Метод проталкивания предпотока

Материал из Викиконспекты

Метод проталкивая предпотока — обобщенный алгоритм нахождения максимального потока в сети. В отличии от алгоритма Эдмондса-Карпа и алгоритма Диница не является частным случаем метода Форда-Фалкерсона.

Содержание

Определения

Определение:

Предпотоком (**preflow**) будем называть функцию $f : V \times V \to \mathbb{R}$, удовлетворяющую следующим свойствам:

1) $f(u,v) = -f(v,u)$ (антисимметричность)

2) $f(u,v) \leqslant c(u,v)$ (ограничение пропускной способностью)

3) $\forall u \in V \setminus \{s,t\} \quad \sum\limits_{v \in V} f(v,u) \geqslant 0$ (ослабленное условие сохранения потока)

Как можно заметить, по своим свойствам предпоток очень похож на поток и отличается лишь тем, что для него не выполняется закон сохранения потока.

Определение:

Избыточным потоком (**excess flow**), входящим в вершину u, назовем величину $e(u) = \sum\limits_{v \in V} f(v,u)$.
Тогда вершина $u \in V \setminus \{s,t\}$ будет называться **переполненной** (**overflowing**), если $e(u) > 0$.

Определение:

Функция $h : V \to \mathbb{Z}_+$ называется **высотой вершины** (**vertex label**), если она удовлетворяет условиям:

1) $h(s) = |V|$

2) $h(t) = 0$

3) $\forall (u, v) \in E_f \quad h(u) \leqslant h(v) + 1$

Идея

Для понимания идеи алгоритма представим, что наша сеть — система из резервуаров, находящихся на определенной высоте, и соединяющих их труб с заданными пропускными способностями, соответствующих вершинам и ребрам в исходной сети. Сам алгоритм можно представить как процесс поочередного "переливания" жидкости (операция проталкивания) из одного резервуара в другие, находящиеся на меньшей высоте, до тех пор пока будет существовать резервуар, соответствующей переполненной вершине. Может случиться ситуация, что все трубы, выходящие из переполненной вершины u, ведут к вершинам, находящимся на такой же высоте что и u или выше ее. В таком случае поднимем резервуар (операция подъема), соответствующий данной вершине, таким образом, чтобы его высота стала на единицу больше, чем высота самого низкого из смежных резервуаров. После подъема будет существовать по крайней мере одна труба, по которой можно пропустить жидкость.

В итоге, у нас не останется ни одной переполненной вершины, та часть потока, которая могла пройти к стоку, окажется там, остальная же вернется в исток. В такой ситуации предпоток превращается в обычный поток, так как для каждой вершины выполняется условие сохранения потока. Как будет показано далее, предпоток становится не только обычным, но и максимальным потоком.

Операции

Как упоминалось ранее, в алгоритме выполняются две основные операции: проталкивание из переполненной вершины избытка потока в смежные вершины, высота которых меньше, чем у переполненной, и подъем вершины.

Проталкивание (push)

Операция проталкивания из вершины u в вершину v может применяться тогда, когда $e(u) > 0$, то есть вершина u является переполненной, $c_f(u, v) > 0$ и $h(u) = h(v) + 1$.

Данная операция работает следующим образом: по ребру (u, v) пропускается максимально возможный поток, то есть минимум из избытка вершины u и остаточной пропускной способности ребра (u, v), вследствие чего избыток вершины u, остаточная пропускная способность ребра (u, v) и поток по обратному ребру (v, u) уменьшаются на величину потока, а избыток вершины v, поток по ребру (u, v) и остаточная пропускная способность обратного ребра (v, u) увеличиваются на эту же величину.

```
push(u, v)
    d = min(e(u), c(u, v) - f(u, v));
```

```
    f(u, v) += d;
    f(v, u) = -f(u, v);
    e(u) -= d;
    e(v) += d;
```

По своему результату все проталкивания можно разделить на 2 группы. Будем называть проталкивание из вершины u в вершину v **насыщающим**, если после него остаточная пропускная способность ребра (u, v) стала равна нулю. Все остальные проталкивания будем называть **ненасыщающими**. Подобная классификация проталкиваний понадобится нам при оценке их количества.

Подъем (relabel)

Операция подъема применима для вершины u, если $e(u) > 0$ и $\forall (u, v) \in E_f \quad h(u) \leqslant h(v)$.

То есть, для переполненной вершины u применима операция подъема, если все вершины, для которых в остаточной сети есть ребра из u, расположены не ниже u. Следовательно, операцию проталкивания для вершины u произвести нельзя.

В результате подъема высота текущей вершины становится на единицу больше высоты самый низкой смежной вершины в остаточной сети, вследствие чего появляется как минимум одно ребро, по которому можно протолкнуть поток.

```
relabel(u)
    h(u) = min(h(v): f(u, v) - c(u, v) > 0) + 1;
```

Схема алгоритма

Для начала проинициализируем предпоток. Пропустим максимально возможный поток по ребрам, инцидентным стоку, увеличив избыточный поток для каждой смежной со стоком вершиной на соответствующую величину. Все остальные потока не несут, следовательно, для вершин не смежных с истоком избыточный поток изначально будет нулевым. Также для всех вершин, кроме, естественно, истока, установим высоту, равную нулю.

Более формально это можно записать так:

$$f(u,v) = \begin{cases} c(u,v), & u = s \\ -c(v,u), & v = s \\ 0, & u \neq s \text{ and } v \neq s \end{cases}$$

$$h(u) = \begin{cases} |V|, & u = s \\ 0, & u \neq s \end{cases}$$

```
initializePreflow(s)
    for ∀u ∈ V
        h(u) = 0;
        e(u) = 0;
    for ∀(u, v) ∈ E
        f(u, v) = 0;
        f(v, u) = 0;
```

```
    for ∀u : (s, u) ∈ E
        f(s, u) = c(s, u);
        f(u, s) = -c(s, u);
        e(u) = c(s, u);
        e(s) -= c(s, u);
    h(s) = |V|;
```

После инициализации будем выполнять операции проталкивания и подъема в произвольном порядке. Утверждается, что количество данных операций конечно, и после завершения работы алгоритма наш предпоток является максимальным потоком.

```
pushRelabelMaxFlow(s, t)
    initializePreflow(s);
    while существует применимая операция push или relabel
        выбрать операцию и выполнить ее
```

Корректность алгоритма

Для доказательства корректности алгоритма докажем несколько вспомогательных лемм.

Лемма (1):

Во время выполнения алгоритма $pushRelabelMaxFlow$ никакие свойства высоты не нарушаются.

Доказательство:

▷

Доказательство проведем по числу выполненных операций проталкивания и подъема.

Для начала покажем, что после выполнения операции $initializePreflow$ h является функцией высоты. Для всех ребер, не инцидентных s, высота обоих концов равна нулю, что удовлетворяет условиям. Единственными ребрами (u, v), для которых не выполняются условия, налагаемые на функцию высоты, то есть для которых $h(u) > h(v) + 1$, являются ребра, у которых $u = s$. Но после операции $initializePreflow$ ребра являются насыщенными и не принадлежат остаточной сети.

Это будет базой нашей индукции.

Теперь докажем переход: если перед выполнением операции проталкивания или подъема h является функцией высоты, то и после выполнения операции она останется функцией высоты. Для этого рассмотрим проталкивание и подъем отдельно.

Покажем, что после окончания операции проталкивания утверждение верно. После выполнения операции $push(u, v)$ в E_f может появиться ребро (v, u), если до операции по ребру (u, v) протекал нулевой поток, или же ребро (u, v) может насытиться, и тогда оно перестанет принадлежать остаточной сети. В первом случае $h(v) = h(u) - 1 < h(u) + 1$, и, следовательно, h остается функцией высоты. Во втором же случае при удалении ребра (u, v) из остаточной сети происходит лишь удаление соответствующих ограничений на высоты u и v, так что h остается функцией высоты.

Теперь рассмотрим операцию подъема вершины u. По определению данной операции гарантируется, что после ее выполнения для любого ребра $(u, v) \in E_f$ выполняется условие $h(u) \leqslant h(v) + 1$, то есть все выходящие ребра удовлетворяют условиям, накладываемым на функцию высоты. Рассмотрим какое-то входящее ребро (w, u). Так как до операции h являлась функцией высоты, то справедливо $h(w) \leqslant h(u) + 1$. После подъема высота вершины u увеличивается как минимум на единицу, следовательно, после выполнения данной операции $h(w) < h(u) + 1$. Таким образом, h и после выполнения операции $relabel(u)$ остается функций

высоты.

◁

Лемма (2):

Пусть f — предпоток в сети G. Тогда для любой переполненной вершины можно выполнить операцию проталкивания либо операцию подъема.

Доказательство:

▷

Рассмотрим вершину u, для которой $e(u) > 0$. Для любого ребра $(u, v) \in E_f$ справедливо $h(u) \leqslant h(v) + 1$ по свойствам функции высоты. Если к вершине u применима операция проталкивания, то лемма доказана. Иначе, для всех ребер $(u, v) \in E_f$ выполняется соотношение $h(u) < h(v) + 1$. Следовательно, $h(u) \leqslant h(v)$. Но тогда данная вершина удовлетворяет условиям операции подъема, следовательно, к вершине u можно применить эту операцию.

◁

Лемма (3):

Пусть G — сеть с истоком s и стоком t. f и h — предпоток и функция высоты соответственно. Тогда в остаточной сети G_f нет пути из s в t.

Доказательство:

▷

Докажем от противного.

Предположим, что в G_f существует путь из s в t. Из теоремы о существовании простого пути следует, что в G_f также существует и простой путь из s в t. Тогда пусть $p = \langle v_0, v_1, \ldots, v_k \rangle$ — простой путь, где $v_0 = s$ и $v_k = t$. А раз p простой, то $k < |V|$. Поскольку h — функция высоты, то $\forall i \in \{0, 1, \ldots, k-1\}$ справедливо $h(v_i) \leqslant h(v_{i+1}) + 1$. Следовательно, $h(s) \leqslant h(t) + k$. А так как по определению функции высоты $h(t) = 0$, то $h(s) \leqslant k < |V|$, что противоречит условию, что $h(s) = |V|$.

◁

Теорема:

> Если алгоритм $pushRelabelMaxFlow$ завершается, то вычисленный им предпоток f является максимальным потоком.

Доказательство:

▷

Для доказательства покажем, что перед каждой проверкой условия в цикле while f является предпотоком.

Перед началом цикла, после завершения операции $initializePreflow$, f является предпотоком.

Внутри цикла выполняются лишь операции проталкивания и подъема. Операция подъема не влияет на величины потока, а лишь изменяет высоту вершины, следовательно от операции подъема не зависит, будет ли f предпотоком. Операция $push(u, v)$ применяется, если $e(u) > 0$, увеличивая поток через ребро (u, v) на величину, не превышающую избыточный поток вершины u и остаточную пропускную способность ребра (u, v). Следовательно, если перед выполнением операции проталкивания f являлся предпотоком, то и после выполнения проталкивания f останется предпотоком.

После завершения алгоритма для каждой вершины $u \in V \setminus \{s, t\}$ справедливо, что $e(u) = 0$, что следует непосредственно из леммы (1), леммы (2) и того, что перед выполнением операций проталкивания или подъема f является предпотоком. Но тогда f удовлетворяет условию сохранения потока, то есть сам является потоком.

Поскольку из леммы (1) следует, что h является функцией высоты и после завершения алгоритма, то по лемме (3) в остаточной сети G_f нет пути из s в t. Но тогда по теореме Форда-Фалкерсона f — максимальный поток.

◁

Оценка быстродействия

Чтобы показать, что алгоритм завершает свою работу, найдем максимальное количество операций проталкивания и подъема. Для того докажем несколько вспомогательных лемм.

Лемма (4):

Пусть G — сеть со стоком s и истоком t, и f — предпоток в G. Тогда из любой переполненной вершины u существует путь в s в остаточной сети G_f.

Доказательство:

▷

Докажем лемму методом от противного.

Пусть для вершины u верно, что $e(u) > 0$. Пусть также $U = \{v : u \rightsquigarrow v \text{ в } G_f\}$ и $\bar{U} = V \setminus U$. Предположим, что $s \in \bar{U}$.

Покажем, что для любой пары вершин $v \in U$ и $w \in \bar{U}$ верно, что $f(w, v) \leqslant 0$. Если $f(w, v) > 0$, то $f(v, w) < 0$. Но тогда из этого утверждения следует, что $c_f(v, w) = c(v, w) - f(v, w) > 0$, что не может быть верным, так как в таком случае существует путь $u \rightsquigarrow v \to w$ в остаточной сети G_f, что противоречит выбору вершины w. Следовательно, для каждой пары вершин $v \in U$ и $w \in \bar{U}$ верно, что $f(w, v) \leqslant 0$. Тогда верно и $f(\bar{U}, U) \leqslant 0$. Следовательно
$e(U) = f(V, U) = f(U, U) + f(\bar{U}, U) = f(\bar{U}, U) \leqslant 0$.

Но по определению избыток потока неотрицателен, из чего следует, что для любой вершины $v \in U$ $e(v) = 0$, в том числе $e(u) = 0$, что противоречит условию переполненности вершины u.

◁

Лемма (5):

Пусть G — сеть с истоком s и стоком t. Тогда во время выполнения алгоритма $pushRealbelMaxFlow$ для любой вершины u в сети G верно, что $h(u) \leqslant 2 \cdot |V| - 1$

Доказательство:

▷

По определению функции высоты $h(s) = |V|$ и $h(t) = 0$, следовательно для истока и стока утверждение леммы выполнено.

Рассмотрим вершину u отличную от истока и стока. Изначально $h(u) = 0 \leqslant 2 \cdot |V| - 1$. Покажем, что после любой операции подъема $h(u) \leqslant 2 \cdot |V| - 1$. Для того, чтобы мы имели право произвести операцию $relabel(u)$, вершина u должна быть переполнена. Тогда по лемме (4) существует простой путь p из u в s в остаточной сети G_f. Рассмотрим этот путь. Пусть $p = \langle v_0, v_1, \ldots, v_k \rangle$, где $v_0 = u$ и $v_k = s$. Так как путь p простой, то $k \leqslant |V| - 1$. По определению функции высоты имеем, что $\forall i \in \{0, 1, \ldots, k-1\}\ h(v_i) \leqslant h(v_{i+1}) + 1$. Но тогда для вершин u и v верно, что
$h(u) \leqslant h(s) + |V| - 1 = 2 \cdot |V| - 1$

◁

На основании предыдущей леммы покажем верхнюю границу числа подъемов.

Лемма (6):

Пусть G — сеть с истоком s и стоком t. Тогда во время выполнения алгоритма $pushRealbelMaxFlow$ общее число подъемов не превышает $2 \cdot |V|^2$

Доказательство:

▷

Так как высоты истока и стока не изменяются в процессе работы алгоритма, то только $|V| - 2$ вершин могут быть подняты. Пусть $u \in V \setminus \{s, t\}$. Изначально $h(u) = 0$, и по лемме (5) известно, что $h(u) \leqslant 2 \cdot |V| - 1$. А так как при выполнении операции $relabel(u)$ высота вершины увеличивается как минимум на единицу, то максимальное количество подъемов вершины u также не превышает $2 \cdot |V| - 1$. Тогда суммарно число подъемов не превышает
$(|V| - 2) \cdot (2 \cdot |V| - 1) \leqslant 2 \cdot |V|^2$.

◁

Лемма (7):

После ненасыщающего проталкивания из u в v вершина u перестает быть переполненной.

Доказательство:

▷

Поскольку проталкивание ненасыщающее, то величина потока, проталкиваемого через ребро (u, v) должна быть равна $e(u)$. После выполнения проталкивания избыточный поток вершины должен уменьшится на величину проталкиваемого потока, следовательно, после ненасыщающего проталкивания из u в v величина $e(u) = 0$.

◁

Следующие две леммы показывают верхнюю границу количества проталкиваний.

Лемма (8):

Количество насыщающих проталкиваний при выполнение алгоритма $pushRealbelMaxFlow$ не превосходит $2 \cdot |V| \cdot |E|$

Доказательство:

▷

Возьмем произвольную пару вершин u и v и рассмотрим насыщающие проталкивания по ребрам (u, v) и (v, u). Предположим, что произошло насыщающее проталкивание по ребру (u, v). Тогда во время проталкивания $h(v) = h(u) - 1$. Для того, чтобы могло произойти еще одно насыщающее проталкивание по ребру (u, v), мы должны для начала протолкнуть поток из v в u, чтобы ребро (u, v) снова появилось в остаточной сети. Но для этого должно выполняться условие $h(v) = h(u) + 1$, то есть высота вершины u должна увеличится как минимум на 2. По лемме (5) высота вершины не превышает $2 \cdot |V| - 1$, следовательно, количество раз, когда высота вершины может увеличится на 2, меньше $|V|$. Поскольку между двумя насыщающими проталкиваниями высота одной из вершин должна увеличится по меньшей мере на 2, то между вершинами u и v их будет не более $2 \cdot |V|$. Тогда суммарно по всем ребрам во время работы алгоритма произойдут не более $2 \cdot |V| \cdot |E|$ насыщающих проталкиваний.

◁

Лемма (9):

Количество ненасыщающих проталкиваний при выполнение алгоритма $pushRealbelMaxFlow$ не превосходит $4 \cdot |V|^2 (|V| + |E|)$

Доказательство:

▷

Пусть $\Phi = \sum_{u:e(u)>0} h(u)$. Так как после завершения алгоритма ни одна из вершин не является переполненной, то и величина Φ после выполнения алгоритма должна равняться нулю.

Для начала рассмотрим, каким образом может увеличиваться величина Φ. Первое, что может увеличить Φ, это подъем, поскольку, осуществляя данную операцию, мы не изменяем избыточный поток ни у одной вершины, а лишь увеличиваем высоту одной из них. При каждой операции подъема Φ увеличивается менее чем на $2 \cdot |V|$, так как подъем не может увеличить высоту вершины больше, чем ее максимальная высота, которая согласно лемме (5) может быть не более $2 \cdot |V| - 1$. А поскольку из леммы (6) известно, что число подъемов не превышает $2 \cdot |V|^2$, то суммарно подъемы всех вершин могут увеличить Φ не более чем на $4 \cdot |V|^3$.

Во-вторых, величина Φ может увеличится при насыщающем проталкивании из u в v, потому что $e(u) > 0$ и после насыщающего проталкивания, а вершина v может стать переполненной. Увеличение меньше $2 \cdot |V|$, так как изменения высот не происходит, а высота вершины v не превосходит $2 \cdot |V| - 1$. Но по лемме (8) известно, что количество насыщающих проталкиваний за все время выполнения алгоритма не превосходит $2 \cdot |V| \cdot |E|$, следовательно, за счет насыщающих проталкиваний Φ увеличится не более чем на $4 \cdot |V|^2 \cdot |E|$.

Итого, получаем, что величина Φ не может быть больше $4 \cdot |V|^2 \cdot (|V| + |E|)$.

Теперь покажем, что ненасыщающее проталкивание уменьшает Φ как минимум на единицу. Пусть произошло ненасыщающее проталкивание из вершины u в v. Согласно лемме (7) после ненасыщающего проталкивания вершина u перестает быть переполненной, следовательно, Φ уменьшается на величину ее высоты. После проталкивания вершина v является переполненной, и поэтому Φ могла увеличится на $h(v)$. Поскольку $h(u) = h(v) - 1$, то при каждом ненасыщающем проталкивании Φ уменьшается по меньшей мере на единицу.

Зная верхнюю границу величины Φ, ее значение после выполнения алгоритма и то, что при каждом ненасыщающем проталкивании Φ уменьшается минимум на единицу, то можно сделать вывод, что количество ненасыщающих проталкиваний не больше чем $4 \cdot |V|^2 (|V| + |E|)$.

◁

Теорема:

> Пусть G — сеть. Тогда любая реализация метода $pushRelabelMaxFlow$ для G завершает свою работу, и число основных операций составляет $O(V^2E)$

Доказательство:

▷

Из леммы (6), леммы (8) и леммы (9) следует, что число операций подъема и проталкиваний не превышает $2 \cdot |V|^2 + 2 \cdot |V| \cdot |E| + 4 \cdot |V|^2 (|V| + |E|) = O(V^2E)$. А так как количество операций ограничено, то независимо от реализации алгоритм завершит свою работу.

◁

Источники

- Кормен Т., Лейзерсон Ч., Ривест Р. Алгоритмы: построение и анализ. — 2-е изд. — М.: Издательский дом «Вильямс», 2011. — С. 762—773.
- Алгоритм проталкивания предпотока — Википедия (http://ru.wikipedia.org /wiki/Алгоритм_проталкивания_предпотока)

Источник — «http://neerc.ifmo.ru/wiki/index.php?title=%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BF%D1%80%D0%BE%D1%82%D0%B0%D0%BB%D0%BA%D0%B8%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%BF%D1%80%D0%B5%D0%B4%D0%BF%D0%BE%D1%82%D0%BE%D0%BA%D0%B0&oldid=37522»

Категории: Алгоритмы и структуры данных | Задача о максимальном потоке

- Последнее изменение этой страницы: 15:22, 3 июня 2014.
- К этой странице обращались 1744 раза.