Введение в теорию графов

Множество самых разнообразных задач естественно формулируется в терминах точек и связей между ними, т.е. в терминах теории графов. Так, например, могу быть сформулированы задачи составления расписаний, анализа цепей в электротехнике, различные задачи в программировании и т.д. Поэтому эффективные алгоритмы решения задач теории графов имеют большое практическое значение.

Основные понятия и определения

Графом называется тройка $\Gamma = (X, U, \Phi)$, где X - конечное множество вершин; U - конечное множество рёбер; Φ - отношение инцидентности; $X \cap U = \emptyset$. Отношение инцидентности Φ является трёхместным отношением $\Phi(x, u, y)$, где $x, y \in X, u \in U$, которое может либо выполняться (быть истинным), либо не выполняться (быть ложным). Кроме того, это отношение должно удовлетворять следующим свойствам:

- $\forall u \in U\ \exists x, y, \in X\ \Phi(\mathrm{x}, \mathrm{u}, \mathrm{y})$ - ребро всегда соединяет пару вершин
- $\big(\Phi(x, u, y) \wedge \Phi(\mathrm{x}', \mathrm{u}, \mathrm{y}')\big) \Longrightarrow ((x = x' \wedge y = y') \vee (x = y' \wedge y = x'))$ - ребро u соответствует не более чем одной паре вершин x, y

Таблица 1 Графическое представление графов

Элементы графов	Графические элементы
$x \in X$ – вершина	Точка в пространстве
$\Phi(x, u, y) \wedge \overline{\Phi(y, u, x)}$ – ориентированное ребро	Направленный отрезок
$\Phi(x, u, y) \wedge \Phi(y, u, x)$ – неориентированное ребро	Отрезок
$\Phi(x, u, x)$ – петля	Замкнутый отрезок

$\Gamma_1 = (X_1, U_1, \Phi_1)$ и $\Gamma_2 = (X_2, U_2, \Phi_2)$ называются изоморфными $\Gamma_1 \cong \Gamma_2$, если существуют два взаимно однозначных соответствия $\varphi: X_1 \longrightarrow X_2$ и $\psi: U_1 \longrightarrow U_2$, сохраняющие отношение инцидентности: $\Phi_2\big(\varphi(x_1), \psi(u_1), \varphi(y_1)\big) = \Phi_1(x_1, u_1, y_1)$.

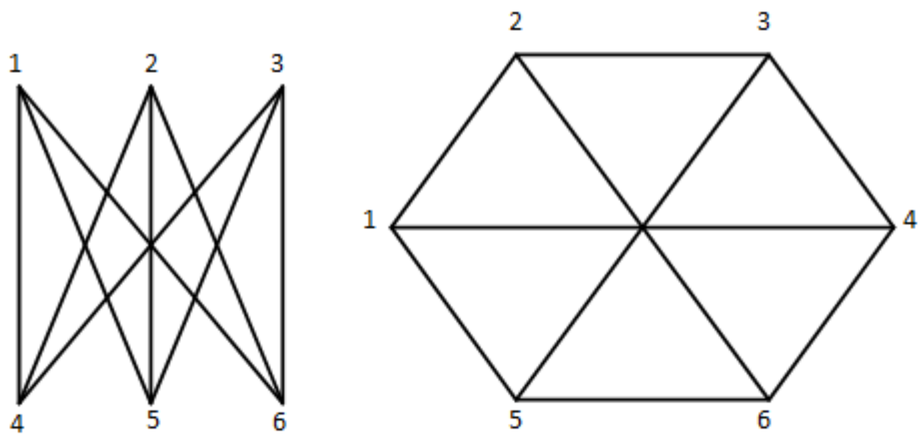

Рисунок 1 Два изоморфных графа

Граф называется ориентированным, если каждое его ребро ориентировано: $\forall x \neq y \in X \forall u \in U\ \Phi(x, u, y) \Longrightarrow \overline{\Phi(y, u, x)}$. Часто бывает полезным преобразовывать неориентированный граф в ориентированный с помощью замены каждого ориентированного ребра, парой неориентированных ребёр с противоположной ориентацией.

Подграфом графа $\Gamma = (X, U, \Phi)$ называется такой граф $\Gamma' = (X', U', \Phi')$, что $X' \subseteq X, U' \subseteq U$. Обозначают $\Gamma' \subseteq \Gamma$.

Граф называется псевдографом, если в нём допускаются петли и кратные рёбра, т.е. две вершины могут быть соединены более чем одним ребром. Псевдограф без петель называется мультиграфом.

Неориентированный граф называется простым, если он не имеет петель и любая пара вершин соединена не более чем одним ребром.

Простой граф называется полным, если каждая пара вершин соединена ребром. Такой граф с n вершинами содержит C_n^2 рёбер.

Дополнением простого графа Γ называется граф $\overline{\Gamma}$, имеющий те же вершины, а его рёбра являются дополнением Γ до полного графа.

Граф называется плоским (планарным), если он может быть изображён на плоскости так, что пересечения рёбер являются его вершинами.

Если вершины x и y соединены ребром u, то говорят, что вершины x, y - смежные, а ребро u инцидентно вершинам x и y. Два ребра называют смежными, если они имеют общую вершину.

Степенью вершины графа называется количество рёбер, инцидентных данной вершине. Вершина графа, имеющая степень 0, называется изолированной, а если её степень равна 1, то такая вершина называется висячей.

Граф называется помеченным, если его вершины отличаются друг от друга какими-либо пометками.

Маршрут на графе $\Gamma = (X, U, \Phi)$ определяется последовательностью вершин и рёбер $x_1, u_1, x_2, u_2, \dots, x_n, u_n, x_{n+1}$, где $x_i \in X, u_i \in U$. Ребро u_i соединяет вершину x_i с вершиной x_{i+1}, т.е. выполняется отношение инцидентности $\Phi(x_i, u_i, x_{i+1})$.

- Маршрут называется цепью, если все его рёбра различные.
- Маршрут называется замкнутым, если $x_1 = x_{n+1}$
- Замкнутая цепь называется циклом
- Цепь называется простой, если не содержит одинаковых вершин
- Простая замкнутая цепь называется простым циклом
- Гамильтоновой цепью называется простая цепь, содержащая все вершины графа
- Гамильтоновым циклом называется простой цикл, содержащий все вершины графа

Граф $\Gamma = (X, U, \Phi)$ называется связным, если для всех $x, y \in X$ существует путь из вершины x в вершину y. Связанный ориентированный граф называется сильно связанным. Орграф называется слабо связным, если соответствующий ему неориентированный граф связный.

Связанный неориентированный ациклический граф называется деревом. Множество деревьев называется лесом.

Большинство задач на графах касается определения компонент связности, поиска маршрутов, расстояний и т.п. В реальных задачах, графу могут быть очень большими, поэтому конечной целью рассмотрения каждой из задач будет являться описание и реализация алгоритма решения данной задачи на ЭВМ.

Способы представления графов

Наиболее известный и популярный способ представления графов состоит в геометрическом изображении точек (вершин) и линий (рёбер) на бумаге. При численном решении задач на вычислительных машинах граф должен быть представлен дискретным способом. Существует довольно много способов такого рода представления графов. Однако простота использования представления графа, как и эффективность алгоритма, в основе которого он лежит, в полной мере зависит от конкретного выбора этого представления. Одно из направлений теории графов связано с их матричным представлением. Существуют различные виды матриц, ассоциированные с графами. Эти алгебраические формы используются для решения многих задач теории графов. Ниже рассматриваются две такие матричные формы и несколько нестандартных представлений, которые наиболее широко используются в алгоритмах на графах.

Матрица смежности

Матрицей смежности ориентированного помеченного графа с n вершинами называется матрица $A = [a_{ij}], i, j = 1, \dots, n$, в которой

$$a_{ij} = \begin{cases} 1, \text{если существует ребро } (x_i, x_j) \\ 0, \text{если вершины } x_i, x_j \text{ не связаны ребром} \end{cases}$$

Матрица смежности однозначно определяет структуру графа. Отметим, что петля в матрице смежности может быть представлена соответствующим единичным диагональным элементом. Кратные рёбра можно представить, позволив элементу матрицы быть больше 1, но это не принято. Также в целях экономии памяти, каждый элемент матрицы часто представляют в виде одного двоичного разряда.

Матрица инцидентности графа

Матрицей инцидентности для неориентированного графа с n вершинами и m рёбрами называется матрица $B = [b_{i,j}], i = 1, \dots, n, j = 1, \dots, m$, строки которой соответствуют вершинам, а столбцы рёбрам.

$$b_{ij} = \begin{cases} 1, \text{если вершина } x_i \text{ инцидентна ребру } u_j \\ 0, \text{если вершина } x_i \text{ не инцидентна ребру } u_j \end{cases}$$

Матрицей инцидентности для ориентированного графа с n вершинами и m рёбрами называется матрица $B = [b_{i,j}], i = 1, \dots, n, j = 1, \dots, m$, строки которой соответствуют вершинам, а столбцы рёбрам. Элементы

$$b_{ij} = \begin{cases} +1, \text{если ребро } u_j \text{ выходит и вершины } x_i \\ -1, \text{если ребро } u_j \text{ входит в вершину } x_i \\ 0, \text{если вершина } x_i \text{ не инцидентна ребру } u_j \end{cases}$$

Матрица весов графа

Граф называется взвешенным, если каждому его ребру сопоставлено число. Простой взвешенный граф может быть представлен своей матрицей весов $W = [w_{ij}]$, где w_{ij} - вес ребра, соединяющего вершины $i, j = 1,2, \dots, n$. Веса несуществующих рёбер полагают равными ∞ или 0 в зависимости от конкретной задачи. Заметим, что матрица весов – обобщение матрицы смежности.

Список рёбер графа

При описании графа списком его рёбер каждое ребро представляется парой инцидентных ему вершин. Это представление можно реализовать двумя массивами $r = (r_1, r_2, \dots, r_m)$ и $t = t_1, t_2, \dots, t_m$, где m – количество рёбер в графе. Каждый элемент в массиве есть метка вершины, а -ое ребро графа выходит из вершины r_i и входит в вершину t_i. Интересно, что с помощью такого представления легко можно описать кратные рёбра и петли.

Структура смежности графа

Ориентированный или неориентированный граф может быть однозначно представлен структурой смежности своих вершин. Структура смежности состоит из списков $Adj[x]$ вершин графа, смежных с вершиной x. Списки $Adj[x]$ составляются для каждой вершины графа.

Структуры смежности могут быть удобно реализованы массивом из n линейно связанных списков. Каждый список содержит вершины, смежные с вершиной, для которой составляется список. Хранение же списков смежности на сцепленной памяти желательно в алгоритмах, в основе которых лежат операции добавления и удаления вершин из списков.

Метод поиска в глубину

Один из наиболее естественных способов систематического исследования всех вершин графа исходит из процедуры прохождения графа методом поиска с возвращением, который исследует граф в глубину. На неориентированном графе $\Gamma = (X, U, \Phi)$ поиск в глубину осуществляется следующим образом. Когда посещаем вершину $x \in X$, то далее идём по одному из рёбер (x, y), инцидентному вершине $y \in X$. Если вершина y уже пройдена, то возвращаемся в x и выбираем другое ребро. Если вершина y не пройдена, то заходим в неё и применяем процесс прохождения рекурсивно уже с вершиной y. Если все рёбра, инцидентные вершине x, просмотрены, то идём назад по ребру (s, x), по которому пришли в x, и продолжаем исследование рёбер, инцидентных вершине $s \in X$, по которому пришли в x, и продолжаем исследование рёбер, инцидентных вершине $s \in X$. Процесс заканчивается, когда попытаемся вернуться из вершины, с которой начали просмотр графа.

Поиск в глубину можно также осуществить и на ориентированном графе. Если граф ориентированный, то находясь в узле x, необходимо выбирать ребро (x, y), только выходящее из x. Исследовав все рёбра, выходящие из y, возвращаемся в x даже тогда, когда в y входят другие рёбра, ещё не рассмотренные. Данная техника просмотра в глубину полезна в практических приложениях при определении различных свойств как ориентированных, так и неориентированных графов.

Метод поиска глубину на простом неориентированном графе представлен в этом листинге:

Листинг 1

```
procedure Depth(x,w);
```

```
begin
  count:=count+1;
  Mark[x]:=count;
  for v ∈ Adj[x] do
  begin
    if Mark[v]=0 then
    begin
      T:= T ∪ {(x,y)};
      Depth(v,x);
    end
    else
    if Mark[v]<Mark[x] and (v ≠ w) then
      B:= B ∪ {(x,y)};
  end;
end;

for v ∈ X do
  Mark[v]:=0;
count:=0;
T:=∅; B:=∅;
for v ∈ X do
  if Mark[v] then
    Depth(v,0)
```

Рекурсивная процедура Depth(x, w) осуществляет поиск в глубину на графе $\Gamma = (X, U, \Phi)$, содержащем $x \in X$, и строит для графа дерево T поиска, которое является ориентированным остовным деревом $\Gamma_0 = (X, T, \Phi)$, $w \in X$ является отцом $x \in X$ в строящемся дереве, где x - исследуемая величина. Граф задан структурой смежности $Adj[x]$, где $Adj[x]$ означает множество вершин, смежных с $x \in X$. Элементы T - это рёбра строящегося дерева поиска, а элементы B - это обратные рёбра, которые не могут принадлежать Γ_0, так как они ведут назад в пройденные ранее вершины. Заметим, что обратное ребро должно идти от потомка к предку по дереву поиска. Чтобы отличить уже пройденные вершины от непройденных, вводится вектор Mark$[x]$ меток вершин, которые постепенно нумеруются от 1 до $|X|$ по мере того, как попадаем в них. Сначала полагается Mark$[x]$=0 для всех $x \in X$ в знак того, что ни одна вершина не пройдена, и когда попадаем в вершину x в первый раз, Mark$[x]$ получает ненулевое значение. Ребро $(x, v) \in T$, если метка вершины Mark$[v]$=0. Если же Mark$[v]$≠0, то условием того, что $(x, v) \in B$ будет обратным ребром, являются соотношения Mark$[v]$<Mark$[x]$ и $(v \neq w)$. Условие Mark$[v]$<Mark$[x]$ означает, что вершина v была пройдена раньше вершины x. Поэтому ребро $(x, v) \in B$ будет обратным, если оно не является ребром дерева T, пройденным от отца w к x, т.е. $v \neq w$.

Сложность поиска в глубину

Поскольку для каждой вершины, которую проходим впервые, выполняется обращение к процедуре Depth ровно один раз, то есть всего обращений будет $|X|$. При каждом обращении количество производимых действий пропорционально числу рёбер, инцидентных рассматриваемой вершине, поэтому сложность поиска составляет $O(|X| + |U|)$.