Понятие графа, основные методы

Пусть $G=(V, E)$ – граф, состоящий из множества вершин V и множества ребер E. Ребра, соединяющие элементы множества V, могут быть ориентированными (их называют в этом случае дугами) или неориентированными и составлять соответственно ***ориентированный граф*** (***орграф***) или ***неориентированный граф***.

Граф называется ***помеченным*** (или пронумерованным), если его вершинам приписаны различные метки. Обычно в качестве меток используются целые положительные числа в интервале от 1 до n, где n равно количеству вершин графа – $|V|$. Количество ребер будем обозначать как $m=|E|$. На рис. 5.1 дан пример непомеченного, неориентированного графа (точки пересечения ребер не являются вершинами графа), а на рис. 5.2 – помеченного и ориентированного.

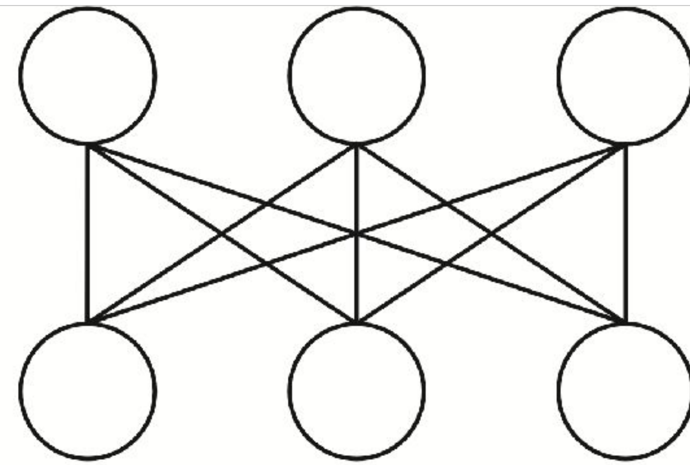
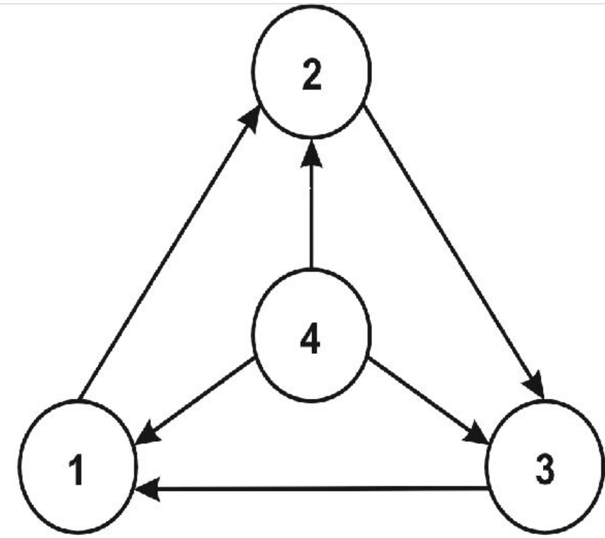


Рис. 5.1 Пример непомеченного и неориентированного графа



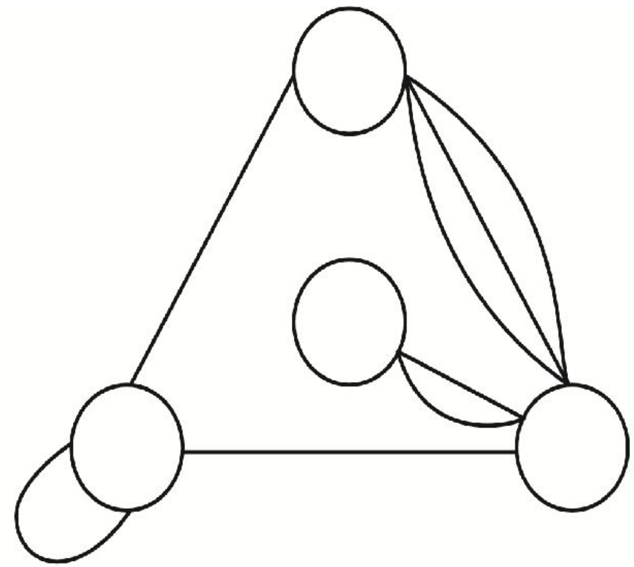


Рис. 5.2. Пример помеченного, ориентированного графаРис. 5.3. Пример непомеченного мультиграфа с петлями

Вершины, соединенные ребром, называются *смежными*. Ребра, имеющие общую вершину, также называются смежными. Ребро и любая из его двух вершин называются *инцидентными*. Ребро (i,j) инцидентно вершинам i и j. Каждый граф можно представить на плоскости множеством точек, соответствующих вершинам, которые соединены линиями, соответствующими ребрам. В трехмерном пространстве любой граф можно представить таким образом, что линии (ребра) не будут пересекаться.

Способы представления графа

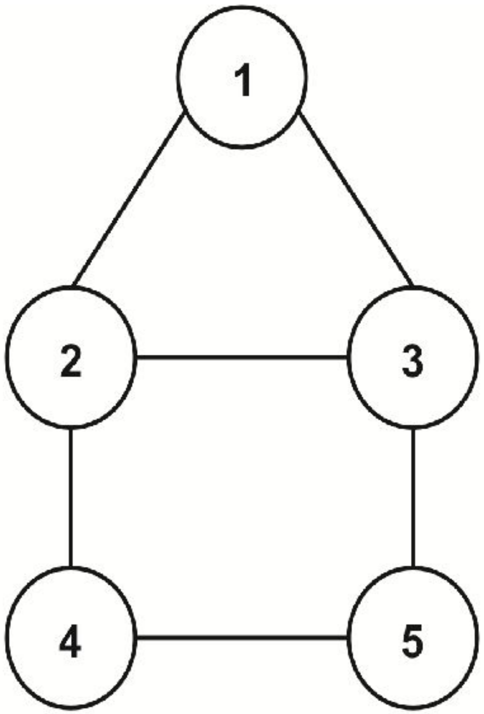
Выбор соответствующей структуры данных для представления графа имеет принципиальное значение при разработке эффективных алгоритмов. При решении задач используются следующие четыре основных способа описания графа: *матрица смежности, матрица инциденций; списки связи и перечень ребер*. На рис. 5.4 приведен пример графа и четыре способа его представления в памяти ЭВМ.

Матрица смежности A – это двумерный массив размерности $n*n$.

$$A[i,j] = \begin{cases} 1, \textit{если вершины с номерами } i \textit{ и } j \textit{ смежны} \\ 0, \textit{если вершины с номерами } i \textit{ и } j \textit{ не смежны} \end{cases}$$

Для задания матрицы инциденций необходимо пометить и ребра графа. На рис. 5.4 эта разметка не приведена. Подразумевается, что ребро (1,2) имеет метку 1, ребро (1,3) – метку 2 и так далее. Матрица инциденций B – это двумерный массив размерности $n*m$.

$$B[i,j] = \begin{cases} 1, \textit{если ребро } j \textit{ инцидентно вершине с номером } i \\ 0, \textit{если ребро } j \textit{ не инцидентно вершине с номером } i \end{cases}$$



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

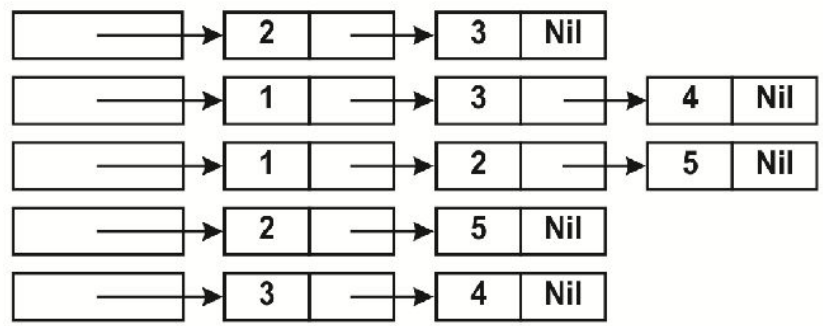
а)

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

б)

$$W = \begin{pmatrix} 1 & 2 \\ 1 & 3 \\ 2 & 3 \\ 2 & 4 \\ 3 & 5 \\ 4 & 5 \end{pmatrix}$$

в)



г)

Рис. 5.4. Пример графа и 4 способа его описания:
а) матрица смежности;
б) матрица инциденций;
в) списки связи;
г) перечень ребер

Связь между помеченными и непомеченными графами. Граф G называется *полным* (будем обозначать как K_n), если любые две его вершины смежны. Число ребер в полном графе равно $C_n^2 = \frac{n*(n-1)}{2}$. Количество помеченных графов с фиксированным множеством вершин V равно количеству подмножеств множества его ребер, то есть $2^{C_n^2}$. На рис. 5.5 показано, как из одного непомеченного графа получаются три разных помеченных графа. Количество непомеченных графов с n вершинами определить достаточно сложно. Обычно используют интуитивно ясную асимптотическую оценку, известную как формула Пойа $\frac{2^{C_n^2}}{n!}$. Другими словами, количество непомеченных графов с n вершинами в $n!$ раз меньше количества помеченных. Это утверждение асимптотически (как правило) верно, и, например, не выполняется при n=3 (рис. 5.5). Так, количество помеченных графов при n=3, равно 3, а не 6.

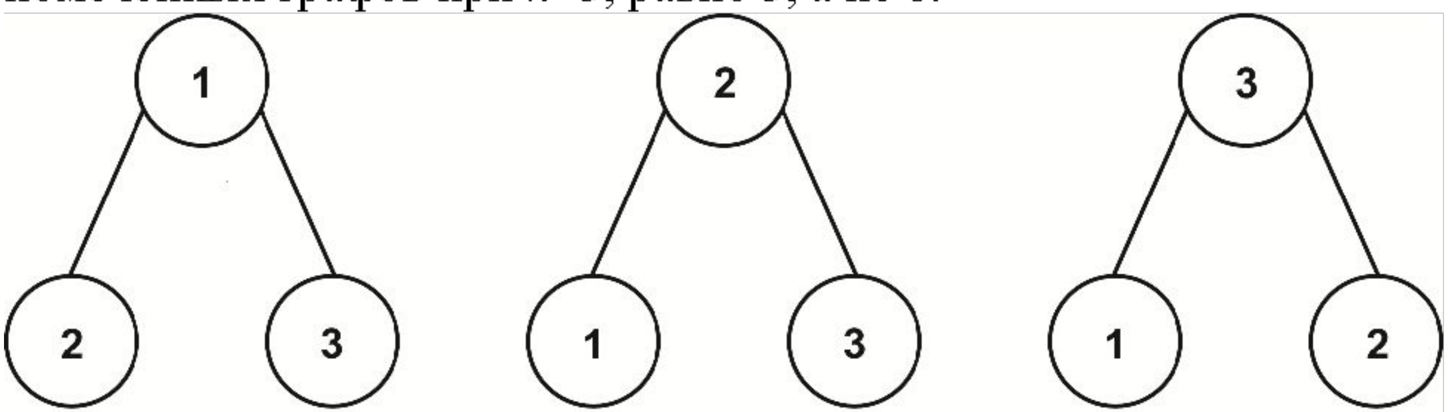


Рис. 5.5. Пример помеченного графа. Количество помеченных графов с *n=3* равно 3

Поиск в глубину

Идея метода. Поиск начинается с некоторой фиксированной вершины v. Рассматривается вершина u, смежная с v. Она выбирается. Процесс повторяется с вершиной u. Если на очередном шаге мы работаем с вершиной q и нет вершин, смежных с q и не рассмотренных ранее (новых), то возвращаемся из вершины q к вершине, которая была до нее. В том случае, когда это вершина v, процесс просмотра закончен. Очевидно, что для фиксации признака, просмотрена вершина графа или нет, требуется структура данных типа: *Nnew : Array*[*1..n*] *Of Boolean.*

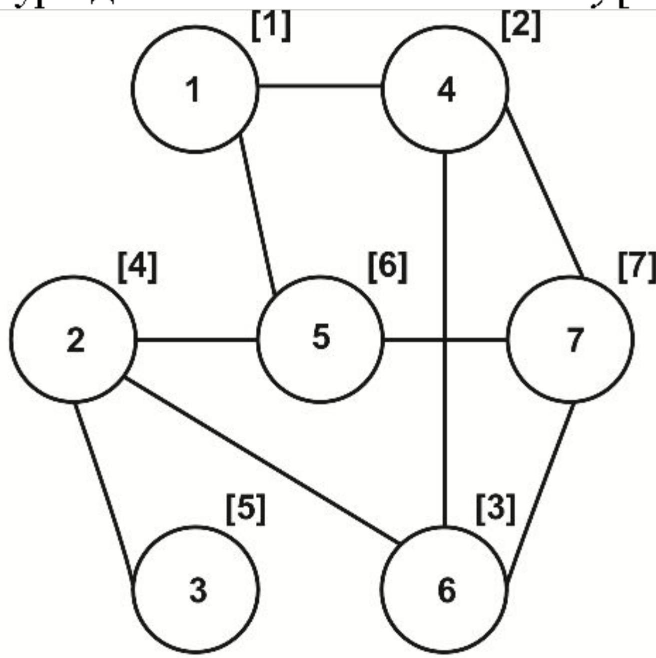


Рис. 5.6. Пример обхода вершин графа методом в глубину. Вершины графа просматриваются в очередности: 1, 4, 6, 2, 3, 5, 7. Очередность (порядок просмотра) приводится в квадратных скобках

Логика.

```
Procedure Pg(v:Integer); {Массивы Nnew и A глобальные.}
 Var j:Integer;
 Begin
  Nnew[v]:=False;
  Write(v:3);
  For j:=1 To n Do
   If (A[v,j]<>0) And Nnew[j] Then Pg(j);
 End;
```

Фрагмент основной логики.

```
For i:=1 To n Do Nnew[i]:=True;
For i:=1 To n Do
  If Nnew[i] Then Pg(i);
```

В силу важности данного алгоритма рассмотрим его нерекурсивную реализацию. Глобальные структуры данных прежние: *A* – матрица смежностей; *Nnew* – массив признаков. Номера просмотренных вершин графа запоминаются в стеке *St*, указатель стека – переменная *yk*.

```
Procedure Pgn(v:Integer);
 Var St:Array[1..n] Of Integer;
 yk, t, j:Integer;
 pp:Boolean;
Begin
 For j:=1 To n Do St[j]:=0;
 yk:=0;
 Inc(yk);
 St[yk]:=v;
 Nnew[v]:=False;
 While yk<>0 Do Begin {Пока стек не пуст.}
  t:=St[yk];{Выбор «самой верхней» вершины из стека.}
  j:=1;
  pp:=False;
  Repeat
   If (A[t,j] <>0) And Nnew[j] Then pp:=True Else Inc(j);
  Until pp Or (j>=n); {Найдена новая вершина или все вершины, связанные с данной вершиной, просмотрены.}
  If pp Then Begin
               Inc(yk);
               St[yk]:=j;
               Nnew[j]:=False;{Добавляем номер вершины в стек.}
              End
   Else Dec(yk); {«Убираем» номер вершины из стека.}
 End;
End;
```

Поиск в ширину

Идея метода. Суть (в сжатой формулировке) заключается в том, чтобы рассмотреть все вершины, связанные с текущей. Начинаем с произвольной вершины, помечаем её как просмотренную. Находим все вершины, связанные с ней, помечаем их и запоминаем. Множество всех вершин графа в данный момент разбито (условно) на три подмножества: подмножество просмотренных и обработанных вершин (оно состоит из одной вершины); подмножество помеченных, но необработанных вершин (оно состоит из всех вершин, смежных с обработанной) и подмножество непомеченных и необработанных вершин. Из второго подмножества выбирается для обработки первая запомненная вершина. Она естественно переходит в первое подмножество и с ней проделываются те же действия – находятся смежные и непомеченные вершины (из третьего подмножества), которые «переводятся» в разряд помеченных и необработанных – второе подмножество. Процесс обработки заканчивается в том случае, когда второе подмножество оказывается пустым – нет помеченных и необработанных вершин. Для реализации данного принципа обработки требуется структура данных для хранения вершин из второго подмножества. Этой структурой данных является очередь, ибо на очередную обработку выбирается первая, помеченная ранее, вершина.

Пример 5.2. На рис. 5.7 приведен тот же граф, что и на рис. 5.6. Просмотр вершин начинается с первой вершины. Очередность просмотра вершин другая, чем при просмотре методом поиска в глубину. В процессе просмотра в ширину осуществляется переход от вершине к вершине по ребрам (1,4), (1,5), (4,6), (4,7), (5,2) и (2,3), всего n-1 ребро. Ребра (2,6), (5,7) и (6,7) не задействованы в просмотре.

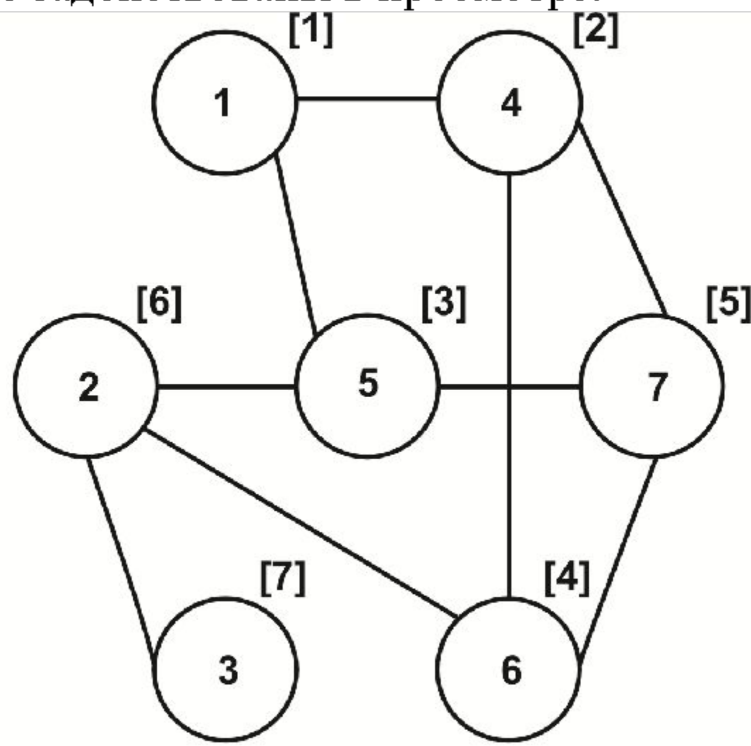


Рис. 5.7. Пример обхода вершин графа методом в ширину. Вершины графа просматриваются в очередности: 1, 4, 5, 6, 7, 2, 3. Очередность (порядок просмотра) приводится в квадратных скобках

```
Procedure Pw(v:Integer);
 Var  Turn:Array[1..n] Of 0..n;{Очередь.}
       up,down:Integer;{Указатели очереди, up - запись; down – чтение.}
       j:Integer;
 Begin
  For j:=1 To n Do Turn[j]:=0;
   down:=0; {Начальная инициализация.}
   up:=0;
   Inc(up); {В очередь записываем вершину с номером v.}
   Turn[up]:=v;
   Nnew[v]:=False;
  While down<up Do Begin {Пока очередь не пуста.}
   Inc(down); {«Берем» элемент из очереди.}
   v:=Turn[down];
   For j:=1 To n Do {Просмотр всех вершин, связанных с вершиной v.}
    If (A[v,j]<>0) And Nnew[j] Then Begin {Если вершина ранее не просмотрена, то заносим её номер в очередь.}
     Inc(up);
     Turn[up]:=j;
     Nnew[j]:=False;
    End;
  End;
 End;
```

Основные понятия

Подграф. Граф H называется *подграфом* графа G, если все его вершины и ребра принадлежат графу G. *Остовной подграф* – это подграф графа G, содержащий все его вершины. *Клика* графа – это его максимальный полный подграф. На рис. 5.9 (б) граф состоит из двух клик.

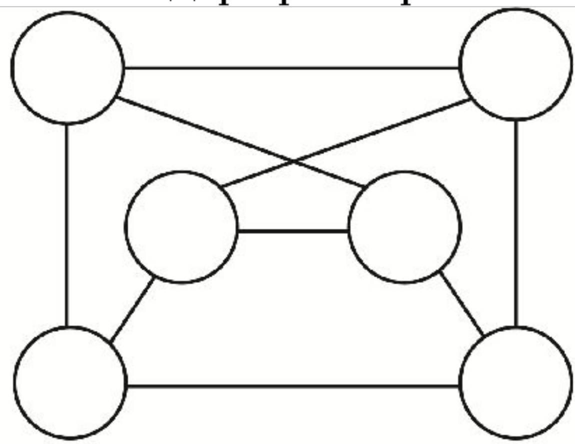
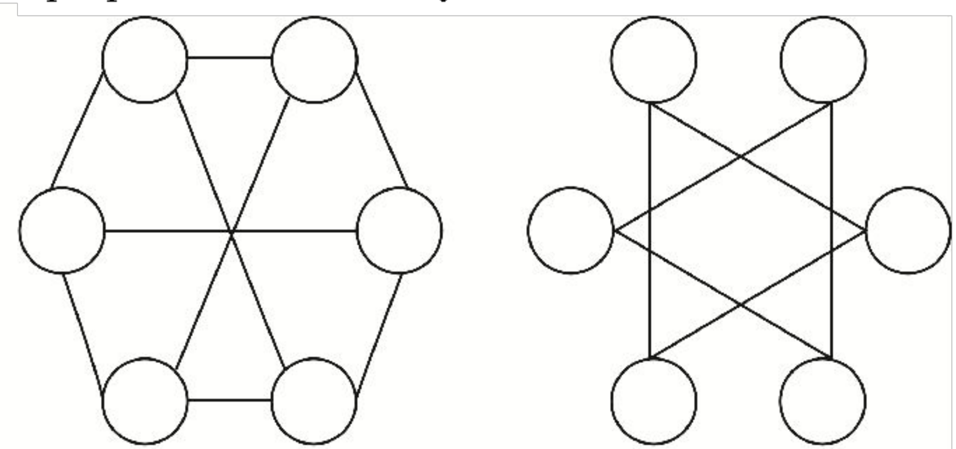


Рис. 5.8. Пример графа, изоморфного графу на рис. 5.1



а) б)

Рис. 5.9. Граф и его дополнение

Изоморфные графы. Два графа G и H изоморфны ($G \cong H$), если между их множествами вершин можно установить соответствие, при котором сохраняется смежность. Другими словами, если вершины v_i, $v_j \in G$ и они смежные, то соответствующие им вершины u_i, $u_j \in H$ также смежные, и наоборот. Графы на рис. 5.1, 5.8 и 5.9 (а) изоморфны между собой.

Степень вершины графа. Степенью вершины графа G называется число инцидентных ей ребер и обозначается d_i или $deg\ i$. Максимальная и минимальная степени вершин графа G обозначаются символами $\Delta(G)$ и $\delta(G)$. Вершина степени 0 называется изолированной, а степени 1 – концевой, или висячей. Ребро, инцидентное концевой вершине, также определяют как концевое, или висячее. Граф с $\Delta(G) = \delta(G) = t$ (некоторое постоянное число) называют регулярным (или однородным).

Теорема о сумме степеней вершин графа. Эйлер доказал следующую теорему (исторически первая теорема теории графов).

Теорема (лемма о рукопожатиях). Сумма степеней всех вершин графа – четное число, равное удвоенному числу ребер: $\sum_{i=1}^{n} d_i = 2 * m$.

Доказательство теоремы очевидно, ибо каждое ребро вносит двойку в значение суммы.

Следствие теоремы. В любом графе число вершин нечетной степени четно.

Цепи, циклы. Чередующаяся последовательность $v_1, e_1, v_2, e_2, \dots, e_l, v_{l+1}$ вершин и ребер графа, такая $e_i=v_iv_{i+1}$ (i=1..l) называется *маршрутом*, соединяющим вершины v_1 и v_{l+1}. Маршрут можно задать как последовательностью его вершин $v_1, v_2, \dots, v_{l+1}$, так и последовательностью его ребер $e_1, e_2, \dots, e_l$. Маршрут определяют как *цепь*, если все его ребра различны, и как *простую цепь*, если все его вершины, кроме, возможно, крайних, различны. При $v_1 = v_{l+1}$ маршрут называется *циклическим*. Циклическая цепь определяется как *цикл*, а простая циклическая цепь – как *простой цикл*. Аналогичные определения могут быть введены и для ориентированных графов. На рис. 5.15 приведен пример графа. Маршрут (1, 4, 2, 4, 5) не является цепью. Цепь (1, 2, 5, 4, 2, 3) не является простой цепью, а (1, 2, 4, 5, 3) – простая цепь. Цепь (1, 4, 2, 5, 3, 2, 1) есть цикл, а (1, 4. 5, 3, 2, 1) – простой цикл.

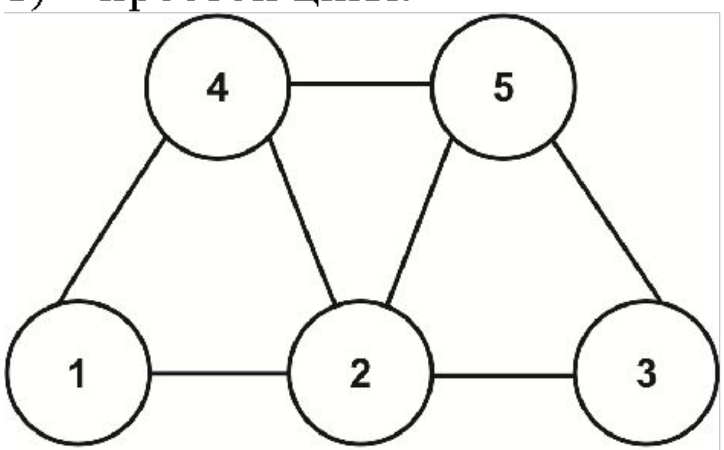


Рис. 5.15. Пример графа для иллюстрации понятий: маршрут, цепь, простая цепь, цикл, простой цикл

Справедливы следующие утверждения.

1. При $u \neq v$ всякий (u,v) маршрут содержит простую (u,v) цепь.

Действительно, при $u \neq v$ из маршрута удалить часть ребер так, что получится простая цепь, то есть цепь, в которой все вершины различны. Аналогичным образом обосновывается и следующее утверждение.

2. Всякий цикл содержит простой цикл.

3. Объединение двух несовпадающих простых (u,v) цепей содержит простой цикл.

Действительно, пусть есть две простые цепи $T=(t_1, t_2, t_3, \ldots, t_\mathrm{n})$, $R=(r_1, r_2, r_3, \ldots, r_\mathrm{m})$, не совпадающие полностью, и $t_1= r_1=u$, $t_n=r_m=v$. Из этого следует существования двух вершин в цепях (если «идти» от u), в которых цепи не совпадают. Обозначим их как t_i и r_i. Кроме того, после этих вершин в цепях есть вершины t_j и r_j, в которых эти цепи вновь совпадают. Это может оказаться и последняя вершина v. Объединение двух фрагментов цепей $(t_{i\text{-}1}, \ldots, t_j)$ и $=(r_{i\text{-}1}, \ldots, r_j)$в единое целое дает простой цикл, ибо повторяющихся вершин в нем не может быть по построению.

4. Если A и B – два несовпадающих простых цикла, имеющих общее ребро q, то граф $(A \cup B)\text{-}q$ содержит простой цикл.

Удаление из циклов A и B ребра q приводит к тому, что они становятся простыми цепями. Из предыдущего утверждения следует, что в объединении двух простых несовпадающих цепей содержится простой цикл.

Связность. Граф G называется *связным*, если любая пара его вершин соединена простой цепью. Максимальный связный подграф графа G называется *компонентой связности*, или просто компонентой графа G.

Справедливы следующие утверждения.

1. Для связности графа необходимо и достаточно, чтобы в нем для какой-либо фиксированной вершины u и каждой другой вершины v существовал (u,v) маршрут.

Утверждение очевидное и следует из определения связности.

2. Каждый граф представляется в виде объединения своих связных компонент. Разложение графа на связные компоненты определяется однозначно.

Метрические характеристики графа. Длина маршрута $v_1, v_2, \ldots, v_{l+1}$ равна l, или количеству ребер в нем. При этом каждое ребро учитывается столько раз, сколько оно встречается в маршруте. *Расстоянием* $d(u,v)$ между двумя вершинами графа G определяется как длина кратчайшей простой цепи, соединяющей вершины u и v. В связном графе расстояние является метрикой, то есть удовлетворяет метрическим аксиомам. Пусть u, v и w любые три вершины G. Для них выполняется:

1) $d(u,v) \geq 0$ и $d(u,v)=0$ тогда и только тогда, когда $u=v$;
2) $d(u,v)=d(v,u)$;
3) $d(u,v)+d(v,w) \geq d(u,w)$.

Для каждой вершины u графа можно определить величину $e(u) = \max\limits_{v \in V} d(u,v)$, которая называется *эксцентриситетом* вершины u. *Диаметр* связного графа G – это максимальное значение эксцентриситета – $d(G) = \max\limits_{u \in V} e(u)$. *Радиус* связного графа G – это значение минимального эксцентриситета, $r(G) = \min\limits_{u \in V} e(u) = \min\limits_{u \in V} \max\limits_{v \in V} d(u,v)$. Вершина v называется *центральной*, если $e(v)=r(G)$. В графе может быть как одна центральная вершина, так и несколько. *Центр* графа G – это множество всех его центральных вершин.

Двудольный граф. Граф $G=(V,E)$ называется двудольным, если его множество вершин V можно разбить на два подмножества V_1 и V_2, таких, что каждое ребро из E соединяет вершины из разных подмножеств.

Пример. Граф на рис. 5.1 является двудольным.

Теорема Д. Кёнига (1936 г.) Для двудольности графа необходимо и достаточно, чтобы он не содержал простых циклов нечетной длины.

Теорема Денеша Кёнига подсказывает простой способ определения того, является ли произвольный связный граф G двудольным. Обратимся к поиску в ширину и рис. 5.7. Вместо расстояний припишем вершинам графа условные номера (рис. 5.16). Вершина с номером 1 имеет условный номер 0. Вершинам с номерами 4 и 5 припишем условный номер 1. Вершинам с номерами 6, 7, 2 – условный номер 2 и вершине с номером 3 – условный номер 3. Далее разбиваем все множество вершин графа G на два подмножества V_1 и V_2. К V_1 относятся вершины с четным значением условного номера, к V_2 – с нечетным значением. Если оба подграфа $G(V_1)$ и $G(V_2)$ пусты, то граф двудольный. Для проверки последнего утверждения достаточно определить наличие смежных вершин в подграфах.

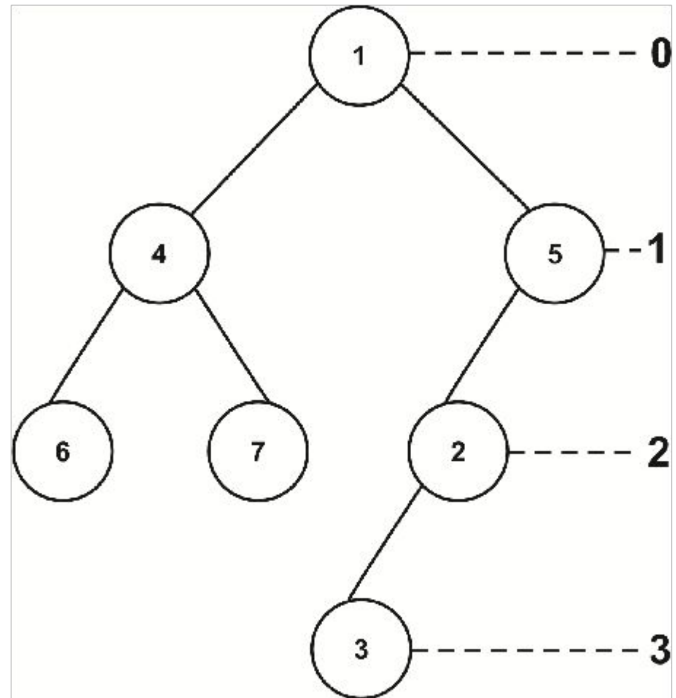


Рис. 5.16. Часть графа, показанного на рис. 5.7. Использованы только ребра, задействованные при поиске в ширину