно удобны для применения пакетов символьного вычисления или компьютерных систем со встроенными матричными алгоритмами.

Построим матрицу смежности графа:

$$A = \begin{vmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{vmatrix}.$$

Будем заполнять матрицу расстояний, рассматривая степени матрицы смежности [2]. В первом приближении единицы матрицы смежности показывают пары вершин, расстояние между которыми равно 1 (т.е. они соединены одним ребром):

$$S_{(1)} = \begin{vmatrix} 0 & 1 & - & - & - \\ 1 & 0 & - & 1 & 1 \\ - & - & 0 & 1 & 1 \\ - & 1 & 1 & 0 & 1 \\ - & 1 & 1 & 1 & 0 \end{vmatrix}.$$

Диагональные элементы матрицы расстояний — нули. Умножаем матрицу смежности на себя:

$$A^2 = \begin{vmatrix} 1 & 0 & 0 & 1 & 1 \\ 0 & 3 & 2 & 1 & 1 \\ 0 & 2 & 2 & 1 & 1 \\ 1 & 1 & 1 & 3 & 2 \\ 1 & 1 & 1 & 2 & 3 \end{vmatrix}.$$

Согласно теореме между вершинами 2 и 3, 1 и 4 и т. д. имеется некоторое число маршрутов длиной 2 (поскольку степень матрицы 2). Число маршрутов здесь не используется, важен сам факт наличия маршрута и его длина, на что и указывает ненулевой элемент степени матрицы, не совпадающий с элементом, отмеченным при вычислении

---

[1] Такие маршруты часто называют (v_i, v_j)-маршрутами.

[2] Для вещественной симметрической матрицы A справедливо представление $A = B^{-1}CB$, где C — диагональная матрица. Возведение в степень при этом упрощается: $A^k = B^{-1}C^kB$. Диагональ матрицы C совпадает с собственными числами матрицы A (спектр графа).

маршрута меньшей длины. Проставляем 2 в незаполненные элементы матрицы расстояний и получаем следующее приближение:

$$S_{(2)} = \begin{vmatrix} 0 & 1 & - & 2 & 2 \\ 1 & 0 & 2 & 1 & 1 \\ - & 2 & 0 & 1 & 1 \\ 2 & 1 & 1 & 0 & 1 \\ 2 & 1 & 1 & 1 & 0 \end{vmatrix}.$$

Осталось неизвестным расстояние между вершинами 1 и 3. Будем умножать матрицу смежности A саму на себя до тех пор, пока в матрице A^k не появится ненулевой элемент $a_{1,3}$. Тогда соответствующий элемент матрицы расстояний (или ее приближения) равен степени матрицы смежности: $s_{1,3} = k$. Получаем

$$A^3 = \begin{vmatrix} 0 & 3 & 2 & 1 & 1 \\ 3 & 2 & 2 & 6 & 6 \\ 2 & 2 & 2 & 5 & 5 \\ 1 & 6 & 5 & 4 & 5 \\ 1 & 6 & 5 & 5 & 4 \end{vmatrix},$$

следовательно, $s_{1,3} = 3$, и окончательно

$$S_{(3)} = \begin{vmatrix} 0 & 1 & 3 & 2 & 2 \\ 1 & 0 & 2 & 1 & 1 \\ 3 & 2 & 0 & 1 & 1 \\ 2 & 1 & 1 & 0 & 1 \\ 2 & 1 & 1 & 1 & 0 \end{vmatrix}.$$

Полученная матрица совпадает с матрицей расстояний S (1.1), найденной непосредственными вычислениями по рисунку.

Эйлерова цепь. Граф на рис. 1.2 связен. Любые две пары вершин соединены непрерывной последовательностью ребер. Найдем степени вершин графа. Вершина № 1 имеет степень 1 (ей инцидентно одно ребро), вершина № 3 — 2, остальные вершины — степень 3. В данном случае в графе четыре вершины нечетной степени (№ 1, 2, 4, 5). Следовательно, согласно теореме 4, граф эйлеровой цепью не обладает, т.е. нет цепи, содержащей все ребра графа по одному разу.

Рассмотрим для сравнения граф, обладающий эйлеровой цепью. В графе на рис. 1.3 две вершины (№1 и 3) имеют нечетную степень — 3, следовательно, эйлерова цепь есть. Найти эту цепь — другая задача. Беспорядочное блуждание по графу не дает результата. Более того, цепей может быть несколько. Например, цепь 1–2–3–4–1–3 (рис. 1.4) и цепь 1–2–3–1–4–3 (рис. 1.5).

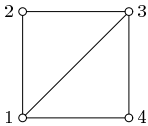


Рис. 1.3

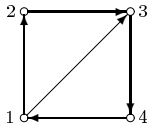


Рис. 1.4

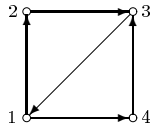


Рис. 1.5

Оценить число эйлеровых цепей можно, используя следствие из теоремы 4. Введем необходимое определение.

Говорят, что реберно-непересекающиеся цепи *покрывают* граф, если каждое ребро графа входит в одну из них [10].

Следствие. *В связном графе, содержащем k вершин нечетной степени, минимальное число покрывающих его реберно-непересекающихся цепей равно $k/2$.*

Если вершины графа удовлетворяют теореме 4, то множество покрывающих реберно-непересекающихся цепей состоит из одной — эйлеровой. В том числе и для $k = 2$ должна существовать одна эйлерова цепь, соединяющая вершины с нечетными степенями. Граф на рис. 1.3 относится к этому случаю.

Три программы нахождения радиуса графа и программа проверки наличия эйлеровой цепи в системе Maple приведены на с. 97–99.

1.2. Реберный граф

Для произвольного графа G реберный [1] граф $L(G)$ определяется следующими условиями:

1) множество вершин реберного графа $L(G)$ совпадает с множеством ребер графа G: $VL(G) = EG$;

2) вершины v_i и v_j смежны в $L(G)$ тогда и только тогда, когда ребра e_i и e_j смежны в G.

Английское название [2] реберного графа — *line graph*, отсюда и обозначение $L(G)$.

Теорема 6. *Если $d_1, d_2, ..., d_n$ — степенная последовательность (n, m) графа G, то $L(G)$ является (m, m_1)-графом, где*

$$m_1 = \frac{1}{2}\sum_{i=1}^{n} d_i^2 - m. \qquad (1.2)$$

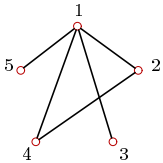
Задача. По заданному графу (рис. 1.6) построить его реберный граф.

---

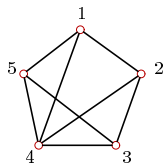
[1] Оре О. называет этот граф графом смежности ребер или *смежностным* графом [25].

[2] Многие английские термины теории графов, принятые в Maple, приведены в предметном указателе книги Н. Кристофидеса [17].

а



б



в

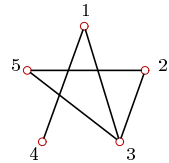
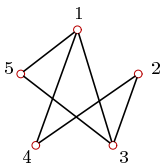
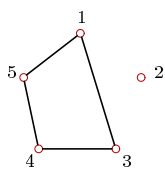


Рис. 1.6

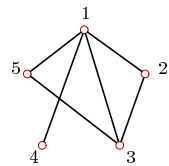
г



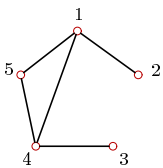
д



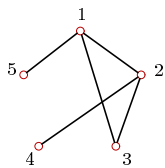
е



ж



з



и

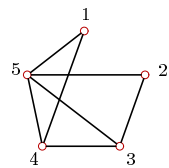


Рис. 1.6 (продолжение)

Ответы

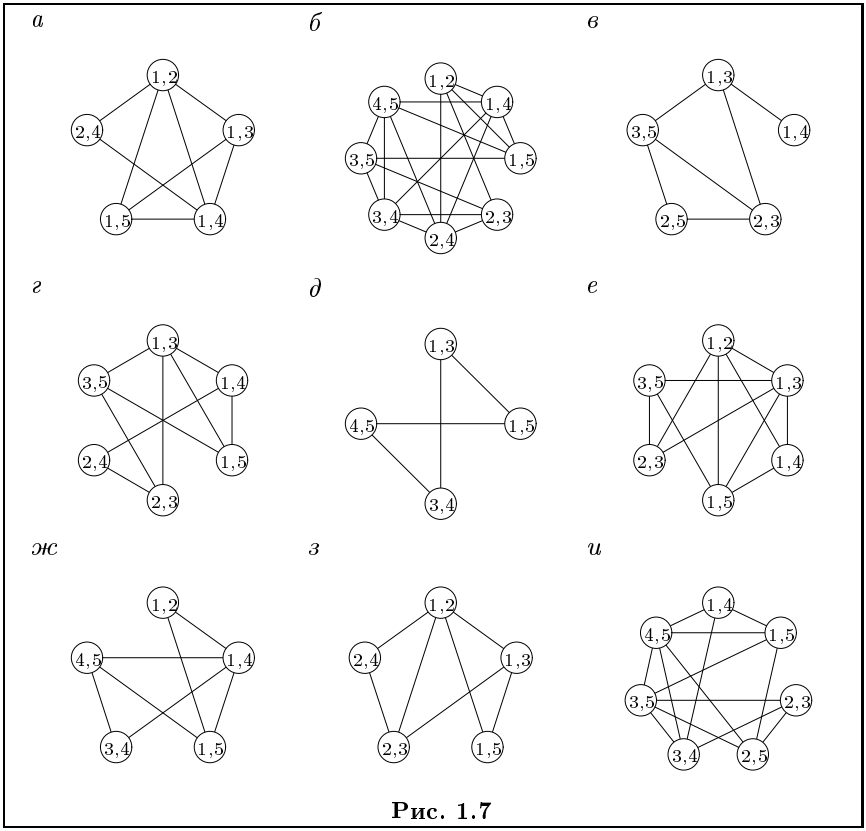
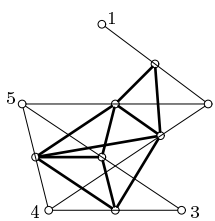


Рис. 1.7

Пример. Построить реберный граф для графа с рис. 1.2.

Решение. Решим задачу графически, с помощью прямого построения реберного графа по определению. В центре каждого ребра исходного графа G поместим вершину будущего реберного графа $L(G)$.



Будем соединять полученные вершины ребрами графа $L(G)$, если ребра графа G, на которых лежат эти вершины, смежны. На рис. 1.8 ребра графа $L(G)$ для наглядности выделены более толстыми линиями. В результате реберный граф получит $m_1 = 10$ ребер и 6 вершин (по числу m ребер графа G).

Рис. 1.8

Число m_1 соответствует значению, которое должно получиться по формуле (1.2). Степенная последовательность графа — 1–3–2–3–3. Находим

$$m_1 = \frac{1}{2}(1^2 + 3^2 + 2^2 + 3^2 + 3^2) - 6 = 16 - 6 = 10.$$

Таким образом, на рисунке совмещены исходный граф G (тонкие линии) и его реберный граф $L(G)$ (толстые линии).

Решение задачи о нахождении реберного графа в системе Maple приведено на с. 101.

1.3. Хроматический полином

Произвольная функция $f(v)$ на множестве вершин графа называется *раскраской* графа. Раскраска называется правильной, если $f(v_1) \neq f(v_2)$ для любых смежных вершин v_1 и v_2. Минимальное число k, при котором граф G является k-раскрашиваемым, называется *хроматическим числом* графа $\chi(G)$. Для хроматического числа имеются оценки.

Теорема 7 (Брукса [1]). *Для любого графа G, не являющегося полным, $\chi(G) \leqslant \Delta(G)$, если $\Delta(G) \geqslant 3$ — максимальная из степеней вершин графа.*

Для определения количества способов раскраски графа в x цветов можно составить *хроматический полином* $P(G, x)$. Значение полинома при некотором конкретном $x = x_0$ равно числу правильных раскрасок графа в x_0 цветов.

Существует лемма, утверждающая, что *хроматический полином графа имеет вид*

$$P(G, x) = P(G_1, x) + P(G_2, x), \tag{1.3}$$

где G_1 — граф, полученный из G добавлением нового ребра (u, v), а граф G_2 получается из G отождествлением вершин u и v.

Другой вариант леммы:

$$P(G, x) = P(G_1, x) - P(G_2, x), \tag{1.4}$$

где G_1 — граф, полученный из G удалением ребра (u, v), а граф G_2 получается из G отождествлением вершин u и v.

Операцию отождествления вершин u и v называют также *стягиванием* ребра (u, v). [2]

---

[1] Brooks R.L.

[2] Граф G называется *стягиваемым* к графу H, если H получается из G последовательным стягиванием его ребер. *Число Хадвигера* (Hadwiger H.) графа G — максимальный порядок полного графа, к которому стягивается граф G.

Оба варианта леммы составляют основу для хроматической редукции графа. Хроматическая редукция графа — представление графа в виде нескольких пустых или полных графов, сумма хроматических полиномов которых равна хроматическому полиному графа. Очевидно, что хроматический полином пустого графа O_n равен x^n (каждая вершина может быть раскрашена независимо от других), а для полного графа $P(K_n, x) = x(x-1)(x-2), ..., (x-n+1)$. Последнее выражение называют *факториальной степенью* [1] переменной x: $P(K_n, x) = x^{(n)}$.

Разложения по пустым и полным графам связаны. Факториальную степень можно представить в виде полинома:

$$x^{(n)} = \sum_{k=0}^{n} s_1(n,k)x^k, \tag{1.5}$$

где $s_1(n,k)$ — числа Стирлинга первого рода, и наоборот, степень x^n можно выразить через факториальные степени:

$$x^n = \sum_{k=0}^{n} s_2(n,k)x^{(k)}, \tag{1.6}$$

где $s_2(n,k)$ — числа Стирлинга [2] второго рода, обладающие следующими свойствами:

$$\begin{aligned} &s_2(n,k) = s_2(n-1,k-1) + ks_2(n-1,k) \text{ при } 0 < k < n, \\ &s_2(n,n) = 1 \text{ при } n \geqslant 0, \quad s_2(n,0) = 0 \text{ при } n > 0. \end{aligned} \tag{1.7}$$

При получении хроматического полинома могут быть полезны следующие теоремы [2].

Теорема 8. *Коэффициенты хроматического полинома составляют знакопеременную последовательность.*

Теорема 9. *Абсолютная величина второго коэффициента хроматического полинома равна числу ребер графа.*

Теорема 10. *Наименьшее число i, для которого отличен от нуля коэффициент при x^i в хроматическом полиноме графа G, равно числу компонент связности графа G.*

Кроме вершинной раскраски, существуют еще реберная раскраска и раскраска граней.

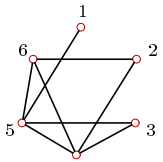
Задача. Найти хроматический полином графа (рис. 1.9) и вычислить количество способов раскраски графа в x_0 цветов.

---

[1] Факториальная степень связана с символом $(a)_k$ Похгаммера (Pochgammer L.), который определен как $(a)_k = a(a+1), ..., (a+k-1)$. Очевидно, $(a)_k = (a+k-1)^{(k)}$. Для символа Похгаммера имеется большое число формул (см., например, *Прудников А.П., Брычков Ю.А., Маричев О.И.* Интегралы и ряды. Дополнительные главы. — М.: Наука. 1986).

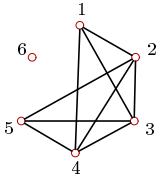
[2] Stirling T.

а



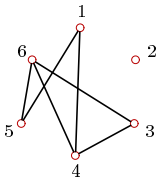
$x_0 = 4$

б



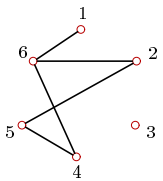
$x_0 = 3$

в



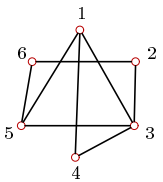
$x_0 = 6$

г



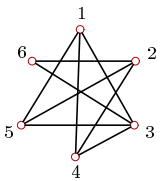
$x_0 = 5$

д



$x_0 = 4$

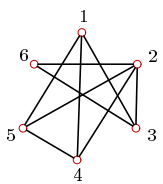
е



$x_0 = 5$

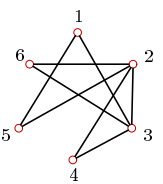
Рис. 1.9

ж



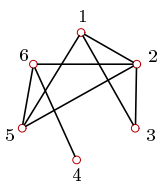
$x_0 = 6$

з



$x_0 = 6$

и



$x_0 = 5$

Рис. 1.9 (продолжение)

Ответы

	$C(x)$	$C(x_0)$
а	$x^6 - 8x^5 + 25x^4 - 38x^3 + 28x^2 - 8x$	288
б	$x^6 - 9x^5 + 29x^4 - 39x^3 + 18x^2$	0
в	$x^6 - 6x^5 + 14x^4 - 15x^3 + 6x^2$	15120
г	$x^6 - 5x^5 + 10x^4 - 9x^3 + 3x^2$	5200
д	$x^6 - 8x^5 + 26x^4 - 43x^3 + 36x^2 - 12x$	336
е	$x^6 - 9x^5 + 34x^4 - 66x^3 + 64x^2 - 24x$	1980
ж	$x^6 - 9x^5 + 33x^4 - 61x^3 + 56x^2 - 20x$	8160
з	$x^6 - 8x^5 + 26x^4 - 43x^3 + 36x^2 - 12x$	10080
и	$x^6 - 8x^5 + 25x^4 - 38x^3 + 28x^2 - 8x$	2160

Пример. Найти хроматический полином графа (рис. 1.10).

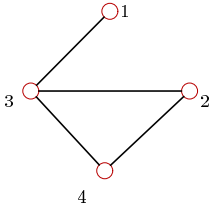
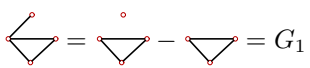


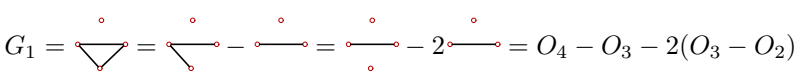
Рис. 1.10

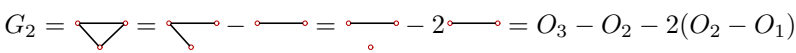
Решение. В зависимости от числа ребер графа можно использовать разложение (1.3) или (1.4). Если граф почти полный, то, добавив несколько ребер по разложению (1.3), получим хроматический полином в виде суммы факториальных степеней. Если же ребер мало и для получения пустого графа требуется удалить только несколько ребер, то следует использовать разложение (1.4) с удалением ребер. Такие действия называются хроматической редукцией.

1. *Хроматическая редукция по пустым графам.* Воспользуемся леммой (1.4). Удаляя ребра и отождествляя соответствующие вершины (стягивая ребра), сведем исходный граф к пустым графам. Сначала разложим граф на два, убрав, а затем стянув ребро 1–3. Выполненное действие запишем в виде условного равенства:



Здесь операция сложения (или вычитания) относится не к самому графу, а к его хроматическому полиному. Таким образом, последнее равенство означает, что $P(G) = P(G_1) - P(G_2)$. Для сокращения записи обозначение $P(...)$ будем опускать. Далее разложим каждый из графов, G_1 и G_2, пользуясь той же леммой:





Приведем подобные члены:

$$\begin{aligned} G = G_1 - G_2 &= O_4 - O_3 - 2(O_3 - O_2) - \\ &- (O_3 - O_2 - 2(O_2 - O_1)) = O_4 - 4O_3 + 5O_2 - 2O_1. \end{aligned} \tag{1.8}$$

В итоге получим искомый хроматический полином:

$$P(G, x) = x^4 - 4x^3 + 5x^2 - 2x. \tag{1.9}$$

Разложение (1.8) называется хроматической редукцией графа по пустым графам.

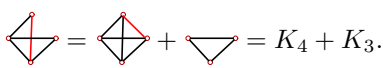
Очевидно, результат соответствует утверждениям теорем 8–10. Коэффициенты в (1.9) образуют знакопеременную последовательность,

а коэффициент при x^3 равен четырем — числу ребер. Наименьшая степень x в полиноме равна 1, т.е. числу компонент связности графа.

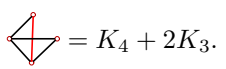
2. *Хроматическая редукция по полным графам.* Добавив к графу (см. рис. 1.10) ребро 1–4, получим граф с большим числом ребер. Затем в этом же (исходном) графе отождествим вершины 1 и 4. В результате получим два графа:

$G =$  (1.10)

Отождествление вершин приводит к уменьшению порядка и иногда размера графа. Второй граф — это полный граф K_3, его преобразовывать больше не требуется. К первому графу добавим ребро 1–2 и отождествим вершины 1 и 2:

 (1.11)

В итоге

 (1.12)

Хроматический полином примет вид

$$P(G,x) = x^{(4)} + 2x^{(3)} = x(x-1)(x-2)(x-3) + \\ + 2x(x-1)(x-2) = x^4 - 4x^3 + 5x^2 - 2x. \tag{1.13}$$

Разложение (1.12) называется хроматической редукцией графа по полным графам.

Оба способа дали один результат, и из редукции по полным графам легко получить редукцию по пустым. Для этого достаточно раскрыть скобки и привести подобные члены, как в (1.13). Однако обратное действие не очевидно. Для того чтобы полином $x^4 - 4x^3 + 5x^2 - 2x$, полученный из пустых графов, выразить в виде суммы факториальных степеней, необходимы числа Стирлинга 2-го рода. Согласно рекуррентным формулам (1.7) имеем следующие числа:

$$s_2(k,1) = 1, \quad k = 1,2,...,$$
$$s_2(3,2) = s_2(2,1) + 2s_2(2,2) = 3,$$
$$s_2(4,2) = s_2(3,1) + 2s_2(3,2) = 1 + 2 \cdot 3 = 7,$$
$$s_2(4,3) = s_2(3,2) + 3s_2(3,3) = 3 + 3 = 6.$$

Пользуясь (1.6) и найденными числами Стирлинга 2-го рода, получим

$$x^2 = x^{(1)} + x^{(2)},$$
$$x^3 = x^{(1)} + 3x^{(2)} + x^{(3)},$$
$$x^4 = x^{(1)} + 7x^{(2)} + 6x^{(3)} + x^{(4)}.$$

Произведем преобразование хроматического полинома:

$$x^4 - 4x^3 + 5x^2 - 2x = x^{(1)} + 7x^{(2)} + 6x^{(3)} + x^{(4)} -$$
$$-4(x^{(1)} + 3x^{(2)} + x^{(3)}) + 5(x^{(1)} + x^{(2)}) + 2x^{(1)} = x^{(4)} + 2x^{(3)}.$$

Хроматическое число $\chi(G)$ графа лучше всего получить, разложив хроматический полином на сомножители:

$$P(G, x) = x(x-1)^2(x-2).$$

Минимальное натуральное число x, при котором $P(G, x)$ не обращается в нуль, равно 3. Отсюда получаем $\chi(G) = 3$. Так как максимальная степень вершин графа $\Delta(G) = 3$, выполняется оценка $\chi(G) \leqslant \Delta(G)$ (см. с. 16).

Maple-программа для нахождения хроматического полинома графа с использованием оператора `chrompoly` приведена на с. 102.

1.4. Ранг-полином графа

Ранг графа определяется как $\nu^* = n - k$, где n — число вершин, k — число компонент связности графа. Коранг графа, или цикломатический ранг, есть $\nu = m - \nu^* = m - n + k$, где m — число ребер.

Ранг-полином [1] графа G имеет вид

$$P_\nu(x, y) = \sum x^{\nu^*_G - \nu^*_H} y^{\nu_H},$$

где $\nu^*_G = n - k$ — ранг графа G, ν_H — коранг остовного (т.е. включающего в себя все вершины графа) подграфа H, а ν^*_H — его ранг. Суммирование ведется по всем остовным подграфам графа G.

Ранг-полином служит для анализа множества остовных подграфов. Так, например, коэффициент при x^{-k} в $P_\nu(x, 1/x)$ есть число подграфов размера k, а значение $P_\nu(0, 1)$ равно числу подграфов (включая несобственный подграф), ранг которых равен рангу самого графа. Другие свойства ранга-полинома приведены в программе 6 на с. 103.

Задача. Найти ранг-полином графа (рис. 1.11).

[1] Название этого полинома (rankpoly) заимствовано из Maple.

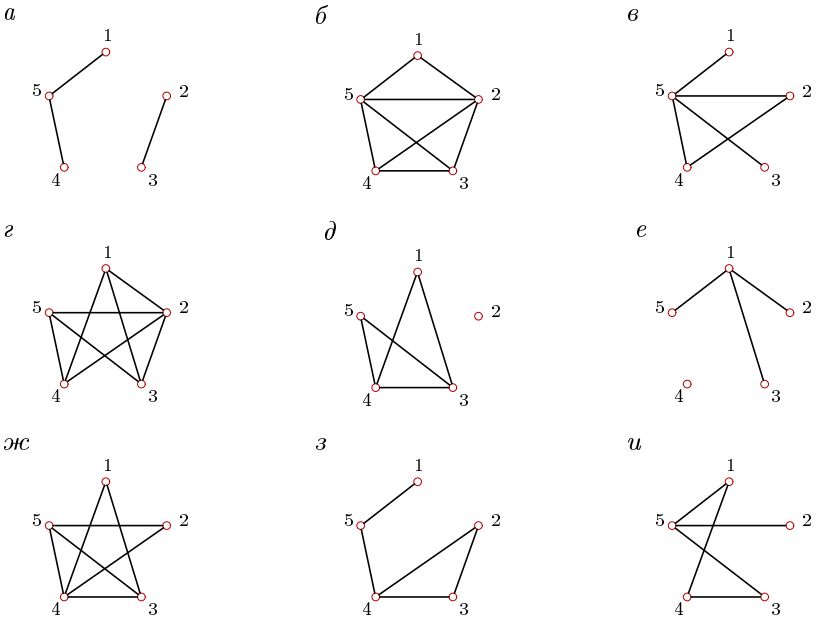


Рис. 1.11

Ответы

	$P_\nu(x,y)$
а	$x^3+3\,x^2+3\,x+1$
б	$y^4+(x+8)\,y^3+(8\,x+27)\,y^2+(5\,x^2+30\,x+48)\,y+x^4+8\,x^3+$ $+\,28\,x^2+51\,x+40$
в	$(x^2+2\,x+1)\,y+x^4+5\,x^3+10\,x^2+9\,x+3$
г	$y^5+9\,y^4+(3\,x+36)\,y^3+(2\,x^2+21\,x+81)\,y^2+(13\,x^2+58\,x+$ $+\,x^3+105)\,y+x^4+9\,x^3+71\,x+35\,x^2+66$
д	$y^2+(2\,x+5)\,y+x^3+5\,x^2+10\,x+8$
е	$x^3+3\,x^2+3\,x+1$
ж	$y^3+(2\,x+7)\,y^2+(14\,x+3\,x^2+19)\,y+x^4+7\,x^3+21\,x^2+21+32\,x$
з	$(x^2+2\,x+1)\,y+x^4+5\,x^3+10\,x^2+9\,x+3$
и	$(x+1)\,y+x^4+5\,x^3+10\,x^2+10\,x+4$

Пример. Найти ранг-полином графа (рис. 1.12).

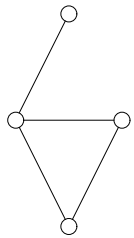


Рис. 1.12

Решение. Найдем все 16 остовных подграфов графа G (рис. 1.13–1.15). Множество представим в виде четырех графов размера 1 (т.е. с одним ребром), шести графов размера 2, четырех графов размера 3 и двух несобственных графов (пустой граф и граф G). Найдем для каждого подграфа ранг (по формуле $\nu^* = 4 - k$, где k — число компонент подграфа, включая изолированные вершины) и коранг ($\nu = m - \nu^*$, где m — число ребер подграфа).

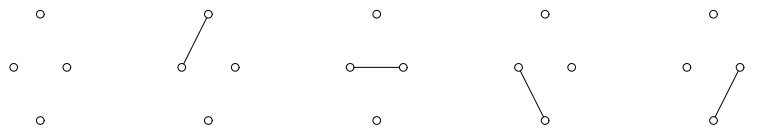


Рис. 1.13

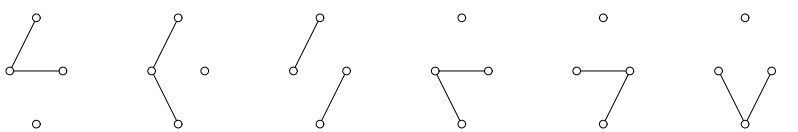


Рис. 1.14

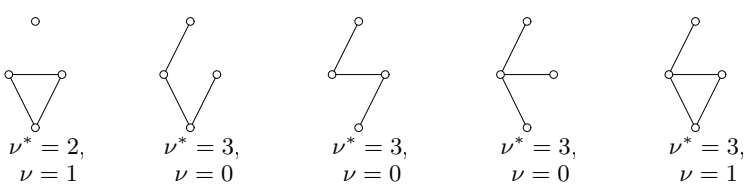


Рис. 1.15

Учитывая, что ранг ν^*_G графа равен 3, получаем сумму

$$\sum x^{\nu^*_G - \nu^*_H} y^{\nu_H} =$$
$$= x^{3-0}y^0 + 4x^{3-1}y^0 + 6x^{3-2}y^0 + 3x^{3-3}y^0 + x^{3-2}y^1 + x^{3-3}y^1 =$$
$$= x^3 + 4\,x^2 + 6\,x + 3 + x\,y + y.$$

Программа нахождения ранга-полинома графа в системе Maple приведена на с. 103.

1.5. Циклы

Маршрут, в котором начало и конец совпадают, — *циклический*. Циклический маршрут называется *циклом*, если он — цепь.

Остовом графа G называют граф, не содержащий циклов и состоящий из ребер графа G и всех его вершин. Остов графа определяется неоднозначно.

Ребра графа, не входящие в остов, называются *хордами*. Цикл, получающийся при добавлении к остову графа его хорды, называется *фундаментальным* относительно этой хорды.

Теорема 11. *Число ребер неографа, которые необходимо удалить для получения остова, не зависит от последовательности их удаления и равно цикломатическому рангу графа.*

Задача. По заданной матрице смежности (рис. 1.16) определить число циклов длины 3 (c_3) и длины 4 (c_4). Записать матрицу инцидентности и матрицу фундаментальных циклов.

а

$$\begin{vmatrix}0&0&0&1&0&0&0&0&0&0\\0&0&1&1&0&1&0&0&0\\0&1&0&0&1&1&0&0&0\\1&1&0&0&1&0&0&1&0\\0&0&1&1&0&1&1&1&1\\0&1&1&0&1&0&0&1&0\\0&0&0&0&1&0&0&1&0\\0&0&0&1&1&1&1&0&1\\0&0&0&0&1&0&0&1&0\end{vmatrix}$$

б

$$\begin{vmatrix}0&1&0&0&0&0&0&0&0&0\\1&0&1&1&1&1&0&0&0\\0&1&0&0&1&1&0&0&0\\0&1&0&0&1&0&0&1&0\\0&1&1&1&0&0&1&1&1\\0&1&1&0&0&0&0&1&1\\0&0&0&0&1&0&0&1&0\\0&0&0&1&1&1&1&0&0\\0&0&0&0&1&1&0&0&0\end{vmatrix}$$

в

$$\begin{vmatrix}0&1&0&1&0&0&0&0&0\\1&0&0&1&0&1&0&0&0\\0&0&0&0&1&1&0&0&0\\1&1&0&0&1&0&1&1&0\\0&0&1&1&0&1&1&0&1\\0&1&1&0&1&0&0&1&1\\0&0&0&1&1&0&0&0&0\\0&0&0&1&0&1&0&0&1\\0&0&0&0&1&1&0&1&0\end{vmatrix}$$

г

$$\begin{vmatrix}0&1&0&0&0&0&0&0&0&0\\1&0&1&1&0&1&0&0&0\\0&1&0&0&1&1&0&0&0\\0&1&0&0&1&0&1&1&0\\0&0&1&1&0&1&1&0&1\\0&1&1&0&1&0&0&1&1\\0&0&0&1&1&0&0&0&0\\0&0&0&1&0&1&0&0&1\\0&0&0&0&1&1&0&1&0\end{vmatrix}$$

д

$$\begin{vmatrix}0&1&0&1&0&0&0&0&0\\1&0&1&1&0&1&0&0&0\\0&1&0&0&1&1&0&0&0\\1&1&0&0&1&0&1&1&0\\0&0&1&1&0&0&1&1&1\\0&1&1&0&0&0&0&1&0\\0&0&0&1&1&0&0&0&0\\0&0&0&1&1&1&0&0&1\\0&0&0&0&1&0&0&1&0\end{vmatrix}$$

е

$$\begin{vmatrix}0&1&0&1&0&0&0&0&0\\1&0&0&1&0&1&0&0&0\\0&0&0&0&1&1&0&0&0\\1&1&0&0&1&0&1&1&0\\0&0&1&1&0&1&1&0&1\\0&1&1&0&1&0&0&1&1\\0&0&0&1&1&0&0&0&0\\0&0&0&1&0&1&0&0&1\\0&0&0&0&1&1&0&1&0\end{vmatrix}$$

ж

$$\begin{vmatrix}0&1&0&0&1&0&0&0&0\\1&0&1&1&1&1&0&0&0\\0&1&0&0&1&1&0&0&0\\0&1&0&0&1&0&1&1&0\\1&1&1&1&0&0&1&1&1\\0&1&1&0&0&0&0&1&0\\0&0&0&1&1&0&0&0&0\\0&0&0&1&1&1&0&0&1\\0&0&0&0&1&0&0&1&0\end{vmatrix}$$

з

$$\begin{vmatrix}0&1&0&0&1&0&0&0&0\\1&0&1&1&1&1&0&0&0\\0&1&0&0&1&0&0&0&0\\0&1&0&0&1&0&1&1&0\\1&1&1&1&0&1&1&0&1\\0&1&0&0&1&0&0&1&1\\0&0&0&1&1&0&0&0&0\\0&0&0&1&0&1&0&0&1\\0&0&0&0&1&1&0&1&0\end{vmatrix}$$

и

$$\begin{vmatrix}0&0&0&1&1&0&0&0&0\\0&0&1&1&1&1&0&0&0\\0&1&0&0&1&0&0&0&0\\1&1&0&0&1&0&1&1&0\\1&1&1&1&0&1&1&1&1\\0&1&0&0&1&0&0&1&1\\0&0&0&1&1&0&0&0&0\\0&0&0&1&1&1&0&0&0\\0&0&0&0&1&1&0&0&0\end{vmatrix}$$

Рис. 1.16

Ответы

№	c_3	c_4	№	c_3	c_4	№	c_3	c_4
а	1	6	г	5	30	ж	6	36
б	4	24	д	4	24	з	6	36
в	5	30	е	5	30	и	6	36

Пример. По заданной матрице смежности:

$$\begin{vmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{vmatrix},$$

определить число циклов длины 3 (c_3) и длины 4 (c_4). Записать матрицу инцидентности и матрицу фундаментальных циклов.

Решение. Матрица смежности данного графа симметричная, поэтому ей соответствует неориентированный граф. Сумма элементов матрицы равна 12, следовательно, по лемме о рукопожатиях (см. с. 6) в графе 6 ребер. Построим этот граф (рис. 1.17). Очевидно, в нем два цикла (3–4–5 и 1–3–5) длиной 3 и один цикл (1–3–4–5) длиной 4. В данной задаче решение получено прямым подсчетом по изображению графа. Для более сложных случаев существует алгоритм решения задачи по матрице смежности.

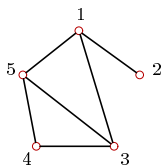


Рис. 1.17

Известно, что след (trace) матрицы смежности, возведенной в k-ю степень, равен числу циклических маршрутов длины k [1]. Это число включает в себя и искомое число циклов. Цикл отличается от циклического маршрута тем, что в нем не повторяются ребра. Кроме того, предполагается, что искомые циклы не помечены, а в след матрицы входят именно помеченные маршруты. Непомеченных циклов длиной 3 в 6 раз меньше, чем помеченных, так как каждый помеченный цикл может отличаться началом, а их в данном случае три, и двумя направлениями обхода (по и против часовой стрелки). Возведем заданную матрицу смежности в третью степень:

$$A^3 = \begin{bmatrix} 2 & 3 & 6 & 2 & 6 \\ 3 & 0 & 1 & 2 & 1 \\ 6 & 1 & 4 & 5 & 5 \\ 2 & 2 & 5 & 2 & 5 \\ 6 & 1 & 5 & 5 & 4 \end{bmatrix}, \tag{1.14}$$

и получим

$$c_3 = \frac{1}{6}\mathrm{trace}A^3 = 2. \tag{1.15}$$

[1] См. теорему 5 на с. 11.

Возведение матрицы в степень лучше выполнить по программе системы Maple: `c3:=A^3`; при этом подключения пакета линейной алгебры `LinearAlgebra` не требуется. Поскольку циклических маршрутов длиной 3, отличных от циклов длиной 3, не существует, найденное число и есть ответ в поставленной задаче. След матрицы в пакете `LinearAlgebra` системы Maple вычисляется оператором `Trace(c3)`.

С циклами длиной 4 немного сложнее. В след четвертой степени матрицы смежности графа:

$$A^4 = \begin{bmatrix} 15 & 2 & 10 & 12 & 10 \\ 2 & 3 & 6 & 2 & 6 \\ 10 & 6 & 16 & 9 & 15 \\ 12 & 2 & 9 & 10 & 9 \\ 10 & 6 & 15 & 9 & 16 \end{bmatrix}, \tag{1.16}$$

входят не только циклы, но и циклические маршруты с двойным и четырехкратным прохождением ребер. Обозначим количества таких маршрутов через x_2 и x_4 соответственно. Очевидно, число маршрутов с четырехкратным прохождением одного ребра для вершины v_i равно степени этой вершины: $x_4 = \sum\limits_{i=1}^{n} \deg(v_i)$. Число маршрутов с двукратным прохождением ребра складывается из числа x_{2v} маршрутов с висячей вершиной v_i и числа x_{2c} маршрутов с вершиной v_i в центре. Легко заметить, что $x_{2c} = \sum\limits_{i=1}^{n} \deg(v_i)(\deg(v_i) - 1)$. Число x_{2v} зависит от степеней вершин, соседних с v_i:

$$x_{2v} = \sum_{i=1}^{n} \sum_{\{k,i\} \subseteq G} (\deg(v_k) - 1),$$

где $\{k, i\}$ — ребро, инцидентное вершинам i и k.

Для графа на рис. 1.17 получим

$$\begin{aligned} &\operatorname{trace} A^4 = 60, \\ &x_{2v} = 4 + 2 + 5 + 4 + 5 = 20, \\ &x_{2c} = 6 + 0 + 6 + 2 + 6 = 20, \\ &x_4 = 3 + 1 + 3 + 2 + 3 = 12. \end{aligned}$$

С учетом того, что непомеченных циклов длиной 4 в 8 раз меньше, получим

$$c_4 = (\operatorname{trace} A^4 - x_{2c} - x_{2v} - x_4)/8 = (60 - 20 - 20 - 12)/8 = 1.$$

После преобразований формула примет вид

$$c_4 = \frac{1}{8}\left(\operatorname{trace} A^4 - \sum_{i=1}^{n}\sum_{\{k,i\}\subseteq G}(\deg v_k - 1) - \sum_{i=1}^{n}\deg^2 v_i\right) = 1. \quad (1.17)$$

Матрица инцидентности. Число строк матрицы инцидентности I равно числу вершин, число столбцов — числу ребер; $i_{ve} = 1$, если вершина v инцидентна ребру e, в противном случае $i_{ve} = 0$. Матрицу строим по рис. 1.17, ребра нумеруем по рис. 1.18:

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}.$$

Два способа построения матрицы инцидентности в системе Maple рассмотрены в программе 8 на с. 105.

Матрица фундаментальных циклов. Пронумеруем ребра графа, начиная нумерацию с хорд (рис. 1.18).

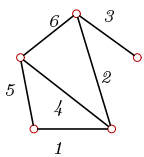


Рис. 1.18

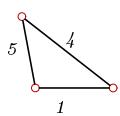


Рис. 1.19

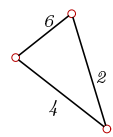


Рис. 1.20

Двум хордам, 1 и 2, соответствуют два фундаментальных цикла: *1–4–5* и *2–4–6* (рисунки 1.19, 1.20). Матрица фундаментальных циклов имеет две строки (число циклов) и шесть столбцов (число ребер):

	1	*2*	*3*	*4*	*5*	*6*
1	1	0	0	1	1	0
2	0	1	0	1	0	1

В первой строке матрицы единицами отмечены столбцы с номерами ребер, входящих в первый цикл, а во второй строке — номера ребер из второго цикла.

Глава 2

ОРИЕНТИРОВАННЫЕ ГРАФЫ

Ребро (v_1, v_2) в графе G может быть ориентированным и иметь начало и конец. Такое ребро называется дугой, а граф, содержащий дуги, называется ориентированным, или орграфом. На рисунке дуга изображается стрелкой, а в Maple дуга вводится как упорядоченная пара вершин: `[v1,v2]`. Многие понятия, введенные для неографов, для орграфов приобретают другое определение. Матрица расстояний для орграфа несимметрична, и эксцентриситет вершины зависит от того, как выбирается максимум. Если максимум ищется в строке, то эксцентриситет вершины называется числом внешнего разделения [17], а если в столбце — числом внутреннего разделения. Соответственно определяются внешний и внутренний центры.

Основание орграфа — неограф с теми же вершинами, но ребрами вместо соответствующих дуг.

Маршрут в орграфе — последовательность вершин, соединенных дугами, направленными в одну сторону.

Маршрут, в котором все дуги разные, есть *путь*.

Путь, в котором начало и конец совпадают, есть *контур*. Длина пути измеряется числом входящих в него дуг, а для взвешенного орграфа это сумма весов дуг.

В орграфе две локальные степени вершины v: $\deg^{\text{in}}(v)$ — число дуг, входящих в v, и $\deg^{\text{out}}(v)$ — число дуг, выходящих из v [1]. Лемма о рукопожатиях (см. с. 6) для орграфа имеет вид

$$\sum \deg^{\text{in}}(v) = \sum \deg^{\text{out}}(v) = m,$$

где суммирование производится по всем вершинам графа.

Множество вершин v, образующих дугу $[v, u]$ с вершиной u, называется множеством $\Gamma^{\text{in}}(u)$ предшественников вершины u, а множество вершин u, образующих дугу $[v, u]$ с вершиной v, называется множеством $\Gamma^{\text{out}}(v)$ преемников вершины v. Мощности этих множеств равны, соответственно, полустепеням вершин: $\Gamma^{\text{in}}(u) = \deg^{\text{in}}(u)$, $\Gamma^{\text{out}}(v) = \deg^{\text{out}}(v)$.

В дальнейшем под графом будем понимать как неограф, так и орграф.

[1] Иногда используются следующие обозначения: d^+ — полустепень исхода и d^- — полустепень захода.

2.1. Маршруты в орграфе

Задачи, связанные с маршрутами в орграфе, имеют большое практическое значение, что и дает стимул к развитию и совершенствованию методов их решения. Наиболее часто встает вопрос о минимальных и максимальных расстояниях, о числе маршрутов определенной длины.

Особенность таких задач для орграфов состоит в том, что несмотря на небольшой порядок графа (в приведенном ниже задании предлагается порядок 5) простое решение неочевидно. В следующей задаче для решения предлагается легко программируемый (особенно в системах компьютерной математики) алгебраический метод. Длина маршрута здесь измеряется в числе дуг, входящих в него. Допускаются замкнутые маршруты.

Задача. Дан орграф порядка 5 (рис. 2.1). Сколько в нем маршрутов длины 3?

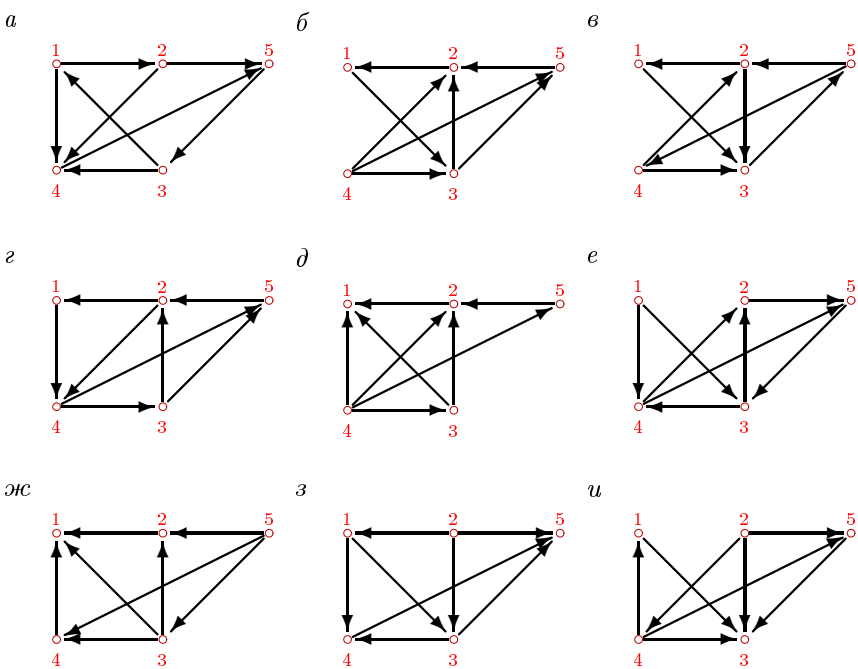


Рис. 2.1

Ответы

	а	б	в	г	д	е	ж	з	и
n_3	15	11	18	21	2	16	2	5	2

Пример. Дан орграф (рис. 2.2). Сколько в нем маршрутов длиной 3?

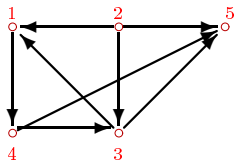


Рис. 2.2

Решение. Используем алгебраический метод решения задачи на основании теоремы 5 (см. с. 11). Запишем матрицу смежности. Матрица смежности орграфа — несимметричная:

$$A := \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Возведем эту матрицу в степень 3:

$$A^3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Суммируя все элементы полученной матрицы, находим, что число маршрутов длиной 3 равно восьми. Три единицы, стоящие по диагонали, показывают, что сюда входит 3 помеченных контура. Очевидно, это контуры 1–4–3, 4–3–1 и 3–1–4.

Проверить наличие контура в орграфе можно также методом топологической сортировки (см. с. 69). Если граф не может быть отсортирован топологически, то в нем есть контуры.

2.2. Транзитивное замыкание

Основные определения. *Прямым* (декартовым) произведением множеств A и B называется множество $A \times B = \{(x, y) |\ x \in A,\ y \in B\}$. *Бинарное отношение* на X — любое подмножество прямого произведения: $\rho \subset X \times X$.

Отношение ρ на X *рефлексивно*, если для любого $x \in X$ пара $(x, x) \in \rho$.

Отношение ρ на X *антирефлексивно*, если для любого $x \in X$ пара $(x, x) \notin \rho$.

Отношение ρ на X *симметрично*, если для любой пары (x, y) из условия $(x, y) \in \rho$ следует $(y, x) \in \rho$.

Отношение ρ на X *антисимметрично*, если из условий $(x, y) \in \rho$ и $(y, x) \in \rho$ следует $x = y$.

Отношение ρ на X *асимметрично*, если для любой пары (x, y) из условия $(x, y) \in \rho$ следует $(y, x) \notin \rho$.

Отношение ρ на X *транзитивно*, если для любых двух пар (x, y) и (y, z) из условий $(x, y) \in \rho$ и $(y, z) \in \rho$ следует $(x, z) \in \rho$.

Отношение $\tilde{\rho}$ называют *замыканием* отношения ρ на свойство A, если $\tilde{\rho}$ обладает свойством A, $\rho \subset \tilde{\rho}$ и для любого отношения σ со свойством A справедливо вложение $\tilde{\rho} \subset \sigma$.

Композицией бинарных отношений $\rho \subset X \times X$ и $\sigma \subset X \times X$ называют отношение $\tau = \sigma \circ \rho$, состоящее из пар (x, z), таких, что $(x, y) \subset \rho$, $(y, z) \subset \sigma$.

Транзитивное замыкание отношения ρ имеет вид $\rho^+ = \bigcup\limits_{k=1}^{\infty} \rho^k$, где $\rho^2 = \rho \circ \rho$, $\rho^k = \rho \circ \rho^{k-1}$.

Теорема 12. *Отношение ρ транзитивно тогда и только тогда, когда $\rho \circ \rho \subset \rho$.*

Граф есть отношение на множестве вершин. Элементами этого отношения являются дуги (или ребра, если отношение симметрично).

Орграф называется *транзитивным*, если для любых его дуг $[v_i, v_k]$, $[v_k, v_j]$ существует замыкающая дуга $[v_i, v_j]$.

Задача. Исследовать отношение, заданное матрицей (рис. 2.3), на симметрию, антисимметрию, асимметрию, рефлексивность, антирефлексивность. Найти транзитивное замыкание отношения. Построить граф отношения ρ и его транзитивного замыкания.

а

$$\begin{vmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

б

$$\begin{vmatrix} 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

в

$$\begin{vmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

г

$$\begin{vmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

д

$$\begin{vmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{vmatrix}$$

е

$$\begin{vmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

ж

$$\begin{vmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

з

$$\begin{vmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

и

$$\begin{vmatrix} 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

Рис. 2.3

Ответы

а
$$\begin{vmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

б
$$\begin{vmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

в
$$\begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

г
$$\begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

д
$$\begin{vmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{vmatrix}$$

е
$$\begin{vmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

ж
$$\begin{vmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

з
$$\begin{vmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

и
$$\begin{vmatrix} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

Рис. 2.4

Пример. Дано отношение, заданное матрицей

$$A = \begin{vmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{vmatrix}.$$

Исследовать отношение на симметрию, антисимметрию, асимметрию, рефлексивность, антирефлексивность. Найти транзитивное замыкание отношения. Построить граф отношения ρ и его транзитивного замыкания.

Решение. Исследуем свойства данного отношения.

1. Данное отношение не является симметричным, так как матрица несимметрична. Например, пара (2, 1) принадлежит ρ, а пара (1, 2) ему не принадлежит.

2. Отношение антисимметрично, так как нет ни одной пары $a_{ij} = a_{ji} = 1$, $i \neq j$.

3. Отношение антисимметрично, но не асимметрично, так как на диагонали матрицы имеются элементы, равные 1.

4. Все диагональные элементы матрицы рефлексивного отношения равны 1. Данное отношение не является рефлексивным.

5. Отношение не обладает свойством антирефлексивности, так как диагональ матрицы ненулевая.

6. Данное отношение не является транзитивным, так как, например, пары (1, 4) и (4, 3) принадлежат ρ, а пара (1, 3) ему не принадлежит.

Найдем транзитивное замыкание графа, заданного отношением ρ. Процедура транзитивного замыкания сводится к добавлению в матрицу смежности графа минимального числа единиц, так, чтобы соответствующее отношение обладало свойством транзитивности.

Способ 1.

1. Вычисляем матрицу композиции $\rho^2 = \rho \circ \rho$. Для этого умножаем [1] матрицу саму на себя: $A_1 = AA$.

Для $i, j = 1, ..., 4$ вычисляем

$$a_{1_{ij}} = (a_{i1} \wedge a_{1j}) \vee (a_{i2} \wedge a_{2j}) \vee (a_{i3} \wedge a_{3j}) \vee (a_{i4} \wedge a_{4j}).$$

Получаем

$$A_1 = \begin{Vmatrix} 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{Vmatrix} \begin{Vmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{Vmatrix} = \begin{Vmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{Vmatrix}.$$

2. Находим логическую сумму (дизъюнкцию) матриц. Поэлементная дизъюнкция матриц дает

$$A_2 = A \vee A_1 = \begin{Vmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{Vmatrix}.$$

3. Сравним A_2 и матрицу A. Если $A_2 = A$, то A_2 — искомая матрица. Если $A_2 \neq A$, то полагаем $A = A_2$, возвращаемся к п. 1 и повторяем всю процедуру для новой матрицы. В данном случае $A_2 \neq \neq A$. Принимаем

$$A = \begin{Vmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{Vmatrix}.$$

[1] Не путать произведение булевых матриц $A = BC$ с поэлементным логическим умножением $B \wedge C$. Очевидно, $a_{ij} = 1$, если хотя бы в одном случае k-й элемент i-й строки первого сомножителя и k-й элемент j-го столбца второго сомножителя одновременно равны 1. В противном случае $a_{ij} = 0$.

1′. Умножаем матрицу саму на себя:

$$A_1 = AA = \left\|\begin{matrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{matrix}\right\| \left\|\begin{matrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{matrix}\right\| = \left\|\begin{matrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{matrix}\right\|.$$

2′. Находим логическую сумму (дизъюнкцию) матриц:

$$A_2 = A \vee A_1 = \left\|\begin{matrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{matrix}\right\|.$$

3′. Сравниваем: $A_2 \neq A$. Полагаем $A = A_2$ и повторяем процедуру еще раз.

1″. Умножаем

$$A_1 = AA = \left\|\begin{matrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{matrix}\right\| \left\|\begin{matrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{matrix}\right\| = \left\|\begin{matrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{matrix}\right\|.$$

2″. Находим сумму:

$$A_2 = A \vee A_1 = \left\|\begin{matrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{matrix}\right\|.$$

3″. Сравниваем: $A_2 = A$. Следовательно, A_2 — матрица транзитивного замыкания заданного отношения.

Способ 2. Алгоритм Уоршелла [1] [27].

Рассматриваем все внедиагональные элементы матрицы. Если $a_{ij} \neq 0$, то i-ю строку заменяем дизъюнкцией i-й и j-й строк.

1. Элемент $a_{14} = 1$. Первую строку заменяем поэлементной дизъюнкцией первой и четвертой строки:

$$A_1 = \left\|\begin{matrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{matrix}\right\|.$$

[1] Warshall S.

2. Элемент $a_{21} = 1$. Вторую строку заменяем поэлементной дизъюнкцией второй и первой строки:

$$A_2 = \begin{vmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{vmatrix}.$$

3. Элемент $a_{43} = 1$. Дизъюнкция четвертой и третьей строки не меняет вид матрицы. Таким образом, полученная матрица является матрицей транзитивного замыкания отношения ρ.

Оба способа дают один и тот же результат.

На рисунках 2.5 и 2.6 представлены графы отношения ρ и его транзитивного замыкания. Диагональные элементы матрицы соответствуют петлям на графе. Матрица несимметрична, поэтому граф отношения ориентированный.

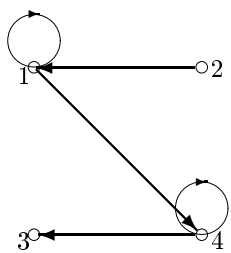


Рис. 2.5

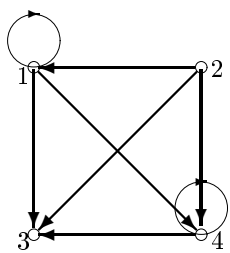


Рис. 2.6

Maple-программа для определения некоторых свойств отношения по его матрице приведена на с. 106. В программе дан также алгоритм транзитивного замыкания.

2.3. Компоненты сильной связности графа

Понятие сильной связности относится только к орграфам.

Основание орграфа — неограф с теми же вершинами, но ребрами вместо соответствующих дуг [1].

Орграф называется *связным*, если связно его основание.

[1] Наоборот, каждому неориентированному графу *канонически* соответствует орграф с теми же вершинами, в котором каждое ребро заменено дугами, инцидентными тем же вершинам и имеющими противоположные направления. Кроме того, при вычислении матрицы Кирхгофа удобно вводить *ориентацию* (обычно произвольную) неографа — замену ребер дугами (см. с. 82).

Вершина u *достижима* из вершины v, если существует маршрут из v в u.

Орграф называется *сильно связным* (или орсвязным), если любая его вершина достижима из любой вершины.

Граф называется *ориентируемым*, если он является основанием сильно связного графа.

Задача. Найти компоненты сильной связности орграфа (рис. 2.7).

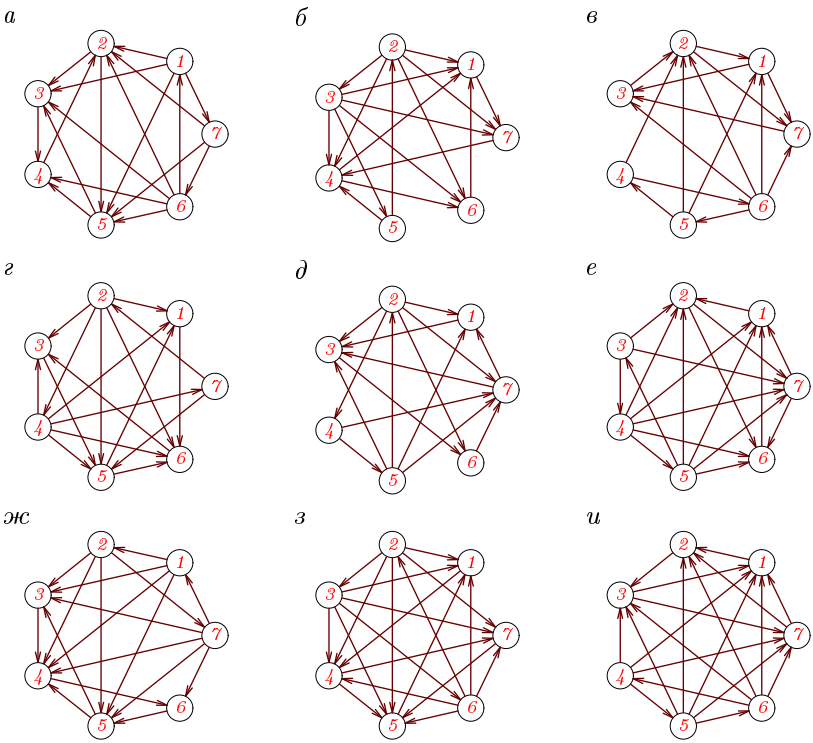


Рис. 2.7

Ответы

	1	2		1	2		1	2
а	2, 5, 3, 4	7, 6, 1	г	5, 1, 6, 3	4, 7, 2	ж	5, 3, 4, 6	7, 1, 2
б	4, 6, 1, 7	2, 3, 5	д	7, 1, 3, 6	2, 4, 5	з	4, 7, 5, 1	6, 2, 3
в	2, 1, 7, 3	5, 4, 6	е	2, 7, 6, 1	3, 4, 5	и	3, 7, 1, 2	4, 5, 6

Пример. Найти компоненты сильной связности графа (рис. 2.8).

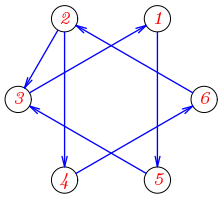


Рис. 2.8

Решение. Матрица смежности графа имеет вид

$$A_1 = A = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

В графе 7 дуг, поэтому наибольший путь будет не длиннее семи. Построим матрицу достижимости:

$$A_7 = \sum_{k=1}^{7} A^k = \begin{bmatrix} 2 & 0 & 2 & 0 & 3 & 0 \\ 3 & 2 & 6 & 3 & 3 & 2 \\ 3 & 0 & 2 & 0 & 2 & 0 \\ 3 & 2 & 3 & 2 & 1 & 3 \\ 2 & 0 & 3 & 0 & 2 & 0 \\ 3 & 3 & 3 & 2 & 3 & 2 \end{bmatrix}.$$

Выделим из этой матрицы главные миноры максимального порядка, не содержащие нули. Если граф связен, то в матрице будут строки, не содержащие нулей. Это строки 2, 4, 6:

$$\begin{bmatrix} . & . & . & . & . & . \\ 3 & 2 & 6 & 3 & 3 & 2 \\ . & . & . & . & . & . \\ 3 & 2 & 3 & 2 & 1 & 3 \\ . & . & . & . & . & . \\ 3 & 3 & 3 & 2 & 3 & 2 \end{bmatrix}.$$

Минор со строками и столбцами с этими номерами соответствует одной компоненте связности:

$$A_{(2,4,6)} = \begin{bmatrix} . & . & . & . & . & . \\ . & 2 & . & 3 & . & 2 \\ . & . & . & . & . & . \\ . & 2 & . & 2 & . & 3 \\ . & . & . & . & . & . \\ . & 3 & . & 2 & . & 2 \end{bmatrix}.$$

Удалим из матрицы строки и столбцы с этими номерами. Получим минор, соответствующий второй компоненте связности:

$$A_{(1,3,5)} = \begin{bmatrix} 2 & . & 2 & . & 3 & . \\ . & . & . & . & . & . \\ 3 & . & 2 & . & 2 & . \\ . & . & . & . & . & . \\ 2 & . & 3 & . & 2 & . \\ . & . & . & . & . & . \end{bmatrix}.$$

Итак, в графе две компоненты сильной связности: подграф с вершинами 1, 3, 5 и подграф с вершинами 2, 4, 6. Изобразим обе компоненты сильной связности в виде отдельных графов (рисунки 2.9, 2.10). Общее число ребер в компонентах меньше размера исходного графа. Дуга [2, 3] не вошла ни в одну компоненту.

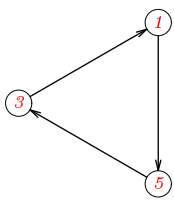


Рис. 2.9

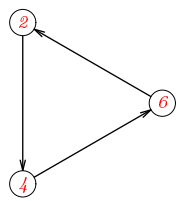


Рис. 2.10

Три варианта решения задачи о нахождении компонент сильной связности графа в системе Maple приведены на с. 109–112.

Глава 3

ДЕРЕВЬЯ

Дерево — связный граф без циклов. *Лес* (или *ациклический* граф) — неограф без циклов. Компонентами леса являются деревья.

Теорема 13. *Для неографа G с n вершинами без петель следующие условия эквивалентны:*

1) *G — дерево;*

2) *G — связный граф, содержащий $n-1$ ребро;*

3) *G — ациклический граф, содержащий $n-1$ ребро;*

4) *любые две несовпадающие вершины графа G соединяет единственная цепь;*

5) *G — ациклический граф, такой, что если в него добавить одно ребро, то в нем появится ровно один цикл.*

Теорема 14. *Неограф G является лесом тогда и только тогда, когда коранг графа $\nu(G) = 0$.*

Висячая вершина в дереве — вершина степени 1. Висячие вершины называются *листьями*, все остальные — *внутренними* вершинами.

Если в дереве особо выделена одна вершина, называемая *корнем*, то такое дерево называется *корневым*, иначе — *свободным*.

Корневое дерево можно считать орграфом с ориентацией дуг из корня или в корень. Очевидно, для любой вершины корневого дерева, кроме корня, $\deg^{\text{in}} = 1$. Для корня $\deg^{\text{in}} = 0$, для листьев $\deg^{\text{out}} = 0$.

Вершины дерева, удаленные на расстояние k (в числе дуг) от корня, образуют k-й *ярус* (уровень) дерева. Наибольшее значение k называется *высотой* дерева.

Если из вершины v корневого дерева выходят дуги, то вершины на концах этих дуг называются сыновьями [1].

3.1. Центроид дерева

Ветвь к вершине v дерева — это максимальный подграф, содержащий v в качестве висячей вершины. *Вес* c_k вершины k — наибольший размер ее ветвей. Центроид [2] дерева $\mathcal{C}$ — множество вершин с наименьшим весом: $\mathcal{C} = \{v | c(v) = c_{\min}\}$ [32].

Вес любого листа дерева равен размеру дерева.

---

[1] В английской литературе и Maple — дочери (daughter). Отсюда, вероятно, и термин «дочерние» (а не «сыновние») компании и т.п.

[2] Или *центр масс* дерева [25].

Высота дерева с корнем, расположенным в центроиде, не больше наименьшего веса его вершин.

Свободное дерево порядка n с двумя центроидами имеет четное количество вершин, а вес каждого центроида равен $n/2$.

Для центроида дерева существует теорема Жордана, аналогичная его же теореме о центре (см. теорему 2 на с. 7).

Теорема 15 (Жордана)**.** *Каждое дерево имеет центроид, состоящий из одной или двух смежных вершин* [32].

Задача. Найти центроид дерева (рис. 3.1) и наименьший вес его вершин.

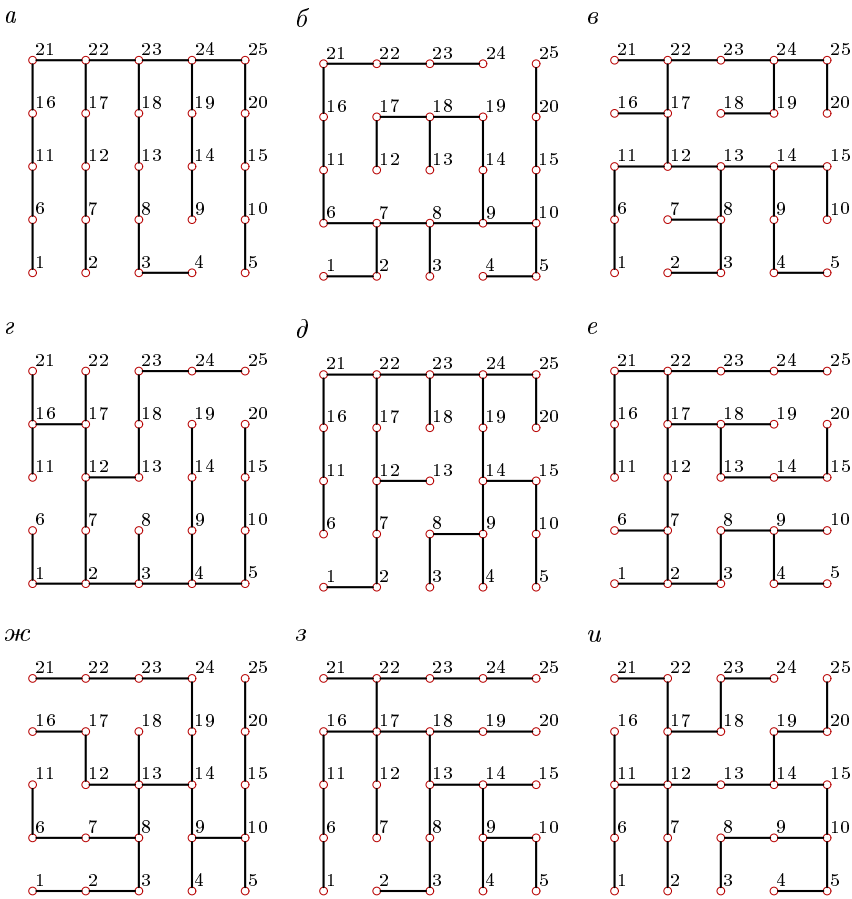


Рис. 3.1

Ответы

	$c_{\min}$	$\mathcal{C}$		$c_{\min}$	$\mathcal{C}$		$c_{\min}$	$\mathcal{C}$
а	10	23	г	12	2	ж	12	14
б	12	9	д	12	23	з	12	18
в	11	12	е	11	17	и	12	12

Пример. Найти наименьший вес вершин дерева (рис. 3.2) и его центроид.

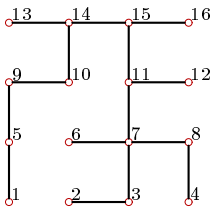


Рис. 3.2

Решение. Очевидно, вес каждой висячей вершины дерева порядка n равен $n-1$. Висячие вершины не могут составить центроид дерева, поэтому исключим из рассмотрения вершины 1, 2, 4, 6, 12, 13 и 16. Для всех остальных вершин найдем их вес, вычисляя длину (размер) их ветвей.

Число ветвей вершины равно ее степени. Размеры ветвей вершин 3, 5 и 8 равны 1 и 14. Следовательно, веса этих вершин равны 14. К вершине 7 подходят четыре ветви размером 1, 2, 2 и 10. Таким образом, ее вес $c_7 = 10$. Аналогично вычисляются веса других вершин: $c_9 = 13$, $c_{10} = c_{14} = 12$, $c_{11} = c_{15} = 8$. Минимальный вес вершин равен 8, следовательно, центроид дерева образуют две вершины с таким весом: 11 и 15.

3.2. Десятичная кодировка

Деревья представляют собой важный вид графов. С помощью деревьев описываются базы данных, деревья моделируют алгоритмы и программы, их используют в электротехнике, химии. Одной из актуальных задач в эпоху компьютерных и телекоммуникационных сетей является задача сжатия информации. Сюда входит и кодировка деревьев. Компактная запись дерева, полностью описывающая его структуру, может существенно упростить как передачу информации о дереве, так и работу с ним. Различные виды кодировки деревьев подробно описаны в [14].

Приведем одну из простейших кодировок помеченных деревьев с выделенным корнем — десятичную.

Кодируя дерево, придерживаемся следующих правил.

1. Кодировка начинается с корня и заканчивается в корне.
2. Каждый шаг на одну дугу от корня кодируется единицей.
3. В узле выбираем направление на вершину с меньшим номером.

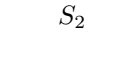
4. Достигнув листа, идем назад, кодируя каждый шаг нулем.

5. При движении назад в узле всегда выбираем направление на непройденную вершину с меньшим номером.

Кодировка в такой форме получается достаточно компактной, однако она не несет в себе информации о номерах вершин дерева. Существуют аналогичные кодировки, где вместо единиц в таком же порядке проставляются номера или названия вершин.

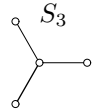
Есть деревья, для которых несложно вывести формулу десятичной кодировки. Рассмотрим, например, графы-звезды $K_{1,n}$, являющиеся полными двудольными графами, одна из долей которых состоит из одной вершины [1]. Другое обозначение звезд — $K_{1,n} = S_n$.

На рисунках 3.3–3.6 приведены звезды и их двоичные и десятичные кодировки. Корень дерева располагается в центральной вершине звезды. Легко получить общую формулу: $Z(S_n) = 2(4^n - 1)/3$.



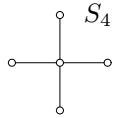
1010
10

Рис. 3.3



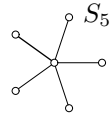
101010
42

Рис. 3.4



10101010
170

Рис. 3.5



1010101010
682

Рис. 3.6

Если корень поместить в любой из висячих вершин, то код Z' такого дерева будет выражаться бóльшим числом. Более того, существует зависимость $Z(S_n) - Z'(S_n) = Z(S_{n-1})$. Аналогично рассматриваются цепи [2] (C_n; рис. 3.7).

В звездах только два варианта расположения корня с различными десятичными кодировками. В цепи же число вариантов кодировок в зависимости от положения корня растет с увеличением n. Рассмотрим самый простой вариант, расположив корень в концевой вершине (листе). Для C_2 получим десятичную кодировку 10 и двоичную 2. Точно так же для остальных цепей: 1100 и 12, 111000 и 56, 11110000

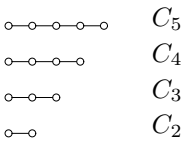


Рис. 3.7

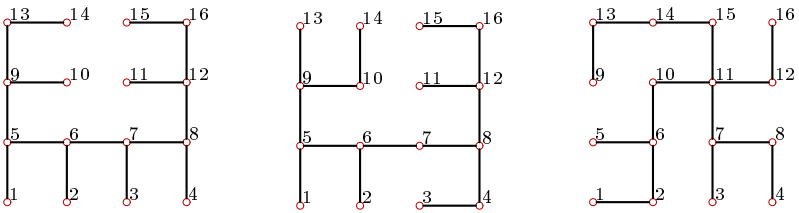
[1] Эта же вершина является центром и центроидом дерева. Наименьший вес вершин звезды равен 1.

[2] У цепей C_{2n} и C_{2n+1} наименьший вес вершин равен n. Центр и центроид цепей совпадают.

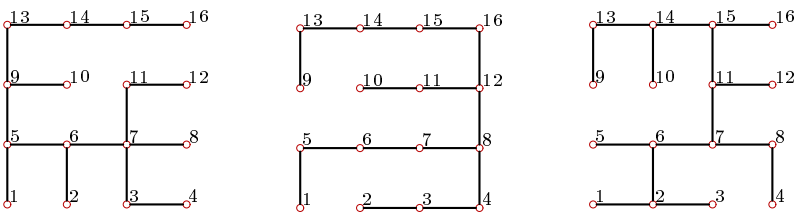
и 240. Общая формула для десятичной кодировки цепи с корнем в концевой вершине имеет вид $Z(C_n) = 2^{n-1}(2^{n-1} - 1)$.

Задача. Записать десятичный код дерева (рис. 3.8) с корнем в вершине 7.

а *б* *в*



г *д* *е*



ж *з* *и*

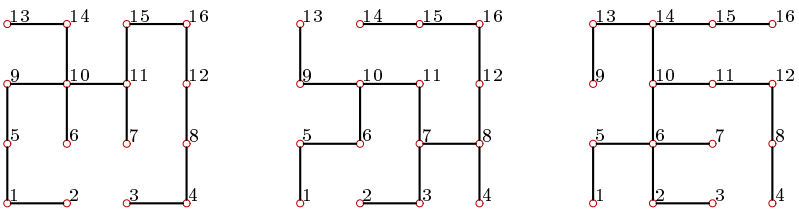


Рис. 3.8

Ответы

	Код(10)		Код(10)		Код(10)
а	766905776	г	862838828	ж	1002643328
б	920926640	д	955371392	з	863518256
в	754522592	е	978115976	и	966725216

Пример. Записать десятичный код дерева (рис. 3.9) с корнем в вершине 3.

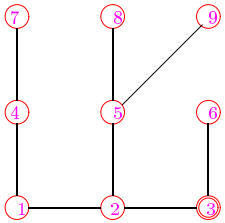


Рис. 3.9

Решение. На основании правила кодировки, двигаясь по дереву, проставим в код единицы и нули. При движении из корня 3 к вершине 7 проходим четыре ребра. В код записываем четыре единицы: 1111. Возвращаясь от вершины 7 к вершине 2 (до ближайшей развилки), проходим три ребра. Записываем в код три нуля: 000. От вершины 2 к 5 и далее к 8 (меньший номер): 11; от 8 назад к 5 и от 5 к 9: 01; от 9 к корню 3: 000.

И наконец, от 3 к 6 и обратно: 10. В итоге, собирая все вместе, получим двоичный код дерева:

$$1\ 111\ 000\ 110\ 100\ 010.$$

Разбивая число на тройки, переводим полученное двоичное представление в восьмеричное [1]. Получаем 1700642_8. Затем переводим это число в десятичное: $2 \cdot 8^0 + 4 \cdot 8^1 + 6 \cdot 8^2 + 0 \cdot 8^3 + 7 \cdot 8^4 + 1 \cdot 8^5 = 61858$.

Можно перевести двоичное число из n цифр в десятичное число непосредственно по формуле

$$\sum_{i=1}^{n} k_i 2^{n-i},$$

где k_i — i-я цифра (0 или 1) в двоичном числе.

Maple-программа для десятичной кодировки приведена на с. 120.

3.3. Кодировка Прюфера

Выбор кодировки дерева зависит от решаемой теоретической или технической задачи. Среди всех возможных кодировок естественно отыскать оптимальные по какому-то качеству решения. Впервые проблемой оптимальности кодировки деревьев занялся А.В. Анисимов (Об оптимальной упаковке деревьев// Кибернетика. — 1976. № 3. С. 89 – 91). Было показано, что существует оптимальный в определенном смысле код дерева — так называемый код Прюфера [2]. Это достаточно редко упоминаемое имя встречается в т. 1 книги Д. Кнута [16] и в книге О. Оре [25] в связи с выводом числа n^{n-2} помеченных деревьев [3].

[1] Имеем $000_2 = 0_8$, $001_2 = 1_8$, $010_2 = 2_8$, $011_2 = 3_8$, $100_2 = 4_8$, $101_2 = 5_8$, $110_2 = 6_8$, $111_2 = 7_8$.

[2] Prufer, Ernst Paul Heinz.

[3] Теорема Кэли (Cayley A.).

Наиболее полно кодировка Прюфера и действия с числами (кодами) Прюфера рассмотрены в фундаментальном труде В.Н. Касьянова и В.А. Евстигнеева [14].

В частности, было показано, как образуется код Прюфера дерева, полученного склейкой (отождествлением вершин) двух других деревьев. Отметим, что кодировка Прюфера применяется к свободным деревьям (неориентированным деревьям, т.е. деревьям, в которых нет выделенного корня).

Приведем алгоритм кодировки помеченного дерева по Прюферу.

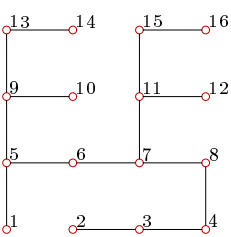
1. Найти висячую вершину с минимальным номером i.

2. Записать в код Прюфера вершину, смежную с i.

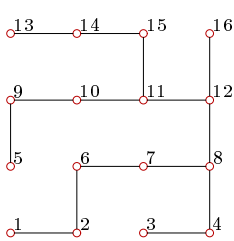
3. Удалить вершину i из дерева. Если дерево не пустое, то перейти к п.1.

Задача. Записать код Прюфера [14] дерева (рис. 3.10).

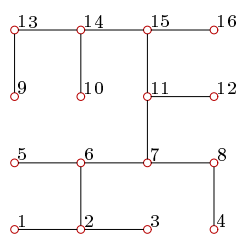
а



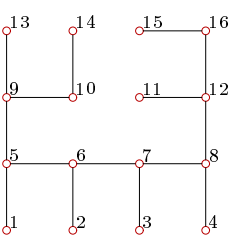
б



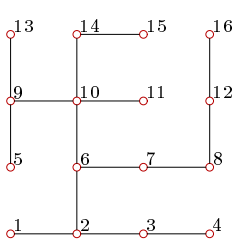
в



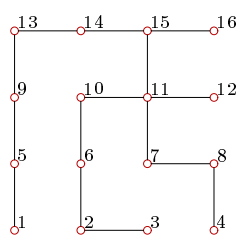
г



д



е



ж *з* *и*

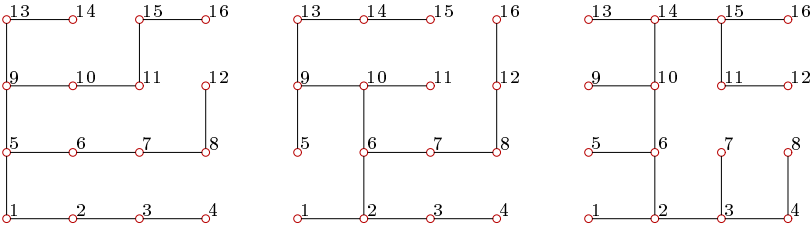


Рис. 3.10

Ответы

	Код Прюфера
а	[5, 3, 4, 8, 7, 9, 11, 13, 9, 5, 6, 7, 11, 15]
б	[2, 6, 4, 8, 9, 7, 8, 12, 10, 11, 14, 15, 11, 12]
в	[2, 2, 6, 8, 6, 7, 7, 11, 13, 14, 11, 15, 14, 15]
г	[5, 6, 7, 8, 12, 9, 10, 9, 5, 6, 7, 8, 12, 16]
д	[2, 3, 2, 6, 9, 10, 9, 10, 14, 10, 6, 7, 8, 12]
е	[5, 2, 6, 8, 9, 10, 7, 11, 13, 11, 11, 15, 14, 15]
ж	[3, 2, 1, 5, 8, 7, 6, 5, 9, 13, 9, 10, 11, 15]
з	[2, 3, 2, 6, 9, 10, 14, 13, 9, 10, 6, 7, 8, 12]
и	[2, 6, 3, 4, 3, 2, 6, 10, 10, 14, 11, 15, 14, 15]

Пример. Записать код Прюфера [14] для дерева (рис. 3.11).

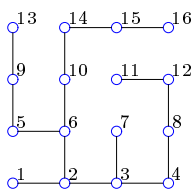
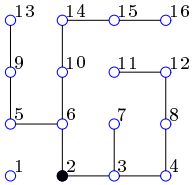
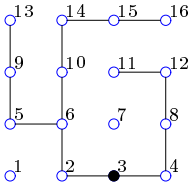


Рис. 3.11

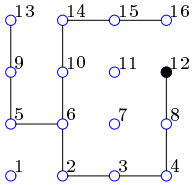
Решение. Находим висячую вершину с минимальным номером, записываем в код Прюфера вершину, смежную с ней, и удаляем ее из дерева. Последовательность определения кода Прюфера для рассматриваемого дерева показана на рисунках 3.12–3.17. Для наглядности изображение удаленной вершины остается на рисунке, а стирается только ребро.



$P = [2]$

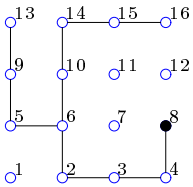


$P = [2,\ 3]$

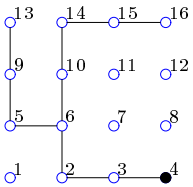


$P = [2,\ 3,\ 12]$

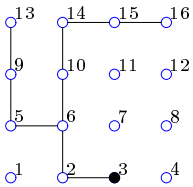
Рис. 3.12



$P = [2,\ 3,\ 12,\ 8]$

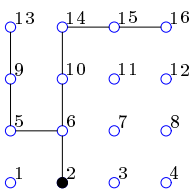


$P = [2,\ 3,\ 12,\ 8,\ 4]$

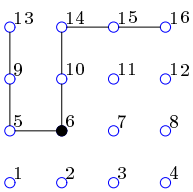


$P = [2,\ 3,\ 12,\ 8,\ 4,\ 3]$

Рис. 3.13

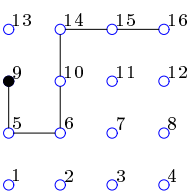


$P = [2,\ 3,\ 12,\ 8,\ 4,\ 3,\ 2]$

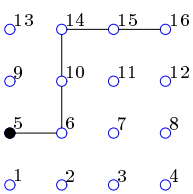


$P = [2,\ 3,\ 12,\ 8,\ 4,\ 3,\ 2,\ 6]$

Рис. 3.14

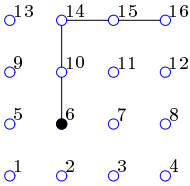


$P = [2,\ 3,\ 12,\ 8,4,\ 3,\ 2,\ 6,\ 9]$

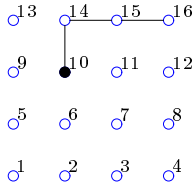


$P = [2,\ 3,\ 12,\ 8,\ 4,\ 3,\ 2,\ 6,\ 9,\ 5]$

Рис. 3.15

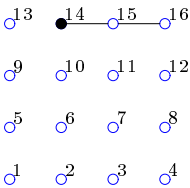


$P = [2, 3, 12, 8, 4, 3, 2, 6, 9, 5, 6]$

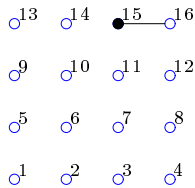


$P = [2, 3, 12, 8, 4, 3, 2, 6, 9, 5, 6, 10]$

Рис. 3.16



$P = [2, 3, 12, 8, 4, 3, 2, 6, 9, 5, 6, 10, 14]$



$P = [2, 3, 12, 8, 4, 3, 2, 6, 9, 5, 6, 10, 14, 15]$

Рис. 3.17

В результате код Прюфера имеет вид

$$P = [2, 3, 12, 8, 4, 3, 2, 6, 9, 5, 6, 10, 14, 15].$$

Вершины в коде Прюфера могут повторяться, более того, в коде Прюфера может быть только одна вершина. Так, если номер центральной вершины звезды на рис. 3.5 равен 5, то код состоит из трех одинаковых цифр — 555.

Две Maple-программы кодировки Прюфера приведены на с. 123.

3.4. Распаковка кода Прюфера

Распаковка кода Прюфера производится по алгоритму, описанному в [14]. Основное требование к алгоритмам кодирования — однозначность восстановления информации.

Задача. По заданному коду Прюфера построить дерево.

а)
[6, 3, 4, 5, 10, 7, 8, 9, 10, 15, 12, 13, 14, 15, 20, 16, 17, 18, 19, 20, 25, 23, 24];
б)
[2, 3, 4, 3, 8, 6, 11, 9, 8, 13, 16, 17, 18, 19, 20, 21, 22, 17, 18, 19, 20, 25, 24];
в)
[2, 7, 4, 4, 9, 11, 8, 9, 9, 14, 12, 17, 14, 21, 18, 22, 23, 18, 19, 14, 15, 20, 25];
г)
[5, 4, 3, 2, 6, 9, 8, 7, 6, 11, 21, 16, 18, 19, 18, 17, 16, 11, 12, 13, 14, 15, 20];

д)
[2, 3, 4, 9, 10, 13, 18, 13, 8, 16, 11, 6, 7, 17, 12, 7, 8, 9, 10, 15, 20, 25, 24];
е)
[6, 2, 7, 9, 10, 11, 7, 12, 14, 15, 12, 17, 17, 22, 18, 15, 14, 13, 18, 23, 22, 23, 24];
ж)
[2, 3, 8, 11, 12, 5, 4, 9, 8, 13, 16, 17, 13, 18, 20, 21, 22, 23, 24, 25, 22, 23, 24];
з)
[6, 7, 8, 4, 9, 9, 11, 14, 15, 14, 16, 17, 18, 17, 16, 11, 6, 7, 8, 9, 14, 19, 24];
и)
[6, 7, 3, 8, 10, 11, 12, 8, 13, 15, 16, 13, 18, 19, 20, 21, 18, 19, 22, 23, 24, 19, 20].

Ответы

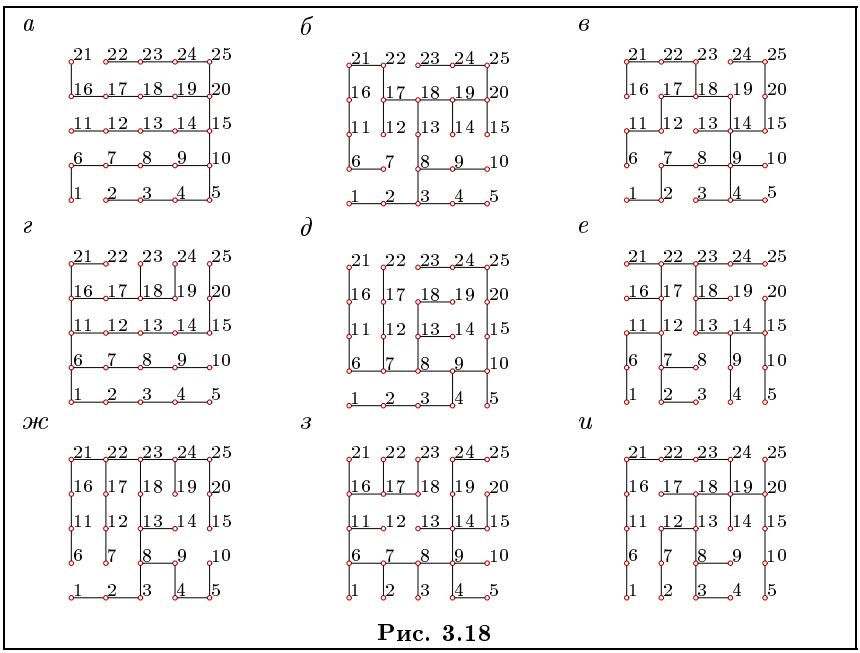


Рис. 3.18

Пример. Построить дерево, соответствующее коду Прюфера $P = [5, 6, 7, 8, 6, 10, 14, 15, 11, 10, 6, 7, 8, 12]$.

Решение. *Алгоритм распаковки кода Прюфера P.*

Введем список (вектор) некоторых элементов $N = [a_1, ..., a_n]$ и операцию укорочения списка на один первый элемент, обозначив ее звездочкой: $N^* = [a_2, a_3, ..., a_n]$. Тогда

1) $A = P$, $N = [1, ..., n]$;
2) $v = \min N$, $v \notin A$, $u = a_1$;
3) вершины u и v соединить ребром;
4) $N = N \setminus v$, $A = A^*$;

5) если $|N| = 2$, то соединить две последние вершины, n_1 и n_2, и завершить процедуру; в противном случае вернуться к п. 2.

Для кода, данного в условии задачи, последовательно получаем

- $A = [5, 6, 7, 8, 6, 10, 14, 15, 11, 10, 6, 7, 8, 12]$,
  $B = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]$,
  ребро $(5,\ 1)$;
- $A = [6, 7, 8, 6, 10, 14, 15, 11, 10, 6, 7, 8, 12]$,
  $N = [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]$,
  ребро $(6,\ 2)$;
- $A = [7, 8, 6, 10, 14, 15, 11, 10, 6, 7, 8, 12]$,
  $N = [3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]$,
  ребро $(7,\ 3)$;
- $A = [8, 6, 10, 14, 15, 11, 10, 6, 7, 8, 12]$,
  $N = [4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]$,
  ребро $(8,\ 4)$;
- $A = [6, 10, 14, 15, 11, 10, 6, 7, 8, 12]$,
  $N = [5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]$,
  ребро $(6,\ 5)$;
- $A = [10, 14, 15, 11, 10, 6, 7, 8, 12]$,
  $N = [6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16]$.
  Вершины 6, 7, 8 есть в A, поэтому вершину 10 (первая из списка A) соединяем с 9; получаем ребро (10, 9). Укорачиваем A и удаляем вершину 9 из N. Курсивом выделены вершины списка N, присутствующие в A. Далее
- $A = [14, 15, 11, 10, 6, 7, 8, 12]$,
  $N = [6, 7, 8, 10, 11, 12, 13, 14, 15, 16]$.
  Вершины 6, 7, 8, 10, 11, 12 из N есть в A, поэтому вершину 14 (первая из A) соединяем с вершиной 13, следующей за списком 6, 7, 8, 10, 11, 12, и получаем ребро (14, 13). Укорачиваем спереди список A, отрезая от него 14, и удаляем вершину 13 из N. Далее
- $A = [15, 11, 10, 6, 7, 8, 12]$,
  $N = [6, 7, 8, 10, 11, 12, 14, 15, 16]$. Аналогично получаем ребро (15, 14). Вершину 15 берем из A, вершину 14 — из N.

- $A = [11, 10, 6, 7, 8, 12]$,
  $N = [6, 7, 8, 10, 11, 12, 15, 16]$,
  ребро (11, 15);
- $A = [10, 6, 7, 8, 12]$,
  $N = [6, 7, 8, 10, 11, 12, 16]$,
  ребро (10, 11);
- $A = [6, 7, 8, 12]$,
  $N = [6, 7, 8, 10, 12, 16]$,
  ребро (6, 10);
- $A = [7, 8, 12]$,
  $N = [6, 7, 8, 12, 16]$,
  ребро (7, 6);
- $A = [8, 12]$,
  $N = [7, 8, 12, 16]$,
  ребро (8, 7);
- $A = [12]$,
  $N = [8, 12, 16]$,
  ребро (12, 8).
- На последнем этапе получаем ребро, образованное двумя вершинами из N:
  $A = [\,]$, $N = [12, 16]$, ребро (12, 16).

Дерево, закодированное по Прюферу, — свободное, т.е. оно не имеет корня. Оно может быть изображено, например, в виде, показанном на рис. 3.19 или на рис. 3.20. В последнем случае в дереве искусственно выделен корень 6. Полученное дерево имеет 6 ярусов [1].

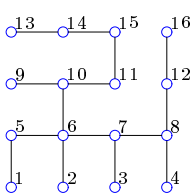


Рис. 3.19

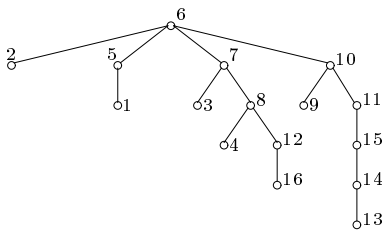


Рис. 3.20

[1] Какую вершину свободного дерева надо принять за корень, чтобы высота дерева была минимальной?

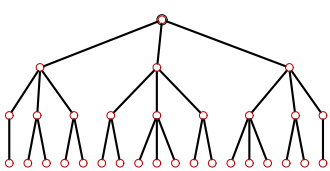
Описанный процесс легко программируется. Соответствующая Maple-программа приведена на с. 125.

3.5. Кодировка Гапта

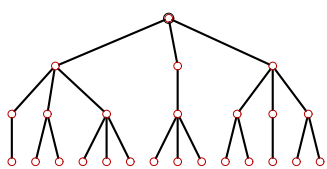
Для деревьев типа 2–3, т.е. деревьев, каждая не концевая вершина которых имеет по 2 или 3 сына, применяется код Гапта [14]. Без какого-либо изменения алгоритма этот код обобщается и на более сложные случаи. Дерево не обязательно должно быть помечено. Кодировка Гапта (в отличие от кодировки Прюфера) не сохраняет информацию об именах вершин.

Задача. Найти код Гапта дерева (рис. 3.21).

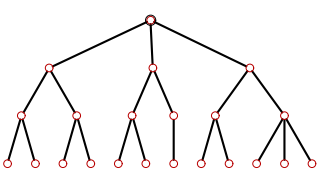
а



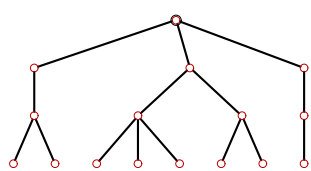
б



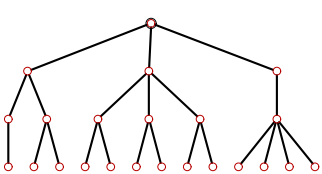
в



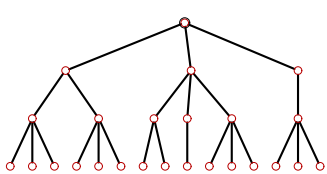
г



д



е



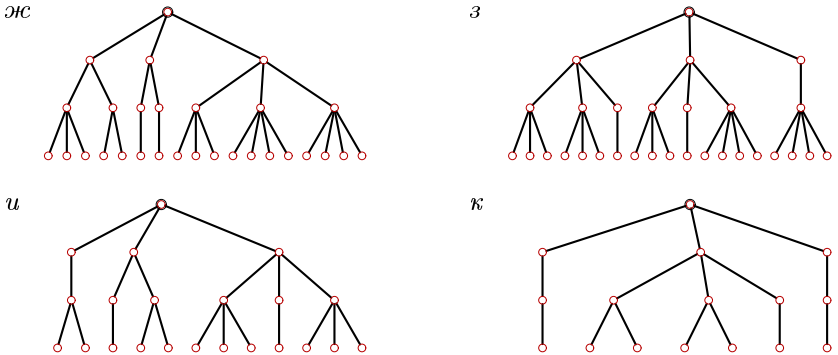


Рис. 3.21

Ответы

	Код		Код
а	[1, 2, 2, 2, 3, 2, 3, 2, 1, 3, 3, 3, 3]	е	[3, 3, 2, 1, 3, 3, 2, 3, 1, 3]
б	[1, 2, 3, 3, 2, 1, 2, 3, 1, 3, 3]	ж	[3, 2, 1, 1, 3, 4, 4, 2, 2, 3, 3]
в	[2, 2, 2, 1, 2, 3, 2, 2, 2, 3]	з	[3, 3, 1, 3, 1, 4, 4, 3, 3, 1, 3]
г	[2, 3, 2, 1, 1, 2, 1, 3]	и	[2, 1, 2, 3, 1, 3, 1, 2, 3, 3]
д	[1, 2, 2, 2, 2, 4, 2, 3, 1, 3]	к	[1, 2, 2, 1, 1, 1, 3, 1, 3]

Пример. Найти код Гапта дерева (рис. 3.22).

Решение. Выберем направление обхода дерева. Пусть код состоит из числа сыновей каждой вершины дерева при обходе дерева *слева направо, снизу вверх*. Висячие вершины (их степень равна 1) не имеют сыновей, поэтому в код дерева порядка n, имеющего $n - n_0$ висячих вершин, войдет $n - -n_0$ чисел. Для дерева на рис. 3.22 кодировка должна содержать $12-8 = 6$ чисел. Начнем кодировку с самой верхней вершины — A. Она имеет три сына — B, C, D. Следовательно, заносим в код число 3:

$$[- - -3].$$

Переходим на следующий ярус. Самая правая вершина, B, имеет три сына. Заносим в код число 3:

$$[- - 3, 3].$$

Продолжая далее, окончательно получаем код [2, 3, 3, 3].

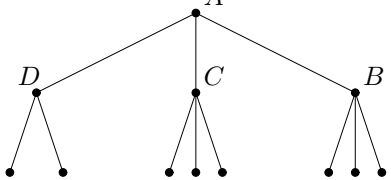


Рис. 3.22

Maple-программа для кодировки Гапта приведена на с. 126.

3.6. Распаковка кода Гапта

Задача. Распаковать код Гапта и построить дерево:

а)
[3, 1, 1, 3, 2, 2, 4, 3, 1, 3, 3];

б)
[2, 2, 2, 3, 2, 4, 4, 2, 3, 3, 2, 3];

в)
[1, 3, 2, 3, 3, 1, 2, 3, 3, 1, 3];

г)
[1, 1, 1, 1, 1, 3, 3, 1, 2, 3];

д)
[2, 3, 2, 2, 3, 3, 4, 2, 2, 3, 3];

е)
[3, 3, 2,1, 1, 2, 2, 1, 3, 3, 2, 3];

ж)
[3, 3, 2, 2, 2, 2, 2, 2, 2, 3];

з)
[1, 1, 2, 2, 3, 1, 2, 2, 3];

и)
[2, 3, 3, 3, 1, 2, 1, 3];

к)
[3, 3, 1, 2, 1, 1, 2, 3, 2, 2, 3].

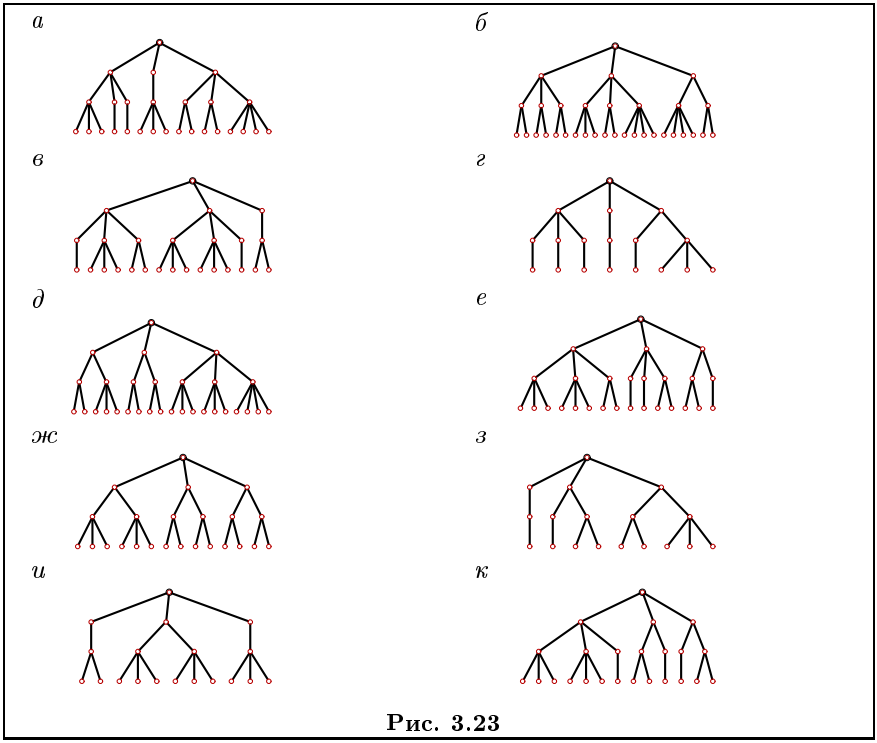


Рис. 3.23

Пример. По заданному коду Гапта $[3,1,1,1,1,4,2,1,2,3,3,3]$, построить дерево.

Решение. Построение начинается с корня. От корня, согласно последнему числу кода, идет три дуги к трем вершинам-сыновьям 2-го яруса. Рассматривая следующие с конца три числа кода, выясняем, что от этих вершин идет 3, 3 и 2 дуги соответственно. Изображаем эту часть дерева и продолжаем строить следующий ярус. В результате получаем рис. 3.24.

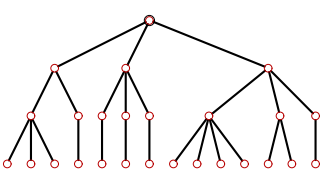


Рис. 3.24

Maple-программа для распаковки кода Гапта приведена на с. 128.

Глава 4

АЛГОРИТМЫ

В этой главе собраны задачи, для решения которых используются либо классические, либо новые алгоритмы, имеющие универсальное применение. Совершенно очевидно, что решение многих задач не является самоцелью. Как правило, это полигон для отладки известных и построения новых методов решения. Большое число алгоритмов на графах с подробным описанием и анализом содержится в [17]. В [11] алгоритмы на графах реализованы в среде Delphi на языке Pascal. Некоторые сведения по теории алгоритмов содержатся в [20].

4.1. Кратчайший путь в орграфе

Дугам во взвешенном орграфе поставлено в соответствие некоторое число (вес). Это может быть евклидово расстояние между вершинами или какая-либо другая числовая характеристика (цена, время, энергия). Ставится задача нахождения кратчайшего пути во взвешенном орграфе от одной заданной вершины до другой. Длина пути — сумма весов дуг пути. Вес дуг может быть положительным, отрицательным, нулевым или бесконечным (что соответствует отсутствию дуги). Здесь мы будем рассматривать положительные веса. Кроме того, предполагается, что решение есть, т.е. указанные вершины соединены путем. Существуют также аналогичные задачи для графов с двойными весами [17].

Задача. Для заданного орграфа (рис. 4.1) найти кратчайший путь от вершины 1 к вершине 12. На рисунке рядом с дугой указан ее вес.

а *б*

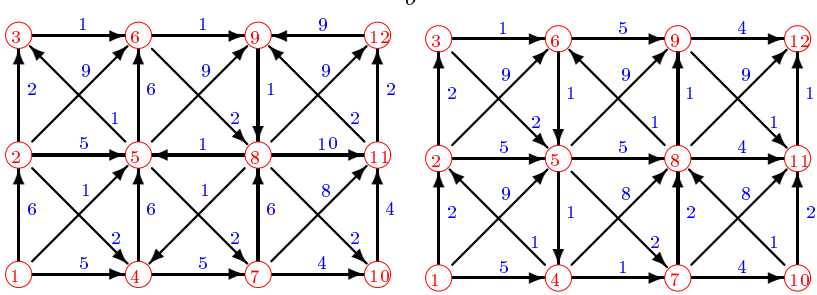
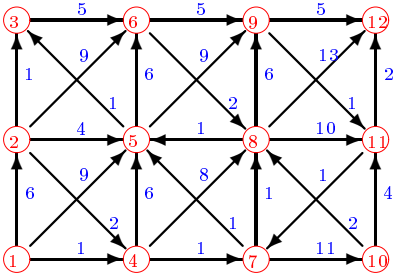
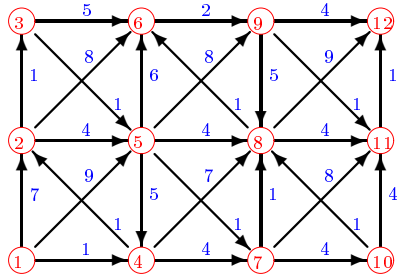


Рис. 4.1

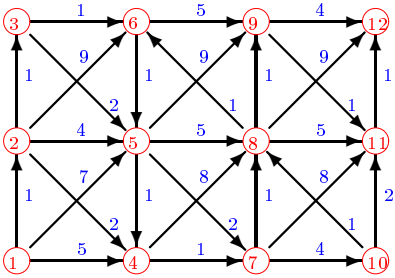
в



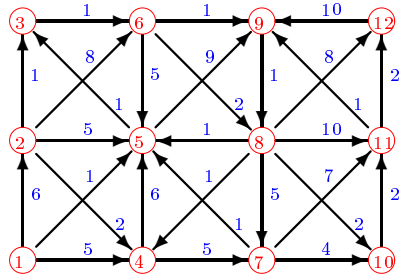
г



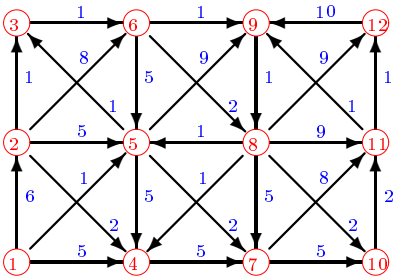
д



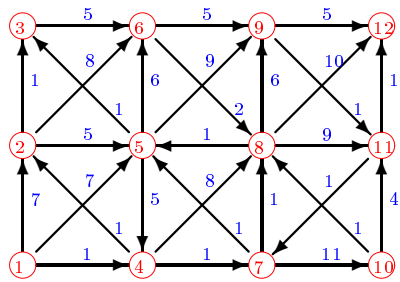
е



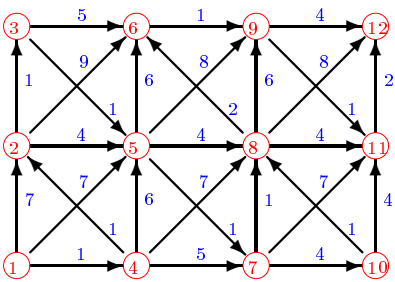
ж



з



и



к

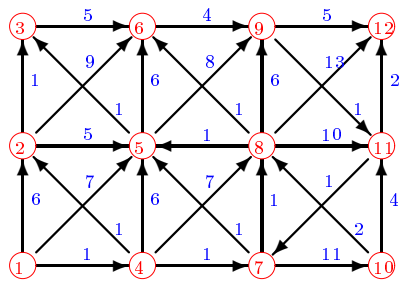


Рис. 4.1 (продолжение)

Ответы

	Путь	Длина		Путь	Длина
а	12 11 7 5 1	13	е	12 11 10 8 6 3 5 1	11
б	12 11 9 8 7 4 1	11	ж	12 11 10 8 6 3 5 1	10
в	12 11 9 8 7 4 1	12	з	12 11 9 8 7 4 1	11
г	12 11 8 7 4 1	11	и	12 11 8 7 5 3 2 4 1	12
д	12 11 9 8 7 4 2 1	8	к	12 11 9 6 8 7 4 1	11

Пример. Дан взвешенный орграф (рис. 4.2). Найти кратчайший путь из вершины A в B.

Решение. Применим алгоритм Е. Дейкстры [1] [10]. Пошаговый алгоритм определения кратчайшего расстояния от вершины A до B состоит в следующем. С каждой вершиной связывается метка. Метка может быть постоянной или временной. Первоначально вершине A приписывается постоянная метка 0, а всем остальным метка ∞. На первом шаге вычисляются расстояния от вершины A с постоянной меткой до всех остальных вершин. Если некоторая вершина не соединена с вершиной с постоянной меткой или дуга направлена в обратную сторону, то расстояние принимается бесконечным. Найденные расстояния являются временными метками вершин. Минимальная из временных меток берется за постоянную. На следующем шаге временные метки всех вершин (кроме тех, у которых постоянные метки) вычисляются как сумма значения последней полученной постоянной метки и расстояния от нее в случае, если это значение не больше предыдущего значения временной метки данной вершины. Таким образом, временная метка вершины может или оставаться прежней, или уменьшаться. Минимальная из временных меток всех вершин опять принимается за постоянную. Процесс продолжается до тех пор, пока вершина B не получит постоянную метку. Значение этой метки и есть кратчайшее расстояние от A до B.

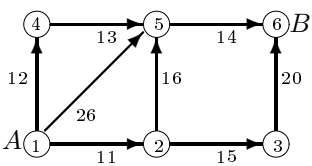


Рис. 4.2

Рассмотрим отдельные шаги решения.

1. Вершина A получает постоянную метку 0, остальные — метку ∞:

[1] Dijkstra E.W.

1	2	3	4	5	6
[0]	∞	∞	∞	∞	∞

.

2. Вычисляем расстояния от вершины 1 с постоянной меткой 0. Вершины 2, 4 и 5 меняют свои временные метки на 11, 12 и 26. Остальные имеют прежние метки — ∞. Очевидно, наименьшей меткой является 11. Она и становится постоянной:

1	2	3	4	5	6
[0]			∞	∞	∞
	[11]	∞	12	26	∞

.

3. Вычисляем расстояния от вершины 2 с постоянной меткой 11. Вершины 3 и 5 имеют расстояния 15 и 16 до вершины 2. Суммируя, получаем значения 26 и 27. Для вершины 5 прежнее значение, 26, было меньше нового значения 27. Следовательно, значение метки 5 не меняем; оно остается равным 26. Из трех временных меток — 12, 26 и 26 — наименьшая принадлежит вершине 4. Эта метка и становится постоянной:

1	2	3	4	5	6
[0]		∞		∞	∞
	[11]	∞	[12]	26	∞
		26			∞

.

4. Вычисляем расстояния от вершины 4 с постоянной меткой 12. Вершина 5 имеет до нее расстояние 13. Суммируя (13+12), получаем значение 25 временной метки вершины 5 вместо прежнего значения 26. Из двух временных меток вершин 3 и 5 наименьшая принадлежит вершине 5. Эта метка и становится постоянной:

1	2	3	4	5	6
[0]		∞			∞
	[11]	∞	[12]		∞
		26			∞
		26		[25]	∞

.

5. На следующем этапе, вычисляя расстояния от вершины 5 с постоянной меткой 25, приходим к конечной вершине B. Но

ее метка $(25 + 14 = 39)$ не становится постоянной, так как она не является минимальной. Расстояние от вершины 5 до вершины 3 принято ∞ (они не соединены). Прежнее значение временной метки вершины 3 меньше ∞. Поэтому метка вершины 3 не меняется. Метка вершины 3 со значением 26, меньшим 39, становится постоянной. На следующем этапе ищем расстояния от нее:

1	2	3	4	5	6
[0]					∞
	[11]		[12]		∞
					∞
				[25]	∞
		[26]			39

6. Расстояние от вершины 3 до вершины 6 составляет 20. Так как $26 + 20 > 39$, значение метки 6 не меняем. На этом шаге она остается прежней и единственной временной меткой. Временная метка вершины 6 становится постоянной, что означает конец процесса. Минимальное расстояние от A до B равно 39.

Две Maple-программы для определения кратчайшего пути в орграфе приведены на с. 115 и 117.

4.2. Поток в сети

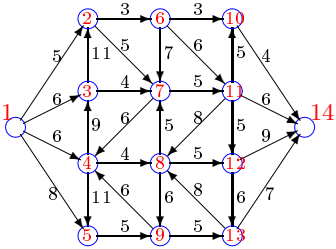
Сетью называют взвешенный орграф с двумя выделенными вершинами: истоком и стоком. Исток имеет нулевую полустепень захода, а сток — нулевую полустепень исхода. Вес дуги означает ее пропускную способность. Поток — еще одно число, приписанное дуге. Поток дуги не больше ее пропускной способности и может меняться. Поток выходит из истока и без потерь, в том же объеме заходит в сток. Условие равновесия (по объему входа и выхода) выполняется и для каждой вершины сети.

Задача о наибольшем потоке в сети — не единственная, но, вероятно, основная задача для потоков в сети. Очевидна возможность практического применения этой задачи для решения транспортных проблем (пробки на дорогах можно условно связывать с насыщением сети или отдельной ее дуги), проблем транспортировки нефтепродуктов или электроэнергии.

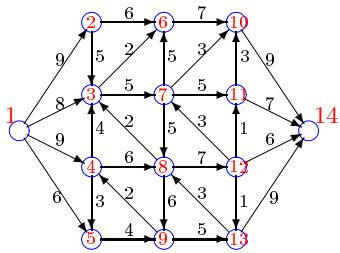
Задача. Задана пропускная способность дуг транспортной сети (рис. 4.3) с началом (истоком) в вершине 1 и концом (стоком) в

вершине 14. Используя алгоритм *Форда–Фалкерсона* [13], найти максимальный поток по сети.

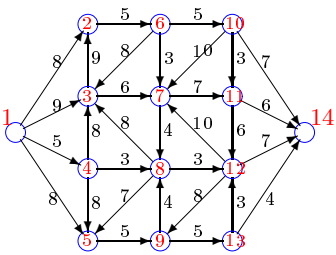
а



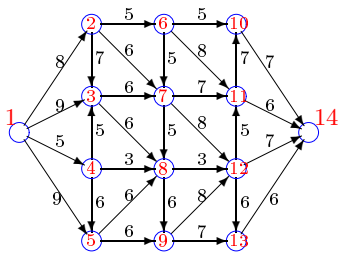
б



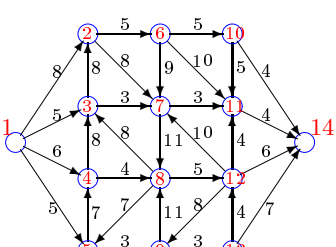
в



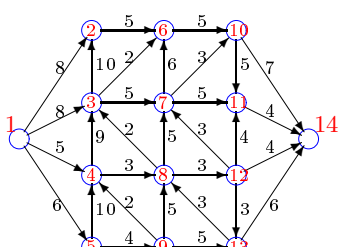
г



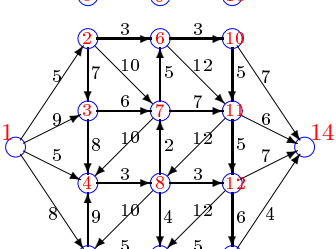
д



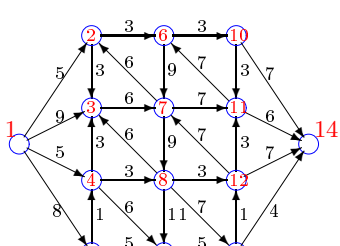
е



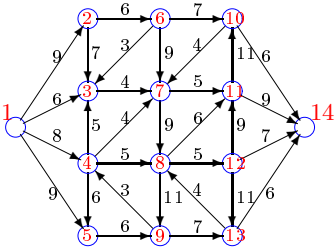
ж



з



и



к

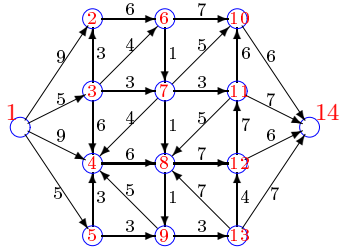


Рис. 4.3

Ответы

	а	б	в	г	д	е	ж	з	и	к
Σ	17	22	19	26	16	17	18	17	24	19

Пример. Задана пропускная способность дуг транспортной сети (рис. 4.4) с началом в вершине 1 и концом в вершине 8. Используя алгоритм *Форда–Фалкерсона*, найти максимальный поток по сети.

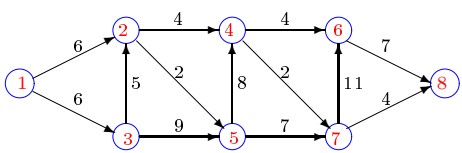


Рис. 4.4

Решение. Алгоритм состоит из двух частей — насыщения потока и его перераспределения. Поток называется насыщенным, если любая цепь из истока в сток содержит насыщенную дугу. В первой части алгоритма разыскиваются цепи из истока в сток и все дуги цепи насыщаются одинаковым возможно большим потоком, определяемым пропускной способностью наиболее «тонкой» дуги или наименьшей разностью между пропускной способностью и потоком в дуге. Различные цепи могут иметь общие дуги. Полученный поток согласован с условием сохранения в узлах (вершинах). Поток, входящий в вершину, равен потоку, выходящему из нее. Поток в сети проходит по *цепям* из истока в сток, т.е. недопустим многократный проход по отдельной дуге. Первая часть задачи считается решенной, если нет ненасыщенных цепей из истока в сток. Первая часть задачи не имеет единственного решения.

Во второй части перераспределение потока выполняется исходя из условия достижения общего по сети максимума потока. Для этого в основании графа (т.е. в графе, в котором снята ориентация дуг) разыскиваются маршруты из истока в сток, состоящие из ребер, соответствующих ненасыщенным дугам, направленным вперед, и непустым дугам, направленным назад. Потоки в дугах прямого направления увеличиваются на величину, на которую уменьшаются потоки в обратных дугах выбранного маршрута. При этом, очевидно, нельзя превышать пропускную способность дуг, направленных вперед, и допускать отрицательных потоков в обратных дугах. В некоторых случаях при удачном выборе цепей в первой части алгоритма перераспределения потока не требуется.

1. **Насыщение потока**. Рассмотрим путь 1–2–4–6–8. Пропустим через этот путь поток, равный 4. При этом дуги [2, 4] и [4, 6] будут насыщенными. Аналогично, путь 1–3–5–7–8 насытим потоком 4. Распределение потока отметим на графе (рис. 4.5). В числителе ставим пропускную способность, в знаменателе — поток. Числитель всегда больше знаменателя, а знаменатель может быть и нулем.

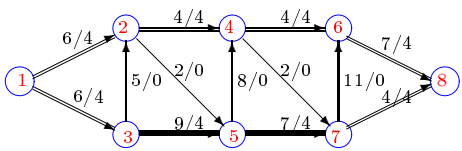


Рис. 4.5

Заметим, что из 1 в 8 есть еще ненасыщенный путь, 1–3–2–5–4–7–6–8,поток в котором можно увеличить на 2. При этом насытятся дуги [1, 3], [2, 5] и [4, 7] (рис. 4.6).

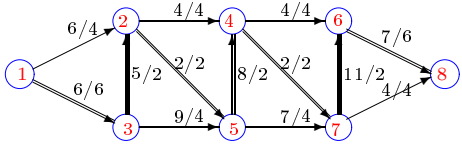


Рис. 4.6

Из 1 в 8 больше нет ненасыщенных путей. По дуге [1,3] двигаться нельзя (она уже насыщена), а движение по дуге [1,2] заканчивается в вершине 2, так как обе выходящие из нее дуги насыщены.

2. **Перераспределение потока.** Найдем последовательность вершин от 1 к 8, такую, что дуги, соединяющие соседние вершины, направленные из 1 в 8, не насыщены, а дуги, направленные в обратном направлении, не пусты. Имеем единственную последовательность: 1–2–3–5–7–6–8. Перераспределяем поток. Поток в дугах прямого направления увеличиваем на 1, а поток в дугах обратного направления уменьшаем на 1. Процесс продолжаем до тех пор, пока одна из прямых дуг не будет насыщена или какая-нибудь обратная дуга не будет пуста.

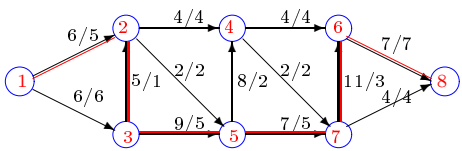


Рис. 4.7

Поток в насыщенной сети можно посчитать по потоку, выходящему из истока 1 или входящему в сток 8. Очевидно, эти числа должны быть равны. Кроме того, для проверки решения следует проверить условие сохранения потока по узлам. Для каждого узла суммарный входящий поток должен быть равен выходящему. В рассматриваемом примере поток равен 11. Распределение потока по дугам при одном и том же суммарном минимальном потоке в сети не единственное.

Maple-программа для определения максимального потока в сети приведена на с. 129.

4.3. Топологическая сортировка сети

Пусть в сети в общем случае имеется несколько истоков и стоков. Стоки и истоки занимают в сети крайние положения. Все остальные вершины могут быть дальше или ближе к ним. Расположение вершины в сети определяется ее *уровнем*. Определим это понятие. Вершины уровня 0 — это истоки; они образуют множество N_0. Если

N_i — множество вершин уровня $i \leqslant k$, то N_{k+1} — множество вершин уровня $k+1$ состоящее из тех и только тех вершин, предшественники которых принадлежат любому из множеств N_0,...,N_k, причем среди этих множеств нет пустых [3].

Порядковой функцией сети называют отображение, сопоставляющее каждой вершине сети ее уровень.

Задача. Найти порядковую функцию сети (рис. 4.8).

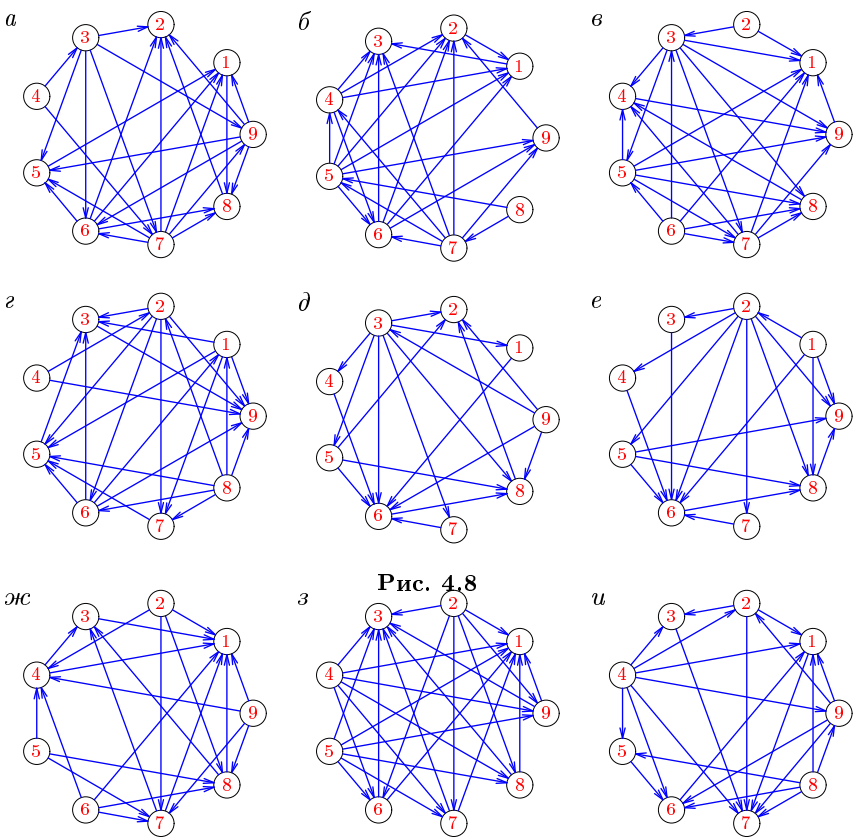


Рис. 4.8 (продолжение)

Ответы

№	Порядковая функция
а	4, 3, 7, 9, 6, 5, 1, 8, 2
б	8, 7, 5, 4, 6, 9, 2, 1, 3
в	(2, 6), 3, 5, 7, 8, 4, 9, 1
г	(4, 8), 2, 6, 1, 7, 5, 3, 9
д	9, 3, (1, 7, 4, 5), 6, 8, 2
е	1, 2, (7, 3, 4, 5), 6, 8, 9
ж	(5, 6, 2, 9), (4, 7, 8), 3, 1
з	(2, 5, 4), (8, 6, 7, 9), (3, 1)
и	(4, 8), (5, 9), (6, 2), (1, 3), 7

Пример. Отсортировать топологически сеть на рис. 4.9.

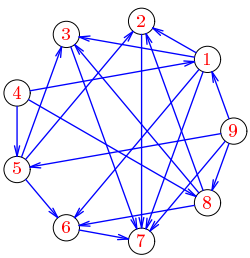


Рис. 4.9

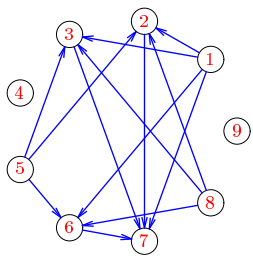


Рис. 4.10

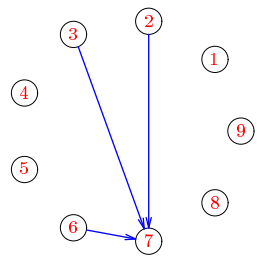


Рис. 4.11

Решение. Находим вершины (4 и 9), полустепень захода которых равна 0. Удаляем эти вершины и дуги, выходящие из них [3]. Удаленные вершины образуют первый уровень упорядоченной сети. Получаем новую сеть (рис. 4.10). В новой сети в вершины 1, 5 и 8 не входят дуги. Удаляем эти вершины (они образуют второй уровень) и получаем сеть, изображенную на рис. 4.11. Очевидно, третий уровень состоит из вершин 2, 3 и 6, а последний (сток сети) — из единственной вершины 7. Изображаем ту же сеть по уровням (рис. 4.12).

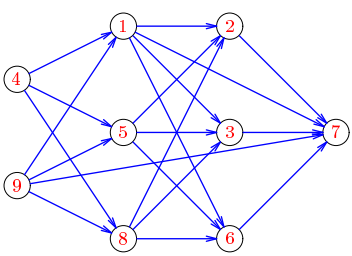


Рис. 4.12

Maple-программа для топологической сортировки сети приведена на с. 133.

4.4. Паросочетание в двудольном графе

Граф называется *двудольным*, если существует такое разбиение множества его вершин на две части, что концы каждого ребра принадлежат разным частям (долям).

Теорема 16 (Кенига [1]). *Для двудольности графа необходимо и достаточно, чтобы он не содержал циклов нечетной длины.*

Если любые две вершины двудольного графа, входящие в разные доли, смежны, то граф называется *полным двудольным*. Обозначение для полного двудольного графа с n и m вершинами в долях — $K_{n,m}$.

Так как по определению в двудольном графе вершины одной доли никогда не соединяются ребрами, для избежания ввода ненужной информации матрицу смежности для двудольного графа можно записать в сокращенной форме: элемент a_{ij} матрицы смежности двудольного графа равен 1, если i-я вершина одной доли соединена с j-й вершиной другой доли; в противном случае $a_{ij} = 0$. Если нумерация вершин графа общая, а не по долям, то j-ю вершину другой доли в определении надо заменить на $(j + n)$-ю вершину графа $K_{n,m}$. Очевидно, матрица двудольного графа в общем случае имеет размер $n \times m$, т.е. не является квадратной.

Паросочетанием графа называется граф, ребра которого являются подмножеством ребер графа, а вершины имеют степень 1.

Паросочетание, не являющееся подмножеством другого паросочетания, называется *максимальным*.

Паросочетание, содержащее наибольшее число ребер, называется *наибольшим*. Наибольшее паросочетание является и максимальным.

Паросочетание, порядок которого равен порядку графа, называется *совершенным*. Совершенное паросочетание включает в себя все вершины двудольного графа. Очевидно, совершенное паросочетание является наибольшим и максимальным. Максимальное паросочетание может быть и не наибольшим. В одном графе могут иметься различные максимальные паросочетания одного порядка.

Число совершенных паросочетаний в двудольном графе, имеющем одинаковое число вершин в долях, можно определить, вычислив перманент [2] его матрицы смежности.

---

[1] König D.

[2] Перманент матрицы (per) равен сумме всех произведений элементов матрицы по одному из каждой строки в разных столбцах. Перманент [25] квадратной матрицы $a_{ij}(i, j = 1, ..., n)$ определяется, подобно определителю, рекуррентным образом. При $n = 1$ имеем $\mathrm{per}(A_1) = a_{11}$. Для матрицы A_{n+1} размером $n + 1$ получим разложение по первой строке: $\mathrm{per}(A_{n+1}) = \sum_{k=1}^{n} a_{1k}\mathrm{per}(A_k)$, где $\mathrm{per}(A_k)$ — перманент матрицы размером n, полученной из A_{n+1} вычеркиванием первой строки и k-го столбца. Таким образом,

Матрица смежности двудольного графа на рис. 4.13 имеет вид

$$\begin{vmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix}.$$

Перманент этой матрицы равен 4. Все четыре покрытия графа изображены на рис. 4.14–4.17.

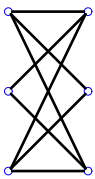
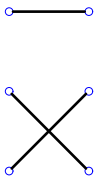
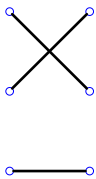
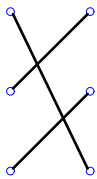
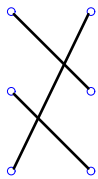
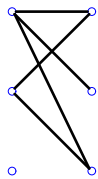







Рис. 4.13 **Рис. 4.14** **Рис. 4.15** **Рис. 4.16** **Рис. 4.17** **Рис. 4.18**

Если граф не имеет совершенного паросочетания, то перманент его матрицы равен нулю. Например, матрица графа с одной изолированной вершиной на рис. 4.18 имеет вид

$$\begin{vmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 0 & 0 \end{vmatrix}.$$

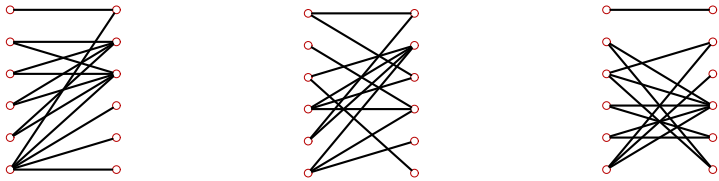
Перманент этой матрицы равен 0 [1].

перманент отличается от определителя только тем, что все его слагаемые берутся со знаком плюс. Например, $\operatorname{per}\begin{vmatrix} 1 & 2 \\ 3 & 4 \end{vmatrix} = 1 \cdot 4 + 3 \cdot 2$.

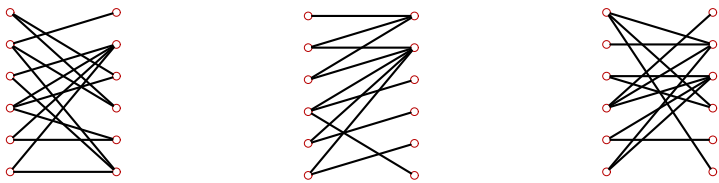
[1] **Теорема 17** [24]. *Перманент матрицы A, состоящей из 0 и 1, порядка n, равен нулю тогда и только тогда, когда в A существует подматрица из нулей размером $s \times t$, где $s + t = n + 1$.*

Задача. В заданном двудольном графе (рис. 4.19) найти число совершенных паросочетаний P и одно из наибольших паросочетаний.

а *б* *в*



г *д* *е*



ж *з* *и*

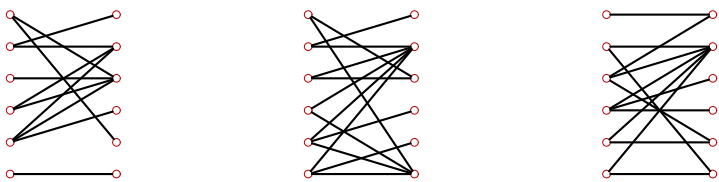
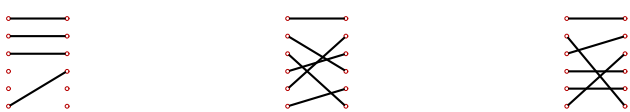


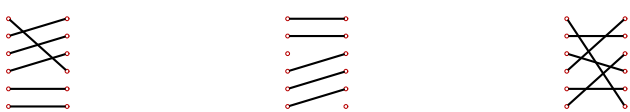
Рис. 4.19

Ответы

а. $P = 0$ *б.* $P = 2$ *в.* $P = 2$



г. $P = 2$ *д.* $P = 0$ *е.* $P = 1$



ж. $P = 1$ *з.* $P = 2$ *и.* $P = 0$

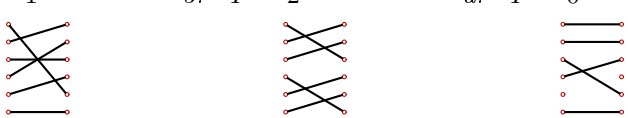


Рис. 4.20

Пример. В заданном двудольном графе (рис. 4.21) найти наибольшее паросочетание.

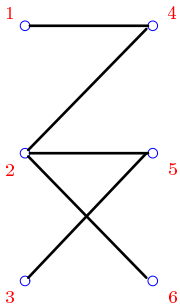


Рис. 4.21

Решение. Применим алгоритм Форда–Фалкерсона (см. с. 65). Образуем из двудольного графа сеть, добавляя исток 7 и сток 8 и ориентируя *ребра* из истока в сток. Пропускную способность всех полученных *дуг* положим равной 1 (рис. 4.22). Максимальный поток по этой сети соответствует искомому паросочетанию графа. Заметим, что решение будет не единственным.

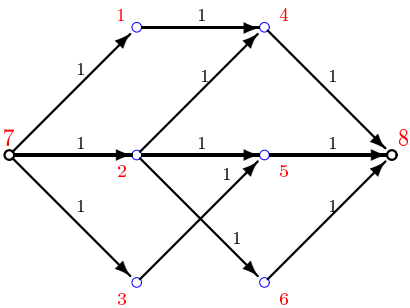


Рис. 4.22

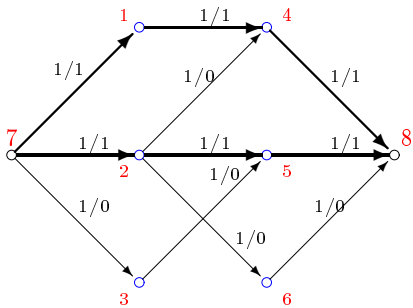


Рис. 4.23

Находим все возможные маршруты из истока в сток и насыщаем их. Таких маршрутов два: 7–1–4–8 и 7–2–5–8 (рис. 4.23). Маршрутов из 7 в 8, по которым можно было бы пропустить дополнительный поток, больше нет [1]. Перераспределяем полученный поток. Находим пути из истока в сток, содержащие ненасыщенные прямые дуги и непустые обратные. В данной задаче это путь 7–3–5–2–6–8. Уменьшаем поток в обратных дугах и увеличиваем на эту же величину поток в прямых дугах (рис. 4.24). В покрытие исходного двудольного графа войдут ребра с потоком, равным 1 (рис. 4.25).

[1] Иногда для фиксации завершения первого этапа алгоритма насыщенные дуги из графа убирают и к следующему этапу переходят, если сеть оказалась разорванной, т.е. исток оторван от стока.

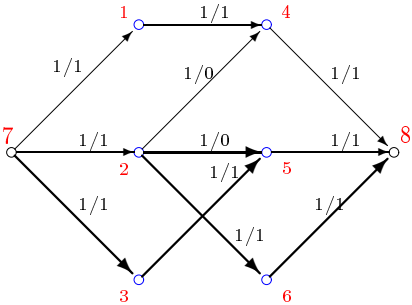


Рис. 4.24

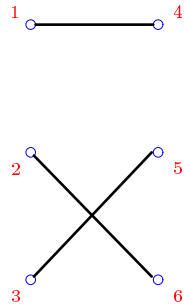


Рис. 4.25

Матрица смежности исходного двудольного графа имеет вид

$$\begin{vmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{vmatrix}.$$

Перманент этой матрицы равен 1. Следовательно, нами найдено единственное совершенное покрытие.

Maple-процедура, реализующая другой алгоритм нахождения наибольшего паросочетания двудольного графа, приведена на с. 137. Пример использования этой процедуры дан на с. 140.

4.5. Задача о назначениях

Известно, что совершенное паросочетание в двудольном графе может быть не единственным. Поэтому, если граф взвешенный, т.е. каждое ребро наделено весом, то естественно поставить задачу о поиске паросочетания с наибольшим или наименьшим весом. Очевидна возможность практического применения решения этой задачи. Рассмотрим, например, задачу о назначении группы n сотрудников по n рабочим местам. Каждый из сотрудников может работать на любом из этих мест, но оплата труда везде разная. Обозначим через a_{ij} оплату труда сотрудника i на рабочем месте j. Матрица коэффициентов a_{ij} в общем случае несимметрична. Оптимальное распределение сотрудников соответствует минимальным расходам на производство. Представляя сотрудников вершинами одной доли графа, а рабочие места — вершинами другой доли, получим полный двудольный взвешенный граф. Совершенное паросочетание наименьшего веса является решением задачи.

Задача. Оплата труда работника i на рабочем месте j определяется коэффициентом a_{ij} (рис. 4.26). Найти одно из оптимальных назначений и суммарные затраты Σ на производство.

а

$$A = \begin{vmatrix} 17 & 18 & 16 & 18 & 18 & 18 & 18 \\ 18 & 39 & 18 & 47 & 45 & 48 & 55 \\ 26 & 30 & 18 & 58 & 59 & 62 & 66 \\ 25 & 29 & 18 & 50 & 51 & 54 & 61 \\ 30 & 37 & 18 & 33 & 57 & 60 & 64 \\ 30 & 34 & 18 & 33 & 34 & 60 & 64 \\ 39 & 40 & 18 & 39 & 43 & 40 & 74 \end{vmatrix}$$

б

$$A = \begin{vmatrix} 21 & 21 & 37 & 29 & 30 & 33 & 45 \\ 16 & 16 & 48 & 43 & 38 & 44 & 56 \\ 19 & 16 & 42 & 46 & 44 & 50 & 59 \\ 25 & 16 & 29 & 50 & 48 & 54 & 66 \\ 24 & 16 & 34 & 29 & 42 & 48 & 57 \\ 16 & 15 & 16 & 16 & 16 & 15 & 16 \\ 36 & 16 & 46 & 38 & 36 & 36 & 75 \end{vmatrix}$$

в

$$A = \begin{vmatrix} 20 & 19 & 19 & 16 & 19 & 19 & 19 \\ 19 & 39 & 49 & 19 & 49 & 51 & 53 \\ 28 & 34 & 56 & 19 & 56 & 58 & 57 \\ 25 & 31 & 26 & 19 & 47 & 49 & 48 \\ 29 & 38 & 42 & 19 & 63 & 65 & 64 \\ 34 & 37 & 38 & 19 & 38 & 69 & 68 \\ 36 & 36 & 46 & 19 & 40 & 39 & 66 \end{vmatrix}$$

г

$$A = \begin{vmatrix} 36 & 36 & 47 & 37 & 44 & 41 & 58 \\ 28 & 47 & 43 & 42 & 49 & 49 & 60 \\ 23 & 23 & 36 & 35 & 39 & 42 & 56 \\ 17 & 23 & 23 & 11 & 23 & 23 & 23 \\ 23 & 35 & 34 & 27 & 48 & 48 & 62 \\ 23 & 29 & 25 & 24 & 25 & 42 & 56 \\ 23 & 42 & 47 & 37 & 41 & 35 & 80 \end{vmatrix}$$

д

$$A = \begin{vmatrix} 30 & 32 & 31 & 46 & 40 & 30 & 50 \\ 21 & 23 & 28 & 43 & 40 & 21 & 47 \\ 21 & 17 & 33 & 45 & 45 & 17 & 55 \\ 21 & 17 & 17 & 12 & 17 & 11 & 17 \\ 33 & 32 & 37 & 37 & 57 & 17 & 67 \\ 28 & 24 & 26 & 32 & 29 & 17 & 57 \\ 37 & 39 & 35 & 44 & 38 & 17 & 72 \end{vmatrix}$$

е

$$A = \begin{vmatrix} 24 & 36 & 32 & 24 & 30 & 39 & 43 \\ 24 & 53 & 58 & 24 & 47 & 56 & 63 \\ 24 & 33 & 42 & 24 & 40 & 46 & 56 \\ 28 & 40 & 39 & 24 & 52 & 58 & 65 \\ 25 & 34 & 33 & 24 & 39 & 45 & 52 \\ 24 & 24 & 24 & 19 & 24 & 14 & 24 \\ 32 & 44 & 34 & 24 & 29 & 35 & 63 \end{vmatrix}$$

ж

$$A = \begin{vmatrix} 15 & 15 & 23 & 35 & 36 & 39 & 34 \\ 16 & 15 & 41 & 41 & 45 & 48 & 46 \\ 19 & 15 & 44 & 47 & 51 & 51 & 46 \\ 15 & 13 & 15 & 18 & 15 & 15 & 15 \\ 30 & 15 & 38 & 38 & 63 & 63 & 61 \\ 30 & 15 & 32 & 38 & 36 & 63 & 61 \\ 28 & 15 & 30 & 39 & 34 & 37 & 54 \end{vmatrix}$$

з

$$A = \begin{vmatrix} 30 & 36 & 35 & 42 & 37 & 51 & 42 \\ 24 & 29 & 34 & 41 & 39 & 47 & 41 \\ 13 & 23 & 20 & 23 & 23 & 23 & 23 \\ 24 & 30 & 32 & 44 & 42 & 53 & 47 \\ 23 & 23 & 31 & 26 & 42 & 53 & 47 \\ 23 & 34 & 33 & 34 & 32 & 69 & 63 \\ 23 & 34 & 27 & 31 & 26 & 37 & 48 \end{vmatrix}$$

и

$$A = \begin{vmatrix} 12 & 26 & 26 & 26 & 19 & 23 & 23 \\ 28 & 56 & 49 & 48 & 28 & 54 & 57 \\ 27 & 38 & 59 & 52 & 27 & 52 & 58 \\ 26 & 31 & 30 & 44 & 26 & 50 & 56 \\ 26 & 34 & 36 & 26 & 26 & 43 & 49 \\ 27 & 32 & 31 & 27 & 23 & 45 & 51 \\ 33 & 35 & 43 & 30 & 23 & 27 & 57 \end{vmatrix}$$

к

$$A = \begin{vmatrix} 15 & 18 & 14 & 18 & 18 & 18 & 18 \\ 19 & 30 & 18 & 40 & 44 & 42 & 47 \\ 28 & 33 & 18 & 53 & 54 & 55 & 60 \\ 25 & 30 & 18 & 41 & 45 & 46 & 51 \\ 27 & 29 & 18 & 30 & 57 & 58 & 63 \\ 28 & 30 & 18 & 28 & 32 & 54 & 59 \\ 33 & 41 & 18 & 36 & 37 & 38 & 66 \end{vmatrix}$$

Рис. 4.26

Ответы

№	Σ	Веса назначения	№	Σ	Веса назначения
а	190	18, 18, 18, 29, 33, 34, 40	е	200	43, 24, 33, 24, 33, 14, 29
б	172	30, 16, 16, 29, 29, 16, 36	ж	180	23, 16, 15, 15, 38, 36, 37
в	194	19, 19, 34, 26, 19, 38, 39	з	201	42, 29, 23, 32, 26, 23, 26
г	205	41, 43, 23, 23, 27, 25, 23	и	193	23, 28, 27, 31, 26, 31, 27
д	169	31, 21, 17, 17, 37, 29, 17	к	185	18, 19, 18, 30, 30, 32, 38

Пример. Оплата труда работника i на рабочем месте j определяется коэффициентом a_{ij}:

$$A = \begin{vmatrix} 1 & 7 & 1 & 3 \\ 1 & 6 & 4 & 6 \\ 17 & 1 & 5 & 1 \\ 1 & 6 & 10 & 4 \end{vmatrix}.$$

Найти одно из оптимальных назначений и суммарные затраты на производство.

Решение. Применим для решения венгерский алгоритм [2]. Алгоритм основан на теории чередующихся цепей Дж. Петерсена. Преобразуем матрицу A. Вычтем из каждой строки минимальный элемент. Получим

$$A' = \begin{vmatrix} 0 & 6 & 0 & 2 \\ 0 & 5 & 3 & 5 \\ 16 & 0 & 4 & 0 \\ 0 & 5 & 9 & 3 \end{vmatrix}.$$

Изобразим двудольный граф, в котором ребра соответствуют нулевым элементам матрицы A' (рис. 4.27).

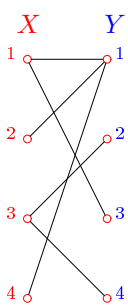


Рис. 4.27

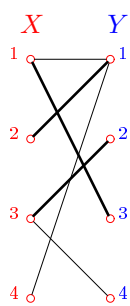


Рис. 4.28

Рассмотрим в этом графе какое-либо максимальное паросочетание. Выделим соответствующие ребра (рис. 4.28).

Чередующейся [1] цепи из X в Y нет. Следовательно, это паросочетание является наибольшим. Выделим множества вершин, не входящие в паросочетание. Это $X_m = \{x_4\}, Y_m = \{y_4\}$. Составим множества вершин, которые входят в цепи [2], соединяющие X_m и X:

$$X' = \{x_2, x_4\},\ Y' = \{y_1\}.$$

Преобразуем матрицу A'. Найдем минимальный элемент в строках с номерами элементов множества X' и столбцах с номерами элементов множества $Y \backslash Y'$, т.е в матрице

$$\begin{vmatrix} . & . & . & . \\ . & 5 & 3 & 5 \\ . & . & . & . \\ . & 5 & 9 & 3 \end{vmatrix}.$$

Минимальный элемент равен 3. Вычтем 3 из строк 2 и 4 и добавим 3 к столбцу 1. Получим

$$A'' = \begin{vmatrix} 3 & 6 & 0 & 2 \\ 0 & 2 & 0 & 2 \\ 19 & 0 & 4 & 0 \\ 0 & 2 & 6 & 0 \end{vmatrix}.$$

Соответствующий граф (см. рис. 4.28) изменился. Исчезло ребро (1, 1), и добавились ребра (2, 3) и (4, 4) (рис. 4.29). В этом графе есть ребро (4, 4), не входящее в выделенное паросочетание и не инцидентное его вершинам, но соединяющее X и Y. Паросочетание стало совершенным (рис. 4.30), множества X_m и Y_m пусты, следовательно, задача решена. Таким образом, выбирая элементы исходной матрицы A с номерами ребер полученного паросочетания, найдем наименьшие затраты: $\Sigma = a_{13} + a_{21} + a_{32} + a_{44} = 1 + 1 + 1 + 4 = 7$.

---

[1] Чередующаяся цепь состоит из нечетного числа чередующихся тонких (из X в Y) и толстых (в обратном направлении) ребер, начиная и кончая тонким ребром. Если бы такая цепь существовала, то число ребер в паросочетании можно было бы увеличить, поменяв толстые и тонкие ребра в этой цепи.

[2] От X к Y можно двигаться по тонким ребрам, в обратном направлении — по толстым.

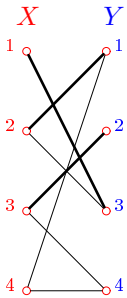


Рис. 4.29

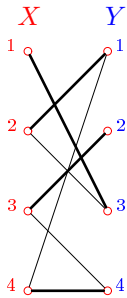


Рис. 4.30

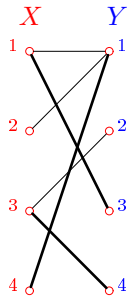


Рис. 4.31

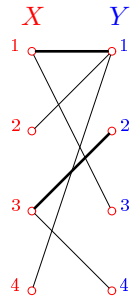


Рис. 4.32

Замечание 1. Паросочетание на рис. 4.28 не единственное. Имеется еще одно наибольшее (3 ребра) паросочетание (рис. 4.31). Множества вершин, не входящие в паросочетание, — это $X_m = \{x_2\}$ и $Y_m = \{y_2\}$. При этом множества X' и Y' не изменятся:

$$X' = \{x_2, x_4\}, \quad Y' = \{y_1\}.$$

Замечание 2. Паросочетание на рис. 4.28 сразу оказалось наибольшим. Однако можно было бы выбрать и не наибольшее, но максимальное паросочетание (рис. 4.32). В этом случае, обратив чередующуюся цепь x_4, y_1, x_1, y_3 (толстые ребра меняются на тонкие, и наоборот), получим наибольшее покрытие (рис. 4.31).

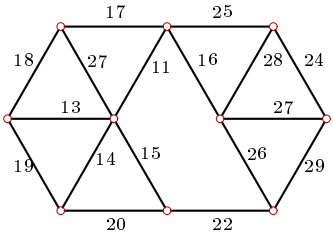
4.6. Остов наименьшего веса

Остовом графа G называют граф, не содержащий циклов и состоящий из ребер графа G и всех его вершин. Таким образом, остов графа является деревом. Число ребер остова равно рангу графа ($\nu^* = n - k$). Число остовов графа можно вычислить по матрице Кирхгофа (см. с. 80).

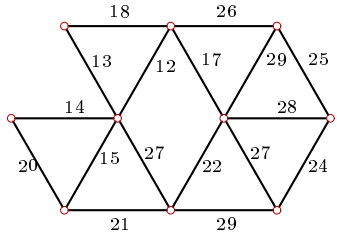
Задача. Дан взвешенный граф (рис. 4.33). Найти число остовов графа и остов графа минимального веса.

В таблице ответов на с. 80 даны веса ребер минимального остова, суммарный вес остова Σ и число остовов графа n_0.

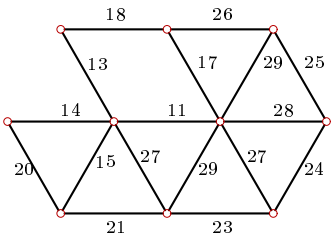
а



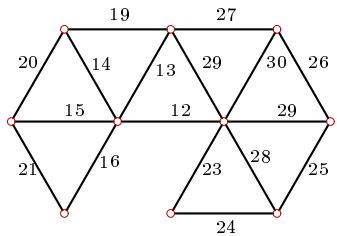
б



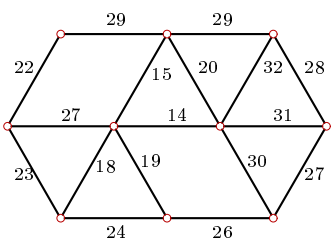
в



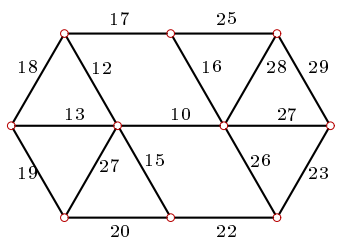
г



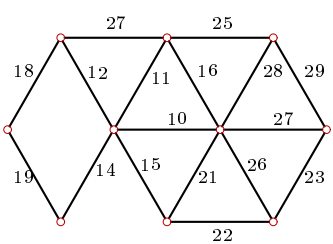
д



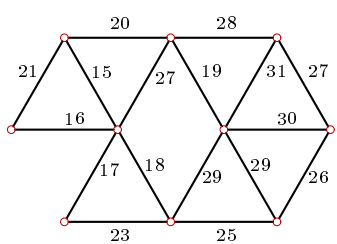
е



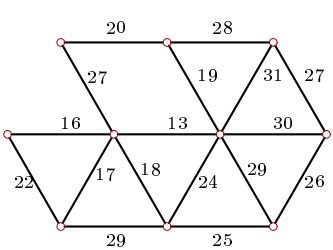
ж



з



и



к

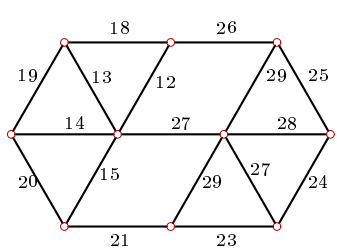


Рис. 4.33

Ответы

№	Веса ребер остова	Σ	n_0
а	16, 11, 13, 14, 15, 17, 22, 25, 24	157	3861
б	17, 12, 13, 14, 15, 21, 26, 25, 24	167	3289
в	11, 13, 14, 15, 17, 21, 23, 24, 25	163	3115
г	12, 13, 14, 15, 16, 23, 24, 25, 26	168	2584
д	14, 15, 18, 19, 23, 22, 26, 27, 28	192	3910
е	10, 12, 13, 15, 16, 19, 22, 23, 25	155	4095
ж	10, 11, 12, 14, 15, 18, 22, 23, 25	150	2944
з	19, 20, 15, 16, 17, 18, 25, 26, 27	183	3289
и	13, 16, 17, 18, 19, 20, 25, 26, 27	181	3115
к	27, 12, 13, 14, 15, 21, 23, 24, 25	174	4095

Пример. Дан взвешенный граф (рис. 4.34).

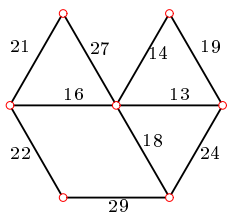


Рис. 4.34

Найти число остовов графа и остов графа минимального веса (экстремальное дерево).

Решение. *Число остовов графа.* Число остовов графа можно вычислить по его матричным характеристикам. Известна следующая теорема.

Теорема 18 (Кирхгофа [1]). *Число остовов графа равно алгебраическому дополнению любого элемента матрицы Кирхгофа.*

Элементы матрицы Кирхгофа B графа G определяются следующим образом. Если вершины i и j графа G смежны, то $b_{ij} = -1$, а если вершины i и j не смежны, то $b_{ij} = 0$. Диагональные элементы матрицы Кирхгофа B равны степеням вершин: $b_{ii} = \deg(i)$. Очевидно свойство: сумма элементов в каждой строке и каждом столбце матрицы Кирхгофа равна 0. Известно также, что алгебраические дополнения всех элементов матрицы Кирхгофа равны, а ранг матрицы Кирхгофа

[1] Kirchhoff G. (1824–1887гг.) — немецкий физик и механик.

неографа порядка n можно вычислить по формуле $\operatorname{rank} B = n - k$, где k — число компонент графа.

Пронумеруем вершины графа (рис. 4.35).

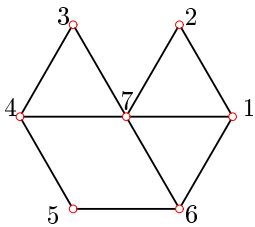


Рис. 4.35

В соответствии с определением запишем матрицу Кирхгофа:

$$B = \begin{bmatrix} 3 & -1 & 0 & 0 & 0 & -1 & -1 \\ -1 & 2 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & 2 & -1 & 0 & 0 & -1 \\ 0 & 0 & -1 & 3 & -1 & 0 & -1 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ -1 & 0 & 0 & 0 & -1 & 3 & -1 \\ -1 & -1 & -1 & -1 & 0 & -1 & 5 \end{bmatrix}.$$

Рассмотрим минор элемента $b_{1,1}$:

$$m_{1,1} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & -1 \\ 0 & 2 & -1 & 0 & 0 & -1 \\ 0 & -1 & 3 & -1 & 0 & -1 \\ 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & -1 & 3 & -1 \\ -1 & -1 & -1 & 0 & -1 & 5 \end{bmatrix}.$$

Вычислим определитель: $\det m_{1,1} = 79$. Таким образом, граф имеет 79 остовов, среди которых надо выбрать экстремальное дерево, т.е дерево, обладающее некоторыми эстремальными свойствами, в данным случае — минимальным весом.

Заметим, что матрицу Кирхгофа можно найти, используя формулу

$$B = II^T - 2A, \tag{4.1}$$

где I и A — матрицы инцидентности и смежности графа. Матрицу инцидентности запишем для следующей последовательности номеров ребер: $(3,7)$, $(6,7)$, $(1,2)$, $(6,1)$, $(5,6)$, $(3,4)$, $(2,7)$, $(4,7)$, $(1,7)$, $(4,5)$. Строки матрицы инцидентности соответствуют вершинам, столбцы — ребрам. В неографе матрица инцидентности I состоит из нулей и

единиц, а матрица смежности A симметричная:

$$I = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 \end{bmatrix}.$$

Формула (4.1) будет проще, если рассмотреть некоторую *ориентацию графа*. Под ориентацией графа понимают орграф, получающийся заменой каждого ребра дугой произвольного направления. Рассмотрим, например, вариант, приведенный на рис. 4.36.

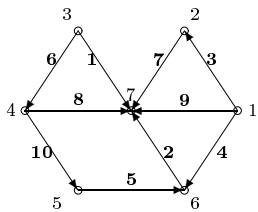


Рис. 4.36

Номера дуг на рисунке указаны полужирным шрифтом в соответствии с номерами ребер исходного графа. Матрица инцидентности I_1 ориентации графа имеет вид

$$I_1 = \begin{bmatrix} 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & -1 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}.$$

Сумма элементов в любом столбце этой матрицы всегда равна 0. Матрица Кирхгофа может быть вычислена по формуле

$$B = I_1 I_1^T. \tag{4.2}$$

Алгоритм Дж. Краскала построения остова минимального веса. Строим граф, присоединяя к пустому графу на множестве вершин заданного графа ребро наименьшего веса [10]. К полученному графу последовательно присоединяем остальные ребра, выбирая на

каждом шаге ребро наименьшего веса, не образующее цикл с имеющимися ребрами. В нашем случае начинаем с ребра весом 13 — наименьшего в графе. На рисунках 4.37–4.42 дана последовательность действий. Ребро весом 19 не включается в остов, так как оно образует цикл с ребрами весом 14 и 13.

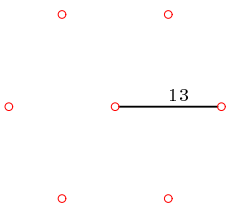


Рис. 4.37

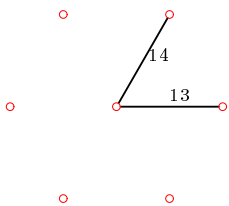


Рис. 4.38

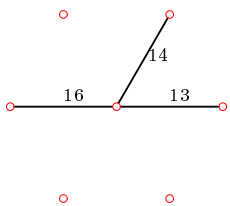


Рис. 4.39

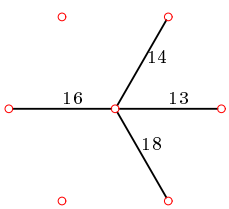


Рис. 4.40

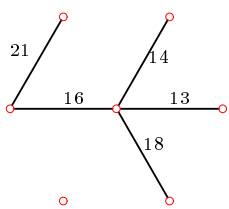


Рис. 4.41

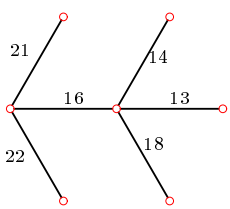


Рис. 4.42

Алгоритм ближайшего соседа построения остова минимального веса. Алгоритм Дж. Краскала требует на каждом шаге проверки на цикличность и предварительной сортировки ребер по весам, что затруднительно для графов с большим числом ребер. Несколько проще следующий алгоритм [13].

1. Отмечаем произвольную вершину графа, с которой начнется построение. Строим ребро наименьшего веса, инцидентное этой вершине.

2. Ищем ребро минимального веса, инцидентное одной из двух полученных вершин. В множество поиска не входит построенное ребро.

3. Продолжаем далее, разыскивая каждый раз ребро наименьшего веса, инцидентное построенным вершинам, не включая в круг поиска все ребра, их соединяющие.

В нашем примере начнем с вершины A. На рисунках 4.43–4.48 дана последовательность действий.

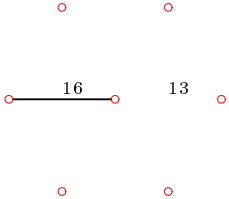


Рис. 4.43

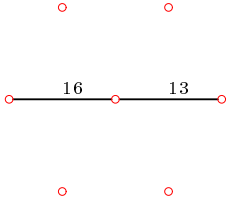


Рис. 4.44

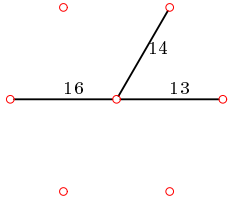


Рис. 4.45

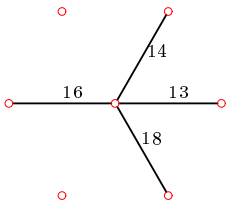


Рис. 4.46

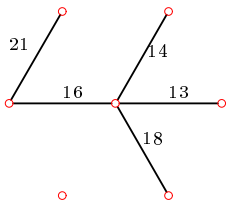


Рис. 4.47

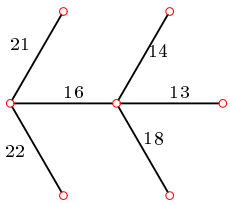


Рис. 4.48

Программа для Maple, использующая встроенные функции, дана на с. 145, программа с визуализацией процесса последовательного построения остова — на с. 147.

4.7. Гамильтоновы циклы

Простой цикл, проходящий через все вершины графа, называется *гамильтоновым* [1]. Простая цепь, проходящая через все вершины графа, называется *гамильтоновой*.

Задача нахождения гамильтоновых циклов получила свое развитие в связи с рядом практических задач. Одной из них является так называемая *задача коммивояжера*, в которой определяется кратчайший гамильтонов цикл (см. с. 88).

В отличие от поиска эйлеровых циклов, проходящих через каждое ребро графа по одному разу, для которых еще Эйлером получено необходимое и достаточное условие существования цикла, для гамильтоновых циклов такого условия не найдено. Существуют, однако, достаточные условия существования гамильтоновых циклов.

Приведем несколько таких признаков [2].

[1] В 1859 г. Уильям Гамильтон (Hamilton W.) предложил математическую игру-головоломку, связанную с обходом вершин додекаэдра, положив тем самым начало одному из самых известных направлений теории графов.

Граф, обладающий гамильтоновым циклом, будем называть *гамильтоновым*.

Теорема 19 (Дирака [1]). *Граф гамильтонов, если степень любой его вершины удовлетворяет неравенству* $\deg(v) \geqslant n/2$.

Теорема 20 (Оре [2]). *Граф гамильтонов, если степени любых двух его несмежных вершин v и u удовлетворяют неравенству* $\deg v + \deg u \geqslant n$.

Для орграфов также имеется достаточное условие.

Теорема 21 (Гуйя-Ури [3]). *Орсвязный граф обладает гамильтоновым циклом* [4], *если для любой его вершины v* $\deg^{\text{in}}(v) \geqslant n/2$, $\deg^{\text{out}}(v) \geqslant n/2$.

Напомним, что орграф называется *орсвязным*, если любая его вершина достижима из любой вершины.

Орграф называется *полугамильтоновым*, если он содержит гамильтонову цепь.

Среди взвешенных орграфов выделяют орграфы, удовлетворяющие неравенству треугольника:

$$c(u, v) \leqslant c(u, w) + c(w, v). \tag{4.3}$$

Очевидно, этому условию удовлетворяют *евклидовы графы* — графы, вершины которых являются точками плоскости, а веса — евклидовыми расстояниями между ними.

Задача. Найти все гамильтоновы циклы графа (рис. 4.49).

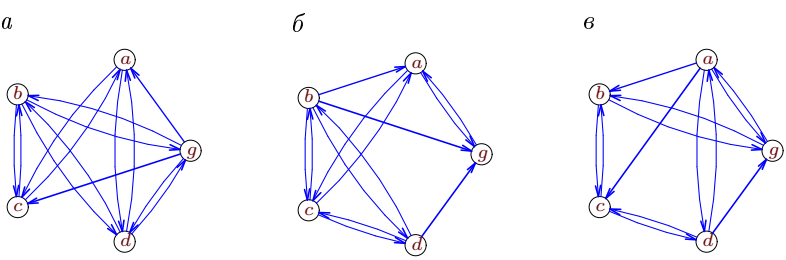


Рис. 4.49

[1] Dirac G.A.

[2] Ore O. [25]

[3] Ghouila-Houri A. [2].

[4] Цикл в орграфе, согласно определению на с. 29, правильнее называть контуром, однако для гамильтоновых циклов, отдавая дань традиции, можно сделать исключение.

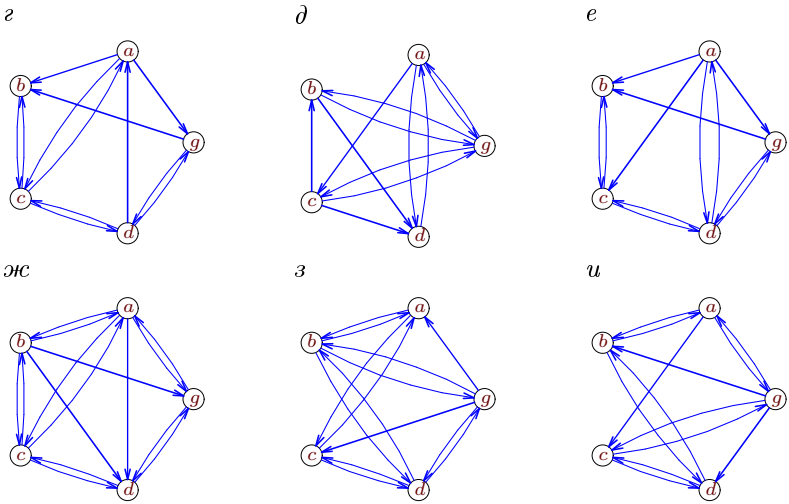


Рис. 4.49 (продолжение)

Ответы

	Гамильтоновы циклы
а	*acbdg*, *adbgc*, *adgbc*, *acbgd*
б	*acdbg*, *acbdg*
в	*agbcd*, *adcbg*, *abcdg*
г	*agbcd*
д	*agcbd*, *acgbd*
е	*agbcd*
ж	*acbdg*, *abgdc*, *agdcb*, *abcdg*, *adcbg*
з	*abdgc*, *acdbg*, *acdgb*, *abgdc*
и	*agcdb*, *abdcg*, *acgdb*

Пример. Дан граф (рис. 4.50). Найти все гамильтоновы циклы графа.

Решение. В [17] описан алгебраический алгоритм [1] нахождения всех гамильтоновых циклов, основанный на теореме 5 (см. с. 11). Эта теорема позволяет найти число маршрутов определенной длины по элементам степени матрицы смежности A графа. Однако эти маршруты остаются безымянными, теорема не дает информации о вершинах в маршрутах. Если немного изменить матрицу смежности и малоинформативные единицы в ней заме-

[1] Yau S.S. (1967 г.), Danielson G.H. (1968 г.), Dhawan V. (1969 г.).

нить на имена вершин, то возведение такой матрицы в степень даст больше информации о маршрутах. Введем таким образом модифицированную матрицу смежности B по правилу: элемент

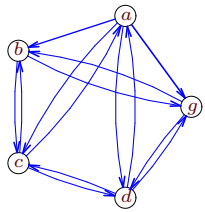


Рис. 4.50

матрицы $\beta_{ij} = v_j$, если существует путь из вершины v_i в вершину v_j.

Для заданного графа получим матрицу смежности A и матрицу B:

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & b & c & d & g \\ 0 & 0 & c & 0 & g \\ a & b & 0 & d & 0 \\ a & 0 & c & 0 & g \\ 0 & b & 0 & d & 0 \end{bmatrix}.$$

Далее последовательно находим вспомогательные матрицы P'_k: и P_k

$$P'_{k+1} = B \cdot P_k, \ P_1 = A, \ P_k = \Phi(P'_k).$$

Матрицы B и P_k умножаются по обычному правилу умножения матриц, а под произведением вершин понимается некоммутативная бинарная операция склеивания, заданная на множестве слов. Слово — упорядоченная последовательность вершин (букв). Произведение слова $u_1u_2, ..., u_k$ и слова $v_1v_2, ..., v_m$ есть слово $u_1u_2, ..., u_kv_1v_2, ..., v_m$.

Матрица P'_k преобразуется в матрицу P_k специальным оператором Ф. Оператор Ф, действующий на элементы p_{ij} матрицы, вычеркивает (обнуляет) те ее элементы, в которых содержится вершина v_i или v_j. Такие элементы содержат контуры, замыкающиеся на v_i или v_j. Так как на главной диагонали P_k содержится информация о циклах, для упрощения и ускорения расчетов можно обнулять главную диагональ всех матриц P_k, кроме, разумеется, последней — P_n, поскольку ее диагональ и содержит искомые циклы без начальной и конечной вершины. Эти вершины легко добавить после окончания основной итеративной процедуры.

Для заданного графа получим последовательность матриц P_k (вспомогательные матрицы P'_k не выписаны). Имеем

$$P_2 = \begin{bmatrix} 0 & c+g & b+d & c+g & b+d \\ c & 0 & 0 & c+g & 0 \\ d & a & 0 & a & a+b+d \\ c & a+c+g & a & 0 & a \\ d & 0 & b+d & 0 & 0 \end{bmatrix}.$$

Заметим, что матрица P_2 получена из P'_2 действием оператора Ф: в первом столбце и первой строке нет слов с вершиной a, во втором столбце и второй строке отсутствует вершина b и т.д. Далее

$$P_3 = \begin{bmatrix} 0 & dc+dg & gb+gd & bc+bg & cb+cd \\ cd+gd & 0 & gd & ca & ca+cd \\ 0 & ag+da+dg & 0 & ag+bg & ab+ad+da \\ 0 & ac+ag+ca & ab+gb & 0 & ab+ca+cb \\ bc+dc & da+dc & da & bc & 0 \end{bmatrix},$$

$$P_4 = \begin{bmatrix} 0 & cdg+gdc & bgd+dgb & cbg+gbc & bcd+dcb \\ gdc & 0 & gda & cag & cad+cda \\ bgd & adg+dag & 0 & abg & dab \\ gbc & cag & agb & 0 & acb+cab \\ bcd & dac+dca & dab & bca & 0 \end{bmatrix}.$$

Последняя матрица (P_5) — диагональная. Все ее элементы — нулевые, кроме $P_{5,11} = bgdc + cbgd + dgbc + gbcd$, $P_{5,22} = cadg + cdag + gdac + gdca$, $P_{5,33} = abgd + adgb + bgda + dagb$, $P_{5,44} = acbg + agbc + cabg + gbca$, $P_{5,55} = bcad + bcda + dacb + dcab$. К полученным слагаемым в элементе матрицы P_{kk} добавим в начале или в конце по вершине v_k, где $v_1 = a$, $v_2 = b$, $v_3 = c$, $v_4 = d$, $v_5 = g$, и выполним круговую перестановку так, чтобы вершина a оказалась первой. Многие гамильтоновы циклы повторяются. Множество ответов для данной задачи будет следующим:

$$abgdc,\ adgbc,\ acbgd,\ agbcd.$$

Полный граф порядка n имеет $(n-1)!$ гамильтоновых циклов.

Очевидно, вручную этот алгоритм не реализуется. Вариант программы для Maple, работающей по описанному алгоритму, дан на с. 150.

4.8. Задача коммивояжера

Это образное название устойчиво закрепилось за одной из самых интересных, практически значимых и одновременно сложных задач теории графов. Задача, берущая свое начало из работ Гамильтона, состоит в определении кратчайшего гамильтонова цикла в графе. Ее

решение связано [17] с решением задачи о назначениях (см. с. 74) и с задачей об остове наименьшего веса (см. с. 78).

Задача. Найти кратчайший гамильтонов цикл графа (рис. 4.51).

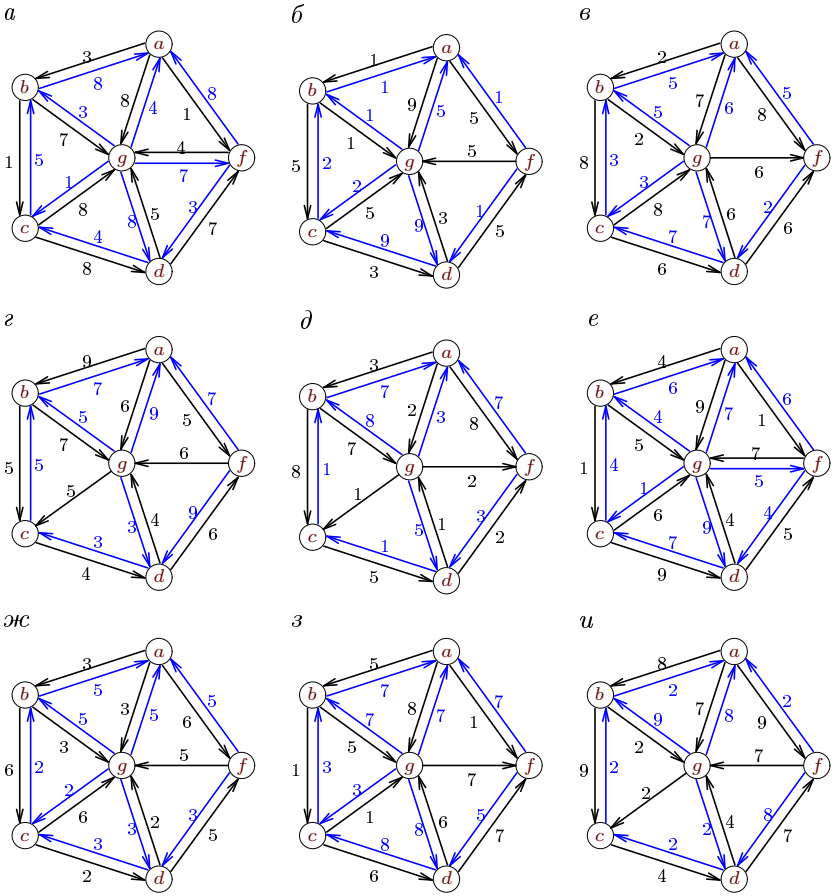


Рис. 4.51

Ответы

	L	Цикл		L	Цикл		L	Цикл
а	23	$afdgcb$	г	29	$afgdcb$	ж	20	$afdgcb$
б	13	$abgcdf$	д	16	$agfdcb$	з	25	$afdgcb$
в	24	$abgcdf$	е	20	$afdgcb$	и	24	$afgdcb$

Пример. Дан взвешенный орграф (рис. 4.52). Найти кратчайший гамильтонов цикл.

Решение. Способ 1. Рассмотрим алгоритм ближайшего соседа (Nva — Nearest vertex add) [1]. Начнем движение по графу из произвольной вершины, например из вершины a. Выбирая из двух возможных дуг, $a \to f$ и $a \to b$, короткую дугу, $a \to b$, длиной 1, далее до вершины c двигаемся без разветвлений: $a \to \to f \to d \to c$. Из c направляемся по более короткой дороге в g, затем в b и возвращаемся в a. Суммарная длина цикла равна $1 + 3 + 8 + 1 + + 3 + 3 = 19$. Алгоритм ближайшего соседа не гарантирует оптимального решения. Решение зависит от выбора начальной вершины и от выбора направления движения при наличии двух и более направлений одинакового веса. Проверим другую вершину в качестве начальной.

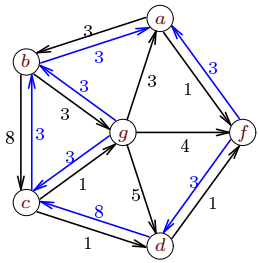


Рис. 4.52

Если начинать движение из b в сторону a по дуге весом 3, то цикл $b \to a \to f \to d \to c \to g \to b$ будет совпадать с найденным, отличаясь только началом отсчета. При выборе направления $b \to g$, также весом 3, получим цикл $b \to g \to c \to d \to f \to a \to b$, имеющий вес $3+3+1+ + 1 + 3 + 3 = 14$. Этот цикл, наименьший по весу, можно принять за окончательный ответ. В действительности, суммарный вес 14 является наименьшим для этой задачи.

Известна оценка алгоритма ближайшего соседа [2] для графа, веса которого удовлетворяют неравенству треугольника (см. с. 85):

$$c_{\mathrm{Nva}} \leqslant 0,5([\ln n] + 1)c_{\mathrm{opt}},$$

где c_{opt} — вес оптимального маршрута. В нашем случае коэффициент в этой оценке $0,5([\ln n] + 1) = 1$.

Соответствующая программа для Maple дана на с. 152.

Способ 2. Рассмотрим алгоритм ближайшей вставки (Nvi — Nearest insert) [2]. Построение цикла начнем с любой вершины, например c. Присоединим к c вершину, образующую вместе с ней цикл. Из трех возможных вариантов, $c \to d \to c$, $c \to g \to c$ и $c \to b \to c$, возьмем самый короткий, $c \to g \to c$, длиной 9 (рис. 4.53). Конечно, можно выбрать и вариант $c \to b \to c$ такой же длины. На следующем шаге можно добавить либо вершину d, вставляя дуги $g \to d$ и $d \to c$ вместо

[1] Аналогичный по стратегии алгоритм использовался для определения остова наименьшего веса (см. с. 83).

дуги $g \to c$, либо b уже с двумя вариантами циклов: $g \to c \to b \to g$ и $g \to b \to c \to g$. Выбираем цикл наименьшей длины: $g \to b \to c \to g$ (рис. 4.54). Аналогично последовательно вставляем в цикл вершины d, f, a (рисунки 4.55–4.57). В результате получаем цикл длиной $3 + 3 + 1 + 3 + 8 + 3 = 21$.

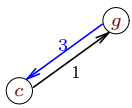


Рис. 4.53

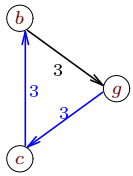


Рис. 4.54

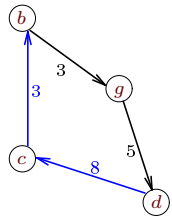


Рис. 4.55

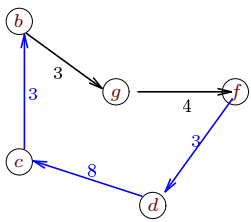


Рис. 4.56

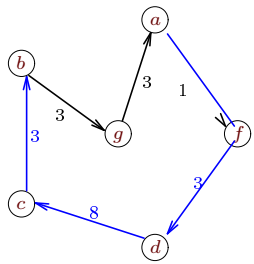


Рис. 4.57

Результат далек от оптимальной длины 14. В рамках этого алгоритма можно добиться меньшей длины, меняя начальную вершину и направления в точках ветвления решения.

Известна оценка алгоритма ближайшей вставки [2], для графа, веса которого удовлетворяют неравенству треугольника (см. с. 85):

$$c_{\mathrm{Nvi}} \leqslant 2c_{\mathrm{opt}},$$

где c_{opt} — вес оптимального маршрута.

Способ 3. Одним из методов искусственного интеллекта является муравьиный алгоритм Марко Дориго [1]. Основная идея алгоритма подсмотрена в природе и имитирует движение колонии муравьев. По форме этот алгоритм похож на жадный Nva (способ 1) и в некоторой степени является его обобщением. Если в алгоритме ближайшего соседа выбор дальнейшего пути производится, исходя из минимального расстояния до очередной вершины, то здесь выбором управляет случайная функция, направляющая движение от текущего положения

[1] Marco Dorigo [33].

с большей вероятностью в вершину j, в которой наибольшее значение некоторой функции $P_{ij,k}$ (где i — номер вершины, в которой производится выбор, k — номер муравья, движущегося по дугам графа). Как и в Nva, во время движения создается список пройденных вершин, что позволяет избежать преждевременного зацикливания. Приведем вид функции, управляющей переходом из данной вершины i в вершину j:

$$P_{ij,k} = \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum\limits_{m}\tau_{im}^{\alpha}\eta_{im}^{\beta}}, \tag{4.4}$$

где τ_{ij} — количество феромона (pheromon), оставленного муравьями на дуге $[i,j]$; η_{ij} — величина, обратная весу (длине) дуги $[i,j]$; α, β — эмпирические коэффициенты. Функция $P_{ij,k}$ подсказывает муравью номер вершины j, в которую он должен направиться. В знаменателе (4.4) стоит нормирующий коэффициент, такой, что $0 \leqslant \leqslant P_{ij,k} \leqslant 1$. Индекс m в сумме пробегает по всем непройденным вершинам, смежным с i. В реальности муравей оставляет след (феромон) во время прохождения пути, и чем чаще он возвращается в исходную точку (а это возможно, если он выбирает оптимальные пути), тем четче след. В математической же модели функция τ_{ij} увеличивается только по завершении маршрута на величину, обратно пропорциональную длине маршрута. При $\alpha = 0$ алгоритм совпадает с Nva — муравей руководствуется только длиной пути. При $\beta = 0$ основой для выбора пути является только опыт (количество феромона, или «глубина следа») предыдущих муравьев-исследователей. Важно отметить еще одно отличие от алгоритма Nva. Выбор пути производится не по максимуму функции $P_{ij,k}$, а случайным образом, но на случай, конечно, влияет значение $P_{ij,k}$. Поясним это на примере. Пусть муравей k подошел к некоторой вершине 8 и обнаружил, что перед ним 7 возможных путей к семи вершинам (на уже пройденные он внимания не обращает). Куда идти? Муравей доверяется случаю. Он «пускает рулетку» (рис. 4.58). В какой

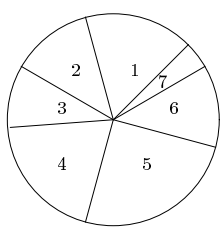


Рис. 4.58

сектор «шарик» закатится, туда и идти. Однако рулетка, размеченная функцией $P_{8j,k}$, $j = 1, ..., 7$, имеет неравные сектора. Чем ближе вершина и чем глубже туда след, тем больше сектор. Таким образом, муравей использует и опыт предшественников (τ_{ij}), и здравый смысл (η_{ij}), и случайный фактор, т.е. все как в жизни.

Для того чтобы не пропустить оптимальное решение, в муравьином алгоритме предусмотрено «испарение» следа. Это достигается введением коэффициента p в итеративной формуле $\tau_{ij} = (1 - p)\tau_{ij}$, применяющейся после каждого цикла обхода графа.

В алгоритме действует целая колония муравьев. Математически это означает, что в каждом цикле обхода движение производится из разных вершин независимым образом.

Здесь приведен самый простой вариант алгоритма. Алгоритм может быть улучшен. Для ускорения сходимости иногда вводят так называемых *элитных* муравьев [33]. В [7] приведены наилучшие комбинации параметров α и β.

Соответствующая программа для Maple дана на с. 154.

Способ 4. Алгоритм отжига. Этот алгоритм, как и муравьиный алгоритм, относится к вероятностным методам решения. Ключевым моментом в таких подходах является случайный выбор одного из нескольких возможных решений вместо анализа каждого. Это позволяет сократить время счета, а время счета в некоторых задачах на графах имеет принципиальное значение. Простой перебор для задач, сложность которых (время счета) растет по показательному или факториальному закону в зависимости от порядка графа, может даже для небольших графов оказаться недопустимо длительным. Оценка этой характеристики в каждом приложении своя. Например, для мобильного робота [1], выполняющего маневр в поиске оптимального решения, бортовой компьютер должен решать задачу коммивояжера за доли секунды. Прямой же перебор вариантов при пяти-десяти пунктах назначения возможен только за несколько минут, что в данном случае никак не-приемлемо. При оптимизации проектируемых в лабораторных условиях тепловых, газовых, электрических или транспортных сетей, как правило, с сотней или более вершин (узлов), проект может затянуться даже на многие годы. Именно поэтому актуальны вероятностные подходы, дающие, может быть, не самые точные, но вполне допустимые решения. Рассмотрим метод отжига, обычно изучаемый в курсах теории искусственного интеллекта.

Образное название лишь отчасти передает содержание и связывается с процессом образования структуры металла при нагревании и дальнейшем охлаждении. Известно, что только контролируемое

[1] Мобильные роботы и мехатронные системы: Материалы научной школы – конференции. — М.: Изд-во Моск. ун-та, 2000.

охлаждение приводит к желаемой структуре металла. При большей температуре степень свободы частиц (в нашем случае — количество вариантов решения) больше, при меньшей — меньше. Если дать возможность алгоритму случайно выбирать решение, оптимальное на каждом шаге (жадный алгоритм), то можно пропустить ход, не лучший локально, но дающий в результате более оптимальное решение.

В методе отжига очередной порядок следования по маршруту между городами выбирается случайно, небольшим изменением предыдущего решения, предположительно оптимального. Самый простой вариант изменения — перестановка двух случайно выбранных городов в маршруте следования. Если полученный маршрут лучше всех существовавших ранее, то этот маршрут берется за очередной [1]. Если маршрут хуже, то самый простой вариант — это не брать его (зачем ухудшать решение?). Однако так решение может закатиться в один из локальных минимумов, которыми изобилуют подобные задачи (рис. 4.59). Именно

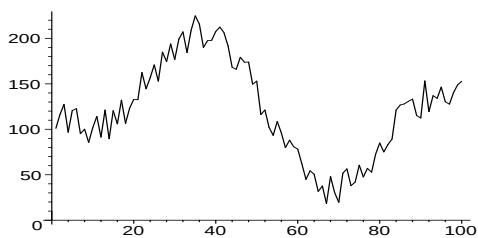


Рис. 4.59

поэтому плохому решению надо дать шанс. Шанс этот (или, правильнее, вероятность) рассчитывается по формуле

$$P = \exp(-\Delta L/T),$$

где ΔL — положительная разность между качеством тестируемого и ранее полученного оптимального решений, T — некоторый постоянно уменьшающийся параметр (условно — температура). В случае задачи коммивояжера качество оценивается длиной маршрута. Очевидно, что чем меньше текущее решение портит минимум и чем больше температура, тем больше вероятность принятия «пробного» выхода из локального минимума.

Снижение температуры обычно производится по формуле $T_{k+1} = \alpha T_k$, где $0 < \alpha < 1$.

[1] Иногда полученный вариант сравнивается с предыдущим.

Соответствующая программа для Maple дана на с. 159.