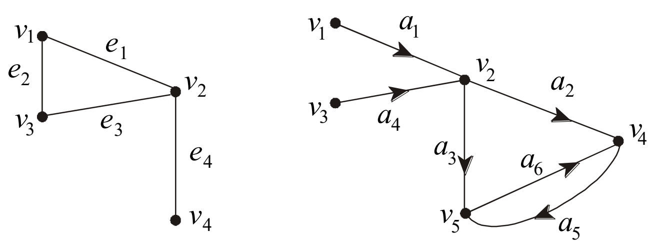
МОДУЛЬ 3

ТЕМА 9 Графы: связность, обходы, кратчайшие пути

9.1 Абстрактный граф

Граф можно определить как совокупность двух множеств: $G = (V, E)$, где V – непустое множество, элементы которого называются *вершинами*, и E – произвольное множество пар (v_i, v_j) элементов из множества V, т. е. $v_i \in V$, $v_j \in V$, $E \subseteq V^2$. Элементы множества E называются *ребрами*.

Само понятие графа подразумевает графическое представление данного объекта. Вершины изображаются точками, а ребра – линиями, соединяющими эти точки. Если ребра представляют упорядоченные пары вершин, соответствующие линии изображаются стрелками (рис. 9.1). Такие ребра называют *ориентированными ребрами* или, чаще, *дугами*. В этом случае имеем дело с *ориентированным* графом в отличие от *неориентированного* графа, на ребрах которого порядок вершин не задан.



а *б*

Рис. 9.1. Примеры графов:
а – неориентированный; *б* – ориентированный

Вершины неориентированного графа, связываемые ребром, считаются *концами* этого ребра. Например, концами ребра e_2 графа на рис. 9.1, *а* являются вершины v_1 и v_3. Принято обозначать ребра также парами их концов, например $e_2 = v_1v_3$. Всякая упорядоченная пара вершин (v_i, v_j), представляющая дугу в ориентированном графе, имеет *начало* v_i и *конец* v_j. Говорят, что дуга *выходит* из начала и *входит* в конец. В ориентированном графе на рис. 9.1, *б* началом дуги a_4 является вершина v_3, концом – вершина v_2. Это можно представить как $a_4 = (v_3, v_2)$.

Между вершинами и ребрами неориентированного графа так же, как между вершинами и дугами ориентированного графа, существует отношение *инцидентности*. При этом в неориентированном графе $G = (V, E)$ вершина $v \in V$ и ребро $e \in E$ *инцидентны*, если v является одним из концов ребра e. В ориентированном графе $G = (V, A)$ вершина $v \in V$ и дуга $a \in A$ *инцидентны*, если v является началом либо концом дуги a. Две вершины неориентированного графа *смежны*, если они инцидентны одному и тому же ребру.

Граф может содержать *петли*, т. е. ребра, концы которых совпадают, или дуги, у которых начало совпадает с концом. Очевидно, ориентация петли несущественна.

Множество всех вершин графа G, смежных с вершиной v, называется *окрестностью* вершины v и обозначается символом $N(v)$. Мощность множества $N(v)$, обозначаемая $d(v)$, называется *степенью* вершины v. В ориентированном графе с некоторой вершиной v подобным образом связаны два множества: *полуокрестность исхода* $N^+(v)$ – множество вершин, в которые входят дуги, исходящие из вершины v, и *полуокрестность захода* $N^-(v)$ – множество вершин, из которых исходят дуги, заходящие в v. Соответственно мощность множества $N^+(v)$ называется *полустепенью исхода* и обозначается $d^+(v)$, а мощность множества $N^-(v)$ – *полустепенью захода* и обозначается $d^-(v)$. Можно говорить об окрестности $N(v)$ и степени $d(v)$ вершины v ориентированного графа. При этом

$$N(v) = N^+(v) \cup N^-(v) \quad \text{и} \quad d(v) = d^+(v) + d^-(v).$$

Для неориентированного графа с множеством ребер E очевидно следующее соотношение:

$$\sum_{v \in V} d(v) = 2|E|,$$

откуда следует, что в любом неориентированном графе число вершин с нечетной степенью всегда четно.

Для ориентированного графа с множеством дуг A имеем

$$\sum_{v \in V} d^+(v) = \sum_{v \in V} d^-(v) = |A|.$$

В практических приложениях граф (ориентированный или неориентированный), как правило, является *конечным*, т. е. его множество вершин конечно. Специальный раздел теории графов изучает также *бесконечные графы*, у которых множество вершин бесконечно.

Граф $G = (V, E)$, у которого множество ребер пусто, т. е. $E = \varnothing$, называется *пустым* графом. Неориентированный граф называется *полным*, если любые две его вершины смежны. Полный граф, число вершин которого n, обозначается символом K_n.

Обозначим множество ребер полного графа символом U. *Дополнением* графа $G = (V, E)$ является граф $\overline{G} = (V, \overline{E})$, у которого $\overline{E} = U \setminus E$. Очевидно, что всякий полный граф является дополнением некоторого пустого графа и, наоборот, всякий пустой граф является дополнением некоторого полного графа.

Граф называется *двудольным*, если множество его вершин V разбито на два непересекающихся подмножества V' и V'', а концы любого его ребра находятся в различных подмножествах. Такой граф задается как $G = (V', V'', E)$ или как $G = (V', V'', A)$. В *полном двудольном* графе (V', V'', E) каждая вершина из V' связана ребром с каждой вершиной из V''. Полный двудольный граф, у которого $|V'| = p$ и $|V''| = q$, обозначается символом $K_{p,\,q}$.

Инварианты графа – характеристики графа, которые не меняются при изоморфных преобразованиях графа.

9.2 Графическое представление бинарного отношения

Граф является, по сути, графическим представлением бинарного отношения. Пусть задано бинарное отношение $R \subseteq A \times B$. Если $A \cap B = \varnothing$, то данное отношение можно представить двудольным ориентированным графом $G = (A, B, R)$, где каждая пара $(a, b) \in R$ представляется дугой, исходящей из вершины a и заходящей в вершину b. На рис. 3.2 представлено отношение R между элементами множеств A и B, где $A = \{a_1, a_2, a_3\}$, $B = \{b_1, b_2, b_3, b_4, b_5\}$, $R = \{(a_1, b_1), (a_1, b_2), (a_1, b_3), (a_1, b_5), (a_2, b_2), (a_2, b_4), (a_3, b_3)\}$.

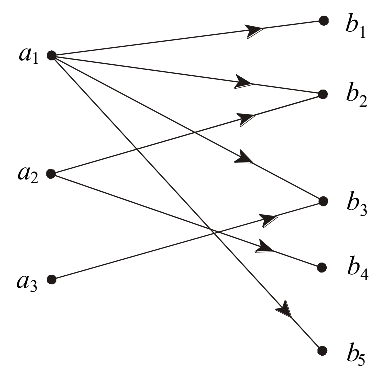


Рис. 9.2. Графическое представление отношения между элементами множеств A и B

Операция композиции отношений проиллюстрирована на рис. 9.3, где отношение R между элементами множеств A и B и отношение S между элементами множеств B и C показаны совместно (рис. 9.3, *а*). В представлении отношения SR на рис. 9.3, *б* видно, что вершина $a \in A$ соединена с вершиной $c \in C$ дугой тогда и только тогда, когда существует вершина $b \in B$, которая в графе на рис. 9.3, *а* является концом дуги, исходящей из a, и началом дуги, заходящей в c.

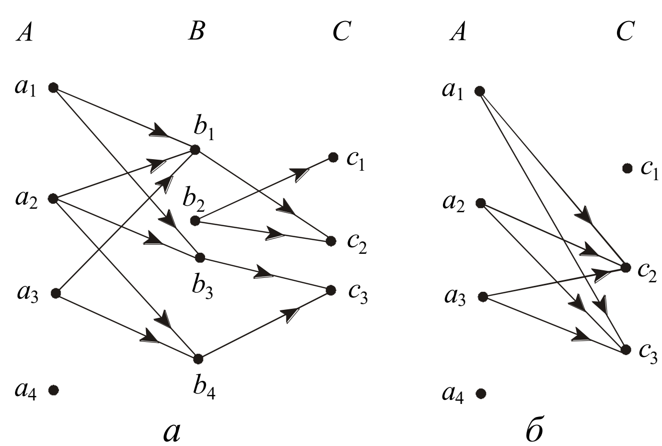


Рис. 9.3. Представление композиции отношений:
а – отношения R и S; *б* – отношение SR

В графическом представлении функционального отношения $R = \{(a, b), (c, b), (b, d), (e, d), (d, d)\}$ между элементами множеств $A = \{a, b, c, d, e\}$ и $B = \{b, d, e\}$, из каждой вершины выходит только одна дуга, включая петли (рис. 9.4).

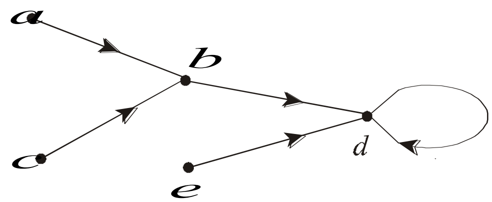


Рис. 9.4. Представление функционального отношения

9.3 Матричные представления графа

Поскольку граф можно рассматривать как графическое представление некоторого бинарного отношения, его можно задать той же булевой матрицей, которая задает данное отношение. Эта матрица называется *матрицей смежности* графа. Строки и столбцы матрицы смежности соответствуют вершинам графа, а элемент ее на пересечении строки v_i и столбца v_j имеет значение 1 тогда и только тогда, когда вершины v_i и v_j смежны. В матрице смежности ориентированного графа этот элемент имеет значение 1, если и только если в данном графе имеется дуга с началом в вершине v_i и концом в вершине v_j. Графы, показанные на рис. 9.1, имеют следующие матрицы смежности:

$$\begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \end{array} \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \end{array}, \quad \begin{array}{c} \begin{array}{ccccc} v_1 & v_2 & v_3 & v_4 & v_5 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array} \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array}.$$

Нетрудно видеть, что матрица смежности неориентированного графа обладает симметрией относительно главной диагонали.

Ориентированный граф можно задать также *матрицей инцидентности*, которая определяется следующим образом. Ее строки соответствуют вершинам графа, столбцы – дугам. Элемент на пересечении строки v и столбца a имеет значение 1, если вершина v является началом дуги a, и значение –1, если v является концом дуги a. Если вершина v и дуга a не инцидентны, то указанный элемент имеет значение 0. Матрица инцидентности неориентированного графа имеет тот же вид, только в ней все значения –1 заменяются на 1. Матрицы инцидентности графов, представленных на рис. 3.1, будут иметь следующий вид:

$$\begin{array}{cc} \begin{array}{cccc} e_1 & e_2 & e_3 & e_4 \end{array} & \\ \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} & \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \end{array} \end{array}, \quad \begin{array}{cc} \begin{array}{cccccc} a_1 & a_2 & a_3 & a_4 & a_5 & a_6 \end{array} & \\ \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 1 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 0 & -1 & 1 \end{bmatrix} & \begin{array}{c} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \end{array} \end{array}.$$

Заметим, что при матричном представлении графа его вершины, а также ребра или дуги оказываются упорядоченными. Любая строка матрицы смежности является векторным представлением окрестности соответствующей вершины. Любой столбец матрицы инцидентности неориентированного графа содержит ровно две единицы. Сумма значений элементов любого столбца матрицы инцидентности ориентированного графа равна нулю.

9.4 Части графа

Граф $H = (W, F)$ называется *подграфом* графа $G = (V, E)$, если $W \subseteq V$, $F \subseteq E$ и обе вершины, инцидентные любому ребру из F, принадлежат W. Подграф H графа G называется его *остовным подграфом*, если $W = V$. Если F является множеством всех ребер графа G, все концы которых содержатся в множестве W, то подграф $H = (W, F)$ называется *подграфом, порожденным множеством* W.

Любая последовательность вида $v_1, e_1, v_2, e_2, \ldots, e_k, v_{k+1}$, где $v_1, v_2, \ldots, v_{k+1}$ – вершины некоторого графа, а $e_1, e_2, \ldots, e_k$ – его ребра, причем $e_i = v_i v_{i+1}$ $(i = 1, 2, \ldots, k)$, называется *маршрутом*. Маршрут может быть конечным либо бесконечным. Одно и то же ребро может встречаться в маршруте не один раз. *Длиной маршрута* называется количество входящих в него ребер, причем каждое ребро считается столько раз, сколько оно встречается в данном маршруте.

Маршрут, все ребра которого различны, называется *цепью*. Цепь, все вершины которой различны, называется *простой цепью*. С понятием длины цепи связано понятие *расстояния* в графе. Под расстоянием между двумя вершинами понимается длина кратчайшей цепи, связывающей данные вершины.

Маршрут $v_1, e_1, v_2, e_2, \ldots, e_k, v_1$ называется *циклическим*. Циклическая цепь называется *циклом*. *Простой цикл* – это циклическая простая цепь.

Любую цепь и любой цикл графа можно рассматривать как его подграф.

Граф является *связным*, если между любыми двумя его вершинами имеется цепь. Связный подграф некоторого графа, не содержащийся ни в каком другом его связном подграфе, называется *компонентой связности* или просто *компонентой* данного графа.

В ориентированном графе *маршрутом* называется последовательность вида $v_1, a_1, v_2, a_2, \ldots, a_k, v_{k+1}$, где для всякой дуги a_i вершина v_i является началом, а v_{i+1} – концом. Вершина v_1 является началом маршрута, а вершина v_{k+1} – его концом. Маршрут, в котором все вершины, кроме, возможно, начальной и конечной, различны, называется *путем*. Путь вида $v_1, a_1, v_2, a_2, \ldots, a_k, v_1$ называется *контуром*.

Вершина v_j в ориентированном графе является *достижимой* из вершины v_j, если в этом графе имеется путь с началом в v_i и концом в v_j. Ориентированный граф является *сильно связным*, если любая его вершина достижима из любой вершины.

Ориентированный граф называется *транзитивным*, если из существования дуг $a_p = (v_i,v_j)$ и $a_q = (v_j,v_k)$ следует существование дуги $a_r = (v_i,v_k)$. *Транзитивным замыканием* ориентированного графа $G = (V, A)$ называется граф $G^* = (V, A^*)$, где A^* получено из A добавлением минимально возможного количества дуг, необходимого для того, чтобы граф G^* был транзитивным.

9.5 Обобщения графа

Существуют различные обобщения понятия графа. Одним из таких обобщений является *мультиграф*. Это граф, в котором любые две вершины могут быть связаны любым количеством ребер, т. е. мультиграф допускает *кратные ребра*.

В некоторых задачах используются графы, на множествах вершин или ребер которых заданы функции, принимающие значения из множеств действительных, целых или натуральных чисел. Эти значения называются *весами*. Тогда речь идет о *взвешенных графах*, о графах со *взвешенными вершинами*, со *взвешенными ребрами* или со *взвешенными дугами*. Графы со взвешенными ребрами используются в транспортных задачах и задачах о потоках в сетях. Мультиграф можно рассматривать как граф, ребра которого взвешены натуральными числами, представляющими кратности ребер.

Иногда рассматриваются *смешанные графы*, у которых наряду с элементами ориентированного графа (дугами) имеются элементы неориентированного графа (ребра). Ребром может быть заменена пара противоположно направленных дуг в ориентированном графе, соединяющих одни и те же вершины. Смешанные графы используются при решении задач, связанных с установлением схемы выполнения операций в технологическом процессе.

Еще одним обобщением понятия графа является *гиперграф*, который также представляет собой два множества – множество вершин и множество ребер, однако если ребром графа является пара вершин, то ребром гиперграфа может быть любое непустое подмножество множества вершин.

Гиперграф может служить моделью принципиальной электрической схемы. При этом полюса элементов данной схемы соответствуют вершинам гиперграфа, а электрические цепи – ребрам. Электрическая цепь здесь рассматривается как множество выводов, соединенных между собой проводниками. Многие понятия, связанные с графами, распространяются на случай гиперграфа, однако графически изобразить гиперграф гораздо труднее, чем граф. Вместе с тем от гиперграфа можно перейти к двудольному графу, долями которого являются множество вершин и множество ребер гиперграфа, а ребра показывают принадлежность вершин гиперграфа его ребрам.

9.6 Достижимость и связность

Граф является *связным*, если между любыми двумя его вершинами имеется цепь. Связный подграф некоторого графа, не содержащийся ни в каком другом его связном подграфе, называется *компонентой связности* или просто *компонентой* данного графа. Вершина v_j в ориентированном графе является *достижимой* из вершины v_i, если в этом графе имеется путь с началом в v_i и концом в v_j. Ориентированный граф является *сильно связным*, если любая его вершина достижима из любой вершины.

Матрицы достижимостей и контрдостижимостей.

Если существует путь, идущий от вершины x_i к вершине x_j, то говорят, что x_j достижима из вершины x_i. Обозначим через $R(x_i)$ множество вершин графа, достижимых из вершины x_i . Определим маршрут достижимостей $R=(r_{ij})$ с элементами

$$r_{ij} = \begin{cases} 1, \text{ если } x_j \text{ достижима из } x_i \\ 0 \text{ в противном случае} \end{cases}$$

Очевидно, что все диагональные элементы матрицы R равны 1 (x_i достижима из x_i).

Пусть через $\Gamma(x_i)$ обозначено множество вершин, которые достижимы из x_i с использованием путей длины 1, через $\Gamma^2(x_i)$ – множество вершин, достижимых из x_i с использованием путей длины 2, … , $\Gamma^p(x_i)$ – множество вершин, достижимых из x_i с использованием путей длины P. Тогда $R(x_i)$ можно представить в виде

$R(x_i) = \{x_i\}\cup\Gamma(x_i)\cup\Gamma^2(x_i)\cup\ldots\cup\Gamma^p(x_i)$,

где $\cup$ – операция объединения множеств.

Алгоритм Кристофидеса.

Множество $R(x_i)$ получается последовательным выполнением (слева направо) операций объединения в соотношении () до тех пор, пока «текущее» множество не перестает увеличиваться по размеру при очередной операции объединения.

Обозначим через $Q(x_i)$ множество вершин, из которых можно достичь вершину x_i. Определим маршрут контрдостижимостей $Q=(q_{ij})$ с элементами

$$q_{ij} = \begin{cases} 1, \text{ если } x_j \text{ достижима из } x_i \\ 0 \text{ в противном случае} \end{cases}$$

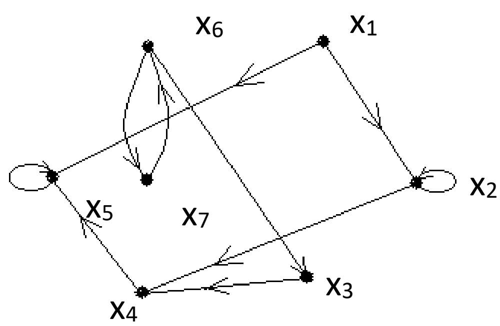
Пусть через $\Gamma^{-1}(x_i)$ обозначено множество вершин, из которых можно достичь вершину x_i. С использованием путей длины 1, $\Gamma^{-p}(x_i)$ – множество вершин, из которых можно достичь вершину x_i с использованием путей длинны p. Тогда $Q(x_i)$ можно представить в виде:

$Q(x_i) = \{x_i\}\cup\Gamma^{-1}(x_i)\cup\Gamma^{-2}(x_i)\cup\ldots\cup\Gamma^{-p}(x_i)$,

где $\cup$ – операция объединения множеств.

Пример:

Построить матрицы достижимостей и контрдостижимостей для графа *G*.



Матрица достижимостей $R = (r_{ij})$

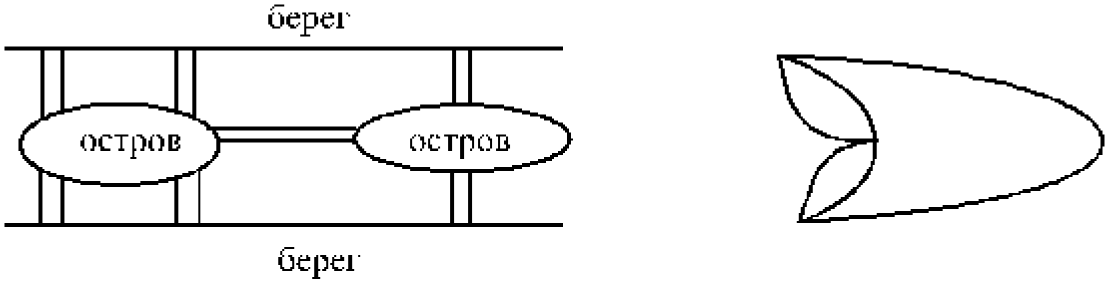
	1	**2**	**3**	**4**	**5**	**6**	**7**
1	1	1	0	1	1	0	0
2	0	1	0	1	1	0	0
3	0	0	1	1	1	0	0
4	0	0	0	1	1	0	0
5	0	0	0	0	1	0	0
6	0	0	1	1	1	1	1
7	0	0	1	1	1	1	1

Матрица контрдостижимостей $Q = (q_{ij})$

	1	**2**	**3**	**4**	**5**	**6**	**7**
1	1	0	0	0	0	0	0
2	1	1	0	0	0	0	0
3	0	0	1	0	0	1	1
4	1	1	1	1	0	1	1
5	1	1	1	1	1	1	1
6	0	0	0	0	0	1	1
7	0	0	0	0	0	1	1

9.7 Эйлеровы и гамильтоновы графы

Именно с задач, поставленных и решенных в этом разделе, началась теория графов. Философ Иммануил Кант, гуляя по городу Кенигсбергу (сейчас этот город называется Калининград), поставил задачу (1736), известную в математике как *задача о семи кенигсбергских мостах*: можно ли пройти по всем этим мостам и при этом вернуться в исходную точку так, чтобы по каждому мосту пройти только один раз. Наш петербургский знаменитый математик швейцарского происхождения Леонард Эйлер блестяще решил эту задачу. На рис. 2 изображена схема семи мостов Кенигсберга (заметим, что сейчас осталось только два из них), а также мультиграф, соответствующий этой схеме (при построении графа считалось, что каждый берег реки и острова – это вершины графа, а мосты – его ребра; видно, что в данном случае у нас получился мультиграф без петель).



В соответствии с поставленной Кантом (и решенной Эйлером) задачей можно дать следующие определения:

Граф (или мультиграф без петель) называется *эйлеровым*, если существует *цикл без повторения ребер* (такой цикл называют эйлеровым), обходящий все вершины графа. Граф называется *полуэйлеровым*, если существует *маршрут без повторения ребер* (эйлеров путь), обходящий все ребра графа ровно один раз. На рис. 9.5 изображены: а – эйлеров граф, б – полуэйлеров граф и в – граф, не являющийся ни эйлеровым, ни полуэйлеровым (люди старшего поколения знают, что в школах раньше было много загадок типа “можно ли нарисовать данную фигуру не отрывая ручку от бумаги”, что и соответствует эйлерову или полуэйлерову графу).

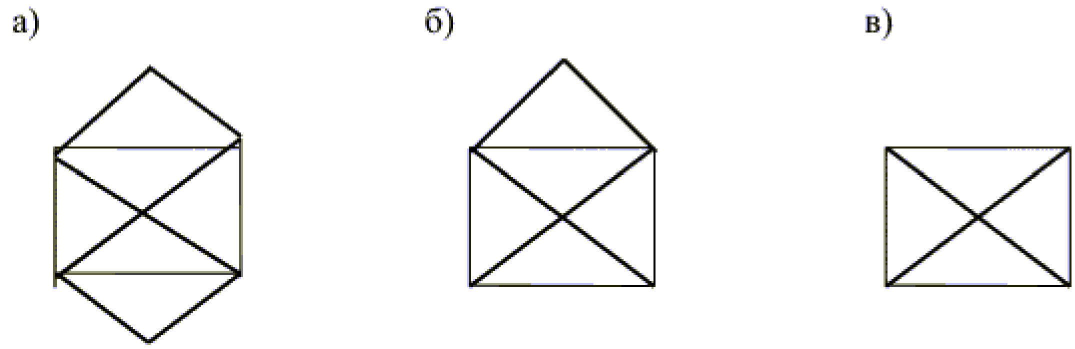


Рис. 9.5

Теорема *(Эйлер). Для того чтобы данный связный граф (не орграф, но, возможно, мультиграф без петель) был эйлеровым, необходимо и достаточно, чтобы степени всех вершин были четными. Данный связный граф будет полуэйлеровым тогда и только тогда, когда степени двух вершин будут нечетными, а степени остальных вершин – четными.*

Лемма о рукопожатиях. *Число вершин в графе (или мультиграфе без петель), имеющих нечетную степень, четно.*

Примечание. Если граф (или мультиграф без петель) содержит $2k$ вершин нечетной степени, то его можно разбить на k полуэйлеровых графов (т. е. нарисовать k росчерками пера). Доказательство аналогично доказательству теоремы Эйлера.

Имеется простой алгоритм (так называемый *алгоритм Флери*) для нахождения эйлерова цикла (конечно, если этот цикл существует), который состоит в следующем: *начинаем с любой вершины и* “*стираем*” *пройденные ребра. При этом по мосту (перешейку) проходим только, если нет других возможностей.*

Очевидно, что для того чтобы построить эйлеров путь достаточно использовать алгоритм Флери, который надо начать с вершины, имеющей нечетную степень.

Рассмотрим некоторые приложения теоремы Эйлера, которые в основном связаны с так называемой *задачей китайского почтальона*.

Пусть имеется некоторый граф (связный), ребрам которого приписаны некоторые числа, называемые *весами* ребер (часто, но не всегда!, в приложениях вес ребра – это его длина). Требуется найти такой цикл, при котором каждое ребро проходится *по крайней мере один раз* и суммарный вес всех ребер, вошедших в цикл, минимален. Заметим, что если граф является эйлеровым, то любой эйлеров цикл решает поставленную задачу (для эйлерова графа веса роли не играют).

Эта задача имеет много приложений, например, поливка улиц одной машиной (здесь ребра графа – дороги, а перекрестки – вершины; веса – это длины дорог), а также сбор мусора, доставка почты или даже наилучший маршрут для осмотра музея или уборка помещений и коридоров в больших учреждениях.

Кратко рассмотрим проблему, связанную с возможным обходом всех вершин в графе: *существует ли в данном (связном) графе цикл (или маршрут), обходящий каждую вершину (кроме первой) только один раз.* Если такой цикл (маршрут) существует (в этом случае такой цикл будет контуром, а маршрут – путем), то граф называется *гамильтоновым (полугамильтоновым)*, и соответствующий цикл (путь) также называют *гамильтоновым циклом (путем).*

На рис. 9.6 изображены гамильтонов, полугамильтонов и не гамильтонов графы.

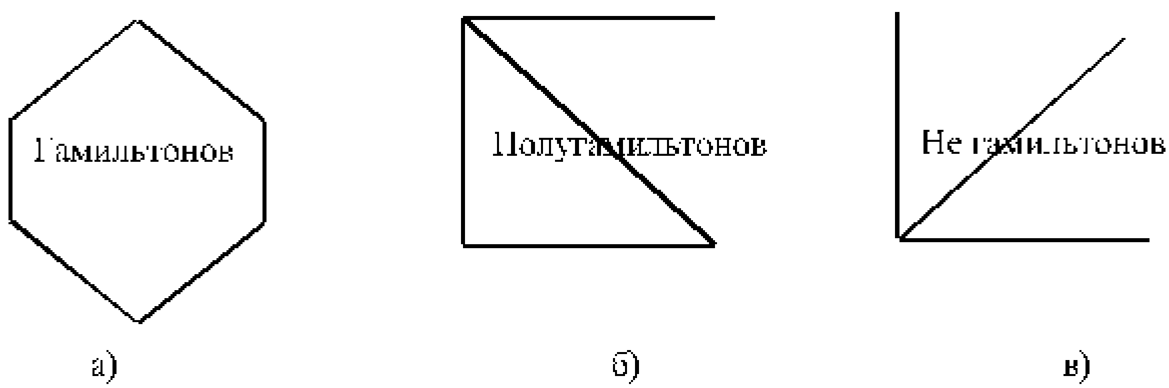


Рис. 9.6

Несмотря на сходство постановки задач для гамильтоновых графов с эйлеровыми, “хорошего” решения для гамильтоновых графов нет. Вообще, о гамильтоновых графах известно очень мало. В основном – это теоремы типа “если в графе достаточное число ребер, то он гамильтонов”.

Приведем без доказательства самую известную теорему.

Теорема (Дирак, 1952). *Если в связном графе с n вершинами (при $n \geq 3$) степени всех вершин больше или равны $n/2$, то граф гамильтонов.*

9.8 Методы поиска в графе

Существует много алгоритмов на графах, в основе которых лежит систематический перебор вершин графа, такой, что каждая вершина просматривается (посещается) в точности один раз. Поэтому важной задачей является нахождение хороших методов поиска в графе.

Под *обходом графов* (*поиском на графах*) понимается процесс систематического просмотра всех ребер или вершин графа с целью отыскания ребер или вершин, удовлетворяющих некоторому условию.

При решении многих задач, использующих графы, необходимы эффективные методы регулярного обхода вершин и ребер графов. К стандартным и наиболее распространенным методам относятся:

– поиск в глубину (Depth First Search, DFS);

– поиск в ширину (Breadth First Search, BFS).

Эти методы чаще всего рассматриваются на ориентированных графах, но они применимы и для неориентированных, ребра которых считаются двунаправленными. Алгоритмы обхода в глубину и в ширину лежат в основе решения различных задач обработки графов, например, построения остовного леса, проверки связности, ацикличности, вычисления расстояний между вершинами и других.

При поиске в глубину посещается первая вершина, затем необходимо идти вдоль ребер графа, до попадания в тупик. Вершина графа является *тупиком*, если все смежные с ней вершины уже посещены. После попадания в тупик нужно возвращаться назад вдоль пройденного пути, пока не будет обнаружена вершина, у которой есть еще не посещенная вершина, а затем необходимо двигаться в этом новом направлении. Процесс оказывается завершенным при возвращении в начальную вершину, причем все смежные с ней вершины уже должны быть посещены. Таким образом, основная идея поиска в глубину формулируется следующим образом: когда возможные пути по ребрам, выходящим из вершин, разветвляются, необходимо сначала полностью исследовать одну ветку и только потом переходить к другим (если они останутся нерассмотренными).

Алгоритм поиска в глубину:

Шаг 1. Всем вершинам графа присваивается название «не посещенная». Выбирается первая вершина и называется «посещенной».

Шаг 2. Для последней посещенной вершины выбирается смежная не посещенная вершина, и ей присваивается название посещенной. Если таких вершин нет, то берется предыдущая посещенная вершина.

Шаг 3. Повторять шаг 2 до тех пор, пока все вершины не окажутся посещенными.

Пример исследования графа в глубину представлен на рис. 9.7.

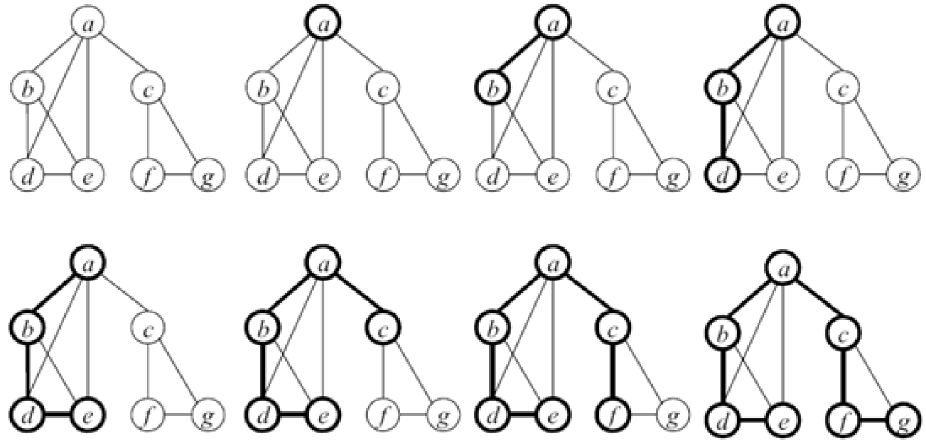


Рис. 9.7. Пример поиска в глубину

При поиске в ширину после посещения первой вершины посещаются все соседние с ней вершины. Потом посещаются все вершины, находящиеся на расстоянии двух ребер от начальной. При каждом новом шаге посещаются вершины, расстояние от которых до начальной на единицу больше предыдущего. Чтобы предотвратить повторное посещение вершин, следует вести список посещенных вершин. Для хранения временных данных, необходимых для работы алгоритма, используется очередь – упорядоченная последовательность элементов, в которой новые элементы добавляются в конец, а старые удаляются из начала.

Таким образом, основная идея поиска в ширину заключается в том, что сначала исследуются все вершины, смежные с начальной (вершина, с которой начинается обход). Эти вершины находятся на расстоянии 1 от начальной. Затем исследуются все вершины на расстоянии 2 от начальной, затем все на расстоянии 3 и т. д. Обратим внимание, что при этом для каждой вершины сразу находится длина кратчайшего маршрута от начальной вершины.

Алгоритм поиска в ширину:

Шаг 1. Всем вершинам графа дается название «не посещенная». Выбирается первая вершина и называется «посещенной» (и заносится в очередь).

Шаг 2. Посещается первая вершина из очереди (если она не посещенная). Все ее соседние вершины заносятся в очередь. После этого она удаляется из очереди.

Шаг 3. Повторяется шаг 2 до тех пор, пока очередь не пуста.

Пример исследования графа в ширину представлен на рис. 9.8.

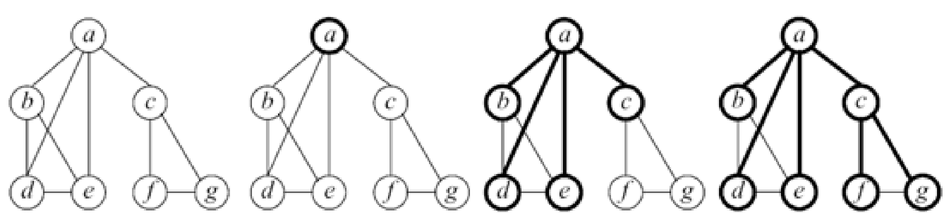


Рис. 9.8. Пример поиска в ширину

Поиск в ширину можно использовать при определении бихроматичности графа. Произвольно выбранной вершине графа припишем индекс 1, а смежным с ней вершинам – индекс 2. Вершинам из окрестностей вершин с индексом 2 пытаемся приписать индекс 1. Затем вершинам из окрестностей вершин с индексом 1 пытаемся приписать индекс 2. Если в результате такого процесса каждая вершина примет либо индекс 1, либо индекс 2, то граф бихроматический. Если же при этом в окрестности вершины с одним индексом окажется вершина с тем же индексом, то граф не является бихроматическим. Вершине a в графе на рис. 9.8 припишем индекс 1, а вершинам b, c, d и e – индекс 2. Вершины d и e из окрестности вершины b имеют тот же индекс, что и вершина b. Значит, этот граф не является бихроматическим.