

Федеральное агентство связи

Государственное образовательное учреждение  
высшего профессионального образования

"Сибирский государственный университет  
телекоммуникаций и информатики"

# ЭЛЕМЕНТЫ ТЕОРИИ ГРАФОВ

УЧЕБНОЕ ПОСОБИЕ

под редакцией В.И. Носова

НОВОСИБИРСК, 2008

УДК 519.1

В.И. Носов, Т.В. Бернштейн, Н.В. Носкова, Т.В. Храмова. **Элементы теории графов.** Учебное пособие. — Новосибирск, 2008. — 107с.

Пособие предназначено для использования в учебном процессе преподавателями и студентами технических специальностей СибГУТИ. В данном учебном пособии изложен основной теоретический материал, необходимый для изучения дискретной математики, а именно, входящего в нее раздела "Теория графов".

Рецензенты: д.т.н. Хрусталеv В.А.,  
к.ф.-м.н. Мельников Л.С.

Ил.90, список лит. — 12 наименований

Для направлений 210400 — телекоммуникации, 230000 — ИВТ.

Утверждено редакционно-издательским советом СибГУТИ в качестве учебного пособия.

© Сибирский государственный университет  
телекоммуникаций и информатики, 2008

# Оглавление

Предисловие . . . . .	4
1. Основные определения . . . . .	5
2. Способы задания графа . . . . .	9
3. Операции на графах . . . . .	12
4. Изоморфизм графов . . . . .	19
5. Представление сетей радиосвязи графами . . . . .	24
6. Связность . . . . .	34
6.1. Алгоритм выделения компонент сильной связности . . .	36
7. Деревья . . . . .	40
8. Обходы . . . . .	42
8.1. Поиск в глубину . . . . .	42
8.2. Поиск в ширину . . . . .	44
8.3. Эйлеров цикл . . . . .	47
9. Кратчайшие остовы в нагруженном графе . . . . .	53
9.1. Алгоритм Краскала построения остова минимального веса (жадный алгоритм) . . . . .	54
9.2. Алгоритм Прима построения остова минимального веса (алгоритм ближайшего соседа) . . . . .	55
10. Кратчайшие пути в нагруженном графе . . . . .	61
10.1. Алгоритм Дейкстры поиска кратчайшего пути в нагру- женном графе . . . . .	61
10.2. Алгоритм Форда-Беллмана поиска кратчайших путей между всеми парами вершин в нагруженном графе .	68
11. Паросочетания . . . . .	78
11.1. Алгоритм построения наибольшего паросочетания в дву- дольном графе . . . . .	79
11.2. Алгоритм построения совершенного паросочетания ми- нимального веса в двудольном нагруженном графе .	82
12. Раскраска графа . . . . .	95
12.1. Жадный алгоритм раскрашивания . . . . .	96
12.2. Алгоритм последовательного раскрашивания . . . . .	100
Список литературы . . . . .	106

# Предисловие

Первые задачи теории графов были связаны с решением математических развлекательных задач и головоломок. Приведем наиболее известные из них:

— *Задача Эйлера о кёнигсбергских мостах*. На реке, протекающей через город Кёнигсберг и омывающей два острова, имеется семь мостов. Может ли пешеход обойти все мосты, пройдя по каждому из них только один раз, и вернуться в исходную точку? Л.Эйлер, рассмотревший эту задачу в 1736г., доказал ее неразрешимость.

— *Задача о четырех красках*. Можно ли любую карту раскрасить четырьмя красками так, чтобы никакие две области, имеющие общий участок границы, не были окрашены в один и тот же цвет?

— *Задача о бродячем торговце (коммивояжере)*. Район, который должен посетить коммивояжер, состоит из  $n$  городов. Известны расстояния между каждыми двумя городами. Торговец должен, выйдя из первого города, обойти все остальные (по одному разу каждый) и вернуться в исходный город так, чтобы пройденное расстояние при этом было минимальным. Задачи подобного рода называются задачами о гамильтоновых циклах.

Данное учебное пособие содержит основные сведения из теории графов: определения, основные теоремы и алгоритмы, а также примеры, иллюстрирующие теоретический материал по отдельным темам. Пособие включает в себя материал по основным разделам теории графов, входящих в курс дискретной математики, а также раздел, посвященный представлению сетей радиосвязи графами. Все разделы снабжены примерами решения типовых задач по изложенным темам. Кроме того, в каждой главе предлагаются упражнения для самостоятельного решения.

В главе, посвященной раскраске графа, сделан акцент на применение теории графов в частотном планировании сетей радиосвязи. Построение передающих сетей радиовещания сводится к частотно - пространственному распределению на заданной территории. Частотное планирование позволяет достичь высокой эффективности использования полос частот, выделенных для развития этих сетей. Решить задачу распределения частот в общем виде позволяют идеализированные сети, в основу построения которых заложены два принципа: геометрически правильная (равномерная) сетка и линейная схема распределения частот. Для реализации этих принципов вводят следующие ограничения: все станции сети имеют одинаковые эффективные излучаемые мощности, эффективные высоты передающих и приемных антенн, поляризацию, условия распространения радиоволн и круговую диаграмму направленности передающих антенн. Задача распределения частот определенных заданным частотным спектром сводится к раскраске вершин данного графа сети связи.

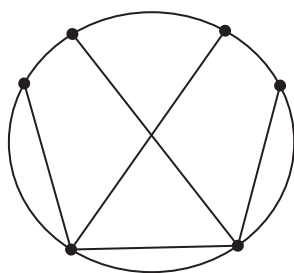
# 1. Основные определения

*Опр.*

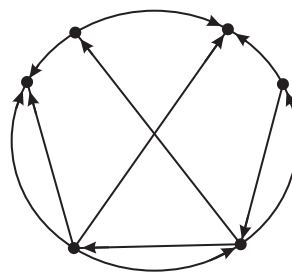
Граф  $G$  — пара  $(V, E)$ , где  $V = \{v_1, v_2, \dots, v_n\}$  — множество *вершин*, а  $E = \{(u, v) | u \in V, v \in V\}$  — множество *ребер* графа. Если множество ребер состоит из неупорядоченных пар (т.е.  $(u, v) = (v, u)$ ), то граф называется *неориентированным*, а если пары  $(u, v)$  упорядоченные (т.е., в общем случае,  $(u, v) \neq (v, u)$ ), то граф называется *ориентированным* или *орграфом*. В таком графе ребра принято называть *дугами*, а граф обозначать  $\vec{G}$ .



**ПРИМЕР 1.1.** На рис. 1 приведены примеры неориентированного и ориентированного графов.



Неориентированный граф



Ориентированный граф

Рис. 1.

*Опр.*

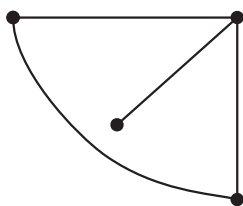
Если множество ребер содержит несколько одинаковых элементов  $(u, v)$ , то такие ребра называются *кратными*. Ребро вида  $(u, u)$ , ведущее из вершины в нее же, называется *петлей*.

*Опр.*

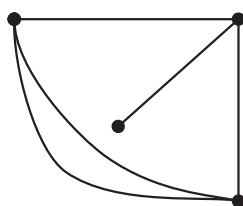
*Простой граф* — граф, множество ребер которого не содержит петель и кратных ребер. *Мультиграф* — граф, множество ребер которого содержит кратные ребра, но не содержит петель. *Псевдограф* — граф, множество ребер которого содержит петли и кратные ребра.



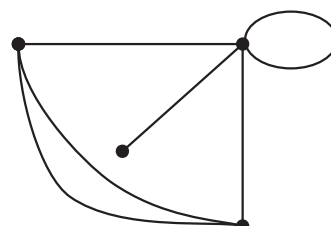
**ПРИМЕР 1.2.** На рис. 2 приведены примеры простого графа, мультиграфа и псевдографа.



Простой граф



Мультиграф



Псевдограф

Рис. 2.

*Опр.*

Вершины  $u$  и  $v$  называются *смежными*, если они соединены ребром, т.е. существует  $(u, v) \in E$ . При этом говорят, что вершина  $u$  (или  $v$ ) и ребро  $(u, v)$  *инцидентны*. Если граф ориентированный, то вершину  $u$  называют *началом (исходом)*, а  $v$  — *концом (заходом)* дуги  $(u, v)$ .

*Опр.*

*Инцидентные* ребра — ребра, инцидентные одной и той же вершине.

*Опр.*

*Окрестность* вершины  $u$  в неориентированном графе — множество смежных с ней вершин:  $\Gamma(u) = \{v \mid (u, v) \in E\}$ . Соответственно *окрестность* множества  $A \subset V$  — множество  $\left( \bigcup_{u \in A} \Gamma(u) \right) \setminus A$ .

*Опр.*

*Окрестность из* вершины  $u$  в ориентированном графе — множество вершин, в которые заходят дуги с началом в  $u$ :  $\Gamma_+(u) = \{v \mid (u, v) \in \vec{E}\}$ . *Окрестность из* множества вершин  $A \subset V$  в ориентированном графе состоит из вершин, которые являются концами дуг, ведущих из  $A$ .

*Опр.*

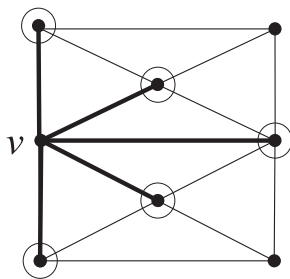
*Окрестность в* вершину  $u$  в ориентированном графе — множество вершин, дуги из которых заходят в  $u$ :

$$\Gamma_-(u) = \{v \mid (v, u) \in \vec{E}\}.$$

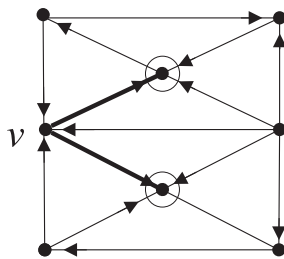
*Окрестность в* множество вершин  $A \subset V$  в ориентированном графе состоит из вершин, которые являются началами дуг, ведущих в множество  $A$ .



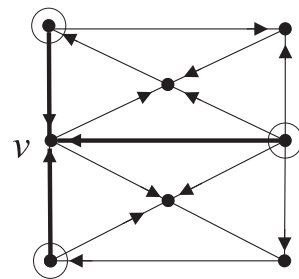
**ПРИМЕР 1.3.** На рис. 3 изображены неориентированный и ориентированный графы и на каждом из них отмечена вершина  $v$  и ее окрестность (вершины окрестности обведены).



$\Gamma(v)$



$\Gamma_+(v)$



$\Gamma_-(v)$

Рис. 3.

*Опр.*

*Степень* вершины — число инцидентных ей ребер. Для обозначения степени вершины  $v$  используется запись  $\deg(v)$ . В орграфе различают также *полустепень исхода*  $\deg_+(v)$  и *полустепень захода*  $\deg_-(v)$ , которые равны числу исходящих и заходящих в вершину  $v$  ребер, соответственно.

*Опр.*

*Висячая* вершина — вершина, степень которой равна 1.

*Опр.*

*Изолированная* вершина — вершина, степень которой равна 0.

---

**Теорема "о рукопожатиях".** Сумма степеней вершин графа  $G(V, E)$  равна удвоенному числу ребер, т.е.

$$\sum_{v \in V(G)} \deg(v) = 2|E(G)|.$$

---

**Замечание.** Запись  $|A|$  обозначает мощность множества  $A$ . Если множество конечно, то его мощность равна числу элементов множества.

---

**Следствие.** В любом графе число вершин нечетной степени чётно.

*Опр.*

*Нагруженный граф (орграф)* — граф (орграф) на множестве ребер которого задана неотрицательная функция  $w : E(G) \rightarrow R$ . Функция  $w$  называется *весовой функцией*.

**Замечание.** Нагруженный граф также называют *взвешенным*.

*Опр.*

*Регулярный граф степени  $d$*  — простой граф, все вершины которого имеют степень  $d$ .

*Опр.*

*Пустой граф порядка  $n$*  — граф, состоящий из  $n$  изолированных вершин:

$$O_n(V, E), \quad V = \{v_1, v_2, \dots, v_n\}, \quad E = \emptyset.$$

*Опр.*

*Простой цикл длины  $n$*  — простой граф, состоящий из  $n$  вершин степени 2:

$$C_n(V, E), \quad V = \{v_1, v_2, \dots, v_n\}, \\ E = \{(v_1, v_2), (v_2, v_3), \dots, (v_{n-1}, v_n), (v_n, v_1)\}.$$

*Опр.*

*Паросочетание* — простой граф, состоящий из  $2n$  висячих вершин:

$$nK_2(V, E), \quad V = \{v_1, v_2, \dots, v_n, u_1, u_2, \dots, u_n\}, \quad E = \{(v_i, u_i) \mid 1 \leq i \leq n\}.$$

**Замечание.** В главе 11 будет приведено еще одно определение паросочетания, которое никак не противоречит данному здесь, но является более удобным для использования в дальнейшем.

*Опр.*

*Полный граф порядка  $n$*  — простой граф, состоящий из  $n$  вершин степени  $n - 1$ :

$$K_n(V, E), \quad V = \{v_1, v_2, \dots, v_n\}, \quad E = \{(v_i, v_j) \mid v_i \in V, v_j \in V, i \neq j\}.$$

**Замечание.** Пустой граф, паросочетание, простой цикл и полный граф являются регулярными графами степени 0, 1, 2 и  $(n - 1)$ , соответственно. На рис. 4 изображены регулярные графы степени 0, 1, 2 и 3 с четырьмя вершинами.

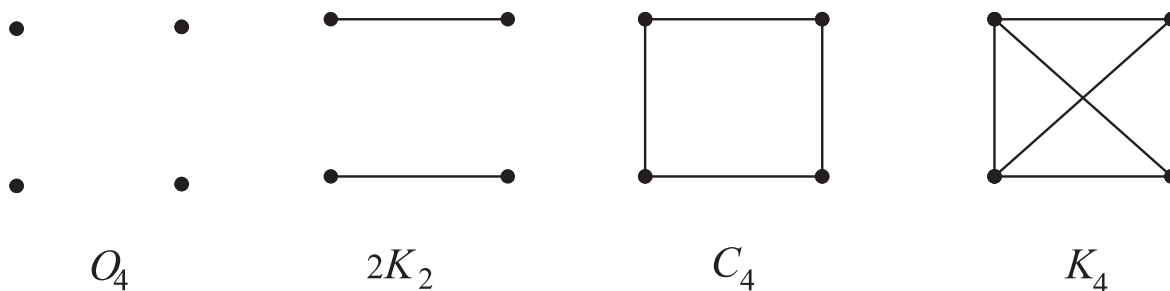


Рис. 4.

## УПРАЖНЕНИЯ

1. Приведите примеры графов:

- а) с пятью вершинами и пятью ребрами;
- б) с пятью вершинами и тремя ребрами;
- в) с тремя вершинами и пятью ребрами.

2. Вычислите, сколько существует различных простых графов с четырьмя вершинами.

3. Нарисуйте все регулярные графы с шестью вершинами. Выполните это же задание для графа с семью вершинами. Почему графов с семью вершинами, удовлетворяющих поставленным условиям, меньше?



## 2. Способы задания графа

Существует множество способов задания графа: рисунок (при этом вершины графа обозначаются точками, а ребра (дуги) — линиями (стрелками)), список вершин и ребер, матрица смежности и матрица инцидентности. Кроме того, для некоторых специальных графов (например, двудольных и нагруженных) существуют специальные способы задания. Способ задания выбирается в зависимости от задачи, которую предстоит решать на данном графе.

*Опр.* Матрица смежности неориентированного графа  $G(V, E)$  — квадратная матрица  $A(G)$  порядка  $n$  ( $n = |V|$ ), элементы которой определяются следующим образом:  $a_{ij}$  равен числу ребер, соединяющих вершины  $v_i$  и  $v_j$  (при этом петли считаем дважды).

*Опр.* Матрица смежности ориентированного графа  $\vec{G}(V, \vec{E})$ , — квадратная матрица  $A(\vec{G})$  порядка  $n$  ( $n = |V|$ ), элементы которой определяются следующим образом:  $a_{ij}$  равен числу дуг, ведущих из вершины  $v_i$  в вершину  $v_j$  (т.е. исходящих из  $v_i$  и заходящих в  $v_j$ ).

**Замечание.** Сумма элементов в строке матрицы смежности орграфа равна полустепени исхода соответствующей вершины, а сумма элементов в столбце — полустепени захода. Матрица смежности неориентированного графа симметрична, сумма элементов в  $i$ -й строке равна сумме элементов в  $i$ -м столбце и, соответственно, степени  $i$ -й вершины.

*Опр.* Матрица инцидентности неориентированного графа  $G(V, E)$ , — матрица  $B(G)$  порядка  $n \times m$  ( $n = |V|$ ,  $m = |E|$ ), элементы которой определяются следующим образом:  $b_{ij} = 1$ , если  $v_i$  и  $e_j$  инцидентны;  $b_{ij} = 0$ , если  $v_i$  и  $e_j$  не инцидентны.

*Опр.* Матрица инцидентности ориентированного графа  $\vec{G}(V, \vec{E})$ , — матрица  $B(\vec{G})$  порядка  $n \times m$  ( $n = |V|$ ,  $m = |E|$ ), элементы которой определяются следующим образом:  $b_{ij} = 1$ , если  $v_i$  — начало дуги  $e_j$ ,  $b_{ij} = -1$ , если  $v_i$  — конец дуги  $e_j$ ;  $b_{ij} = 0$ , если  $v_i$  и  $e_j$  не инцидентны.

**Замечание 1.** Если граф содержит петли, то значение соответствующего элемента  $b_{ij}$  выбирается в зависимости от дальнейшего применения этой матрицы. В нашем случае, будем использовать запись  $b_{ij} = 2$  для неориентированного графа и запись  $b_{ij} = \pm 1$  для орграфа.

**Замечание 2.** Если ребра графа пронумерованы, то  $i$ -й столбец матрицы инцидентности соответствует  $i$ -му ребру. Если ребра графа (орграфа) не помечены, то при составлении матрицы инцидентности будем придержи-

ваться следующего правила: сначала перечисляем ребра (дуги) инцидентные (исходящие из) первой вершины в вершины ее окрестности (в порядке возрастания номеров вершин), затем из второй и т.д. (см. пример 2.1).

**Замечание 3.** Сумма положительных элементов в  $i$ -й строке матрицы инцидентности орграфа равна полустепени исхода  $i$ -й вершины, а сумма отрицательных элементов — полустепени захода. Для неориентированного графа, сумма элементов в  $i$ -й строке равна степени  $i$ -й вершины.

*Опр.*

Матрица весов  $W(G)$  нагруженного графа — квадратная матрица, в которой элемент  $w_{ij}$  равен весу ребра  $(v_i, v_j)$ . Если вершины  $v_i$  и  $v_j$  не смежны, то  $w_{ij} = \infty$ . Значение элемента  $w_{ii}$  выбирается в зависимости от приложений (как правило 0 или  $\infty$ ).

*Опр.*

Матрица весов  $W(\vec{G})$  нагруженного орграфа — квадратная матрица, в которой элемент  $w_{ij}$  равен весу дуги  $(v_i, v_j)$ . Если вершины  $v_i$  и  $v_j$  не смежны, то  $w_{ij} = \infty$ . Значение элемента  $w_{ii}$  выбирается в зависимости от приложений.

**Замечание.** Матрицу весов также называют *матрицей длин дуг*.

✓ **ПРИМЕР 2.1.** Для мультиграфа  $G$ , изображенного на рис. 5 записать матрицу инцидентности  $B(G)$  и матрицу смежности  $A(G)$ .

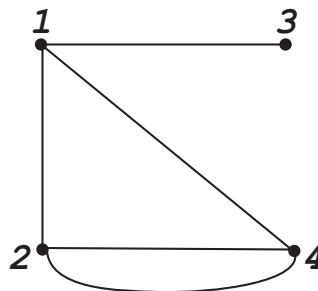


Рис. 5.

**Решение.**

$$B(G) = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix}, \quad A(G) = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 2 \\ 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 \end{pmatrix}.$$

✓ **ПРИМЕР 2.2.** Для заданного на рис. 6 орграфа  $\vec{G}$  записать матрицу инцидентности  $B(\vec{G})$  и матрицу смежности  $A(\vec{G})$ .

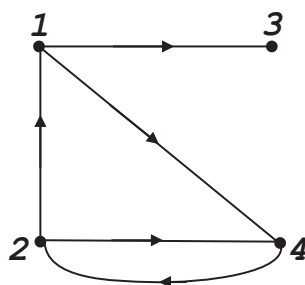


Рис. 6.

**Решение.**

$$B(\vec{G}) = \begin{pmatrix} 1 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & 1 & -1 \\ -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & -1 & 1 \end{pmatrix}, \quad A(\vec{G}) = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

✓ **ПРИМЕР 2.3.** Для заданных на рис. 7 псевдографов записать матрицы смежности и найти степени вершин.

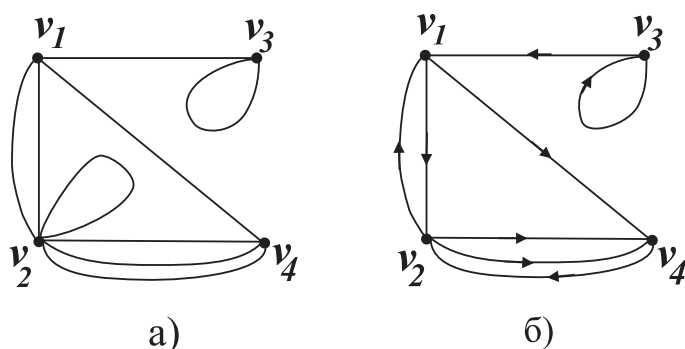


Рис. 7.

**Решение.**

$$\text{а) } A(G) = \begin{pmatrix} 0 & 2 & 1 & 1 \\ 2 & 2 & 0 & 3 \\ 1 & 0 & 2 & 0 \\ 1 & 3 & 0 & 0 \end{pmatrix}; \quad \deg(v_1) = \sum_i a_{1i} = 4, \deg(v_2) = 7, \\ \deg(v_3) = 3, \deg(v_4) = 4.$$

$$\text{б) } A(\vec{G}) = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 2 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}; \quad \deg_+(v_1) = \sum_i a_{1i} = 2, \deg_+(v_2) = 3, \\ \deg_+(v_3) = 2, \deg_+(v_4) = 1; \\ \deg_-(v_1) = \sum_i a_{i1} = 2, \deg_-(v_2) = 2, \\ \deg_-(v_3) = 1, \deg_-(v_4) = 3.$$

## УПРАЖНЕНИЯ

1. Графы, представленные на рис. 8, задать с помощью матриц смежности и инцидентности.

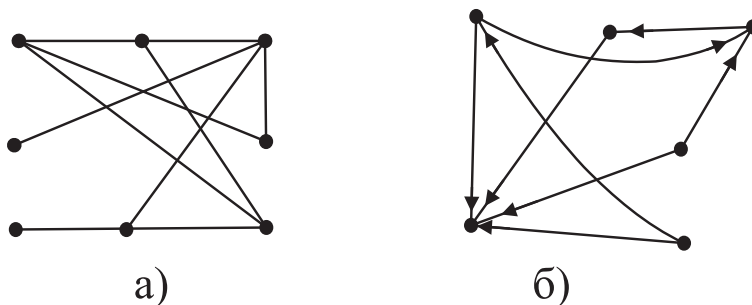


Рис. 8.

2. Нарисовать орграф, заданный матрицей смежности:

$$A(\vec{G}) = \begin{pmatrix} 0 & 1 & 0 & 2 \\ 3 & 0 & 4 & 0 \\ 0 & 5 & 0 & 4 \\ 3 & 0 & 2 & 0 \end{pmatrix}.$$

3. Нарисовать нагруженный орграф, заданный матрицей весов:

$$W(\vec{G}) = \begin{pmatrix} \infty & 1 & \infty & 2 \\ 3 & \infty & 4 & \infty \\ \infty & 5 & \infty & 4 \\ 3 & \infty & 2 & \infty \end{pmatrix}.$$

## 3. Операции на графах

*Опр.*

Подграф графа — граф, все вершины и рёбра которого содержатся среди вершин и рёбер исходного графа. На рис. 9 изображен граф  $G$  и два его подграфа  $G_1$  и  $G_2$ .

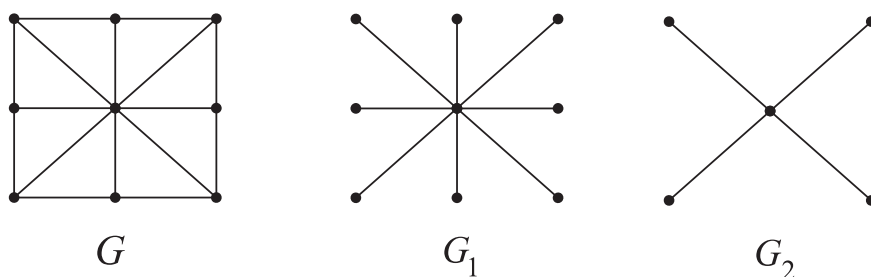


Рис. 9.

1. **Удаление ребра.** При удалении ребра из графа  $G$  получаем новый граф  $G'$ , который является подграфом исходного. Множество вершин графа  $G'$  совпадает с множеством вершин графа  $G$ , а ребер на одно (удаленное) меньше.
2. **Удаление вершины.** Удаление вершины  $v$  из графа  $G$  производится вместе с удалением всех инцидентных ей ребер. В результате, получаем новый граф  $G'$ , который является подграфом исходного. Вершин у графа  $G'$  на одну (удаленную) меньше, чем у графа  $G$ , а ребер меньше на  $\deg v$ .
3. **Добавление вершины.** При добавлении вершины в граф  $G$  получаем новый граф  $G'$ , для которого исходный граф  $G$  является подграфом. Множество ребер графа  $G'$  такое же как у графа  $G$ , а вершин на одну (добавленную) больше. Добавленная вершина является изолированной.
4. **Добавление ребра.** При добавлении ребра в граф  $G$  получаем новый граф  $G'$ , для которого исходный граф  $G$  является подграфом. Множество вершин графа  $G'$  такое же как у графа  $G$ , а ребер на одно (добавленное) больше.

**Замечание.** Операции добавления и удаления применимы как к простым графам, так и к мульти- и псевдографам.

5. **Отождествление вершин.** При отождествлении вершин  $u$  и  $v$  графа  $G$  получаем новый граф  $G'$  следующим образом: удалим вершины  $u$  и  $v$ ; добавим новую вершину  $w$ ; добавим ребра  $(w, v_i)$ , где  $v_i \in \Gamma(u) \cup \Gamma(v)$  (т.е. ребра, соединяющие новую вершину со всеми вершинами, которые были смежными с удаленными вершинами).
6. **Стягивание ребра.** При стягивании ребра  $e$  графа  $G$  получаем новый граф  $G'$ , у которого отождествлены вершины, инцидентные данному ребру, а само ребро исключено из множества ребер.

✓ **ПРИМЕР 3.1.** На рис. 10 изображен граф  $G$  и графы, полученные применением к  $G$  операций удаления и добавления: удаление ребра  $(v_4, v_5)$  (рис. 10,  $G_1$ ); удаление вершины  $v_5$  (рис. 10,  $G_2$ ); добавление вершины  $v_7$  (рис. 10,  $G_3$ ); добавление ребра  $(v_2, v_3)$  (рис. 10,  $G_4$ ).

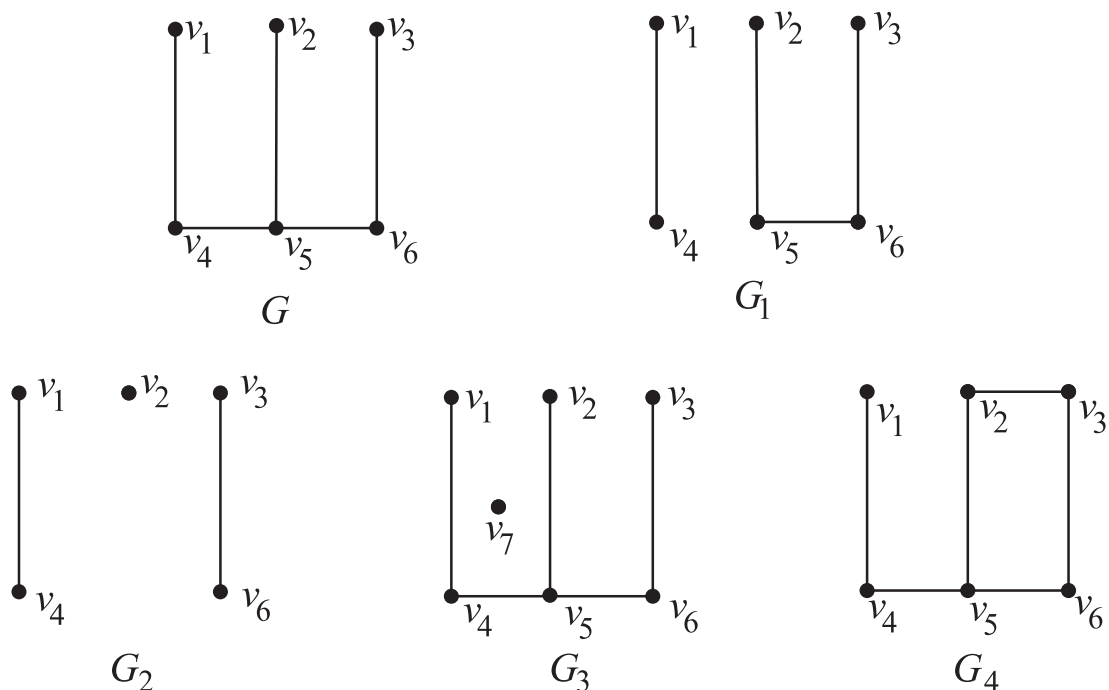


Рис. 10.

На рис. 11 изображены графы, полученные применением к  $G$  операций отождествление вершин  $v_2, v_3$  (рис. 11,  $G_5$ ) и стягивание ребра  $(v_4, v_5)$  (рис. 11,  $G_6$ ).

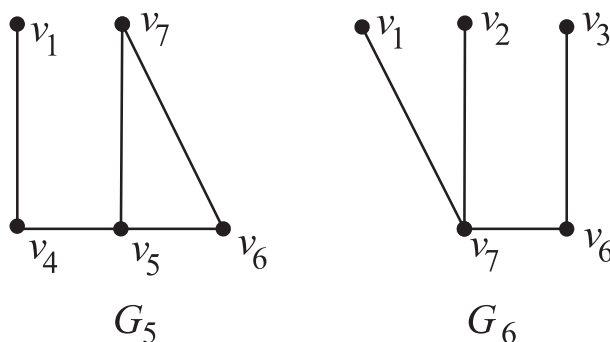


Рис. 11.

**7. Размножение вершины.** При размножении вершины  $u$  графа  $G$  получаем новый граф  $G'$  следующим образом: добавим новую вершину  $u'$ ; добавим новое ребро  $(u, u')$ ; добавим ребра  $(u', v_i)$ ,  $v_i \in \Gamma(u)$  (т.е. ребра, соединяющие новую вершину со всеми вершинами из окрестности вершины  $u$ ).

**8. Расщепление вершины.** При расщеплении вершины  $u$  графа  $G$  получаем новый граф  $G'$  следующим образом: разобьем окрестность вершины  $u$  на два произвольных непересекающихся подмножества  $\Gamma_1$  и  $\Gamma_2$ ; удалим вершину  $u$  из графа; добавим новые вершины  $u_1$  и

$u_2$ ; добавим ребра  $(u_1, v_i)$ ,  $v_i \in \Gamma_1$ ,  $(u_2, w_j)$ ,  $w_j \in \Gamma_2$  и  $(u_1, u_2)$  (т.е. ребра, соединяющие новые вершины со всеми вершинами, которые были смежными с удаленной вершиной и ребро, соединяющее новые вершины между собой).

**9. Дублирование вершины.** При дублировании вершины  $u$  графа  $G$  получаем новый граф  $G'$  следующим образом: добавим вершину  $u'$ ; добавим ребра  $(u', v_i)$ ,  $v_i \in \Gamma(u)$  (т.е. ребра, соединяющие новую вершину со всеми вершинами, которые смежны с вершиной  $u$ ).

**10. Разбиение ребра.** При разбиении ребра  $(u, v)$  графа  $G$  получаем новый граф  $G'$  следующим образом: удалим ребро  $(u, v)$  из множества рёбер графа; добавим новую вершину  $w$ ; добавим рёбра  $(u, w)$  и  $(w, v)$ .

**Замечание.** В некоторых случаях используется обобщенное разбиение ребра — замена его простой цепью.

✓ **ПРИМЕР 3.2.** На рис. 12 продемонстрировано размножение вершины  $v_5$  графа  $G$ , расщепление вершины  $v_7$  графа  $G_7$ , дублирование вершины  $v_7$  графа  $G_7$  (графы  $G_7$ ,  $G_8$  и  $G_9$  соответственно).

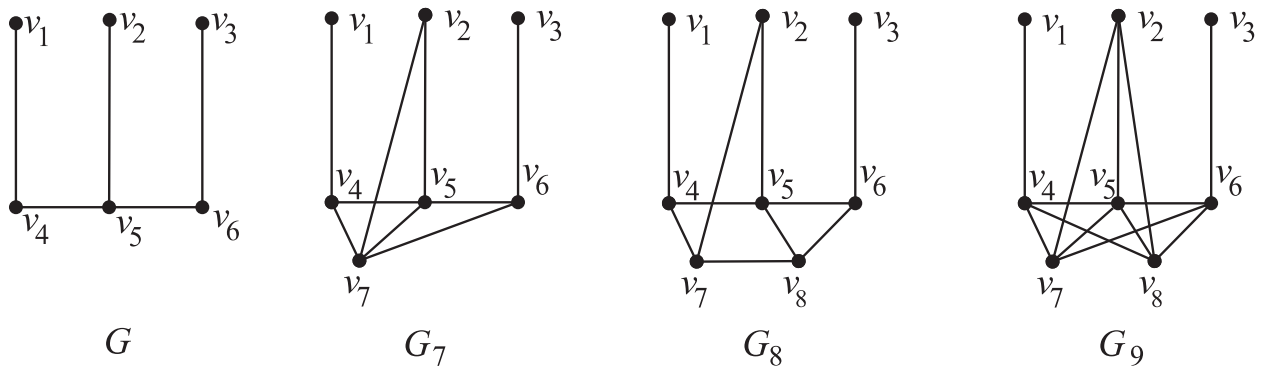


Рис. 12.

**11. Объединение графов.** При объединении графов получаем граф, состоящий из всех вершин и ребер исходных графов:

$$G_1(V_1, E_1) \cup G_2(V_2, E_2) = G(V_1 \cup V_2, E_1 \cup E_2).$$

**Замечание 1.** Если множества вершин и ребер исходных графов не пересекались, то объединение называется *дизъюнктивным*.

**Замечание 2.** Как правило, операция объединения применяется к нескольким (необязательно двум) подграфам одного и того же графа.

**12. Пересечение графов.** При пересечении графов получаем граф, множества вершин и рёбер которого состоят из вершин и рёбер, принадлежащих исходным графам:

$$G_1(V_1, E_1) \cap G_2(V_2, E_2) = G(V_1 \cap V_2, E_1 \cap E_2).$$

**Замечание.** Если множества вершин исходных графов не пересекались, то пересечение графов пусто. Непустым может быть пересечение некоторых подграфов одного и того же графа.

✓ **ПРИМЕР 3.3.** На рис. 13 изображены графы  $G_1$ ,  $G_2$ , их объединение  $G_1 \cup G_2$  и пересечение  $G_1 \cap G_2$ .

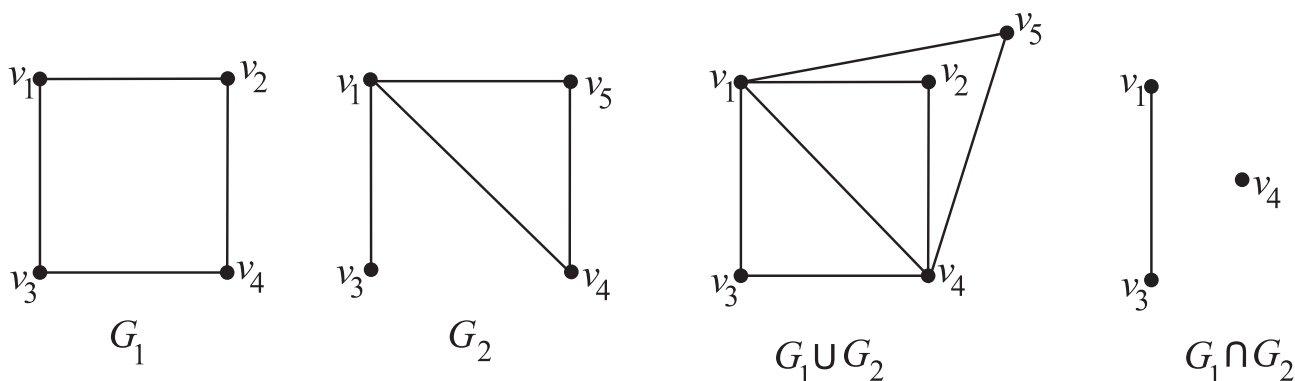


Рис. 13.

**13. Соединение графов.** При соединении графов получаем граф, который является результатом их объединения и добавления рёбер, инцидентных вершинам из разных графов:

$$G_1(V_1, E_1) + G_2(V_2, E_2) = G(V_1 \cup V_2, E_1 \cup E_2 \cup \{(v_1, v_2) \mid v_1 \in V_1, v_2 \in V_2\})$$

(при условии, что  $V_1 \cap V_2 = \emptyset$ ,  $E_1 \cap E_2 = \emptyset$ ).

✓ **ПРИМЕР 3.4.** На рис. 14 изображены графы  $G_1$ ,  $G_2$  и граф  $G_1 + G_2$ , полученный в результате их соединения.

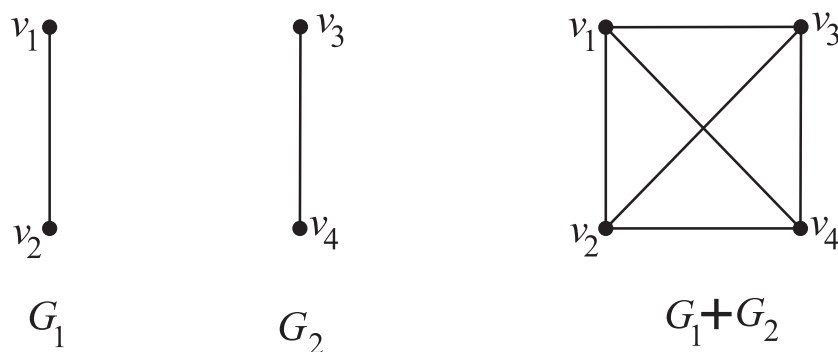


Рис. 14.



14. **Декартово произведение графов.** Множество вершин произведения  $G_1 \times G_2$  графов  $G_1(V_1, E_1)$  и  $G_2(V_2, E_2)$  состоит из упорядоченных пар  $(u_1, u_2)$ ,  $u_1 \in G_1$ ,  $u_2 \in G_2$ :

$$V(G_1 \times G_2) = V(G_1) \times V(G_2).$$

Множество ребер декартова произведения графов определяется следующим образом:  $((u, v_1), (u, v_2))$  ребро в графе  $G_1 \times G_2$ , если  $v_1, v_2$  смежны в графе  $G_2$  и  $((u_1, v), (u_2, v))$  ребро в графе  $G_1 \times G_2$ , если  $u_1, u_2$  смежны в графе  $G_1$ :

$$E(G_1 \times G_2) = \{((u, v_1), (u, v_2)) | u \in V_1, (v_1, v_2) \in E_2\} \cup \{((u_1, v), (u_2, v)) | v \in V_2, (u_1, u_2) \in E_1\}$$

(при условии, что  $V_1 \cap V_2 = \emptyset$ ,  $E_1 \cap E_2 = \emptyset$ ).

**Замечание.** Для того, чтобы изобразить декартово произведение графов  $G_1 \times G_2$  на рисунке, "заменяем" каждую вершину графа  $G_1$  на граф  $G_2$  и "размножим" ребра, инцидентные двум различным копиям  $G_2$  так, чтобы они соединяли идентичные вершины.

✓ **ПРИМЕР 3.5.** Декартово произведение графов используется для задания  $n$ -мерных кубов. На рис. 15 изображен одномерный куб  $Q_1$  (который получен в результате умножения изолированной вершины на себя) и двумерный куб  $Q_2$ , который является декартовым произведением  $Q_1 \times Q_1$ .

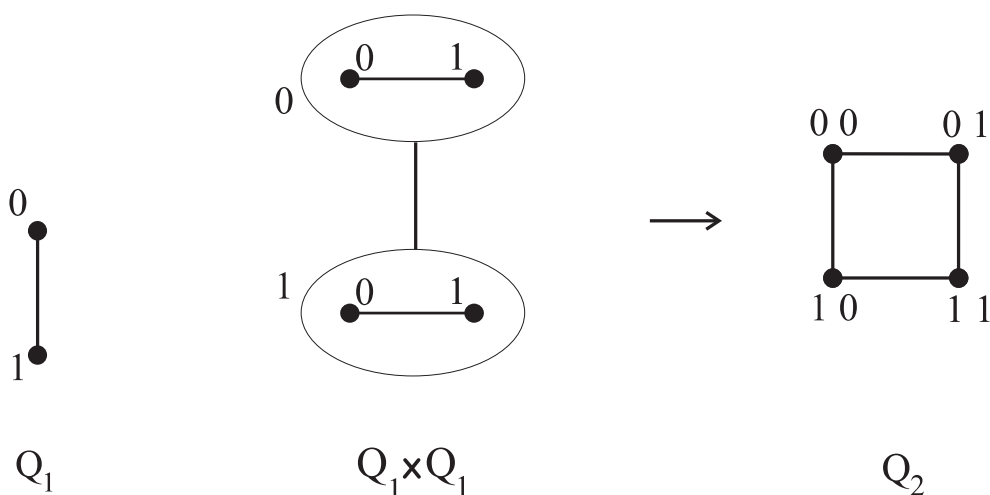


Рис. 15.

Трехмерный куб (рис. 16) может быть получен последовательным умножением:

$$Q_3 = Q_1 \times Q_1 \times Q_1 = Q_1 \times Q_2.$$

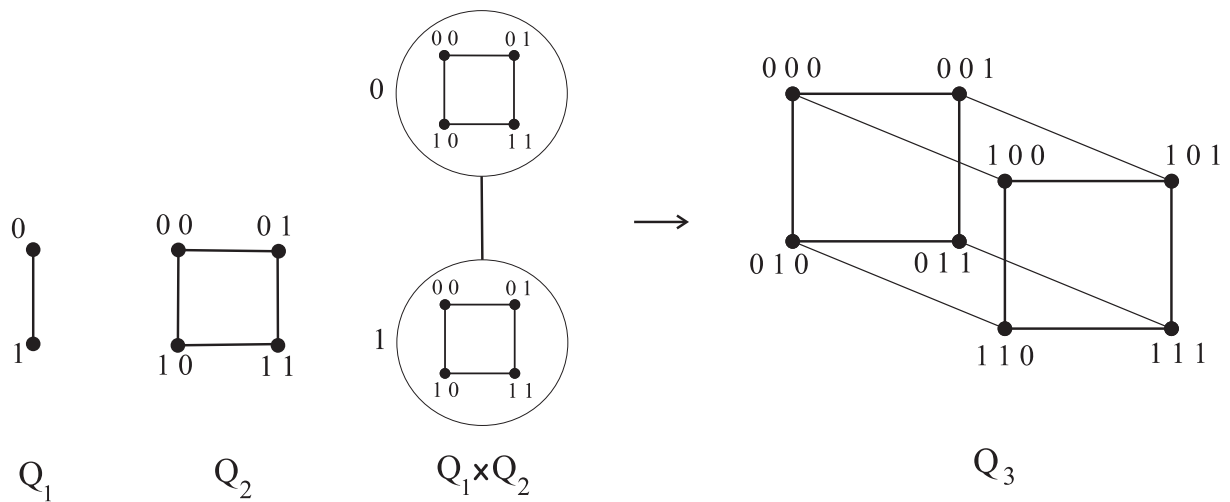


Рис. 16.

Для  $Q_4$ :  $Q_4 = Q_1 \times Q_1 \times Q_1 \times Q_1 = Q_1 \times Q_1 \times Q_2 = Q_1 \times Q_3 = Q_2 \times Q_2$ .

На рис. 17. — изображена проекция четырехмерного куба на плоскость.

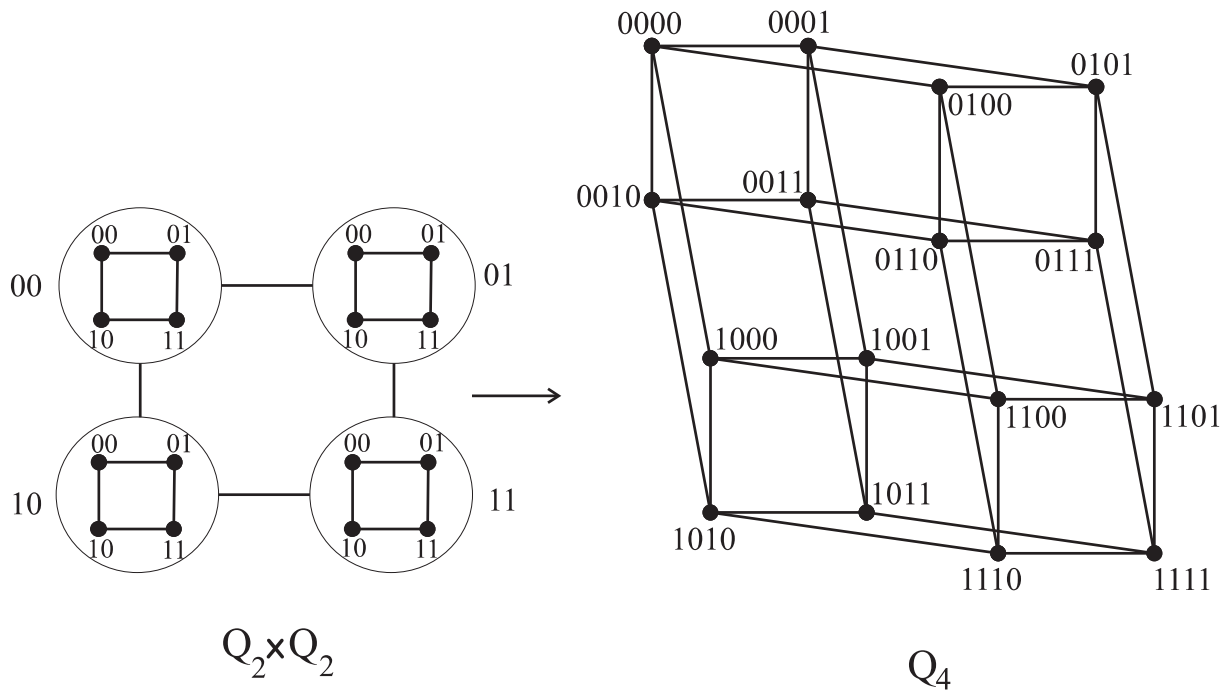


Рис. 17.

**Замечание.** Операции 11—13 применимы к любому конечному числу графов (в том числе мульти- и псевдографов). Операции 10 и 11 применимы к ориентированным графам.

## УПРАЖНЕНИЯ

1. Для полного графа  $K_4$  выполнить операции:

- а) удаление ребра  $(v_1, v_2)$ ;
- б) удаление вершины  $v_1$ ;
- в) добавление вершины  $v_5$  и добавление ребра  $(v_2, v_5)$ ;
- г) отождествление вершин  $v_1, v_2$ ;
- д) стягивание ребра  $(v_1, v_2)$ ;
- е) размножение вершины  $v_1$ ;
- ж) расщепление вершины  $v_1$ ;
- з) разбиение ребра  $(v_1, v_2)$ .

2. Для полных графов  $K_3$  и  $K_2$  выполнить операции соединения и произведения (предполагается, что множества вершин графов не пересекаются).

## 4. Изоморфизм графов

*Опр.*

Изоморфные графы — графы, между множествами вершин которых существует взаимно - однозначное соответствие сохраняющее отношение инцидентности (для орграфов сохраняющее также начало и конец каждой дуги).

**Замечание.** Для того чтобы доказать изоморфность графов достаточно показать, что матрица смежности одного графа получается из матрицы смежности другого перестановкой строк и соответствующих им столбцов. Для изоморфных графов верно следующее:

$$G_1 \simeq G_2 \Rightarrow |V(G_1)| = |V(G_2)|, |E(G_1)| = |E(G_2)|,$$

$$\{\deg(v) | v \in V(G_1)\} = \{\deg(v) | v \in V(G_2)\}.$$

Последнее равенство означает, что у изоморфных графов одинаковые наборы степеней вершин (обратное утверждение не верно).

Приведенный ниже **алгоритм преобразования матрицы смежности** позволяет установить изоморфность (или неизоморфность) графов. Преобразование применяется к матрице смежности каждого из предложенных графов, на каждом шаге преобразованные матрицы сравниваются описанным далее способом.

**Шаг 0.** Упорядочиваем множество вершин каждого из рассматриваемых графов по убыванию степеней:

$V = \{v_1, v_2, \dots, v_n\}$  — множество вершин графа;

$\{n_1, n_2, \dots, n_k\}, (k \leq n)$  — множество степеней вершин графа, причем  $n > n_1 > n_2 > \dots > n_k \geq 0$ ;

$$V = V_1 \cup V_2 \cup \dots \cup V_k, V_i = \{v | \deg(v) = n_i\}.$$

В матрицах смежности переставляем строки и соответствующие им столбцы согласно новому порядку. Матрица смежности при этом разбивается на  $k^2$  "блоков" по вершинам равной степени.

$$A(G) \simeq \begin{pmatrix} V_1 & V_2 & \dots & V_k \\ \hline \star & \star & \dots & \star \\ \hline \star & \star & \dots & \star \\ \hline \dots & \dots & \dots & \dots \\ \hline \star & \star & \dots & \star \end{pmatrix}$$

Для графов, изоморфность которых требуется подтвердить, должны выполняться следующие требования: во-первых, количество блоков должно совпадать; во-вторых, соответствующие блоки в матрицах смежности должны иметь одинаковый размер; в-третьих, соответствующие блоки в матрицах смежности должны содержать равное количество 1. Если хотя бы одно из этих требований не выполняется, то графы не изоморфны.

**Шаг 1.** Упорядочиваем множество вершин, соответствующих каждому блоку, полученному на предыдущем шаге таким образом, чтобы сначала были перечислены вершины не смежные с первой вершиной, а затем — смежные с ней. В матрице смежности переставляем строки и соответствующие им столбцы согласно этому порядку. Каждый блок матрицы смежности при этом разбивается на четыре подблока: первый подблок содержит первую строку из нулей, а второй — из единиц.

$$A(G) \simeq \begin{pmatrix} V_1^0 & V_1^1 & V_2^0 & V_2^1 & \dots & \dots & V_k^0 & V_k^1 \\ \hline 00\dots 0 & 11\dots 1 & 00\dots 0 & 11\dots 1 & \dots & \dots & 00\dots 0 & 11\dots 1 \\ 0\star\dots\star & \star\dots\star & \star\dots\star & \star\dots\star & \dots & \dots & \star\dots\star & \star\dots\star \\ 0\star\dots\star & \star\dots\star & \star\dots\star & \star\dots\star & \dots & \dots & \star\dots\star & \star\dots\star \\ \hline 1\star\dots\star & \star\dots\star & \star\dots\star & \star\dots\star & \dots & \dots & \star\dots\star & \star\dots\star \\ 1\star\dots\star & \star\dots\star & \star\dots\star & \star\dots\star & \dots & \dots & \star\dots\star & \star\dots\star \\ \hline \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \hline 0\star\dots\star & \star\dots\star & \star\dots\star & \star\dots\star & \dots & \dots & \star\dots\star & \star\dots\star \\ 0\star\dots\star & \star\dots\star & \star\dots\star & \star\dots\star & \dots & \dots & \star\dots\star & \star\dots\star \\ \hline 1\star\dots\star & \star\dots\star & \star\dots\star & \star\dots\star & \dots & \dots & \star\dots\star & \star\dots\star \\ 1\star\dots\star & \star\dots\star & \star\dots\star & \star\dots\star & \dots & \dots & \star\dots\star & \star\dots\star \end{pmatrix}$$

Сравниваем соответствующие блоки матриц смежности графов. Требования такие же, как и на первом шаге: во-первых, количество блоков должно совпадать; во-вторых, соответствующие блоки в матрицах смежности должны иметь одинаковый размер; в-третьих, соответствующие блоки в матрицах смежности должны содержать равное

количество 1. Если хотя бы одно из этих требований не выполняется, то графы не изоморфны.

**Шаг 2.** Упорядочиваем множество вершин, соответствующих каждому блоку, полученному на предыдущем шаге таким образом, чтобы сначала были перечислены вершины не смежные со второй, а затем — смежные с ней. В матрице смежности переставляем строки и соответствующие им столбцы согласно этому порядку. Матрица смежности разбивается на более мелкие блоки.

$V_1^{00}$	$V_1^{01}$	$V_1^{10}$	$V_1^{11}$	$V_2^{00}$	$V_2^{01}$	$V_2^{10}$	$V_2^{11}$	...	$V_k^{00}$	$V_k^{01}$	$V_k^{10}$	$V_k^{11}$
0..0	0..0	1..1	1..1	0..0	0..0	1..1	1..1	...	0..0	0..0	1..1	1..1
0..0	1..1	0..0	1..1	0..0	1..1	0..0	1..1	...	0..0	1..1	0..0	1..1
00★	★	★	★	★	★	★	★	...	★	★	★	★
00★	★	★	★	★	★	★	★	...	★	★	★	★
01★	★	★	★	★	★	★	★	...	★	★	★	★
01★	★	★	★	★	★	★	★	...	★	★	★	★
10★	★	★	★	★	★	★	★	...	★	★	★	★
10★	★	★	★	★	★	★	★	...	★	★	★	★
11★	★	★	★	★	★	★	★	...	★	★	★	★
11★	★	★	★	★	★	★	★	...	★	★	★	★
...	...	...	...	...	...	...	...	...	...	...	...	...
00★	★	★	★	★	★	★	★	...	★	★	★	★
00★	★	★	★	★	★	★	★	...	★	★	★	★
01★	★	★	★	★	★	★	★	...	★	★	★	★
01★	★	★	★	★	★	★	★	...	★	★	★	★
10★	★	★	★	★	★	★	★	...	★	★	★	★
10★	★	★	★	★	★	★	★	...	★	★	★	★
11★	★	★	★	★	★	★	★	...	★	★	★	★
11★	★	★	★	★	★	★	★	...	★	★	★	★

Сравниваем соответствующие блоки матриц смежности графов также как на предыдущем шаге: во-первых, количество блоков должно совпадать; во-вторых, соответствующие блоки в матрицах смежности должны иметь одинаковый размер; в-третьих, соответствующие блоки в матрицах смежности должны содержать равное количество 1. Если хотя бы одно из этих требований не выполняется, то графы не изоморфны.

...

**Шаг  $m$ .** Упорядочиваем множество вершин, соответствующих каждому блоку, полученному на предыдущем шаге таким образом, чтобы сначала были перечислены вершины не смежные с  $m$ -й, а затем — смеж-

ные с ней. В матрице смежности переставляем строки и соответствующие им столбцы согласно этому порядку. Сравниваем соответствующие блоки матриц смежности графов: во-первых, количество блоков должно совпадать; во-вторых, соответствующие блоки в матрицах смежности должны иметь одинаковый размер; в-третьих, соответствующие блоки в матрицах смежности должны содержать равное количество 1. Если хотя бы одно из этих требований не выполняется, то графы не изоморфны.

...

Дальнейшее преобразование матриц становится невозможным, если каждый блок состоит либо из 0, либо из 1. Если на некотором шаге матрицы смежности графов совпадут, то алгоритм прекращает работу (в этом случае, графы изоморфны).

✓ **ПРИМЕР 4.1.** Являются ли изоморфными графы  $G$ ,  $G'$ ,  $G''$  (рис. 18).

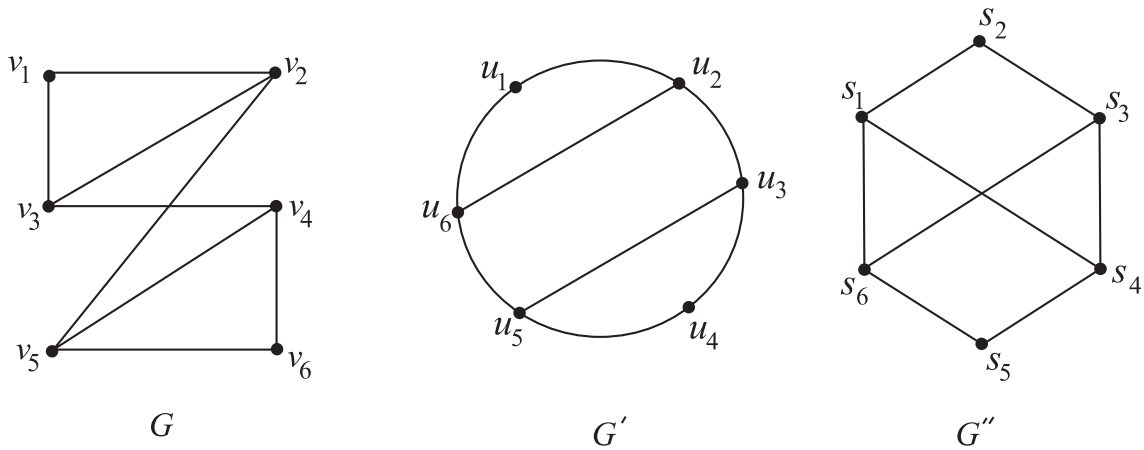


Рис. 18.

**Решение.** Запишем матрицы смежности графов:

$$A(G) = \begin{pmatrix} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}, \quad A(G') = \begin{pmatrix} u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}, \quad A(G'') = \begin{pmatrix} s_1 & s_2 & s_3 & s_4 & s_5 & s_6 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}.$$

Шаг 0. Упорядочим вершины каждого графа по убыванию степеней:

$$G : \underbrace{v_2, v_3, v_4, v_5}_{\deg(v)=3}, \underbrace{v_1, v_6}_{\deg(v)=2}; \quad G' : \underbrace{u_2, u_3, u_5, u_6}_{\deg(v)=3}, \underbrace{u_1, u_4}_{\deg(v)=2}; \quad G'' : \underbrace{s_1, s_3, s_4, s_6}_{\deg(v)=3}, \underbrace{s_2, s_5}_{\deg(v)=2}.$$

В соответствии с этим порядком, переставим строки и столбцы в матрицах смежности:

$$A(G) \simeq \begin{pmatrix} v_2 & v_3 & v_4 & v_5 & v_1 & v_6 \\ \hline 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}, \quad A(G') \simeq \begin{pmatrix} u_2 & u_3 & u_5 & u_6 & u_1 & u_4 \\ \hline 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}, \quad A(G'') \simeq \begin{pmatrix} s_1 & s_3 & s_4 & s_6 & s_2 & s_5 \\ \hline 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ \hline 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}.$$

Таким образом, каждая матрица разбилась на "блоки" из вершин степени 3 и вершин степени 2.

Шаг 1. Упорядочим столбцы (и строки) в каждом "блоке" матриц таким образом, чтобы сначала были перечислены вершины не смежные с первой в блоке, а затем — смежные с первой:

$$A(G) \simeq \begin{pmatrix} v_2 & v_4 & v_3 & v_5 & v_6 & v_1 \\ \hline 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ \hline 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}, \quad A(G') \simeq \begin{pmatrix} u_2 & u_5 & u_3 & u_6 & u_4 & u_1 \\ \hline 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ \hline 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ \hline 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}, \quad A(G'') \simeq \begin{pmatrix} s_1 & s_3 & s_4 & s_6 & s_5 & s_2 \\ \hline 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ \hline 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Шаг 2, Шаг 3, Шаг 4 оставляют матрицы без изменений, т.к. требуемый порядок в блоках по 2-й, 3-й и 4-й строкам матриц уже имеется.

Шаг 5. В матрице  $A(G')$  переставляем строки и столбцы, соответствующие вершинам  $u_3$ ,  $u_6$ . Разбиваем блоки матриц по 5-й строке.

$$A(G) \simeq \begin{pmatrix} v_2 & v_4 & v_3 & v_5 & v_6 & v_1 \\ \hline 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ \hline 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}, \quad A(G') \simeq \begin{pmatrix} u_2 & u_5 & u_6 & u_3 & u_4 & u_1 \\ \hline 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ \hline 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}, \quad A(G'') \simeq \begin{pmatrix} s_1 & s_3 & s_4 & s_6 & s_5 & s_2 \\ \hline 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ \hline 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Дальнейшее преобразование матриц невозможно, т.к. каждый блок состоит либо из 0, либо из 1. В результате преобразований, матрицы смежности графов  $G$  и  $G'$  совпали, следовательно, эти графы изоморфны. Взаимно

- однозначное соответствие между множествами их вершин можно задать следующей таблицей:

$$\frac{G}{G'} \left| \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \\ \hline u_1 & u_2 & u_6 & u_5 & u_3 & u_4 \end{array} \right.$$

Графы  $G$ ,  $G''$  не изоморфны, т.к. их преобразованные матрицы смежности содержат разное количество блоков.

## УПРАЖНЕНИЯ

1. Являются ли изоморфными графы, изображенные на рис. 19.

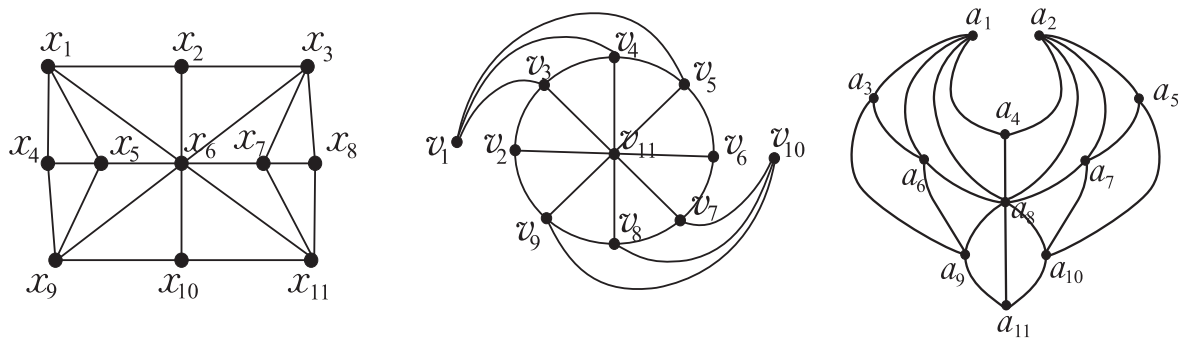


Рис. 19.

2. Нарисовать все неизоморфные простые графы

- с пятью вершинами;
- с шестью вершинами и десятью ребрами.

## 5. Представление сетей радиосвязи графами

- Опр.* | Сеть сотовой связи представляет собой совокупность приемо-передающих станций, обслуживающих определенную территорию.
- Опр.* | Вокруг каждой приемо-передающей станции выделяется ее зона покрытия, называемая *сотой*.
- Опр.* | Модуль сети радиосвязи  $R_0$  — расстояние между соседними станциями сети сотовой структуры.
- Опр.* | Координационное расстояние  $D$  — расстояние между станциями, работающими в одном частотном канале.
- Опр.* | Радиус зоны обслуживания — область, в которой обеспечивается прием сигналов с заданным качеством.




*Опр.*


*Граф сети радиосвязи* — граф, вершины которого соответствуют пунктам установки передающих станций, а ребрами соединены те передатчики, которые создают помехи приему в соответствующей зоне обслуживания друг друга.

**Замечание 1.** Поскольку распространение радиоволн одинаково во всех направлениях, ребра графа сети связи являются неориентированными, однократными и без петель. Т.е. граф сети связи — простой.


**Замечание 2.** Чтобы разделить территорию на соты оптимально, т.е. без перекрытия или пропусков некоторых участков, могут быть использованы три геометрические фигуры: треугольник, квадрат или шестиугольник. Наиболее подходящей фигурой является шестиугольник, поскольку антенна, установленная в его центре будет покрывать почти всю его площадь. Модуль сети вычисляется как расстояние между центрами смежных шестиугольников, а радиус зоны обслуживания — расстояние от центра шестиугольника до вершины.

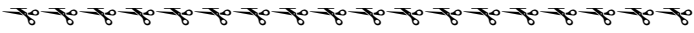
## Построение графа связи с помощью координационных колец

**Идея алгоритма.** 

Задано расположение станций на местности, рабочие частоты, координационное расстояние  $D$ . Строим вокруг первой станции окружность радиуса  $D$ . Для всех станций, попавших внутрь круга проверяем совпадают ли их рабочие частоты с частотой первой станции. Если частоты совпадают, то соответствующие данным станциям вершины в графе связи соединяем ребром с первой вершиной. Повторяем все вышеописанные действия для остальных станций. 

## Построение графа связи с помощью матриц

**Идея алгоритма.** 

Задано расположение станций на местности, рабочие частоты, координационное расстояние  $D$ . Формируем матрицу  $T$  такую, что  $t_{ij} = 1$  ( $i \neq j$ ) если рабочие частоты станций  $i$  и  $j$  совпадают, и  $t_{ij} = 0$  при  $i = j$  или при несовпадении частот. Вычисляем расстояния между станциями с одинаковыми частотами ( $t_{ij} = 1$ ) и на основе матрицы  $T$  формируем матрицу  $A$  у которой элемент  $a_{ij}$  равен 1, если станции  $i$  и  $j$  ( $i \neq j$ ) имеют одинаковые частоты (т.е.  $t_{ij} = 1$ ) и расстояние между ними не больше координационного. Во всех остальных случаях  $a_{ij} = 0$ . Полученная матрица  $A$  — матрица смежности графа сети связи. 

✓ **ПРИМЕР 5.1.** Структура сети сотовой связи показана на рис. 20 (масштаб — 1 клетка=1 км). Координационное расстояние равно 5 км, рабочие частоты станций удовлетворяют условиям:  $f_1 = f_4$ ,  $f_2 = f_3 = f_5$ ,  $f_6 = f_7 = f_8$ . Построить граф сети связи и задать его матрицей смежности  $A(G)$ .

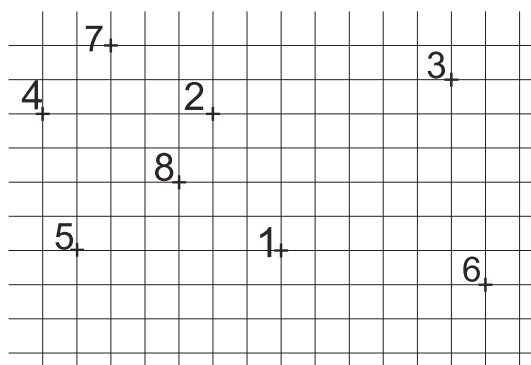


Рис. 20.

### Решение:

Построение графа связи с помощью координационных колец.

Шаг 1. Вокруг каждой станции строятся окружности радиусом, равным координационному расстоянию, в нашем случае 5 км. На рис. 21 приведен пример координационного кольца для станции 8.

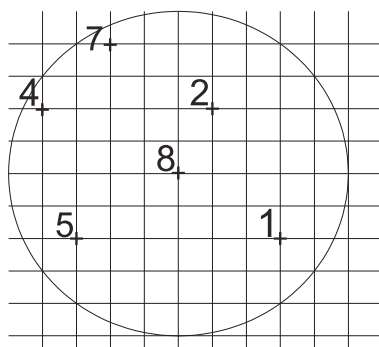


Рис. 21.

Шаг 2. Станции, лежащие внутри данного кольца, могут создавать помехи для станции 8, если работают с ней на одной частоте. Поэтому проверяем, какие частоты принадлежат станциям, лежащим внутри кольца. В данном случае станции 7 и 8 имеют одинаковые рабочие частоты, следовательно, оказывают друг на друга мешающее влияние. Ребром соединяем вершины 7 и 8.

Шаг 3. Аналогичные операции выполняются и для остальных станций сети. В результате построения колец и анализа помех, новых ребер в нашем

случае не обнаруживается. Следовательно, получаем граф связи изображенный на рис. 22.

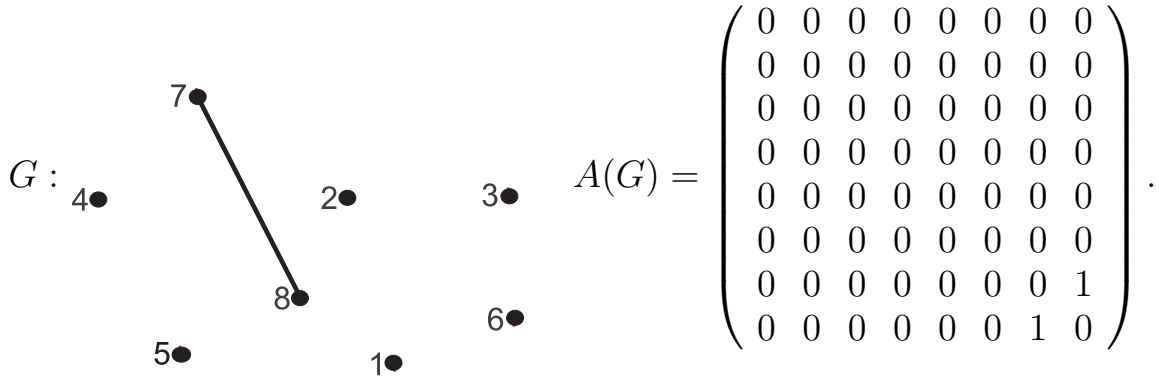


Рис. 22.

**Замечание.** На рисунке, задающем граф сети связи, нет необходимости соблюдать порядок расположения вершин соответствующий реальной сети и выдерживать расстояния. Наличие помех отмечается ребрами.

Построение графа связи с помощью матриц.

Анализируя условия, связывающие частоты, строим матрицу  $T$ :

$$T = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

Матрица симметрична, для построения матрицы смежности  $A(G)$  достаточно вычислить расстояния между станциями, которым соответствуют единицы в треугольнике над диагональю матрицы  $T$ :

$$T = \begin{pmatrix} 0 & 0 & 0 & \boxed{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & \boxed{1} & 0 & \boxed{1} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & \boxed{1} & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \boxed{1} & \boxed{1} \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & \boxed{1} \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

$$\rho(1, 4) = \sqrt{7^2 + 4^2} = \sqrt{65} \approx 8,06 > 5, \Rightarrow a_{1,4} = a_{4,1} = 0;$$

$$\begin{aligned}
\rho(2, 3) &= \sqrt{7^2 + 1^2} = \sqrt{50} \approx 7,07 > 5, \Rightarrow a_{2,3} = a_{3,2} = 0; \\
\rho(2, 5) &= \sqrt{4^2 + 4^2} = \sqrt{32} \approx 5,65 > 5, \Rightarrow a_{2,5} = a_{5,2} = 0; \\
\rho(3, 5) &= \sqrt{11^2 + 5^2} = \sqrt{136} \approx 11,66 > 5, \Rightarrow a_{3,5} = a_{5,3} = 0; \\
\rho(6, 7) &= \sqrt{11^2 + 7^2} = \sqrt{170} \approx 13,03 > 5, \Rightarrow a_{6,7} = a_{7,6} = 0; \\
\rho(6, 8) &= \sqrt{9^2 + 3^2} = \sqrt{90} \approx 9,48 > 5, \Rightarrow a_{6,8} = a_{8,6} = 0; \\
\rho(7, 8) &= \sqrt{2^2 + 4^2} = \sqrt{20} \approx 4,47 < 5, \Rightarrow a_{7,8} = a_{8,7} = 1.
\end{aligned}$$

$$A(G) = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

**Замечание.** Расстояния в сотовой сети будем вычислять следующим образом. Все станции можно разбить на группы, элементы которых равноудалены от начальной станции  $A_0$ , т.е. расположены на окружности с центром  $A_0$  определенного радиуса (см. рис. 23).

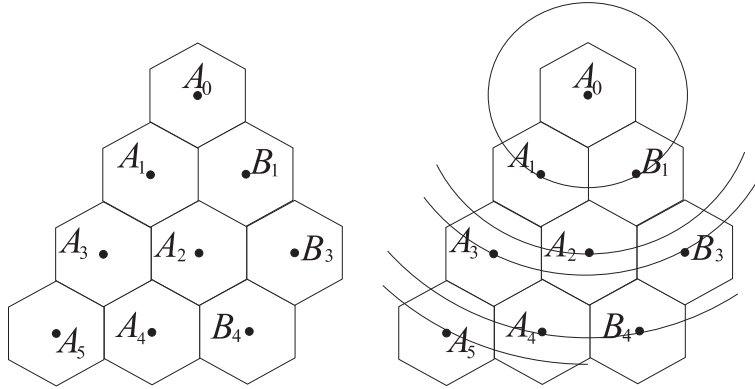


Рис. 23.

В правильном шестиугольнике со стороной  $a$  расстояние от центра до вершины равно  $a$ , а расстояние от центра до стороны равно  $\frac{a\sqrt{3}}{2}$  (высота правильного треугольника). В сети сотовой связи с модулем  $R_0$  длина стороны вычисляется по формуле  $a = \frac{R_0}{\sqrt{3}}$  (рис. 24). Вычислим расстояние от центра верхнего шестиугольника  $A_0$  до центров остальных. Заметим, что  $|A_0A_1| = |A_0B_1|$ ,  $|A_0A_3| = |A_0B_3|$ ,  $|A_0A_4| = |A_0B_4|$ . Расстояние между  $A_0$  и

$A_1$  равно модулю сети  $R_0$  (см. рис. 24 а)); расстояние между  $A_0$  и  $A_2$  равно  $R_0\sqrt{3}$  (см. рис. 24 б)); расстояние между  $A_0$  и  $A_3$  равно  $2R_0$  (см. рис. 24 в)). Расстояние между  $A_0$  и  $A_4$  найдем по теореме косинусов (см. рис. 24 г)):

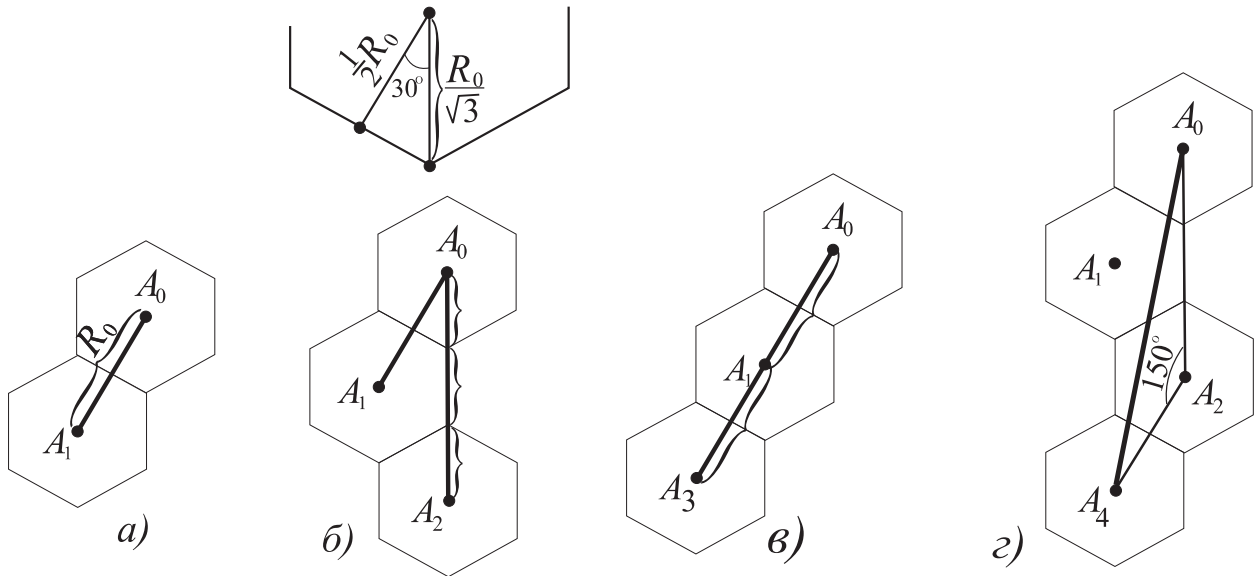


Рис. 24.

$$\begin{aligned}
 |A_0A_4| &= \sqrt{|A_0A_2|^2 + |A_2A_4|^2 - 2|A_0A_2| \cdot |A_2A_4| \cdot \cos 150^\circ} = \\
 &= \sqrt{3R_0^2 + R_0^2 + 2\sqrt{3}R_0^2 \cdot \frac{\sqrt{3}}{2}} = \sqrt{7}R_0.
 \end{aligned}$$

Если координационное расстояние достаточно большое, то необходимо рассматривать более крупный фрагмент сети, но принцип вычисления расстояний будет тот же.

✓ **ПРИМЕР 5.2.** Фрагмент однородной сотовой сети представлен на рис. 25. Радиус зоны обслуживания (радиус окружности, описанной вокруг шестиугольника) равен 4 км, координационное расстояние равно 18,3 км. Определить, какие из указанных базовых станций могут оказывать мешающее влияние на базовую станцию 1.

**Решение:**

Шаг 1. Зная радиус зоны обслуживания, определяется модуль сети:

$$R_0 = R_{\text{зоны}} \cdot \sqrt{3} = 4\sqrt{3} \approx 6,93 \text{ км.}$$

Шаг 2. Определяется расстояние  $R_1$  до ближайших станций 2, 3, 4, 5, 6, 7 — оно равно модулю сети

$$R_1 = 4\sqrt{3} \approx 6,93 \text{ км.}$$

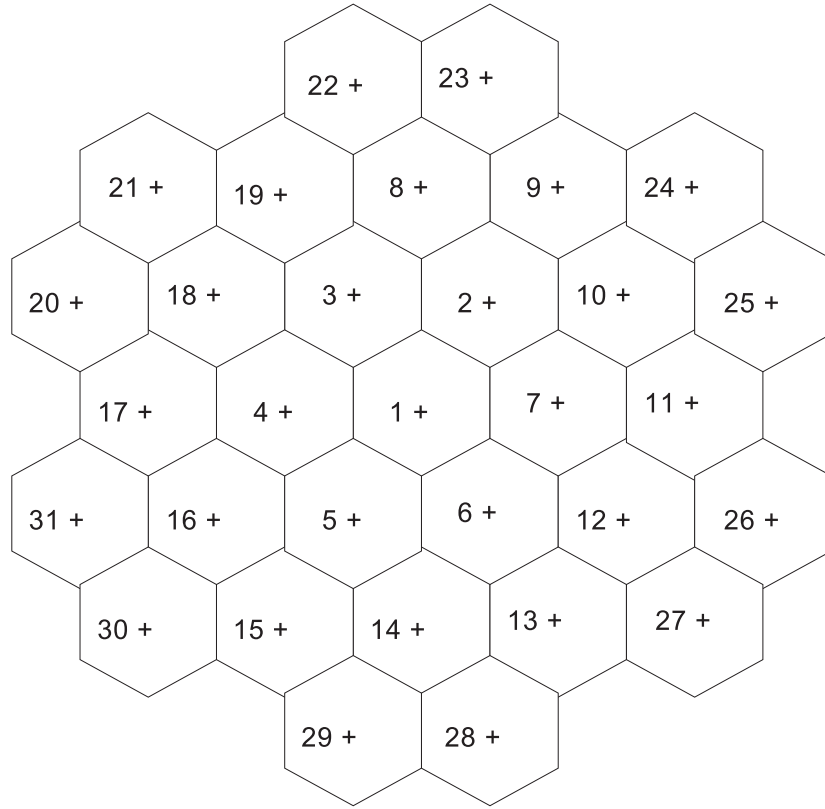


Рис. 25.

Шаг 3. Определяем расстояние  $R_2$  до станций 8, 10, 12, 14, 16, 18:

$$R_2 = R_0\sqrt{3} = 4\sqrt{3}\sqrt{3} = 12\text{км.}$$

Шаг 4. Определяем расстояние  $R_3$  до станций 9, 11, 13, 15, 17, 19:

$$R_3 = 2R_0 = 8\sqrt{3} \approx 13,86\text{км.}$$

Шаг 5. Определим расстояние  $R_4$  до станций 21, 23, 25, 27, 29, 31:

$$R_4 = \sqrt{7}R_0 = \sqrt{7} \cdot 4\sqrt{3} = 4\sqrt{21} \approx 18,33\text{км.}$$

Расстояние  $R_4$  больше координационного. В результате получаем часть графа связи, характеризующую станцию 1 (рис. 26):

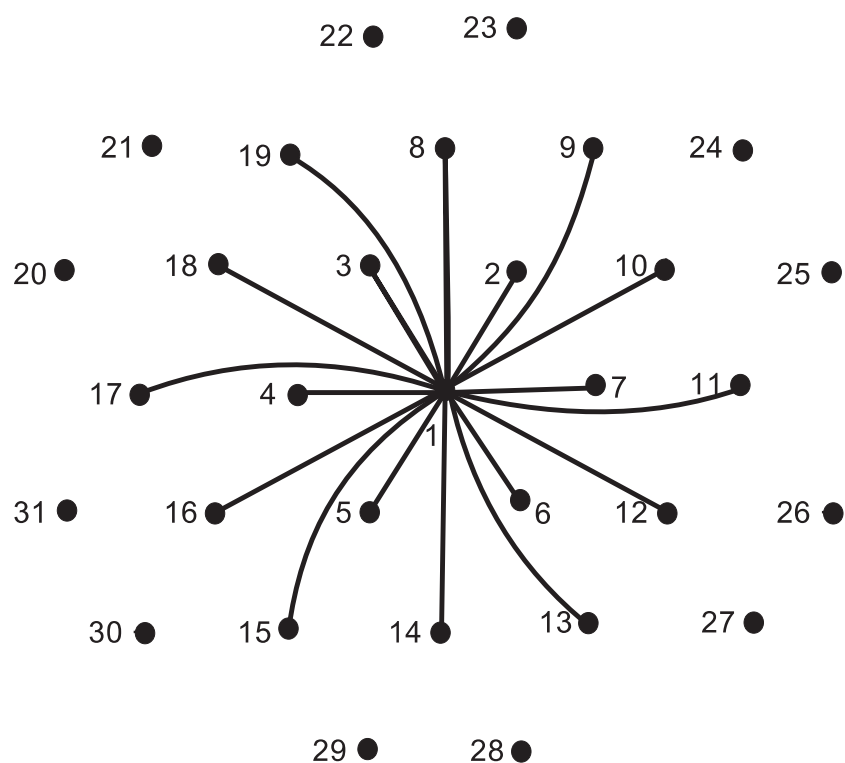


Рис. 26.

Для того, чтобы построить граф всей сети необходимо проанализировать мешающее влияние всех станций друг на друга. Очевидно, что для каждой станции мешающие будут располагаться таким же характерным "веером", как для станции 1.

## УПРАЖНЕНИЯ

1. Сеть радиосвязи состоит из 8 станций, расположение которых показано на рис. 27а). Координационное расстояние равно 4 км, рабочие частоты всех станций одинаковы. Построить для данной сети граф.
2. Сеть радиовещания состоит из 5 передающих станций, ее структура показана на рис. 27б) (масштаб — 1 клетка=1 км). Координационное расстояние равно 4 км, рабочие частоты всех станций одинаковы. Построить граф сети.
3. Для радиопокрытия города N установлено 8 базовых станций (см. рис. 27в)). Координационное расстояние равно 9 км, рабочие частоты станций следующие:  $f_1 = f_4 = f_7$ ,  $f_2 = f_3 = f_6$ ,  $f_5 = f_8$ . Определить матрицу смежности и построить граф сети.
4. Сеть радиовещания состоит из 6 передающих станций (рис. 27г)). Координационное расстояние равно 18 км, рабочие частоты всех станций одинаковы. Построить для данной сети граф.

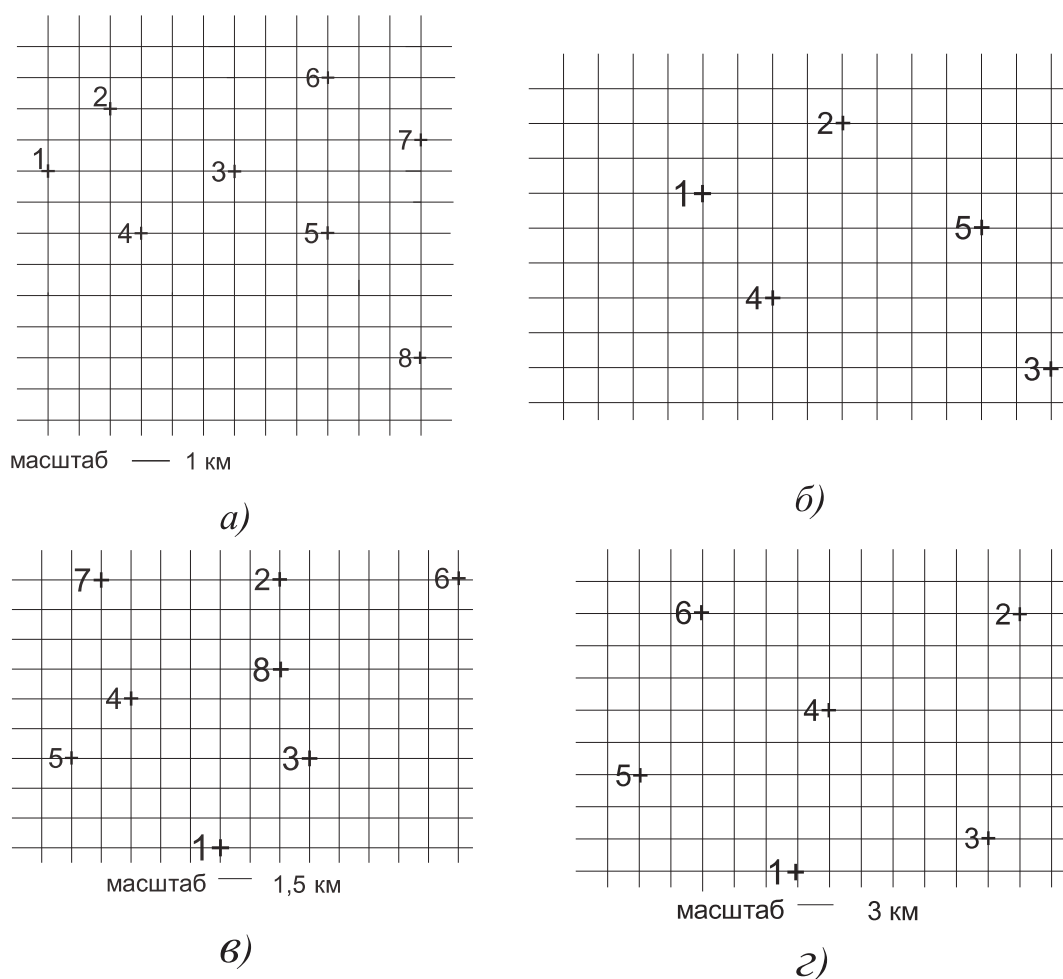


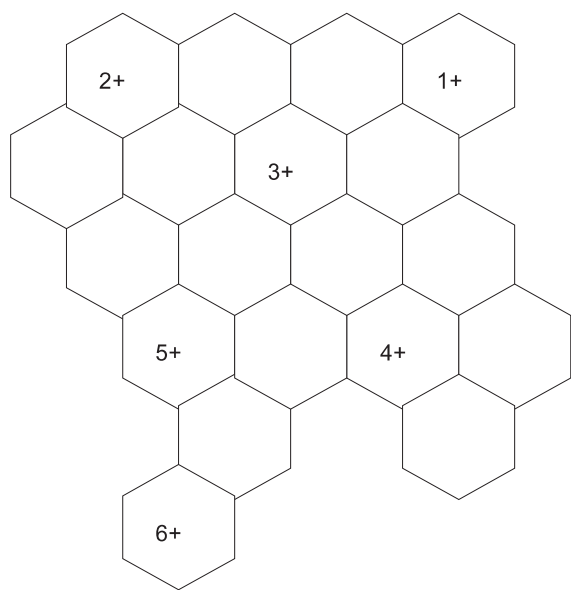
Рис. 27.

**5.** Структура сети радиовещания показана на рис. 28. а) Радиус зоны обслуживания равен 38 км, координационное расстояние — 114 км. Определить, будут ли указанные станции оказывать взаимные влияния друг на друга, построить граф сети и матрицу смежности.

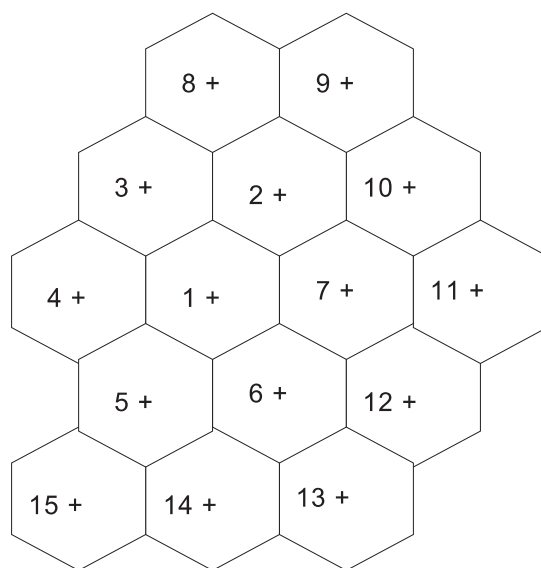
**6.** В городе N установлены 15 базовых станций (рис. 28. б)). Радиус зон покрытия базовых станций равен 3,5 км, координационное расстояние — 18,2 км. Определить, будут ли станции сети оказывать взаимные влияния друг на друга, построить граф сети и матрицу смежности.

**7.** Структура сети сотовой связи города N представлена на рис. 28. в) Радиус зон покрытия базовых станций равен 1,6 км, координационное расстояние — 10 км. Определить, какие станции будут оказывать взаимные влияния друг на друга и построить граф сети.

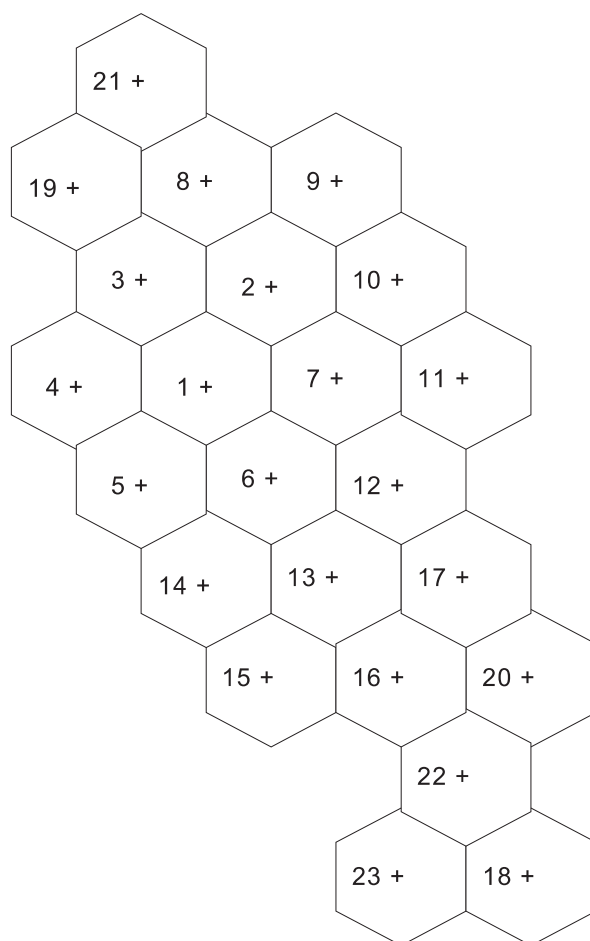




a)



б)



в)

Рис. 28.

## 6. СВЯЗНОСТЬ

*Опр.*

Маршрут в неориентированном графе — последовательность из вершин и ребер  $v_{i_0}, e_{i_1}, v_{i_1}, e_{i_2}, \dots, e_{i_k}, v_{i_k}$ , где любые два соседних элемента инцидентны. Если  $v_{i_0} = v_{i_k}$ , то маршрут называется *замкнутым*, иначе — *открытым*.

**Замечание.** Для простого графа достаточно указать только последовательность вершин или только последовательность ребер.

*Опр.*

Цепь — маршрут, все ребра которого различны. Замкнутая цепь называется *циклом*.

*Опр.*

Простая цепь — маршрут, все вершины которого различны. Замкнутая простая цепь называется *простым циклом*.

**Замечание.** Цепь в ориентированном графе — *путь*, а цикл — *контур*.

*Опр.*

Длина маршрута равна количеству ребер в нем (с повторениями).

*Опр.*

Расстояние  $d(u, v)$  от вершины  $u$  до вершины  $v$  равно количеству ребер в кратчайшей простой цепи (пути) из  $u$  в  $v$ .

**Замечание 1.** Если в графе не существует пути, соединяющего вершины  $u$  и  $v$ , то расстояние между ними полагаем равным бесконечности.

**Замечание 2.** В ориентированном графе расстояние от вершины  $u$  до вершины  $v$  не всегда равно расстоянию от вершины  $v$  до вершины  $u$  (см. рис. 29).

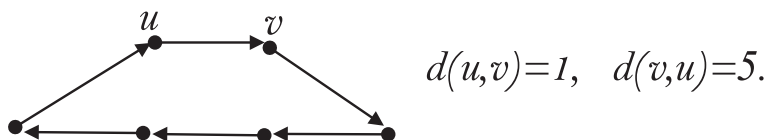


Рис. 29.

*Опр.*

Вершина  $u$  *достижима* из вершины  $v$ , если существует цепь (путь) из  $v$  в  $u$ .

*Опр.*

Связный неориентированный граф — граф, для любых двух вершин которого существует соединяющий их маршрут.

*Опр.*

Компонента связности неориентированного графа — максимальный по включению связный подграф.

*Опр.*

Точка сочленения — вершина, удаление которой увеличивает число компонент связности.

*Опр.*

Мост — ребро, удаление которого увеличивает число компонент связности.

✓ **ПРИМЕР 6.1.** На рис. 30 приведен пример связного и несвязного неориентированного графа.

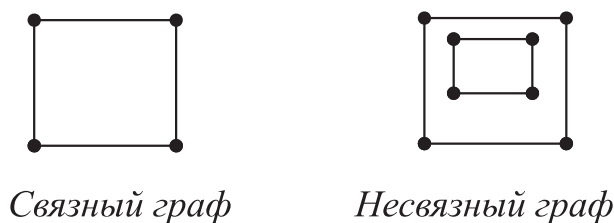


Рис. 30.

**Замечание.** Мост в графе также называют *перешейком*.

В ориентированном графе из того, что вершина  $u$  достижима из вершины  $v$  не следует, что  $v$  достижима из  $u$ , поэтому различают три типа связности.

*Опр.*

Если любые две вершины орграфа  $\vec{G}$  достижимы друг из друга, то орграф *сильно связный*. Если для любых  $u, v \in V(\vec{G})$   $u$  достижима из  $v$  или  $v$  достижима из  $u$ , то орграф *связный*. Если связным является граф  $G$ , полученный из  $\vec{G}$  заменой всех дуг на ребра (отменой ориентации), то  $\vec{G}$  — *слабо связный*.

✓ **ПРИМЕР 6.2.** На рис. 31 приведены примеры орграфов с различными типами связности.

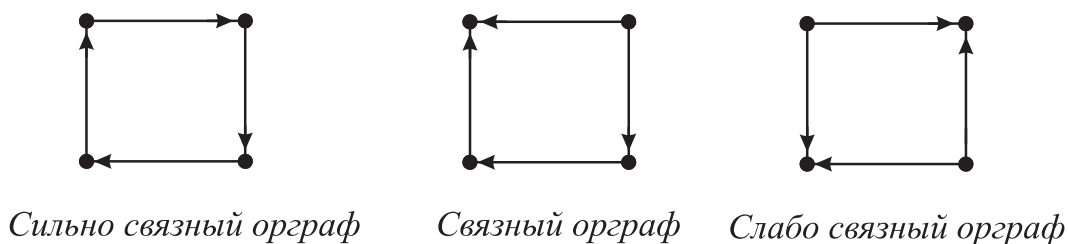


Рис. 31.

*Опр.*

Матрица достижимости  $R(\vec{G})$  орграфа  $\vec{G}$  — матрица порядка  $n$ , у которой  $r_{ij} = 1$ , если вершина  $v_j$  достижима из вершины  $v_i$  и  $r_{ij} = 0$  — в противном случае.

*Опр.*

Матрица сильной связности орграфа  $\vec{G}$  — матрица  $S(\vec{G})$  порядка  $n$ , у которой  $s_{ij} = 1$ , если вершина  $v_j$  достижима из вершины  $v_i$  и вершина  $v_i$  достижима из вершины  $v_j$ ;  $s_{ij} = 0$  — в противном случае.

**Замечание 1.** Любая вершина считается достижимой сама из себя, поэтому все элементы на главной диагонали в матрицах достижимости и сильной связности равны 1.

**Замечание 2.** Матрица сильной связности орграфа получается логическим поэлементным умножением матрицы достижимости  $R(\vec{G})$  на транспонированную матрицу достижимости —  $R^T(\vec{G})$ :

$$S = R \& R^T \Leftrightarrow s_{ij} = r_{ij} \& r_{ji}.$$

**Замечание 3.** В неориентированном графе матрица связности совпадает с матрицей достижимости.


*Опр.*

Конденсат ориентированного графа — орграф, полученный из данного отождествлением вершин, входящих в одну компоненту сильной связности.

**Замечание.** Конденсат графа также называется *графом Краутца*.

## 6.1. Алгоритм выделения компонент сильной связности

Алгоритм выделения компонент сильной связности **Идея алгоритма.**

Объединяем в первую компоненту связности все вершины, которые сильно связаны с первой и исключаем их из дальнейшего рассмотрения. Затем, из вершин, не попавших в первую компоненту, выбираем вершину с минимальным номером, и, во вторую компоненту объединяем вершины сильно связанные с ней. Исключаем вершины второй компоненты из рассмотрения и повторяем процедуру. Продолжаем до тех пор, пока все вершины не будут распределены по компонентам. 

**Шаг 0.** Для орграфа  $\vec{G}(V, E)$  строим матрицу сильной связности  $S(\vec{G})$ .

**Шаг 1.** В первую компоненту сильной связности зачисляем все вершины, которым соответствует единица в первой строке матрицы  $S(\vec{G})$  :

$$K_1 := \{v_i \mid s_{1i} = 1\}$$

(эти вершины сильно связаны с первой и, следовательно, друг с другом). Удаляем из матрицы  $S(\vec{G})$  строки и столбцы, соответствующие вершинам из  $K_1$ , полученную при этом матрицу обозначим  $S_1(\vec{G})$ .

**Шаг 2.** Во вторую компоненту зачисляем все вершины, которым соответствует единица в первой строке матрицы  $S_1(\vec{G})$  :

$$K_2 := \{v_i \mid s_{1i} = 1\}.$$

Удаляем из матрицы  $S_1(\vec{G})$  строки и столбцы, соответствующие вершинам из  $K_2$ , полученную при этом матрицу обозначим  $S_2(\vec{G})$ .

**Шаг  $m$ .** В  $m$ -ю компоненту зачисляем все вершины, которым соответствует единица в первой строке матрицы  $S_{m-1}(\vec{G})$  :

$$K_m := \{v_i \mid s_{1i} = 1\}.$$

Удаляем из матрицы  $S_{m-1}(\vec{G})$  строки и столбцы, соответствующие вершинам из  $K_m$ , полученную при этом матрицу обозначим  $S_m(\vec{G})$ .

Алгоритм прекращает работу, как только все вершины будут распределены по компонентам.

✓ **ПРИМЕР 6.3.** Выделить компоненты сильной связности в орграфе, заданном матрицей смежности:

$$A(\vec{G}) = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

**Решение.**

Запишем матрицу достижимости данного орграфа и матрицу сильной связности (для удобства изобразим орграф на рисунке (рис. 32)).

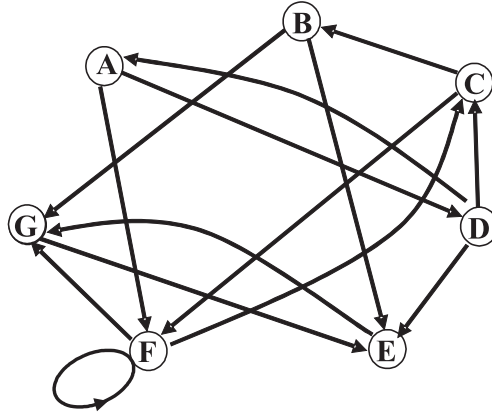


Рис. 32.

$$R(\vec{G}) = \begin{pmatrix} \mathbf{A} & \mathbf{B} & \mathbf{C} & \mathbf{D} & \mathbf{E} & \mathbf{F} & \mathbf{G} \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}, \quad R^T(\vec{G}) = \begin{pmatrix} \mathbf{A} & \mathbf{B} & \mathbf{C} & \mathbf{D} & \mathbf{E} & \mathbf{F} & \mathbf{G} \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix},$$

$$S(\vec{G}) = \begin{pmatrix} \mathbf{A} & \mathbf{B} & \mathbf{C} & \mathbf{D} & \mathbf{E} & \mathbf{F} & \mathbf{G} \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix}, \quad K_1 = \{\mathbf{A}, \mathbf{D}\};$$

$$S_1(\vec{G}) = \begin{pmatrix} \mathbf{B} & \mathbf{C} & \mathbf{E} & \mathbf{F} & \mathbf{G} \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix}, \quad S_2(\vec{G}) = \begin{pmatrix} \mathbf{C} & \mathbf{E} & \mathbf{F} & \mathbf{G} \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}, \quad S_3(\vec{G}) = \begin{pmatrix} \mathbf{E} & \mathbf{G} \\ 1 & 1 \\ 1 & 1 \end{pmatrix},$$

$$K_2 = \{\mathbf{B}\}; \quad K_3 = \{\mathbf{C}, \mathbf{F}\}; \quad K_4 = \{\mathbf{E}, \mathbf{G}\}.$$

Так как все вершины уже распределены по компонентам, алгоритм прекращает работу.

Конденсат данного орграфа представлен на рис. 33.

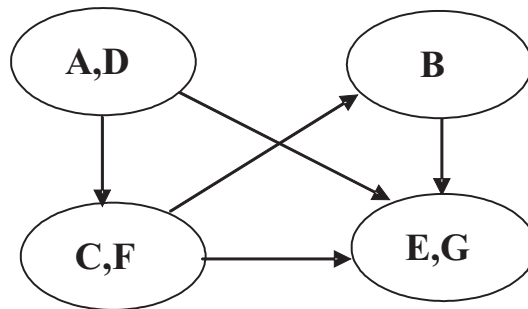


Рис. 33.

✓ **ПРИМЕР 6.4.** Выделить компоненты сильной связности в орграфе, изображенном на рис. 34.

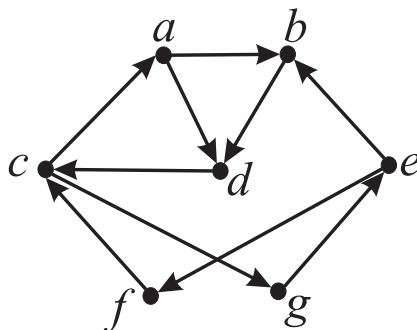


Рис. 34.

**Решение.**

Запишем матрицу достижимости данного графа и матрицу сильной связности.

$$R(\vec{G}) = \begin{pmatrix} a & b & c & d & e & f & g \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix},$$

$$R(\vec{G}) = R^T(\vec{G}) = S(\vec{G}).$$

$K_1 = V(\vec{G})$  — орграф сильно связный.

✓ **ПРИМЕР 6.5.** Выделить компоненты сильной связности в орграфе, изображенном на рис. 35.

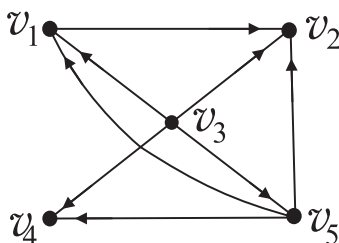


Рис. 35.

**Решение.**

Запишем матрицу достижимости данного орграфа и матрицу сильной связности.

$$R(\vec{G}) = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 \end{pmatrix}, \quad R^T(\vec{G}) = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{pmatrix},$$

$$S(\vec{G}) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

Орграф распадается на одновершинные компоненты:

$$K_1 = \{v_1\}, \quad K_2 = \{v_2\}, \quad K_3 = \{v_3\}, \quad K_4 = \{v_4\}, \quad K_5 = \{v_5\}.$$

## УПРАЖНЕНИЯ

Выделить компоненты сильной связности в орграфах, заданных матрицами смежности:

$$1) A(\vec{G}_1) = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}, \quad 2) A(\vec{G}_2) = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

## 7. Деревья

*Опр.* | *Дерево* — связный граф без циклов. Несвязный граф без циклов называется *лесом*.

**Замечание.** Граф без циклов также называется *ациклическим*.

**Теорема.** Следующие утверждения эквивалентны:

- 1) граф является деревом;
- 2) граф связный и число его ребер на единицу меньше числа вершин;
- 3) любые две вершины графа соединяет единственная простая цепь.



✓ **ПРИМЕР 7.1.** На рис. 36 изображены все возможные деревья с пятью вершинами.

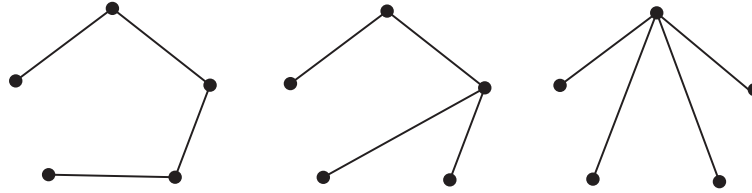


Рис. 36.

*Опр.*

*Остов* или *остовное дерево* графа — любой его подграф, содержащий все вершины графа и являющийся деревом.

✓ **ПРИМЕР 7.2.** На рис. 37 изображен граф  $G$  и некоторые его остовные деревья  $T_i$ . Заметим, что деревья  $T_1, T_2, T_3$  изоморфны.

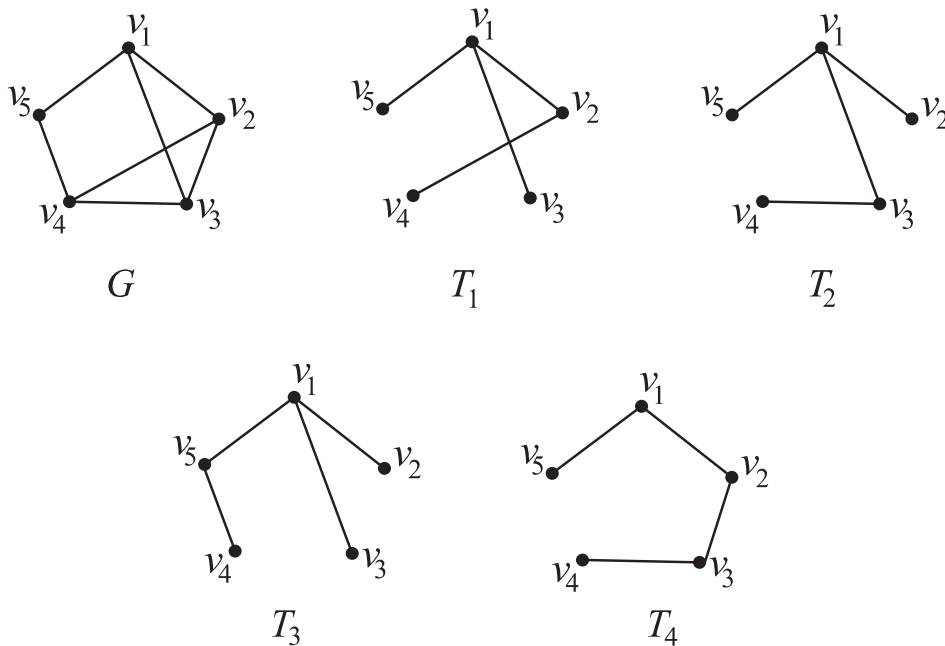


Рис. 37.

*Опр.*

*Цикломатическое число*  $\lambda(G)$  графа  $G$  — число ребер, которые необходимо удалить, чтобы граф стал ациклическим (т.е., деревом, если  $G$  связный и лесом, если  $G$  — несвязный).

---

**Теорема.** Для любого графа  $G(V, E)$  имеет место формула:

$$\lambda(G) = |E(G)| - |V(G)| + k(G),$$

где  $k(G)$  — число компонент связности графа.

---

## УПРАЖНЕНИЯ

1. Нарисовать все неизоморфные остовные деревья полного графа с шестью вершинами.
2. Найти цикломатическое число псевдографа, заданного матрицей смежности:

$$A(G) = \begin{pmatrix} 2 & 2 & 0 & 0 & 1 \\ 2 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 3 & 0 & 0 \\ 1 & 1 & 0 & 0 & 2 \end{pmatrix}.$$


## 8. Обходы


*Опр.*

*Обход графа* — некоторое систематическое перечисление его вершин или ребер.

Среди вершинных обходов наиболее известны "поиск в глубину" и "поиск в ширину". На их основе сформулированы многие из встречающихся далее алгоритмов. К реберным обходам относится эйлеров цикл.

### 8.1. Поиск в глубину

**Идея алгоритма.** 

Выходя из начальной вершины, строим простую цепь. Пройденным вершинам приписываем метки. Продолжаем строить цепь до тех пор, пока не встретим уже отмеченную вершину или не попадем в висячую вершину. В этом случае возвращаемся на шаг назад и выбираем ребро, ведущее к непомяченной вершине. Процесс продолжаем до тех пор, пока всем вершинам не будут присвоены метки. 

Пусть  $G(V, E)$  — связный неориентированный граф. Для каждой вершины будем запоминать ее метку  $s(v)$  и вершину, из которой попали в данную (предшественника)  $p(v)$ . Кроме того, в описании алгоритма будут использоваться следующие обозначения:  $v^*$  — текущая вершина,  $V^*$  — множество еще не помеченных вершин,  $\Gamma(v^*)$  — окрестность вершины  $v^*$ .

**Шаг 0.**  $u \in V$  — произвольная вершина графа. Приписываем ей метку 1.

$$s(u) := 1, \quad s := 1, \quad v^* := u, \quad V^* := V \setminus \{u\}.$$

**Шаг 1.** Алгоритм продолжает работу до тех пор, пока  $|V^*| \neq 0$ . Если  $V^* \cap \Gamma(v^*) = \emptyset$ , то переходим на шаг 2. Если  $V^* \cap \Gamma(v^*) \neq \emptyset$ , то мы выбираем  $v \in V^* \cap \Gamma(v^*)$  и приписываем ей очередную метку:

$$s(v) := s + 1, \quad s := s + 1, \quad p(v) := v^*, \quad V^* := V^* \setminus \{v\}, \quad v^* := v.$$

Переходим на начало шага 1.

**Шаг 2.**  $V^* \cap \Gamma(v^*) = \emptyset$ , и мы "возвращаемся назад", т.е.

$$v^* := p(v).$$

Переходим на шаг 1.

**Замечание 1.** Если граф несвязный, то поиск в глубину осуществляется для вершин, достижимых из начальной.

**Замечание 2.** Поиск может быть осуществлен и на орграфе. В этом случае необходимо учесть направление дуг.

✓ **ПРИМЕР 8.1.** В заданной матрицей смежности графе провести поиск в глубину из вершины  $v_1$ :

$$A(G) = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

**Решение.** Поиск может быть осуществлен несколькими способами, некоторые из них приведены на рис. 38. В скобках указана метка вершины.

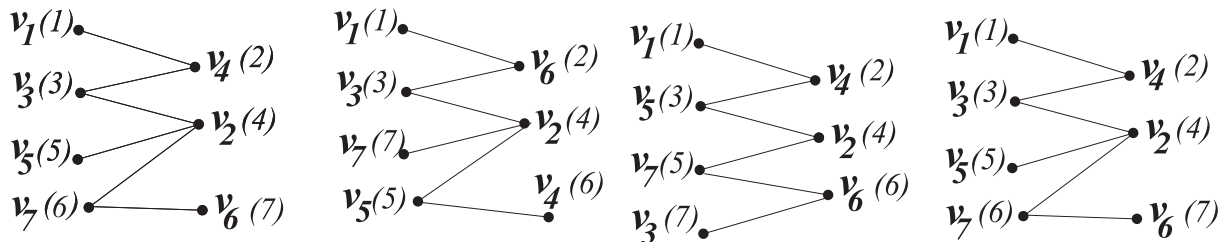


Рис. 38.

✓ **ПРИМЕР 8.2.** В заданном матрицей смежности орграфе обойти все вершины, используя поиск в глубину.

$$A(\vec{G}) = \begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

**Решение.**

Проводим поиск из первой вершины (рис. 39). В скобках указана метка вершины.

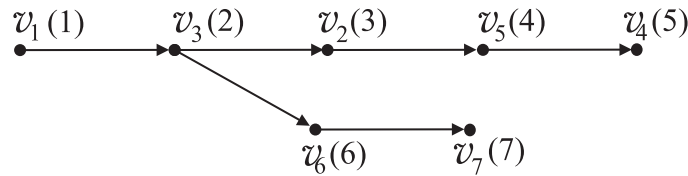



Рис. 39.

## 8.2. Поиск в ширину

**Идея алгоритма.** 

Выходя из начальной вершины, помечаем все вершины из ее окрестности, как вершины первого "уровня". Затем, у каждой из вершин первого уровня помечаем еще не отмеченные вершины ее окрестности, как вершины второго уровня и т.д.. У вершин уровня  $n$  помечаем еще не отмеченные вершины окрестности, как вершины уровня  $n + 1$ . Процесс продолжаем до тех пор, пока все вершины не получают метки. 

Пусть  $G(V, E)$  — связный неориентированный граф. Для каждой вершины будем запоминать ее метку (уровень)  $s(v)$ . Кроме того,  $V^*$  — множество еще не помеченных вершин,  $\Gamma(v)$  — окрестность вершины  $v$ ,  $C_i$  — множество вершин с метками  $s(v) = i$ . В итоге множество вершин  $V$  будет разбито на классы (уровни)  $C_i$ .

**Шаг 0.**  $v \in V$  — начальная вершина поиска. Приписываем ей метку 0.

$$s(v) := 0, \quad C_0 = \{v\}, \quad V^* := V \setminus C_0.$$

**Шаг 1.** Всем вершинам из окрестности  $v$  присваиваем метку 1 и объединяем их в класс  $C_1$ :

$$\forall v \in V^* \cap \Gamma(v), \quad s(v) := 1, \quad C_1 = \{v | s(v) = 1\}, \quad V^* := V^* \setminus C_1.$$

...

**Шаг  $k+1$ .** Пусть  $\Gamma(C_k) = \bigcup_{v \in C_k} \Gamma(v)$ .

Всем вершинам из множества  $V^* \cap \Gamma(C_k)$  присваиваем метку  $n+1$  и объединяем их в класс  $C_{k+1}$ :

$$\forall v \in V^* \cap \Gamma(C_k), \quad s(v) := k+1, \quad C_{k+1} = \{v | s(v) = k+1\}, \quad V^* := V^* \setminus C_{k+1}.$$

...

Процесс продолжаем до тех пор, пока  $V^*$  не пусто.

**Замечание 1.** Если граф несвязный, то поиск как в ширину осуществляется для вершин, достижимых из начальной.

**Замечание 2.** Поиск в ширину можно провести из множества вершин  $X \in V(G)$ . В этом случае, каждый шаг алгоритма выполняется для всех вершин заданного множества по очереди: вершинам данного множества  $X$  присваиваем метку 0, вершинам из окрестности множества  $X$  — метку 1 и т.д. Процесс продолжаем до тех пор, пока все вершины, достижимые из вершин множества  $X$ , не получают метки.

**Замечание 3.** Поиск может быть осуществлен и на орграфе. В этом случае необходимо учесть направление дуг.

✓ **ПРИМЕР 8.3.** В заданном матрицей смежности графе провести поиск в ширину из первой вершины:

$$A(\vec{G}) = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

**Решение.**

Поиск может быть осуществлен несколькими способами, некоторые из них приведены на рис. 40. В скобках указан уровень (метка) вершины.

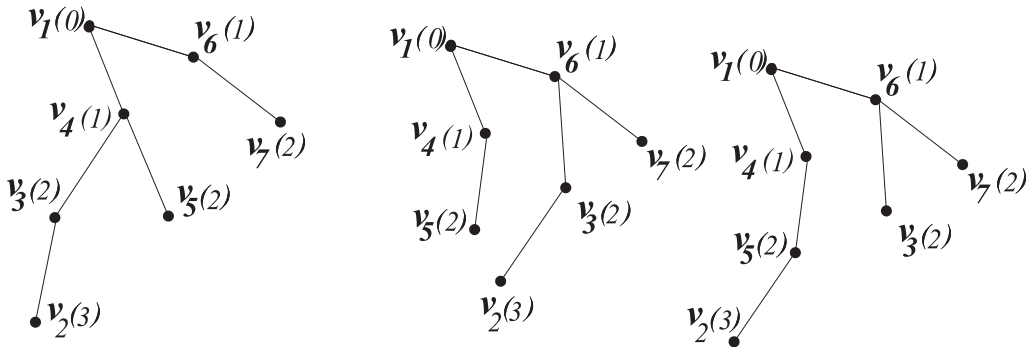


Рис. 40.

Как видно, уровень вершины не зависит от способа поиска. Фактически, метка вершины равна расстоянию от нее до вершины из которой идет поиск (в данном случае — до первой).

✓ **ПРИМЕР 8.4.** В заданном матрицей смежности  $A(\vec{G})$  орграфе провести:

- а) поиск в ширину из вершины  $v_3$ ;
- б) поиск в глубину из вершины  $v_3$ ;
- в) поиск в ширину из множества вершин  $\{v_1, v_3\}$ .

$$A(\vec{G}) = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

**Решение.**

а) Из вершины  $v_3$  ведут дуги в  $v_4$ ,  $v_5$  и  $v_6$  (им соответствуют единицы в третьей строке матрицы смежности). Приписываем вершинам  $v_4$ ,  $v_5$  и  $v_6$  метку 1 и рассматриваем окрестность каждой из них. Из  $v_4$  ведут дуги в  $v_1$ ,  $v_3$  и  $v_5$ . Но вершины  $v_3$  и  $v_5$  уже рассмотрены ранее, поэтому остается только вершина  $v_1$ , которой присваивается метка 2. Из  $v_5$  ведут ребра в  $v_1$ ,  $v_6$  и  $v_7$ . Вершины  $v_1$  и  $v_6$  рассмотрены ранее, присваиваем метку 2 вершине  $v_7$ . Из  $v_6$  ведут дуги в  $v_2$ ,  $v_3$ ,  $v_5$  и  $v_7$ . Ранее не рассматривалась только  $v_2$ , присваиваем ей метку 2. Заметим, что при рассмотрении вершин из окрестности вершины  $v_3$  в другом порядке (например,  $v_6$ ,  $v_5$ ,  $v_4$ ) метки приписанные им были бы такими же, а вот дерево, иллюстрирующее поиск (рис. 41а)) могло бы выглядеть иначе.

б) Из вершины  $v_3$  переходим в вершину с наименьшим номером —  $v_4$  (присваиваем ей метку 1). Из  $v_4$  ведут дуги в  $v_1$ ,  $v_3$  и  $v_5$ . Наименьший номер у  $v_1$  (присваиваем ей метку 2). Из  $v_1$  ведут дуги в  $v_3$ , и  $v_6$ , но  $v_4$  рассмотрена ранее, поэтому метку 3 присваиваем вершине  $v_6$ . Из  $v_6$  ведут дуги в  $v_2$ ,  $v_3$ ,  $v_5$  и  $v_7$ . Вершина  $v_2$  не рассматривалась ранее, присваиваем ей метку 4. Из  $v_2$  ведут дуги в  $v_1$ , и  $v_7$ . Присваиваем вершине  $v_7$  метку 5. Из  $v_7$  ведут дуги в  $v_2$ ,  $v_3$ , и  $v_6$ . Все эти вершины были рассмотрены ранее, поэтому "поднимаемся" в вершину  $v_2$ . В вершине  $v_2$  ситуация та же: все вершины из ее окрестности помечены ранее, "поднимаемся" в вершину  $v_6$ . Из непомеченных вершин смежных с  $v_6$  наименьший номер у  $v_5$ . Так как метка вершины  $v_6$  — 3, то метка  $v_5$  — 4. Все вершины графа помечены. Дерево, иллюстрирующее поиск в глубину представлено на рис. 41б).

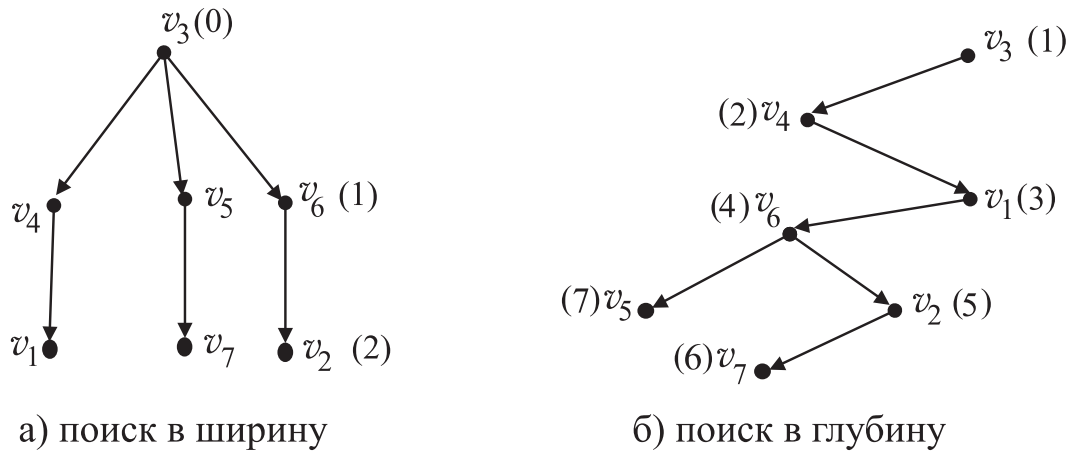


Рис. 41.

в) Вершинам  $v_1$  и  $v_3$  приписываем метку 0. Из вершины  $v_1$  ведут дуги в  $v_4$  и  $v_6$ , а из вершины  $v_3$  ведут дуги в  $v_4$ ,  $v_5$  и  $v_6$ . Приписываем вершинам  $v_4$ ,  $v_5$  и  $v_6$  метку 1 и рассматриваем окрестность каждой из них. Из вершин  $v_4$ ,  $v_5$  и  $v_6$  ведут дуги во все вершины орграфа, кроме  $v_4$ . На данном шаге осталось две не рассмотренных ранее вершины  $v_2$  и  $v_7$ , которым присваивается метка 2. Все вершины графа помечены (рис. 42).

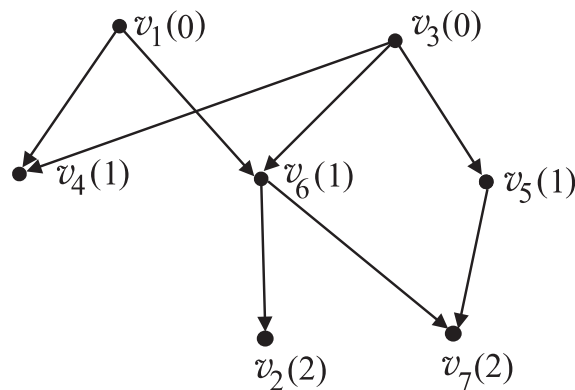


Рис. 42.

### 8.3. Эйлеров цикл

*Опр.* | Эйлеров цикл (цепь) — цикл (цепь), содержащий все ребра графа ровно один раз и все вершины графа (возможно, несколько раз).

*Опр.* | Эйлеров граф — граф, содержащий эйлеров цикл.

**Замечание.** В ориентированном графе эйлеров цикл или цепь определяются аналогично и строятся с учетом направления дуг.

---

**Критерий существования эйлерова цикла.**

- А) Неориентированный граф содержит эйлеров цикл тогда и только тогда, когда он связан и степени всех его вершин четны.
- Б) Ориентированный граф содержит эйлеров цикл тогда и только тогда, когда он сильно связан и для каждой его вершины полустепень захода равна полустепени исхода.
- 

**Критерий существования эйлеровой цепи.**

- А) Неориентированный граф содержит эйлерову цепь тогда и только тогда, когда он связан и ровно две его вершины имеют четную степень.
- Б) Ориентированный граф с  $n$  вершинами содержит эйлерову цепь тогда и только тогда, когда он связан и для  $(n - 2)$ -х его вершины полустепень захода равна полустепени исхода, а для двух вершин  $u$  и  $v$  имеет место равенство:

$$\deg_+(u) - \deg_-(u) = 1, \quad \deg_-(v) - \deg_+(v) = 1.$$

---

✓ **ПРИМЕР 8.5.** Возможно ли нарисовать представленные на рис.43 фигуры не отрывая карандаш от бумаги и не проходя по одной и той же линии дважды?

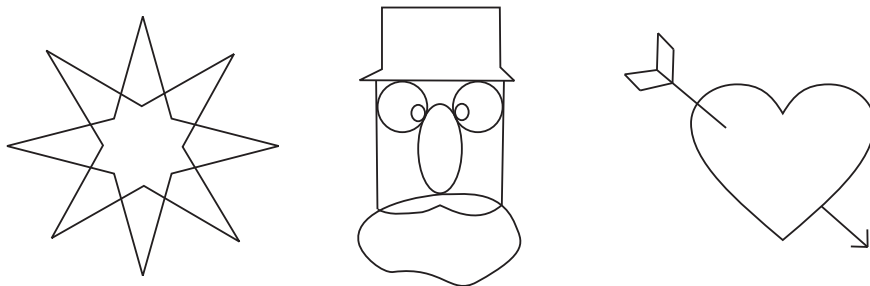


Рис. 43.

**Решение.** Представим данные фигуры как графы. При этом, определим вершины в точках пересечения линий. Если в полученном графе существует эйлеров цикл или цепь, то можно нарисовать не отрывая карандаш от бумаги и не проходя по одной и той же линии дважды. Из рис.44 следует, что первый граф ("звездочка") удовлетворяет критерию существования эйлерова цикла (т.к. все его вершины имеют степень 4), а второй ("рожица") — критерию существования эйлеровой цепи (т.к. ровно две его вершины имеют нечетную степень 3). В третьем графе ("сердечко") шесть из восьми вершин имеют нечетную степень, и, следовательно, нарисовать его



одной линией нельзя.

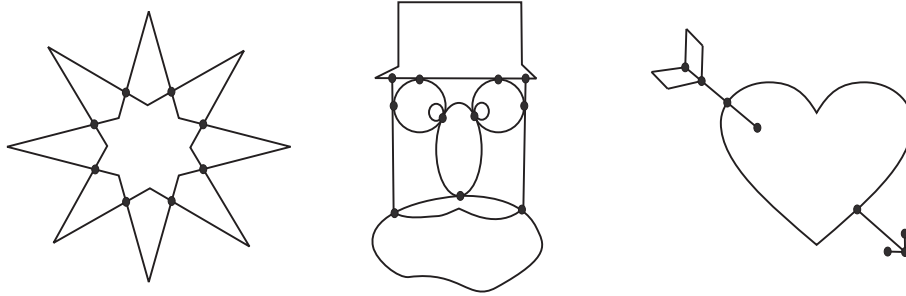


Рис. 44.

### Алгоритм выделения эйлерова цикла

**Идея алгоритма.** Пусть  $G(V, E)$  — неориентированный граф, удовлетворяющий критерию существования эйлерова цикла. По ходу алгоритма будем строить эйлеров цикл  $C$ , как последовательность ребер графа. Выбираем вершину и проходим по любому инцидентному ей ребру, удаляя его. Если в некоторый момент мы оказываемся в вершине  $v$  из которой нет выхода, но непройденные ребра еще остались, то переходим в какую-либо неизolated вершину  $u$  и формируем цикл, который начинается в этой вершине. Новый цикл объединяем с полученным ранее. Если ситуация повторяется (т.е. мы снова оказываемся в вершине из которой нет выхода), то формируем еще один цикл на оставшихся вершинах и объединяем с полученным ранее.

**Шаг 1.** Начиная обход из произвольной вершины  $u$ , строим некоторый цикл

$$u, e_1, u_1, e_2, u_2, \dots, u_{k-1}, e_k, u.$$

Все ребра, вошедшие в цикл, удаляем из графа:

$$E := E \setminus \{e_1, e_2, \dots, e_k\}, \quad V_C := \{u, u_1, u_2, \dots, u_{k-1}\}, \quad C : e_1, e_2, \dots, e_k.$$

**Шаг 2.** Если  $E = \emptyset$ , то найденный цикл  $C$  является эйлеровым и алгоритм заканчивает работу. Иначе, находим некоторую неизolated вершину  $u' \in V_C$  (т.е. вершину, инцидентную ребру, вошедшему в цикл  $C$ ) и строим новый цикл

$$u', e'_1, u'_1, e'_2, u'_2, \dots, u'_{k'-1}, e'_{k'}, u'.$$

**Шаг 3.** Ребра цикла

$$u', e'_1, u'_1, e'_2, u'_2, \dots, u'_{k'-1}, e'_{k'}, u'$$

удаляем из графа и включаем в цикл  $C$ :

$$E := E \setminus \{e'_1, e'_2, \dots, e'_{k'}\}, \quad V_C := \{u, u_1, u_2, \dots, u_{k-1}\} \cup \{u'_1, u'_2, \dots, u'_{k'-1}\},$$

$$C : e_1, e_2, \dots, e_i, \underbrace{e'_1, e'_2, \dots, e'_{k'}}_{}, e_{i+1}, \dots, e_k.$$

Перенумеровываем ребра цикла  $C$ :

$$C : e_1, e_2, \dots, e_i, e_{i+1}, e_{i+2}, \dots, e_{i+k'}, e_{i+k'+1}, \dots, e_{k+k'}.$$

Переходим на Шаг 2.

✓ **ПРИМЕР 8.6.** Найти эйлеров цикл в графе (рис. 45).

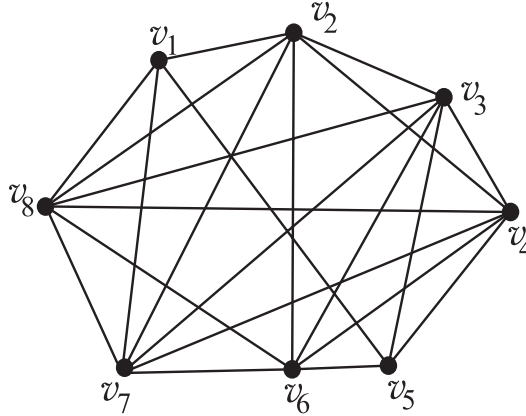
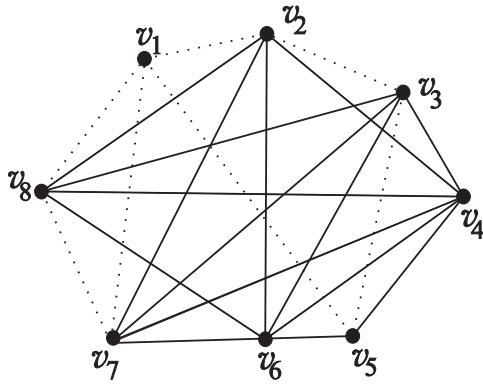
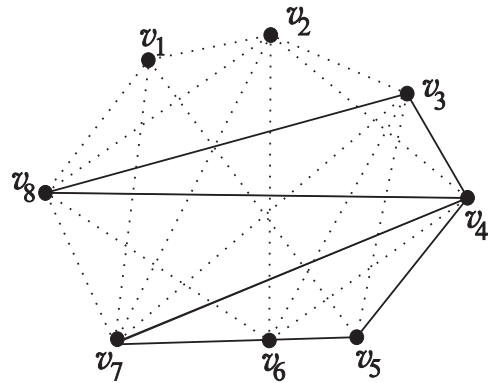


Рис. 45.

**Решение.** Начинаем обход из вершины  $v_1$ , пройденные ребра удаляем (рис. 46 а)):  $(v_1, v_2, v_3, v_5, v_1, v_7, v_8, v_1)$ . Из первой вершины выхода больше нет, перейдем в вершину  $v_2$  (рис. 46б)):  $(v_2, v_4, v_6, v_2, v_7, v_3, v_6, v_8, v_2)$ .



а)



б)

Рис. 46.

Из оставшихся вершин выбираем  $v_3$ :  $(v_3, v_4, v_5, v_6, v_7, v_4, v_8, v_3)$ . Объединим первые два цикла, для этого впишем второй цикл в то место, где встречается вершина  $v_2$  в первом:  $(v_1, \underbrace{v_2}_{}, v_3, v_5, v_1, v_7, v_8, v_1)$  и  $(v_2, v_4, v_6, v_2, v_7, v_3, v_6, v_8, v_2)$

$$\Rightarrow (v_1, v_2, v_4, v_6, v_2, v_7, v_3, v_6, v_8, v_2, v_3, v_5, v_1, v_7, v_8, v_1).$$

К полученному циклу добавим третий:

$(v_1, v_2, v_4, v_6, v_2, v_7, \underbrace{v_3}, v_6, v_8, v_2, v_3, v_5, v_1, v_7, v_8, v_1)$  и  $(v_3, v_4, v_5, v_6, v_7, v_4, v_6, v_3)$

$\Rightarrow (v_1, v_2, v_4, v_6, v_2, v_7, v_3, v_4, v_5, v_6, v_7, v_4, v_6, v_3, v_6, v_8, v_2, v_3, v_5, v_1, v_7, v_8, v_1)$ .

**Замечание.** Возможна эквивалентная формулировка критерия существования эйлерова цикла. Неориентированный граф содержит эйлеров цикл тогда и только тогда, когда он связан и множество его ребер разбивается на множество простых циклов (такие циклы и строятся в ходе описанного выше алгоритма).

### Алгоритм Флёрри выделения эйлерова цикла

Пусть  $G(V, E)$  — неориентированный граф, удовлетворяющий условию существования эйлерова цикла. Выбираем произвольную вершину и проходим по любому инцидентному ей ребру, удаляя его. Из следующей вершины выбираем путь по мосту только в том случае, если нет пути по циклу (пройденные ребра удаляем).

**Замечание 1.** Если в графе существует эйлерова цепь, то для того чтобы ее найти используем любой из предложенных выше алгоритмов, но начинаем обход не из произвольной вершины, а из вершины нечетной степени.

**Замечание 2.** В ориентированном графе алгоритм применяется с учетом направления дуг.

✓ **ПРИМЕР 8.7.** Найти эйлеров цикл (или цепь) в неориентированном графе, заданном матрицей смежности:

$$A(G) = \begin{pmatrix} 2 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 2 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 2 \end{pmatrix}.$$

**Решение.**

Изобразим граф (рис. 47). Так как в графе ровно две вершины нечетной степени (2 и 3), то он содержит эйлерову цепь:

$(2, 3, 4, 5, 7, 7, 6, 6, 4, 1, 7, 2, 5, 1, 6, 3)$ .

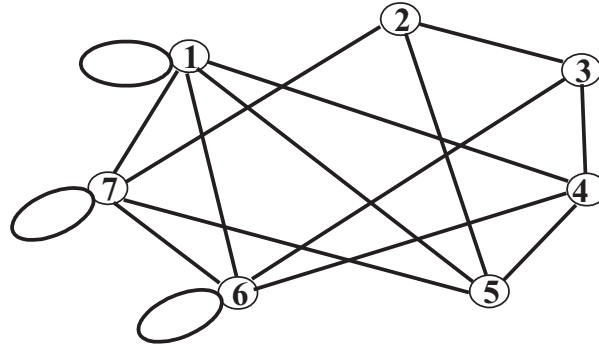


Рис. 47.

✓ **ПРИМЕР 8.8.** Найти эйлеров цикл в орграфе, заданном матрицей смежности:

$$A(\vec{G}) = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

**Решение.**

Проверим, выполняется ли для данного орграфа условие существования эйлерова цикла. Для этого вычислим для каждой вершины степень исхода и степень захода (проверку сильной связности предлагаем провести самостоятельно):

$$\deg_+ v_1 = \sum_{i=1}^7 a_{1i} = 4, \quad \deg_- v_1 = \sum_{i=1}^7 a_{i1} = 4, \quad \deg_+ v_1 = \deg_- v_1 = 4;$$

$$\deg_+ v_2 = \deg_- v_2 = 3; \quad \deg_+ v_3 = \deg_- v_3 = 4;$$

$$\deg_+ v_4 = \deg_- v_4 = 3; \quad \deg_+ v_5 = \deg_- v_5 = 4;$$

$$\deg_+ v_6 = \deg_- v_6 = 3; \quad \deg_+ v_7 = \deg_- v_7 = 3.$$

Будем строить цикл из вершины  $v_1$ . Изображать граф на рисунке не обязательно, достаточно вычеркивать пройденную дугу из матрицы смежности. Применим алгоритм Флёрри. Итак:

$(v_1, v_2, v_3, v_1, v_4, v_1, v_5, v_1, v_7, v_5, v_7, v_2, v_4, v_3, v_4, v_5, v_2, v_5, v_3, v_6, v_6, v_3, v_7, v_6, v_1)$ .

## УПРАЖНЕНИЯ

1. В графах, изображенных на рис. 48, найти эйлеров цикл или цепь.

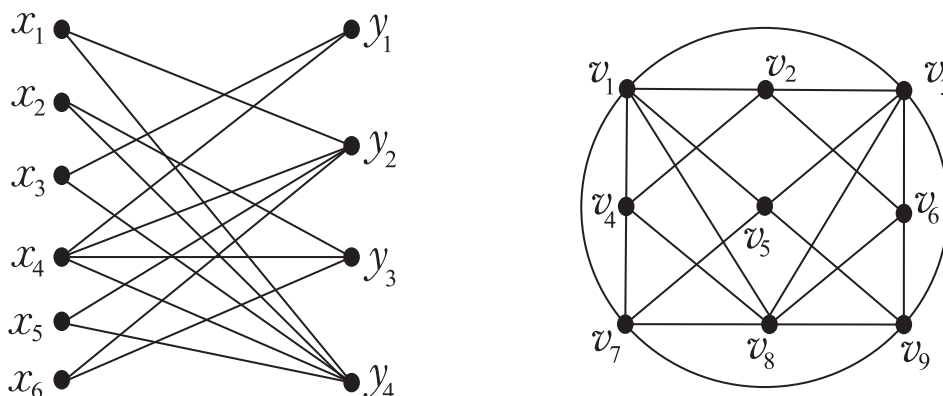


Рис. 48.

2. В графе, заданном матрицей смежности  $A(G)$ , выполнить следующие действия:

- а) поиск в ширину из вершины  $x_1$ ; б) поиск в глубину из вершины  $x_1$ ;
- в) поиск в ширину из множества вершин  $\{x_1, x_3\}$ .

$$A(G) = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

3. В орграфе, заданном матрицей смежности  $A(\vec{G})$ , выполнить поиск в ширину и поиск в глубину из первой вершины.

$$A(\vec{G}) = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}.$$

## 9. Кратчайшие остовы в нагруженном графе


Пусть  $G(V, E)$  — нагруженный неориентированный граф с  $n$  вершинами, заданный матрицей весов  $W(G)$ :

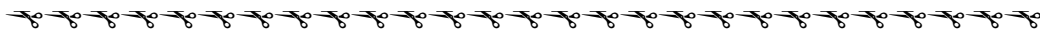
$$w_{ij} = \begin{cases} w(v_i, v_j), & (v_i, v_j) \in E; \\ \infty, & (v_i, v_j) \notin E, \\ \infty, & i = j. \end{cases} \quad i, j = 1, \dots, n;$$

где  $w : E \mapsto \mathbb{R}$  — весовая функция. Как было определено ранее, остов связного графа — это любой его подграф, содержащий все вершины и являющийся деревом.

*Опр.* | Вес остова равен сумме весов всех ребер, составляющих остов.

## 9.1. Алгоритм Краскала построения остова минимального веса (жадный алгоритм)

**Идея алгоритма.** 

Строим остов  $T(V, E_T)$  графа  $G(V, E)$  следующим образом. На начальном этапе  $T$  представляет собой граф, состоящий из  $n$  компонент связности —  $n$  изолированных вершин графа, множество ребер  $E_T$  пусто. На каждом шаге добавляем в  $E_T$  новое ребро из  $E \setminus E_T$ , которое имеет минимальный вес и не образует циклов с ребрами из  $E_T$ . Добавляя ребра, мы уменьшаем число компонент связности графа  $T$ . Алгоритм продолжает работу до тех пор, пока число компонент связности не уменьшится до одной — остова графа  $G$ . 

**Шаг 0.** Определяем граф  $T(V, E_T)$ :  $E_T := \emptyset$ ,

$C_i := \{v_i\}$ ,  $v_i \in V$ ,  $i = 1, \dots, n$  — компоненты связности графа  $T(V, E_T)$ ,

$E^* := E$  — множество "необработанных" ребер графа  $G$ .

**Шаг 1.** Из всех еще ребер  $(u, v) \in E^*$  выбираем ребро минимального веса:

$$(u, v) \in E^*, w(u, v) = \min_{(u, v) \in E^*} \{w(u, v)\}.$$


**Шаг 2.** Если вершины  $u$  и  $v$  принадлежат разным компонентам связности  $u \in C_i$ ,  $v \in C_j$ ,  $i < j$ , то добавляем ребро  $(u, v)$  в множество ребер  $E_T$  и объединяем рассматриваемые компоненты в одну:

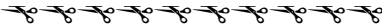
$$E_T := E_T \cup \{(u, v)\}, E^* := E^* \setminus \{(u, v)\}, C_i := C_i \cup C_j, C_j := \emptyset.$$

Если число непустых компонент связности равно 1, то алгоритм заканчивает работу, иначе переходим на шаг 1.

**Шаг 3.** Если вершины  $u$  и  $v$  принадлежат одной и той же компоненте связности  $u \in C_i$ ,  $v \in C_j$ ,  $i = j$ , то удаляем ребро  $(u, v)$  из множества ребер  $E^*$ :  $E^* := E^* \setminus (u, v)$ . Переходим на шаг 1.

## 9.2. Алгоритм Прима построения остова минимального веса (алгоритм ближайшего соседа)

**Идея алгоритма.** 

На каждом шаге алгоритма будем формировать (достраивать) остовное дерево  $T(V_T, E_T)$  следующим образом: к множеству ребер уже построенного дерева добавляем ребро минимального веса один конец которого находится в множестве  $V_T$ , а второй — в множестве  $V \setminus V_T$ . 

**Шаг 0.**  $V_T := \emptyset, \quad E_T := \emptyset$ .

**Шаг 1.** Выбираем в графе произвольную вершину  $u$  и инцидентное ей ребро минимального веса  $(u, v)$ ,

$$(u, v) \in E, \quad w(u, v) = \min_{v \in \Gamma(u)} \{w(u, v)\}, \quad V_T := \{u, v\}, \quad E_T := \{(u, v)\}.$$

**Шаг 2.** Из всех ребер, инцидентных только одной вершине из дерева  $T$ , выбираем ребро минимального веса

$$(u, v) \in E, \quad w(u, v) = \min_{u \in V_T, v \in V \setminus V_T} \{w(u, v)\}.$$

$$V_T := V_T \cup \{v\}, \quad E_T := E_T \cup \{(u, v)\}.$$

Если  $|V_T| = n$ , то алгоритм заканчивает работу, иначе — возвращаемся на начало шага 2.

✓ **ПРИМЕР 9.1.** В графе, изображенном на рис. 49, найти остов минимального веса:

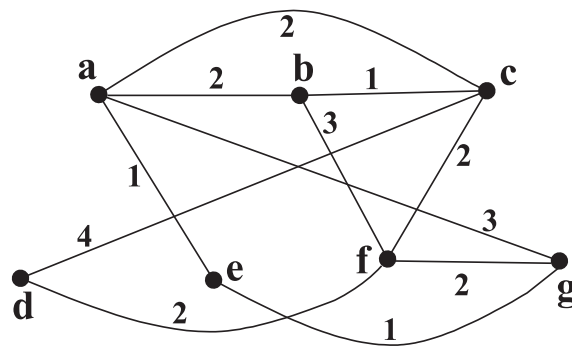


Рис. 49.

**Решение.** Найдем минимальный остов двумя способами.

Поиск минимального остова методом Краскала.

На начальном этапе граф состоит из семи компонент связности:

$$C_1 = \{a\}, \quad C_2 = \{b\}, \quad C_3 = \{c\}, \quad C_4 = \{d\}, \quad C_5 = \{e\}, \quad C_6 = \{f\}, \quad C_7 = \{g\}.$$

Множество  $E^*$  включает все ребра графа, множество  $E_T$  — пусто.  
В качестве ребра с минимальным весом выберем ребро  $(\mathbf{a}, \mathbf{e})$ ,  $w(\mathbf{a}, \mathbf{e}) = 1$ :

$$E_T := \{(\mathbf{a}, \mathbf{e})\},$$

$$C_1 := C_1 \cup C_5 = \{\mathbf{a}, \mathbf{e}\}, \quad C_5 := \emptyset.$$

Следующее ребро с минимальным весом  $(\mathbf{b}, \mathbf{c})$ ,  $w(\mathbf{b}, \mathbf{c}) = 1$ :

$$E_T := \{(\mathbf{a}, \mathbf{e}), (\mathbf{b}, \mathbf{c})\},$$

$$C_2 := C_2 \cup C_3 = \{\mathbf{b}, \mathbf{c}\}, \quad C_3 := \emptyset.$$

Следующее ребро с весом 1 —  $(\mathbf{e}, \mathbf{g})$  (заметим, что вершина  $\mathbf{e}$  на данный момент находится в компоненте  $C_1$ ):

$$E_T := \{(\mathbf{a}, \mathbf{e}), (\mathbf{b}, \mathbf{c}), (\mathbf{e}, \mathbf{g})\},$$

$$C_1 := C_1 \cup C_7 = \{\mathbf{a}, \mathbf{e}, \mathbf{g}\}, \quad C_7 := \emptyset.$$

Добавляем в множество ребер ребро  $(\mathbf{a}, \mathbf{c})$   $w(\mathbf{a}, \mathbf{c}) = 2$ :

$$E_T := \{(\mathbf{a}, \mathbf{e}), (\mathbf{b}, \mathbf{c}), (\mathbf{e}, \mathbf{g}), (\mathbf{a}, \mathbf{c})\},$$

$$C_1 := C_1 \cup C_2 = \{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{e}, \mathbf{g}\}, \quad C_2 := \emptyset.$$

Следующее ребро с весом 2 — ребро  $(\mathbf{a}, \mathbf{b})$ . Обе вершины, инцидентные данному ребру, находятся в одной компоненте связности  $C_1$ , следовательно исключаем данное ребро из множества  $E^*$  и выбираем другое. Ребро  $(\mathbf{c}, \mathbf{f})$  также имеет вес 2 и  $\mathbf{c} \in C_1$ ,  $\mathbf{f} \in C_6$ , следовательно,

$$E_T := \{(\mathbf{a}, \mathbf{e}), (\mathbf{b}, \mathbf{c}), (\mathbf{e}, \mathbf{g}), (\mathbf{a}, \mathbf{b}), (\mathbf{c}, \mathbf{f})\},$$

$$C_1 := C_1 \cup C_6 = \{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{e}, \mathbf{f}, \mathbf{g}\}, \quad C_6 := \emptyset.$$

Минимальный вес у ребра  $(\mathbf{d}, \mathbf{f})$ ,  $w(\mathbf{d}, \mathbf{f}) = 2$ ,  $\mathbf{d} \in C_4$ ,  $\mathbf{f} \in C_1$  следовательно,

$$E_T := \{(\mathbf{a}, \mathbf{e}), (\mathbf{b}, \mathbf{c}), (\mathbf{e}, \mathbf{g}), (\mathbf{a}, \mathbf{b}), (\mathbf{c}, \mathbf{f}), (\mathbf{d}, \mathbf{f})\},$$

$$C_1 := C_1 \cup C_4 = \{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}, \mathbf{g}\}, \quad C_4 := \emptyset.$$

Так как не пустая компонента связности только одна —  $C_1$ , то алгоритм заканчивает работу. Решение задачи продемонстрировано на рис. 50.



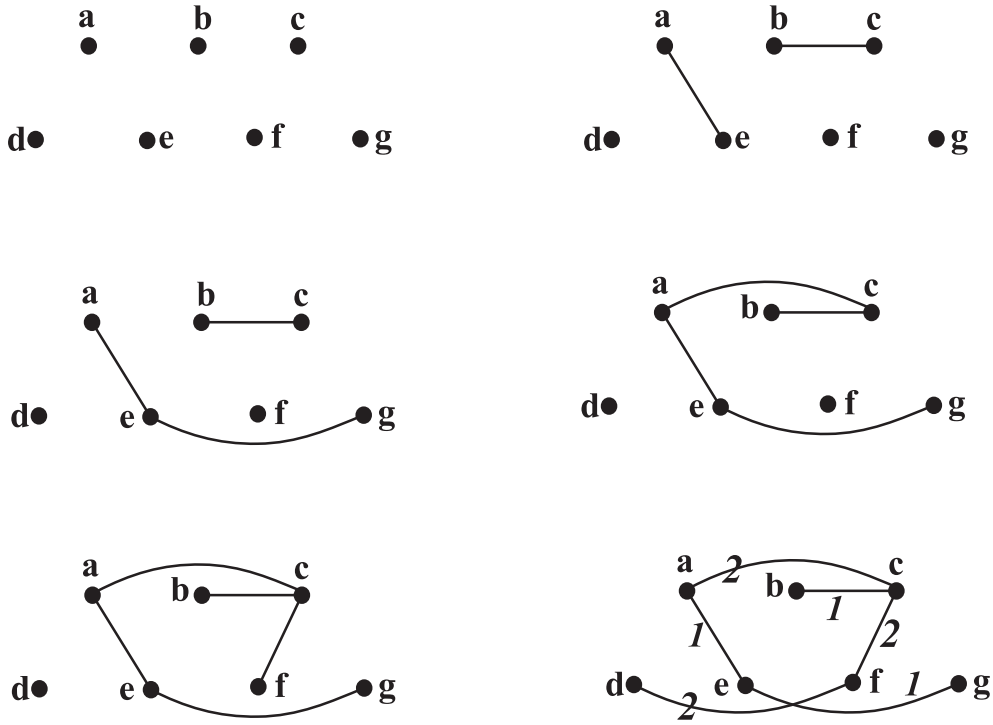


Рис. 50.

Поиск минимального остова методом Прима.

На начальном этапе

$$E_T := \emptyset, \quad V_T := \emptyset.$$

Рассмотрим в качестве начальной вершину **d**. Из ребер, инцидентных вершине **d**, наименьший вес имеет ребро  $(\mathbf{d}, \mathbf{f})$ ,  $w(\mathbf{d}, \mathbf{f}) = 2$ :

$$V_T := \{\mathbf{d}, \mathbf{f}\}, \quad E_T := \{(\mathbf{d}, \mathbf{f})\}.$$

Из ребер, инцидентных одной из вершин множества  $\{\mathbf{d}, \mathbf{f}\}$  наименьший вес 2 у ребер  $(\mathbf{f}, \mathbf{c})$  и  $(\mathbf{f}, \mathbf{g})$ . Выберем одно ребро:  $(\mathbf{f}, \mathbf{c})$ :

$$V_T := \{\mathbf{c}, \mathbf{d}, \mathbf{f}\}, \quad E_T := \{(\mathbf{d}, \mathbf{f}), (\mathbf{f}, \mathbf{c})\}.$$

Следующее ребро определяется однозначно —  $(\mathbf{c}, \mathbf{b})$ ,  $w(\mathbf{c}, \mathbf{b}) = 1$ :

$$V_T := \{\mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{f}\}, \quad E_T := \{(\mathbf{d}, \mathbf{f}), (\mathbf{f}, \mathbf{c}), (\mathbf{c}, \mathbf{b})\}.$$

Из ребер, инцидентных одной из вершин  $\{\mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{f}\}$  с наименьшим весом 2, выберем ребро  $(\mathbf{b}, \mathbf{a})$ :

$$V_T := \{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{f}\}, \quad E_T := \{(\mathbf{d}, \mathbf{f}), (\mathbf{f}, \mathbf{c}), (\mathbf{c}, \mathbf{b}), (\mathbf{b}, \mathbf{a})\}.$$

Следующее ребро —  $(\mathbf{a}, \mathbf{e})$ ,  $w(\mathbf{a}, \mathbf{e}) = 1$ :

$$V_T := \{\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}, \mathbf{e}, \mathbf{f}\}, \quad E_T := \{(\mathbf{d}, \mathbf{f}), (\mathbf{f}, \mathbf{c}), (\mathbf{c}, \mathbf{b}), (\mathbf{b}, \mathbf{a}), (\mathbf{a}, \mathbf{e})\}.$$

Последнее (шестое) ребро —  $(e, g)$ ,  $w(e, g) = 1$ :

$$V_T := \{a, b, c, d, e, f, g\}, \quad E_T := \{(d, f), (f, c), (c, b), (b, a), (a, e), (e, g)\}.$$

Вес минимального остова — 9. Решение задачи продемонстрировано на рис. 51.

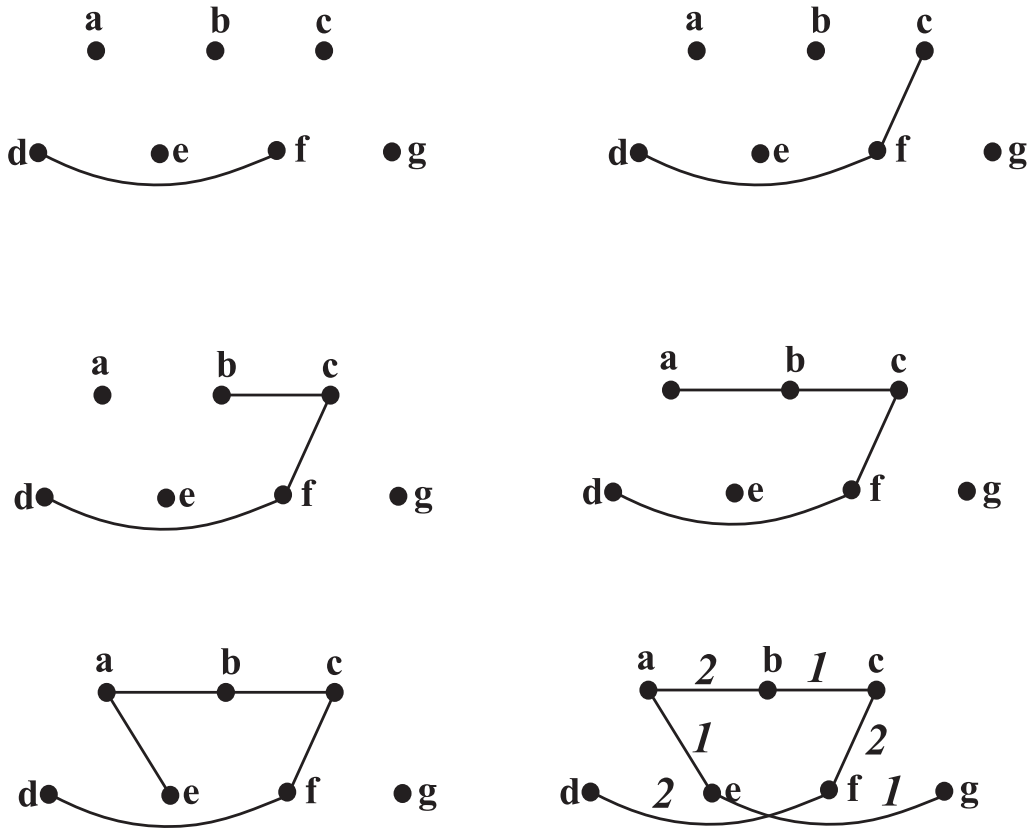


Рис. 51.

✓ **ПРИМЕР 9.2.** В графе, заданном матрицей весов, найти все остовные деревья минимального веса.

$$W(G) = \begin{pmatrix} \infty & 2 & \infty & 3 & \infty & \infty & \infty & \infty & \infty \\ 2 & \infty & 3 & \infty & 3 & \infty & \infty & \infty & \infty \\ \infty & 3 & \infty & \infty & \infty & 1 & \infty & \infty & \infty \\ 3 & \infty & \infty & \infty & 1 & \infty & 2 & \infty & \infty \\ \infty & 3 & \infty & 1 & \infty & 1 & \infty & 4 & \infty \\ \infty & \infty & 1 & \infty & 1 & \infty & \infty & \infty & 4 \\ \infty & \infty & \infty & 2 & \infty & \infty & \infty & 3 & \infty \\ \infty & \infty & \infty & \infty & 4 & \infty & 3 & \infty & 2 \\ \infty & \infty & \infty & \infty & \infty & 4 & \infty & 2 & \infty \end{pmatrix}.$$

**Решение.**

Для решения данной задачи воспользуемся алгоритмом Краскала. На первых шагах выбираем ребра  $(v_4, v_5)$ ,  $(v_3, v_6)$ ,  $(v_5, v_6)$ . Вес этих ребер минимальный — 1, а также данные ребра не образуют цикла. Далее, включаем в остов ребра  $(v_1, v_2)$ ,  $(v_4, v_7)$  и  $(v_8, v_9)$  с весом 2 (рис. 52). В результате мы получаем лес, состоящий из трех компонент:

$$C_1 = \{v_1, v_2\}, \quad C_2 = \{v_4, v_5, v_6, v_7\}, \quad C_3 = \{v_8, v_9\}.$$

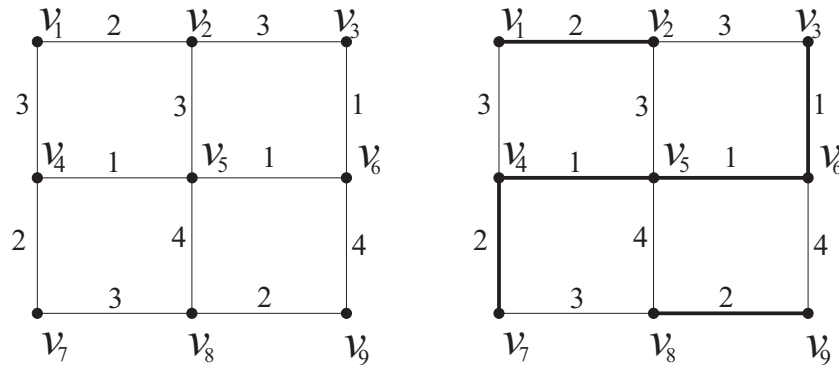


Рис. 52.

Компоненты  $C_1$  и  $C_2$  можно соединить тремя различными способами (рис. 53): а) ребром  $(v_1, v_4)$ ; б) ребром  $(v_2, v_5)$ ; в) ребром  $(v_2, v_3)$ . Вес всех перечисленных ребер одинаковый — 3, и все их можно рассматривать как возможные варианты.

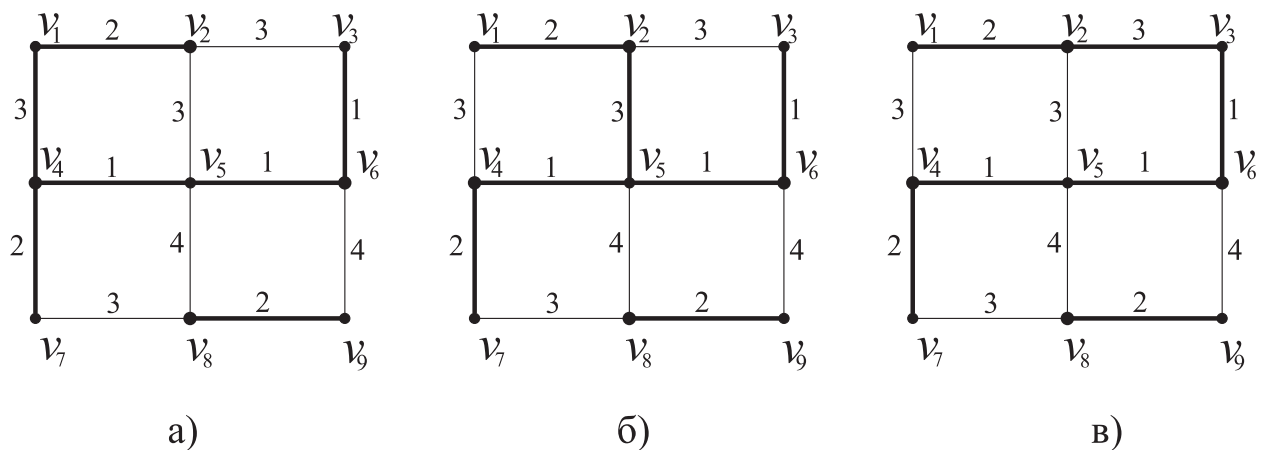


Рис. 53.

Компоненты  $C_1$  и  $C_3$  соединить нельзя. Для компонент  $C_2$  и  $C_3$  в качестве моста может служить одно из ребер  $(v_7, v_8)$ ,  $(v_5, v_8)$  или  $(v_6, v_9)$ , но у ребра  $(v_7, v_8)$  наименьший вес — 3. Таким образом, возможны три варианта:  $T_1$ ,  $T_2$ ,  $T_3$ , остова минимального веса, изображенные на рис. 54.


$$E(T_3) = \{(v_4, v_5), (v_3, v_6), (v_5, v_6), (v_1, v_2), (v_4, v_7)(v_8, v_9), (v_2, v_3), (v_7, v_8)\}.$$

Вес минимального остова равен 15.

## УПРАЖНЕНИЯ

1. Нарисовать все неизоморфные остовные деревья полного неориентированного графа с шестью вершинами.
2. В неориентированных графах, заданных матрицами весов, найти кратчайшие остовы.

$$\text{a) } W(G) = \begin{pmatrix} \infty & 4 & 5 & 2 & 3 & 4 \\ 4 & \infty & 3 & 2 & 1 & 2 \\ 5 & 3 & \infty & 5 & 3 & 1 \\ 2 & 2 & 5 & \infty & 4 & 1 \\ 3 & 1 & 3 & 4 & \infty & 3 \\ 4 & 2 & 1 & 1 & 3 & \infty \end{pmatrix},$$

## 10. Кратчайшие пути в нагруженном графе


*Опр.*

Длина пути в нагруженном орграфе — это сумма весов дуг из которых состоит данный путь. Длина маршрута в неориентированном графе определяется аналогично.

Далее приведены два алгоритма: для кратчайшего пути от заданной вершины до остальных и для поиска кратчайших путей между всеми парами вершин. Алгоритмы применимы как для неориентированных связных, так и для ориентированных сильно связных графов.

### 10.1. Алгоритм Дейкстры поиска кратчайшего пути в нагруженном графе

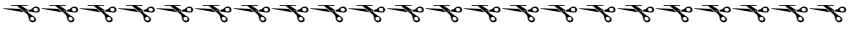
Алгоритм Дейкстры позволяет найти кратчайшие пути от заданной вершины до остальных вершин графа.

**Идея алгоритма.** 

Пусть  $G(V, E)$  — нагруженный сильно связный граф с  $n$  вершинами, заданный матрицей весов  $W(G)$ :

$$w_{ij} = \begin{cases} w(v_i, v_j), & (v_i, v_j) \in E; \\ \infty, & (v_i, v_j) \notin E; \\ 0, & i = j, \end{cases} \quad i, j = 1, \dots, n,$$

где  $w : E \mapsto \mathbb{R}_+$  — неотрицательная весовая функция. На первом этапе предполагаем, что кратчайшие пути из начальной вершины во все вершины из ее окрестности равны соответствующим весам ребер. Кратчайшие пути, ведущие в вершины не смежные с начальной, полагаем равными  $\infty$ . Выбираем ребро с минимальным весом из инцидентных вершине  $u$ , из которой начинается поиск, и переходим по этому ребру в некоторую вершину  $v$ . Очевидно, что из  $u$  в  $v$  не существует пути с меньшим весом, т.е. кратчайший путь из  $u$  в  $v$  найден. Далее, находясь в вершине  $v$ , просматриваем все вершины смежные с ней (кроме  $u$ ). Вычисляем длины путей, ведущих в эти вершины из  $u$  и проходящих через  $v$ . Для вершин, смежных с  $u$  и  $v$  одновременно, возможно найдутся пути, длины которых меньше, чем длины ребер, ведущих в эти вершины непосредственно из  $u$ . Далее, выбираем из окрестности множества вершин  $\{u, v\}$  вершину, в которую ведет самый короткий путь; просматриваем все вершины смежные с ней (кроме  $u$  и  $v$ ), проверяем наличие более коротких путей в уже рассмотренные вершины; выбираем вершину, в которую ведет самый короткий путь и повторяем процесс заново. Фактически, алгоритм является модифицированным поиском в ширину, в котором, для определения "уровня" вершины, необходимо

учитывать вес ребер. 

На каждом шаге алгоритма каждой вершине приписывается метки  $d(v)$ ,  $p(v)$ , где  $d(v)$  — кратчайшее, на текущий момент, расстояние до вершины  $u$ , а  $p(v)$  — предыдущая вершина на кратчайшем пути (эта метка необходима, чтобы восстановить маршрут). Также на каждом шаге будем формировать множество  $V^*$  — вершин, кратчайшие пути до которых уже найдены, и выбирать "ведущую" вершину  $v^*$ .

**Шаг 0.** Полагаем

$$V^* := \{u\}, p(u) := \emptyset, d(u) := 0.$$

Для всех остальных вершин графа:

$$v \in V \setminus V^* \Rightarrow p(v) := \emptyset, d(v) := \infty, v^* := u.$$

**Шаг 1.** Для всех вершин  $v \in V \setminus V^*$  смежных с  $v^*$ :


$$d(v) := \min\{d(v), d(v^*) + w(v^*, v)\},$$

если эта метка изменилась, то меняем и другую метку  $p(v) := v^*$ . После того, как будут пересчитаны все метки, добавляем вершину  $v^*$  в множество  $V^*$ .

**Шаг 2.** Выбираем новую ведущую вершину с минимальной меткой  $d(v)$  т.е.:

$$v^* := \min_{d(v)}(v), V^* := V^* \cup \{v^*\}.$$

Возвращаемся на начало шага 1 и продолжаем процесс до тех пор, пока в множество  $V^*$  не попадут все вершины графа.

 **ПРИМЕР 10.1.** Найти кратчайшие пути из вершины  $f$  в остальные, в орграфе, заданном матрицей весов:

$$W(\vec{G}) = \begin{pmatrix} & \mathbf{a} & \mathbf{b} & \mathbf{c} & \mathbf{d} & \mathbf{e} & \mathbf{f} & \mathbf{g} \\ \begin{pmatrix} 0 \\ 2 \\ \infty \\ \infty \\ 1 \\ \infty \\ 3 \end{pmatrix} & \begin{pmatrix} 1 \\ 0 \\ 2 \\ 2 \\ \infty \\ 6 \\ \infty \end{pmatrix} & \begin{pmatrix} 1 \\ 1 \\ 0 \\ \infty \\ \infty \\ 2 \\ \infty \end{pmatrix} & \begin{pmatrix} \infty \\ 1 \\ 4 \\ 0 \\ 3 \\ 3 \\ \infty \end{pmatrix} & \begin{pmatrix} 1 \\ \infty \\ \infty \\ 1 \\ 0 \\ \infty \\ 1 \end{pmatrix} & \begin{pmatrix} \infty \\ 3 \\ 2 \\ 2 \\ \infty \\ 0 \\ 2 \end{pmatrix} & \begin{pmatrix} 1 \\ \infty \\ \infty \\ \infty \\ 1 \\ 1 \\ 0 \end{pmatrix} \end{pmatrix}.$$

**Решение.** По ходу решения задачи, будем строить "дерево расстояний" данного графа, на котором присутствуют рассматриваемые вершины. Ведущие вершины выделены рамкой, также отмечен номер шага, на котором была выбрана данная вершина.

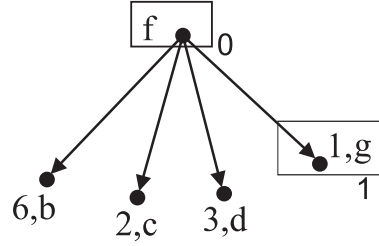
0)  $\boxed{f \bullet}_0$

$$V^* := \{f\}, p(f) := \emptyset, d(f) := 0.$$

$$d(a) := \infty, p(a) := \emptyset; d(b) := \infty, p(b) := \emptyset; d(c) := \infty, p(c) := \emptyset;$$

$$d(d) := \infty, p(d) := \emptyset; d(e) := \infty, p(e) := \emptyset; d(g) := \infty, p(g) := \emptyset.$$

1)

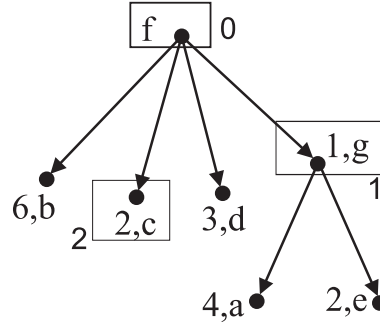


$$d(b) := 6, p(b) := f; d(c) := 2, p(c) := f;$$

$$d(d) := 3, p(d) := f; d(g) := 1, p(g) := f;$$

$$v^* := g, V^* := \{f, g\}.$$

2)



Из вершин, не входящих в  $V^*$ , с  $g$  смежны  $a$  и  $e$ . Пересчитаем их метки:

$$d(a) := \min\{\infty, 1 + 3\} = \min\{\infty, 4\} = 4$$

– метка изменилась, следовательно,

$$p(a) := g;$$

$$d(e) := \min\{\infty, 1 + 1\} = \min\{\infty, 2\} = 2$$

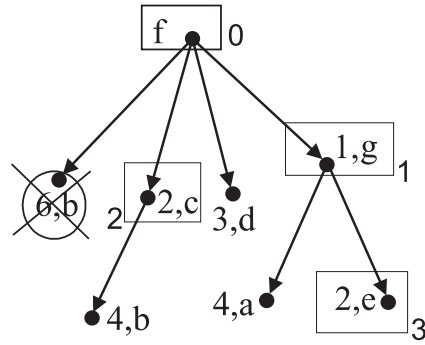
– метка изменилась, следовательно,

$$p(e) := g.$$

Минимальная метка  $d(v) = 2$  у вершин  $c$  и  $e$ , в качестве "ведущей" можно выбрать любую из них.

$$v^* := c, V^* := \{f, g, c\}.$$

3)



С **c** смежны **b**, **d** и **f**, но вершина **f** уже рассмотрена и более нас не интересует. Пересчитаем метки **b** и **d**:

$$d(\mathbf{f}, \mathbf{b}) := \min\{6, 2 + 2\} = \min\{6, 4\} = 4$$

— метка изменилась, следовательно,

$$p(\mathbf{b}) := \mathbf{c}$$

(вершину с большей меткой удаляем из дерева);

$$d(\mathbf{d}) := \min\{3, 2 + 4\} = \min\{3, 6\} = 3$$

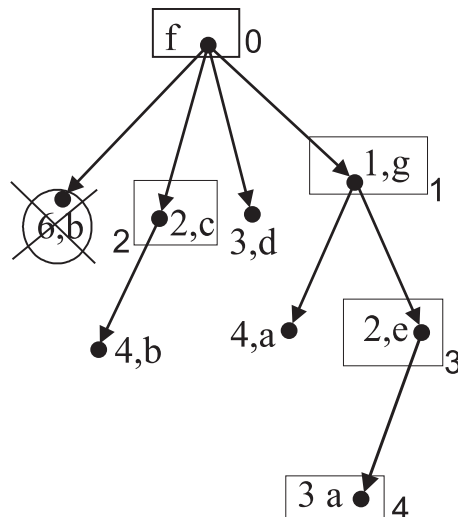
— метка не изменилась, следовательно,

$$p(\mathbf{d}) := \mathbf{f}$$

— оставляем прежнее значение. Минимальная метка  $d(v) = 2$  у вершины **e**:

$$v^* := \mathbf{e}, V^* := \{\mathbf{f}, \mathbf{g}, \mathbf{c}, \mathbf{e}\}.$$

4)





С **e** смежны **a**, **d** и **f**, но  $\mathbf{f} \in V^*$ . Пересчитаем их метки **a** и **d**:

$$d(\mathbf{a}) := \min\{4, 2 + 1\} = \min\{4, 3\} = 3$$

— метка изменилась, следовательно,

$$p(\mathbf{a}) := \mathbf{e}$$

(вершину с большей меткой вычеркиваем из дерева);

$$d(\mathbf{d}) := \min\{3, 2 + 3\} = \min\{3, 5\} = 3$$

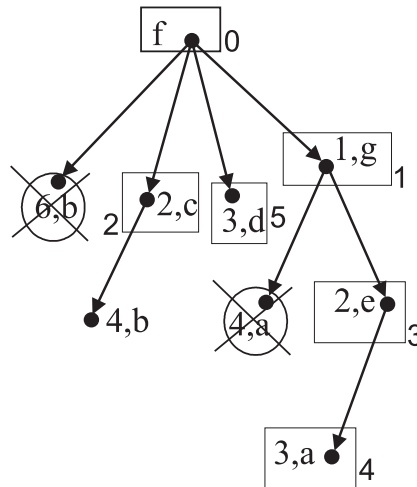
— метка не изменилась, следовательно,

$$p(\mathbf{d}) := \mathbf{f}$$

— прежнее значение. Минимальная метка  $d(v) = 3$  у вершины **a**:

$$v^* := \mathbf{a}, V^* := \{\mathbf{f}, \mathbf{g}, \mathbf{c}, \mathbf{e}, \mathbf{a}\}.$$

5)



Пересчитываем метки вершины **b**:

$$d(\mathbf{b}) := \min\{4, 3 + 1\} = \min\{4, 4\} = 4$$

— метка не изменилась, следовательно,

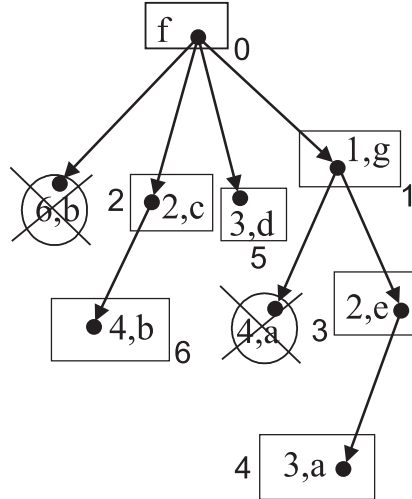
$$p(\mathbf{b}) := \mathbf{c}.$$

Минимальная метка  $d(v) = 3$  у вершины **d**:

$$v^* := \mathbf{d}, V^* := \{\mathbf{f}, \mathbf{g}, \mathbf{c}, \mathbf{e}, \mathbf{a}, \mathbf{d}\}.$$

6) Пересчитываем метку вершины **b**:  $d(\mathbf{b}) := \min\{4, 3+2\} = \min\{4, 5\} = 4$  — метка не изменилась, следовательно,  $p(\mathbf{b}) := \mathbf{c}$ ; минимальная метка  $d(v) = 4$  у вершины **b**:

$$v^* := \mathbf{b}, V^* := \{\mathbf{f}, \mathbf{g}, \mathbf{c}, \mathbf{e}, \mathbf{a}, \mathbf{d}, \mathbf{b}\}.$$



Так как  $V^* = V$ , алгоритм прекращает работу. Итак, кратчайшие пути из **f** в остальные вершины:

$l(\mathbf{f}, \mathbf{a})$  : **f, g, e, a** (длина пути равна 3);

$l(\mathbf{f}, \mathbf{b})$  : **f, c, b** (длина пути равна 4);

$l(\mathbf{f}, \mathbf{c})$  : **f, c** (длина пути равна 2);

$l(\mathbf{f}, \mathbf{d})$  : **f, d** (длина пути равна 3);

$l(\mathbf{f}, \mathbf{e})$  : **f, g, e** (длина пути равна 2);

$l(\mathbf{f}, \mathbf{g})$  : **f, g** (длина пути равна 1).

✓ **ПРИМЕР 10.2.** Найти кратчайшие пути из вершины  $v_1$  в остальные, в орграфе, заданном матрицей весов:

$$W(\vec{G}) = \begin{pmatrix} 0 & 4 & \infty & 2 & 1 & \infty & \infty \\ \infty & 0 & 1 & \infty & 2 & 2 & 2 \\ 2 & \infty & 0 & 3 & \infty & 2 & \infty \\ \infty & 2 & 1 & 0 & 4 & 4 & 1 \\ 5 & \infty & 6 & 4 & 0 & \infty & 1 \\ 2 & 3 & \infty & \infty & 3 & 0 & 1 \\ 2 & \infty & 2 & 4 & \infty & 2 & 0 \end{pmatrix}.$$

**Решение.**

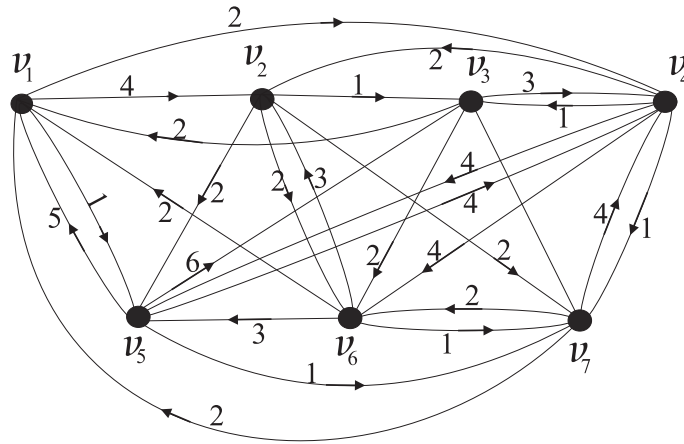


Рис. 55.

В данном примере проиллюстрируем работу алгоритма другим образом. По ходу решения, будем заполнять таблицу, с каждым шагом добавляя новую строку. В первом столбце указывается ведущая вершина, длина кратчайшего пути, ведущего в нее и предшественник на этом пути. Далее, в столбцах, соответствующих всем вершинам, записываются их пересчитанные метки. Если кратчайший путь до вершины уже найден, то в соответствующем ей столбце ставим прочерк. На первом шаге  $v^* = v_1$ ,  $d(v_i)$  равно бесконечности или весу дуги  $w(v_1, v_i)$ . Первая строка таблицы практически совпадает с первой строкой матрицы длин дуг.

$v^*$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$	$d(v_7)$
$v_1, 0$	—	4	$\infty$	2	1	$\infty$	$\infty$

Наименьшее значение имеет метка  $d(v_5)$ , следовательно, на следующем шаге  $v^* := v_5$ . Пересчитываем метки следующим образом: к элементам пятой строки матрицы длин дуг (кроме первого и пятого — в  $v_1, v_5$  уже найдены кратчайшие пути) прибавляем  $d(v_5) = 1$ , сравниваем новое значение метки с предыдущим, и заполняем новую строку в таблице:

$v^*$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$	$d(v_7)$
$v_1, 0$	—	4	$\infty$	2	1	$\infty$	$\infty$
$v_5, 1$	—	4	7	2	—	$\infty$	2

Следующей ведущей вершиной может быть  $v_4$  или  $v_7$  ( $d(v_4) = d(v_7) = 2$ ). Выберем, например,  $v_4$ . Действуя описанным выше способом, заполняем

таблицу:

$v^*, d(v^*), p(v^*)$	$d(v_1)$	$d(v_2)$	$d(v_3)$	$d(v_4)$	$d(v_5)$	$d(v_6)$	$d(v_7)$
$v_1, 0, \emptyset$	—	4	$\infty$	2	1	$\infty$	$\infty$
$v_5, 1, v_1$	—	4	6	2	—	$\infty$	2
$v_4, 2, v_1$	—	4	3	—	—	6	2
$v_7, 2, v_5$	—	4	3	—	—	4	—
$v_3, 3, v_4$	—	4	—	—	—	4	—
$v_2, 4, v_1$	—	—	—	—	—	4	—
$v_6, 4, v_7$	—	—	—	—	—	—	—

Последняя строка играет чисто символическую роль. Теперь восстановим кратчайшие маршруты из первой вершины в остальные. Кратчайшее расстояние от  $v_1$  до  $v_2$  равно 4 (смотрим на самое последнее — "нижнее" значение метки  $d(v_2)$ ). Находим в первом столбце строку, первый элемент которой  $v_2$  и узнаем предшественника —  $v_1$ . Следовательно,

$$l(v_1, v_2) : v_1, v_2 \quad (\text{длина пути равна } 4).$$

Кратчайшее расстояние от  $v_1$  до  $v_3$  равно 3. Находим в первом столбце строку, первый элемент которой  $v_3$  и узнаем предшественника —  $v_4$ . Аналогично найдем предшественника  $v_4$  — это  $v_1$ .

$$l(v_1, v_3) : v_1, v_4, v_3 \quad (\text{длина пути равна } 3).$$

Действуя подобным образом, восстановим маршруты для остальных вершин:

$$l(v_1, v_4) : v_1, v_4 \quad (\text{длина пути равна } 2);$$

$$l(v_1, v_5) : v_1, v_5 \quad (\text{длина пути равна } 1);$$

$$l(v_1, v_6) : v_1, v_5, v_7, v_6 \quad (\text{длина пути равна } 4);$$


$$l(v_1, v_7) : v_1, v_5, v_7 \quad (\text{длина пути равна } 2).$$

## 10.2. Алгоритм Форда-Беллмана поиска кратчайших путей между всеми парами вершин в нагруженном графе

Пусть  $\vec{G}(V, \vec{E})$  нагруженный оргграф, имеющий  $n$  вершин. Оргграф задан матрицей весов  $W$ , порядка  $n$ :

$$w_{ij} = \begin{cases} 0, & \text{если } i = j, \\ w_{ij} & \text{— конечная величина, если существует дуга из } v_i \text{ в } v_j, \\ \infty, & \text{если } v_i \text{ и } v_j \text{ не смежны.} \end{cases}$$

**Замечание.** Будем считать, что в орграфе  $\vec{G}$  отсутствуют контуры отрицательного веса, поскольку двигаясь по такому контуру достаточное число раз, можно получить путь, имеющий вес, меньший любого заданного числа, и тем самым задача нахождения расстояния становится бессмысленной.


**Идея алгоритма.** 

На шаге  $k$  алгоритма рассматриваются все пути из вершины  $v_i$  в вершину  $v_j$ , содержащие  $k$  дуг. Из них выбирается путь, имеющий наименьший вес. Затем его вес сравнивается с весом кратчайшего пути из  $v_i$  в  $v_j$ , содержащим менее  $k$  дуг. Веса кратчайших путей между парами вершин, состоящих не более чем из  $k$  дуг, записываются в матрицу  $D^{(k)}$  порядка  $n$ :

$$d_{ij}^{(k)} = \begin{cases} d_{ij}^{(k-1)} & \text{— длина кратчайшего пути из } v_i \text{ в } v_j \text{ на шаге } k, \\ \infty, & \text{если путь из } v_i \text{ в } v_j \text{ за } k \text{ шагов не найден.} \end{cases}$$

Информация о путях заносится в матрицу "предшественников"  $P^{(k)}$  порядка  $n$ :

$$p_{ij}^{(k)} = \begin{cases} m, & \text{если } v_m \text{ — предшествующая } v_j \text{ вершина} \\ & \text{на кратчайшем пути из } v_i \text{ в } v_j; \\ 0, & \text{если путь из } v_i \text{ в } v_j \text{ не найден;} \\ i, & \text{если } i = j. \end{cases}$$

Алгоритм требует не более  $n - 1$  шагов и заканчивает работу, если  $D^{(k)} = D^{(k-1)}$ , при этом  $P^{(k)} = P^{(k-1)}$ . Итак, на входе алгоритма — матрица весов  $W = (w_{ij})_{n \times n}$ ; на выходе — матрица расстояний  $D = (d_{ij})_{n \times n}$  и матрица  $P = (p_{ij})_{n \times n}$ . 

**Шаг 1.** В качестве матрицы расстояний на первом шаге берём матрицу весов  $W$ :

$$d_{ij}^{(1)} = w_{ij}.$$

Строим матрицу  $P^{(1)}$ :

$$p_{ij}^{(1)} = \begin{cases} 0 & \text{если } w_{ij} = \infty, \\ i, & \text{если } w_{ij} \neq \infty. \end{cases}$$

...

**Шаг  $k$ .** Пересчитываем элементы матрицы расстояний:

$$d_{ij}^{(k)} = \min_{1 \leq m \leq n} \{d_{im}^{(k-1)} + w_{mj}\}.$$

**Замечание 1.** Очевидно, что элементы  $d_{ii} = 0$ ,  $i = 1, \dots, n$ .

**Замечание 2.** При  $m = j$  и  $m = i$  имеют место равенства:

$$d_{im}^{(k-1)} + w_{mj} = d_{ij}^{(k-1)} + w_{jj} = d_{ij}^{(k-1)} + 0 = d_{ij}^{(k-1)},$$

$$d_{ii}^{(k-1)} + w_{ij} = 0 + w_{ij} = d_{ij}^{(1)}.$$

Пересчитываем элементы матрицы  $P^{(k)}$ :

$$p_{ij}^{(k)} = \begin{cases} p_{ij}^{(k-1)}, & \text{если } d_{ij}^{(k)} = d_{ij}^{(k-1)}, \\ m, & \text{если элемент } d_{ij}^{(k)} < d_{ij}^{(k-1)} \text{ и } d_{ij}^{(k)} = d_{im}^{(k-1)} + w_{mj}. \end{cases}$$

Алгоритм заканчивает работу, если  $D^{(k)} = D^{(k-1)}$ .

$$D := D^{(k)}; P := P^{(k)}.$$

**Замечание.** Для того, чтобы восстановить кратчайший путь из вершины  $v_i$  в  $v_j$ , рассмотрим  $i$ -ю строку матрицы  $P$ . Предшествующая  $v_j$  вершина на пути из  $v_i$  в  $v_j$  имеет номер  $m_1 = p_{ij}$ . Следующая вершина имеет номер  $m_2 = p_{im_1}$  (номер предшествующей  $v_{m_1}$  вершина на пути из  $v_i$  в  $v_{m_1}$ ) и т. д., пока номер "предшественника" не будет равен  $i$ :  $v_i, v_{m_k}, \dots, v_{m_2}, v_{m_1}, v_j$ .

✓ **ПРИМЕР 10.3.** Найти расстояния между всеми парами вершин в орграфе, заданном матрицей весов:

$$W(\vec{G}) = \begin{pmatrix} 0 & 2 & \infty & \infty & 1 \\ \infty & 0 & 7 & \infty & \infty \\ \infty & \infty & 0 & 5 & \infty \\ \infty & \infty & \infty & 0 & 2 \\ \infty & 3 & 1 & \infty & 0 \end{pmatrix}.$$

Указать кратчайшие пути из 1-й вершины в 4-ю и из 2-й в 3-ю.

**Решение.** Граф изображен на рис. 56.

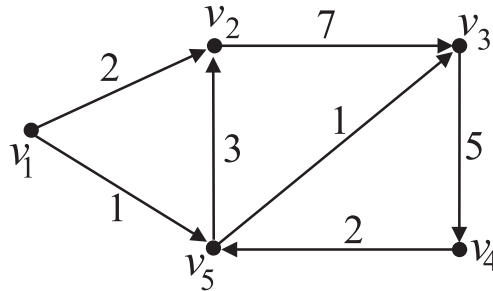


Рис. 56.

1)  $D^{(1)} = W$ , строим матрицу  $P^{(1)}$ :

$$D^{(1)} = \begin{pmatrix} 0 & 2 & \infty & \infty & 1 \\ \infty & 0 & 7 & \infty & \infty \\ \infty & \infty & 0 & 5 & \infty \\ \infty & \infty & \infty & 0 & 2 \\ \infty & 3 & 1 & \infty & 0 \end{pmatrix}, \quad P^{(1)} = \begin{pmatrix} 1 & 1 & 0 & 0 & 1 \\ 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 3 & 3 & 0 \\ 0 & 0 & 0 & 4 & 4 \\ 0 & 5 & 5 & 0 & 5 \end{pmatrix};$$

**2)** Рассматриваем пути, содержащие две дуги. Если находим путь между вершинами  $v_i$  и  $v_j$ , который имеет меньшую длину, чем найденный ранее, то изменяем соответствующие элементы матриц  $D^{(1)}$  и  $P^{(1)}$  (все, кроме диагональных):

$$d_{12}^{(2)} = \min\{d_{11}^{(1)} + w_{12}; d_{12}^{(1)} + w_{22}; d_{13}^{(1)} + w_{32}; d_{14}^{(1)} + w_{42}; d_{15}^{(1)} + w_{52}\} = \\ = \min\{2; 2; \infty; \infty; 4\} = 2;$$

$$d_{13}^{(2)} = \min\{d_{11}^{(1)} + w_{13}; d_{12}^{(1)} + w_{23}; d_{13}^{(1)} + w_{33}; d_{14}^{(1)} + w_{43}; d_{15}^{(1)} + w_{53}\} = \\ = \min\{\infty; 9; \infty; \infty; 2\} = 2 \Rightarrow p_{13}^{(k)} = 5;$$

$$d_{14}^{(2)} = \min\{d_{11}^{(1)} + w_{14}; d_{12}^{(1)} + w_{24}; d_{13}^{(1)} + w_{34}; d_{14}^{(1)} + w_{44}; d_{15}^{(1)} + w_{54}\} = \\ = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{15}^{(2)} = \min\{d_{11}^{(1)} + w_{15}; d_{12}^{(1)} + w_{25}; d_{13}^{(1)} + w_{35}; d_{14}^{(1)} + w_{45}; d_{15}^{(1)} + w_{55}\} = \\ = \min\{1; \infty; \infty; \infty; 1\} = 1;$$

$$d_{21}^{(2)} = \min\{d_{21}^{(1)} + w_{11}; d_{22}^{(1)} + w_{21}; d_{23}^{(1)} + w_{31}; d_{24}^{(1)} + w_{41}; d_{25}^{(1)} + w_{51}\} = \\ = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{23}^{(2)} = \min\{d_{21}^{(1)} + w_{13}; d_{22}^{(1)} + w_{23}; d_{23}^{(1)} + w_{33}; d_{24}^{(1)} + w_{43}; d_{25}^{(1)} + w_{53}\} = \\ = \min\{\infty; 7; 7; \infty; \infty\} = 7;$$

$$d_{24}^{(2)} = \min\{d_{21}^{(1)} + w_{14}; d_{22}^{(1)} + w_{24}; d_{23}^{(1)} + w_{34}; d_{24}^{(1)} + w_{44}; d_{25}^{(1)} + w_{54}\} = \\ = \min\{\infty; \infty; 12; \infty; \infty\} = 12 \Rightarrow p_{24}^{(2)} = 3;$$

$$d_{25}^{(2)} = \min\{d_{21}^{(1)} + w_{15}; d_{22}^{(1)} + w_{25}; d_{23}^{(1)} + w_{35}; d_{24}^{(1)} + w_{45}; d_{25}^{(1)} + w_{55}\} = \\ = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{31}^{(2)} = \min\{d_{31}^{(1)} + w_{11}; d_{32}^{(1)} + w_{21}; d_{33}^{(1)} + w_{31}; d_{34}^{(1)} + w_{41}; d_{35}^{(1)} + w_{51}\} = \\ = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{32}^{(2)} = \min\{d_{31}^{(1)} + w_{12}; d_{32}^{(1)} + w_{22}; d_{33}^{(1)} + w_{32}; d_{34}^{(1)} + w_{42}; d_{35}^{(1)} + w_{52}\} = \\ = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{34}^{(2)} = \min\{d_{31}^{(1)} + w_{14}; d_{32}^{(1)} + w_{24}; d_{33}^{(1)} + w_{34}; d_{34}^{(1)} + w_{44}; d_{35}^{(1)} + w_{54}\} = \\ = \min\{\infty; \infty; 5; 5; \infty\} = 5;$$

$$d_{35}^{(2)} = \min\{d_{31}^{(1)} + w_{15}; d_{32}^{(1)} + w_{25}; d_{33}^{(1)} + w_{35}; d_{34}^{(1)} + w_{45}; d_{35}^{(1)} + w_{55}\} = \\ = \min\{\infty; \infty; \infty; 7; \infty\} = 7 \Rightarrow p_{35}^{(2)} = 4;$$

$$d_{41}^{(2)} = \min\{d_{11}^{(1)} + w_{12}; d_{12}^{(1)} + w_{22}; d_{13}^{(1)} + w_{32}; d_{14}^{(1)} + w_{42}; d_{15}^{(1)} + w_{52}\} = \\ = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{42}^{(2)} = \min\{d_{41}^{(1)} + w_{11}; d_{42}^{(1)} + w_{21}; d_{43}^{(1)} + w_{31}; d_{44}^{(1)} + w_{41}; d_{45}^{(1)} + w_{51}\} = \\ = \min\{\infty; \infty; \infty; \infty; 5\} = 5 \Rightarrow p_{42}^{(2)} = 5;$$

$$d_{43}^{(2)} = \min\{d_{41}^{(1)} + w_{13}; d_{42}^{(1)} + w_{23}; d_{43}^{(1)} + w_{33}; d_{44}^{(1)} + w_{43}; d_{45}^{(1)} + w_{53}\} = \\ = \min\{\infty; \infty; \infty; \infty; 3\} = 3 \Rightarrow p_{43}^{(2)} = 5;$$

$$d_{45}^{(2)} = \min\{d_{41}^{(1)} + w_{15}; d_{42}^{(1)} + w_{25}; d_{43}^{(1)} + w_{35}; d_{44}^{(1)} + w_{45}; d_{45}^{(1)} + w_{55}\} = \\ = \min\{\infty; \infty; \infty; 2; 2\} = 2;$$

$$d_{51}^{(2)} = \min\{d_{51}^{(1)} + w_{11}; d_{52}^{(1)} + w_{21}; d_{53}^{(1)} + w_{31}; d_{54}^{(1)} + w_{41}; d_{55}^{(1)} + w_{51}\} = \\ = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{52}^{(2)} = \min\{d_{51}^{(1)} + w_{12}; d_{52}^{(1)} + w_{22}; d_{53}^{(1)} + w_{32}; d_{54}^{(1)} + w_{42}; d_{55}^{(1)} + w_{52}\} = \\ = \min\{\infty; 3; \infty; \infty; 3\} = 3;$$

$$d_{53}^{(2)} = \min\{d_{51}^{(1)} + w_{13}; d_{52}^{(1)} + w_{23}; d_{53}^{(1)} + w_{33}; d_{54}^{(1)} + w_{43}; d_{55}^{(1)} + w_{53}\} = \\ = \min\{\infty; 10; 1; \infty; 1\} = 1;$$

$$d_{54}^{(2)} = \min\{d_{51}^{(1)} + w_{14}; d_{52}^{(1)} + w_{24}; d_{53}^{(1)} + w_{34}; d_{54}^{(1)} + w_{44}; d_{55}^{(1)} + w_{54}\} = \\ = \min\{\infty; \infty; 6; \infty; \infty\} = 6 \Rightarrow p_{54}^{(2)} = 3.$$

$$D^{(2)} = \begin{pmatrix} 0 & 2 & 2 & \infty & 1 \\ \infty & 0 & 7 & 12 & \infty \\ \infty & \infty & 0 & 5 & 7 \\ \infty & 5 & 3 & 0 & 2 \\ \infty & 3 & 1 & 6 & 0 \end{pmatrix}, \quad P^{(2)} = \begin{pmatrix} 1 & 1 & 5 & 0 & 1 \\ 0 & 2 & 2 & 3 & 0 \\ 0 & 0 & 3 & 3 & 4 \\ 0 & 5 & 5 & 4 & 4 \\ 0 & 5 & 5 & 3 & 5 \end{pmatrix};$$

**3)** Рассматриваем пути, содержащие три дуги и пересчитываем элементы матриц  $D^{(2)}$  и  $P^{(2)}$ :



$$d_{12}^{(3)} = \min\{2; 2; \infty; \infty; 4\} = 2;$$

$$d_{13}^{(3)} = \min\{\infty; 9; 2; \infty; 2\} = 2;$$

$$d_{14}^{(3)} = \min\{\infty; \infty; 7; \infty; \infty\} = 7 \Rightarrow p_{14}^{(3)} = 3;$$

$$d_{15}^{(3)} = \min\{1; \infty; \infty; \infty; 1\} = 1;$$

$$d_{21}^{(3)} = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{23}^{(3)} = \min\{\infty; 7; 7; \infty; \infty\} = 7;$$

$$d_{24}^{(3)} = \min\{\infty; 12; 12; \infty; \infty\} = 12;$$

$$d_{25}^{(3)} = \min\{\infty; \infty; \infty; 14; \infty\} = 14 \Rightarrow p_{25}^{(3)} = 4;$$

$$d_{31}^{(3)} = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{32}^{(3)} = \min\{\infty; \infty; \infty; \infty; 10\} = 10 \Rightarrow p_{32}^{(3)} = 5;$$

$$d_{34}^{(3)} = \min\{\infty; \infty; 5; 5; \infty\} = 5;$$

$$d_{35}^{(3)} = \min\{\infty; \infty; \infty; 7; 7\} = 7;$$

$$d_{41}^{(3)} = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{51}^{(3)} = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{42}^{(3)} = \min\{\infty; 5; \infty; \infty; 5\} = 5;$$

$$d_{52}^{(3)} = \min\{\infty; 3; \infty; \infty; 3\} = 3;$$

$$d_{43}^{(3)} = \min\{\infty; 12; 3; \infty; 3\} = 3;$$

$$d_{53}^{(3)} = \min\{\infty; 10; 1; \infty; 1\} = 1;$$

$$d_{45}^{(3)} = \min\{\infty; \infty; \infty; 2; 2\} = 2;$$

$$d_{54}^{(3)} = \min\{\infty; \infty; 6; 6; \infty\} = 6.$$

$$D^{(3)} = \begin{pmatrix} 0 & 2 & 2 & 7 & 1 \\ \infty & 0 & 7 & 12 & 14 \\ \infty & 10 & 0 & 5 & 7 \\ \infty & 5 & 3 & 0 & 2 \\ \infty & 3 & 1 & 6 & 0 \end{pmatrix}, \quad P^{(3)} = \begin{pmatrix} 1 & 1 & 5 & 3 & 1 \\ 0 & 2 & 2 & 3 & 4 \\ 0 & 5 & 3 & 3 & 4 \\ 0 & 5 & 5 & 4 & 4 \\ 0 & 5 & 5 & 3 & 5 \end{pmatrix};$$

4) Рассматриваем пути, содержащие четыре дуги и пересчитываем элементы матриц  $D^{(3)}$  и  $P^{(3)}$ :

$$d_{12}^{(4)} = \min\{2; 2; \infty; \infty; 4\} = 2;$$

$$d_{21}^{(4)} = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{13}^{(4)} = \min\{\infty; 9; 2; \infty; 2\} = 2;$$

$$d_{23}^{(4)} = \min\{\infty; 7; 7; \infty; 15\} = 7;$$

$$d_{14}^{(4)} = \min\{\infty; \infty; 7; 7; \infty\} = 7;$$

$$d_{24}^{(4)} = \min\{\infty; \infty; 12; 12; \infty\} = 12;$$

$$d_{15}^{(4)} = \min\{1; \infty; \infty; 9; 1\} = 1;$$

$$d_{25}^{(4)} = \min\{\infty; \infty; \infty; 14; 14\} = 14;$$

$$d_{31}^{(4)} = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{41}^{(4)} = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{32}^{(4)} = \min\{\infty; 10; \infty; \infty; 10\} = 10;$$

$$d_{42}^{(4)} = \min\{\infty; 5; \infty; \infty; 5\} = 5;$$

$$d_{34}^{(4)} = \min\{\infty; \infty; 5; 5; \infty\} = 5;$$

$$d_{43}^{(4)} = \min\{\infty; 12; 3; \infty; 3\} = 3;$$

$$d_{35}^{(4)} = \min\{\infty; \infty; \infty; 7; 7\} = 7;$$

$$d_{45}^{(4)} = \min\{\infty; \infty; \infty; 2; 2\} = 2;$$

$$d_{51}^{(4)} = \min\{\infty; \infty; \infty; \infty; \infty\} = \infty;$$

$$d_{53}^{(4)} = \min\{\infty; 10; 1; \infty; 1\} = 1;$$

$$d_{52}^{(4)} = \min\{\infty; 3; \infty; \infty; 3\} = 3;$$

$$d_{54}^{(4)} = \min\{\infty; \infty; 6; 6; \infty\} = 6.$$

$$D^{(4)} = \begin{pmatrix} 1 & 1 & 5 & 3 & 1 \\ 0 & 2 & 2 & 3 & 4 \\ 0 & 5 & 3 & 3 & 4 \\ 0 & 5 & 5 & 4 & 4 \\ 0 & 5 & 5 & 3 & 0 \end{pmatrix}, \quad D^{(3)} = D^{(4)}.$$

$$\text{Алгоритм заканчивает работу, } P := \begin{pmatrix} 1 & 1 & 5 & 3 & 1 \\ 0 & 2 & 2 & 3 & 4 \\ 0 & 5 & 3 & 3 & 4 \\ 0 & 5 & 5 & 4 & 4 \\ 0 & 5 & 5 & 3 & 5 \end{pmatrix}.$$

Восстановим по матрице  $P$  кратчайшие пути из вершины  $v_1$  в  $v_4$  и из  $v_2$  в  $v_3$ :

$$l(v_1, v_4) : p_{14} = 3, p_{13} = 5, p_{15} = 1 \Rightarrow$$

искомый путь:  $v_1, v_5, v_3, v_4$ , длины  $d_{14} = 7$ ;

$$l(v_2, v_3) : p_{23} = 2, \Rightarrow \text{искомый путь: } v_2, v_3, \text{ длины } d_{23} = 7.$$

**Замечание.** Элемент  $d_{ij}^{(k)}$  является минимальным элементом в вектор-строке, полученной сложением  $i$ -й строки матрицы  $D^{(k-1)}$  и  $j$ -го столбца матрицы весов  $W$ . Следовательно, для вычислений элементов матрицы  $D^{(k)}$  можно использовать следующий метод: составляем  $n$  вспомогательных матриц  $W_i$ , для этого к каждому столбцу матрицы  $W$  прибавляем  $\left(D^{(k-1)}\right)_i^T$  (транспонированную  $i$ -ю строку матрицы  $D^{(k-1)}$ ); в каждом столбце полученной матрицы  $W_i$  ищем минимальный элемент. Вектор-строка из этих минимальных элементов равна  $i$ -й строке матрицы  $D^{(k)}$ . В матрице  $P^{(k)}$  заменяем только элементы соответствующие измененным в  $D^{(k)}$ . При этом,  $p_{ij}^{(k)}$  равен номеру строки в матрице  $W_i$  содержащей минимальный элемент в  $j$ -м столбце. Применение этого метода продемонстрировано на следующем примере.

✓ **ПРИМЕР 10.4.** Найти расстояния между всеми парами вершин в орграфе, заданном матрицей весов:

$$W(\vec{G}) = \begin{pmatrix} 0 & 3 & 7 & 4 & 2 \\ 1 & 0 & 2 & 5 & \infty \\ 2 & 2 & 0 & \infty & 3 \\ \infty & 3 & \infty & 0 & 2 \\ 1 & 3 & 1 & \infty & 0 \end{pmatrix}.$$

**Решение.** Граф изображен на рис. 57.

1)  $D^{(1)} = W$ , строим матрицу  $P^{(1)}$ :

$$D^{(1)} = \begin{pmatrix} 0 & 3 & 7 & 4 & 2 \\ 1 & 0 & 2 & 5 & \infty \\ 2 & 2 & 0 & \infty & 3 \\ \infty & 3 & \infty & 0 & 2 \\ 1 & 3 & 1 & \infty & 0 \end{pmatrix}, \quad P^{(1)} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 0 \\ 3 & 3 & 3 & 0 & 3 \\ 0 & 4 & 0 & 4 & 4 \\ 5 & 5 & 5 & 0 & 5 \end{pmatrix};$$

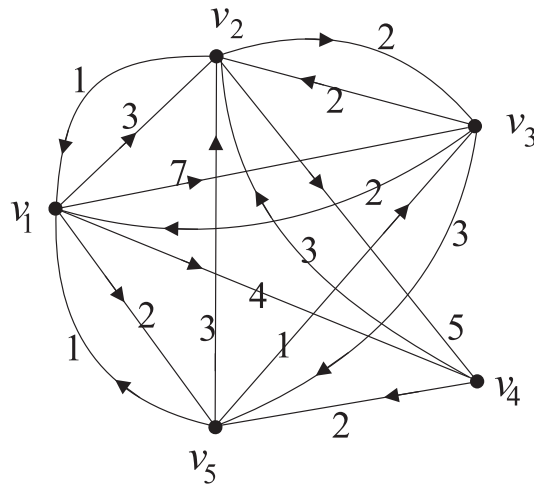


Рис. 57.

2) Вычисляем элементы матриц  $W_i$ :

$$\begin{pmatrix} (D^{(1)})_1^T \\ 0 \\ 3 \\ 7 \\ 4 \\ 2 \end{pmatrix}, \quad \begin{pmatrix} (D^{(1)})_2^T \\ 1 \\ 0 \\ 2 \\ 5 \\ \infty \end{pmatrix}, \quad \begin{pmatrix} (D^{(1)})_3^T \\ 2 \\ 2 \\ 0 \\ \infty \\ 3 \end{pmatrix}, \quad \begin{pmatrix} (D^{(1)})_4^T \\ \infty \\ 3 \\ \infty \\ 0 \\ 2 \end{pmatrix}, \quad \begin{pmatrix} (D^{(1)})_5^T \\ 1 \\ 3 \\ 1 \\ \infty \\ 0 \end{pmatrix}, \quad W = \begin{pmatrix} 0 & 3 & 7 & 4 & 2 \\ 1 & 0 & 2 & 5 & \infty \\ 2 & 2 & 0 & \infty & 3 \\ \infty & 3 & \infty & 0 & 2 \\ 1 & 3 & 1 & \infty & 0 \end{pmatrix}.$$

$$\begin{aligned}
W_1 &= \begin{pmatrix} \mathbf{0} & \mathbf{3} & 7 & 4 & \mathbf{2} \\ 4 & \mathbf{3} & 5 & 8 & \infty \\ 9 & 9 & 7 & \infty & 10 \\ \infty & 7 & \infty & 4 & 6 \\ 3 & 5 & \mathbf{3} & \infty & \mathbf{2} \end{pmatrix}, \quad W_2 = \begin{pmatrix} \mathbf{1} & 4 & 8 & \mathbf{5} & \mathbf{3} \\ \mathbf{1} & \mathbf{0} & \mathbf{2} & \mathbf{5} & \infty \\ 4 & 4 & \mathbf{2} & \infty & 5 \\ \infty & 8 & \infty & \mathbf{5} & 7 \\ \infty & \infty & \infty & \infty & \infty \end{pmatrix}, \\
W_3 &= \begin{pmatrix} \mathbf{2} & 5 & 9 & \mathbf{6} & 4 \\ 3 & \mathbf{2} & 4 & 7 & \infty \\ \mathbf{2} & \mathbf{2} & \mathbf{0} & \infty & \mathbf{3} \\ \infty & \infty & \infty & \infty & \infty \\ 4 & 6 & 4 & \infty & \mathbf{3} \end{pmatrix}, \quad W_4 = \begin{pmatrix} \infty & \infty & \infty & \infty & \infty \\ 4 & \mathbf{3} & 5 & 8 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ \infty & \mathbf{3} & \infty & \mathbf{0} & \mathbf{2} \\ \mathbf{3} & 5 & \mathbf{3} & \infty & \mathbf{2} \end{pmatrix}, \\
W_5 &= \begin{pmatrix} \mathbf{1} & 4 & 8 & \mathbf{5} & \mathbf{3} \\ 3 & \mathbf{3} & 5 & 8 & \infty \\ \infty & \mathbf{3} & \mathbf{1} & \infty & 4 \\ \infty & \infty & \infty & \infty & \infty \\ \mathbf{1} & \mathbf{3} & \mathbf{1} & \infty & \mathbf{0} \end{pmatrix}.
\end{aligned}$$

В матрицах  $W_i$  выделены минимальные элементы. Составляем матрицы  $D^{(2)}$  и  $P^{(2)}$  (измененные элементы матрицы  $D^{(2)}$  подчеркнуты):

$$D^{(2)} = \begin{pmatrix} 0 & 3 & \underline{3} & 4 & 2 \\ 1 & 0 & 2 & 5 & \underline{3} \\ 2 & 2 & 0 & \underline{6} & 3 \\ \underline{3} & 3 & \underline{3} & 0 & 2 \\ 1 & 3 & 1 & \underline{5} & 0 \end{pmatrix}, \Rightarrow P^{(2)} = \begin{pmatrix} 1 & 1 & 5 & 1 & 1 \\ 2 & 2 & 2 & 2 & 1 \\ 3 & 3 & 3 & 1 & 3 \\ 2 & 4 & 5 & 4 & 4 \\ 5 & 5 & 5 & 1 & 5 \end{pmatrix}.$$

3) Вычисляем элементы матриц  $W_i$ :

$$\begin{pmatrix} 0 \\ 3 \\ 3 \\ 4 \\ 2 \end{pmatrix}_1^T, \quad \begin{pmatrix} 1 \\ 0 \\ 2 \\ 5 \\ 3 \end{pmatrix}_2^T, \quad \begin{pmatrix} 2 \\ 2 \\ 0 \\ 6 \\ 3 \end{pmatrix}_3^T, \quad \begin{pmatrix} 4 \\ 3 \\ 3 \\ 0 \\ 2 \end{pmatrix}_4^T, \quad \begin{pmatrix} 1 \\ 3 \\ 1 \\ 5 \\ 0 \end{pmatrix}_5^T, \quad W = \begin{pmatrix} 0 & 3 & 7 & 4 & 2 \\ 1 & 0 & 2 & 5 & \infty \\ 2 & 2 & 0 & \infty & 3 \\ \infty & 3 & \infty & 0 & 2 \\ 1 & 3 & 1 & \infty & 0 \end{pmatrix}.$$

$$\begin{aligned}
W_1 &= \begin{pmatrix} \mathbf{0} & \mathbf{3} & 7 & 4 & \mathbf{2} \\ 4 & \mathbf{3} & 5 & 8 & \infty \\ 5 & 5 & \mathbf{3} & \infty & 6 \\ \infty & 7 & \infty & 4 & 6 \\ 3 & 5 & \mathbf{3} & \infty & \mathbf{2} \end{pmatrix}, \quad W_2 = \begin{pmatrix} \mathbf{1} & 4 & 8 & \mathbf{5} & \mathbf{3} \\ \mathbf{1} & \mathbf{0} & \mathbf{2} & \mathbf{5} & \infty \\ 4 & 4 & \mathbf{2} & \infty & 5 \\ \infty & 8 & \infty & \mathbf{5} & 7 \\ 4 & 6 & 4 & \infty & \mathbf{3} \end{pmatrix}, \\
W_3 &= \begin{pmatrix} \mathbf{2} & 5 & 9 & \mathbf{6} & 4 \\ 3 & \mathbf{2} & 4 & 7 & \infty \\ \mathbf{2} & \mathbf{2} & \mathbf{0} & \infty & \mathbf{3} \\ \infty & 9 & \infty & \mathbf{6} & 8 \\ 4 & 6 & 4 & \infty & \mathbf{3} \end{pmatrix}, \quad W_4 = \begin{pmatrix} 4 & 7 & 11 & 8 & 6 \\ 4 & \mathbf{3} & 5 & 8 & \infty \\ 5 & 5 & \mathbf{3} & \infty & 6 \\ \infty & \mathbf{3} & \infty & \mathbf{0} & \mathbf{2} \\ \mathbf{3} & 5 & \mathbf{3} & \infty & \mathbf{2} \end{pmatrix},
\end{aligned}$$

$$W_5 = \begin{pmatrix} \mathbf{1} & 4 & 8 & \mathbf{5} & 3 \\ 4 & \mathbf{3} & 5 & 8 & \infty \\ 3 & \mathbf{3} & \mathbf{1} & \infty & 4 \\ \infty & 8 & \infty & \mathbf{5} & 7 \\ \mathbf{1} & \mathbf{3} & \mathbf{1} & \infty & \mathbf{0} \end{pmatrix} \Rightarrow D^{(3)} = \begin{pmatrix} 0 & 3 & 3 & 4 & 2 \\ 1 & 0 & 2 & 5 & 3 \\ 2 & 2 & 0 & 6 & 3 \\ 3 & 3 & 3 & 0 & 2 \\ 1 & 3 & 1 & 5 & 0 \end{pmatrix}.$$

Так как  $D^{(3)} = D^{(2)}$ , алгоритм заканчивает работу.

$$D = \begin{pmatrix} 0 & 3 & 3 & 4 & 2 \\ 1 & 0 & 2 & 5 & 3 \\ 2 & 2 & 0 & 6 & 3 \\ 3 & 3 & 3 & 0 & 2 \\ 1 & 3 & 1 & 5 & 0 \end{pmatrix}, \quad P = \begin{pmatrix} 1 & 1 & 5 & 1 & 1 \\ 2 & 2 & 2 & 2 & 1 \\ 3 & 3 & 3 & 1 & 3 \\ 2 & 4 & 5 & 4 & 4 \\ 5 & 5 & 5 & 1 & 5 \end{pmatrix}.$$

В орграфе имеется шесть кратчайших путей, состоящих из более чем одного ребра (им соответствуют элементы матрицы  $D$  отличные от соответствующих элементов матрицы  $W$ ):

$$l(v_1, v_3) : v_1, v_5, v_3; \quad l(v_2, v_5) : v_2, v_1, v_5; \quad l(v_3, v_4) : v_3, v_1, v_4;$$

$$l(v_4, v_1) : v_4, v_2, v_1; \quad l(v_4, v_3) : v_4, v_5, v_3; \quad l(v_5, v_4) : v_5, v_1, v_4.$$

## УПРАЖНЕНИЯ

**1. а)** В неориентированном графе, заданном матрицей весов  $W(G_1)$ , найти длину кратчайшего пути из первой вершины в остальные.

**б)** В ориентированном графе, заданном матрицей весов, найти длину кратчайшего пути из первой вершины в остальные.

$$W(G_1) = \begin{pmatrix} 0 & \infty & \infty & 2 & 3 & 4 \\ \infty & 0 & 3 & \infty & 1 & 2 \\ \infty & 3 & 0 & 5 & \infty & 1 \\ 2 & \infty & 5 & 0 & 4 & 1 \\ 3 & 1 & \infty & 4 & 0 & \infty \\ 4 & 2 & 1 & 1 & \infty & 0 \end{pmatrix}; \quad W(\vec{G}_2) = \begin{pmatrix} 0 & 4 & \infty & 9 & 6 & 5 \\ \infty & 0 & 8 & 4 & \infty & 2 \\ \infty & 3 & 0 & 5 & \infty & 1 \\ 4 & 4 & 5 & 0 & 4 & 1 \\ 5 & 4 & \infty & 4 & 0 & 2 \\ \infty & 2 & 2 & 1 & \infty & 0 \end{pmatrix}.$$

**2.** В ориентированном графе, заданном матрицей весов, найти кратчайшие пути между всеми парами вершин.

$$W(\vec{G}) = \begin{pmatrix} 0 & \infty & \infty & 1 & 8 & \infty \\ 2 & 0 & 3 & \infty & 2 & 3 \\ \infty & 3 & 0 & 5 & \infty & 5 \\ 1 & \infty & 5 & 0 & \infty & 1 \\ 3 & 2 & 5 & 4 & 0 & \infty \\ \infty & \infty & 1 & 5 & \infty & 0 \end{pmatrix}.$$

## 11. Паросочетания

Пусть  $G(V, E)$  — неориентированный граф без петель.

*Опр.*

*Паросочетание* — произвольное подмножество попарно несмежных ребер графа.

**Замечание.** Паросочетанием можно назвать и сам граф, если все его ребра составляют паросочетание (регулярный граф степени 1).

*Опр.*

*Мощность паросочетания* равна числу ребер, составляющих паросочетание.

*Опр.*

*Максимальное паросочетание* — паросочетание, не содержащееся в паросочетании большей мощности.

*Опр.*

*Наибольшее паросочетание* — максимальное паросочетание с наибольшей мощности среди всех максимальных паросочетаний графа.

*Опр.*

*Совершенное паросочетание* — паросочетание, покрывающее все вершины графа (т.е. каждая вершина графа инцидентна какому-либо ребру паросочетания).

В графах, изображенных на рис. 58 выделены максимальные паросочетания. Причем в последнем случае паросочетание является совершенным.

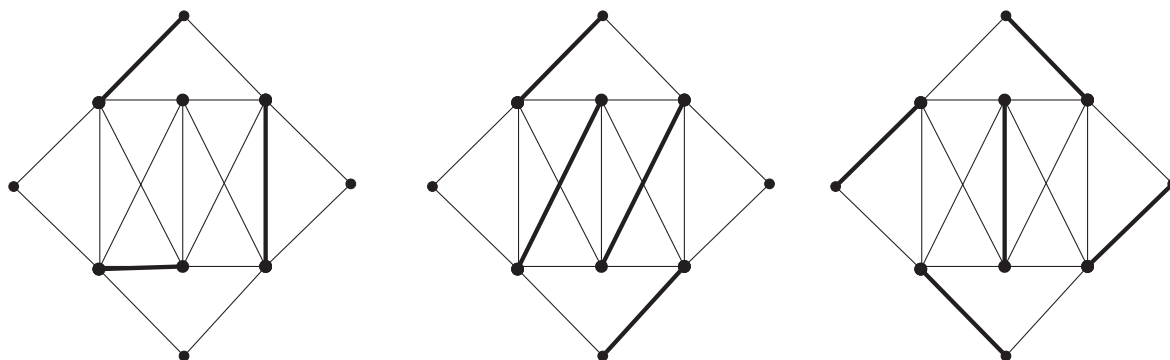


Рис. 58.

*Опр.*

*Двудольный граф* — граф, множество вершин которого состоит из двух непересекающихся подмножеств (долей) попарно несмежных вершин.

**Замечание.** В двудольном графе смежными могут быть только вершины из разных долей.

*Опр.*

*Полный двудольный граф* — двудольный граф, множество ребер которого содержит все возможные ребра.

*Опр.*

*Матрица смежности двудольного графа  $G(X, Y, E)$  где  $X = \{x_1, x_2, \dots, x_m\}$ ,  $Y = \{y_1, y_2, \dots, y_n\}$  — матрица  $A_{XY}(G)$  порядка  $m \times n$ , элементы которой определяются следующим образом:  $a_{ij}$  равен числу ребер, соединяющих вершины  $x_i \in X$  и  $y_j \in Y$ .*

*Опр.*

Матрица весов двудольного нагруженного графа  $G(X, Y, E)$ , где  $X = \{x_1, x_2, \dots, x_m\}$ ,  $Y = \{y_1, y_2, \dots, y_n\}$ , — матрица  $W_{XY}(G)$  порядка  $m \times n$ , элементы которой определяются следующим образом:  $w_{ij}$  равен весу ребра  $(x_i, y_j)$ , если вершины смежны;  $w_{ij} = \infty$ , если вершины не смежны.

**Критерий двудольности графа.** Граф является двудольным тогда и только тогда, когда все его простые циклы имеют четную длину.

**Теорема Холла.** Для того чтобы в двудольном графе  $G(X, Y, E)$  существовало паросочетание покрывающее все вершины множества  $X$  необходимо и достаточно, чтобы для любого подмножества  $A \subseteq X$  выполнялось неравенство  $|A| \leq |\Gamma(A)|$ .

**Следствие 1.** Для того чтобы в двудольном графе существовало совершенное паросочетание необходимо чтобы доли имели одинаковое количество вершин.

**Следствие 2.** Для того чтобы в двудольном графе существовало совершенное паросочетание достаточно, чтобы все его вершины имели одинаковую степень (не равную нулю).

### 11.1. Алгоритм построения наибольшего паросочетания в двудольном графе

Пусть  $G(X, Y, E)$  — двудольный граф, в котором выделено некоторое паросочетание  $P$ . Обозначим  $X_P$  и  $Y_P$  множества вершин, инцидентных ребрам паросочетания  $P$  ( $X_P \subseteq X$  и  $Y_P \subseteq Y$ ).

**Шаг 1.** Строим оргграф  $\vec{G}(X, Y, \vec{E})$  следующим образом: все ребра, входящие в паросочетание  $P$  преобразуем в дуги, направленные из  $Y$  в  $X$ , а все ребра, не входящие в паросочетание  $P$  преобразуем в дуги, направленные из  $X$  в  $Y$ .

**Шаг 2.** В оргграфе  $\vec{G}$  ищем путь из какой-либо вершины  $u \in X \setminus X_P$  в какую-либо вершину  $v \in Y \setminus Y_P$ . Для этого выполняем поиск в ширину из множества вершин  $X \setminus X_P$ . Если в результате поиска ни

одна из вершин множества  $Y \setminus Y_P$  не получила метки, то алгоритм заканчивает работу;  $P$  — наибольшее паросочетание.

**Шаг 3.** Пусть  $v \in Y \setminus Y_P$  некоторая вершина, получившая метку при поиске в ширину. Следовательно в  $\vec{G}$  существует путь  $u, w_1, w_2, \dots, w_k, v$ ,  $u \in X \setminus X_P$ . Преобразуем паросочетание  $P$  и, соответственно, граф  $\vec{G}$  следующим образом. Добавляем в паросочетание  $P$  ребра, соответствующие дугам пути, идущим из вершин множества  $X \setminus X_P$  в вершины множества  $Y \setminus Y_P$ , а ребра, соответствующие дугам пути, идущим из вершин множества  $Y$  в вершины множества  $X$  удаляем:

$$X_P := X_P \cup \{u\}, \quad Y_P := Y_P \cup \{v\},$$

$$\vec{E}_P := \left( \vec{E}_P \setminus \{(w_1, w_2), (w_3, w_4), \dots, (w_{k-1}, w_k)\} \right) \cup \{(u, w_1), (w_2, w_3), \dots, (w_k, v)\}.$$

Мощность паросочетания  $P$  при этом увеличивается на 1. Возвращаемся на шаг 1.



**ПРИМЕР 11.1.** В двудольном графе, заданном матрицей смежности, найти наибольшее паросочетание.

$$A_{XY}(G) = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{pmatrix}.$$

**Решение.**

Построим данный граф (рис. 59). Выделим в нем некоторое максимальное паросочетание  $P$ :

$$X_P = \{x_3, x_5\}, \quad Y_P = \{y_2, y_3\}, \quad P = \{(x_3, y_2), (x_5, y_3)\}.$$

Построим также оргграф  $\vec{G}(X, Y, \vec{E})$  (рис. 59).

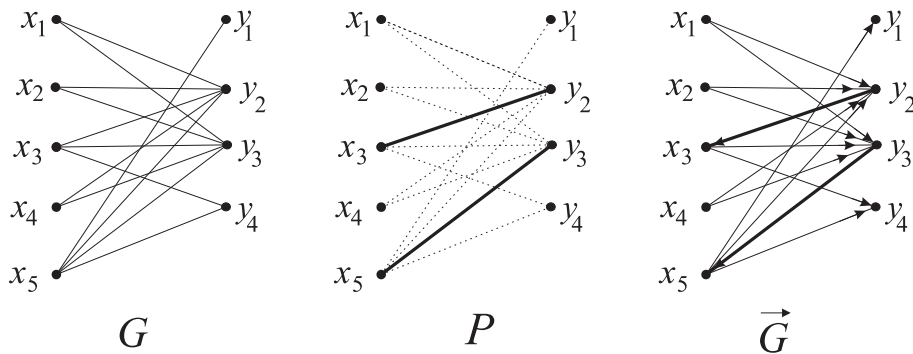


Рис. 59.



Выполним из множества вершин  $\{x_1, x_2, x_4\}$  поиск в ширину (рис. 60). В результате поиска получит метку вершина  $y_4$ :  $x_1, y_2, x_3, y_4$ . В паросочетание добавляем ребра  $(x_1, y_2)$ ,  $(x_3, y_4)$  и удаляем ребро  $(x_3, y_2)$  (рис. 60).

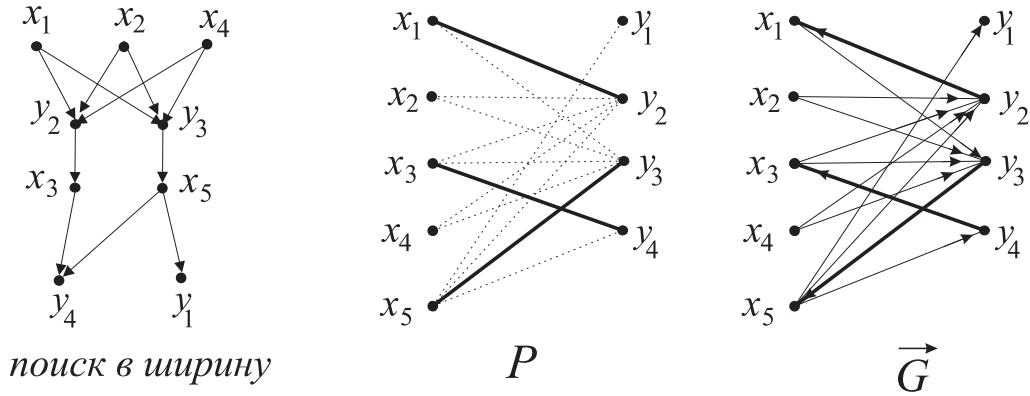


Рис. 60.

$$X_P = \{x_1, x_3, x_5\}, \quad Y_P = \{y_2, y_3, y_4\}, \quad P = \{(x_1, y_2), (x_3, y_4), (x_5, y_3)\}.$$

Выполним поиск в ширину из множества вершин  $\{x_2, x_4\}$ . В результате поиска получит метку вершина  $y_1$ :  $x_4, y_3, x_5, y_1$ . В паросочетание добавляем ребра  $(x_4, y_3)$ ,  $(x_5, y_1)$  и удаляем ребро  $(x_5, y_3)$  (рис. 61).

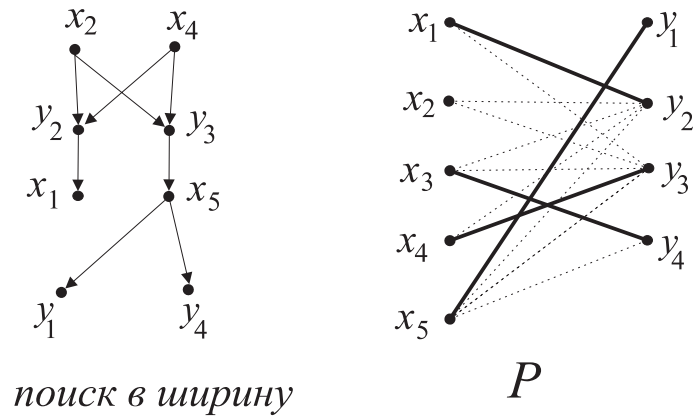


Рис. 61.

$$X_P = \{x_1, x_3, x_4, x_5\}, \quad Y_P = \{y_1, y_2, y_3, y_4\},$$

$$P = \{(x_1, y_2), (x_3, y_4), (x_4, y_3), (x_5, y_1)\}.$$

Так как  $Y_P = Y$ , то алгоритм прекращает работу.  $P$  — наибольшее паросочетание.

## 11.2. Алгоритм построения совершенного паросочетания минимального веса в двудольном нагруженном графе

Пусть  $G(X, Y, E)$ , где  $|X| = |Y| = n$  — полный двудольный взвешенный граф, заданный матрицей весов  $W_{XY}(G)$ . Требуется найти в данном графе паросочетание, вес которого минимален.

**Замечание 1.** Для существования совершенного паросочетания в двудольном графе, необходимо, чтобы доли имели равное количество вершин (см. следствие 1 теоремы Холла). Полнота двудольного графа с долями равной мощности обеспечивает достаточное условие существования совершенного паросочетания (см. следствие 2 теоремы Холла). Алгоритм применим к любому двудольному графу (не обязательно полному), удовлетворяющему теореме Холла.

**Замечание 2.** Так как для каждой вершины в совершенное паросочетание входит только одно инцидентное ей ребро, то решение задачи не изменится, если вес всех ребер инцидентных какой-либо вершине, увеличить (или уменьшить) на одно и то же число. Исходя из этого, можем считать, что веса всех ребер графа неотрицательны.

**Замечание 3.** Если веса всех ребер графа неотрицательны и некоторое совершенное паросочетание состоит из ребер веса 0, то оно является решением задачи.

Перед началом работы алгоритма преобразуем матрицу весов графа (а значит, и сам граф): из каждой строки матрицы  $W_{XY}(G)$  вычтем минимальный элемент этой строки, а затем каждого столбца вычтем минимальный элемент этого столбца. В преобразованном графе будет не менее  $n$  ребер веса 0.

**Шаг 1.** Строим граф  $G_0(X, Y, E_0)$ , где  $E_0$  — множество ребер графа  $G$ , имеющих вес 0. Выделяем в графе  $G_0$  какое-либо максимальное паросочетание  $P$ . Пусть  $X_P$  и  $Y_P$  множества вершин, инцидентных ребрам паросочетания  $P$  ( $X_P \subseteq X$  и  $Y_P \subseteq Y$ ).

**Шаг 2.** Если  $X \setminus X_P = Y \setminus Y_P = \emptyset$ , то алгоритм заканчивает работу,  $P$  — искомое совершенное паросочетание. В противном случае, строим оргграф  $\vec{G}_0(X, Y, \vec{E}_0)$  следующим образом: все ребра, входящие в паросочетание  $P$ , преобразуем в дуги, направленные из  $Y$  в  $X$ , а все ребра, не входящие в паросочетание  $P$ , преобразуем в дуги, направленные из  $X$  в  $Y$ .

**Шаг 3.** Применим к оргграфу  $\vec{G}_0$  поиск в ширину из множества вершин  $X \setminus X_P$ . Если будет найден путь  $u, w_1, w_2, \dots, w_k, v$ , такой, что

$u \in X \setminus X_P$  и  $v \in Y \setminus Y_P$ , то переходим к шагу 4. Иначе — переходим к шагу 6.

**Шаг 4.** Преобразуем оргграф  $\vec{G}_0$ . Заменяем каждую дугу найденного пути на обратную ей:

$$E_P := (E_P \setminus \{(w_1, w_2), (w_3, w_4), \dots, (w_{k-1}, w_k)\}) \cup \{(u, w_1), (w_2, w_3), \dots, (w_k, v)\},$$

$$X_P := X_P \cup \{u\}, \quad Y_P := Y_P \cup \{v\}.$$

При этом, в графе  $G_0$  будет найдено новое максимальное паросочетание  $P$  (из ребер, соответствующих дугам из  $Y_P \cup \{v\}$  в  $X_P \cup \{u\}$ ), мощность которого больше мощности предыдущего на 1.

**Шаг 5.** Если  $X \setminus X_P = Y \setminus Y_P = \emptyset$ , то алгоритм заканчивает работу,  $P$  — искомое совершенное паросочетание. Иначе переходим к шагу 3.

**Шаг 6.** Пусть  $X' \subset X$ ,  $Y' \subset Y$  — подмножества вершин оргграфа  $\vec{G}_0$ , получивших метки в результате поиска в ширину (на шаге 3). Среди ребер  $(u, v)$  графа  $G$  таких, что  $u \in X'$ ,  $v \in Y \setminus Y'$  ищем ребро минимального веса (обозначим этот вес  $m$ ).

**Шаг 7.** Преобразуем веса ребер исходного графа  $G$  следующим образом: из веса каждого ребра инцидентного вершине из множества  $X'$  вычтем  $m$ , а затем к весу каждого ребра, инцидентного вершине из множества  $Y'$ , прибавим  $m$  (вес ребер инцидентных вершинам обоих множеств при этом не изменится — останется равным 0).

**Шаг 8.** Преобразуем оргграф  $\vec{G}_0(X, Y, \vec{E}_0)$ . Добавим появившиеся в результате преобразования на шаге 7 дуги веса 0 из  $X'$  в  $Y \setminus Y'$  и удалим дуги из  $X \setminus X'$  в  $Y'$ , вес которых увеличился. Переходим к шагу 3.

✓ **ПРИМЕР 11.2.** В двудольном графе, заданном матрицей весов, найти совершенное паросочетание минимального веса (проверить выполнение условий теоремы Холла).

$$W_{XY}(G) = \begin{pmatrix} 1 & 4 & \infty & 2 & \infty & 1 \\ 2 & \infty & \infty & 3 & 10 & \infty \\ \infty & 3 & 10 & \infty & 10 & 6 \\ 4 & 8 & \infty & \infty & 12 & 10 \\ 6 & 5 & 6 & \infty & 11 & 6 \\ 10 & 9 & 6 & 6 & 6 & \infty \end{pmatrix}.$$

## Решение.

Убедимся в том, что для любого подмножества вершин выполняется условие теоремы Холла:  $|A| \leq |\Gamma(A)|$ . Минимальная степень вершины в графе равна 3. Следовательно, мощность окрестности любого подмножества вершин не менее 3. Следовательно, для одно-, двух- и трехэлементных подмножеств вершин условие теоремы выполнено. В доле  $X$  вершина степени 3 одна, а в доле  $Y$  таких вершин две. Значит любое множество из четырех вершин содержит несколько вершин, степень которых больше трех (это верно для обеих долей). Следовательно, условие теоремы имеет место для четырехэлементных подмножеств вершин каждой доли. Любое пятиэлементное подмножество множества  $X$  содержит вершины  $x_5$  или  $x_6$ , а любое пятиэлементное подмножество множества  $Y$  содержит вершины  $y_1$  или  $y_2$  (перечисленные вершины имеют степень 5). Следовательно, для пятиэлементных подмножеств условие теоремы также выполняется. Наконец, для самих множеств  $X$  и  $Y$  выполнение неравенства  $|A| \leq |\Gamma(A)|$  равносильно отсутствию изолированных вершин (если бы такие вершины в графе были, то им соответствовали бы строки или столбцы, полностью состоящие из " $\infty$ ").

Преобразуем матрицу смежности графа так, как предлагается в алгоритме. А именно, вычтем из каждой строки, а затем из каждого столбца минимальный элемент:

$$W_{XY}(G) \Rightarrow \begin{pmatrix} 1 & 4 & \infty & 2 & \infty & 1 \\ 2 & \infty & \infty & 3 & 10 & \infty \\ \infty & 3 & 10 & \infty & 10 & 6 \\ 4 & 8 & \infty & \infty & 12 & 10 \\ 6 & 5 & 6 & \infty & 11 & 6 \\ 10 & 9 & 6 & 6 & 6 & \infty \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 3 & \infty & 1 & \infty & 0 \\ 0 & \infty & \infty & 1 & 8 & \infty \\ \infty & 0 & 7 & \infty & 7 & 3 \\ 0 & 4 & \infty & \infty & 8 & 6 \\ 1 & 0 & 1 & \infty & 6 & 1 \\ 4 & 3 & 0 & 0 & 0 & \infty \end{pmatrix}.$$

Построим граф  $G_0(X, Y, E_0)$  (рис. 62), где  $E_0$  — множество ребер графа с весом 0. Выделим в  $G_0$  некоторое максимальное паросочетание  $P$  (рис. 62):

$$X_P = \{x_1, x_3, x_6\}, \quad Y_P = \{y_1, y_2, y_3\}, \quad E_P = \{(x_1, y_1), (x_3, y_2), (x_6, y_3)\}.$$

Построим также оргграф  $\vec{G}_0(X, Y, \vec{E}_0)$  (рис. 62): все ребра, входящие в паросочетание  $P$  преобразуем в дуги, направленные из множества  $Y$  в множество  $X$ , а все ребра, не входящие в паросочетание  $P$  преобразуем в дуги, направленные из множества  $X$  в множество  $Y$ .

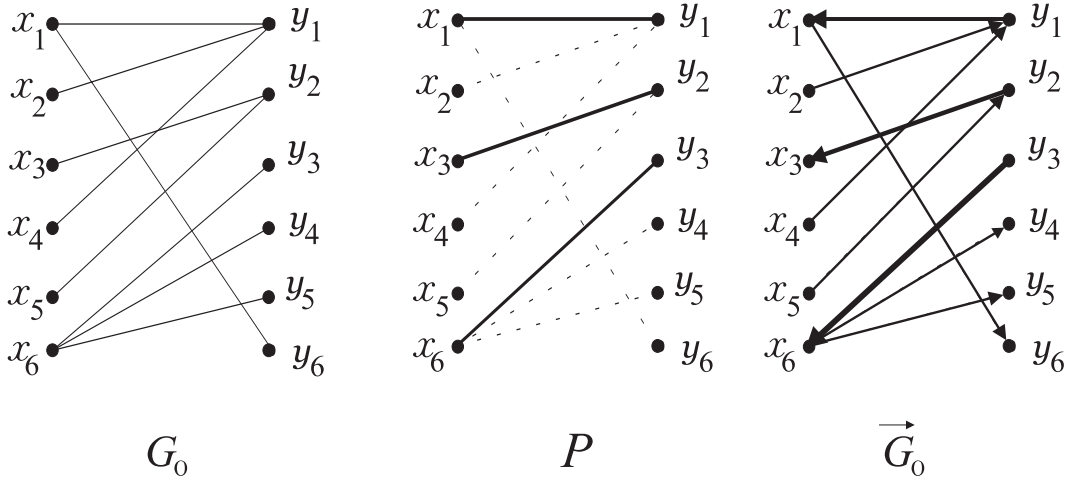


Рис. 62.

Так как  $X \setminus X_P = \{x_2, x_4, x_5\} \neq \emptyset$  и  $Y \setminus Y_P = \{y_4, y_5, y_6\} \neq \emptyset$ , то в графе  $\vec{G}_0$  из множества  $X \setminus X_P$  выполняем поиск в ширину (рис. 63). Наша цель — найти путь в вершины из доли  $Y$ , не инцидентные ребрам паросочетания  $P$ :  $y_4, y_5, y_6$ . В результате находим путь  $x_2, y_1, x_1, y_6$  в вершину  $y_6 \in Y \setminus Y_P$ . В паросочетание добавляем ребра  $(x_2, y_1)$ ,  $(x_1, y_6)$  и удаляем ребро  $(x_1, y_1)$  (рис. 63):

$$X_P = \{x_1, x_2, x_3, x_6\}, \quad Y_P = \{y_1, y_2, y_3, y_6\},$$

$$E_P = \{(x_2, y_1), (x_1, y_6), (x_3, y_2), (x_6, y_3)\}.$$

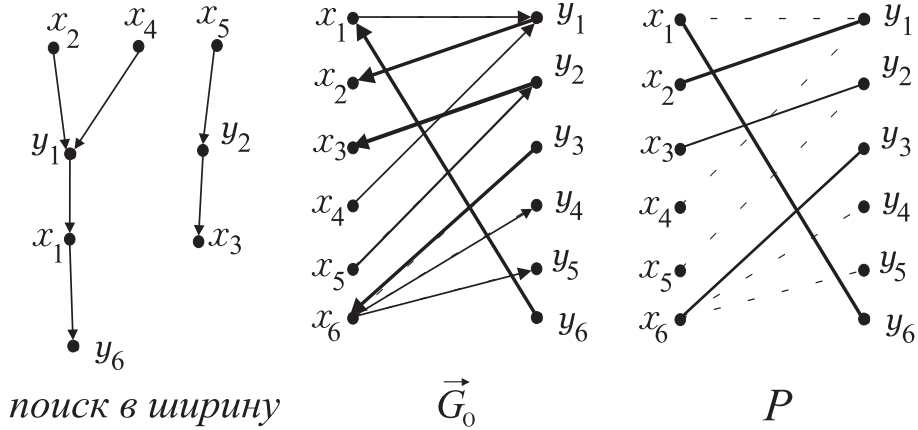


Рис. 63.

Поиск в ширину из множества не инцидентных ребрам паросочетания вершин  $\{x_4, x_5\}$  во множество  $Y \setminus Y_P = \{y_4, y_5\}$  не дает результатов (рис. 64). Следовательно, переходим к шестому шагу алгоритма. В результате поиска в ширину метки получили следующие вершины:

$$X' = \{x_2, x_3, x_4, x_5\}, \quad Y' = \{y_1, y_2\}.$$

Среди ребер  $(u, v)$  графа  $G$  таких, что  $u \in X'$ ,  $v \in Y \setminus Y' = \{y_3, y_4, y_5, y_6\}$  (в приведенной ниже матрице они выделены жирным шрифтом) найдем ребро минимального веса. В данном случае это может быть ребро  $(x_2, y_4)$  веса 1. Преобразуем веса ребер графа  $G$  следующим образом: из веса всех ребер инцидентных вершинам  $x_2, x_3, x_4, x_5$  вычтем 1 (из элементов соответствующих строк матрицы  $W_{XY}(G)$  вычтем 1), а к весу всех ребер инцидентных вершинам  $y_1, y_2$  прибавим 1 (к элементам первого и второго столбцов матрицы  $W_{XY}(G)$  прибавим 1).

$$\begin{pmatrix} 0 & 3 & \infty & 1 & \infty & 0 \\ 0 & \infty & \infty & \mathbf{1} & \mathbf{8} & \infty \\ \infty & 0 & \mathbf{7} & \infty & \mathbf{7} & \mathbf{3} \\ 0 & 4 & \infty & \infty & \mathbf{8} & \mathbf{6} \\ 1 & 0 & \mathbf{1} & \infty & \mathbf{6} & \mathbf{1} \\ 4 & 3 & 0 & 0 & 0 & \infty \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 3 & \infty & 1 & \infty & 0 \\ -1 & \infty & \infty & \mathbf{0} & \mathbf{7} & \infty \\ \infty & -1 & \mathbf{6} & \infty & \mathbf{6} & \mathbf{2} \\ -1 & 3 & \infty & \infty & \mathbf{7} & \mathbf{5} \\ 0 & -1 & \mathbf{0} & \infty & \mathbf{5} & \mathbf{0} \\ 4 & 3 & 0 & 0 & 0 & \infty \end{pmatrix} \Rightarrow \begin{pmatrix} 1 & 4 & \infty & 1 & \infty & 0 \\ 0 & \infty & \infty & \mathbf{0} & \mathbf{7} & \infty \\ \infty & 0 & \mathbf{6} & \infty & \mathbf{6} & \mathbf{2} \\ 0 & 4 & \infty & \infty & \mathbf{7} & \mathbf{5} \\ 1 & 0 & \mathbf{0} & \infty & \mathbf{5} & \mathbf{0} \\ 5 & 4 & 0 & 0 & 0 & \infty \end{pmatrix}.$$

В результате, вес ребер  $(x_5, y_3)$ ,  $(x_4, y_6)$  и  $(x_2, y_4)$  стал равен нулю, а вес ребра  $(x_1, y_1)$ , который раньше был равен нулю — увеличился. Преобразуем граф  $\vec{G}_0(X, Y, \vec{E}_0)$ . Добавим дуги  $(x_5, y_3)$ ,  $(x_4, y_6)$ ,  $(x_2, y_4)$  и удалим дугу  $(x_1, y_1)$  (рис. 64).

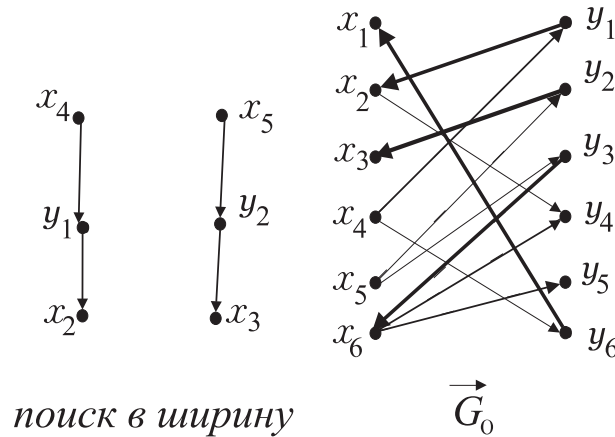


Рис. 64.

Так как  $X \setminus X_P = \{x_4, x_5\} \neq \emptyset$  и  $Y \setminus Y_P = \{y_4, y_5\} \neq \emptyset$ , то в графе  $\vec{G}_0$  из множества  $X \setminus X_P$  выполняем поиск в ширину во множество  $\{y_4, y_5\}$  (рис. 65). В результате найдем путь  $x_4, y_1, x_2, y_4$  и получит метку вершина  $y_4$ . В паросочетание добавляем ребра  $(x_4, y_1)$ ,  $(x_2, y_4)$  и удаляем ребро  $(x_2, y_1)$  (рис. 65):

$$X_P = \{x_1, x_2, x_3, x_4, x_6\}, \quad Y_P = \{y_1, y_2, y_3, y_4, y_6\},$$

$$E_P = \{(x_4, y_1), (x_2, y_4), (x_1, y_6), (x_3, y_2), (x_6, y_3)\}.$$

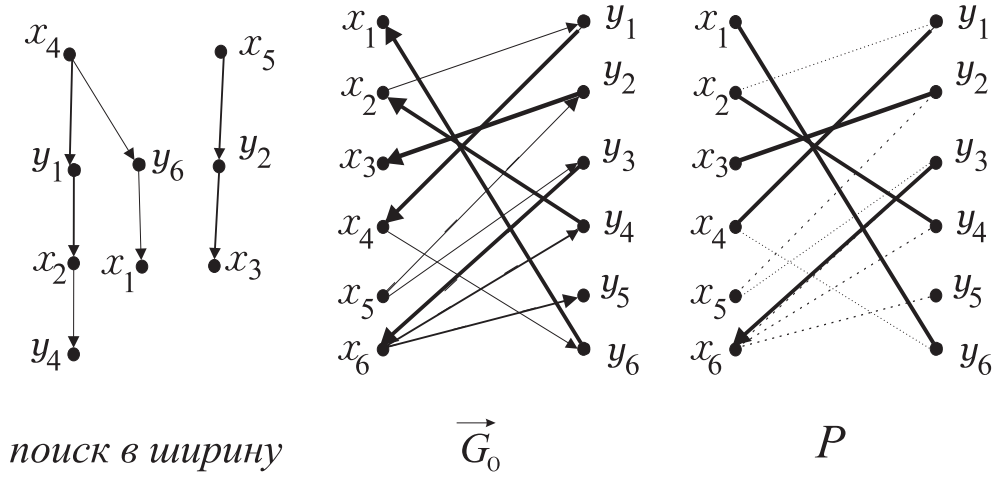


Рис. 65.

Так как  $X \setminus X_P = \{x_5\} \neq \emptyset$  и  $Y \setminus Y_P = \{y_5\} \neq \emptyset$ , то в графе  $\vec{G}_0$  из вершины  $x_5$  выполняем поиск в ширину (рис. 66). В результате найдем путь  $x_5, y_3, x_6, y_5$  и получит метку вершина  $y_5$ . В паросочетание добавляем ребра  $(x_5, y_3)$ ,  $(x_6, y_5)$  и удаляем ребро  $(x_6, y_3)$  (рис. 66):

$$P = \{(x_4, y_1), (x_2, y_4), (x_1, y_6), (x_3, y_2), (x_5, y_3), (x_6, y_5)\},$$

$$X_P = \{x_1, x_2, x_3, x_4, x_5, x_6\} = X, \quad Y_P = \{y_1, y_2, y_3, y_4, y_5, y_6\} = Y.$$

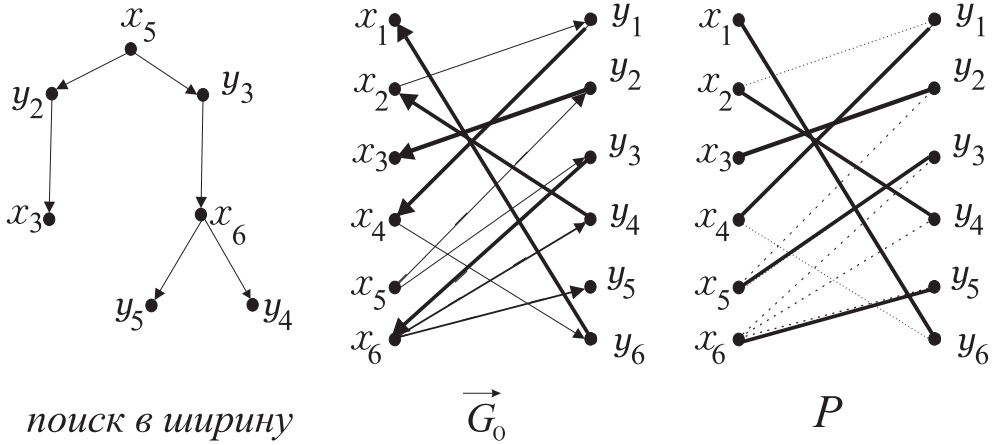


Рис. 66.

Так как  $X \setminus X_P = Y \setminus Y_P = \emptyset$ , то алгоритм заканчивает работу.  $P$  — искомое паросочетание. Вычислим его вес:

$$w(P) = w_{16} + w_{24} + w_{32} + w_{41} + w_{53} + w_{65} = 1 + 3 + 3 + 4 + 6 + 6 = 23.$$

✓ **ПРИМЕР 11.3.** В полном двудольном графе, заданном матрицей длин дуг, найти совершенное паросочетание минимального веса.

$$W_{XY}(G) = \begin{pmatrix} 2 & 4 & 5 & 2 & 6 \\ 1 & 12 & 15 & 11 & 13 \\ 2 & 7 & 10 & 6 & 10 \\ 3 & 15 & 15 & 11 & 13 \\ 1 & 12 & 12 & 11 & 10 \end{pmatrix}.$$

**Решение.**

Преобразуем матрицу весов графа так, как предлагается в алгоритме. А именно, вычтем из каждой строки, а затем из каждого столбца их минимальный элемент:

$$W_{XY}(G) \Rightarrow \begin{pmatrix} 0 & 2 & 3 & 0 & 4 \\ 0 & 11 & 14 & 10 & 12 \\ 0 & 5 & 8 & 4 & 8 \\ 0 & 12 & 12 & 8 & 10 \\ 0 & 11 & 11 & 10 & 9 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 9 & 11 & 10 & 8 \\ 0 & 3 & 5 & 4 & 4 \\ 0 & 10 & 9 & 8 & 6 \\ 0 & 9 & 8 & 10 & 5 \end{pmatrix}.$$

Построим граф  $G_0(X, Y, E_0)$  (рис. 67), где  $E_0$  — множество ребер графа с весом 0. Выделим в  $G_0$  некоторое максимальное паросочетание  $P$ :

$$X_P = \{x_1, x_2\}, \quad Y_P = \{y_1, y_2\}, \quad P = \{(x_1, y_2), (x_2, y_1)\}.$$

Построим также оргграф  $\vec{G}_0(X, Y, \vec{E}_0)$  (рис. 67): все ребра, входящие в паросочетание  $P$ , преобразуем в дуги, направленные из множества  $Y$  в множество  $X$ , а все ребра, не входящие в паросочетание  $P$  преобразуем в дуги, направленные из множества  $X$  в множество  $Y$ .

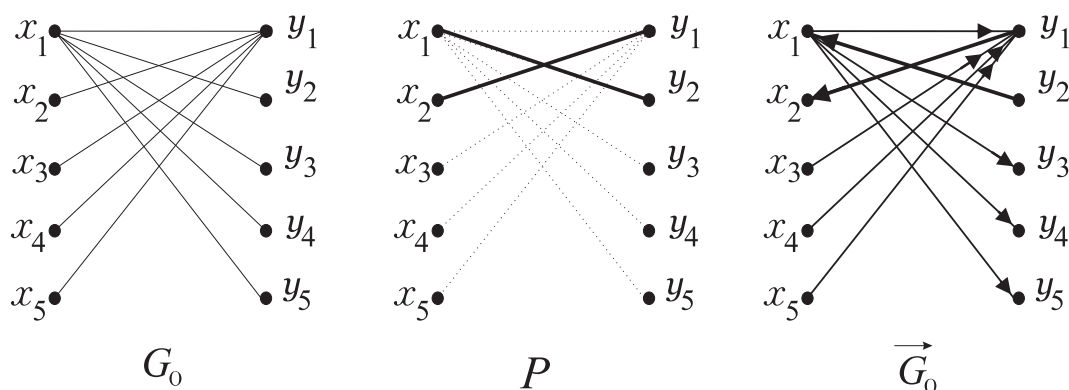


Рис. 67.



Так как  $X \setminus X_P = \{x_3, x_4, x_5\} \neq \emptyset$  и  $Y \setminus Y_P = \{y_3, y_4, y_5\} \neq \emptyset$ , то в графе  $\vec{G}_0$  из множества  $X \setminus X_P = \{x_3, x_4, x_5\}$  выполняем поиск в ширину во множество  $Y \setminus Y_P = \{y_3, y_4, y_5\}$  (рис. 68). В результате ни одна из вершин множества  $Y \setminus Y_P$  не получает метки и, следовательно переходим к шестому шагу алгоритма. В результате поиска в ширину метки получили следующие вершины:

$$X' = \{x_2, x_3, x_4, x_5\}, \quad Y' = \{y_1\}.$$

Среди ребер  $(u, v)$  графа  $G$  таких, что  $u \in X', v \in Y \setminus Y' = \{y_2, y_3, y_4, y_5\}$  (а это фактически все ненулевые ребра графа) выберем ребро минимального веса —  $(x_3, y_2)$ ,  $w(x_3, y_2) = 3$ . Из веса всех ребер инцидентных вершинам  $x_2, x_3, x_4, x_5$  вычтем 3, а к весу всех ребер инцидентных вершине  $y_1$  прибавим 3 (т.е. из элементов 2-й, 3-й, 4-й и 5-й строк матрицы весов вычтем 3, а к элементам 1-го столбца прибавим 3).

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 9 & 11 & 10 & 8 \\ 0 & 3 & 5 & 4 & 4 \\ 0 & 10 & 9 & 8 & 6 \\ 0 & 9 & 8 & 10 & 5 \end{pmatrix} \Rightarrow \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ -3 & 6 & 8 & 7 & 5 \\ -3 & 0 & 2 & 1 & 1 \\ -3 & 7 & 6 & 5 & 3 \\ -3 & 6 & 5 & 7 & 2 \end{pmatrix} \Rightarrow \begin{pmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 6 & 8 & 7 & 5 \\ 0 & 0 & 2 & 1 & 1 \\ 0 & 7 & 6 & 5 & 3 \\ 0 & 6 & 5 & 7 & 2 \end{pmatrix}.$$

В результате получим новое ребро  $(x_3, y_2)$  с весом 0, а вес ребра  $(x_1, y_1)$ , ранее равный нулю — увеличился. Преобразуем граф  $\vec{G}_0(X, Y, \vec{E}_0)$  (рис. 68). Добавим к графу дугу  $(x_3, y_2)$  и удалим дугу  $(x_1, y_1)$ .

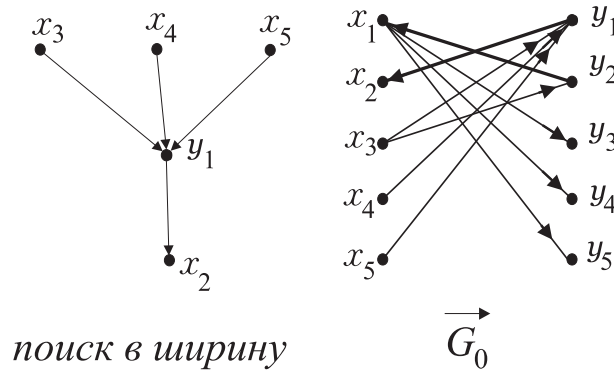


Рис. 68.

В преобразованном графе  $\vec{G}_0$  проведем поиск в ширину из множества вершин  $X \setminus X_P = \{x_3, x_4, x_5\}$  (рис. 69) во множество  $Y \setminus Y_P = \{y_3, y_4, y_5\}$ . В результате получают метки все вершины из множества  $Y \setminus Y_P$ . Например,

в вершину  $y_3$  ведет путь  $x_3, y_2, x_1, y_3$ .

Добавим к паросочетанию  $P$  ребра  $(x_3, y_2)$ ,  $(x_1, y_3)$  и удалим ребро  $(x_1, y_2)$ :

$$X_P = \{x_1, x_2, x_3\}, \quad Y_P = \{y_1, y_2, y_3\}, \quad P = \{(x_3, y_2), (x_1, y_3), (x_2, y_1)\}.$$

Преобразуем соответствующий орграф  $\vec{G}_0(X, Y, \vec{E}_0)$  (рис. 69).

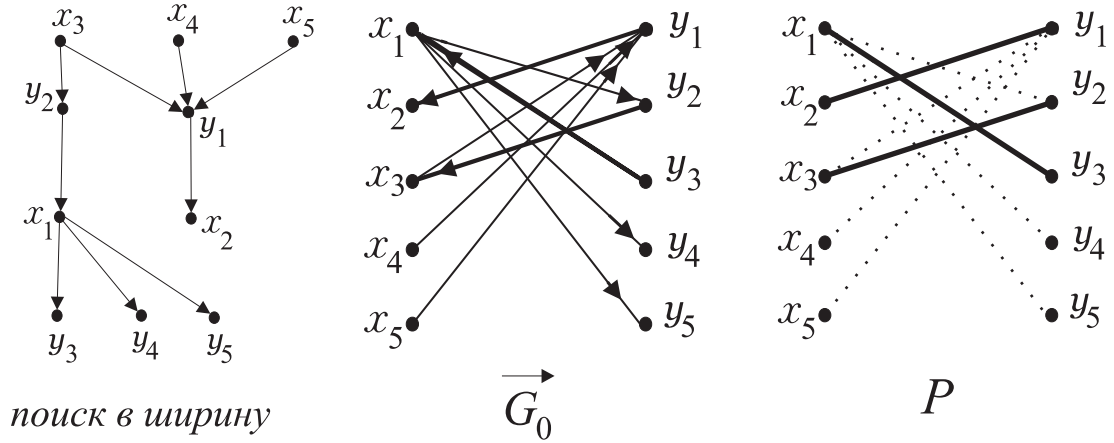


Рис. 69.

Так как  $X \setminus X_P = \{x_4, x_5\} \neq \emptyset$  и  $Y \setminus Y_P = \{y_4, y_5\} \neq \emptyset$ , то в графе  $\vec{G}_0$  из множества  $X \setminus X_P$  выполняем поиск в ширину во множество  $Y \setminus Y_P$  (рис. 70). Так как в результате поиска вершины из  $Y \setminus Y_P$  меток не получают, то переходим к шестому шагу алгоритма. В результате последнего поиска в ширину, метки получили следующие вершины:

$$X' = \{x_2, x_4, x_5\}, \quad Y' = \{y_1\}.$$

Среди ребер  $(u, v)$  графа  $G$  таких, что  $u \in X'$ ,  $v \in Y \setminus Y' = \{y_2, y_3, y_4, y_5\}$  выберем ребро  $(x_5, y_5)$  минимального веса 2. Из веса всех ребер, инцидентных вершинам  $x_2, x_4, x_5$  вычтем 2, а к весу всех ребер, инцидентных вершине  $y_1$  прибавим 2:

$$\begin{pmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{6} & \mathbf{8} & \mathbf{7} & \mathbf{5} \\ 0 & 0 & 2 & 1 & 1 \\ 0 & \mathbf{7} & \mathbf{6} & \mathbf{5} & \mathbf{3} \\ 0 & \mathbf{6} & \mathbf{5} & \mathbf{7} & \mathbf{2} \end{pmatrix} \Rightarrow \begin{pmatrix} 3 & 0 & 0 & 0 & 0 \\ -2 & 4 & 6 & 5 & 3 \\ 0 & 0 & 2 & 1 & 1 \\ -2 & 5 & 4 & 3 & 1 \\ -2 & 4 & 3 & 5 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 5 & 0 & 0 & 0 & 0 \\ 0 & 4 & 6 & 5 & 3 \\ 2 & 0 & 2 & 1 & 1 \\ 0 & 5 & 4 & 3 & 1 \\ 0 & 4 & 3 & 5 & 0 \end{pmatrix}.$$

Добавим к паросочетанию  $P$  появившееся в результате преобразования ребро  $(x_5, y_5)$  веса 0 и удалим ребро  $(x_3, y_1)$ , вес которого увеличился. Преобразуем граф  $\vec{G}_0(X, Y, \vec{E}_0)$  (рис. 70).

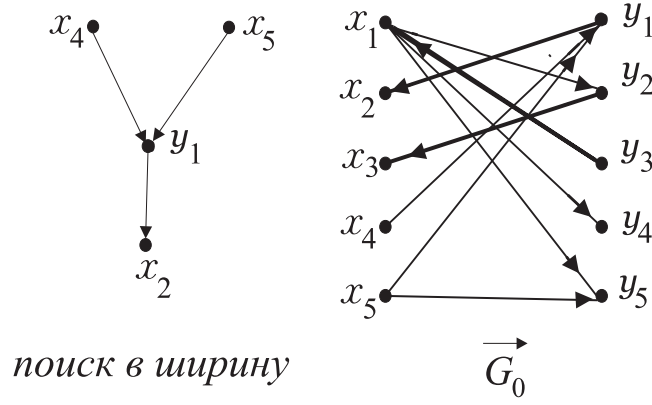


Рис. 70.

В преобразованном графе  $\vec{G}_0$  проведем поиск в ширину из множества вершин  $X \setminus X_P = \{x_4, x_5\}$  (рис. 71) во множество  $Y \setminus Y_P = \{y_4, y_5\}$ . В результате поиска, найден путь  $x_5, y_5$  в вершину  $y_5$ . Добавим к паросочетанию  $P$  ребро  $(x_5, y_5)$  и преобразуем граф  $\vec{G}_0$  (рис. 71):

$$X_P = \{x_1, x_2, x_3, x_5\}, \quad Y_P = \{y_1, y_2, y_3, y_5\}, \quad P = \{(x_3, y_2), (x_1, y_3), (x_2, y_1), (x_5, y_5)\}.$$

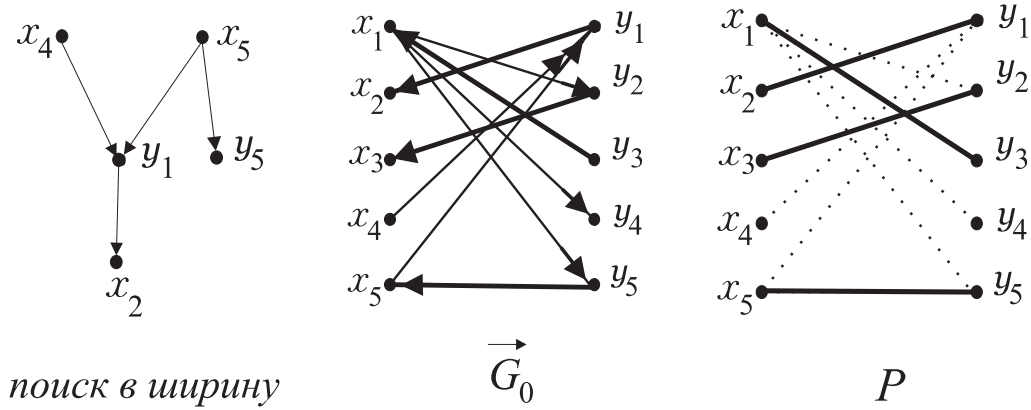


Рис. 71.

Так как  $X \setminus X_P = \{x_4\} \neq \emptyset$  и  $Y \setminus Y_P = \{y_4\} \neq \emptyset$ , то в графе  $\vec{G}_0$  из вершины  $x_4$  выполняем поиск в ширину (рис. 72). Поиск не дает новых вершин из  $Y \setminus Y_P$ , переходим к шестому шагу алгоритма.

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

Среди ребер  $(u, v)$  графа  $G$  таких, что  $u \in X'$ ,  $v \in Y \setminus Y' = \{y_2, y_3, y_4, y_5\}$  выберем ребро  $(x_4, y_5)$  минимального веса 1. Из веса всех ребер, инцидентных вершинам  $x_2, x_4$ , вычтем 1, а к весу всех ребер, инцидентных вершине

$y_1$ , прибавим 1:

$$\begin{pmatrix} 5 & 0 & 0 & 0 & 0 \\ 0 & 4 & \mathbf{6} & \mathbf{5} & \mathbf{3} \\ 2 & 0 & 2 & 1 & 1 \\ 0 & \mathbf{5} & 4 & \mathbf{3} & \mathbf{1} \\ 0 & 4 & 3 & 5 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 5 & 0 & 0 & 0 & 0 \\ -1 & 3 & 5 & 4 & 2 \\ 2 & 0 & 2 & 1 & 1 \\ -1 & 4 & 3 & 2 & 0 \\ 0 & 4 & 3 & 5 & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 6 & 0 & 0 & 0 & 0 \\ 0 & 3 & 5 & 4 & 2 \\ 3 & 0 & 2 & 1 & 1 \\ 0 & 4 & 3 & 2 & 0 \\ 1 & 4 & 3 & 5 & 0 \end{pmatrix}.$$

Добавим к паросочетанию  $P$  ребро  $(x_4, y_5)$  веса 0 и удалим ребро  $(x_5, y_1)$ , вес которого увеличился (рис. 72). Преобразуем граф  $\vec{G}_0(X, Y, \vec{E}_0)$ .

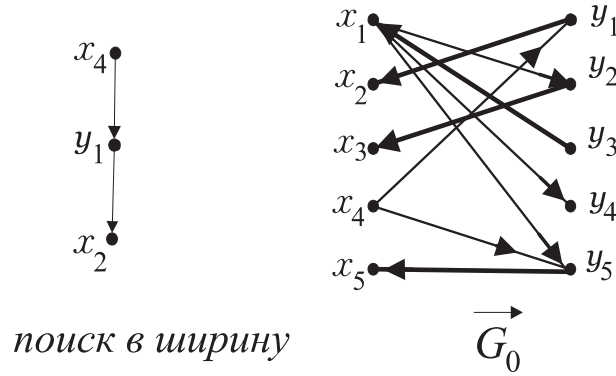


Рис. 72.

В преобразованном графе  $\vec{G}_0$  проведем поиск в ширину из из вершины  $x_4$  (рис. 73). В результате поиска не удастся найти путь в  $y_4$ , следовательно, переходим к шестому шагу алгоритма:

$$X' = \{x_2, x_4, x_5\}, Y' = \{y_1, y_5\}.$$

Среди ребер  $(u, v)$  графа  $G$  таких, что  $u \in X'$ ,  $v \in Y \setminus Y' = \{y_2, y_3, y_4\}$  выберем ребро  $(x_4, y_4)$  минимального веса 2. Из веса всех ребер, инцидентных вершинам  $x_2, x_4, x_5$ , вычтем 2, а к весу всех ребер, инцидентных вершинам  $y_1, y_5$ , прибавим 2:

$$\begin{pmatrix} 6 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{3} & \mathbf{5} & 4 & 2 \\ 3 & 0 & 2 & 1 & 1 \\ 0 & 4 & \mathbf{3} & \mathbf{2} & 0 \\ 1 & 4 & \mathbf{3} & \mathbf{5} & 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 6 & 0 & 0 & 0 & 0 \\ -2 & 1 & 3 & 2 & 0 \\ 3 & 0 & 2 & 1 & 1 \\ -2 & 2 & 1 & 0 & -2 \\ -1 & 2 & 1 & 3 & -2 \end{pmatrix} \Rightarrow \begin{pmatrix} 8 & 0 & 0 & 0 & 2 \\ 0 & 1 & 3 & 2 & 2 \\ 5 & 0 & 2 & 1 & 3 \\ 0 & 2 & 1 & 0 & 0 \\ 1 & 2 & 1 & 3 & 0 \end{pmatrix}.$$

Добавим к паросочетанию  $P$  ребро  $(x_4, y_4)$  веса 0 и удалим ребро  $(x_1, y_5)$ , вес которого увеличился. Преобразуем граф  $\vec{G}_0(X, Y, \vec{E}_0)$  (рис. 73).

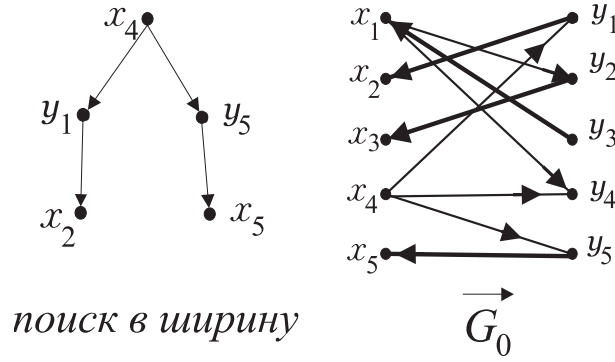


Рис. 73.

В преобразованном графе  $\vec{G}_0$  проведем поиск в ширину из вершины  $x_4$  (рис. 74). В результате поиска найдем путь  $x_4, y_4$  в вершину  $y_4$  (рис. 74). Добавим к паросочетанию  $P(X_P, Y_P, E_P)$  ребро  $(x_4, y_4)$  и преобразуем граф  $\vec{G}_0$ :

$$X_P = \{x_1, x_2, x_3, x_4, x_5\}, \quad Y_P = \{y_1, y_2, y_3, y_4, y_5\},$$

$$P = \{(x_3, y_2), (x_1, y_3), (x_2, y_1), (x_5, y_5), (x_4, y_4)\}.$$

Так как  $X \setminus X_P = \emptyset$ ,  $Y \setminus Y_P = \emptyset$ , то алгоритм заканчивает работу.  $P$  — искомое паросочетание (рис. 74).

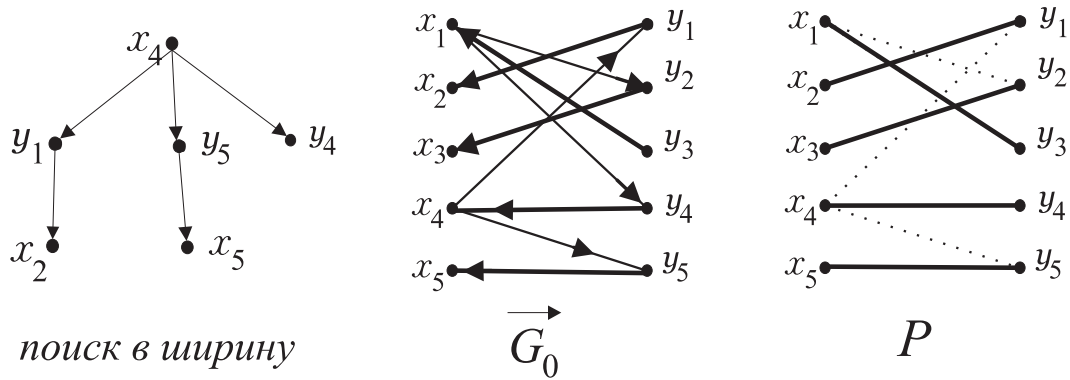


Рис. 74.

Вычислим его вес:

$$w(P) = w_{13} + w_{21} + w_{32} + w_{44} + w_{55} = 7 + 5 + 1 + 10 + 11 = 34.$$

## УПРАЖНЕНИЯ

1. В графах, изображенных на рис. 75, найти все максимальные паросочетания.

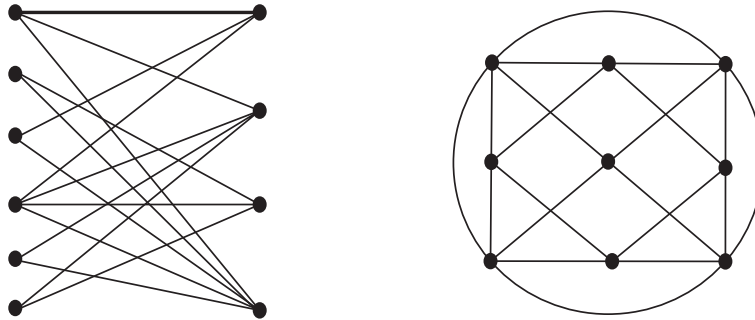


Рис. 75.

2. Найти наибольшее паросочетание в двудольном графе, заданном матрицей смежности:

$$A_{XY}(G_1) = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad A_{XY}(G_2) = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}.$$

3. Найти совершенное паросочетание минимального веса в полном двудольном графе, заданном матрицей весов:

$$\text{а) } W_{XY}(G) = \begin{pmatrix} 3 & 2 & 1 & 3 & 4 & 7 & 1 & 3 & 4 \\ 2 & 5 & 3 & 5 & 3 & 11 & 11 & 2 & 5 \\ 3 & 3 & 5 & 4 & 11 & 3 & 6 & 5 & 9 \\ 3 & 7 & 11 & 5 & 4 & 4 & 2 & 4 & 6 \\ 12 & 3 & 1 & 7 & 4 & 1 & 7 & 4 & 8 \\ 3 & 4 & 1 & 4 & 5 & 3 & 12 & 5 & 4 \\ 7 & 2 & 5 & 4 & 8 & 9 & 1 & 3 & 7 \\ 2 & 5 & 2 & 6 & 4 & 5 & 3 & 6 & 2 \\ 7 & 7 & 9 & 8 & 11 & 8 & 7 & 9 & 9 \end{pmatrix}; \quad \text{б) } W_{XY}(G) = \begin{pmatrix} 0 & 2 & 1 & 3 & 4 & 7 & 1 \\ 2 & 5 & 3 & 5 & 3 & 0 & 0 \\ 0 & 3 & 5 & 4 & 0 & 3 & 6 \\ 3 & 0 & 0 & 5 & 4 & 4 & 2 \\ 0 & 3 & 1 & 5 & 1 & 1 & 7 \\ 3 & 4 & 1 & 4 & 5 & 3 & 0 \\ 7 & 2 & 5 & 4 & 0 & 0 & 1 \end{pmatrix}.$$

## 12. Раскраска графа

Пусть  $G(V, E)$  — неориентированный граф без петель.

*Опр.*

*Раскраска* графа — функция, которая каждой вершине ставит в соответствие некоторое натуральное число (цвет). Раскраска называется *правильной*, если смежные вершины имеют разные цвета

*Опр.*

*Хроматическое число*  $\chi(G)$  графа  $G$  равно минимально возможному числу цветов в раскраске графа.

**Теорема 1.** Хроматическое число графа не больше, чем максимальная степень вершины увеличенная на 1:  $\chi(G) \leq \max_{v \in V(G)} \{d(v)\} + 1$ .

*Опр.*

*Независимое множество вершин* графа — множество попарно несмежных вершин графа.

*Опр.*

*Максимальное независимое множество* вершин графа — независимое множество вершин графа, к которому невозможно добавить вершины.

*Опр.*

*Число вершинной независимости*  $\beta(G)$  графа  $G$  равно мощности наибольшего независимого множества вершин графа.

**Замечание.** Независимое множество ребер графа — паросочетание.

**Теорема 2.** Для графа  $G(V, E)$  с хроматическим числом  $\chi(G)$  и числом вершинной независимости  $\beta(G)$  имеет место двойное неравенство:

$$\frac{|V(G)|}{\beta(G)} \leq \chi(G) \leq |V(G)| - \beta(G) + 1.$$

**Теорема 3.** Хроматическое число полного графа равно числу его вершин.

**Теорема 4.** Хроматическое число графа равно 2 тогда и только тогда, когда он является двудольным.

**Теорема 5.** Хроматическое число дерева равно 2.

*Опр.*

*Планарный граф* — граф, который может быть изображен на плоскости без пересекающихся ребер.





**Замечание.** Для графа с достаточно большим числом вершин применение точного алгоритма требует большого числа операций, т.к. выделение максимальных независимых множеств связано с перебором вершин.

✓ **ПРИМЕР 12.1.** Задать раскраску вершин графа, изображенного на рис. 76, используя жадный алгоритм раскрашивания.

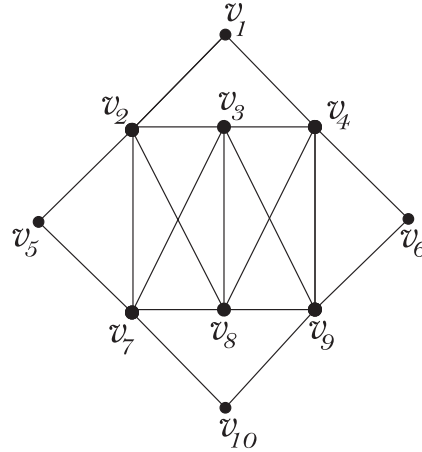


Рис. 76.

**Решение.**

На начальном этапе:

$$S := \{v_1, v_2, \dots, v_{10}\}, \quad k := 0, \quad \forall v \in V \quad \varphi(v) \text{ — не определено.}$$

1) Формируем первое максимальное независимое множество вершин:

$$k := 1, \quad C_1 := \{v_1\}.$$

$$S \setminus (C_1 \cup \Gamma(C_1)) = S \setminus \{v_1, v_2, v_4\} = \{v_3, v_5, v_6, v_7, v_8, v_9, v_{10}\},$$

$$C_1 := \{v_1, v_3\};$$

$$S \setminus (C_1 \cup \Gamma(C_1)) = S \setminus \{v_1, v_2, v_3, v_4, v_7, v_8, v_9\} = \{v_5, v_6, v_{10}\},$$

$$C_1 := \{v_1, v_3, v_5\};$$

$$S \setminus (C_1 \cup \Gamma(C_1)) = S \setminus \{v_1, v_2, v_3, v_4, v_5, v_7, v_8, v_9\} = \{v_6, v_{10}\},$$

$$C_1 := \{v_1, v_3, v_5, v_6\};$$

$$S \setminus (C_1 \cup \Gamma(C_1)) = S \setminus \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9\} = \{v_{10}\},$$

$$C_1 := \{v_1, v_3, v_5, v_6, v_{10}\};$$

$$S \setminus (C_1 \cup \Gamma(C_1)) = S \setminus \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}\} = \emptyset.$$

Красим вершины  $C_1$  в первый цвет (рис. 77), удаляем их из множества неокрашенных:

$$\varphi(v_1) = \varphi(v_3) = \varphi(v_5) = \varphi(v_6) = \varphi(v_{10}) = 1, \quad S := \{v_2, v_4, v_7, v_8, v_9\}.$$

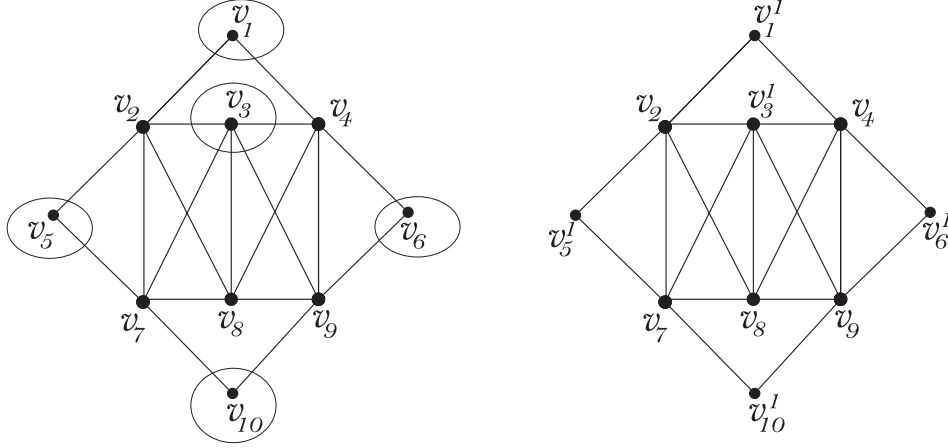


Рис. 77.

2) Формируем второе максимальное независимое множество вершин:

$$k := 2, \quad C_2 := \{v_2\}.$$

$$S \setminus (C_2 \cup \Gamma(C_2)) = S \setminus \{v_2, v_7, v_8\} = \{v_4, v_9\}, \quad C_2 := \{v_2, v_4\};$$

$$S \setminus (C_2 \cup \Gamma(C_2)) = S \setminus \{v_2, v_4, v_7, v_8, v_9\} = \emptyset.$$

Красим вершины  $C_2$  во второй цвет (рис. 78), удаляем их из множества неокрашенных:

$$\varphi(v_2) = \varphi(v_4) = 2, \quad S := \{v_7, v_8, v_9\}.$$

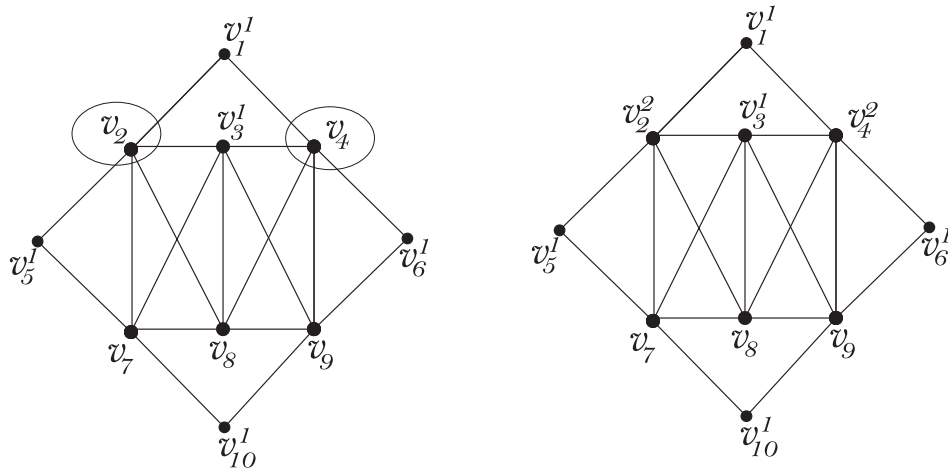


Рис. 78.

3) Формируем третье максимальное независимое множество вершин:

$$k := 3, \quad C_3 := \{v_7\}.$$

$$S \setminus (C_3 \cup \Gamma(C_3)) = S \setminus \{v_7, v_8\} = \{v_9\}, \quad C_3 := \{v_7, v_9\};$$

$$S \setminus (C_3 \cup \Gamma(C_3)) = S \setminus \{v_7, v_8, v_9\} = \emptyset.$$

Красим вершины  $C_3$  в третий цвет (рис. 79), удаляем их из множества неокрашенных:

$$\varphi(v_7) = \varphi(v_9) = 3, \quad S := \{v_8\}.$$

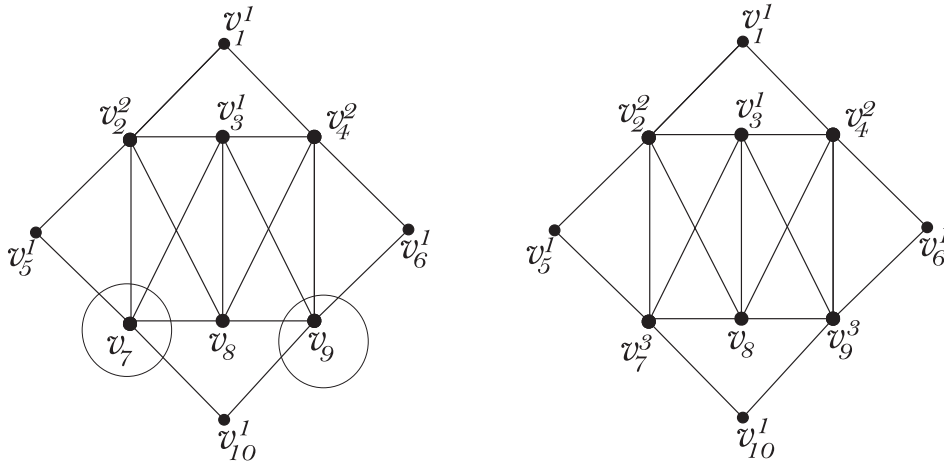


Рис. 79.

4) Оставшуюся вершину  $v_8$  красим в четвертый цвет (рис. 80):

$$\varphi(v_8) = 4, \quad S := \emptyset.$$

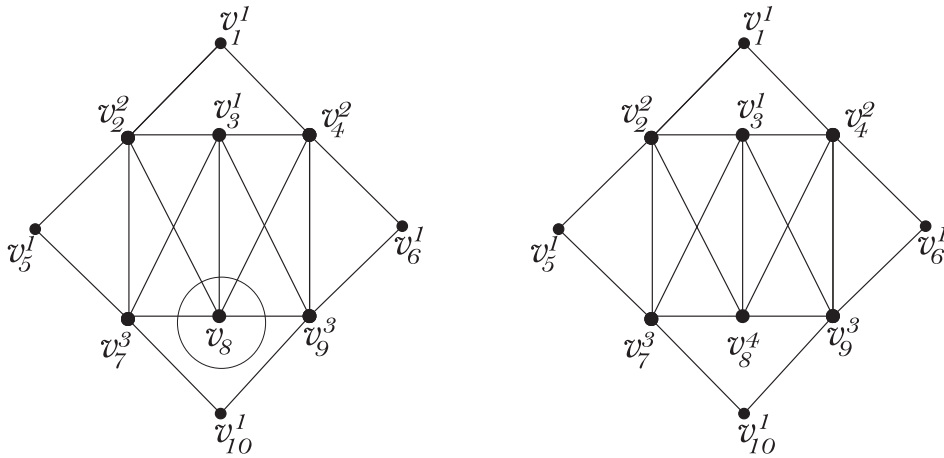


Рис. 80.

## 12.2. Алгоритм последовательного раскрашивания

**Идея алгоритма.** Для каждой вершины  $v$  графа  $G$  определяем множество допустимых цветов  $S(v)$ . На начальном этапе, для каждой вершины множество допустимых цветов  $S(v) = \{1, 2, \dots, n\}$ ,  $n = |V(G)|$ . Красим вершины последовательно: сначала  $v_1$ , затем  $v_2$  и т.д., выбирая из допустимых цветов минимальный и удаляя его из множества допустимых цветов вершин, смежных с окрашенной.

**Шаг 1.** Вершине  $v_1$  ставим в соответствие цвет 1 и удаляем его из множеств допустимых цветов вершин из окрестности  $v_1$ :

$$\varphi(v_1) := 1, \forall v_i \in \Gamma(v_1) S(v_i) := \{2, \dots, n\}.$$

**Шаг 2.** Вершине  $v_2$  ставим в соответствие минимальный цвет из  $S(v_2)$  и удаляем его из множеств допустимых цветов вершин из окрестности  $v_2$ :

$$\varphi(v_2) := \min_{s \in S(v_2)} s, \forall v_i \in \Gamma(v_2) S(v_i) := S(v_i) \setminus \{\varphi(v_2)\}.$$

...

**Шаг  $k$ .** Вершине  $v_k$  ставим в соответствие минимальный цвет из  $S(v_k)$  и удаляем его из множеств допустимых цветов вершин из окрестности  $v_k$ :

$$\varphi(v_k) := \min_{s \in S(v_k)} s, \forall v_i \in \Gamma(v_k) S(v_i) := S(v_i) \setminus \{\varphi(v_k)\}.$$

...

**Шаг  $n$ .** Вершине  $v_n$  ставим в соответствие минимальный цвет из  $S(v_n)$

$$\varphi(v_n) := \min_{s \in S(v_n)} s.$$

**Замечание 1.** В общем случае, алгоритм последовательного раскрашивания не является точным (т.е. число используемых цветов не минимально). Однако, среди  $n!$  различных способов упорядочивания вершин графа существует как минимум один, приводящий к раскраске в хроматическое число цветов.

**Замечание 2.** Построение передающих сетей радиовещания сводится к частотно - пространственному распределению на заданной территории. Задача распределения частот определенных заданным частотным спектром сводится к раскраске вершин данного графа сети связи.



**ПРИМЕР 12.2.** Структура сети сотовой связи города  $N$  представлена

на рис. 81. Радиус зон покрытия базовых станций равен 3 км, координационное расстояние — 9,2 км. Определите, какие станции будут оказывать взаимные влияния друг на друга, постройте граф сети и распределите с помощью алгоритма точного раскрашивания вершин частоты, таким образом, чтобы станции не создавали помех друг другу.

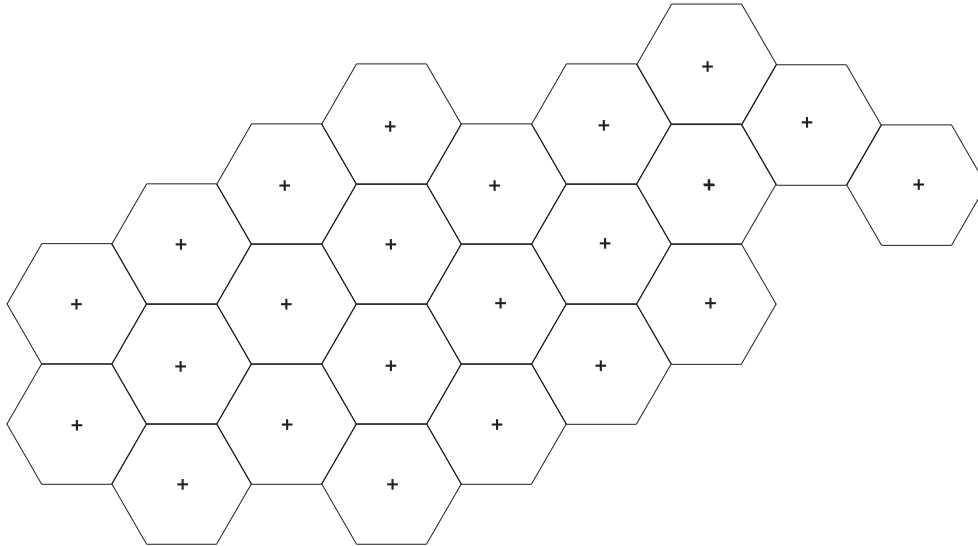


Рис. 81.

### Решение.

Возьмем в качестве точки отсчета некоторую станцию и пронумеруем станции в порядке возрастания по мере удаления от первой (рис. 82).

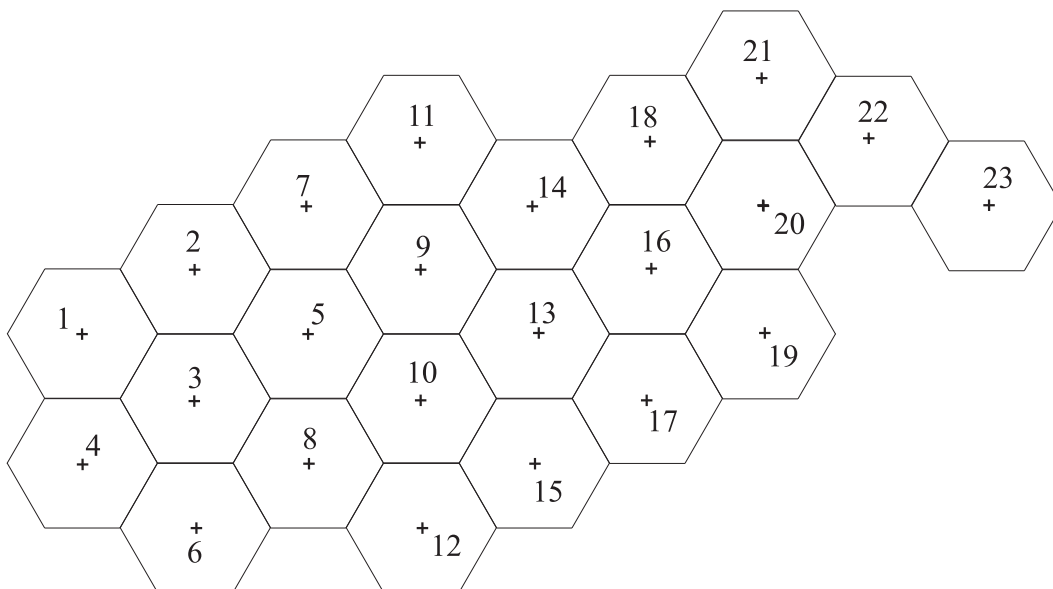


Рис. 82.

Вычислим расстояния от первой станции до ближайших ( $R_0$  — модуль сети):  
 $\rho_1 = \rho(1; 2) = \rho(1; 3) = \rho(1; 4) = R_0 = 3\sqrt{3} \approx 5,1;$

$$\rho_2 = \rho(1; 5) = \rho(1; 6) = 3 \cdot R_{\text{зоны}} = 9;$$

$$\rho_3 = \rho(1; 7) = \rho(1; 8) = 2 \cdot R_0 = 6\sqrt{3} \approx 10,3.$$

Расстояние до седьмой и восьмой станций превышает координационное, следовательно эти и более дальние станции не создают помех первой. Итак, в графе сети соединяем ребрами вершины, которые соответствуют станциям удаленным друг от друга на расстояние  $R_0$  или  $3R_{\text{зоны}}$  (рис. 83).

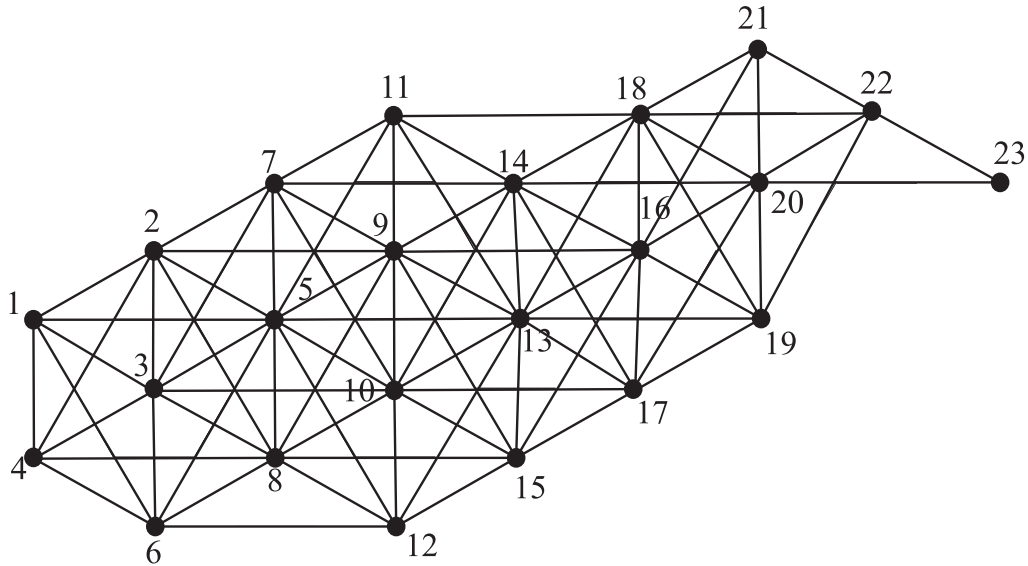


Рис. 83.

Выделяем максимальное независимое множество вершин —  $\{1; 7; 8; 13; 20\}$  и красим их в первый цвет (рис. 84):  $\varphi(1) = \varphi(7) = \varphi(8) = \varphi(13) = \varphi(20) = 1$ .

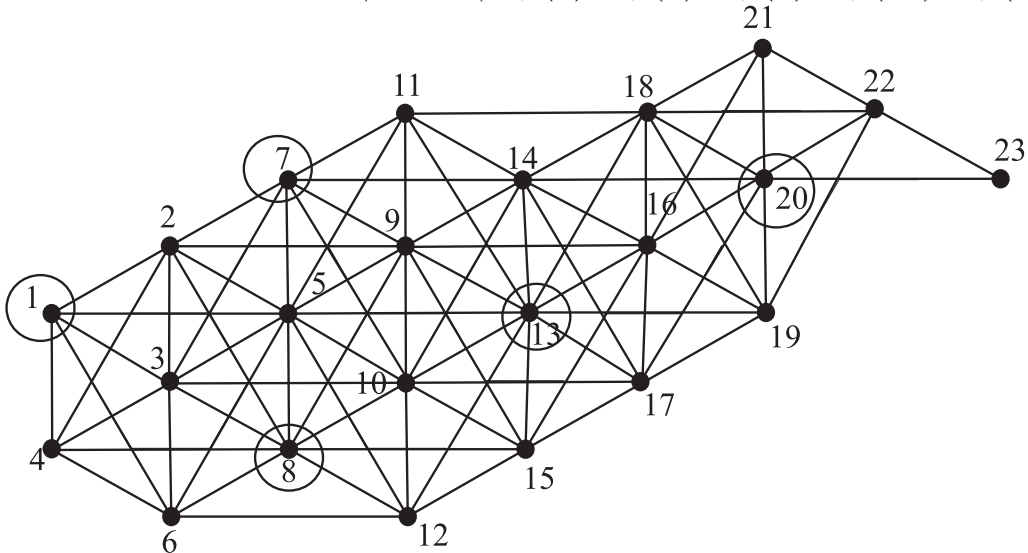


Рис. 84.

Удаляем покрашенные вершины вместе с инцидентными им ребрами из графа связи. Выделяем максимальное независимое множество вершин в

"урезанном графе" (рис. 85):  $\{2; 6; 10; 11; 16; 22\}$ . Красим их во второй цвет и удаляем из графа:

$$\varphi(2) = \varphi(6) = \varphi(10) = \varphi(11) = \varphi(16) = \varphi(22) = 2.$$

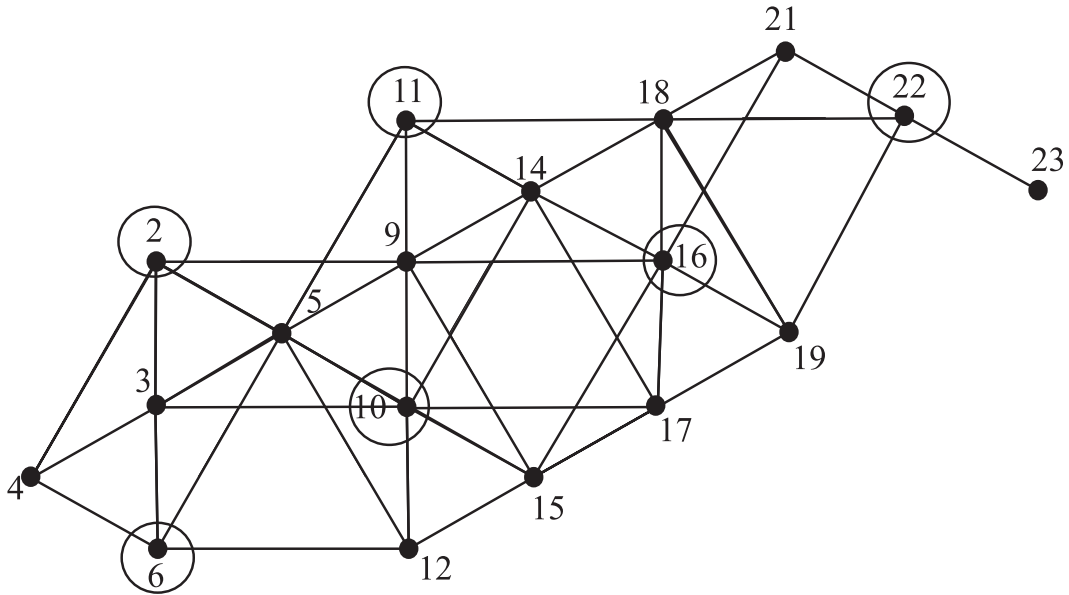


Рис. 85.

В полученном графе выделяем максимальное независимое множество вершин (рис. 86):  $\{4; 5; 14; 15; 19; 21; 23\}$ . Красим их в третий цвет и удаляем из графа:  $\varphi(4) = \varphi(5) = \varphi(14) = \varphi(15) = \varphi(19) = \varphi(21) = \varphi(23) = 3$ .

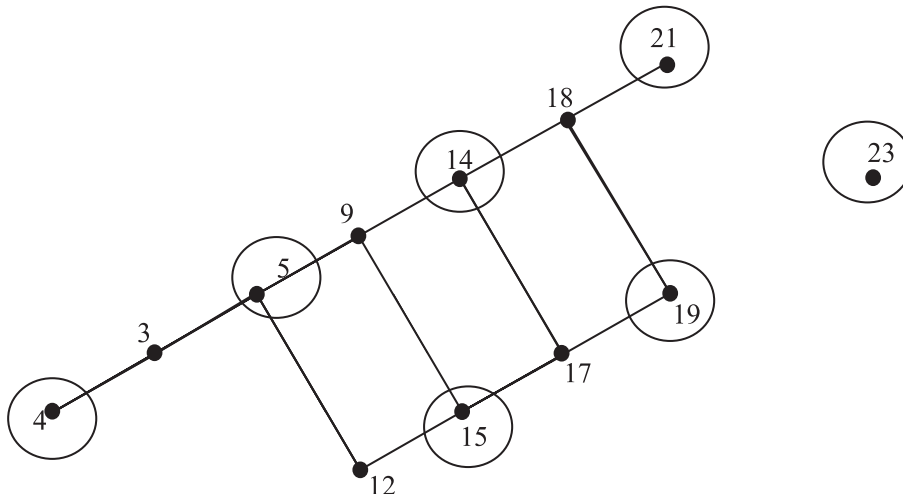


Рис. 86.

В графе, полученном после удаления вершин третьего цвета, максимальное независимое множество вершин (рис. 87) —  $\{3; 9; 12; 17; 18\}$ .

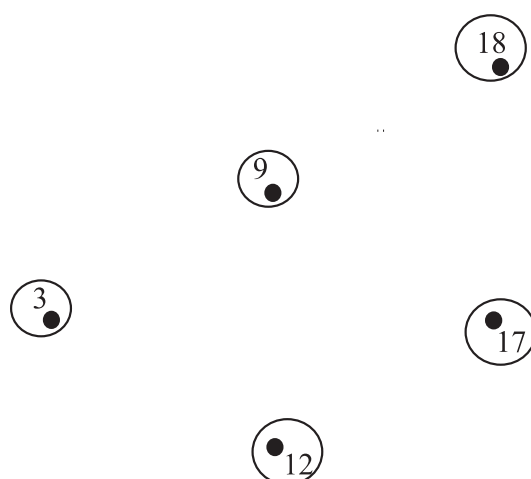


Рис. 87.

Красим их в четвертый цвет. Распределение частот продемонстрировано на рис. 88.

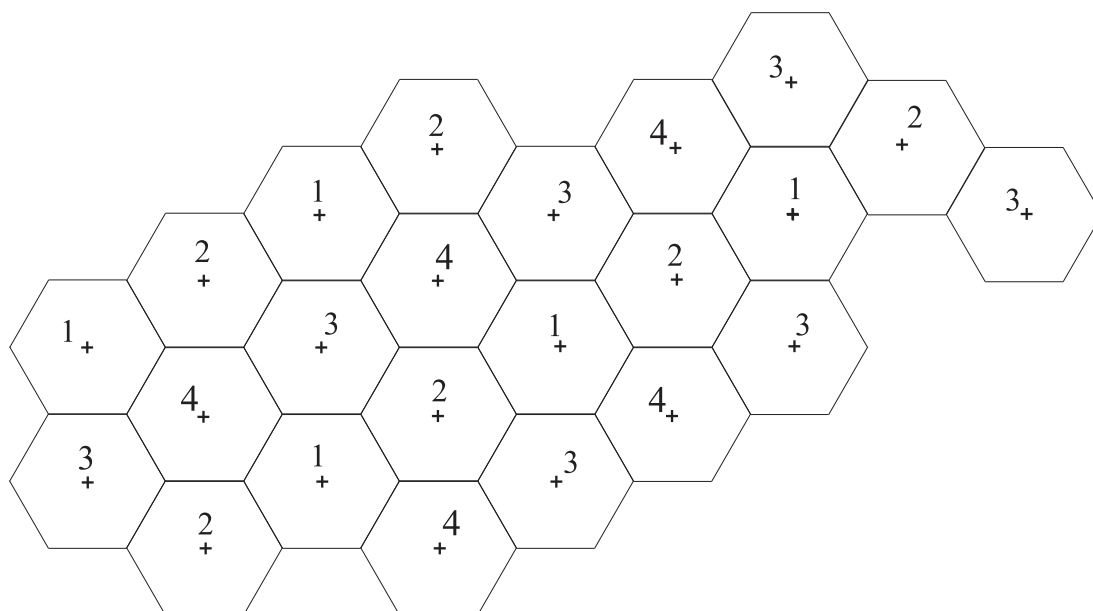


Рис. 88.

## УПРАЖНЕНИЯ

1. Структура сети сотовой связи представлена на рис. 89. Радиус зон покрытия базовых станций равен 1 км, координационное расстояние — 4 км. Определите, какие станции будут оказывать взаимные влияния друг на друга, постройте граф сети и распределите с помощью точного алгоритма раскрашивания и алгоритма последовательного раскрашивания вершин частоты, таким образом, чтобы станции не создавали помех друг другу.



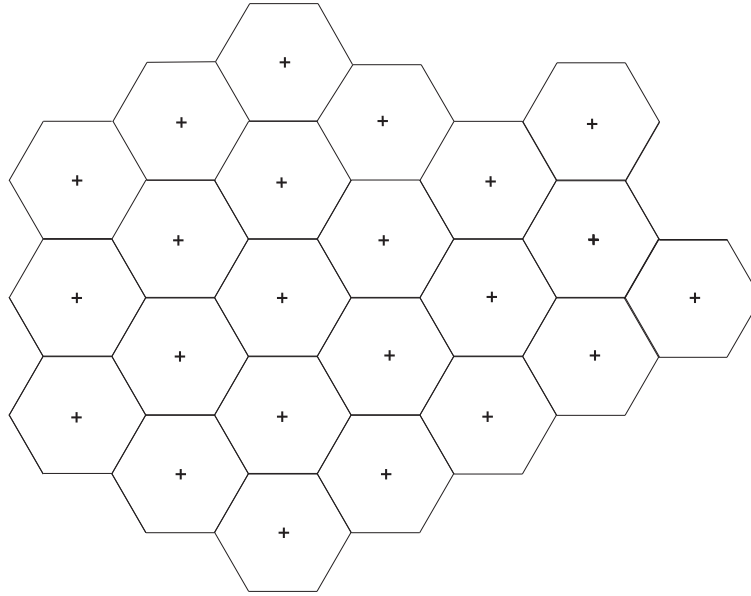
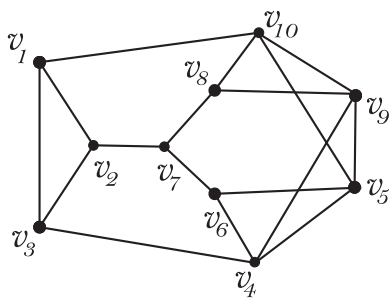


Рис. 89.

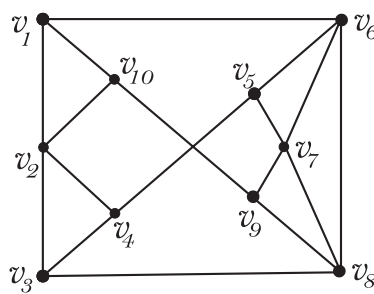
**2.** Задать раскраску вершин графа, заданного матрицей смежности  $A(G)$ , используя а) жадный алгоритм раскрашивания; б) алгоритм последовательного раскрашивания.

$$A(G) = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{pmatrix}.$$

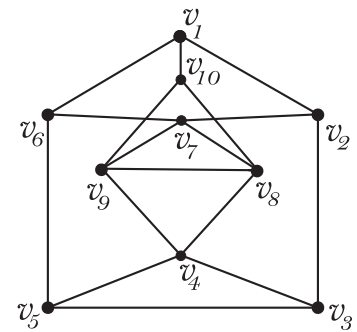
**3.** Задать раскраску вершин графов а)-в), изображенных на рис. 90, используя а) жадный алгоритм раскрашивания; б) алгоритм последовательного раскрашивания.



а)



б)



в)

Рис. 90.

## Список литературы

- [1] Оре О. Теория графов. — М.: Наука, 1968. — 352 с.
- [2] Харари Ф. Теория графов. — М.: Мир, 1973. — 300 с.
- [3] Емеличев В.А., Мельников О.И., Сарванов В.И., Тышкевич Р.И. Лекции по теории графов. — М.: Наука, 1990. — 384 с.
- [4] Евстигнеев В.А., Касьянов В.Н. Толковый словарь по теории графов. — Н-ск.: Наука, 1999. — 291 с.
- [5] Иванов Б.Н. Дискретная математика. Алгоритмы и программы. — М.: Лаборатория Базовых Знаний, 2001. — 288 с.
- [6] Носов В.И., Фадеева Н.Е., Минеева Т.В., Ахтырский В.Н. Новый подход к планированию сети телевизионного и звукового вещания.// Электросвязь. — 1989. — № 9. — с.18-21.
- [7] Носов В.И., Мельников Л. С. Использование обобщенной раскраски графов линейно упорядоченным множеством цветов в сетях радиовещания./ Всесоюзное совещание СОРАН СССР, Новосибирск, 1989. Тезисы. — с.129 -131.
- [8] Носов В.И., Мельников Л. С. Определение оптимального шага сетки частот в сети звукового вещания./ НТК СОРАН СССР, ЦП НТОР-ЭС, Новосибирск, 1991. Тезисы. — с.221-228.
- [9] Носов В.И., Мельников Л. С. Оптимизация параметров ОБЧ ЧМ сети на основе обобщенной раскраски графов./ Российская НТК, Новосибирск, 1993. Тезисы. — с.130.
- [10] V.I. Nosov, L.S. Mel'nikov. ALLOCATION OF FREQUENCIES IN A NETWORK OF TERRESTRIAL DIGITAL AUDIO BROADCASTING /2001 MICROWAVE ELECTRONICS: Measurement, Identification, Applications, CONFERENCE PROCEEDING MEMIA'2001, September 18-20, 2001, Novosibirsk, Russia/ – P.171-174.
- [11] Мельников Л. С., Носкова Н.В., Носов В.И., Использование метода ветвей и границ при планировании сетей радиосвязи./ Международный научно-технический семинар "Перспективы развития средств и систем телекоммуникаций", Новосибирск, 2003. — с.82-87.
- [12] Бернштейн Т.В., Макаров Р.Н., Храмова Т.В., Дискретная математика (учебное пособие) — СибГУТИ, Новосибирск, 2004 — 91 с.

Татьяна Викторовна Бернштейн  
Владимир Иванович Носов  
Наталья Владимировна Носкова  
Татьяна Викторовна Храмова

## **ЭЛЕМЕНТЫ ТЕОРИИ ГРАФОВ**

УЧЕБНОЕ ПОСОБИЕ

Корректор

Редактор: В.И. Носов

Подписано в печать..... Формат бумаги 60x84/16

Бумага писчая №1. Уч. изд. л. .... Тираж 450 экз. Заказ №

СибГУТИ, 630102, г. Новосибирск, ул. Кирова, 86