Потоки в сетях

Постановка задачи и основные понятия

Дан ориентированный граф (сеть) $G=(V,E)$, вершина s графа (источник) и вершина t (сток). Каждой дуге (i,j) приписана некоторая пропускная способность $c(i,j)\geq 0$ (без потери общности считаем её целочисленной величиной), определяющая максимальное значение потока, который может протекать по данной дуге. Потоком в сети называют целочисленную функцию $f(i,j)$, заданную на множестве дуг E и обладающей следующими свойствами:

- Для любой дуги $(i,j) \in E$ выполняется неравенство $f(i,j) \leq c(i,j)$.
- Для любой вершины $q \in V$, $q \neq s$ и $q \neq t$ выполняется равенство $\sum_{i \in V \text{ и } (i,q) \in E} f(i,q) = \sum_{j \in V \text{ и } (q,j) \in E} f(q,j)$. Другими словами, сумма потока, заходящего в q, равна сумме потока, выходящего из q (поток без потерь и накоплений).

Требуется определить значение максимального потока, который можно пропустить от источника s к стоку t, и его распределение по дугам.

Формальная запись. Определим $Div_f(q) = \sum_{i \in V \text{ и } (i,q) \in E} f(i,q) - \sum_{j \in V \text{ и } (q,j) \in E} f(q,j)$. Значение $Div_f(q) = 0$ для каждой вершины $q \in V \setminus \{s,t\}$. Величину $W(f) = Div_f(s)$ называют величиной потока f.

Теорема Л. Форда и Д. Фалкерсона. Величина каждого потока из s в t не превосходит пропускной способности минимального разреза, разделяющего s и t, причем существует поток, достигающий этого значения.

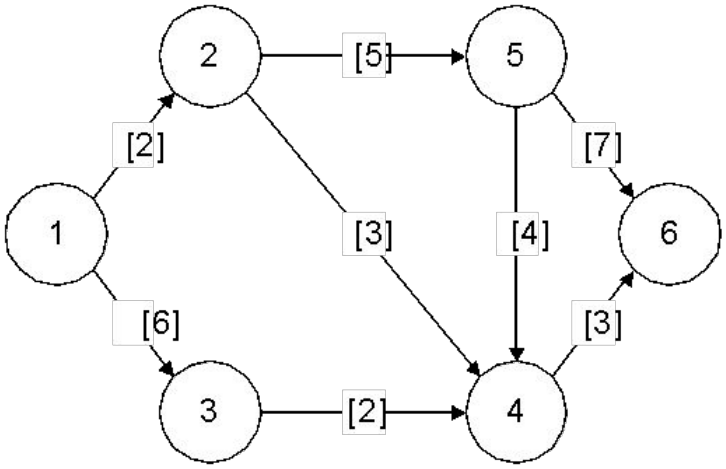
Напомним, что разрезом называют множество дуг, удаление которых из сети приводит к «разрыву» всех путей, ведущих из s в t. Пропускная способность разреза – это суммарная пропускная способность дуг, его составляющих. Более формально. Под разрезом $P(A)$ сети S, соответствующим подмножеству $A \subseteq V$ ($A \neq \varnothing$, $A \neq V$), понимается множество дуг $(u,v) \in E$, таких, что $u \in A$ и $v \in V \setminus A$. Таким образом, $P(A) = E \cap (A \times (V \setminus A))$. Для потока f в сети S поток через разрез $P(A)$ определяется как $f(A, V \setminus A) = \sum_{q \in P(A)} f(q)$. Пропускная способность разреза $P(A)$ есть $c(A, V \setminus A) = \sum_{q \in P(A)} c(q)$. Под минимальным разрезом, разделяющим s и t, понимается произвольный разрез $P(A)$, $s \in A$, $t \in V \setminus A$ с минимальной пропускной способностью. Теорема устанавливает эквивалентность задач нахождения максимального потока и минимального разреза, однако не определяет метода их поиска. Генерация всех подмножеств дуг и определение, является ли очередное подмножество разрезом – «лобовое решение», приводит к экспоненциальной сложности алгоритма.

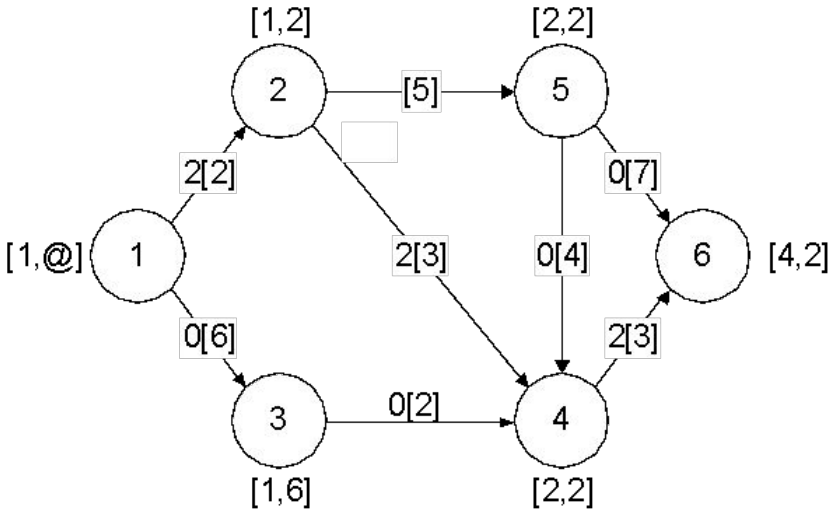
Вычитая из сети поток f, мы получаем *остаточную сеть*: (G_f, c_f, s, t). Множество вершин остаточной сети остается прежним, а пропускные способности изменяются по правилу $c_f(u, v) := c(u, v) - f(u, v)$. *Остаточная пропускная способность* ребра $c_f(u, v)$ показывает насколько может быть увеличен поток по дуге без превышения пропускной способности. *Остаточное ребро* – ребро с положительной остаточной пропускной способностью. Остаточные ребра образуют остаточную сеть. Множество ребер остаточной сети меняется следующим образом: ребра $(u,v) \in E$ такие, что $c_f(u, v) = 0$ уже не принадлежат E_f.

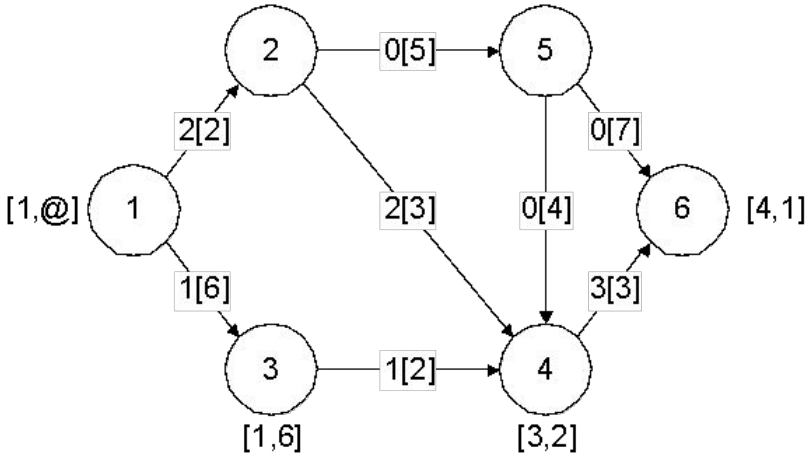
Другая формулировка теоремы *Л. Форда и Д. Фалкерсона.* Если в графе G_f существует путь $p: s \to t$ (обозначение пути, который начинается в s и заканчивается в t), то поток f не максимален. Или обратное утверждение – если в графе G_f не существует путь $p: s \to t$, то поток f максимален.

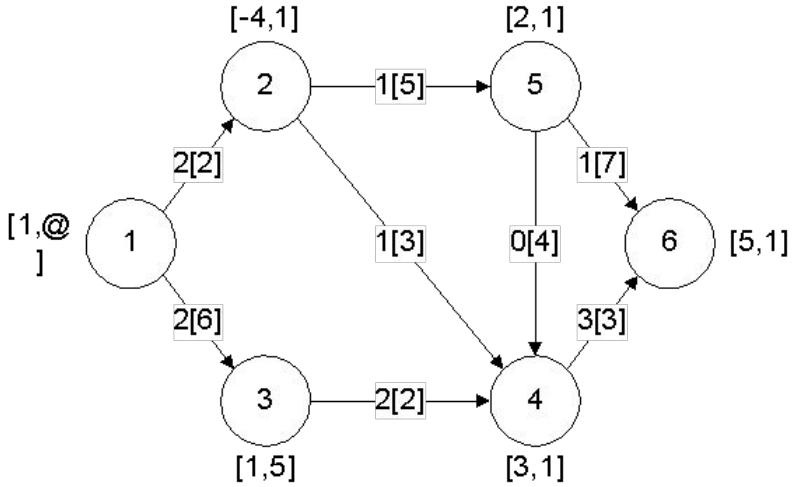
Алгоритм Эдмондса- Карпа

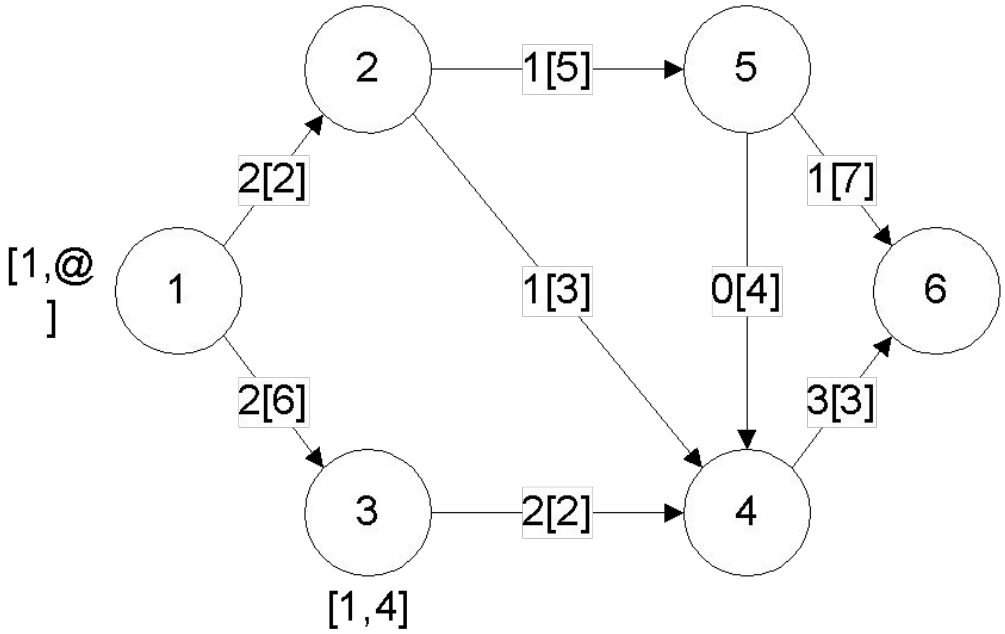
Метод решения задачи о максимальном потоке от s к t был предложен Л. Фордом и Д. Фалкерсоном, и их «техника меток» составляет основу других алгоритмов решения многочисленных задач, являющихся обобщениями или расширениями указанной задачи. «Техника меток» Л. Форда и Д. Фалкерсона заключается в последовательном (итерационном) построении максимального потока путем поиска на каждом шаге увеличивающейся цепи, то есть пути (последовательности дуг), поток по которой можно увеличить. При этом узлы (вершины графа) специальным образом помечаются. Отсюда и возник термин «метка». Формальная запись. Дуга $e=(u,v)$ сети S является допустимой дугой из u в v относительно потока f, если $e=(u,v)$ и $f(e)<c(e)$ (дуги первого типа) или $e=(v,u)$ и $f(e)>0$ (дуги второго типа). Второе условие говорит о том, что допустимыми являются и дуги, входящие в вершину u, по которым «уже пропущен ненулевой поток». Увеличивающаяся цепь – это последовательность попарно различных вершин и дуг v_0, e_1, v_1, e_2, v_2, …, v_{l-1}, e_l,v_l, такая, что $v_0=s$, $v_l=t$, и для каждого $i\le l$ дуга e_i является допустимой из v_{i-1} в v_i относительно потока f. Если найдена такая цепь, то значение потока может быть увеличено на $\delta = \min\{t(e_i) : 1 \le i \le l\}$, где $t(e_i) = \begin{cases} c(e_i) - f(e_i), \text{ если дуга } e_i \text{ первого типа} \\ f(e_i), \text{ если дуга } e_i \text{ второго типа} \end{cases}$. Величина потока увеличивается (уменьшается) на значение δ на дугах первого типа (на дугах второго типа).


1
2
3
4
5
6
[2]
[6]
[5]
[2]
[3]
[4]
[7]
[3]


[1,@]
1
2
3
4
5
6
[1,2]
[2,2]
[1,6]
[2,2]
[4,2]
2[2]
0[6]
[5]
2[3]
0[2]
0[7]
0[4]
2[3]


[1,@]
1
2
3
4
5
6
[1,6]
[3,2]
[4,1]
2[2]
1[6]
0[5]
2[3]
1[2]
0[7]
0[4]
3[3]


[1,@
]
1
2
3
4
5
6
[-4,1]
[1,5]
[2,1]
[3,1]
[5,1]
2[2]
2[6]
1[5]
1[3]
2[2]
1[7]
0[4]
3[3]


2
1[5]
5
2[2]
1[7]
[1,@
]
1
1[3]
0[4]
6
2[6]
3[3]
3
2[2]
4
[1,4]

Первая итерация. Присвоим вершине 1 метку [1,∞]. Первый шаг. Рассмотрим дуги, началом которых является вершина 1 – дуги (1,2) и (1,3). Вершины 2 и 3 не помечены, поэтому присваиваем им метки, для 2-й – [1,2] и 3-й – [1,6]. Что представляют из себя метки? Первая цифра – номер вершины, из которой идет поток, вторая цифра – численное значение потока, который можно передать по этой дуге. Второй шаг. Выберем помеченную, но непросмотренную вершину. Первой в соответствующей структуре данных записана вершина 2. Рассмотрим дуги, для которых она является началом – дуги (2,4) и (2,5). Вершины 4 и 5 не помечены. Присвоим им метки – [2,2] и [2,2]. Итак, на втором шаге вершина 2 просмотрена, вершины 3, 4, 5 помечены, но не просмотрены, остальные вершины не помечены. Третий шаг. Выбираем вершину 3. Рассмотрим дугу (3,4). Вершина 4 помечена. Переходим к следующей вершине – четвертой, соответствующая дуга – (4,6). Вершина 6 не помечена. Присваиваем ей метку [4,2]. Мы достигли вершины-стока, тем самым найдя путь (последовательность дуг), поток по которым можно увеличить. Информация об этом пути содержит метки вершин. В данном случае путь, или увеличивающаяся цепочка имеет вид: 1→2→4→6. Максимально возможный поток, который можно передать по дугам этого пути, определяется второй цифрой метки вершины стока, то есть 2. Поток в сети стал равным 2. Заметим, что построение цепочки не что иное, как модифицированный просмотр в ширины вершин графа.

Вторая итерация. Первый шаг. Присвоим вершине 1 метку [1,∞]. Рассмотрим дуги, началом которых является помеченная вершина 1. Это дуги (1,2) и (1,3). Вершина 2 не может быть помечена, так как пропускная способность дуги (1,2) исчерпана. Вершине 3 присваиваем метку [1,6]. Второй шаг. Выберем помеченную, но непросмотренную вершину. Это вершина 3. Повторяем действия. В результате вершина 4 получает метку [3,2]. Третий шаг. Выбираем вершину 4, только она помечена и не просмотрена. Вершине 6 присваиваем метку [4,1]. Почему только одна единица потока? На предыдущей итерации израсходованы две единицы пропускной способности данной дуги, осталась только одна. Вершина-сток достигнута. Найдена увеличивающая поток цепочка, это $1 \rightarrow 3 \rightarrow 4 \rightarrow 6$, по которой можно «протащить» единичный поток. Результирующий поток в сети равен 3.

Третья итерация. Вершине 1 присваиваем метку [1,∞]. Первый шаг. Результат – метка [1,5] у вершины 3. Второй шаг – метка [3,1] у вершины 4. Третий шаг. Пропускная способность дуги (4,6) израсходована полностью. Однако есть *обратная* дуга (2,4), по которой передается *поток, не равный нулю* (обратите внимание на текст, выделенный курсивом, – «изюминка» метода). Попробуем перераспределить поток. Нам необходимо передать из вершины 4 поток, равный единице (зафиксирован в метке вершины). Задержим единицу потока в вершине 2, то есть вернем единицу потока из вершины 4 в вершину 2. Эту особенность зафиксируем в метке вершины 2 знаком «минус» (технический прием, упрощающий реализацию) – [-4,1]. Тогда единицу потока из вершины 4 мы передадим по сети вместо той, которая задержана в вершине 2, а единицу потока из вершины 2 попытаемся «протолкнуть» по сети, используя другие дуги. Итак, вершина 4 просмотрена, вершина 2 помечена, вершины 5 и 6 не помечены. Четвертый и пятый шаги очевидны. Передаем единицу потока из вершины 2 в вершину 6 через вершину 5. Вершина-сток достигнута, найдена цепочка $1\to3\to4\to2\to5\to6$, по которой можно передать поток, равный единице. При этом по прямым дугам поток увеличивается на единицу, по обратным – уменьшается. Суммарный поток в сети – 4 единицы.

Четвертая итерация. Вершине 1 присваиваем метку [1,∞]. Первый шаг. Помечаем вершину 3 – [1,4]. Второй шаг. Рассматриваем помеченную, но не просмотренную вершину 3. Одна дуга – (3,4). Вершину 4 пометить не можем – пропускная способность дуги исчерпана. Помеченных вершин больше нет, и вершина-сток не достигнута. Увеличивающую поток цепочку построить не можем. Найден максимальный поток в сети. Можно заканчивать работу.

Итак, в чем суть алгоритма? Первое. На каждой итерации вершины сети могут находиться в одном из трех состояний: вершине присвоена метка, и она просмотрена; вершине присвоена метка, и она не просмотрена, то есть не все смежные с ней вершины обработаны; вершина не имеет метки. Второе. На каждой итерации мы выбираем помеченную, но непросмотренную вершину v и пытаемся найти вершину u, смежную с v, которую можно пометить. Помеченные вершины, достижимые из вершины-источника, образуют множество вершин сети G. Если среди этих вершин окажется вершина-сток, то это означает успешный результат поиска цепочки, увеличивающей поток, при неизменности этого множества работа заканчивается – поток изменить нельзя.