Тема 6. Орграфы и сети

6.1. Основные определения

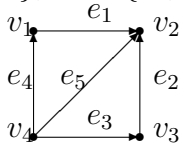
Определение. *Ориентированным графом* (*орграфом*) $D(V, E)$ называется совокупность двух множеств – непустого множества V и множества E упорядоченных пар различных элементов множества V. Множество V называется множеством *узлов*, множество E называется множеством *дуг*.

$$G(V, E) =< V, E >, \quad V \neq \emptyset, \quad E \in V \times V.$$

Число узлов графа обозначим через p, число дуг через q.

На диаграмме дуги изображаются стрелками.

Пример. $V = \{v_1, v_2, v_3, v_4\}$, $E = \{e_1 = \{v_1, v_2\}, e_2 = \{v_3, v_2\}, e_3 = \{v_4, v_3\}, e_4 = \{v_4, v_1\}, e_5 = \{v_4, v_2\}\}$.



Определение. Для орграфа число дуг, исходящих из вершины v, называется *полустепенью исхода* $d^-(v)$, а входящих – *полустепенью захода* $d^+(v)$.

Лемма (Эйлера). Сумма полустепеней исхода и захода узлов орграфа равна удвоенному количеству дуг

$$\sum_{v \in V} d^-(v) + \sum_{v \in V} d^+(v) = 2q.$$

Цепь в орграфе называется *путем*, а цикл – *контуром*.

Определение. Если в орграфе полустепень захода некоторого узла равна нулю, то такой узел называется *источником*, если же нулю равна полустепень исхода, то такой узел называется *стоком*. Бесконтурный орграф с одним источником и одним стоком называется *сетью*.

6.2. Топологическая сортировка

Уровень узла сети – это натуральное число, определяемое следующим образом:

– если $d^+(v) = 0$, то уровень узла v равен нулю (т. е. множество узлов нулевого уровня N_0 состоит только из источников);

– если определены множества $N_0, \ldots, N_k$ узлов уровней $0, \ldots, k$, и в узел v заходят дуги только из этих множеств, причем среди них есть хотя бы одна дуга из узла уровня k, то v – узел уровня $k + 1$.

Иными словами, уровень узла есть длина максимального пути от источника до узла.

Во многих прикладных задачах возникает проблема такого упорядочивания узлов сети, при котором узлы одного уровня располагаются друг под другом, номера уровней

возрастают слева направо, а значит, дуги также направлены слева направо. Такой упорядочивание называется *топологической сортировкой*. Например, такая задача возникает при распараллеливании алгоритмов, когда по некоторому описанию алгоритма нужно составить граф зависимости его операций и, отсортировать его топологически, выяснить, какие операции могут выполняться параллельно. Также топологическая сортировка может применяться к бесконгурным орграфам.

Теорема. *Если орграф не имеет контуров, то в нем имеется вершина, полустепень захода которой равна нулю.*

Алгоритм топологической сортировки

Начало. Задан бесконтурный орграф. Положим $k = 0$.

Шаг 1. Найдем в графе узлы, полустепени захода которых равны нулю. Образуем из них множество N_k.

Шаг 2. Удалим из графа узлы множества N_k вместе с исходящими дугами.

Шаг 3. Если множество узлов графа не пусть, положим $k = k + 1$ и пойдем на шаг 1.

Конец.

Алгоритм можно модифицировать таким образом, чтобы осуществить проверку того, является ли граф сетью. Если это не так, то на очередном шаге 1 в графе не окажется узлов с нулевой полустепенью захода.

6.3. Сетевые графики

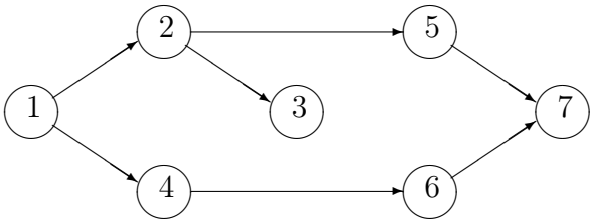
При планировании и управлении сложными комплексами работ используют их графические модели – *сетевые графики*. С математической точки зрения *сетевой график* – это орграф без петель и контуров. Основными понятиями сетевого планирования являются понятия *работы* и *события*.

Работа – это любые действия, сопровождающиеся затратами ресурсов и времени и приводящие к определенным результатам. *Событие* – результат завершения одной или нескольких работ – является в свою очередь предпосылкой для выполнения некоторых других работ. Любая работа может быть определена двумя событиями, между которыми она выполняется. Работы – это дуги сети, а события – ее узлы. Выделяют *исходное* и *завершающее* события.

Если работа представляется дугой $\{u, v\}$, то в узел u входят только дуги, соответствующие непосредственно предшествующим работам. При необходимости вводятся фиктивные дуги, соответствующие работам нулевой продолжительности, и фиктивные узлы, отмечающие конец таких дуг.

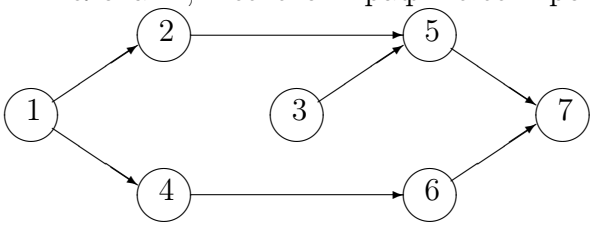
При построении сетевого графика необходимо соблюдать ряд правил.

1. *В сетевой модели не должно быть «тупиковых» событий, за исключением завершающего.* Другими словами, в сетевой модели есть ровно один сток.



Здесь либо работа $\{2, 3\}$ не нужна, и ее необходимо аннулировать, либо не до конца изучены взаимосвязи работ (например, работа $\{2, 3\}$ должна предшествовать какой-либо из имеющихся работ).

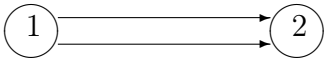
2. *В сетевом графике не должно быть «хвостовых» событий, за исключением исходного.* Другими словами, в сетевом графике есть ровно один источник.

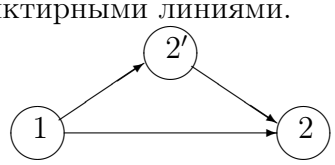


Здесь работы, предшествующие событию 3, не предусмотрены, а значит, событие не может свершиться, и не могут быть выполнена следующая за ним работа $\{3, 5\}$.

3. *В сети не должно быть контуров и петель.* При возникновении этой проблемы необходимо пересмотреть состав работ.

4. *В сети нет кратных дуг.* Иначе говоря, во избежание путаницы любые два события связываются лишь одной работой. В случае, если две работы имеют одинаковые предшествующие работы, вводится фиктивная работа продолжительностью 0 и фиктивное событие. Фиктивные работы обозначаются пунктирными линиями.



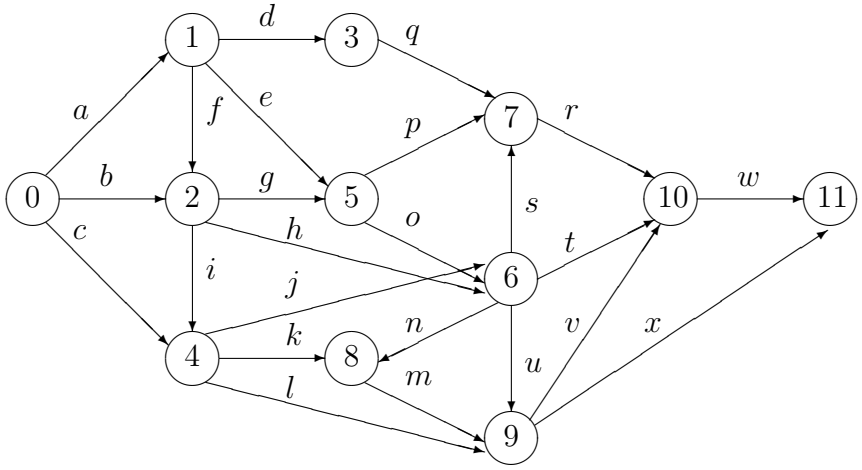


Пример. Пусть заданы работы

работа	время	предшеств. работы
a	2	
b	3	
c	5	
d	2	a
e	4	a
f	3	a
g	5	b, f
h	8	b, f
i	7	b, f
j	9	i, c
k	1	i, c
l	3	i, c

работа	время	предшеств. работы
m	4	k
n	1	h, j, o
o	2	e, g
p	5	e, g
q	6	d
r	1	q, p, s
s	7	h, j, o
t	8	h, j, o
u	2	h, j, o
v	8	m, l, u
w	3	r, t, v
x	5	m, l, u

График имеет вид

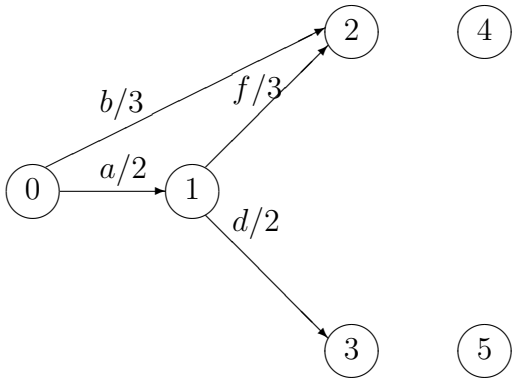


Упорядочение сетевого графика это, в сущности, топологическая сортировка событий. Кроме того, события нумеруются таким образом, чтобы их номера увеличивались слева направо.

Пример.

слой	события
I	0
II	1
III	2, 3
IV	4, 5
V	6
VI	7, 8
VII	9
VII	10
IX	11

Упорядоченный график имеет вид



Найдем все пути, соединяющие источник и сток и определим продолжительность всех путей. Наиболее продолжительный путь называется *критическим*. Он определяет минимальное время выполнения всех работ, называемое *критическим сроком* (обозначение t^*).

Все работы и события, лежащие на критическом пути, называют *критическими*, остальные работы и события – *некритическими*. Задержка выполнения любой критической работы вызывает задержку выполнения всего комплекса работ. Некритические работы могут допускать некоторое запаздывание их выполнения без нарушения срока окончания работ.

Свершением события называется момент, к которому заканчиваются все предшествующие ему работы и может быть начата любая последующая работа.

Ранним сроком $E(v)$ свершения события v называется самый ранний момент времени, к которому завершатся все работы, предшествующие этому событию. Ранние сроки для всех событий могут быть рассчитаны по формуле

$$E(v) = \max_{\{u,v\}\in E}(E(u) + t(u,v)),$$

где $t(u,v)$ – продолжительность работы $\{u,v\}$.

Поздним сроком $L(v)$ свершения события v называется самый поздний момент времени, после которого остается ровно столько времени, сколько требуется для завершения всех работ, следующих за событием. Поздние сроки для всех событий могут быть рассчитаны по формуле

$$L(v) = \min_{\{v,u\}\in E}(L(u) - t(v,u)).$$

Разница между поздним и ранним сроками свершения событий составляет *резерв времени* $R(v)$ для этого события

$$R(v) = L(v) - E(v).$$

Эта величина показывает, на сколько можно задержать свершение события без нарушения критического срока выполнения комплекса работ. У критических событий ранние и поздние сроки совпадают, и резерв равен нулю.

Зная сроки свершения событий, можно найти ранние и поздние сроки начала и окончания работ.

$$T_1^E(u,v) = E(u), \qquad T_2^E(u,v) = E(u) + t(u,v);$$
$$T_1^L(u,v) = L(v) - t(u,v), \quad T_2^L(u,v) = L(v).$$

Для работ определяют два резерва времени. *Полный* резерв времени работы – это максимальное количество времени, на которое можно задержать начало работы или увеличить ее продолжительность, не нарушая критический срок:

$$R(u,v) = L(v) - E(u) - t(u,v).$$

Отдельные работы, помимо полного резерва, имеют *свободный* резерв времени – это часть полного резерва, остающаяся после исключения резерва конечного события данной работы:

$$S(u,v) = E(v) - E(u) - t(u,v).$$

Свободный резерв – это запас времени, на который можно отсрочить начало работы или увеличить ее продолжительность при условии, что ранние сроки начала последующих работ не изменятся. Понятно, что все резервы критических работ равны нулю.

Анализ сетевого графика проводится в четыре этапа.

Этап 1. Вычисление ранних сроков свершения событий $E(v)$. При вычислении перемещаются от события s к событию t, причем $E(s) = 0$, а $E(v)$ рассчитываются по формуле

$$E(v) = \max_{\{u,v\} \in E} (E(u) + t(u,v)).$$

Полученный ранний срок события $E(t)$ – это критический срок выполнения работ.

Этап 2. Вычисление поздних сроков свершения событий $L(v)$. При вычислении перемещаются от события t к событию s, причем $E(s) = 0$, а $E(v)$ рассчитываются по формуле

$$L(v) = \min_{\{v,u\} \in E} (L(u) - t(v,u)).$$

Этап 3. Вычисление резервов времени событий производится по формуле

$$R(v) = L(v) - E(v).$$

Вычисление полного и свободного резервов времени работ производится по формулам

$$R(u,v) = L(v) - E(u) - t(u,v), \quad S(u,v) = E(v) - E(u) - t(u,v).$$

Этап 4. Поиск критического пути. У критических событий резерв времени равен нулю, они и определяют критические работы и критический путь.

Для решения задачи поиска критического пути (т. е. самого длинного пути от источника до стока) можно применить алгоритм Дейкстра, заменив min на max, но он не учитывает структуры графа. Рассмотрим алгоритм, специально разработанный для ациклических орграфов.

Пусть вершины пронумерованы так, что $\forall\ \{x_i, x_j\} : i < j$. Тогда веса самых длинных путей находятся из условия

$$\lambda(x_j) = \max_{x_i \in \Gamma^{-1}(x_j)} \{\lambda(x_i) + Cij\}$$

для $j = 1, \ldots, p$.

6.4. Потоки в сетях

Пусть $D(V, E)$ – сеть, s и t – соответственно источник и сток сети. Дуги сети нагружены неотрицательными числами $C(u, v)$, где $C(u, v)$ называется *пропускной способностью* дуги $\{u, v\}$.

Пусть залана функция $f^E \to \mathcal{R}$. *Дивергенцией* функции f в вершине v называется число $\mathrm{div}(f, v)$, которое определяется следующим образом:

$$\mathrm{div}(f, v) = \sum_{v:\{u,v\}\in E} f(u, v) - \sum_{v:\{v,u\}\in E} f(v, u).$$

Функция $f^E \to \mathcal{R}$ называется *потоком* в сети D, если:

1) $\forall \{u, v\} \in E : \ 0 \le f(u, v) \le C(u, v)$, т. е. поток через дугу неотрицателен и не превосходит пропускной способности дуги;

2) $\forall v \in V \setminus \{s, t\} : \ \mathrm{div}(f, v) = 0$, т. е. дивергенция потока равна нулю во всех вершинах, кроме источника и стока.

Число $w(f) = \mathrm{div}(f, s)$ (т. е. *дивергенция источника*) называется *величиной потока* f.

Разобьем множество вершин сети на два непересекающихся подмножества S и T так, что $s \in S$ и $t \in T$. Множество дуг, соединяющих узлы из S с узлами из T, называется *разрезом*. Разрез P можно разбить на два множества: $P = P^+ \cup P^-$, где P^+ – дуги от S к T, P^- – дуги от T к S. Сумма потоков через дуги разреза P обозначается $F(P)$. Сумма пропускных способностей дуг подмножества разреза P^+ называется *пропускной способностью* разреза и обозначается $C(P)$:

$$F(P) = \sum_{e\in P} f(e),$$
$$C(P) = \sum_{e\in P^+} C(e).$$

6.4.1. Теорема о максимальном потоке и минимальном разрезе

Лемма. $w(f) = F(P^+) - F(P^-)$.

Доказательство. Рассмотрим сумму

$$W = \sum_{v\in S} \mathrm{div}(f, v).$$

Пусть дуга $\{u, v\} \in E$. Если $u, v \in S$, то в сумму W попадают два слагаемых для этой дуги: $+f(u, v)$ при вычислении $\mathrm{div}(f, u)$ и $-f(u, v)$ при вычислении $\mathrm{div}(f, v)$. Итого вклад

этой дуги в сумму равен нулю. Если $u \in S$, $v \in T$, то в сумму W попадает одно слагаемое $+f(u,v)$, все такие слагаемые дают $F(P^+)$. Если $v \in S$, $u \in T$, то в сумму W попадает одно слагаемое $-f(u,v)$, все такие слагаемые дают $F(P^-)$. Таким образом,

$$W = F(P^+) - F(P^-).$$

С другой стороны, по свойствам дивергенции лишь одно слагаемое в этой сумме не равно нулю

$$W = \sum_{v \in S} \mathrm{div}(f,v) = \mathrm{div}(f,s) = w(f).$$

Лемма. $\mathrm{div}(f,s) = -\mathrm{div}(f,t)$.

Доказательство. Рассмотрим разрез, где $S = V \setminus \{t\}$, $T = \{t\}$. В такам разрезе $P^- = \emptyset$. По определению дивергенции и предыдущей лемме имеем:

$$\mathrm{div}(f,s) = w(f) = F(P^+) - F(P^-) = F(P^+) = \sum_{v \in S} f(v,t) = -\mathrm{div}(f,t).$$

Лемма. $w(f) \le F(P^+)$.

Доказательство.

$$w(f) = F(P^+) - F(P^-) \le F(P^+).$$

Лемма. $\max_f w(f) \le \min_P C(P)$.

Доказательство. По предыдущей лемме $w(f) \le F(P^+)$, следовательно,

$$\max_f w(f) \le \min_P F(P^+).$$

По определению $F(P^+) \le C(P)$, следовательно,

$$\min_P F(P) \le \min_P C(P).$$

Из этих двух неравенств следует утверждение леммы.

Дуга называется *насыщенной*, если поток через нее равен ее пропускной способности, т. е. $f(e) = C(e)$.

Теорема Форда и Фалкерсона. *Максимальный поток в сети равен минимальной пропускной способности разреза, т. е. существует поток f^* такой, что*

$$w(f^*) = \max_f w(f) = \min_P C(P).$$

Доказательство. Пусть f – некоторый максимальный поток. Покажем, что существует разрез P такой, что $w(f) = C(P)$. Рассмотрим граф G, полученный из сети D отменой ориентации ребер. Построим множество вершин S следующим образом:

$$\exists < s,u > \in G : < s,u > = v_0 v_1 \dots v_n \ (v_0 = s,\ v_n = u) :$$
$$\{v_i, v_{i+1}\} \in E \Rightarrow f(v_i, v_{i+1}) < C(v_i, v_{i+1});$$
$$\{v_{i+1}, v_i\} \in E \Rightarrow f(v_i, v_{i+1}) > 0.$$

Это означает, что существует цепь $< s, u >$ в графе G, причем дуги графа D, направленные по направлению этой цепи, не насыщены, а дуги против направления имеют положительный поток. Такие цепи называются *увеличивающими*. Отметим, что $s \in S$.

Покажем, что $t \notin S$. Предположим противное, т. е. что $t \in S$. Тогда существует увеличивающая цепь $< s, t >$. Определим величину δ следующим образом:

$$\delta = \min_{e \in <u,t>} \Delta(e), \quad \Delta(e) = \begin{cases} C(e) - f(e), & \text{если } e \text{ ориентировано вдоль } < s, t >; \\ f(e), & \text{если } e \text{ ориентировано против } < s, t > . \end{cases}$$

По определению увеличивающей цепи $\delta > 0$. Величину потока можно увеличить на δ, изменив поток для всех дуг увеличивающей цепи: вычесть эту величину из потока через дуги, направленные против цепи, и прибавив к потоку через дуги, направленные вдоль цепи. При этом условия потока продолжают выполняться: поток не больше пропускной способности и дивергенция во всех узлах, кроме источника и стока, равна нулю. Значит, поток был не максимальный, и $t \notin S$.

Введем обозначение $T = V \setminus S$. Так как $t \in T$, то $T \neq \emptyset$, и множества S и T определяют разрез $P = P^+ \cup P^-$. В этом разрезе все дуги из P^+ насыщены ($f(e) = C(e)$), а все дуги из P^- не нагружены ($f(e) = 0$), иначе множество S можно было бы расширить. Имеем:

$$w(f) = F(P^+) - F(P^-) = C(P^+),$$

значит, P – искомый разрез.

6.4.2. Поиск максимального потока

Введем обозначения:

– N – множество дуг графа, поток в которых не может ни увеличиваться, ни уменьшаться;

– I – множество дуг графа, поток в которых может увеличиваться; $i(u, v)$ – максимальная величина, на которую можно увеличить поток в дуге $\{u, v\}$;

– R – множество дуг графа, поток в которых может уменьшаться; $r(u, v)$ – максимальная величина, на которую можно уменьшить поток в дуге $\{u, v\}$.

Дуги из I называются *увеличивающими*, дуги из R – *уменьшающими*, дуги из N – *промежуточными*. В увеличивающей цепи все прямые дуги – увеличивающие, все обратные – уменьшающие.

Алгоритм поиска увеличивающей цепи

Шаг 1. Определить состав множеств N, I, R. Дуги из N исключить из дальнейшего рассмотрения. Окрасить вершину s.

Шаг 2. Окрашиваеть узлы и дуги по правилу: при окрашенном узле u, если $\{u, v\} \in I$, окрасить неокрашенный узел v и дугу $\{u, v\}$; если $\{v, u\} \in R$, окрасить неокрашенный узел v и дугу $\{v, u\}$. Повторяем шаг 2, пока это возможно.

Шаг 3. Если узел t не окрашен, увеличивающей цепи нет. Иначе существует единственная увеличивающая цепь.

Обоснование алгоритма. Так как дуга окрашивается только вместе с неокрашенным узлом, все окрашенные дуги образуют граф без циклов. Отменяя ориентацию ребер, получаем дерево. Если в него включен узел t, то существует единственная простая цепь, соединяющая s и t. И наоборот, если имеется увеличивающая цепь, то каждая ее вершина окрашивается, в том числе и t.

Алгоритм поиска максимального потока (основан на теореме Форда-Фалкерсона)

Шаг 1. Выбрать любой поток как начальное приближение (можно нулевой поток).

Шаг 2. Для всех дуг $\{u, v\}$ проделать следующее:

– если $f(u, v) < C(u, v)$, положить $i(u, v) = C(u, v) - f(u, v)$ и считать $\{u, v\} \in I$;

– если $f(u, v) > 0$, положить $r(u, v) = f(u, v)$ и считать $\{u, v\} \in R$.

Шаг 3. На множествах I и R применить к сети алгоритм поиска увеличивающей цепи. Если таковую найти не удается, текущий поток является максимальным, конец. Иначе осуществить максимально возможное увеличение потока вдоль увеличивающей цепи и вернуться на шаг 2.

6.4.4. Поиск потока минимальной стоимости

Рассмотрим задачу определения потока заданной величины θ в сети, дуги которой характеризуются не только пропускной способностью $C(u, v)$, но и стоимостью пересылки единичного потока $d(u, v)$ вдоль дуги $\{u, v\}$, где $d(u, v)$ – целое положительное число. Если $\theta > w(f^*)$, где f^* – максимальный поток, то решения нет. Если же $\theta \leq w(f^*)$, то в общем случае существует несколько различных потоков величины θ. Поставим задачу поиска такого потока минимальной стоимости, т. е. минимизации функции

$$\sum_{\{u,v\} \in E} d(u, v) f(u, v) \to \min$$

при ограничениях

$$\begin{aligned}
&(1) \quad \forall \{u, v\} \in E : \ 0 \leq f(u, v) \leq C(u, v); \\
&(2) \quad \forall v \in V \setminus \{s, t\} : \ \operatorname{div}(f, v) = 0; \\
&(3) \quad \operatorname{div}(f, s) = \theta; \\
&(4) \quad \operatorname{div}(f, t) = -\theta.
\end{aligned}$$

Данную задачу можно рассматривать как задачу линейного программирования. Заметим, что задача о максимальном потоке является частным случаем задачи о потоке минимальной стоимости, в которой все $d(u, v) = 0$, а θ равна величине максимального потока. алгоритм решения задачи предложен Фордом и Фалкерсоном и представляет собой модификацию алгоритма поиска максимального потока.

6.4.4.1. Алгоритм Форда и Фалкерсона

Переформулируем задачу

$$P\theta - \sum_{\{u,v\}\in E} d(u,v)f(u,v) \to \max,$$

где P – достаточно большое число. Если интерпретировать P как доход, получаемый от пересылки единицы потока по сети, то это соотношение равно чистой прибыли от пересылки. Например, P может быть равным максимальной полной стоимости пересылки единицы потока по сети.

Алгоритм выполняет следующие действия. Сначала по сети из s в t пересылается как можно больше единиц потока, полная стоимость прохождения по сети каждой единицы которого равна нулю. Затем по сети из s в t пересылается как можно больше единиц потока, полная стоимость прохождения по сети каждой единицы которого равна 1, и т. д., каждый раз стоимость пересылки единицы потока увеличивается на 1. Работа алгоритма заканчивается, когда пропущено θ единиц потока, либо когда дальнейшее увеличения потока невозможно. Иначе говоря, поставленная задача решается многократно для $P = 0, 1, \ldots$.

Задача линейного программирования имеет вид

$$P\theta - \sum_{\{u,v\}\in E} d(u,v)f(u,v) \to \max$$

при ограничениях

$$\begin{array}{ll}
(1) & \forall\{u,v\}\in E:\ 0 \le f(u,v) \le C(u,v); \\
(2) & \forall v \in V\setminus\{s,t\}: \sum\limits_{\{v,u\}\in E} f(v,u) - \sum\limits_{\{u,v\}\in E} f(u,v) = 0; \\
(3) & \sum\limits_{\{s,u\}\in E} f(s,u) - \theta = 0; \\
(4) & \sum\limits_{\{u,t\}\in E} f(u,t) + \theta = 0.
\end{array}$$

Сформулируем двойственную задачу линейного программирования, при этом рассматриваем параметр θ как переменную. Ограничениям (1) поставим в соответствие переменную $\gamma(u,v)$, ограничениям (2), (3), (4) – переменные $p(v)$, $p(s)$, $p(t)$ соответственно. Двойственная задача имеет вид

$$\sum_{\{u,v\}\in E} C(u,v)\gamma(u,v) \to \min$$

при ограничениях

$$\begin{array}{ll}
(1') & \forall\{u,v\}\in E:\ p(u) - p(v) + \gamma(u,v) \ge -d(u,v); \\
(2') & \forall\{u,v\}\in E:\ \gamma(u,v) \ge 0; \\
(3') & -p(s) + p(t) = P.
\end{array}$$

Максимум в исходной задаче и минимум в двойственной достигается, когда

$$\begin{array}{ll}
(a) & \forall\{u,v\}\in E:\ p(u) - p(v) + \gamma(u,v) > -d(u,v) \Rightarrow f(u,v) = 0; \\
(b) & \forall\{u,v\}\in E:\ \gamma(u,v) > 0 \Rightarrow f(u,v) = C(u,v).
\end{array}$$

Положим

$$(*)\qquad \begin{aligned} p(s) &= 0;\\ p(t) &= P;\\ \gamma(u,v) &= \max\{0, p(v) - p(u) - d(u,v)\}; \end{aligned}$$

тогда условия (a), (b) принимают вид

$$(a')\qquad \forall\{u,v\} \in E:\ p(v) - p(u) < d(u,v) \Rightarrow f(u,v) = 0;$$
$$(b')\quad \forall\{u,v\} \in E:\ p(v) - p(u) > d(u,v) \Rightarrow f(u,v) = C(u,v).$$

Таким образом, достаточно найти такие значения $p(v)$ для всех вершин, и такие значения $f(u,v)$ для всех дуг, чтобы выполнялись условия $(*)$, (a'), (b'). Отсюда следует, что поток можно изменять только в тех дугах, где $p(v) - p(u) = d(u,v)$. При этом $f(u,v)$ должны удовлетворять условиям потока (1)–(4).

Алгоритм поиска потока минимальной стоимости

Шаг 1. Полагаем $p(v) = 0$ для всех узлов и $f(u,v) = 0$ для всех дуг (выполнено условие (a')).

Шаг 2. Формируем множество I дуг, в которых допускается увеличение потока:

$$I = \{\{u,v\} : p(v) - p(u) = d(u,v), \quad f(u,v) < c(u,v).$$

Формируем множество R дуг, в которых допускается уменьшение потока:

$$R = \{\{u,v\} : p(v) - p(u) = d(u,v), \quad f(u,v) > 0.$$

Шаг 3. Применить алгоритм поиска максимального потока на множествах I, R. Если в сумме из s в t уже передано θ единиц потока, то получен искомый поток минимальной стоимости, конец. Иначе проверить, не является ли текущий поток максимальным в исходном графе (это осуществляется исследованием разреза, полученного в процессе поиска увеличивающей цепи: если он является разрезом в исходном графе и насыщен, то текущий поток максимален, конец).

Шаг 4. Увеличиваем на 1 переменные $p(v)$ для всех вершин, не окрашенных на шаге 3 (при этом обязательно увеличится $p(t)$). Идем на шаг 2.

Покажем, что на шаге 4 изменение переменных обеспечивает выполнение условий (a'), (b') для любой дуги $\{u,v\}$.

Если оба конца дуги неокрашены, или оба окрашены, то либо обе узловые переменные не увеличиваются, либо обе увеличиваются на 1, и разность $p(v) - p(u)$ не меняется.

Если узел u окрашен, а v не окрашен, то возвожны 3 варианта:

1) $p(v) - p(u) < d(u,v) \Rightarrow p(v) + 1 - p(u) \le d(u,v)$, и условие (a') не нарушается;

2) $p(v) - p(u) > d(u,v) \Rightarrow p(v) + 1 - p(u) > d(u,v)$, и условие (b') не нарушается;

3) $p(v) - p(u) = d(u,v),\ f(u,v) = c(u,v) \Rightarrow p(v) + 1 - p(u) > d(u,v)$, и выполнено условие (b').

Если узел v окрашен, а u не окрашен, то возвожны 3 варианта:

1) $p(v) - p(u) < d(u,v) \Rightarrow p(v) - (p(u)+1) < d(u,v)$, и условие (a') не нарушается;

2) $p(v) - p(u) > d(u,v) \Rightarrow p(v) - (p(u)+1) \geq d(u,v)$, и условие (b') не нарушается;

3) $p(v) - p(u) = d(u,v),\ f(u,v) = 0 \Rightarrow p(v) - (p(u)+1) > d(u,v)$, и выполнено условие (a').

Выполнено также условие $(*)$. При этом P есть стоимость пересылки по всей сети последней единицы потока. Для завершения требуется не более $P+1$ итерации алгоритма, т. е. если стоимости - целые положительные числа, то P – также целое положительное конечное число, т. е. алгоритм закончит работу за конечное число шагов.

Пример. Рассмотрим матрицу пропускных способностей/стоимостей дуг.

	s	a	b	t
s		2/1	1/3	
a			2/1	4/3
b				2/1
t				

Ход решения представим в таблице.

Ит.	$p(s)$	$p(a)$	$p(b)$	$p(t)$	I	R	Окр. дуги	Окр. узлы
0	0	0	0	0	$\emptyset$	$\emptyset$	$\emptyset$	s
1	0	1	1	1	$\{s,a\}$	$\emptyset$	$\{s,a\}$	s, a
2	0	1	2	2	$\{s,a\}$, $\{a,b\}$	$\emptyset$	$\{s,a\}$, $\{a,b\}$	s, a, b
3	0	1	2	3	$\{s,a\}$, $\{a,b\}$, $\{b,t\}$	$\emptyset$	$\{s,a\}$, $\{a,b\}$, $\{b,t\}$	s, a, b, t

Узел t окрашен, найдена увеличивающая цепь $s \to a \to b \to t$, посылаем по ней 2 единицы потока.

Ит.	$p(s)$	$p(a)$	$p(b)$	$p(t)$	I	R	Окр. дуги	Окр. узлы
4	0	1	2	3	$\emptyset$	$\emptyset$	$\emptyset$	s
5	0	2	3	4	$\{s,b\}$	$\{a,b\}$	$\{s,b\}$, $\{a,b\}$	s, a, b
6	0	2	3	5	$\{s,b\}$, $\{a,t\}$	$\{a,b\}$	$\{s,b\}$, $\{a,b\}$, $\{a,t\}$	s, a, b, t

Узел t окрашен, найдена увеличивающая цепь $s \to b \leftarrow a \to t$, посылаем по ней 1 единицу потока.

Ит.	$p(s)$	$p(a)$	$p(b)$	$p(t)$	I	R	Окр. дуги	Окр. узлы
7	0	2	3	5	$\emptyset$	$\emptyset$	$\emptyset$	s

Дуги, исходящие из окрашенного узла, насыщены, значит, найден максимальный поток.

6.4.4.2. Поиск потока минимальной стоимости, основанный на выявлении отрицательных циклов

Сеть $G =< V, E, C, D >$ с потоком f преобразуем в *модифицированную сеть* $\tilde{G} =< V, \tilde{E}, \tilde{C}, \tilde{D} >$ следующим образом:

– $\tilde{E} = E \cup F$, где F – некоторое множество фиктивных дуг. Если поток f проходит по дуге $\{i,j\}$, т. е. $f(i,j) > 0$, то противоположно направленную дугу $\{j,i\}$ включаем в F. При этом $\tilde{c}(j,i) = f(i,j)$, $\tilde{d}(j,i) = -d(i,j)$;

– для всех ненасыщенных дуг, где нет противоположного потока, т. е. для дуг $\{i,j\}$, где $f(i,j) < c(i,j)$, $f(j,i) = 0$, полагаем $\tilde{c}(i,j) = c(i,j) - f(i,j)$, $\tilde{d}(i,j) = d(i,j)$;

– для всех насыщенных дуг, т. е. для дуг $\{i, j\}$, где $f(i, j) = c(i, j)$, $f(j, i) = 0$, полагаем $\tilde{c}(i, j) = 0$, $\tilde{d}(i, j) = \infty$.

Новая сеть дает модифицированные пропускные способности и стоимости (относительно начального потока f) любого дополнительного потока в G. При этом поиск увеличивающей цепи в исходной сети эквивалентен поиску произвольного пути в модифицированной сети. Если дуга насыщена, пропускная способность ее становится равной нулю. Если есть фиктивная дуга, то она направлена против потока, и увеличение потока в фиктивной дуге есть уменьшение потока в дуге исходной сети. Для ненасыщенных дуг с ненулевым потоком пропускная способность уменьшается.

Теорема. *Поток f будет потоком минимальной стоимости тогда и только тогда, когда в сети $\tilde{G}$ не существует никакого контура μ, сумма стоимостей дуг которого отрицательна.*

Доказательство.

Необходимость. Пусть в сети есть контур μ, сумма стоимостей дуг которого отрицательна. Увеличиваем поток в дугах контура на 1. При этом величина потока θ остается неизменной, а к стоимости добавляется сумма стоимостей дуг контура, т. е. общая стоимость потока уменьшается.

Достаточность. Пусть в модифицированной сети все циклы имеют положительную стоимость, однако существует поток f^* величины θ, причем стоимость $C(f^*) < C(f)$. Потоки f и f^* можно разложить на потоки единичной стоимости, проходящие по простым путям и контурам (по одному пути или контуру может проходить несколько единичных потоков). Построим поток $f^* - f$, его величина равна 0, а значит, он отличен от нуля на некоторых контурах в $\tilde{G}$, а так как все эти контуры имеют неотрицательную стоимость, общая стоимость этого потока также будет неотрицательна, что противоречит предположению $C(f^*) < C(f)$.

Алгоритм поиска потока минимальной стоимости

Шаг 1. С помощью алгоритма поиска максимального потока находим любой допустимый поток f с величиной θ.

Шаг 2. Строим модифицированную сеть $\tilde{G} =< V, \tilde{E}, \tilde{C}, \tilde{D} >$.

Шаг 3. Определяем, существует ли в модифицированной сети цикл μ отрицательного веса (можно использовать алгоритм Флойда). Если такого цикла нет, найден поток минимальной стоимости, конец.

Шаг 4. Вычислить величину потока по циклу μ:

$$\delta = \max_{e \in \mu} \tilde{c}(e).$$

Если $\{u, v\} \in \mu$ $\tilde{d}(u, v) < 0$, уменьшить поток в противоположно направленной дуге исходной сети: $f(v, u) = f(v, u) - \delta$. Если $\{u, v\} \in \mu$ $\tilde{d}(u, v) > 0$, увеличить поток в дуге исходной сети: $f(u, v) = f(u, v) + \delta$. Перейти на шаг 2 с новым потоком f.

6.4.4.3. Поиск потока минимальной стоимости, основанный на построении минимального пути

Пусть $\mu^* =< s, t >$ – минимальный путь (т. е. путь минимальной стоимости) от s к t, а $c_{\min}(\mu^*)$ – минимальная из пропускных способностей дуг, входящих в путь μ^*. Если $\theta \leq c_{\min}(\mu^*)$, то задача имеет тривиальное решение – весь поток θ направляется вдоль пути μ^*. Иначе сеть модифицируют, затем опять находят минимальный путь и максимально возможный поток вдоль этого пути. Процедуры модификации сети и нахождения минимального потока повторяются до тех пор, пока нужное количество θ не будет перенаправлено, либо не возникнет граф, в котором нет пути от s к t, что означает отсутствие решения исходной задачи.

Теорема. *Если f – поток минимальной стоимости величины θ в сети G, а μ^* – минимальный путь от s к t в модифицированной сети $\tilde{G}$, то поток $f + 1 \circ \mu^*$ является потоком минимальной стоимости величины $\theta + 1$ в сети G.*

Доказательство. Достаточно показать, что если $\tilde{G}$ относительно f не содержит циклов отрицательного веса, то $\tilde{G}$ относительно $f + 1 \circ \mu^*$ также не содержит циклов отрицательного веса. Если $\tilde{G}$ относительно f не содержит циклов отрицательного веса, то цикл отрицательного веса может появиться, только если в граф $\tilde{G}$ относительно $f + 1 \circ \mu^*$ включена новая дуга $\{u, v\}$ стоимости $-d(v, u)$. Это происходит, если добавленная единица потока проходит по дуге $\{v, u\}$, а $f(v, u) = 0$. С другой стороны, для существования цикла отрицательного веса требуется наличие пути $< v, u >$ стоимости $D(< v, u >)$ такой, что $D(< v, u >) - d(v, u) < 0$, т. е. $D(< v, u >) < d(v, u)$, что противоречит тому, что дуга $\{v, u\}$ входит в путь минимальной стоимости μ^*.

Алгоритм поиска потока минимальной стоимости

Шаг 1. Находим минимальный путь $\mu^* =< s, t >$ от s к t. Если таких путей нет, то найден максимальный поток, конец.

Шаг 2. Находим δ – минимальную из пропускных способностей дуг, входящих в путь μ^*. Если $\theta < \delta$, то полагаем $\delta = \theta$. Если $\{u, v\} \in \mu^*$ $\tilde{d}(u, v) < 0$, уменьшаем поток в противоположно направленной дуге исходной сети: $f(v, u) = f(v, u) - \delta$. Если $\{u, v\} \in \mu$ $\tilde{d}(u, v) > 0$, увеличиваем поток в дуге исходной сети: $f(u, v) = f(u, v) + \delta$. Полагаем $\theta = \theta - \delta$. Если $\theta = 0$, поток минимальной стоимости найден, конец.

Шаг 3. Сеть $G =< V, E, C, D >$ с потоком f преобразуем в сеть $\tilde{G} =< V, \tilde{E}, \tilde{C}, \tilde{D} >$ следующим образом:

– $\tilde{E} = E \cup F$, где F – некоторое множество фиктивных дуг. Если поток f проходит по дуге $\{i, j\}$, т. е. $f(i, j) > 0$, то противоположно направленную дугу $\{j, i\}$ включаем в F. При этом $\tilde{c}(j, i) = f(i, j)$, $\tilde{d}(j, i) = -d(i, j)$;

– для всех ненасыщенных дуг, где нет противоположного потока, т. е. для дуг $\{i, j\}$, где $f(i, j) < c(i, j)$, $f(j, i) = 0$, полагаем $\tilde{c}(i, j) = c(i, j) - f(i, j)$, $\tilde{d}(i, j) = d(i, j)$;

– для всех насыщенных дуг, т. е. для дуг $\{i, j\}$, где $f(i, j) = c(i, j)$, $f(j, i) = 0$, полагаем $\tilde{c}(i, j) = 0$, $\tilde{d}(i, j) = \infty$.

Идем на шаг 1.

Конец.

Минимальный путь можно найти с помощью алгоритма Форда-Беллмана либо Беллмана–Мура. Алгоритм Дейкстра не подходит, т. к. веса дуг могут быть отрицательными. Предполагается, что граф не содержит контуров отрицательного веса, иначе, включая этот контур в путь, можно добиться сколь угодно малого (отрицательного) веса пути.