Алгоритм Форда-Беллмана

Задача:

Для заданного взвешенного графа $G = (V, E)$ найти кратчайшие пути из заданной вершины s до всех остальных вершин. В случае, когда в графе G содержатся отрицательные циклы, достижимые из s, сообщить, что кратчайших путей не существует.

Содержание

- 1 Введение
- 2 Псевдокод
- 3 Корректность
- 4 Реализация алгоритма и ее корректность
- 5 Сложность
- 6 Нахождение отрицательного цикла
- 7 Источники информации

Введение

Количество путей длины k рёбер можно найти с помощью метода динамического программирования. Пусть $d[k][u]$ — количество путей длины k рёбер, заканчивающихся в вершине u. Тогда $d[k][u] = \sum\limits_{v:vu\ \in E} d[k-1][v]$.

Аналогично посчитаем пути кратчайшей длины. Пусть s — стартовая вершина. Тогда $d[k][u] = \min\limits_{v:vu\ \in E} (d[k-1][v]\ +\ \omega(u, v))$, при этом $d[0][s] = 0$, а $d[0][u] = +\infty$

Лемма:

Если существует кратчайший путь от s до t, то $\rho(s,\ t)\ =\ \min\limits_{k=0..n-1} d[k][t]$

Доказательство:

$\triangleright$

Пусть кратчайший путь состоит из k ребер, тогда корректность формулы следует из динамики, приведенной ниже.

$\triangleleft$

Псевдокод

Используя приведенные формулы, алгоритм можно реализовать методом динамического программирования.

```
for k = 1 to |V| − 1                                        // вершины нумеруются с единицы
   for v ∈ V
      for (u, v) ∈ E
         d[k + 1][u] = min(d[k + 1][u], d[k][v] + ω(u, v))      // ω(u, v) — вес ребра uv
```

Также релаксацию можно свести к одномерному случаю, если не хранить длину пути в рёбрах. Одномерный массив будем обозначать d', тогда $d'[u] = \min(d'[u],\ d'[v] + \omega(vu))$

Корректность

Лемма:
Пусть $G = (V, E)$ — взвешенный ориентированный граф, s — стартовая вершина. Тогда после завершения k итераций цикла for выполняется неравенство $\rho(s, u) \leqslant d'[u] \leqslant \min\limits_{i=0..k} d[i][u]$.

Доказательство:
▷

Воспользуемся индукцией по k:

База индукции

При $k = 0$ выполнено: $\rho(s, u) \leqslant +\infty \leqslant +\infty$

Индукционный переход

Сначала докажем, что $\rho(s, u) \leqslant d'[u]$.
Пусть после $k - 1$ итерации выполняется $\rho(s, u) \leqslant d'[u] \leqslant \min\limits_{i=0..k-1} d[i][u]$ для всех u.

Тогда после k итераций
$\rho(s, v) = \min\limits_{u \in V}(\rho(s, u) + \omega(uv)) \leqslant \min\limits_{u \in V}(d'[u] + \omega(uv)) = d'[v]$.

Переходим ко второму неравенству.
Теперь возможно два случая:

1. $\min\limits_{i=0..k+1} d[i][u] = d[k + 1][u]$
2. $\min\limits_{i=0..k+1} d[i][u] = d[j][u] = \min\limits_{i=0..j} d[i][u]$

Рассмотрим 1 случай:

$$\min_{i=0..k+1} d[i][u] = d[k+1][u]$$
$$d'[u] \leqslant d'[v] + \omega(vu) \leqslant d[k][v] + \omega(vu) = d[k+1][u]$$

2 случай расписывается аналогично.

Таким образом переход выполнен и $\rho(s,u) \leqslant d'[u] \leqslant \min\limits_{i=0..k} d[i][u]$ выполняется.

◁

Реализация алгоритма и ее корректность

```
bool fordBellman(s):
    for v ∈ V
        d[v] = 1
    d[s] = 0
    for i = 0 to |V| − 1
        for (u, v) ∈ E
            if d[v] > d[u] + ω(u, v)      // ω(u, v) — вес ребра uv
                d[v] = d[u] + ω(u, v)
    for (u, v) ∈ E
        if d[v] > d[u] + ω(u, v)
            return false
    return true
```

В этом алгоритме используется релаксация, в результате которой $d[v]$ уменьшается до тех пор, пока не станет равным $\delta(s,v)$. $d[v]$ — оценка веса кратчайшего пути из вершины s в каждую вершину $v \in V$.
$\delta(s,v)$ — фактический вес кратчайшего пути из s в вершину v.

Лемма:
Пусть $G = (V, E)$ — взвешенный ориентированный граф, s — стартовая вершина. Тогда после завершения $|V| - 1$ итераций цикла для всех вершин, достижимых из s, выполняется равенство $d[v] = \delta(s,v)$.
Доказательство:
▷

Рассмотрим произвольную вершину v, достижимую из s. Пусть $p = \langle v_0, \ldots, v_k \rangle$, где $v_0 = s$, $v_k = v$ — кратчайший ациклический путь из s в v. Путь p содержит не более $|V| - 1$ ребер. Поэтому $k \leqslant |V| - 1$.

Докажем следующее утверждение:

После $n : (n \leqslant k)$ итераций первого цикла алгоритма, $d[v_n] = \delta(s, v_n)$

Воспользуемся индукцией по n:

База индукции

Перед первой итерацией утверждение очевидно выполнено: $d[v_0] = d[s] = \delta(s, s) = 0$

Индукционный переход

Пусть после $n : (n < k)$ итераций, верно что $d[v_n] = \delta(s, v_n)$. Так как (v_n, v_{n+1}) принадлежит кратчайшему пути от s до v, то $\delta(s, v_{n+1}) = \delta(s, v_n) + \omega(v_n, v_{n+1})$. Во время $l + 1$ итерации релаксируется ребро (v_n, v_{n+1}), следовательно по завершению итерации будет выполнено

$$d[v_{n+1}] \leqslant d[v_n] + \omega(v_n, v_{n+1}) = \delta(s, v_n) + \omega(v_n, v_{n+1}) = \delta(s, v_{n+1}).$$

Ясно, что $d[v_{n+1}] \geqslant \delta(s, v_{n+1})$, поэтому верно что после $l + 1$ итерации $d[v_{n+1}] = \delta(s, v_{n+1})$.
Индукционный переход доказан.

Итак, выполнены равенства $d[v] = d[v_k] = \delta(s, v_k) = \delta(s, v)$.
◁

Теорема:

> Пусть $G = (V, E)$ — взвешенный ориентированный граф, s — стартовая вершина. Если граф G не содержит отрицательных циклов, достижимых из вершины s, то алгоритм возвращает $true$ и для всех $v \in V$ $d[v] = \delta(s, v)$. Если граф G содержит отрицательные циклы, достижимые из вершины s, то алгоритм возвращает $false$.

Доказательство:
▷

Пусть граф G не содержит отрицательных циклов, достижимых из вершины s.

Тогда если вершина v достижима из s, то по лемме $d[v] = \delta(s, v)$. Если вершина v не достижима из s, то $d[v] = \delta(s, v) = 1$ из несуществования пути.

Теперь докажем, что алгоритм вернет значение $true$.

После выполнения алгоритма верно, что для всех $(u, v) \in E,\ d[v] = \delta(s, v) \leqslant \delta(s, u) + \omega(u, v) = d[u] + \omega(u, v)$, значит ни одна из проверок не вернет значения $false$.

Пусть граф G содержит отрицательный цикл $c = v_0, \ldots, v_k$, где $v_0 = v_k$, достижимый из вершины s. Тогда $\sum\limits_{i=1}^{k} \omega(v_{i-1}, v_i) < 0$.

Предположим, что алгоритм возвращает $true$, тогда для $i = 1, \ldots, k$ выполняется $d[v_i] \leqslant d[v_{i-1}] + \omega(v_{i-1}, v_i)$.

Просуммируем эти неравенства по всему циклу: $\sum\limits_{i=1}^{k} d[v_i] \leqslant \sum\limits_{i=1}^{k} d[v_{i-1}] + \sum\limits_{i=1}^{k} \omega(v_{i-1}, v_i)$.

Из того, что $v_0 = v_k$ следует, что $\sum\limits_{i=1}^{k} d[v_i] = \sum\limits_{i=1}^{k} d[v_{i-1}]$.

Получили, что $\sum\limits_{i=1}^{k} \omega(v_{i-1}, v_i) \geqslant 0$, что противоречит отрицательности цикла c.

$\triangleleft$

Сложность

Инициализация занимает $\Theta(V)$ времени, каждый из $|V| - 1$ проходов требует $\Theta(E)$ времени, обход по всем ребрам для проверки наличия отрицательного цикла занимает $O(E)$ времени. Значит алгоритм Беллмана-Форда работает за $O(VE)$ времени.

Нахождение отрицательного цикла

Приведенная выше реализация позволяет определить наличие в графе цикла отрицательного веса. Чтобы найти сам цикл, достаточно хранить вершины, из которых производится релаксация.

Если после $|V| - 1$ итерации найдется вершина v, расстояние до которой можно уменьшить, то эта вершина либо лежит на каком-нибудь цикле отрицательного веса, либо достижима из него. Чтобы найти вершину, которая лежит на цикле, можно $|V| - 1$ раз пройти назад по предкам из вершины v. Так как наибольшая длина пути в графе из $|V|$ вершин равна $|V| - 1$, то полученная вершина u будет гарантированно лежать на отрицательном цикле.

Зная, что вершина u лежит на цикле отрицательного веса, можно восстанавливать путь по сохраненным вершинам до тех пор, пока не встретится та же вершина u. Это обязательно произойдет, так как в цикле отрицательного веса релаксации происходят по кругу.

```
int[] negativeCycle(s):
    for v ∈ V
        d[v] = ∞
        p[v] = -1
    d[s] = 0
    for i = 0 to |V| − 1
        for (u, v) ∈ E
            if d[v] > d[u] + ω(u, v)
                d[v] = d[u] + ω(u, v)
                p[v] = u
    for (u, v) ∈ E
        if d[v] > d[u] + ω(u, v)
            for i = 0 to |V| − 1
                v = p[v]
            u = v
            while u != p[v]
                ans.add(v)             // добавим вершину к ответу
                v = p[v]
```

```
            reverse(ans)
            break
    return ans
```

Источники информации

- Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн Алгоритмы: построение и анализ — 2-е изд — М.: Издательский дом «Вильямс», 2009. — ISBN 978-5-8459-0857-5.
- MAXimal :: algo :: Алгоритм Форда-Беллмана (http://e-maxx.ru/algo/ford_bellman)

Источник — «http://neerc.ifmo.ru/wiki/index.php?title=Алгоритм_Форда-Беллмана&oldid=50473»

- Эта страница последний раз была отредактирована 27 декабря 2015 в 21:34.