Тема 6. Алгоритмы на графах

6.9. Кратчайшие пути

Пусть $G = (V, E)$ – связный взвешенный простой ориентированный граф с матрицей весов дуг $W = [w_{ij}]$, где w_{ij} – длина дуги (i, j). Если в графе отсутствует дуга (i, j), то $w_{ij} = \infty$. Длина ориентированного пути есть сумма длин дуг, входящих в путь. Путь $P(s, f)$ между вершинами s и f ($s, f \in V$), имеющий минимальную длину, называется *кратчайшим путем*. Длина $dist(s, f)$ кратчайшего пути называется *расстоянием* от s до f. Очевидно, что $dist(i, i) = 0$ для любого $i \in V$.

Многие графы можно свести к рассмотренному типу орграфа. Если орграф не является простым, то можно отбросить все петли и заменить каждое множество параллельных дуг дугой с наименьшей длиной из этого множества. Если граф неориентированный, то можно рассматривать граф, который получается заменой каждого неориентированного ребра (i, j) парой дуг (i, j) и (j, i) с весом, равным весу исходного неориентированного ребра. Если граф не взвешенный, то можно считать, что все дуги имеют один и тот же вес.

6.9.1. Кратчайшие пути от фиксированной вершины

Большинство алгоритмов нахождения кратчайших путей от фиксированной вершины s основаны на следующем. Вычисляются некоторые верхние ограничения $D(v)$ на расстояния от s до всех вершин $v \in V$. На каждом этапе, когда устанавливается, что для некоторой вершины $u \in V$ $D(u) + w_{u,v} < D(v)$, оценка $D(v)$ улучшается и принимает новое значение $D(u) + w_{u,v}$. Процесс завершается, когда дальнейшее улучшение ни одного из ограничений невозможно. Тогда значение каждого элемента $D(v) = dist(s, v)$, т. е. расстоянию от s до v. При поиске кратчайшего пути $P(s, f)$ между двумя фиксированными вершинами s и f процесс завершается, если дальнейшее улучшение элемента $D(f)$ невозможно. Тогда расстояние от s до f есть $dist(s, f) = D(f)$.

Для случая, когда веса всех дуг неотрицательны, эффективным алгоритмом нахождения кратчайших путей от фиксированной вершины s до всех остальных вершин в графе является *алгоритм Дейкстры*. Он основан на просмотре графа в ширину, в процессе которого производится специальная расстановка меток вершин. Метки вершин $v \in V$ есть значения длин путей от фиксированной вершины s до v. Сначала вершине v присваивается временная метка, значение которой есть длина пути от s до v. В этот момент еще не все возможные пути от s до v рассмотрены. В процессе исследования орграфа значение метки модифицируется до тех пор, пока не будут исследованы все возможные пути от s до v. После этого метка вершины v не изменяется и принимает окончательное значение, равное длине кратчайшего пути от s до v. Процесс завершается, когда все вершины будут иметь окончательные метки. Если требуется определить кратчайший путь только до некоторой фиксированной вершины f, то процесс можно завершить после получения окончательной метки для f.

Алгоритм Дейкстры поиска кратчайших путей от фиксированной вершины s до всех остальных вершин орграфа $G = (V, E)$ с матрицей весов $W = [w_{ij}]$ представлен алгоритмом 6.18. Для записи значений меток каждой вершине $v \in V$ сопоставлен элемент $label(v)$. Множество T есть множество вершин графа, еще не имеющих окончательных меток. Значением элемента *last* является вершина, которая последней на данном шаге получила окончательную метку.

for $v \in V$ **do** $label(v) \leftarrow \infty$
$label(s) \leftarrow 0$
$last \leftarrow s$
$T \leftarrow V - \{s\}$
while $T \neq \varnothing$ **do**
{
  for $v \in T$ **do**
    if $label(v) > label(last) + w_{last,v}$
      then { $label(v) \leftarrow label(last) + w_{last,v}$; $pred(v) \leftarrow last$ }
  $u \leftarrow$ любая вершина с $label(u) = \min\{label(k) : k \in T\}$
  $T \leftarrow T - \{u\}$
  $last \leftarrow u$
}
// определить кратчайшие пути $P(s, v)$ для всех $v \in V - \{s\}$
for $v \in V - \{s\}$ **do**
  if $label(v) \neq \infty$
    then { // существует путь от s до v ; $P(s, v) \leftarrow (s, \ldots, pred(pred(v)), pred(v), v)$ }
    else // не существует пути от s до v

Алгоритм 6.18. Алгоритм Дейкстры поиска кратчайших путей

Вначале вершине s присваивается окончательная метка $label(s) = 0$ (нулевое расстояние до самой себя), а остальным $|V| - 1$ вершинам – временная метка ∞. На каждом шаге метки модифицируются следующим образом. Каждой вершине $v \in T$ (не имеющей окончательной метки) присваивается новая временная метка

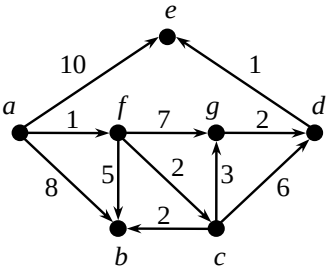
$$label(v) = \min(label(v), label(last) + w_{last,v}),$$

где $last$ – вершина, которая получила окончательную метку на предыдущем шаге. Затем находится вершина u с наименьшей из всех временных меток, которая становится окончательной для этой вершины. Таким образом, на каждом шаге одной вершине с временной меткой присваивается окончательная метка. Процесс продолжается до тех пор, пока все вершины графа не получат окончательные метки. Тогда окончательная метка каждой вершины $v \in V$ равна длине кратчайшего пути (расстоянию) от s до v, т. е. $label(v) = dist(s, v)$. Если $label(v) = \infty$, то это означает, что не существует ориентированного пути от s до v.

Для определения самого пути $P(s, v)$ от s до v каждой вершине $v \in V$ сопоставлен указатель $pred(v)$ – вершина, предшествующая вершине v в кратчайшем пути. Тогда последовательность вершин кратчайшего пути от s до v есть вершины

$$P(s, v) = (s, \ldots, pred(pred(pred(v))), pred(pred(v)), (pred(v), v).$$

Работа алгоритма Дейкстры проиллюстрирована на рис. 6.10 (ищутся кратчайшие пути от вершины a).



а

Шаг	*a*	*b*	*c*	*d*	*e*	*f*	*g*
0	[0]	∞	∞	∞	∞	∞	∞
1	[0]	8	∞	∞	10	[1]	∞
2	[0]	6	[3]	∞	10	[1]	8
3	[0]	[5]	[3]	9	10	[1]	6
4	[0]	[5]	[3]	9	10	[1]	[6]
5	[0]	[5]	[3]	[8]	10	[1]	[6]
6	[0]	[5]	[3]	[8]	[9]	[1]	[6]

б

Шаг	*a*	*b*	*c*	*d*	*e*	*f*	*g*
0	–	–	–	–	–	–	–
1	–	*a*	–	–	*a*	*a*	–
2	–	*f*	*f*	–	*a*	*a*	–
3	–	*c*	*f*	*c*	*a*	*a*	*c*
4	–	*c*	*f*	*c*	*a*	*a*	*c*
5	–	*c*	*f*	*g*	*a*	*a*	*c*
6	–	*c*	*f*	*g*	*d*	*a*	*c*

в

Путь	*dist* (*v*)	Вершины пути
P(*a*, *b*)	5	*a*, *f*, *c*, *b*
P(*a*, *c*)	3	*a*, *f*, *c*
P(*a*, *d*)	8	*a*, *f*, *c*, *g*, *d*
P(*a*, *e*)	9	*a*, *f*, *c*, *g*, *d*, *e*
P(*a*, *f*)	1	*a*, *f*
P(*a*, *g*)	6	*a*, *f*, *c*, *g*

г

Рис. 6.10. Процесс работы алгоритма Дейкстры:
а – взвешенный орграф; *б* – процесс присвоения меток *label* (*v*)
(окончательные метки заключены в квадратные скобки);
в – процесс формирования указателей *pred* (*v*);
г – кратчайшие пути от вершины *a* до всех остальных вершин орграфа

Если необходимо найти кратчайший путь от s только до некоторой фиксированной вершины f, то в алгоритме достаточно заменить условие $T \neq \varnothing$ цикла **while** на **while** $f \in T$ и определить единственный путь $P(s, f)$.

Определим вычислительную сложность алгоритма. Цикл **while** выполняется $|V| - 1$ раз (если ищется путь между фиксированной парой вершин s и f – не более $|V| - 1$). Число операций при каждом выполнении цикла, очевидно, равно $O(|V|)$. Поэтому алгоритму требуется $O(|V|^2)$ операций.

Следует обратить внимание на то, что алгоритм Дейкстры не работает, если некоторые из весов дуг отрицательные. Это связано с тем, что после присвоения вершине окончательной метки алгоритм не может ее изменить, а наличие отрицательных весов может этого потребовать.

6.9.2. Кратчайшие пути между всеми парами вершин

Для нахождения кратчайших путей между всеми $|V|\,(|V|-1)$ упорядоченными парами вершин в орграфе с $|V|$ вершинами можно использовать $|V|$-кратное применение алгоритма Дейкстры, т. е. применить его к каждой вершине графа. Такой подход, очевидно, потребует $O(|V|^3)$ операций для графов с неотрицательными весами дуг. Однако существуют более эффективные алгоритмы, которые применимы даже тогда, когда в графе присутствуют отрицательные веса дуг (но не существует ориентированных циклов отрицательной длины – в этом случае задача становится бессмысленной). Одним из таких алгоритмов является *алгоритм Флойда* (Floyd).

Процесс вычисления кратчайших путей алгоритмом Флойда основан на алгоритме Уоршелла для вычисления транзитивного замыкания (см. алгоритм 6.14). Пусть $W = [w_{ij}]$ – матрица весов орграфа $G = (V, E)$ и пусть вершины графа обозначаются числами $1, 2, \ldots, |V|$. Необходимо вычислить матрицу $W^* = [w^*_{ij}]$, в которой w^*_{ij} – длина кратчайшего пути от $i \in V$ до $j \in V$. Для этого определяется последовательность матриц $W^0 = [w^0_{ij}]$, $W^1 = [w^1_{ij}]$, …, $W^{|V|} = [w^{|V|}_{ij}] = W^*$ следующим образом:

$$w^0_{ij} = w_{ij},$$
$$w^l_{ij} = \min(w^{l-1}_{ij}, w^{l-1}_{il} + w^{l-1}_{lj}),$$

т. е. матрица $W^l = [w^l_{ij}]$ $(l \geq 1)$ получается из матрицы W^{l-1} после обработки вершины l в графе, соответствующем матрице W^{l-1}. Очевидно, что значение w^l_{ij} есть длина кратчайшего пути от $i \in V$ до $j \in V$ с промежуточными вершинами, принадлежащими только множеству $\{1, 2, \ldots, l\} \subseteq V$.

Для порождения самих $|V|(|V|-1)$ кратчайших путей можно построить матрицу $P = [p_{ij}]$, в которой p_{ij} – вершина, непосредственно следующая за вершиной i в кратчайшем пути от $i \in V$ до $j \in V$. Тогда кратчайший путь от i до j представляет собой последовательность вершин $v_1, v_2, \ldots, v_t$, где $v_1 = i$, $v_t = j$, $v_k = p_{k-1,j}$ для $1 < k \le t$. Матрица P легко строится по мере вычисления матриц $W^0, W^1, \ldots, W^{|V|}$. Сначала

$$p_{ij} = \begin{cases} j, \text{ если } w_{ij} \neq \infty, \\ 0, \text{ если } w_{ij} = \infty. \end{cases}$$

Если на шаге с номером l выполняется условие $w_{il} + w_{lj} < w_{ij}$, то $p_{ij} = p_{il}$, в противном случае – сохраняется предыдущее значение. Очевидно, что в конце работы алгоритма $p_{ij} = 0$ в том случае, когда $w^*_{ij} = \infty$, т. е. когда не существует пути от i до j.

Реализация метода Флойда представлена алгоритмом 6.19. Использование матрицы P для выписывания самих кратчайших путей как последовательности вершин в алгоритме не отражено. Очевидно, что сложность этого алгоритма есть $O(|V|^3)$.

$$
\begin{array}{l}
\textbf{for } i \leftarrow 1 \textbf{ to } |V| \textbf{ do } w_{ii} \leftarrow 0 \\
\textbf{for } i \leftarrow 1 \textbf{ to } |V| \textbf{ do} \\
\quad \textbf{for } j \leftarrow 1 \textbf{ to } |V| \textbf{ do if } w_{ij} \neq \infty \\
\qquad\qquad\qquad\qquad\qquad\qquad \textbf{then } p_{ij} \leftarrow j \\
\qquad\qquad\qquad\qquad\qquad\qquad \textbf{else } p_{ij} \leftarrow 0 \\
\textbf{for } l \leftarrow 1 \textbf{ to } |V| \textbf{ do} \\
\quad \textbf{for } i \leftarrow 1 \textbf{ to } |V| \textbf{ do} \\
\qquad \textbf{if } w_{il} \neq \infty \\
\qquad\quad \textbf{then for } j \leftarrow 1 \textbf{ to } |V| \textbf{ do} \\
\qquad\qquad\qquad\quad \textbf{if } w_{il} + w_{lj} < w_{ij} \\
\qquad\qquad\qquad\qquad\quad \textbf{then} \begin{cases} p_{ij} \leftarrow p_{il} \\ w_{ij} \leftarrow w_{il} + w_{lj} \end{cases}
\end{array}
$$

Алгоритм 6.19. Алгоритм Флойда поиска кратчайших путей

Следует обратить внимание на то, что рассмотренный алгоритм является наиболее эффективным из всех методов вычисления единственного кратчайшего пути между фиксированной парой вершин орграфа с отрицательными весами (при отсутствии циклов отрицательной длины). Другими словами, для орграфов с отрицательными весами дуг без циклов отрицательной длины не известен ни один алгоритм нахождения расстояния между одной фиксированной парой вершин, который был бы значительно эффективнее алгоритма нахождения расстояний между всеми парами вершин.