Министерство образования Республики Беларусь
Учреждение образования «Витебский государственный университет имени П.М. Машерова»
Кафедра алгебры и методики преподавания математики

ЭЛЕМЕНТЫ ТЕОРИИ ГРАФОВ

Методические рекомендации

Витебск
ВГУ имени П.М. Машерова
2020

УДК 519.17(075.8)
ББК 22.181.2я73
Э45

Печатается по решению научно-методического совета учреждения образования «Витебский государственный университет имени П.М. Машерова». Протокол № 6 от 18.06.2020.

Составители: доцент кафедры алгебры и методики преподавания математики ВГУ имени П.М. Машерова, кандидат физико-математических наук **А.П. Мехович;** преподаватель кафедры алгебры и методики преподавания математики ВГУ имени П.М. Машерова **Т.Б. Караулова**

Р е ц е н з е н т :
заведующий кафедрой информатики и информационных технологий ВГУ имени П.М. Машерова, кандидат физико-математических наук
Е.А. Витько

Э45 **Элементы теории графов :** методические рекомендации / сост.: А.П. Мехович, Т.Б. Караулова. – Витебск : ВГУ имени П.М. Машерова, 2020. – 48 с.

Издание содержит материал по следующим разделам теории графов: изоморфизм графов, расстояния в графе, задача о кратчайшем пути, задача о минимальном остовном дереве, гамильтоновы и эйлеровы графы, раскраска графа, компоненты связности.

Адресовано студентам IT-специальностей ВГУ имени П.М. Машерова и может успешно использоваться для подготовки к занятиям по дисциплинам: «Теория графов», «Теория графов в управлении проектами», «Дискретная математика и математическая логика», «Исследование операций», а также при изучении спецкурсов, где затрагиваются вопросы теории графов и разработки алгоритмов на графах.

УДК 519.17(075.8)
ББК 22.181.2я73

СОДЕРЖАНИЕ

Предисловие

Теория графов – один из важнейших математических инструментов, который широко используется в теории автоматов, исследовании операций, теории информации, геоинформационных системах, теории связи, теории кодирования, теории игр и др.

Основоположником теории графов считают Л. Эйлера, который решил «задачу о кенигсбергских мостах», а термин «граф» был введен спустя 200 лет (в 1936 г.) Д. Кёнигом.

Граф можно определить как совокупность двух множеств: множества вершин V и множества ребер E, между которыми определено отношение инцидентности, причем каждый элемент $e \in E$ инцидентен ровно двум элементам v_i, $v_j \in V$.

В виде графов можно интерпретировать схемы дорог, географические карты, электрические цепи, блок-схемы компьютерных программ, молекулы химических соединений и многое другое.

В начале каждого параграфа излагаются основные теоретические понятия и фундаментальные факты теории графов. Затем помещены разобранные решения задач. В конце параграфа предложены задачи для самостоятельного решения по следующим разделам теории графов: способы задания графов, изоморфизм графов, расстояния в графах, задача о кратчайшем пути, задача о минимальном остовном дереве, гамильтоновы и эйлеровы графы, раскраска графов, компоненты связности.

В методических рекомендациях очень кратко приведены сведения о тех, чьи имена встречаются на страницах.

Адресовано студентам Витебского государственного университета имени П.М. Машерова, обучающимся по специальностям:

1-26 03 01 Управление информационными ресурсами;

1-31 03 07-01 02 Прикладная информатика (программное обеспечение компьютерных систем);

1-31 03 07-03 01 Прикладная информатика (веб-программирование и компьютерный дизайн);

1-31 03 03-02 Прикладная математика (научно-педагогическая деятельность);

1-40 01 01 03 Программное обеспечение информационных технологий;

1-98 01 01-02 Компьютерная безопасность (радиофизические методы и программно-технические средства) и может успешно использоваться для подготовки к занятиям по дисциплинам: Теория графов, Дискретная математика и математическая логика, Исследование операций, Теория графов в управлении проектами, а также при изучении спецкурсов, где затрагиваются вопросы теории графов и разработки алгоритмов на графах.

§ 1 Изоморфизм графов

Графом называется пара $G = (V, E)$, где V – произвольное конечное множество элементов, E – произвольное семейство пар из V. Множество $V = V(G)$ при этом называется *множеством вершин* графа G, а его элементы – *вершинами*; множество $E = E(G)$ называется *множеством ребер* графа G, а его элементы – *ребрами*. И вершины, и ребра графа G называются его элементами. Поэтому если u – вершина графа G, а e – ребро G, то вместо $u \in V(G)$, $e \in E(G)$ можно писать $u \in G$, $e \in G$.

Степенью вершины v графа G называется число инцидентных ей рёбер, т.е. число рёбер, выходящих из данной вершины.

В случае псевдографов каждая петля учитывается дважды при подсчете степени вершины.

Обозначается степень вершины v графа G: $deg_G(v)$ или просто $deg(v)$, если ясно, о каком графе G идет речь.

Последовательность степеней вершин графа G, записанная в каком-либо порядке, называется ***степенной последовательностью*** *графа* G.

Два графа G и H называются ***изоморфными,*** если существует биекция $f : V(G) \to V(H)$, сохраняющая смежность, т.е. такое биективное отображение, при котором образы вершин v и u графа G смежны в H тогда и только тогда, когда прообразы u и v смежны в графе G. Отображение f, обладающее указанным свойством, называется ***изоморфизмом***.

Если графы G и H изоморфны, то пишут $G \cong H$.

Для изоморфных графов верно следующее:

$$G_1 \cong G_2 \Rightarrow |V(G_1)| = |V(G_2)|, |E(G_1)| = |E(G_2)|,$$
$$\Sigma \deg_{G_1} v = \Sigma \deg_{G_2} v.$$

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

Пример 1.1. Покажем, что граф G изоморфен графу H (рис. 1.1).

Граф G Граф H

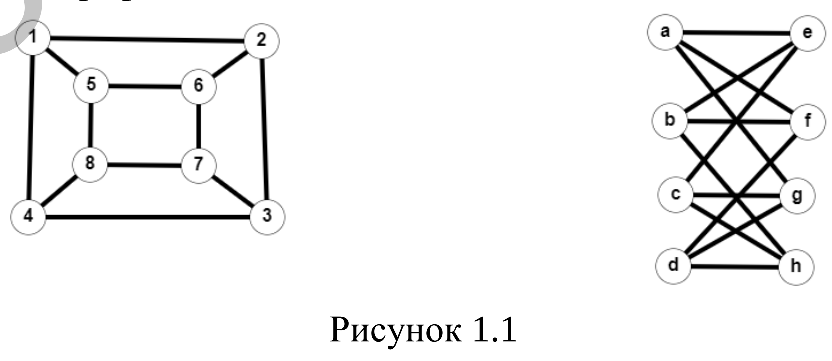


Рисунок 1.1

$|V(G)| = |V(H)| = 8$, $|E(G)| = |E(H)| = 12$.

$\deg(1) = 3$	$\deg(a) = 3$
$\deg(2) = 3$	$\deg(b) = 3$
$\deg(3) = 3$	$\deg(c) = 3$
$\deg(4) = 3$	$\deg(d) = 3$
$\deg(5) = 3$	$\deg(e) = 3$
$\deg(6) = 3$	$\deg(f) = 3$
$\deg(7) = 3$	$\deg(g) = 3$
$\deg(8) = 3$	$\deg(h) = 3$

$$\Sigma \deg_G v = \Sigma \deg_H v = 24.$$

Значит, у графов G и H одинаковые наборы степеней вершин. Зададим отображение

$$f(\mathrm{g}) = 1,\ f(\mathrm{c}) = 2,\ f(\mathrm{a}) = 5,\ f(\mathrm{e}) = 6,\ f(\mathrm{f}) = 8,\ f(\mathrm{h}) = 3,\ f(\mathrm{b}) = 7,\ f(\mathrm{d}) = 4.$$

Значит, графы G и H изоморфны.

Графы, изображенные на рис. 1.2 не изоморфны, потому что они имеют неодинаковое число вершин. Не изоморфны и графы рис. 1.3, так как у них неодинаковое число ребер.

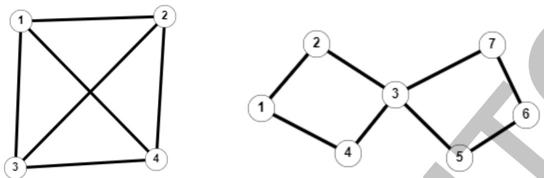


Рисунок 1.2

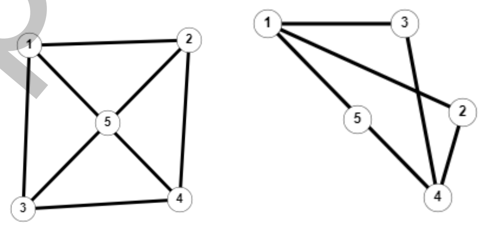


Рисунок 1.3

На рис. 1.4, первый граф имеет последовательность из восьми смежных ребер (т. е. ребер, попарно имеющих общую вершину): {1,2}, {2,3}, {3,4}, {4,8}, {8,7}, {7,6}, {6,5}, {5,1}, в то время как на втором графе такой последовательности нет. Значит, как бы мы ни обозначили вершины второго графа, мы не сможем для каждой пары соединенных ребром вершин одного графа указать во втором соответствующую пару вершин, тоже соединенных ребром.

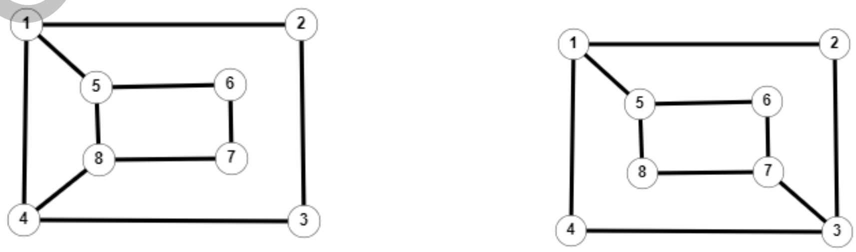
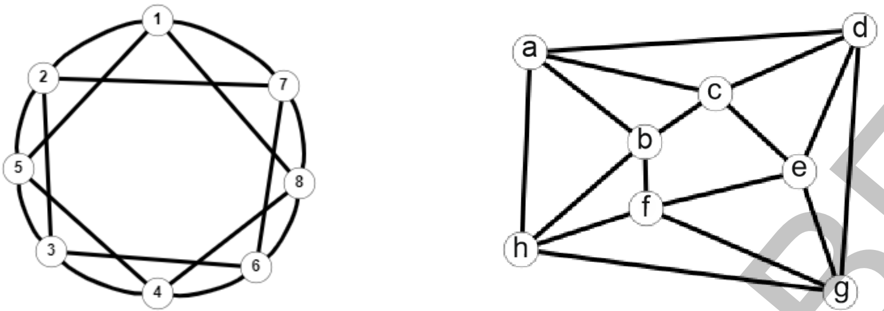


Рисунок 1.4

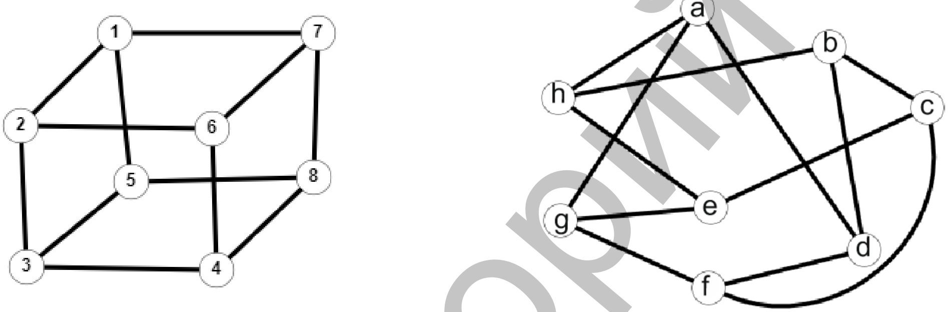
Задачи для самостоятельного решения

1. Доказать, что следующие графы являются изоморфными.

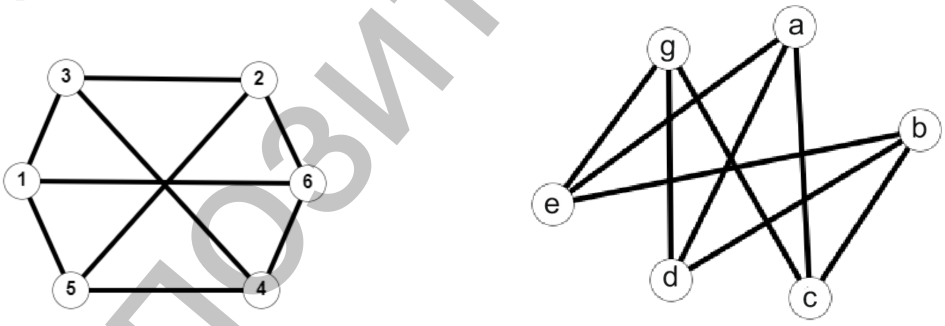
Вариант 1



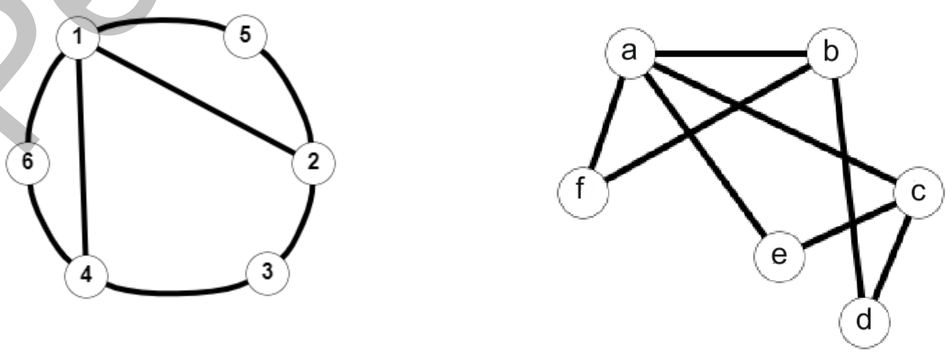
Вариант 2



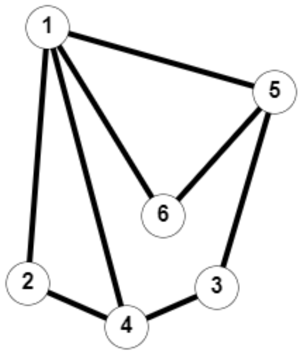
Вариант 3

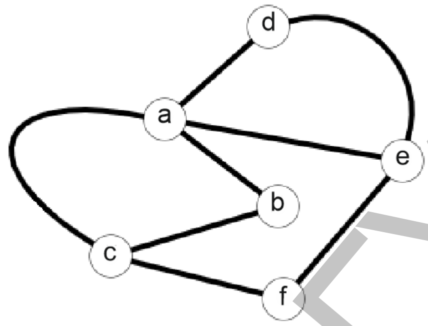


Вариант 4

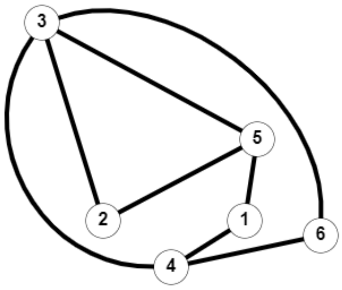


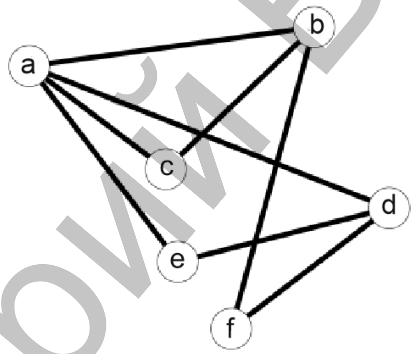
Вариант 5



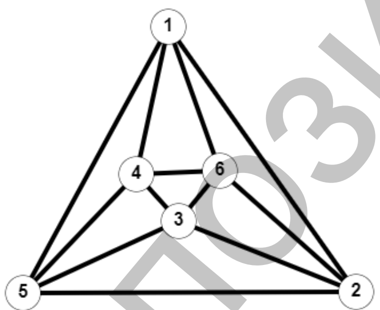


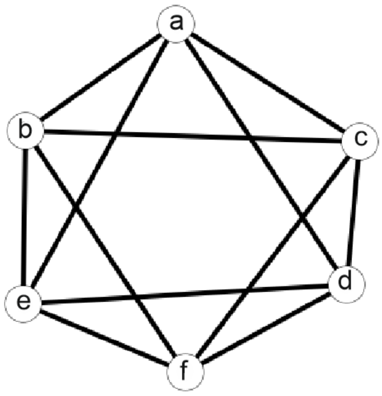
Вариант 6



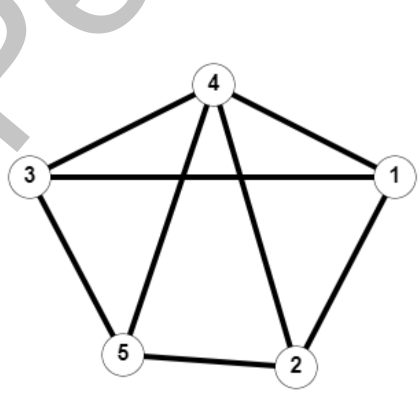


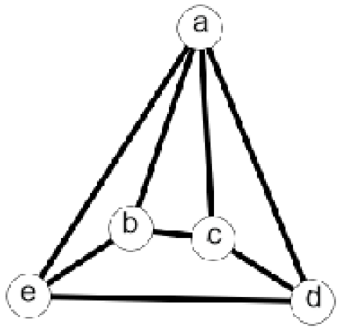
Вариант 7



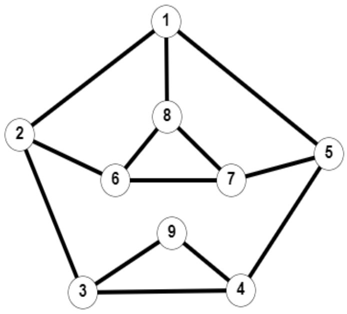


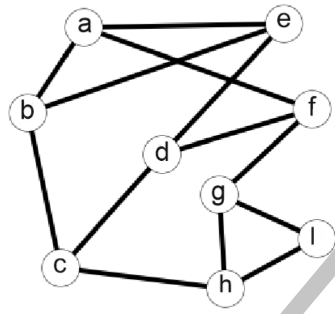
Вариант 8





Вариант 9





§ 2 Расстояния в графе

Пусть G – мульти- или псевдограф. Последовательность вершин и рёбер

$$v_1, e_1, v_2, e_2, \ldots, e_n, v_{n+1}$$

такая, что $e_i = \{v_i, v_{i+1}\}$ – ребро в графе G, соединяющее v_i, с v_{i+1} называется (v_1, v_{n+1})-***маршрутом***. Вершина v_1 при этом называется началом маршрута, а v_{n+1} – концом маршрута. Число рёбер n в маршруте называется ***длиной маршрута.***

Последовательность вершин и дуг $v_1e_1v_2e_2v_3...v_ne_ne_{n+1}$ такая, что $e_i = (v_i, v_{i+1})$, называется ***путем*** из v_1 в v_{n+1}.

Пусть G – связный граф и u, v – его вершины.

Длина кратчайшего (u,v)-маршрута (понятно, что он является простой цепью) называется ***расстоянием*** между u и v. Обозначается расстояние между вершинами u и v графа G: $d(u,v)$.

По определению полагают, что $d(u,u) = 0$ для всякой вершины u и $d(v,w) = \infty$, если не существует такого пути или маршрута.

Удалённостью (или, иначе, ***эксцентриситетом)*** вершины v графа G называется наибольшее из расстояний от данной вершины до других вершин графа G:

$$e(v) = \max_{u \in V(G)} d(v,u).$$

Радиусом графа G называется наименьшая из удалённостей его вершин:

$$R(G) = \min_{v \in V(G)} e(v).$$

Диаметром графа G называется наибольшая из удалённостей его вершин:

$$D(G) = \max_{v \in V(G)} e(v).$$

Вершина v графа G, удалённость которой минимальная (и значит, равна радиусу), называется ***центром*** графа G.

Вершина, удалённость которой максимальна в графе (и значит, равна диаметру), называется ***периферийным центром.***

Пример 2.1. Найдем радиус, диаметр, центры и периферийные центры в ориентированном графе G, изображенном на рис. 2.1. В данной задаче количество вершин $n=7$, следовательно, матрицы смежности и минимальных расстояний между вершинами ориентированного графа G будут иметь размерность 7×7.

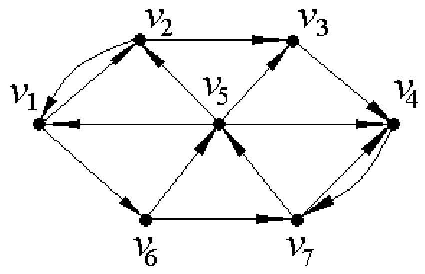


Рисунок 2.1

Составляем матрицу смежности:

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

Начинаем заполнять матрицу $A(G)$ минимальных расстояний:

Сначала ставим нули по главной диагонали и $a_{ij}=m_{ij}$, если $m_{ij}=1$, (т.е. переносим единицы из матрицы смежности).

Рассматриваем первую строку. Здесь есть две единицы, то есть из первой вершины за один шаг можно попасть во вторую и шестую. Из второй вершины можно попасть за один шаг в третью (путь в первую вершину нас не интересует), следовательно, можно записать $a_{13}=2$. Из шестой вершины можем добраться за один шаг в пятую и седьмую, а значит, $a_{15}=2$, $a_{17}=2$. Теперь ищем маршруты, исходящие из первой вершины, состоящие из 3 шагов: за 2 шага идем в третью вершину, оттуда за один шаг попадаем в четвертую, поэтому $a_{14}=3$. В итоге получаем следующую матрицу:

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

Для каждой вершины заданного ориентированного графа найдем удалённость: $e(v_1)=3$, $e(v_2)=3$, $e(v_3)=5$, $e(v_4)=4$, $e(v_5)=2$, $e(v_6)=2$, $e(v_7)=3$. Значит, радиусом графа G будет

$$R(G)=\min_{v\in V(G)} e(v)=2.$$

Диаметр графа

$$D(G)=\max_{v\in V(G)} e(v)=5.$$

Соответственно, центрами графа G будут вершины v_5 и v_6, так как величины их эксцентриситетов совпадают с величиной радиуса $R(G)$, а периферийным центром – v_3.

§ 3 Матрицы смежности и инцидентности

Пусть G – неориентированный простой граф порядка n, $V(G)$ = {v_1, v_2, v_3, ..., v_n}. ***Матрицей смежности*** графа G называется $n \times n$-матрица $M(G) = (m_{ij})$ такая, что

$$m_{ij}=\begin{cases} 1, \textit{ если вершина } v_i \textit{ смежна с вершиной } v_j; \\ 0, \textit{ в противном случае.} \end{cases}$$

Легко видеть, что матрица смежности простого графа G является симметричной, с нулями на главной диагонали. Число единиц в каждой строке (каждом столбце) равно степени соответствующей вершины. Понятно, что и всякой матрице с указанными свойствами соответствует некоторый простой граф. Таким образом, матрица смежности является одним из способов задания графов.

Для мульти- и псевдо- неориентированных графов матрица смежности определяется следующим образом:

$$m_{ij}=\begin{cases} \textit{число ребер, соединяющих вершины } v_i \textit{ и } v_j, \textit{ если } i\neq j \\ 2\cdot(\textit{число петель, инцидентных вершине } i), \textit{ если } i=j. \end{cases}$$

Матрица смежности для мульти- и псевдо- ориентированных графов:

$$m_{ij}=\begin{cases} 1, \textit{ если } (v_i,v_j) \textit{ является дугой } (v_i - \textit{начало}, v_j - \textit{конец}); \\ 0, \textit{ в противном случае.} \end{cases}$$

Таким образом, всякая бинарная матрица является матрицей смежности соответствующего ориентированного графа. Например, матрице, приведенной на рис. 3.1 соответствует граф, изображенный на рис. 3.2.

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

Рисунок 3.1

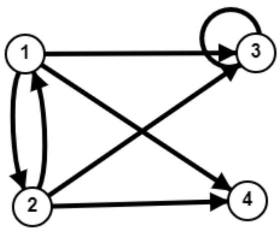


Рисунок 3.2

Пусть G – простой (n, m)-граф, $V(G) = \{v_1, v_2, v_3, \ldots, v_n\}$, $E(G) = \{e_1, e_2, e_3, \ldots, e_m\}$. ***Матрицей инцидентности*** графа G называется $n \times m$-матрица $I(G) = (I_{ij})$ такая, что

$$I_{ij} = \begin{cases} 1, \textit{ если вершина } v_i \textit{ инцидентна ребру } e_j; \\ 0, \textit{ в противном случае.} \end{cases}$$

Понятно, что такая матрица имеет ровно по две единицы в каждом столбце (ведь всякое ребро имеет два конца – две инцидентные данному ребру вершины). Число единиц в каждой строке матрицы инцидентности равно степени соответствующей вершины.

Для ориентированного графа:

$$m_{ij} = \begin{cases} 1, \textit{ если вершина } v_i - \textit{начало дуги } e_j; \\ -1, \textit{ если вершина } v_i - \textit{конец дуги } e_j; \\ 0, \textit{ в противном случае, если вершина } v_i \textit{ и дуга } e_j \textit{ не инцидентны.} \end{cases}$$

Например, матрице, приведенной на рис. 3.3 соответствует граф на рис. 3.4.

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 & -1 \\ 0 & -1 & 0 & 1 & 0 & 1 & 1 \\ -1 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & -1 & -1 & 0 \end{pmatrix}$$

Рисунок 3.3

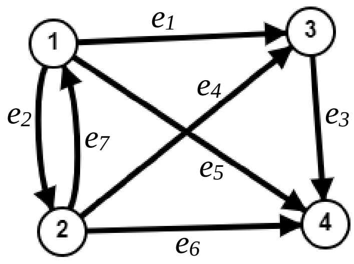
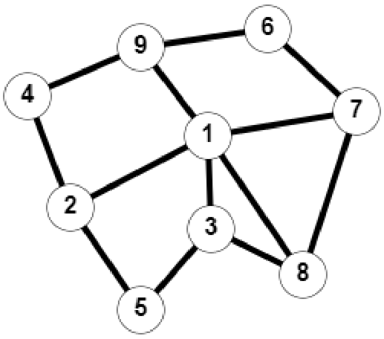


Рисунок 3.4

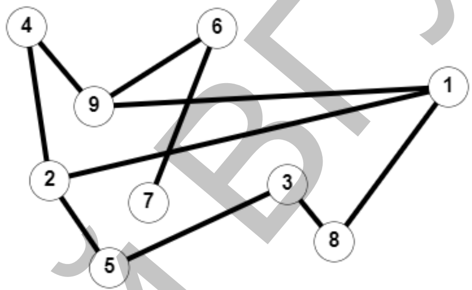
Задачи для самостоятельного решения

1. На заданных неориентированных графах:
а) найти степенную последовательность графа;
б) записать матрицы смежности и инцидентности;
в) найти радиус, диаметр, центры и периферийные центры графа.

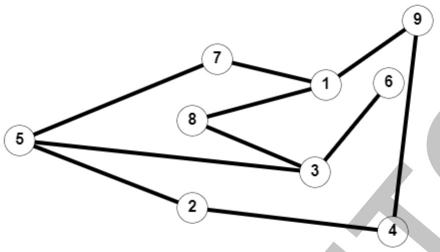
Вариант 1



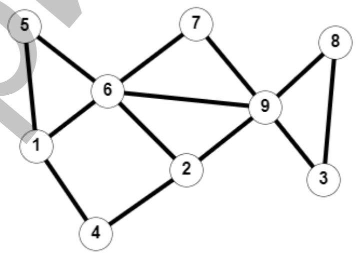
Вариант 2



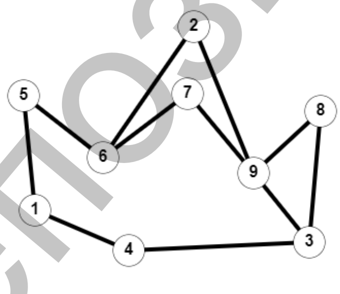
Вариант 3



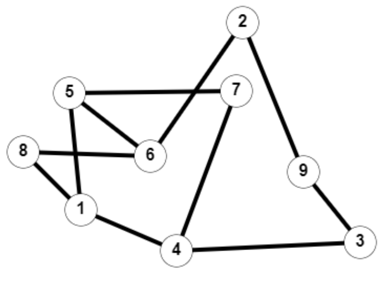
Вариант 4



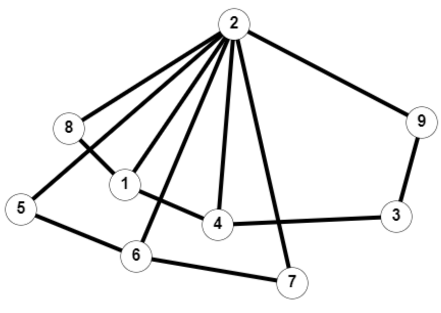
Вариант 5



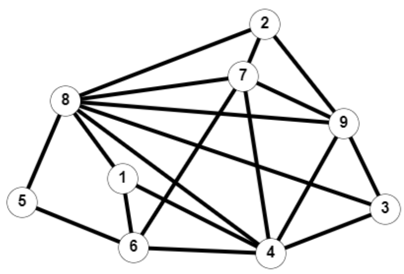
Вариант 6



Вариант 7

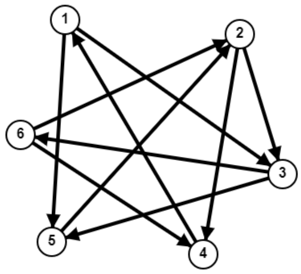


Вариант 8

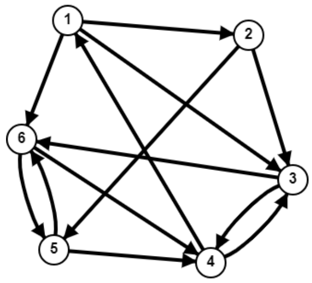


2. На заданных ориентированных графах выполнить пункты а) – в) из задания 1.

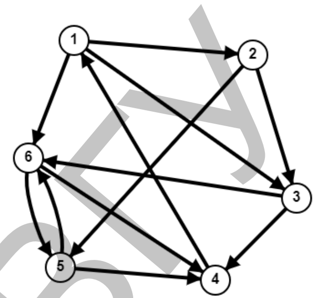
Вариант 1



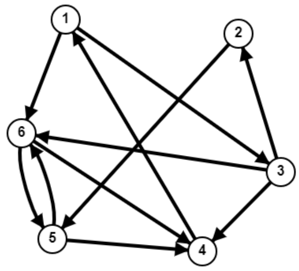
Вариант 2



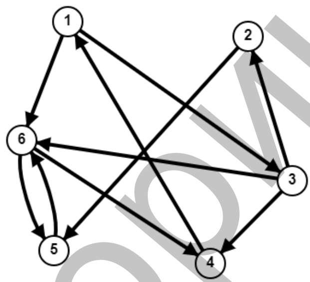
Вариант 3



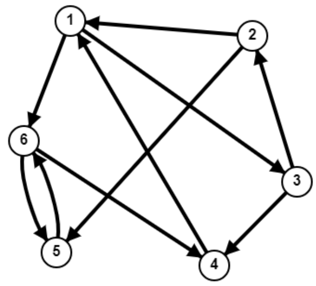
Вариант 4



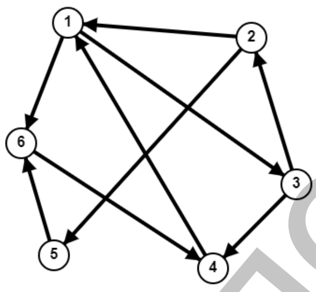
Вариант 5



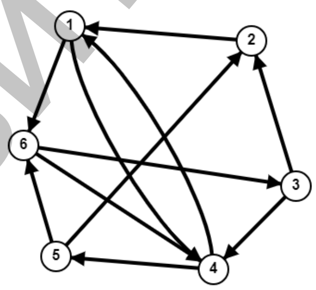
Вариант 6



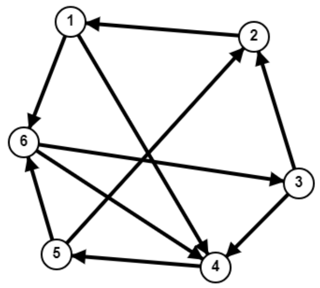
Вариант 7



Вариант 8



Вариант 9



§ 4 Поиски в ширину и глубину

Обходом графа называется некоторое систематическое перечисление его вершин или ребер. Вершинными обходами являются, например, «поиск в глубину» и «поиск в ширину», а реберным – нахождение эйлерова цикла.

Поиск в ширину.

Что же это такое? Давайте немного отойдем от формального описания графов, и представим себе такую картину. Выложим на земле веревки, пропитанные чем-нибудь горючим, одинаковой длины так, чтобы ни одна из них не пересекалась, но некоторые из них касались концами друг с другом. А теперь подожжем один из концов. Как будет вести себя огонь? Он равномерно будет перекидываться по веревкам на соседние пересечения, пока не загорится все. Именно так в жизни будет выглядеть обход графа в ширину. Теперь опишем более формально. Пусть дан связный неориентированный граф.

1) Выходя из начальной вершины, помечаем все вершины из ее окрестности как вершины первого «уровня».

2) У каждой из вершин первого уровня помечаем еще не помеченные вершины ее окрестности как вершины второго уровня и т. д.

3) Процесс продолжаем до тех пор, пока все вершины не получат метки.

Если граф несвязный, то обходим вершины, достижимые из начальной. В орграфе учитываем направление дуг.

Поиск в глубину.

Пусть дан связный неориентированный граф.

1) Выходя из начальной вершины, строим простую цепь, пройденным вершинам приписываем метки.

2) Цепь строим до тех пор, пока не встретим уже помеченную вершину или не окажемся в висячей вершине. В этом случае возвращаемся на шаг назад и выбираем ребро, ведущее к непомеченной вершине.

3) Процесс продолжаем до тех пор, пока всем вершинам не будут присвоены метки.

Пример 4.1. В заданном матрицей смежности орграфе обойти все вершины, используя поиск в глубину.

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

Решение. Проводим поиск из первой вершины. Вершина с наименьшим номером, в которую ведет дуга из v_1, это v_3. Из v_3 исходит дуга в v_2, из v_2 в v_5. Из v_5 поиск ведет в v_1 (наименьший номер из смежных), но эта вершина уже помечена, следовательно, останавливаем свой выбор на вершине v_4. Из вершины v_4 дуги ведут в v_1 и v_5, обе эти вершины уже помечены. В соответствии с алгоритмом, возвращаемся на шаг назад, в вершину v_5 Из пятой вершины дуги ведут также только в помеченные вершины. Возвращаясь еще на шаг назад в v_3, обнаружим дугу в непомеченную еще вершину v_6. Из v_6 переходим в v_7. Поскольку все вершины теперь помечены, поиск останавливает работу. Поиск проиллюстрирован на рис. 4.1, в скобках указана метка вершины.

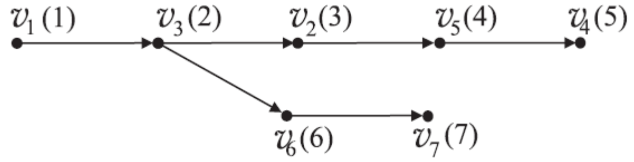


Рисунок 4.1

Пример 4.2. В заданном матрицей смежности $A(\vec{G})$ орграфе провести поиск в глубину из вершины 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. Из вершины v_3, которой присваивается метка 1, переходим в вершину с наименьшим номером v_4 (присваиваем ей метку 2). Из v_4 ведут дуги в v_1, v_3 и v_5. Наименьший номер у v_1 (присваиваем ей метку 3). Из v_1 ведут дуги в v_4, и v_6, но v_4 рассмотрена ранее, поэтому метку 4 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 5. Из v_2 ведут дуги в v_1, v_3 и v_7. Присваиваем вершине v_7 метку 6. 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 4, то метка v_5 7. Все вершины графа помечены. Дерево, иллюстрирующее поиск в глубину представлено на рис. 4.2.

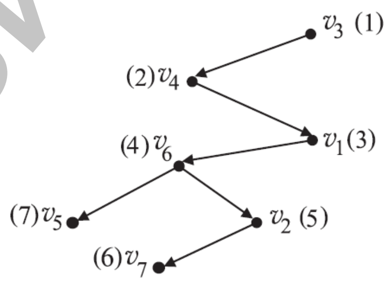


Рисунок 4.2

Пример 4.3. В заданном 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 проиллюстрирован на рис. 4.3, в скобках указана метка вершины.

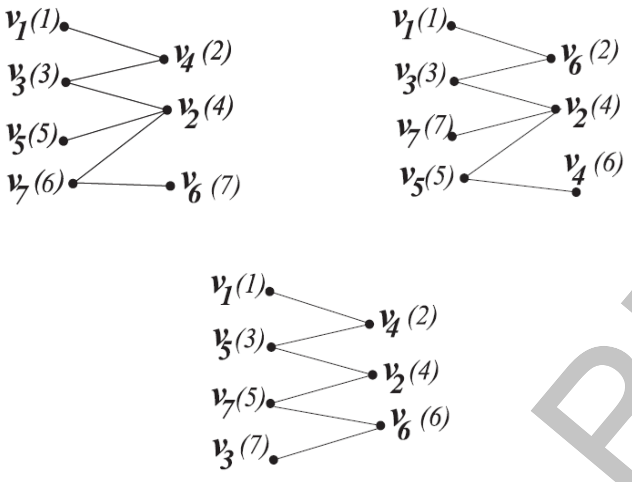


Рисунок 4.3

Пример 4.4. 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 представлены на рис. 4.4. В скобках указан уровень (метка) вершины. Метка вершины равна расстоянию от нее до вершины, из которой идет поиск (в данном случае до первой).

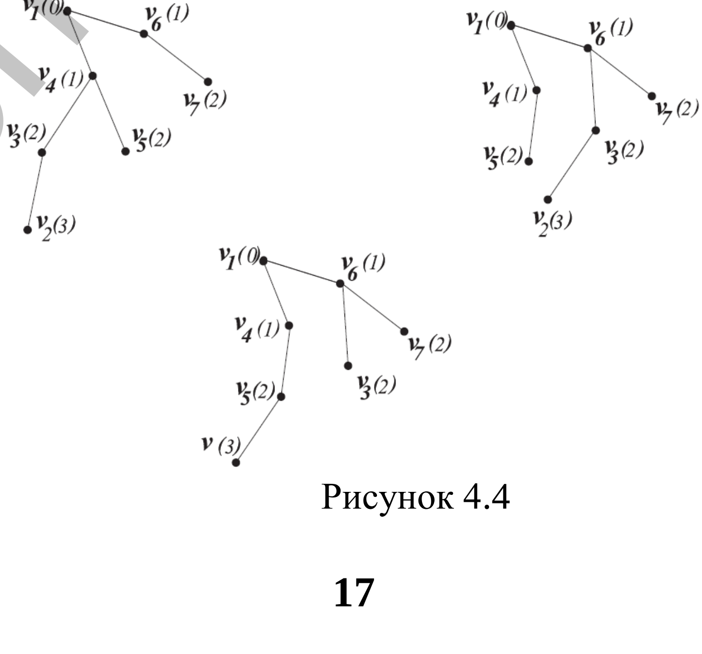


Рисунок 4.4

<u>Задачи для самостоятельного решения</u>

1. По заданной матрице смежности определить число циклов длины 3 и 4.
2. Записать матрицу инцидентности.
3. Построить рисунок графа.
4. Осуществить поиск в ширину и поиск в глубину из вершины ____ .

а) $\begin{pmatrix} 0&0&0&1&0&0&0&0&0 \\ 0&0&1&1&0&1&0&0&0 \\ 0&1&0&0&1&1&0&0&0 \\ 1&1&0&0&1&0&0&1&0 \\ 0&0&1&1&0&1&1&1&1 \\ 0&1&1&0&1&0&0&1&0 \\ 0&0&0&0&1&0&0&1&0 \\ 0&0&0&1&1&1&1&0&1 \\ 0&0&0&0&1&0&0&1&0 \end{pmatrix}$ б) $\begin{pmatrix} 0&1&0&0&0&0&0&0&0 \\ 1&0&1&1&1&1&0&0&0 \\ 0&1&0&0&1&1&0&0&0 \\ 0&1&0&0&1&0&0&1&0 \\ 0&1&1&1&0&0&1&1&1 \\ 0&1&1&0&0&0&0&1&1 \\ 0&0&0&0&1&0&0&1&0 \\ 0&0&0&1&1&1&1&0&0 \\ 0&0&0&0&1&1&0&0&0 \end{pmatrix}$ в) $\begin{pmatrix} 0&1&0&1&0&0&0&0&0 \\ 1&0&0&1&0&1&0&0&0 \\ 0&0&0&0&1&1&0&0&0 \\ 1&1&0&0&1&0&1&1&0 \\ 0&0&1&1&0&1&1&0&1 \\ 0&1&1&0&1&0&0&1&1 \\ 0&0&0&1&1&0&0&0&0 \\ 0&0&0&1&0&1&0&0&1 \\ 0&0&0&0&1&1&0&1&0 \end{pmatrix}$

г) $\begin{pmatrix} 0&1&0&0&0&0&0&0&0 \\ 1&0&1&1&0&1&0&0&0 \\ 0&1&0&0&1&1&0&0&0 \\ 0&1&0&0&1&0&1&1&0 \\ 0&0&1&1&0&1&1&0&1 \\ 0&1&1&0&1&0&0&1&1 \\ 0&0&0&1&1&0&0&0&0 \\ 0&0&0&1&0&1&0&0&1 \\ 0&0&0&0&1&1&0&1&0 \end{pmatrix}$ д) $\begin{pmatrix} 0&1&0&1&0&0&0&0&0 \\ 1&0&1&1&0&1&0&0&0 \\ 0&1&0&0&1&1&0&0&0 \\ 1&1&0&0&1&0&1&1&0 \\ 0&0&1&1&0&0&1&1&1 \\ 0&1&1&0&0&0&0&1&0 \\ 0&0&0&1&1&0&0&0&0 \\ 0&0&0&1&1&1&0&0&1 \\ 0&0&0&0&1&0&0&1&0 \end{pmatrix}$ е) $\begin{pmatrix} 0&1&0&1&0&0&0&0&0 \\ 1&0&0&1&0&1&0&0&0 \\ 0&0&0&0&1&1&0&0&0 \\ 1&1&0&0&1&0&1&1&0 \\ 0&0&1&1&0&1&1&0&1 \\ 0&1&1&0&1&0&0&1&1 \\ 0&0&0&1&1&0&0&0&0 \\ 0&0&0&1&0&1&0&0&1 \\ 0&0&0&0&1&1&0&1&0 \end{pmatrix}$

ж) $\begin{pmatrix} 0&1&0&0&1&0&0&0&0 \\ 1&0&1&1&1&1&0&0&0 \\ 0&1&0&0&1&1&0&0&0 \\ 0&1&0&0&1&0&1&1&0 \\ 1&1&1&1&0&0&1&1&1 \\ 0&1&1&0&0&0&0&1&0 \\ 0&0&0&1&1&0&0&0&0 \\ 0&0&0&1&1&1&0&0&1 \\ 0&0&0&0&1&0&0&1&0 \end{pmatrix}$ з) $\begin{pmatrix} 0&1&0&0&1&0&0&0&0 \\ 1&0&1&1&1&1&0&0&0 \\ 0&1&0&0&1&0&0&0&0 \\ 0&1&0&0&1&0&1&1&0 \\ 1&1&1&1&0&1&1&0&1 \\ 0&1&0&0&1&0&0&1&1 \\ 0&0&0&1&1&0&0&0&0 \\ 0&0&0&1&0&1&0&0&1 \\ 0&0&0&0&1&1&0&1&0 \end{pmatrix}$ и) $\begin{pmatrix} 0&0&0&1&1&0&0&0&0 \\ 0&0&1&1&1&1&0&0&0 \\ 0&1&0&0&1&0&0&0&0 \\ 1&1&0&0&1&0&1&1&0 \\ 1&1&1&1&0&1&1&1&1 \\ 0&1&0&0&1&0&0&1&1 \\ 0&0&0&1&1&0&0&0&0 \\ 0&0&0&1&1&1&0&0&0 \\ 0&0&0&0&1&1&0&0&0 \end{pmatrix}$

§ 5 Связность в графах

Говорят, что вершина w графа G достижима из вершины v, если либо $w = v$, либо существует путь (маршрут) из v в w.

Орграф называется ***сильно связным***, если для любых двух его вершин v, u существует путь, соединяющий v и u.

Компонентой сильной связности орграфа *G* называется его сильно связный подграф, не являющийся собственным подграфом никакого другого сильно связного подграфа графа *G*.

Пусть $A = A(G)$ – матрица смежности ориентированного графа $G = (V, E)$, где $V = \{v_1, ..., v_n\}$. Обозначим через $a_{ij}^{(k)}$ элемент матрицы A^k. Тогда элемент $a_{ij}^{(k)}$ равен числу всех путей длины *k* из v_i в v_j.

Матрицей достижимости орграфа *G* называется $n \times n$-матрица $T(G) = (t_{ij})$ такая, что

$$t_{ij} = \begin{cases} 1, \textit{если вершина } v_j \textit{ достижима из вершины } v_i; \\ 0, \textit{ в противном случае}. \end{cases}$$

Матрицей сильной связности орграфа *G* называется $n \times n$-матрица $S(G) = (s_{ij})$ такая, что

$$s_{ij} = \begin{cases} 1, \textit{ если } v_j \textit{ достижима из вершины } v_i \textit{ и } v_i \textit{ достижима из } v_j; \\ 0, \textit{ в противном случае}. \end{cases}$$

Матрица связности неориенированного графа *G* − квадратная матрица *S*(*G*)=(s_{ij}) порядка *n*, элементы которой равны

$$s_{ij} = \begin{cases} 1, & \textit{если } \exists \textit{ маршрут, соединяющий } v_j \textit{ и } v_i, \\ 0, & \textit{в противном случае.} \end{cases}$$

Алгоритм выделения компонент сильной связности.

Пусть $G = (V, E)$ – орграф, $V = \{v_1, ..., v_n\}$. Для выделения компонент сильной связности выполняем следующую последовательность действий:

1. Составляем матрицу смежности $A = A(G)$.

2. Находим матрицу достижимости по формуле

$$T(G) = sign\,(E + A + A^2 + ... + A^{n-1}).$$

3. Находим матрицу сильной связности по формуле $S(G) = T(G)\ \&\ T^T(G)$ (T^T – транспонированная матрица, & – поэлементное умножение).

Используя матрицу сильной связности, выделяем компоненты сильной связности по следующему алгоритму.

а) Присваиваем *p* = 1 (*p* — количество компонент сильной связности), $S_1 = S(G)$.

б) Включаем в множество вершин V_p компоненты сильной связности G_p вершины, соответствующие единицам первой строки матрицы S_p. В качестве матрицы $A(G_p)$ возьмем подматрицу матрицы $A(G)$, состоящую из элементов матрицы M, находящихся на пересечении строк и столбцов, соответствующих вершинам из V_p.

в) Вычеркиваем из S_p строки и столбцы, соответствующие вершинам из V_p. Если не остается ни одной строки (и столбца), то *p* — количество компонент сильной связности. В противном случае обозначим оставшуюся после вычеркивания срок и столбцов матрицу как S_{p+1}, присваиваем $p = p + 1$ и переходим к п. б) алгоритма.

Задачи для самостоятельного решения

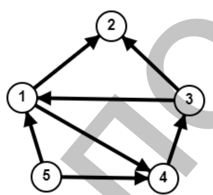
1. Выделить компоненты сильной связности в орграфе, заданном матрицей смежности.

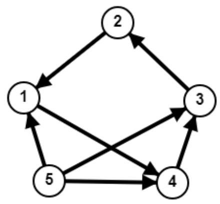
а) $A(\vec{G}) = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$ б) $A(\vec{G}) = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\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

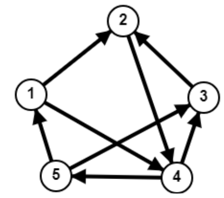
г) $A(\vec{G}) = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix}$ д) $A(\vec{G}) = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\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

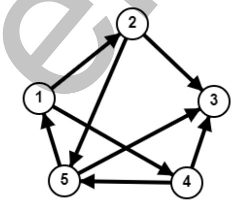
ж) $A(\vec{G}) = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix}$ з) $A(\vec{G}) = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\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

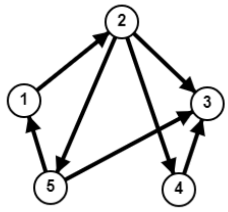
2. С помощью матрицы смежности найти компоненты сильной связности ориентированного графа.

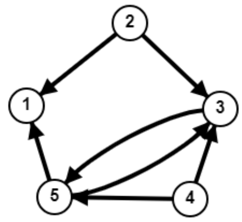
а) 

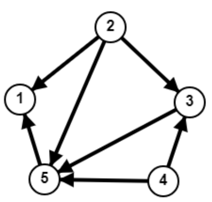
б) 

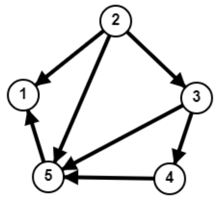
в) 

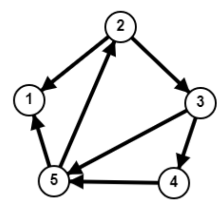
г) 

д) 

е) 

ж) 

з) 

и) 

§ 6 Деревья

1. Код Прюфера

Деревом называется связный граф без циклов.

Свойства деревьев:

1. Любая пара вершин соединена единственным маршрутом.
2. Количество ребер меньше на одну чем вершин.
3. Если в дерево добавить хотя бы одно ребро, то появиться цикл.

Деревья – очень удобный инструмент представления информации самого разного вида. Они отличаются от простых графов тем, что при обходе дерева невозможны циклы. Это делает графы очень удобной формой организации данных для различных алгоритмов.

Для представления деревьев можно использовать те же приёмы, что и для представления графов общего вида – матрицы смежности и инцидентности. Но используя особенные свойства деревьев, можно предложить более эффективный способ – так называемый код Прюфера[1].

Пусть $T=(V,E)$ – дерево, вершины которого занумерованы числами $1,\ldots,n$. Пусть a_1 – вершина степени 1 с наименьшим номером, b_1 – смежная с ней вершина. Удалив из T вершину a_1 и ребро $\{a_1, b_1\}$, получим граф T_1, к которому также применим описанную процедуру. Повторяем ее до тех пор, пока после удаления вершины a_{n-2} и ребра $\{a_{n-2}, b_{n-2}\}$ не получим дерево T_{n-2}, состоящее из одного ребра $\{a_{n-1}, b_{n-1}\}$. Дереву T ставим в соответствие упорядоченный набор чисел, который называется кодом Прюфера.

Таким образом, построение кода Прюфера дерева T ведётся путем последовательного удаления вершин из дерева, пока не останутся только две вершины. При этом каждый раз выбирается концевая вершина с наименьшим номером и в код записывается номер единственной вершины, с которой она соединена. В результате получаем последовательность, составленную из чисел, возможно с повторениями.

Опишем процедуру восстановления дерева T с множеством вершин

$V = \{v_1, v_2, v_3, \ldots, v_n\}$ по коду Прюфера.

Находим наименьший элемент a_1 множества V, не содержащийся в $p(T)$, и восстанавливаем ребро $e_1 = a_1b_1$ дерева T. Далее удаляем a_1 из множества V и первую компоненту b_1 из последовательности $p(T)$.

---

[1] **Хайнц Прюфер** – немецкий математик, работавший в теории абелевых групп, алгебраической теории чисел, теории узлов и теории Штурма-Лиувилля. Прюфер родился в городе Вильгельмсхафен, обучался в Берлинском университете. Он был учеником Фердинанда Фробениуса, а впоследствии – Исая Шура, под руководством которого защитил диссертацию. После этого он работал в Гамбургском, а с 1927 года – в Мюнхенском университете. В 1934 году, в возрасте 37 лет, умер от рака лёгких.

Продолжаем процедуру для оставшихся чисел, пока не будут удалены все компоненты последовательности $p(T)$. Два оставшихся элемента множества V – есть последнее ребро дерева T.

Таким образом, для восстановления дерева по коду $(p_1, p_2, \ldots, p_{n-2})$, заготовим список номеров вершин $(1, 2, \ldots, n)$. Выберем первый номер i_1, который не встречается в коде. Добавим ребро $\{i_1, p_1\}$, после этого удалим i_1 из $(1, 2, \ldots, n)$ и p_1 из $(p_1, p_2, \ldots, p_{n-2})$.

Повторяем процесс до момента, когда код $p(T)$ становится пустым. В этот момент список $(1, 2, \ldots, n)$ содержит ровно два числа i_{n-1} и n. Остается добавить ребро $\{i_{n-1}, n\}$, и дерево построено.

Пример 6.1. Восстановить дерево по коду Прюфера.

Первый шаг

Код Прюфера: $p(T) = (1, 5, 2, 6, 6, 2, 1, 3)$.

Массив вершин дерева: $V = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$.

Минимальная вершина, не содержащаяся в коде Прюфера – это 4.

Список ребер: $\{1, 4\}$.

Второй шаг

Код Прюфера: $p(T) = (1, 5, 2, 6, 6, 2, 1, 3)$.

Массив вершин дерева: $V = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$.

Минимальная вершина, не содержащаяся в коде Прюфера – это 7.

Список ребер: $\{1, 4\}, \{5, 7\}$.

Третий шаг

Код Прюфера: $p(T) = (1, 5, 2, 6, 6, 2, 1, 3)$.

Массив вершин дерева: $V = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$.

Минимальная вершина, не содержащаяся в коде Прюфера – это 5.

Список ребер: $\{1, 4\}, \{5, 7\}, \{2, 5\}$.

Четвертый шаг

Код Прюфера: $p(T) = (1, 5, 2, 6, 6, 2, 1, 3)$.

Массив вершин дерева: $V = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$.

Минимальная вершина, не содержащаяся в коде Прюфера – это 8.

Список ребер: $\{1, 4\}, \{5, 7\}, \{2, 5\}, \{6, 8\}$.

Пятый шаг

Код Прюфера: $p(T) = (1, 5, 2, 6, 6, 2, 1, 3)$.

Массив вершин дерева: $V = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$.

Минимальная вершина, не содержащаяся в коде Прюфера – это 9.

Список ребер: $\{1, 4\}, \{5, 7\}, \{2, 5\}, \{6, 8\}, \{6, 9\}$.

Шестой шаг

Код Прюфера: $p(T) = (1, 5, 2, 6, 6, 2, 1, 3)$.

Массив вершин дерева: $V = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)$.

Минимальная вершина, не содержащаяся в коде Прюфера – это 6.

Список ребер: $\{1, 4\}, \{5, 7\}, \{2, 5\}, \{6, 8\}, \{6, 9\}, \{2, 6\}$.

Седьмой шаг

Код Прюфера: $p(T) = (1, 5, 2, 6, 6, 2, 1, 3)$.

Массив вершин дерева: V = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10).

Минимальная вершина, не содержащаяся в коде Прюфера – это 2.

Список ребер: {1, 4}, {5, 7}, {2, 5}, {6. 8}, {6, 9}, {2, 6}, {1, 2}.

Восьмой шаг

Код Прюфера: $p(T)$ = (1, 5, 2, 6, 6, 2, 1, 3).

Массив вершин дерева: V = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10).

Минимальная вершина, не содержащаяся в коде Прюфера – это 1.

Список ребер: {1, 4}, {5, 7}, {2, 5}, {6, 8}, {6, 9}, {2, 6}, {1, 2}, {3, 1}.

Завершение алгоритма

Код Прюфера: $p(T)$ = (1, 5, 2, 6, 6, 2, 1, 3).

Массив вершин дерева: V = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10).

Минимальная вершина, не содержащаяся в коде Прюфера – это 1.

Список ребер: {1, 4}, {5, 7}, {2, 5}, {6, 8}, {6, 9}, {2, 6}, {1, 2}, {3, 1}, {3, 10}.

2. Бинарный код

Любое дерево, в котором выделена одна вершина, называется ***корневым*** деревом. При этом выделенная вершина называется корнем.

Для корневого дерева используется следующий специальный способ представления (изображения) дерева:

1) Все вершины располагают по ярусам.

2) На нулевом ярусе располагается корень дерева.

3) На 1-м ярусе располагают все вершины дерева, смежные с корнем.

4) На 2 ярусе – все вершины, смежные с вершинами 1-го яруса.

5) На 3-ем – вершины, смежные с вершинами 2-го яруса и так далее.

Каждому корневому дереву ставится в соответствие бинарный код, который строится в процессе полного обхода дерева. Обход начинается с корня и заканчивается корнем. Обход осуществляется слева направо, т. е. сначала проходится левая ветвь, затем следующая и так далее, в конце – самая правая. При обходе необходимо подниматься по ветви до тех пор, пока это возможно. Затем по ветви опускаются до тех пор, пока не появится возможность продолжить подъем по еще не пройденной ветви. При подъеме с одного яруса на следующий в код дерева записывается 1, при опускании с яруса на ярус записывается 0.

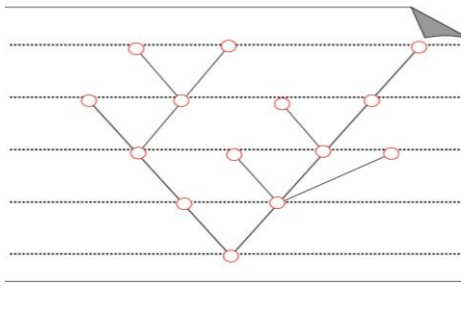


Рисунок 6.1

Так дерево на рис. 6.1 имеет код

$$B(T) = (11101101000011011011000100).$$

3. Код Гапта

Для деревьев применяется код Гапта.

Код Гапта дерева *T* – последовательность чисел $a_1, a_2, \ldots, a_n$, которую можно получить при обходе вершин дерева T по уровням, путем вписывания количества сыновей вершин, составляющих каждый уровень, в порядке их расположения слева направо.

Пример 6.2. Найти код Гапта дерева (рис. 6.2).

Пусть код состоит из числа сыновей каждой вершины дерева при обходе дерева слева направо, сверху вниз. Висячие вершины (их степень равна 1) не имеют сыновей, поэтому в код дерева порядка n, имеющего n_0 висячих вершин, войдет $n - n_0$ чисел. Для дерева на рис. 6.2 кодировка должна содержать 13−7 = 6 чисел и имеет следующий вид

$$[2, 3, 2, 2, 1, 2].$$

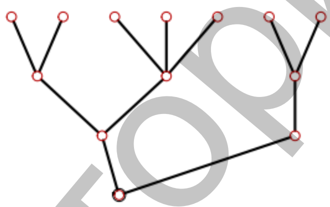


Рисунок 6.2

Пример 6.3. По заданному коду Гапта [1, 2, 4, 1, 1, 1, 3, 3, 2], построить дерево.

Построение начинается с корня. От корня, согласно последнему числу кода, идет два ребра к двум вершинам-сыновьям 2-го яруса. Рассматривая следующие с конца два числа кода, выясняем, что от этих вершин идет 3 и 3 ребра соответственно. Изображаем эту часть дерева и продолжаем строить следующий ярус. В результате получаем рис. 6.3.

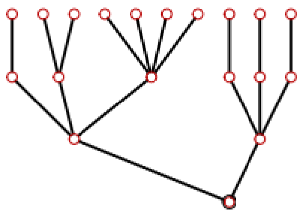
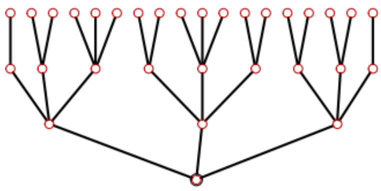


Рисунок 6.3

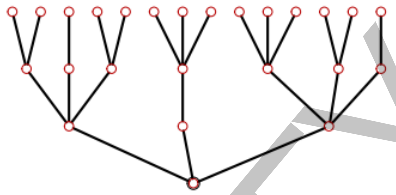
Задачи для самостоятельного решения

1. Записать бинарный код и код Гапта.

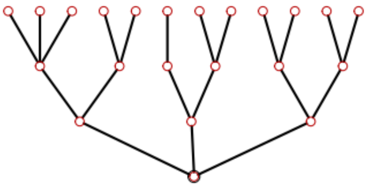
а)



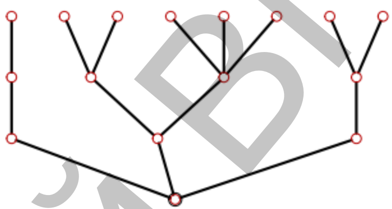
б)



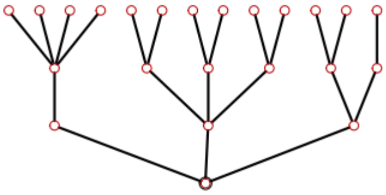
в)



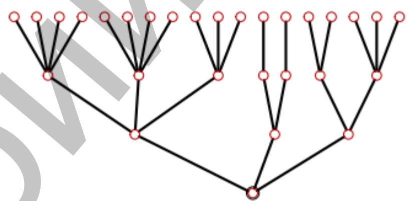
г)



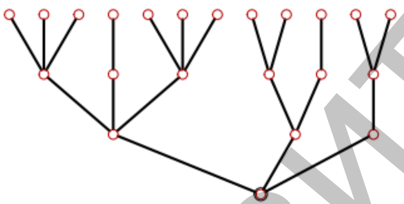
д)



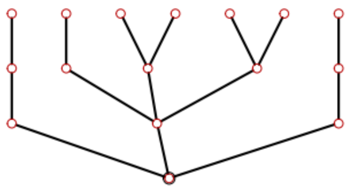
е)



ж)



з)



2. Распаковать год Гапта.

а) (3, 1, 1, 3, 2, 2, 4, 3, 1, 3, 3).

б) (2, 2, 2, 3, 2, 4, 4, 2, 3, 3, 2, 3).

в) (1, 3, 2, 3, 3, 1, 2, 3, 3, 1, 3).

г) (1, 1, 1, 1, 1, 3, 3, 1, 2, 3).

д) (2, 3, 2, 2, 3, 3, 4, 2, 2, 3, 3).

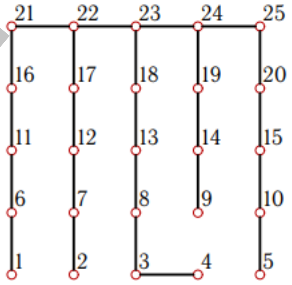
е) (3, 3, 2,1, 1, 2, 2, 1, 3, 3, 2, 3).

ж) (3, 3, 2, 2, 2, 2, 2, 2, 2, 3).

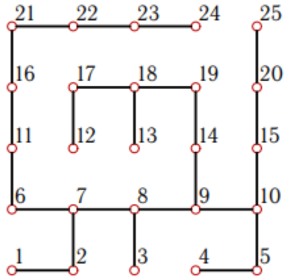
з) (1, 1, 2, 2, 3, 1, 2, 2, 3).

3. Записать код Прюфера.

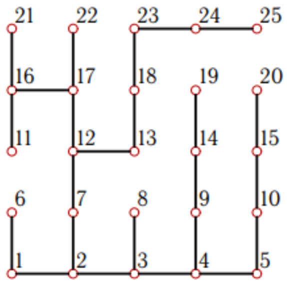
а)

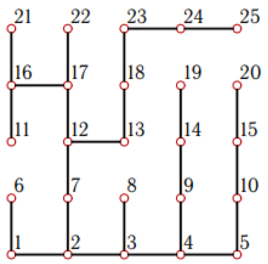


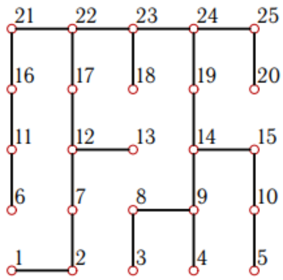
б)

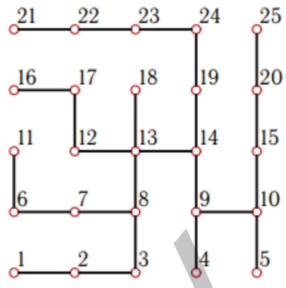


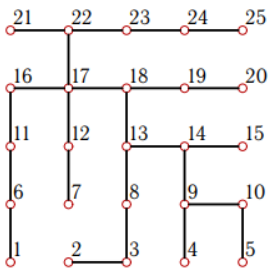
в)

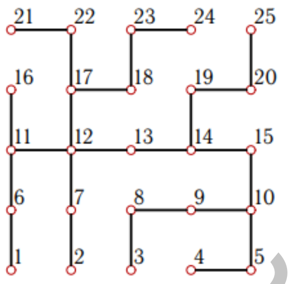


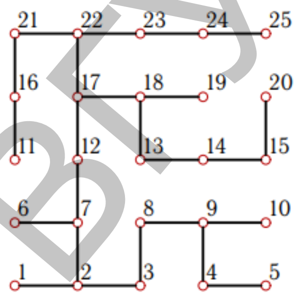
г)


д)




ж)


з)




4. Распаковать код Прюфера.

а) $p(T)$ = (6,3,4,5,10,7,8,9,10,15,12,13,14,15,20,16,17,18,19,20,25,23,24).
б) $p(T)$ = (2,3,4,3,8,6,11,9,8,13,16,17,18,19,20,21,22,17,18,19,20,25,24).
в)$p(T)$ = (2,7,4,4,9,11,8,9,9,14,12,17,14,21,18,22,23,18,19,14,15,20,25).
г) $p(T)$ = (5,4,3,2,6,9,8,7,6,11,21,16,18,19,18,17,16,11,12,13,14,15,20).
д) $p(T)$ = (2,3,4,9,10,13,18,13,8,16,11,6,7,17,12,7,8,9,10,15,20,25,24).
е)$p(T)$= (6,2,7,9,10,11,7,12,14,15,12,17,17,22,18,15,14,13,18,23,22,23,24).
ж) $p(T)$ = (2,3,8,11,12,5,4,9,8,13,16,17,13,18,20,21,22,23,24,25,22,23,24).
з) $p(T)$ = (6,7,8,4,9,9,11,14,15,14,16,17,18,17,16,11,6,7,8,9,14,19,24).
и)$p(T)$ = (6,7,3,8,10,11,12,8,13,15,16,13,18,19,20,21,18,19,22,23,24,19,20)

§ 7 Эйлеровы и гамильтоновы циклы

1. Эйлеровы циклы

Эйлеров цикл − цикл, содержащий все ребра графа по одному разу.

***Утверждение* 7.1.** Для того чтобы связный псевдограф G обладал эйлеровым циклом, необходимо и достаточно, чтобы степени всех его вершин были четными.

***Утверждение* 7.2.** Для того чтобы связный псевдограф G обладал эйлеровой цепью, необходимо и достаточно, чтобы он имел ровно 2 вершины нечетной степени.

Алгоритм выделения эйлерова цикла:

1) выбираем произвольно некоторую вершину v;

2) выбираем произвольно ребро *u*, инцидентное вершине *v*, присваиваем ему номер 1;

3) каждое пройденное ребро вычеркиваем и присваиваем ему номер на единицу больший предыдущего вычеркнутого ребра;

4) ребро, ведущее в начальную вершину *v*, выбираем только в том случае, если нет другого выбора;

5) ребро, являющееся мостом, выбираем только в том случае, если нет другого выбора;

6) после того, как будут пройдены все ребра, в графе образуется эйлеров цикл, причем нумерация ребер соответствует порядку их обхода.

***Пример* 7.1.** Найти эйлеров цикл в графе (рис. 7.1).

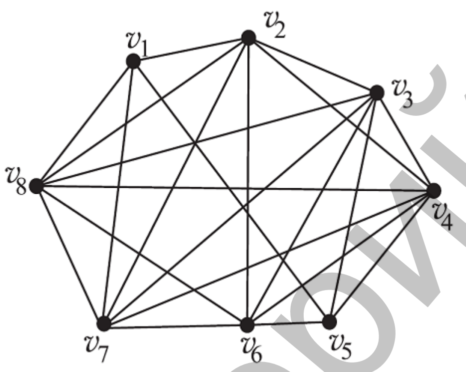


Рисунок 7.1

Решение. Начинаем обход из вершины v_1, пройденные ребра удаляем (рис. 7.2 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. 7.2 б)): (v_2, v_4, v_6, v_2, v_7, v_3, v_6, v_8, v_2).

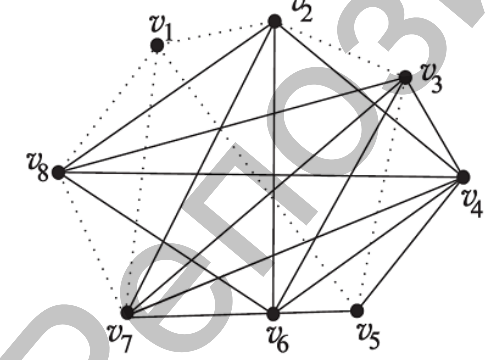


Рисунок 7.2 а).

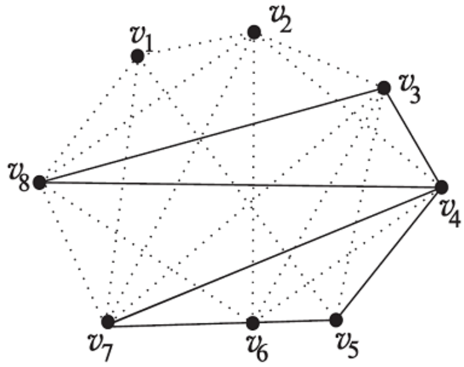


Рисунок 7.2 б).

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

Следовательно, (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 ($v_1, v_2, v_4, v_6, v_2, v_7, \mathbf{v_3}, v_6, v_8, v_2, v_3, v_5, v_1, v_7, v_8, v_1$) добавим следующий цикл ($v_3, v_4, v_5, v_6, v_7, v_4, v_8, v_3$). Имеем

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

2. Гамильтоновы циклы

***Гамильтонова цепь* (*цикл*)** − простая цепь (цикл), проходящая через все вершины графа.

Слово «гамильтонов» связано с именем известного ирландского математика У. Гамильтона[2], который в 1859 г. предложил игру «Кругосветное путешествие»[3].

Сходство и различия гамильтоновых и эйлеровых графов

	Эйлеровы циклы	**Гамильтоновы циклы**
Ребра	Эйлеров цикл проходит по каждому ребру ровно один раз.	Гамильтонов цикл может не проходить по некоторым ребрам.
Вершины	Эйлеров цикл может проходить через одну вершину несколько раз.	Гамильтонов цикл проходит ровно один раз по каждой вершине.

Алгоритм нахождения гамильтонова цикла в графе.

Дан связный граф $G=(V,E)$. Определена начальная вершина V_s.

Составляем матрицы P', $P_{V_s}^{(l)}$, $P^{(0)}$, $P_{V_s}^{(0)}$.

Матрица $P'=\left(p'_{i,j}\right)_{i,j\in\overline{1,|V|}}$ определяется следующим образом

$$p'_{i,j}=\begin{cases} V_j, \text{ если } \left(V_i,V_j\right)\in E,\\ 0, \text{ если } \left(V_i,V_j\right)\notin E.\end{cases}$$

---

[2] ГАМИЛЬТОН Уильям Роуан (Hamilton William Rowan) (04.08.1805-02.09.1865) – ирландский математик, член Ирландской Академии Наук. Родился в Дублине. В 3 года Гамильтон умел читать, неплохо знал арифметику и географию, в 10 лет стал студентом, к 12 годам изучил 12 языков. Достав латинский перевод «Начал» Евклида, он изучил это сочинение с 13 до 17 лет, изучал сочинения И. Ньютона и П. Лапласа. Окончил Тринити-колледж Дублинского университета, в 22 года стал профессором астрономии в Дублинском университете и директором университетской астрономической обсерватории.

[3] «Путешественники по додекаэдру» или «Кругосветное путешествие». В игре требуется, двигаясь по ребрам додекаэдра, пройти через 20 вершин его и, пройдя каждую лишь один раз, вернутся в конце к исходному пункту.

$P^{(0)}=\left(p_{i,j}^{0}\right)_{i,j\in\overline{1,|V|}}$ получаем из матрицы P' :

$p_{i,j}^{(0)},=\begin{cases}0, \text{ если } p'_{i,j}=0,\\ 1, \text{ если } p'_{i,j}\neq 0.\end{cases}$

$P_{V_s}^{(l)}$ – столбец s матрицы $P^{(l)}$.

$P_{V_s}^{(l)}=P'\times P_{V_s}^{(l\text{-}1)}$, где $l=1,2,...,(|V|-1)$.

$P_{V_s}^{(0)}$ – столбец номер s матрицы $P^{(0)}$.

<u>Внимание в столбце $P_{V_s}^{(i)}$ обнуляются</u>:

1) «произведение вершин» в строке, которое содержит вершину, равную метке строки $(l=\overline{1;|V|-1})$;

2) все элементы в V_s-ой строке $(l=\overline{1;|V|-2})$;

3) произведение, содержащее одинаковые множители $(l=\overline{1;|V|-1})$.

При $l=|V|-1$ в V_s-ой строке получим количество гамильтоновых циклов и последовательности вершин в цикле.

<u>Конец алгоритма</u>. Перечисляем количество циклов в графе G и последовательности вершин в каждом цикле.

Пример 7.2. Для графа на рис. 7.3 найти количество гамильтоновых циклов из вершины V_1 и указать их.

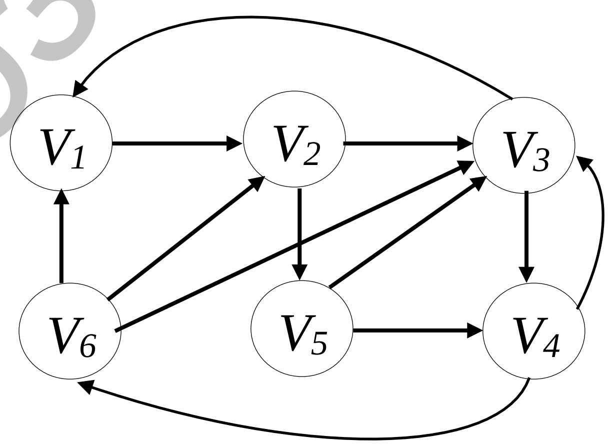


Рисунок 7.3

$$P'=\begin{pmatrix} 0 & V_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & V_3 & 0 & V_5 & 0 \\ V_1 & 0 & 0 & V_4 & 0 & 0 \\ 0 & 0 & V_3 & 0 & 0 & V_6 \\ 0 & 0 & V_3 & V_4 & 0 & 0 \\ V_1 & V_2 & V_3 & 0 & 0 & 0 \end{pmatrix}, \qquad P_{V_1}^{(0)} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

В графе шесть вершин, следовательно, будет пять итераций.

1) $$P_{V_1}^{(1)} = P' \times P_{V_1}^{(0)} = \begin{pmatrix} 0 \\ V_3 \\ 0 \\ V_3 + V_6 \\ V_3 \\ V_3 \end{pmatrix}.$$

2) $$P_{V_1}^{(2)} = P' \times P_{V_1}^{(1)} = \begin{pmatrix} \underline{V_2V_3} \\ V_5V_3 \\ \underline{V_4V_3} + V_4V_6 \\ V_6V_3 \\ V_4V_3 + V_4V_6 \\ V_2V_3 \end{pmatrix} \Rightarrow P_{V_1}^{(2)} = \begin{pmatrix} 0 \\ V_5V_3 \\ V_4V_6 \\ V_6V_3 \\ V_4V_3 + V_4V_6 \\ V_2V_3 \end{pmatrix}.$$

3) $$P_{V_1}^{(3)} = P' \times P_{V_1}^{(2)} = \begin{pmatrix} \underline{V_2V_5V_3} \\ V_3V_4V_6 + V_5V_4V_3 + V_5V_4V_6 \\ \underline{V_4V_6V_3} \\ \underline{V_3V_4V_6} + V_6V_2V_3 \\ V_3V_4V_6 + V_4V_6V_3 \\ V_2V_5V_3 + \underline{V_3V_4V_6} \end{pmatrix} \Rightarrow P_{V_1}^{(3)} = \begin{pmatrix} 0 \\ V_3V_4V_6 + V_5V_4V_3 + V_5V_4V_6 \\ 0 \\ V_6V_2V_3 \\ V_3V_4V_6 + V_4V_6V_3 \\ V_2V_5V_3 \end{pmatrix}.$$

4) $$P_{V_1}^{(4)} = P' \times P_{V_1}^{(3)} = \begin{pmatrix} \underline{V_2V_3V_4V_6 + V_2V_5V_4V_3 + V_2V_5V_4V_6} \\ V_5V_3V_4V_6 + V_5V_4V_6V_3 \\ \underline{V_4V_6V_2V_3} \\ V_6V_2V_5V_3 \\ V_4V_6V_2V_3 \\ \underline{V_2V_3V_4V_6} + V_2V_5V_4V_3 + \underline{V_2V_5V_4V_6} \end{pmatrix} \Rightarrow P_{V_1}^{(4)} = \begin{pmatrix} 0 \\ V_5V_3V_4V_6 + V_5V_4V_6V_3 \\ 0 \\ V_6V_2V_5V_3 \\ V_4V_6V_2V_3 \\ V_2V_5V_4V_3 \end{pmatrix}.$$

$$5)\, P_{V_1}^{(5)} = P' \times P_{V_1}^{(4)} = \begin{pmatrix} V_2V_5V_3V_4V_6 + V_2V_5V_4V_6V_3 \\ \underline{V_5V_4V_6V_2V_3} \\ \underline{V_4V_6V_2V_5V_3} \\ \underline{V_6V_2V_5V_4V_3} \\ \underline{V_4V_6V_2V_5V_3} \\ \underline{V_2V_5V_3V_4V_6} + \underline{V_2V_5V_4V_6V_3} \end{pmatrix} \Rightarrow P_{V_1}^{(5)} = \begin{pmatrix} V_2V_5V_3V_4V_6 + V_2V_5V_4V_6V_3 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Получаем два цикла: 1) $V_1V_2V_5V_3V_4V_6V_1$ и 2) $V_1V_2V_5V_4V_6V_3V_1$.

Пример 7.3. Для графа на рис. 7.4 найти количество гамильтоновых циклов из вершины V_1 и указать их.

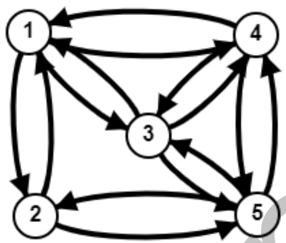


Рисунок 7.4

$$P' = \begin{pmatrix} 0 & V_2 & V_3 & V_4 & 0 \\ V_1 & 0 & 0 & 0 & V_5 \\ V_1 & 0 & 0 & V_4 & V_5 \\ V_1 & 0 & V_3 & 0 & V_5 \\ 0 & V_2 & V_3 & V_4 & 0 \end{pmatrix}, \qquad P_{V_1}^{(0)} = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}.$$

В графе пять вершин, следовательно, будет четыре итерации.

$$1) \qquad P_{V_1}^{(1)} = P' \times P_{V_1}^{(0)} = \begin{pmatrix} \underline{V_2 + V_3 + V_4} \\ 0 \\ V_4 \\ V_3 \\ V_2 + V_3 + V_4 \end{pmatrix} \Rightarrow P_{V_1}^{(1)} = \begin{pmatrix} 0 \\ 0 \\ V_4 \\ V_3 \\ V_2 + V_3 + V_4 \end{pmatrix}.$$

2) $$P_{V_1}^{(2)} = P' \times P_{V_1}^{(1)} = \begin{pmatrix} \underline{V_3V_4 + V_4V_3} \\ \underline{V_5V_2} + V_5V_3 + V_5V_4 \\ \underline{V_4V_3} + V_5V_2 + \underline{V_5V_3} + V_5V_4 \\ \underline{V_3V_4} + V_5V_2 + V_5V_3 + \underline{V_5V_4} \\ V_3V_4 + V_4V_3 \end{pmatrix} \Rightarrow P_{V_1}^{(2)} = \begin{pmatrix} 0 \\ V_5V_3 + V_5V_4 \\ V_5V_2 + V_5V_4 \\ V_5V_2 + V_5V_3 \\ V_3V_4 + V_4V_3 \end{pmatrix}.$$

3) $$P_{V_1}^{(3)} = P' \times P_{V_1}^{(2)} = \begin{pmatrix} \underline{V_2V_5V_3 + V_2V_5V_4 + V_3V_5V_2 + V_3V_5V_4 + V_4V_5V_2 + V_4V_5V_3} \\ V_5V_3V_4 + V_5V_4V_3 \\ V_4V_5V_2 + \underline{V_4V_5V_3} + \underline{V_5V_3V_4} + \underline{V_5V_4V_3} \\ V_3V_5V_2 + \underline{V_3V_5V_4} + \underline{V_5V_3V_4} + \underline{V_5V_4V_3} \\ \underline{V_2V_5V_3 + V_2V_5V_4 + V_3V_5V_2 + V_3V_5V_4 + V_4V_5V_2 + V_4V_5V_3} \end{pmatrix} \Rightarrow$$

$$\Rightarrow P_{V_1}^{(3)} = \begin{pmatrix} 0 \\ V_5V_3V_4 + V_5V_4V_3 \\ V_4V_5V_2 \\ V_3V_5V_2 \\ 0 \end{pmatrix}.$$

4) $$P_{V_1}^{(4)} = P' \times P_{V_1}^{(3)} = \begin{pmatrix} V_2V_5V_3V_4 + V_2V_5V_4V_3 + V_3V_4V_5V_2 + V_4V_3V_5V_2 \\ 0 \\ \underline{V_4V_3V_5V_2} \\ \underline{V_3V_4V_5V_2} \\ \underline{V_2V_5V_3V_4 + V_2V_5V_4V_3 + V_3V_4V_5V_2 + V_4V_3V_5V_2} \end{pmatrix} \Rightarrow$$

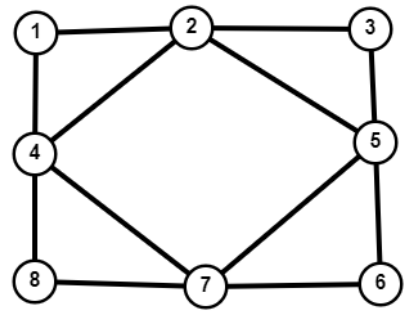
$$P_{V_1}^{(4)} = \begin{pmatrix} V_2V_5V_3V_4 + V_2V_5V_4V_3 + V_3V_4V_5V_2 + V_4V_3V_5V_2 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

Получаем четыре цикла: 1) $V_1V_2V_5V_3V_4V_1$; 2) $V_1V_2V_5V_4V_3V_1$; 3) $V_1V_3V_4V_5V_2V_1$; 4) $V_1V_4V_3V_5V_2V_1$.

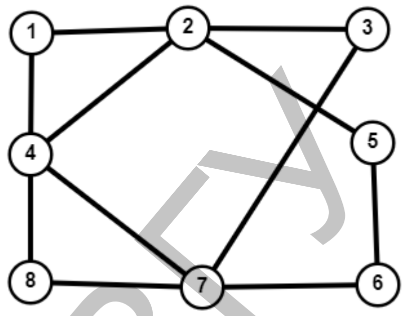
Задачи для самостоятельного решения

1. Найти эйлеров цикл в графе.

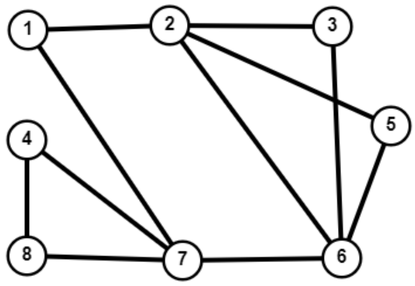
а)



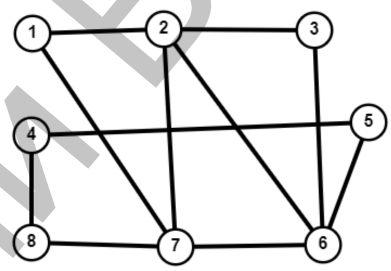
б)



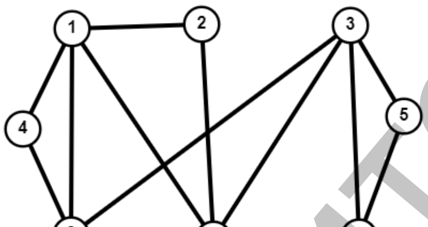
в)



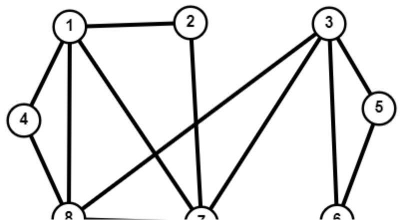
г)



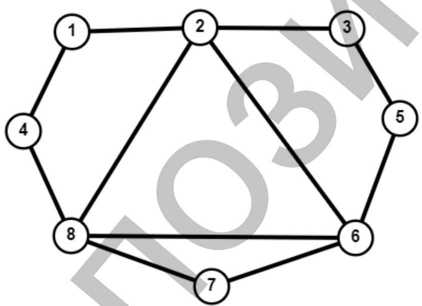
д)



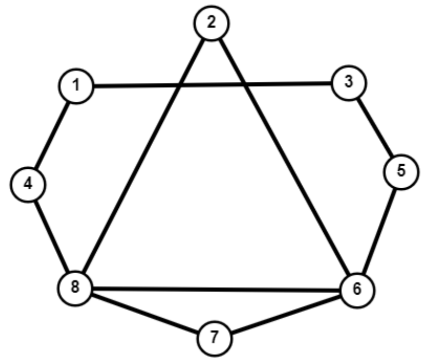
е)



ж)

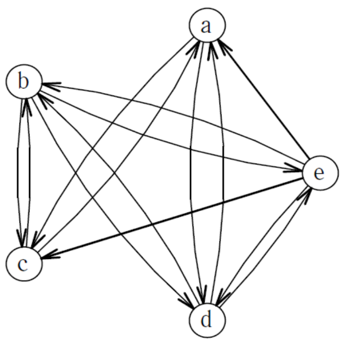


з)

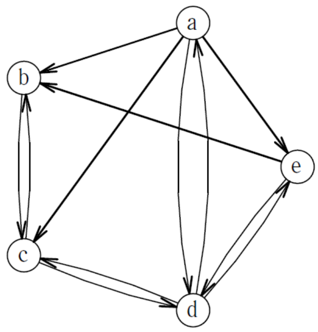


2. Найти все гамильтоновы циклы графа из вершин ____

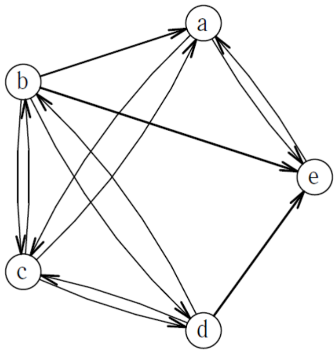
Вариант 1



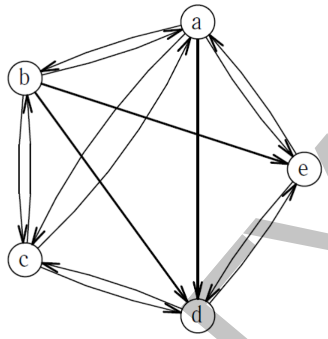
Вариант 2



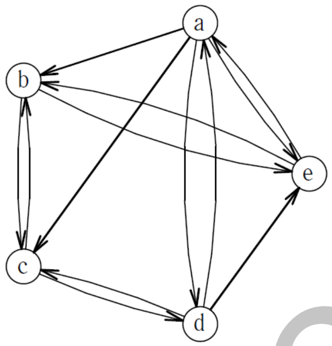
Вариант 3



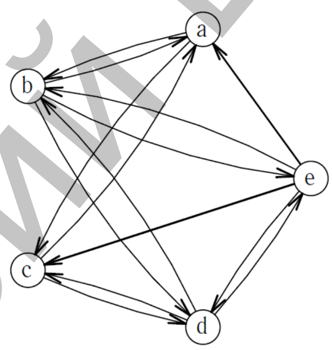
Вариант 4



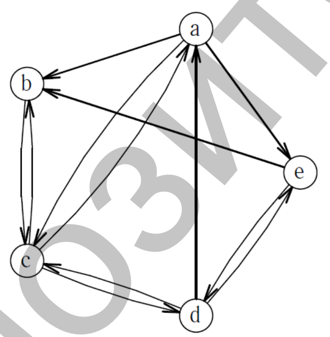
Вариант 5



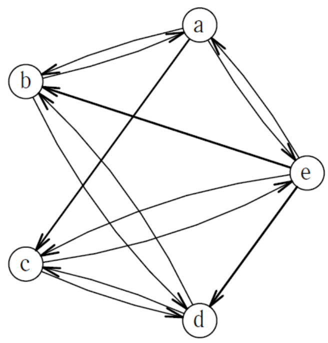
Вариант 6



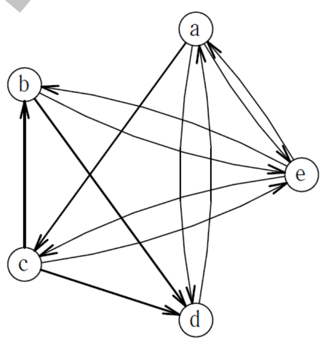
Вариант 7



Вариант 8



Вариант 9



§ 8 Раскраска графа

Пусть $G = (V, E)$ – неориентированный граф без петель. ***Раскраска*** графа – функция, которая каждой вершине ставит в соответствие некоторое натуральное число (цвет). Раскраска называется ***правильной***, если смежные вершины имеют разные цвета.

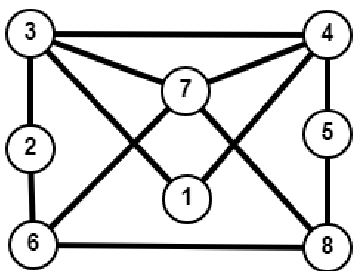
Хорошо известна проблема четырех красок – математическая задача, предложенная Ф. Гутри (англ.) в 1852 году, сформулированная следующим образом: «Выяснить, можно ли всякую расположенную на сфере карту раскрасить четырьмя красками так, чтобы любые две области, имеющие общий участок границы, были раскрашены в разные цвета».

Жадный алгоритм раскраски графов описан в [21].

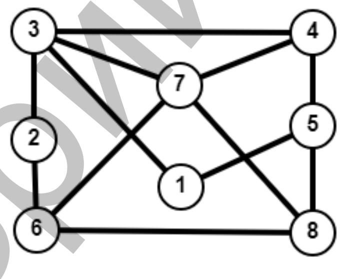
Задания для самостоятельного решения

1. Задать правильную раскраску вершин графа, используя жадный алгоритм раскраски.

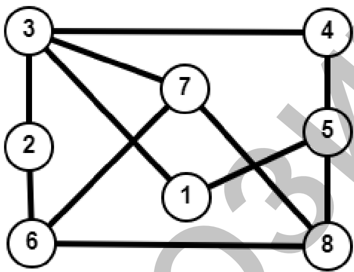
а)



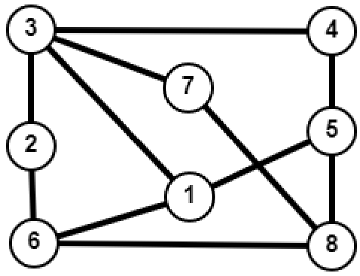
б)



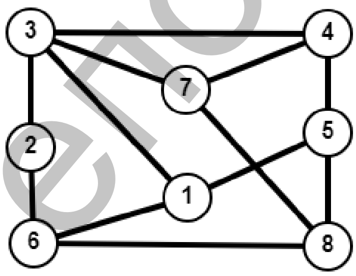
в)



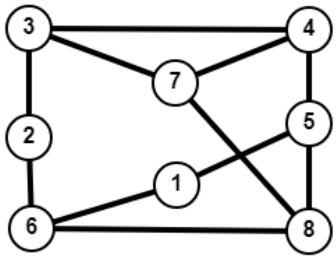
г)



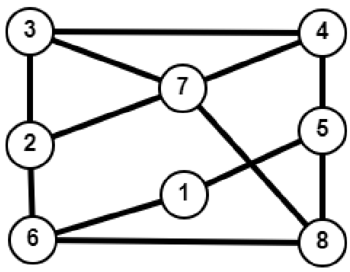
д)



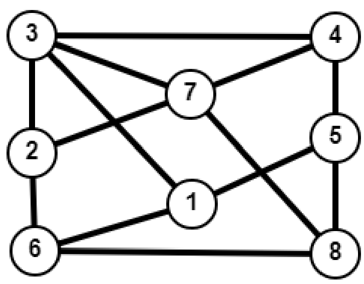
е)



ж)



з)



§ 9 Остов наименьшего веса

Остовным деревом связного неориентированного графа G называется любой его подграф, содержащий все вершины графа и являющийся деревом.

Минимальным остовным деревом называется остов минимального веса.

Задача, в которой требуется найти в связном взвешенном графе минимальный остов, называется ***задачей о кратчайшем остове.***

Для того чтобы отыскать в связном взвешенном графе остов минимального веса используют: алгоритм Краскала и алгоритм Прима.

Пример 9.1. Дан взвешенный граф (рис. 9.1).

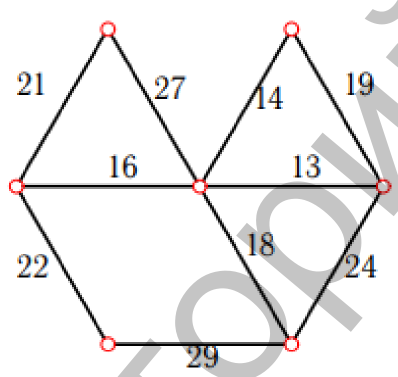


Рисунок 9.1

Алгоритм Краскала[4]

Строим граф, присоединяя к пустому графу на множестве вершин заданного графа ребро наименьшего веса. К полученному графу последовательно присоединяем остальные ребра, выбирая на каждом шаге ребро наименьшего веса, не образующее цикл с имеющимися ребрами. В нашем случае начинаем с ребра весом 13 — наименьшего в графе. На рисунке 9.2 дана последовательность действий. Ребро весом 19 не включается в остов, так как оно образует цикл с ребрами весом 14 и 13.

---

[4] **Джозеф Бернард Краскал** (29 января 1928 – 19 сентября 2010) американский математик, статистик, программист и психометрик. Краскал родился в Нью-Йорке. Он был студентом Чикагского университета, получил степень бакалавра наук по математике в 1948 году и степень магистра математики в 1949 году. После работы в Чикагском университете Краскал посещал Принстонский университет, где он защитил кандидатскую диссертацию в 1954 году. В 1956 году им был создан *алгоритм Краскала*.

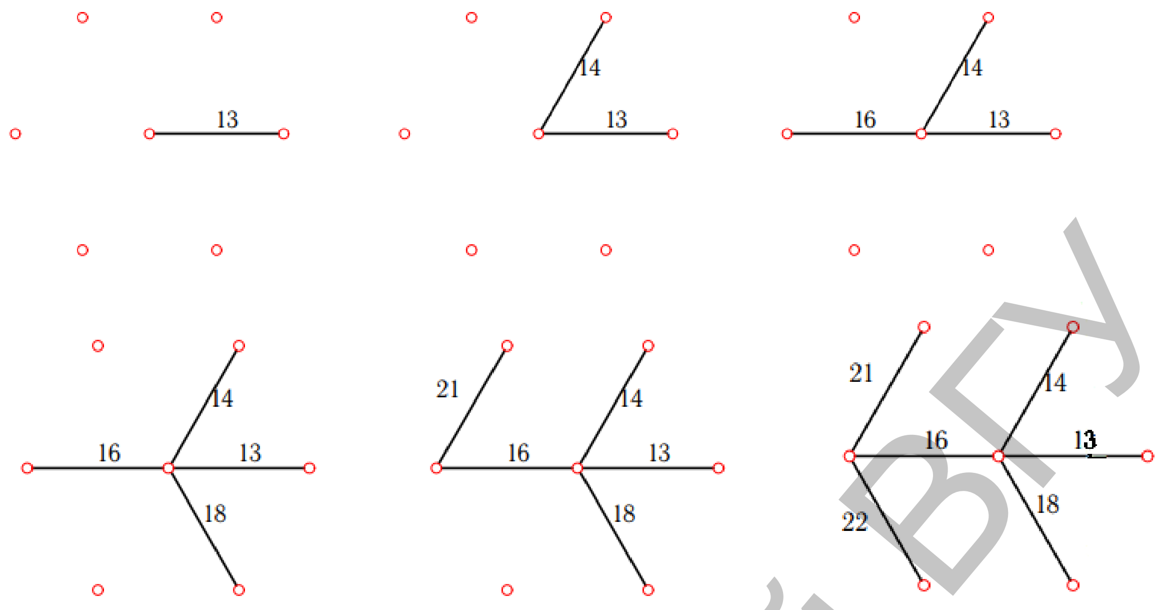


Рисунок 9.2

Алгоритм Прима[5]

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

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

3. Продолжаем далее, разыскивая каждый раз ребро наименьшего веса, инцидентное построенным вершинам, не включая в круг поиска все ребра, их соединяющие. В нашем примере начнем с вершины А. На рисунке 9.3. дана последовательность действий.

---

[5] **Роберт Клэй Прим** (родился 25 сентября 1921 года) – американский математик и специалист по компьютерам.

В 1941 году Прим получил степень бакалавра в области электротехники в Техасском университете в Остине. Позже в 1949 году он получил степень доктора философии по математике в Принстонском университете, где он также работал научным сотрудником с 1948 по 1949 год.

В Bell Laboratories Роберт Прим разработал алгоритм для поиска минимального остовного дерева во взвешенном графе, который является основным камнем преткновения в проектировании компьютерных сетей. Его самоназванный алгоритм, алгоритм Прима, был первоначально открыт в 1930 году математиком Войтехом Ярником, а позже независимо – Примом в 1957 году.

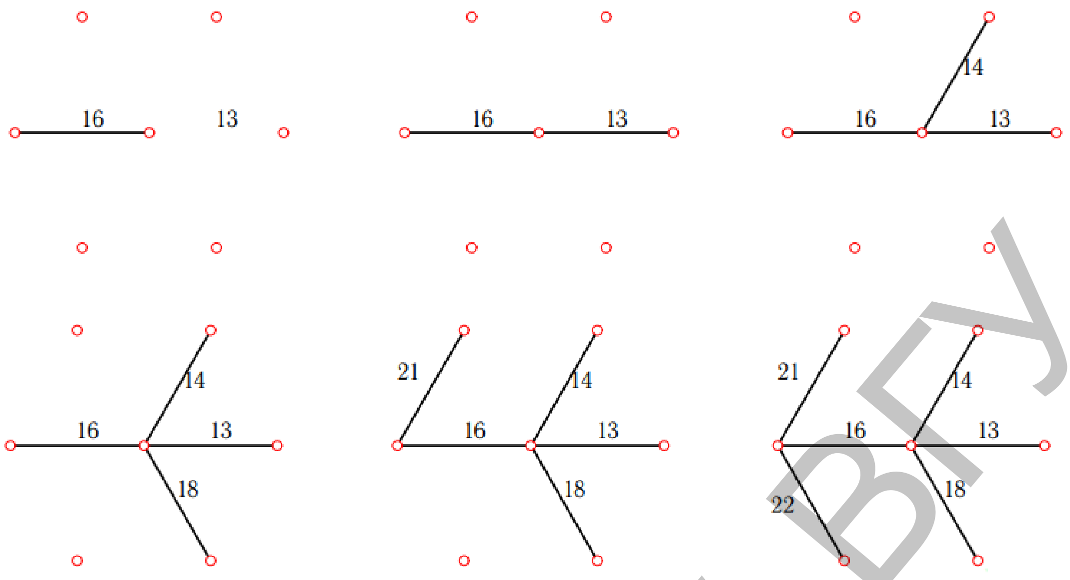
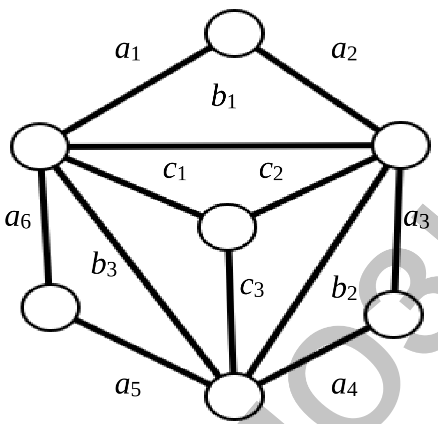


Рисунок 9.3

<u>Задачи для самостоятельного решения</u>

1. Найти остов наименьшего веса в графе (двумя способами).



1) a=(4, 3, 4, 5, 2, 1), b=(1, 2, 2), c=(5, 3, 4).
2) a=(3, 1, 2, 3, 4, 1), b=(2, 1, 2), c=(5, 5, 4).
3) a=(4, 3, 3, 2, 4, 3), b=(2, 2, 3), c=(2, 1, 1).
4) a=(3, 2, 3, 4, 3, 4), b=(2, 3, 1), c=(4, 4, 5).
5) a=(3, 4, 5, 2, 1, 4), b=(1, 2, 2), c=(4, 5, 3).
6) a=(1, 2, 3, 4, 1, 3), b=(2, 1, 2), c=(4, 5, 5).
7) a=(4, 5, 2, 1, 4, 3), b=(2, 2, 3), c=(1, 2, 1).
8) a=(2, 3, 4, 3, 4, 3), b=(2, 3, 1), c=(5, 4, 4).

§ 10 Задачи о кратчайшем пути

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

Алгоритм Дейкстры[6]**.**

Данный алгоритм позволяет найти кратчайшие пути от заданной вершины до остальных вершин графа. Недостаток алгоритма состоит в

[6] Алгоритм Дейкстры (англ. Dijkstra’s algorithm) – алгоритм на графах, изобретенный нидерландским ученым Эдсгером Дейкстрой в 1959 году.

том, что он будет работать некорректно, если граф имеет дуги отрицательного веса. Принцип работы алгоритма рассмотрим на следующем примере.

Пример 10.1. Найти кратчайшие пути в орграфе (рис. 10.1) от первой вершины ко всем остальным, используя алгоритм Дейкстры. Постройте дерево кратчайших путей.

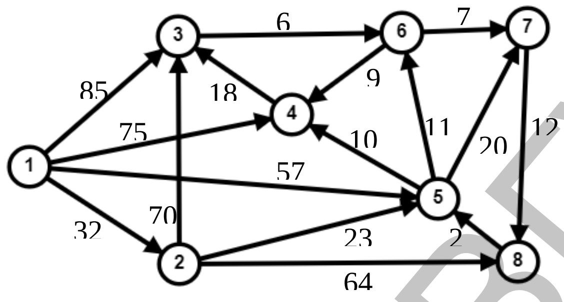


Рисунок 10.1

Решение. Найдем кратчайший путь от вершины x_1 до всех вершин, используя алгоритм Дейкстры. Он заключается в том, что вершинам графа присваиваются временные метки, которые затем по определенным правилам заменяются на постоянные метки. Будем использовать обозначения:

$L^*(x_i)$ – постоянная метка вершины x_i;

$L^н(x_i)$ – новая временная метка вершины x_i;

$L^c(x_i)$ – старая временная метка вершины x_i;

R_{ij} – вес ребра, соединяющего вершины x_i и x_j.

Новая временная метка вычисляется по формуле:

$L^н(x_i) = \min\{ L^c(x_i), R_{ij} + L^*(x_i)\}$.

После этого из всех временных меток выбирается наименьшая, и она становится постоянной меткой. Действия продолжаются, пока не будут найдены постоянные метки для всех вершин графа. Результаты действий на каждом шаге будем заносить в таблицу. В предпоследний столбец заносим вершину, получившую постоянную метку, в последний столбец – величину этой метки (для данного шага).

Шаг 1. Начальная вершина x_1, имеет постоянную метку $L^*(x_1) = 0$, остальные вершины имеют временную метку ∞.

Шаг 2. Определяем множество последователей вершиныΓ$(x_1) = \{x_3, x_4, x_5, x_2\}$. Пересчитываем их временные метки по основной формуле. $L^н(x_3) = 85$, $L^н(x_4) = 75$, $L^н(x_5) = 57$, $L^н(x_2) = 32$. Берем вершину x_2 с минимальной временной меткой 32, присваиваем этой вершине постоянную метку $L^*(x_2) = 32$.

Шаг 3. Определяем множество последователей вершины вершиныΓ$(x_2) = \{x_3, x_5, x_8\}$. Пересчитываем их временные метки по основной формуле. $L^н(x_3) = \min\{85{,}32 + 70\} = 85$, 5 $L^н(x_5) = \min\{57, 32 + + 23\} = 55$, $L^н(x_8) = 32 + 64 = 96$. Берем вершину x_5 с минимальной временной меткой 55, присваиваем этой вершине постоянную метку 5 $L^*(x_5) = 55$.

Шаг 4. Определяем множество последователей вершины вершины $\Gamma(x_5) = \{x_4, x_6, x_7\}$. Пересчитываем их временные метки по основной формуле. $L^н(x_4) = \min\{75, 55 + 10\} = 65$, $L^н(x_6) = 55 + 11 = 66$, $L^н(x_7) = 55 + 20 = 75$. Берем вершину x_4 с минимальной временной меткой 65, присваиваем этой вершине постоянную метку $L^*(x_4) = 65$.

Шаг 5. Определяем множество последователей вершины $\Gamma(x_4) = \{x_3\}$. Пересчитываем их временные метки по основной формуле. $L^н(x_3) = \min\{85, 65 + 18\} = 83$. Берем вершину x_6 с минимальной временной меткой 66, присваиваем этой вершине постоянную метку $L^*(x_6) = 66$.

Шаг 6. Определяем множество последователей вершины $\Gamma(x_6) = \{x_4, x_7\}$. Пересчитываем их временные метки по основной формуле. $L^н(x_7) = \min\{75, 66 + 7\} = 73$. Берем вершину x_7 с минимальной временной меткой 73, присваиваем этой вершине постоянную метку $L^*(x_7) = 73$.

Шаг 7. Определяем множество последователей вершины $\Gamma(x_7) = \{x_8\}$. Пересчитываем их временные метки по основной формуле. $L^н(x_8) = \min\{96, 73 + 12\} = 85$. Берем вершину x_3 с минимальной временной меткой 83, присваиваем этой вершине постоянную метку $L^*(x_3) = 83$.

Шаг 8. Определяем множество последователей вершины $\Gamma(x_3) = \{x_6\}$. Эта вершина уже имеет постоянную метку. Поэтому берем последнюю вершину x_8 с временной меткой 85, присваиваем этой вершине постоянную метку $L^*(x_8) = 85$.

Шаги	**Вершины**								x_i	$L^*(x_i)$
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8		
1	0	∞	∞	∞	∞	∞	∞	∞	x_1	0
2		**32**	85	75	57	∞	∞	∞	x_2	32
3			85	75	**55**	∞	∞	96	x_5	55
4			85	**65**		66	75	96	x_4	65
5			83			**66**	75	96	x_6	66
6			83				**73**	96	x_7	73
7			**83**					85	x_3	83
8								**85**	x_8	85

Кратчайшие пути найдены, их длина приведена в последних двух столбцах расчетной таблицы. Построим дерево кратчайших путей (рис. 10.2) (ребра дерева обведены жирным**)** – ребра (1,2), (2,5), (5,4), (4,3), (5,6), (6,7), (7,8).

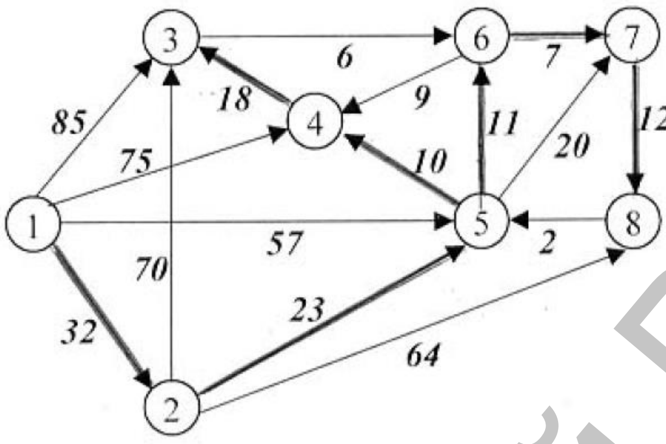


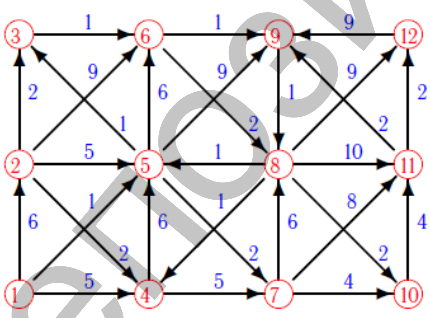
Рисунок 10.2

Для решения задачи о кратчайшем пути также используются алгоритмы Форда-Беллмана[7] [21], Флойда-Уоршелла[8] [9, 14], Флойда [7, 13].

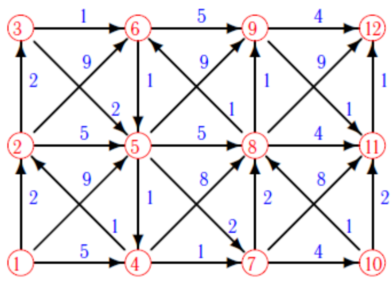
<u>Задачи для самостоятельного решения</u>

1. Для заданного орграфа найти кратчайший путь от вершины 1 до вершины 12.

а)



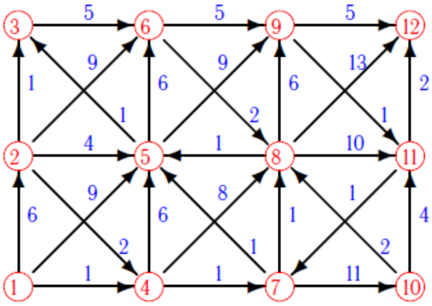
б)



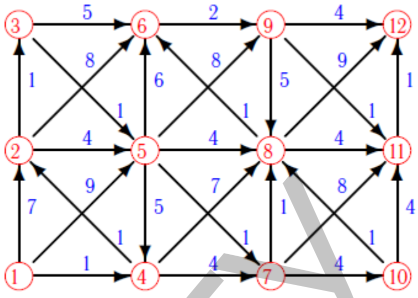
[7] Алгоритм Форда-Беллмана позволяет произвести поиск кратчайшего пути во взвешенном графе. Данный алгоритм находит кратчайшие пути от одной вершины графа до всех остальных. Алгоритм допускает рёбра с отрицательным весом. Предложен независимо Ричардом Беллманом и Лестером Фордом.

[8] Алгоритм Флойда-Уоршелла – алгоритм нахождения длин кратчайших путей между всеми парами вершин во взвешенном ориентированном графе. Работает корректно, если в графе нет циклов отрицательной величины, а в случае, когда такой цикл есть, позволяет найти, хотя бы один такой цикл.

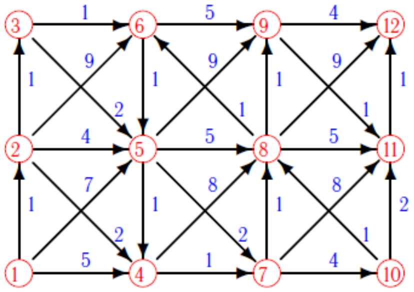
в)



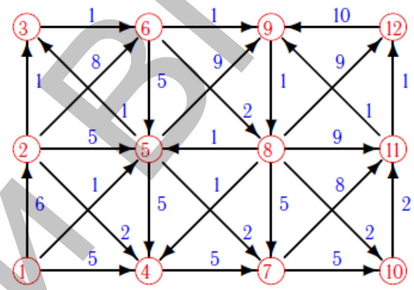
г)



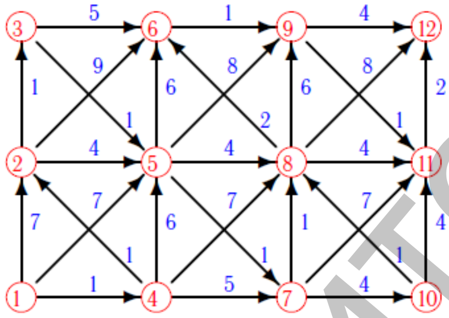
д)



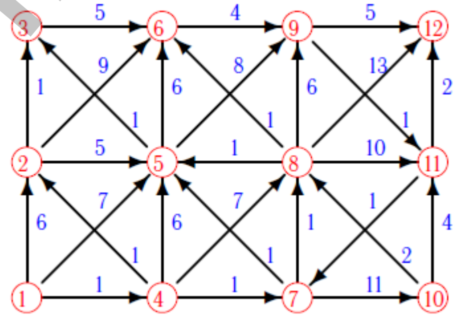
е)



ж)



з)



§ 11 Двудольные графы

Граф называется ***двудольным,*** если множество его вершин можно разбить на два непустых подмножества (доли) так, что никакие две вершины одной доли не являются смежными. Таким образом, в двудольном графе смежными могут быть только вершины из разных долей (не обязательно каждая с каждой).

Критерий Кёнига двудольности графа. *Для двудольности графа необходимо и достаточно, чтобы он не содержал циклов нечетной длины*.

Если любые две вершины двудольного графа, входящие в разные доли, смежны, то граф называется *полным двудольным*.

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

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

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

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

Паросочетание, порядок которого равен порядку графа, называется *совершенным*. Совершенное паросочетание включает в себя все вершины двудольного графа. Совершенное паросочетание является наибольшим и максимальным. Максимальное паросочетание может быть и не наибольшим.

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

Перманент матрицы (per) равен сумме всех произведений элементов матрицы по одному из каждой строки в разных столбцах. Перманент квадратной матрицы α_{ij} ($i, j = 1,\ldots, n$) определяется, подобно определителю, рекуррентным образом. При $n = 1$ имеем $\mathrm{per}(A_1) = \alpha_{11}$. Для матрицы A_{n+1} размером n+1 получим разложение по первой строке: $\mathrm{per}(A_{n+1}) = \sum_{k=1}^{n} \alpha_{1k}\mathrm{per}(A_k)$, где $\mathrm{per}(A_k)$ – перманент матрицы размером n, полученной из A_{n+1} вычеркиванием первой строки и k-ого столбца. Таким образом, перманент отличается от определителя только тем, что все его слагаемые берутся со знаком плюс. Например, $\mathrm{per}\begin{vmatrix} 1 & 2 \\ 3 & 4 \end{vmatrix}$.

Рассмотрим двудольный граф G рис. 11.1.

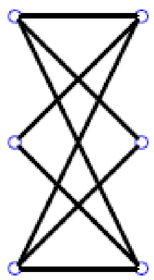


Рисунок 11.1

Матрица смежности данного графа

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

Перманент данной матрицы равен 4. На рисунке 11.2 изображены покрытия графа.

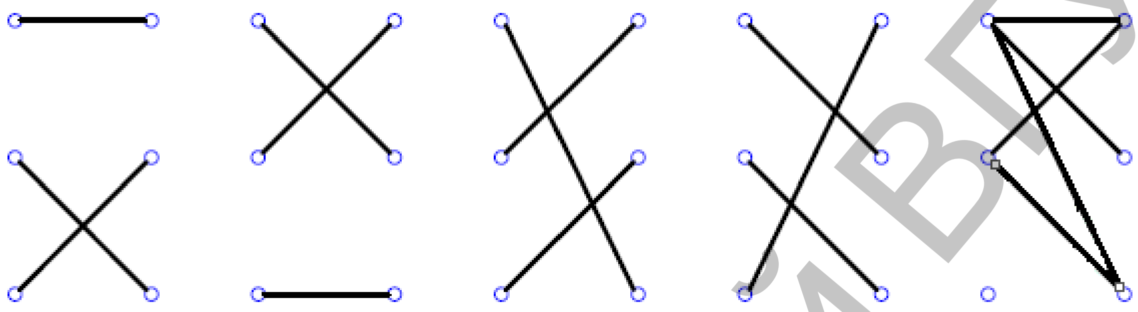


Рисунок 11.2

Если граф не имеет совершенного паросочетания, то перманент его матрицы равен 0. Например, матрица графа с одной изолированной вершиной на рис. 11.2 имеет вид:

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

Перманент этой матрицы равен 0.

Пример 11.1. В заданном двудольном графе (рис. 11.3) найти наибольшее паросочетание.

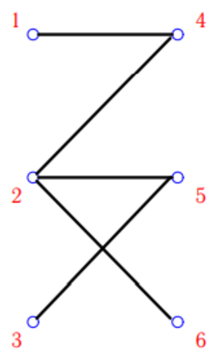


Рисунок 11.3

Решение. Применим алгоритм Форда–Фалкерсона [14]. Образуем из двудольного графа сеть, добавляя исток 7 и сток 8 и ориентируя ребра из истока в сток. Пропускную способность всех полученных дуг положим равной 1 (рис. 11.4). Максимальный поток по этой сети соответствует искомому паросочетанию графа. Заметим, что решение будет не единственным.

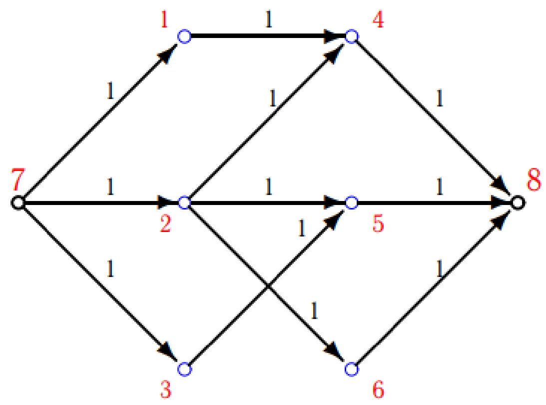


Рисунок 11.4

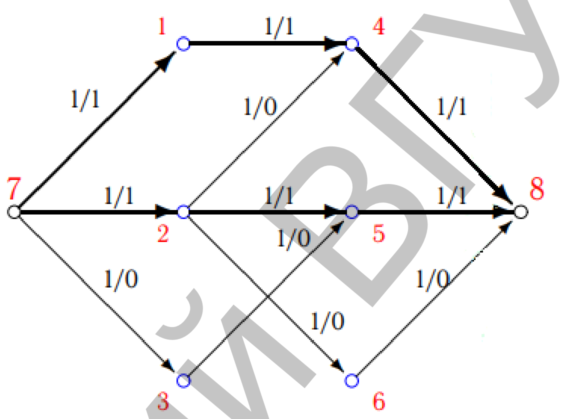


Рисунок 11.5

Находим все возможные маршруты из истока в сток и насыщаем их. Таких маршрутов два: 7–1–4–8 и 7–2–5–8 (рис. 11.5). Маршрутов из 7 в 8, по которым можно было бы пропустить дополнительный поток, больше нет. Перераспределяем полученный поток. Находим пути из истока в сток, содержащие ненасыщенные прямые дуги и непустые обратные. В данной задаче это путь 7–3–5–2–6–8. Уменьшаем поток в обратных дугах и увеличиваем на эту же величину поток в прямых дугах (рис. 11.6). В покрытие исходного двудольного графа войдут ребра с потоком, равным 1 (рис. 11.7).

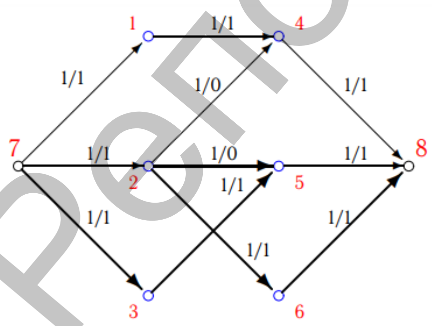


Рисунок 11.6

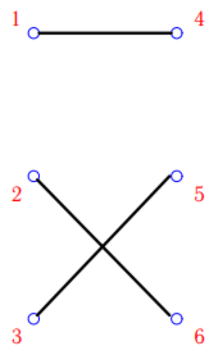


Рисунок 11.7

Матрица смежности исходного двудольного графа имеет вид

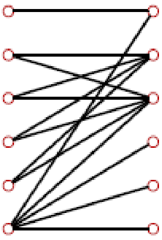
$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

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

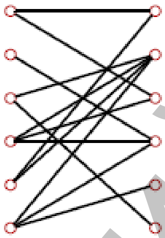
Задачи для самостоятельного решения

1. В заданном двудольном графе найти наибольшее паросочетание.

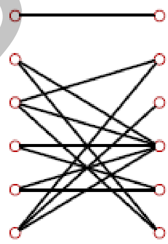
а



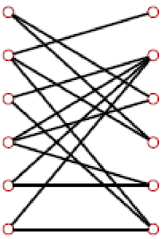
б



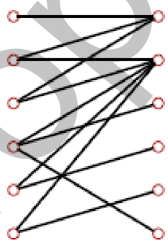
в



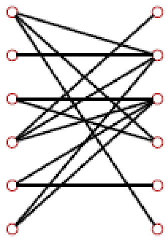
г



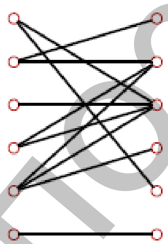
д



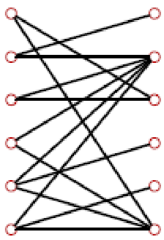
е



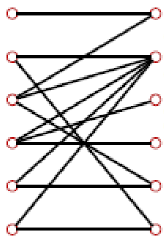
ж



з



и



Литература

1. Алексеев, В.Е. Графы и алгоритмы. Структуры данных. Модели вычислений / В.Е. Алексеев, В.А. Таланов. – М. : Интернет-университет информационных технологий; БИНОМ. Лаборатория знаний, 2006. – 320 с.
2. Березина, Л.Ю. Графы и их применение / Л.Ю. Березина. – М. : Просвещение, 1979. – 144 с.
3. Берж, К. Теория графов и ее применения / К. Берж. – М. : ИЛ, 1962. – 319 с.
4. Белов, В.В. Теория графов: Учебн. пособие / В.В. Белов, Е.М. Воробьев, В.Е. Шаталов. – М. : Высш. шк., 1976. – 392 с.
5. Гаврилов, Г.П. Задачи и упражнения по дискретной математике: Учебн. пособие / Г.П. Гаврилов, А.А. Сапожко. – М. : ФИЗМАТЛИТ, 2005. – 416 с.
6. Гайдамака, Ю.В. Лекции по дискретной математике. Часть II. Комбинаторика. Теория конечных графов. Учебное пособие / Ю.В. Гайдамака, Э.Р. Зарипова, М.Г. Кокотчикова, Л.А. Севастьянова. – М. : Изд-во РУДН, 2008. – 60 с.
7. Домнин, Л.Н. Элементы теории графов / Л.Н. Домнин. – Пенза, 2004. – 139 с.
8. Емеличев, В.А. Лекции по теории графов / В.А. Емеличев, О.И. Мельников, В.И.Сарванов, Р.И. Тышкевич. – М. : Наука, 1990. – 383 с.
9. Зарипова, Э.Р. Дискретная математика. Часть III. Теория графов. Учебной пособие / Э.Р. Зарипова, М.Г. Кокотчикова. – М. : Изд-во РУДН, 2008. – 179 с.
10. Зыков, А.А. Теория конечных графов / А.А. Зыков. – Новосибирск : Наука, 1969. – 554 с.
11. Зыков, А.А. Основы теории графов / А.А. Зыков. – М. : Вузовская книга, 2004. – 664 с.
12. Касьянов, В.Н. Графы в программировании: обработка, визуализация и применение / В.Н. Касьянов, В.А. Евстигнеев. – СПб. : БХВ-Петербург, 2003. 1104 с.
13. Кристофидес, Н. Теория графов. Алгоритмический подход / Н. Кристофидес. – М. : Мир, 1978. – 432 с.
14. Кормен, Т.Х. Алгоритмы: построение и анализ / Т.Х. Кормен, Ч.И. Лейзерсон, Р.Л. Ривест, К. Штайн. – М. : Вильямс, 2005. – 1296 с.
15. Мельников, О.И. Занимательные задачи по теории графов: Учебн.-метод. пособие / О.И. Мельников. – Минск : Тетра Системс, 2001. – 144 с.
16. Носов, В.И. Элементы теории графов. Учебное пособие / В.И. Носов, Т.В. Бернштейн, Н.В. Носкова, Т.В. Храмова. – Новосибирск, 2008. – 107 с.
17. Оре, О. Теория графов / О. Оре. – М. : Наука, 1980. – 336 с.
18. Уилсон, Р. Введение в теорию графов / Р. Уилсон. – М. : Мир, 1977. – 207 с.
19. Харари, Ф. Теория графов / Ф. Харари. – М. : Мир, 1973. – 300 с.
20. Харари, Ф. Перечисление графов / Ф. Харари, Э. Палмер. М. : Мир, 1977. – 324 с.
21. Храмова, Т.В. Лекции по теории графов. Учебное пособие / Т.В. Храмова. – Новосибирск : СибГУТИ, 2011. – 98 с.
22. Хаггарти, Р. Дискретная математика для программистов / Р. Хаггарти. – М. : Техносфера, 2005. – 400 с.

Учебное издание

ЭЛЕМЕНТЫ ТЕОРИИ ГРАФОВ

Методические рекомендации

Составители:

МЕХОВИЧ Андрей Павлович

КАРАУЛОВА Татьяна Борисовна

Технический редактор *Г.В. Разбоева*

Компьютерный дизайн *Л.Р. Жигунова*

Подписано в печать 16.09.2020. Формат 60x84 $^1/_{16}$. Бумага офсетная.
Усл. печ. л. 2,79. Уч.-изд. л. 2,12. Тираж 60 экз. Заказ .

Издатель и полиграфическое исполнение – учреждение образования
«Витебский государственный университет имени П.М. Машерова».

Свидетельство о государственной регистрации в качестве издателя,
изготовителя, распространителя печатных изданий
№ 1/255 от 31.03.2014.

Отпечатано на ризографе учреждения образования
«Витебский государственный университет имени П.М. Машерова».
210038, г. Витебск, Московский проспект, 33.