МАКСИМАЛЬНЫЙ ПОТОК

МАКСИМАЛЬНЫЙ ПОТОК

Рассмотрим ориентированный граф. Будем рассматривать его как сеть труб, по которым некоторое вещество движется от истока к стоку, или как движение тока по проводам, информации по линиям связи или товаров от производителя к потребителю и т.д. Пометки ребер будем рассматривать, напрмер, как ширину дороги, или пропускную способность трубы.

Будем считать, что в вершинах вещество не накапливается — сколько приходит, столько и уходит (если вершина не является истоком или стоком). Это свойство называется «законом сохранения потока» (flow conservation).

Задача о максимальном потоке для данной сети состоит в следующем: найти максимально возможную скорость производства (и потребления) вещества, при которой его ещё можно доставить от истока к стоку при данных пропускных способностях труб.

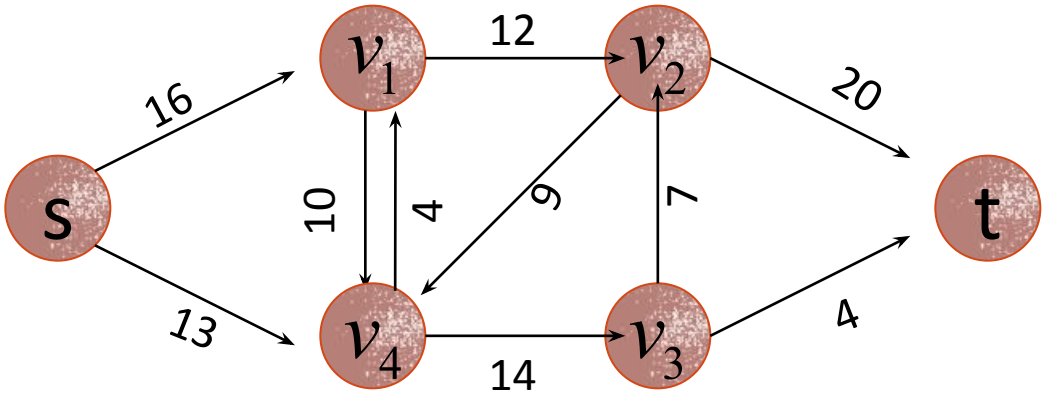
ОПРЕДЕЛЕНИЕ СЕТИ

Назовем *сетью* ориентированный граф $G = (V, E)$, каждому ребру $(u, v) \in E$ которого поставлено в соответствие число $c(u, v) \geq 0$, называемое

пропускной способностью ребра.

В случае $(u, v) \notin E$ полагаем $c(u, v) = 0$.

В графе выделены две вершины: *исток* s и *сток* t.



(а) **Сеть 1** с указанием пропускных способностей ребер

ОПРЕДЕЛЕНИЕ ПОТОКА

Пусть дана сеть $G = (V,E)$, пропускная способность которой задаётся функцией c. Сеть имеет исток s и сток t.

Потоком в сети G назовём функцию $f : V \times V \longrightarrow R$, удовлетворяющую трём свойствам:

1) Ограничение, связанное с *пропускной способностью*:

$f(u, v) \leq c(u, v)$ для всех u, v из V.

Поток из одной вершины в другую не превышает пропускной

способности ребра.

2) *Кососимметричность*: $f(u,v) = -f(v,u)$ для всех u, v из V.

Величина $f(u,v)$ может быть как положительной, так и отрицательной, отрицательные значения соответствуют движению в обратную сторону.

$\Rightarrow$ $f(u, u) = 0$ для любой вершины u.

3) *Сохранение* потока: $\sum_{v \in V} f(u,v) = 0$ для всех $u \in \{V - \{s, t\}\}$.

Для любой вершины u (кроме стока и истока) сумма потоков во все другие вершины равна нулю.

ВЕЛИЧИНА ПОТОКА

Величина потока f определяется как сумма $\sum_{v \in V} f(s,v)$ и

обозначается как $|f|$.

Складываем потоки по всем рёбрам, выходящим из истока.

Учитывая кососимметричность, свойство 3) можно

переписать как $\sum_{u \in V} f(u,v) = 0$, т.е. сумма всех потоков из

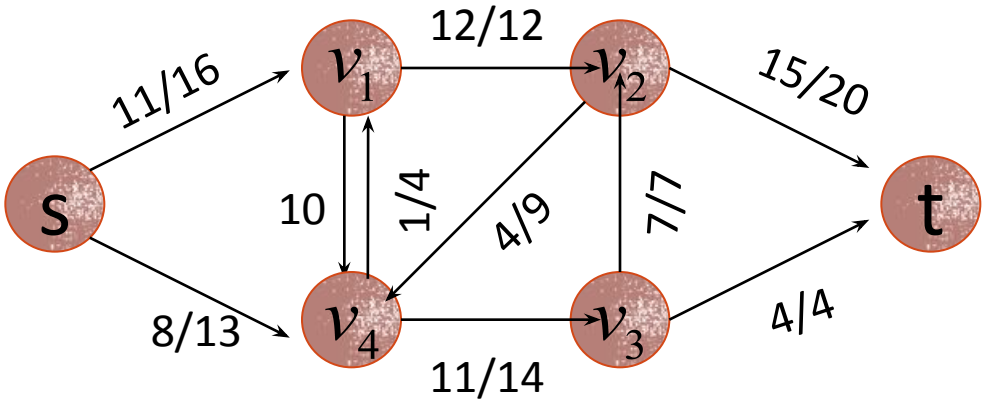
других вершин равна нулю.

Задача о максимальном потоке состоит в следующем:

для данной сети G с истоком s и стоком t найти поток

максимальной величины.

ПОТОК В СЕТИ 1 ВЕЛИЧИНЫ 19



(б) Поток величины 19

На ребрах указана величина потока $f(u, v)$; если $f(u, v) = 0$, поток не указывается.

Заметим также, что если вершины *и* и *v* не соединены ребром, то поток между ними равен нулю.

Действительно, если $(и, v) \notin E$ и $(v, и) \notin E$, то $c(и, v) = c(v, и) = 0$. Тогда из первого свойства следует, что $f(u,v) \leq 0$ и $f(v,u) \leq 0$. Вспоминая, что $f(u,v) = —f(v,u)$ получим: $f(u,v) = f(v,u) = 0$.

Разделим вещество, поступающее в данную вершину *v* и вещество, из неё выходящее, то есть положительные и отрицательные значения $f(u,v)$.

Сумму $\sum_{u \in V} f(u,v)$ назовём **входящим** (в вершину *v*) потоком.

$f(u,v)>0$

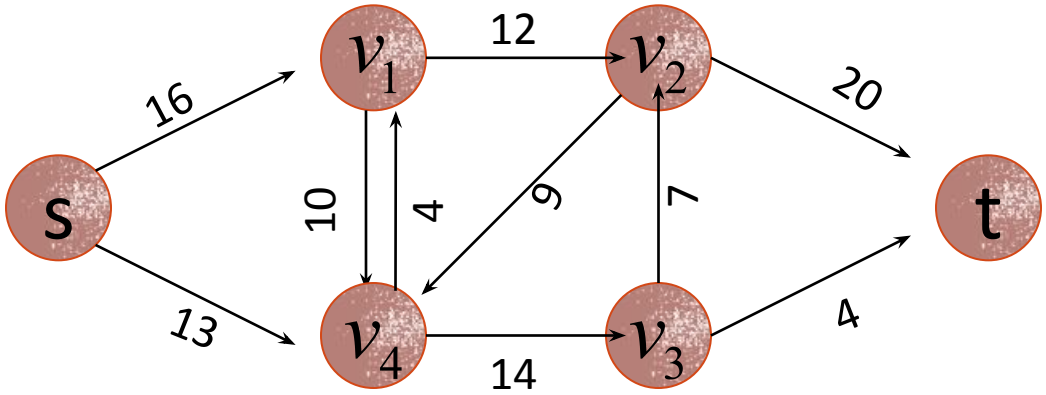
Аналогично определим **выходящий** поток.

Тогда свойство 3): для любой вершины, кроме истока и стока, **входящий поток равен исходящему.**

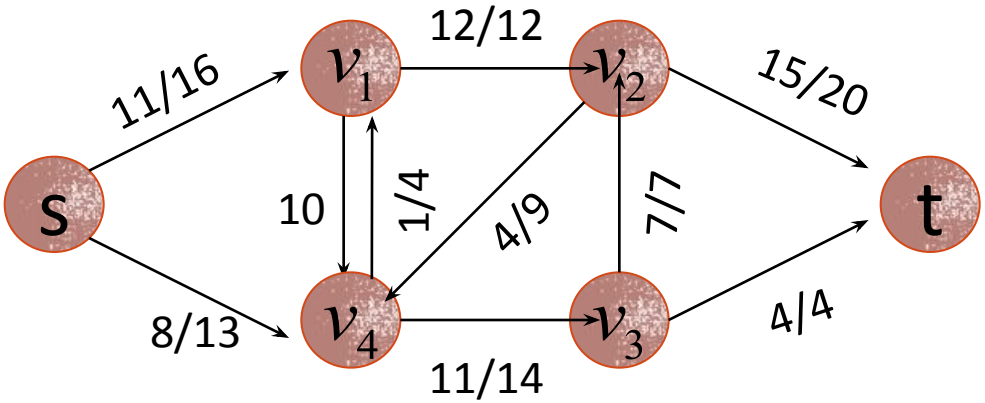
ПРИМЕР

Компания производит изделия на фабрике в городе A (исток s) и складирует их в городе B (сток t). Она арендует место в грузовиках другой фирмы, и место это ограничено: из города u в город v можно доставить не более $c(u, v)$ ящиков в день. Ограничения $c(u,v)$ показаны на рисунке. Задача состоит в том, чтобы перевозить максимально возможное количество изделий из г. A в г. B ежедневно. При этом путь может занимать несколько дней, и ящики могут ждать отправки в промежуточных пунктах, но необходимо, чтобы для каждого пункта число ежедневно прибывающих ящиков было равно числу увозимых, иначе ящиков нехватит или они будут накапливаться. Тем самым выполнено свойство сохранения потока. Величиной потока будет число изделий, ежедневно отгружаемых из г. A, и нас интересует поток максимальной величины.

Пример



(а) На каждом ребре написано максимальное число ящиков, которые можно отправить в день.



(б) Пример потока f величины 19. Показаны только положительные значения $f(u, v) > 0$, после косой черты стоит пропускная способность $c(u,v)$.

ПРИМЕР

Модель не учитывает встречные перевозки. Если из вершины v_1 в v_2 ежедневно везут восемь ящиков, а из v_2 в v_1 ежедневно везут три ящика, чему должны быть равны $f(v_1,v_2)$ и $f(v_2,v_1)$?

Эти величины должны быть противоположны.

Мы полагаем $f(v_1,v_2) = 8 — 3 = 5$, а $f(v_2,v_1) = —5$.

Те же значения функции f соответствуют ежедневным перевозкам пяти ящиков из v_1 в v_2, так что в модели встречные перевозки автоматически сокращаются.

Это разумно, экономика должна быть экономной.

ПРИМЕР СОКРАЩЕНИЯ ПОТОКОВ МЕЖДУ ВЕРШИНАМИ

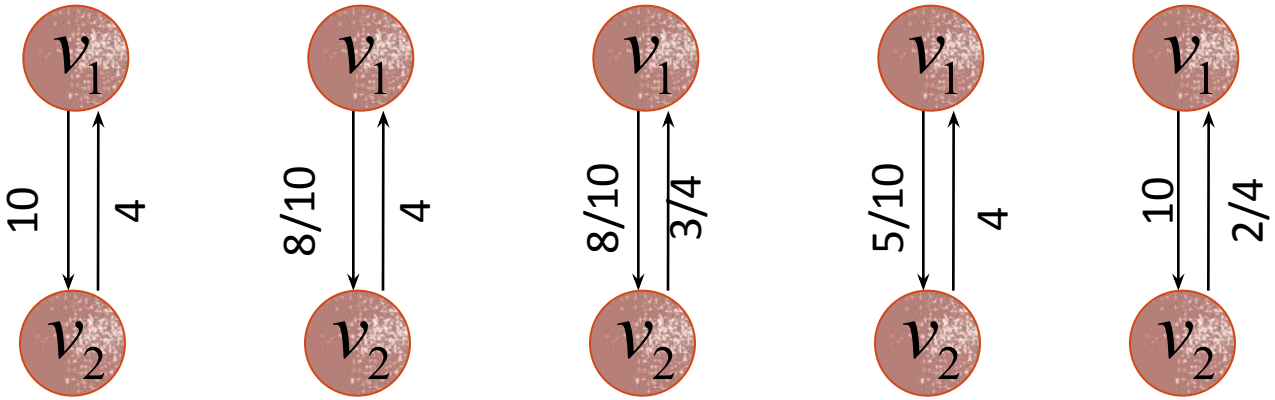
$c(v_1, v_2,) = 10$; $c(v_2, v_1) = 4$.

Сначала из *v*1 в *v*2 возили ежедневно 8 ящиков.

Затем стали возить 3 ящика в обратном направлении.

Потом уменьшили число перевозимых в обратную сторону ящиков на 3.

Эти две различные (в жизни) ситуации соответствуют одной и той же функции *f*, в обоих случаях $f(v_1,v_2) = 5$, а $f(v_2, v_1) = -5$.



(а) (б) (в) (г) (д)

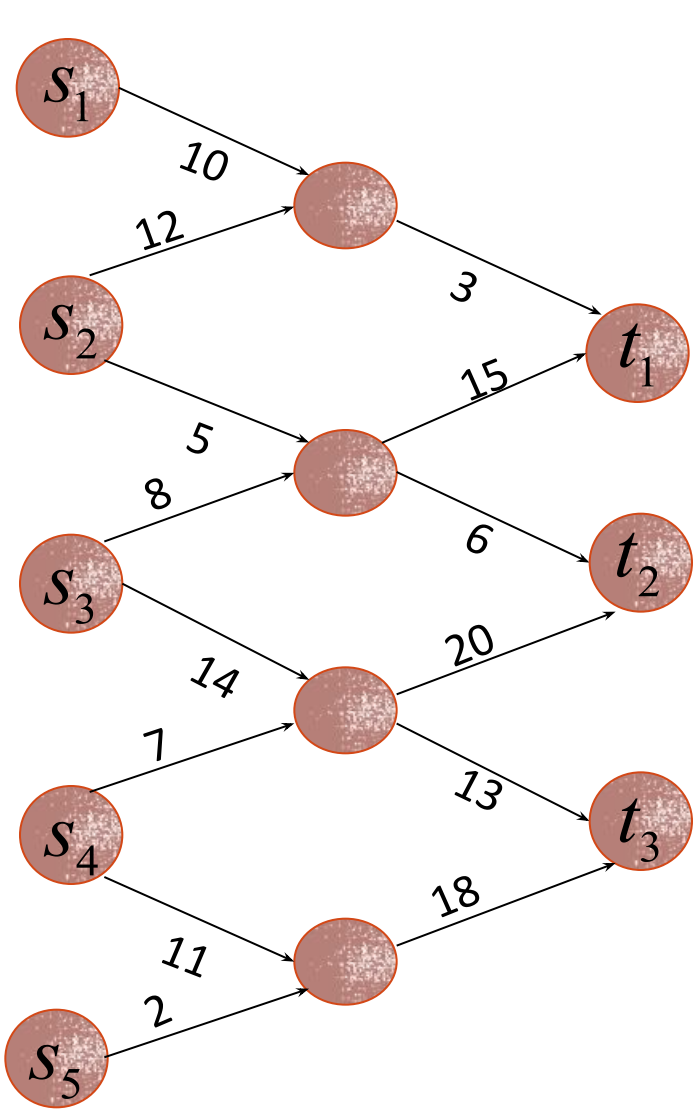
Если требуется начать перевозки дополнительных 7 ящиков в день из v_2 в v_1, то нужно прежде всего отменить перевозки 5 ящиков в обратную сторону, после чего назначить перевозку дополнительных 2 ящиков.

НЕСКОЛЬКО ИСТОКОВ

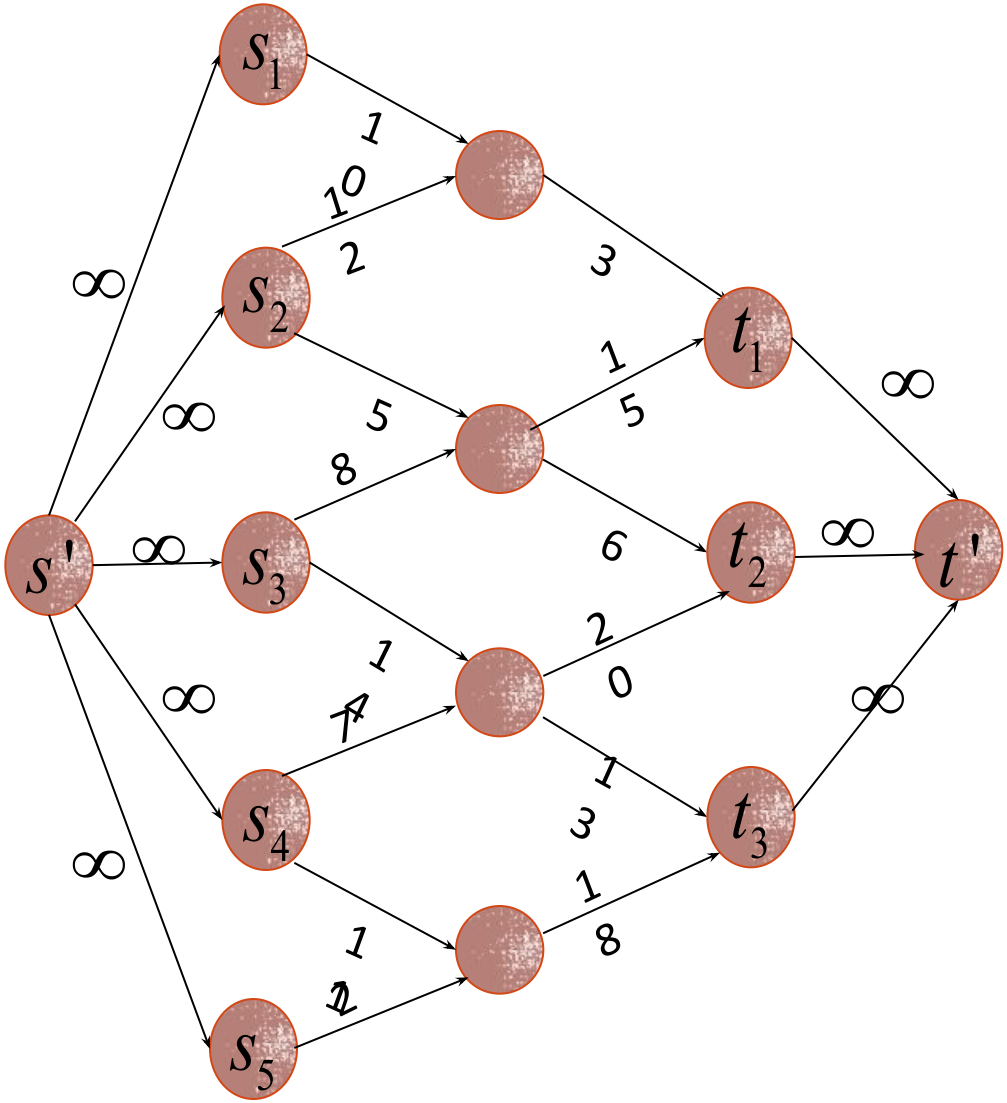
Задача о максимальном потоке для нескольких истоков и стоков сводится к обычной построением эквивалентной сети с одним истоком и одним стоком. А именно: в сеть добавляется *общий* исток *s*, из которого ведут рёбра бесконечной пропускной способности во все прежние истоки. Аналогичным образом из всех прежних стоков проведены рёбра в *общий* сток *t*.

Добавленные рёбра имеют бесконечную пропускную способность.

В примере (след. слад) легко видеть, что каждый поток в сети (а) соответствует потоку в сети (Ь) и наоборот.


s_1
s_2
s_3
s_4
s_5
t_1
t_2
t_3
10
12
3
15
5
8
6
14
20
7
13
11
18
2

(а)


s'
s_1
s_2
s_3
s_4
s_5
t_1
t_2
t_3
t'
∞
1
10
2
3
5
8
6
20
7
13
11
18
2

(б)

ОБОЗНАЧЕНИЯ

Будем использовать следующее соглашение: если в выражении на месте вершины стоит множество вершин, то имеется в виду сумма по всем элементам этого множества (*неявное суммирование*).

Это относится и к случаю нескольких переменных. Например, если X и Y — множества вершин, то

$$f(X,Y) = \sum_{x \in X} \sum_{y \in Y} f(x,y)$$

В этих обозначениях закон сохранения потока запишется как $f(u,V) = 0$ для всех $u \in V - \{s,t\}$.

Кроме того, в неявных суммах мы опускаем фигурные скобки.

ЛЕММА 1

Пусть f — поток в сети $G = (V,E)$.

Тогда для любого $X \subset V$ выполнено

$f(X,X) = 0$.

Для любых $X, Y \subset V$ выполнено

$f(X,Y) = -f(Y,X)$.

Для любых $X, Y, Z \subset V$ из $X \cap Y = \varnothing$ следует

$f(XUY, Z) = f(X, Z) + f(Y, Z)$ *и*

$f(Z, XUY) = f(Z, X) + f(Z, Y)$.

ОСТАТОЧНЫЕ СЕТИ

Пусть дана сеть $G = (V, E)$ с истоком s и стоком t. Пусть f — поток в этой сети. Для любой пары вершин u и v определяется *остаточная пропускная способность* из u в v:

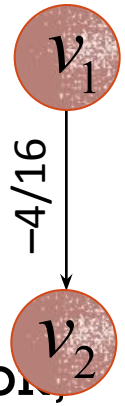
$$c_f(u,v) = c(u,v) - f(u,v).$$

Она определяет, сколько ещё потока можно направить из u в v.

Например, если $c(u,v) = 16$, а $f(u,v) = 11$ то мы можем переслать ещё $c_f(u, v) = 5$ единиц по ребру (u, v).

Остаточная пропускная способность $c_f(u, v)$ может превосходить $c(u, v)$, если в данный момент поток $f(u, v)$ отрицателен.

Например, если $c(u, v) = 16$, а $f(u, v) = -4$, то $c_f(u,v) = 20$.



Мы можем увеличить поток на 4, отменив встречный поток, и ещё отправить 16 единиц, не превышая пропускной способности ребра (u, v).

Неформально говоря, остаточная сеть состоит из тех рёбер, поток по которым можно увеличить.

ОПРЕДЕЛЕНИЕ

Сеть $G_f = (V, E_f)$, где

$$E_f = \{ (u,v) \in V \times V : c_f(u, v) > 0 \}$$

назовём *остаточной сетью* сети G, порождённой потоком f.

Её рёбра, называемые *остаточными* рёбрами, допускают положительный поток.

Заметим, что остаточное ребро $(и, v)$ не обязано быть ребром

сети G, может оказаться, что $E_f \not\subset E$.

Такое ребро из $и$ в v появляется, когда f(и, v) < 0, то есть когда

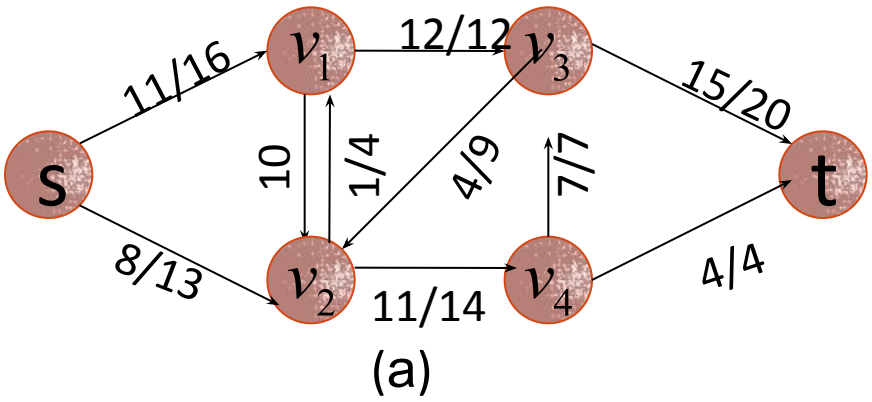
имеется имеется поток вещества в обратном направлении

(по ребру $(v, и)$) — ведь этот поток можно уменьшить.

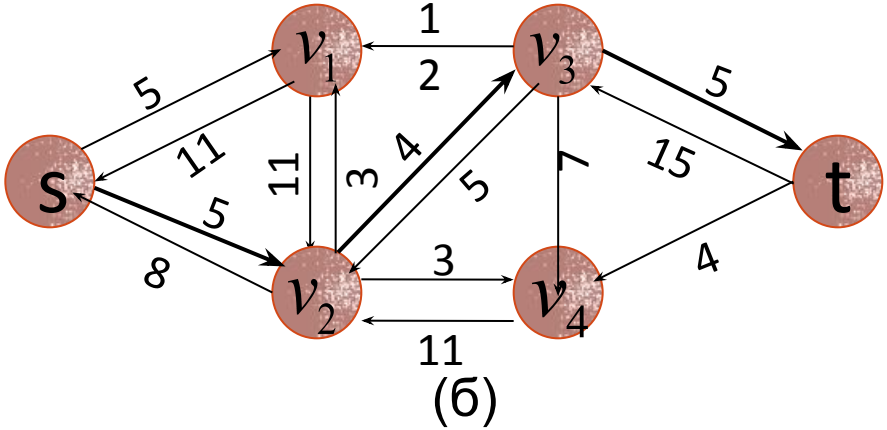
Если ребро $(и, v)$ принадлежит остаточной сети, то хотя бы одно

из рёбер $(и, v)$ и $(v, и)$ было в исходной сети.

ПРИМЕР ОСТАТОЧНОЙ СЕТИ



(а)



(б)

На рис. а) изображен поток *f* в сети *G*, на рис. б) – остаточная сеть G_f и выделен путь *p*, по которому можно еще пропустить 4 единицы. Рёбер (v_1, s) и (v_2, v_3) не было в исходной сети.

$$c(v_1, v_2) = 10;\ c(v_2, v_1) = 4;\ f(v_2, v_1) = 1;$$
$$c_f(v_2, v_1) = 4 - 1 = 3;\ c_f(v_1, v_2) = 10 - (-1) = 11;$$

$$c(v_3, v_2) = 9;\ c(v_2, v_3) = 0;\ f(v_3, v_2) = 4;$$
$$c_f(v_3, v_2) = 9 - 4 = 5;\ c_f(v_2, v_3) = 0 - (-4) = 4;$$

СООТНОШЕНИЕ ПОТОКОВ СЕТИ И ОСТАТОЧНОЙ СЕТИ

Остаточная сеть G_f является сетью с пропускными способностями C_f.

Пусть имеется сеть $G = (V, E)$ и два потока f_1 и f_2 на ней.

Рассмотрим их сумму (функцию из V X V в R):

$$(f_1 + f_2)(u, v) = f_1(u, v) + f_2(u, v) \qquad (1)$$

Лемма 2

Пусть $G = (V, E)$ — сеть с истоком s и стоком t, а f — поток в ней.

Пусть G_f — остаточная сеть сети G, порождённая потоком f.

Пусть f' — поток в G_f. Тогда сумма $f + f'$, определенная как в (1), является потоком в сети G величины $|f + f'| = |f| + |f'|$.

ДОКАЗАТЕЛЬСТВО ЛЕММЫ 2

Сначала докажем, что $f + f'$ будет потоком.

1) Проверим условие, связанное с ограниченной пропускной

способностью. Заметим, что $f'(u,v) \le c_f(u,v)$ для всех $u, v \in V$,

Поэтому

$(f + f')(u, v) = f(u, v) + f'(u, v) \le f(u, v) + (c(u, v) - f(u, v)) = c(u, v).$

2) Кососимметричность. Для всех $u, v \in V$ выполнено

$(f + f')(u, v) = f(u,v) + f'(u,v) = -f(v,u) - f'(v,u) = -(f(v,u) + f'(v,u)) =$

$-(f + f')(u, v)$

3) Закон сохранения потока. Для всех $u \in V \setminus \{s, t\}$ выполнено:

$(f + f')(u,V) = f(u,V) + f'(u,V) = f(u,V) + f'(u,V) = 0 + 0 = 0.$

Наконец, найдём величину суммарного потока:

$|f + f'| = (f + f')(s,V) = f(s,V) + f'(s,V) = |f| + |f'|.$

ДОПОЛНЯЮЩИЕ ПУТИ

Пусть f — поток в сети $G = (V,E)$.

Назовём *дополняющим путём* простой путь из истока s в сток t в остаточной сети G_f.

Из определения остаточной сети вытекает, что по всем ребрам (u, v) дополняющего пути можно переслать ещё сколько-то вещества, не превысив пропускную способность ребра.

Величину наибольшего потока, который можно переслать по дополняющему пути p, назовём *остаточной пропускной способностью* пути p:

$$c_f = \min \{ c_f(u,v) : (u,v) \in p \}.$$

ЛЕММА 3

Пусть f — поток в сети $G = (V, E)$ и

p — дополняющий путь в G_f.

Определим функцию $f_p : V \times V \to R$:

$$f_p(u,v) = \begin{cases} C_f(p), & \text{если } (u, v) \in p \\ -C_f(p), & \text{если } (v,u) \in p \\ 0, & \text{в остальных случаях.} \end{cases} \qquad (1)$$

Тогда f_p — поток в сети G_f и $|f_p| = c_f(p) > 0$.

⇒ Если добавить поток f_p к потоку f, то получится поток

в сети G с большим значением.

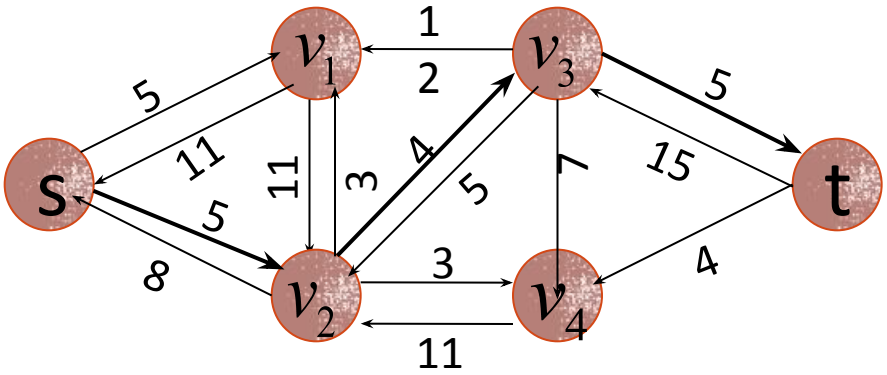
СЛЕДСТВИЕ 4

Пусть f — поток в сети $G = (V, E)$, а p — дополняющий путь в сети G_f, заданный равенством (1).

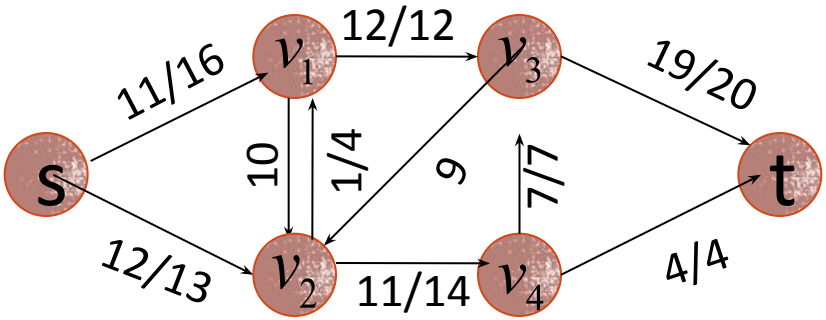
Тогда функция $f' = f + f_p$ является потоком в сети G величины $|f'| = |f| + |f_p| > |f|$.

Доказательство вытекает из лемм 2 и 3.

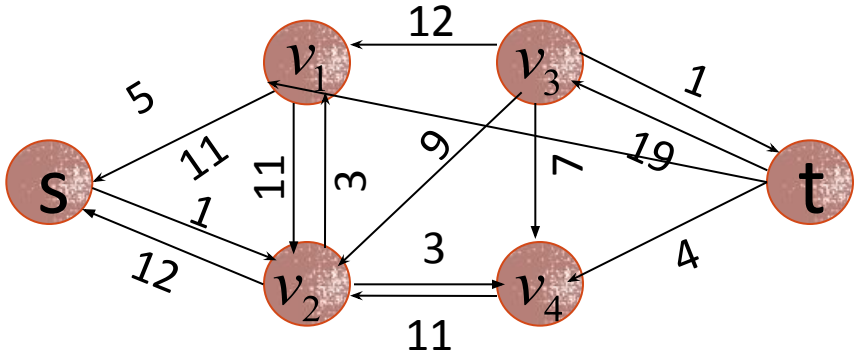
ПРИМЕР. ДОБАВЛЕНИЕ ПОТОКА ВЕЛИЧИНЫ 4 И ОСТАТОЧНАЯ СЕТЬ



(б)



(в)



(г)

На рис. в) показана сеть – результат добавления потока величины 4, проходящего вдоль пути *p*. На рис. г) показана остаточная сеть, порожденная потоком с рис. в).

РАЗРЕЗЫ В СЕТЯХ

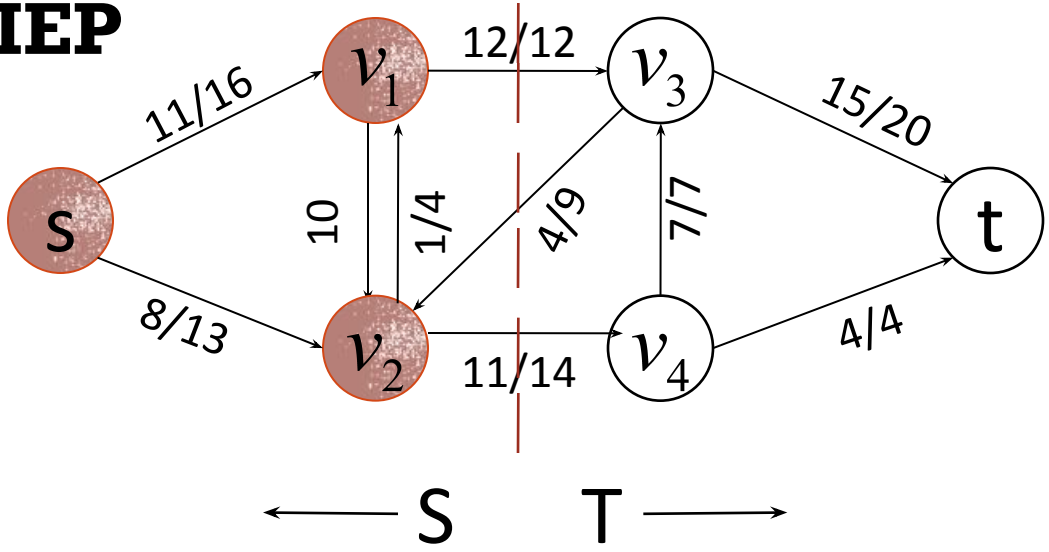
Назовём разрезом сети $G = (V, E)$ разбиение множества V на две части S и $T = V \setminus S$, для которых $s \in S$ и $t \in T$.

Пропускной способностью разреза (S, T) называют сумму $c(S, T)$ пропускных способностей пересекающих разрез ребер (по всем ребрам из $S \rightarrow T$) .

Для заданного потока f величина потока через разрез (S, T) определяется как сумма $f(S, T)$ по пересекающим разрез ребрам.

Минимальным разрезом называют разрез наименьшей пропускной способности среди всех разрезов данной сети.

ПРИМЕР



На рис. изображён разрез ($\{s, v_1, v_2\}, \{v_3, v_4, t\}$).

Поток через него равен

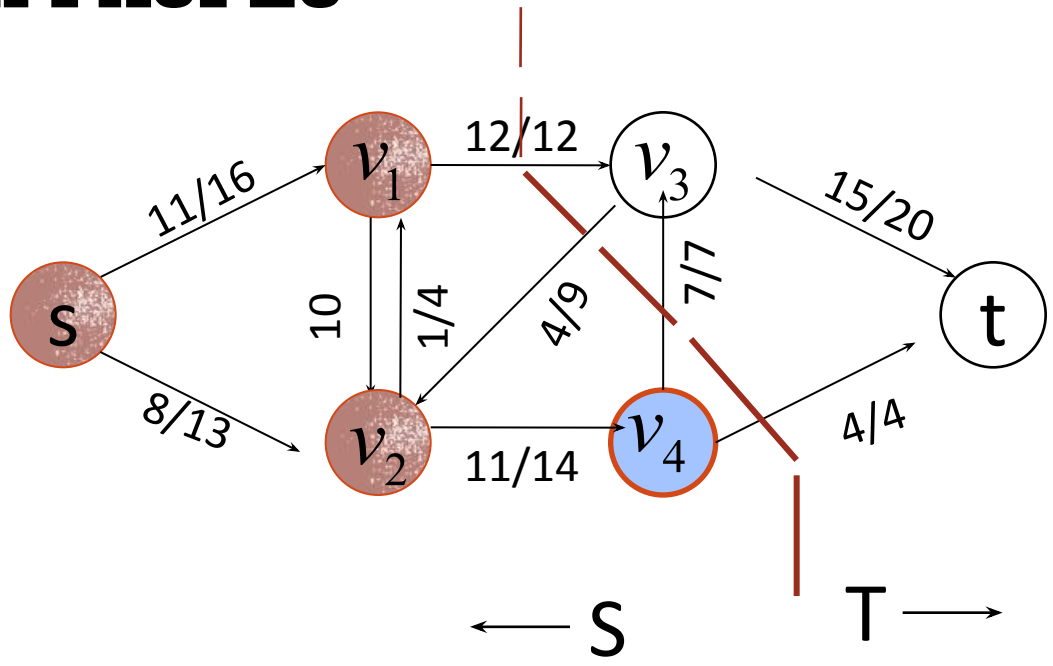
$$f(v_1, v_3) + f(v_2, v_3) + f(v_2, v_4) = 12 + (-4) + 11 = 19,$$

Пропускная способность разреза равна

$$c(v_1, v_3) + c(v_2, v_4) = 12 + 14 = 26.$$

Поток через разрез, в отличие от пропускной способности разреза, может включать и отрицательные слагаемые.

ДРУГОЙ РАЗРЕЗ



На рис. изображён разрез $(\{s, v_1, v_2, v_4\}, \{v_3, t\})$.

Поток через него равен

$$f(v_1, v_3) + f(v_2, v_3) + f(v_4, v_3) + f(v_4, t) = 12 + (-4) + 7 + 4 = 19,$$

Пропускная способность разреза равна

$$c(v_1, v_3) + c(v_4, v_3) + c(v_4, t) = 12 + 7 + 4 = 23.$$

ЛЕММА 5

Пусть f – поток в сети G, а (S, T) разрез сети G. Поток f (S, T) через разрез равен $|f|$.

Следствие 6

Величина любого потока f в сети G не превосходит пропускной способности любого разреза сети G.

Доказательство

$$|f| = f(S, T) = \sum_{u \in S} \sum_{y \in T} f(u,v) \leq \sum_{u \in S} \sum_{y \in T} c(u,v) = c(S, T)$$

ТЕОРЕМА 7 О МАКСИМАЛЬНОМ ПОТОКЕ И МИНИМАЛЬНОМ РАЗРЕЗЕ

Пусть f — поток в сети $G = (V, E)$.

Тогда следующие утверждения равносильны:

1. Поток f максимален (является потоком максимальной величины) в сети G.
2. Остаточная сеть G_f не содержит дополняющих путей.
3. Для некоторого разреза (S, T) сети G выполнено

   равенство $|f| = c(S, T)$.

   В этом случае разрез является минимальным, то есть имеет минимально возможную пропускную способность.

ДОКАЗАТЕЛЬСТВО

1) => (2) От противного. Пусть что поток f максимален, но G_f содержит дополняющий путь p. Рассмотрим сумму $f + f_p$, где f_p задается равенством 1). По следствию 4 эта сумма является потоком в G, величина которого больше $|f|$, что противоречит максимальности.

(2) => (3) Пусть в сети G_f нет пути из истока s в сток t.

Рассмотрим множество $S = \{ v \in V:$ в G_f существует путь из s в $v\}$.

Положим $T = V \backslash S$. Очевидно, что $s \in S$, а $t \in T$, так как в G_f нет пути

из s в t. Поэтому пара (S,T) — разрез. Ни для каких $u \in S$ и $v \in T$ ребро (u, v) не принадлежит E_f (в противном случае вершина v попала бы в S). Поэтому $f(u, v) = c(u, v)$.

По лемме 5 $|f| = f(S,T) = c(S,T)$.

(3) => (1) Для любого разреза (S,T) верно $|f| \leq c(S,T)$ (следствие 6).

МЕТОД ФОРДА-ФАЛКЕРСОНА

Поиск максимального потока проводится последовательно. Вначале поток нулевой. На каждом шаге мы увеличиваем значение потока. Для этого мы находим дополняющий путь, по которому можно пропустить ещё немного вещества, и используем его для увеличения потока. Этот шаг повторяется, пока есть дополняющие пути. Полученный поток будет максимальным.

```
Ford-Fulkerson-Method(G, s, t) {
  положить поток f равным 0;
  while ( существует дополняющий путь )
      дополнить f  вдоль p;
  return f;
}
```

АЛГОРИТМ

На каждом шаге вибираем *произвольный* дополняющий путь *p* и увеличиваем поток *f*, добавляя поток величины $c_f(p)$ по пути *p*. В массиве *f* [*u*, *v*] хранятся текущие значения потока.

```
Ford-Fulkerson(G,s,t)
 for(для каждого ребра (u,v) ) {
         f[u,v] = 0; f[v,u]= 0;  }
  while( в Gf существует путь p из s в t){
      cf(p)← min{cf(u,v):(u,v)входит в p}
          for(для каждого ребра(u,v)пути p){
              f[u,v]=f[u,v]+cf(p);
              f[v,u]= -f[u,v];
      }
 }
}
```

АНАЛИЗ

Время работы процедуры Ford-Fulkerson зависит от того, как ищется путь p. Если выбирать дополняющий путь при помощи поиска в ширину, то алгоритм работает полиномиальное время.

Рассмотрий случай, когда пропускные способности – целые числа. Инициализация требует $O(E)$. Цикл while выполняется не более $|f^*|$ раз, где f^* - максимальный поток.

Поиск дополняющего пути в остаточной сети займет время $O(E)$.

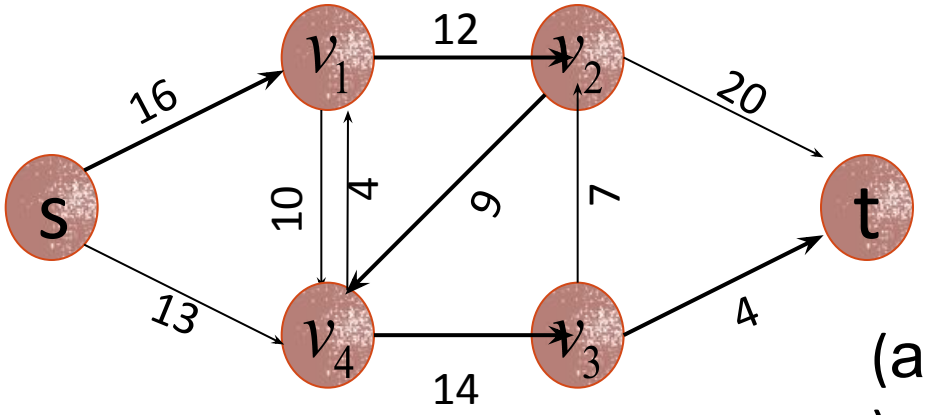
Поэтому время работы процедуры будет $O(E\,|f^*|)$.

Если использовать поиск в ширину, то путь p будет кратчайшим из дополняющих путей (длину каждого ребра считаем равной 1)

Эта реализация метода Форда-Фалкерсона называется алгоритмом Эдмондса-Карпа.

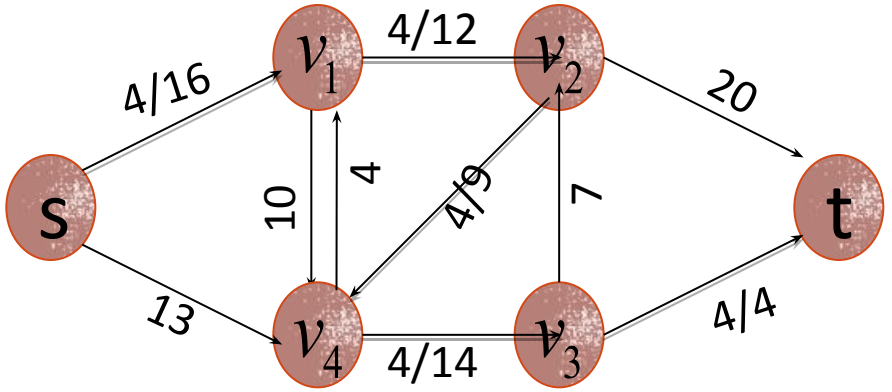
Пример

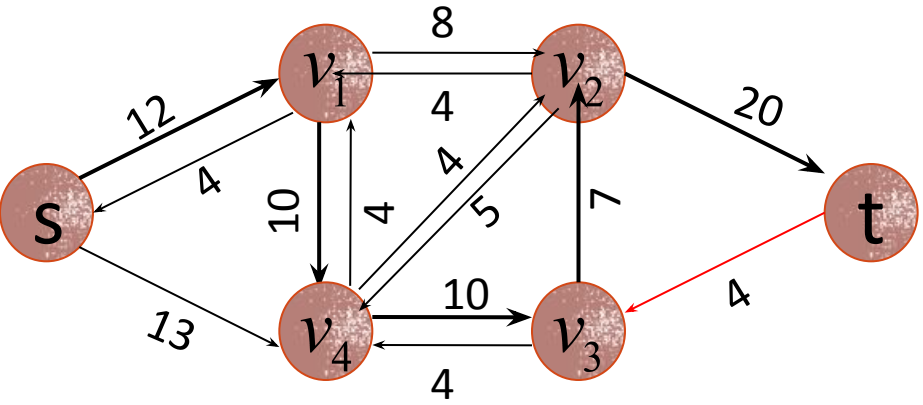
Остаточная сеть и дополняющий путь



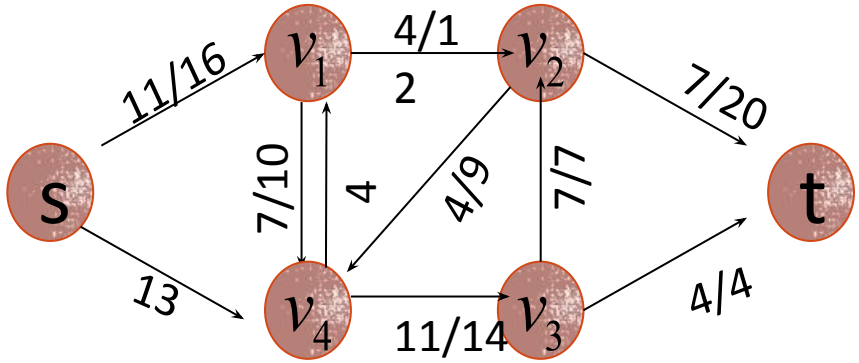
(а)

Увеличенный поток $f_p = 4$



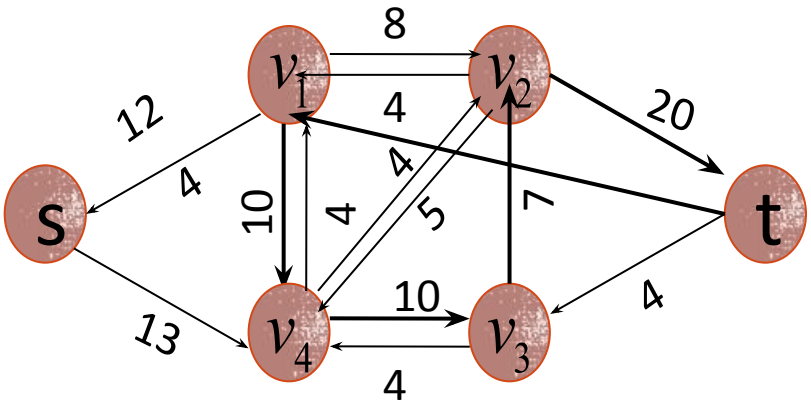


(б)



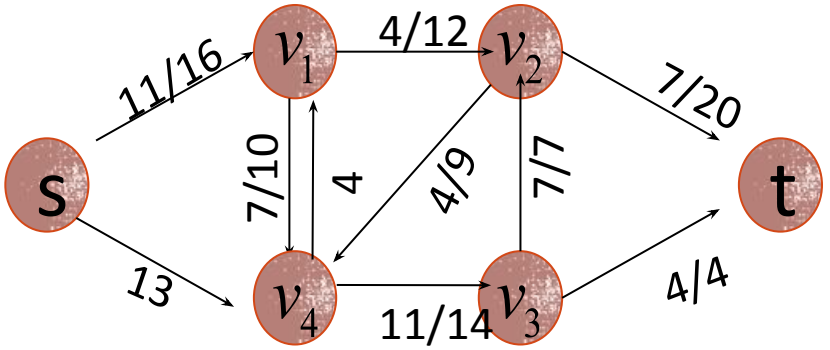
$f_p = 4 + 7 = 11$

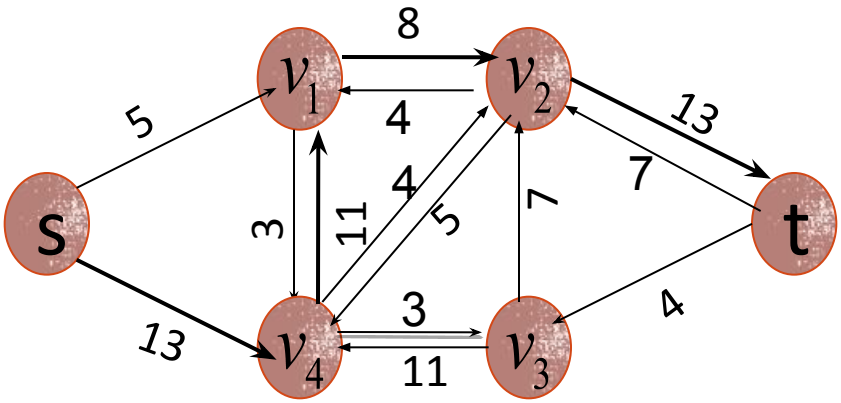
Остаточная сеть и дополняющий путь



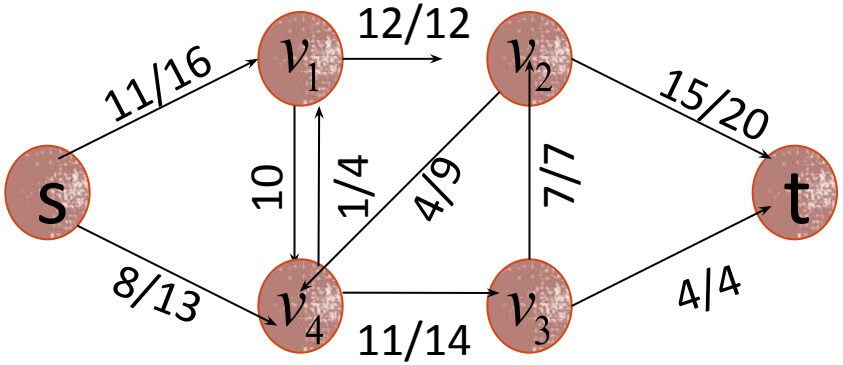
(б)

Увеличенный поток



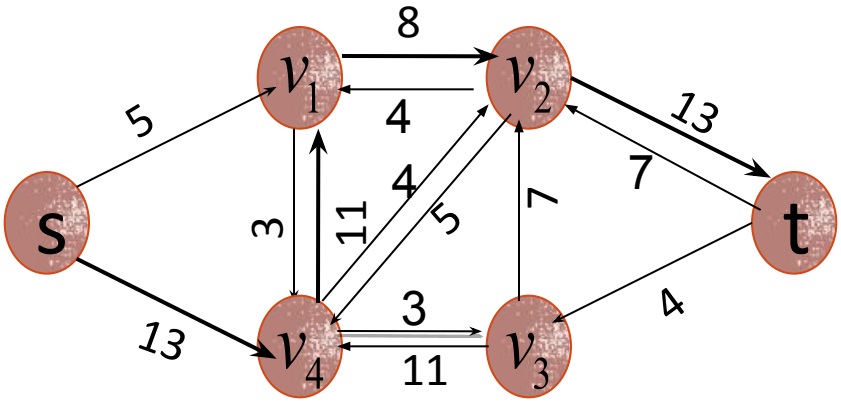


(в)



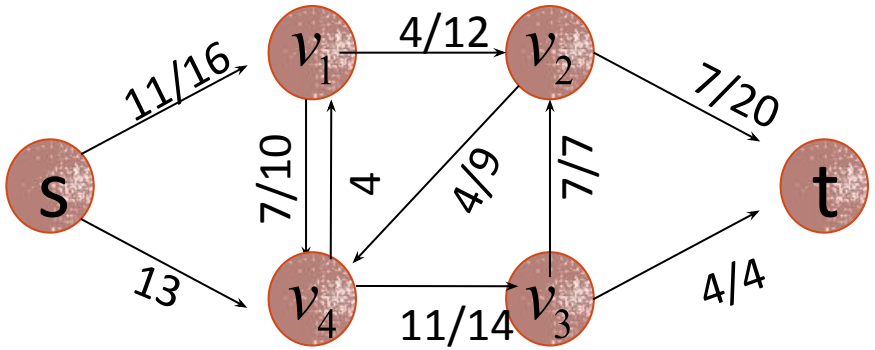
$f_p = 4 + 7 + 8 = 19$

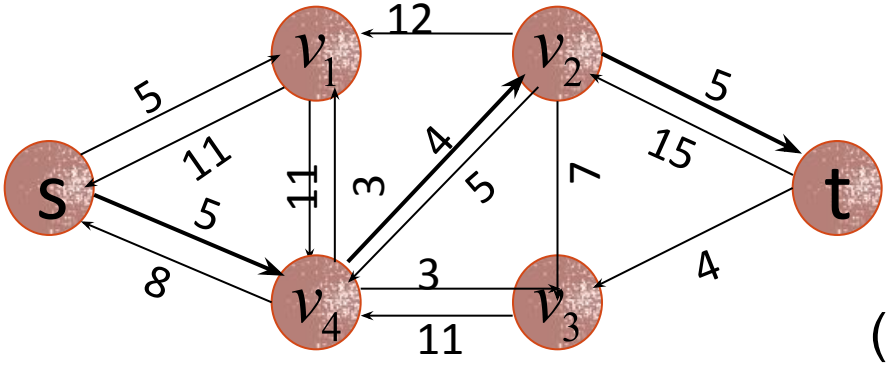
Остаточная сеть и дополняющий путь



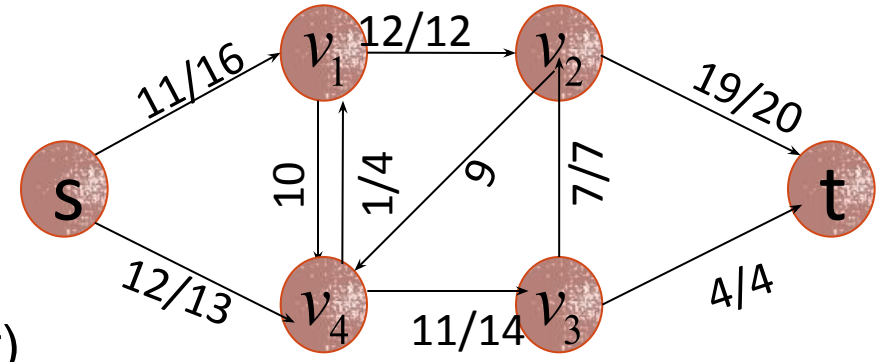
(в)

Увеличенный поток



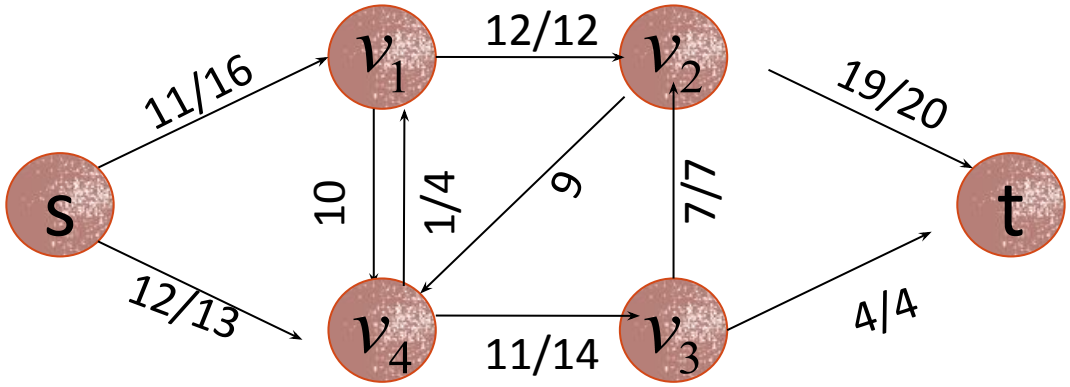


(г)



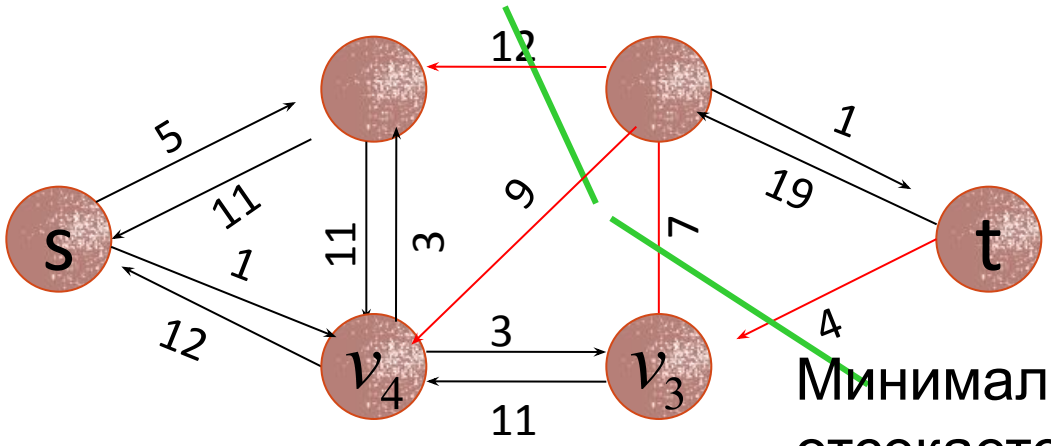
$f_p = 4 + 7 + 8 + 4 = 23$

Увеличенный поток



(г)

Остаточная сеть



(д)

Минимальный разрез: отсекается та часть вершин, до которых есть путь из s.

ЗАДАЧА О МАКСИМАЛЬНОМ ПАРОСОЧЕТАНИИ В ДВУДОЛЬНОМ ГРАФЕ

Пусть $G = (V, E)$ — неориентированный граф.

Паросочетанием назовем множество рёбер $M \subset E$, не имеющих общих концов (каждая вершина $v \in V$ является концом максимум одного ребра из M).

Будем говорить, что вершина $v \in V$ *входит* в паросочетание M, если в M есть ребро с концом v;

в противном случае v *свободна*.

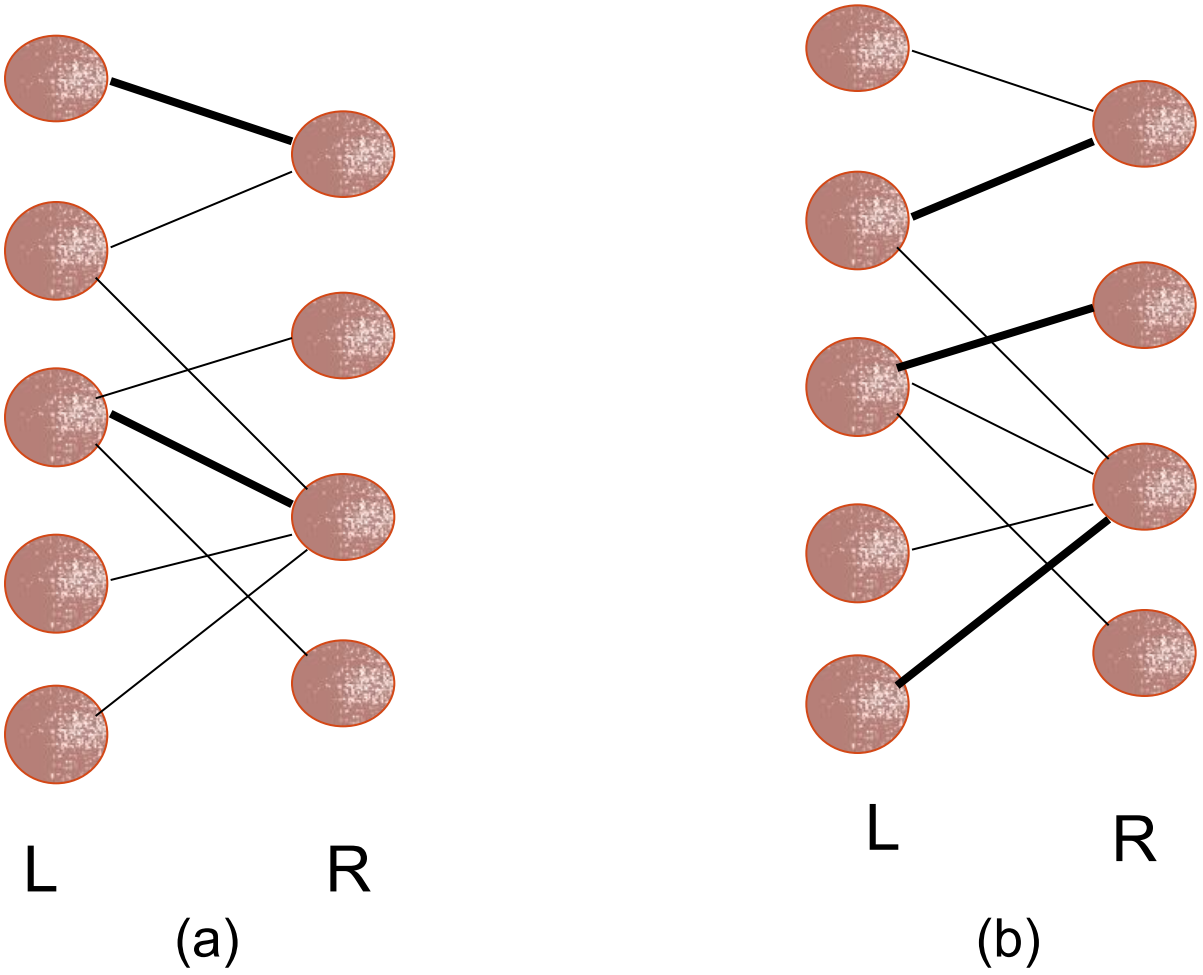
Максимальное паросочетание — это паросочетание M, содержащее максимально возможное число рёбер ($|M| \geq |M'|$ для любого паросочетания M').

ДВУДОЛЬНЫЙ ГРАФ

Рассмотрим паросочетания в двудольных графах. Множество V разбито на два непересекающихся подмножества L и R, и любое ребро из E соединяет некоторую вершину из L с некоторой вершиной из R.

Например, L — женихи, R — невесты, наличие ребра (u, v) означает, что u и v согласны стать супругами. Максимальное паросочетание доставляет ЗАГСу больше всего работы.

ПРИМЕР



Множество вершин разбито на две части *L* и *R*. (a) Паросочетание из двух рёбер. (b) Максимальное паросочетание состоит из трёх ребер.

ПОИСК МАКСИМАЛЬНОГО ПАРОСОЧЕТАНИЯ

Будем использовать метод Форда-Фалкерсона для поиска максимального паросочетания в двудольном графе $G = (V, E)$ за полиномиальное от $|V|$ и $|E|$ время.

Для этого рассмотрим сеть $G' = (V', E')$, построенную следующим образом: добавляются две новые вершины: исток (s) и сток (t): $V' = V \cup \{s, t\}$.

$E' = \{(s, u) : u \in L\} \cup \{(u, v) : u \in L, v \in R, (u, v) \in E\} \cup \{(v, t) : v \in R\}$

(L и R обозначаются доли графа, ребра направленные).

Будем считать, что пропускная способность каждого ребра равна единице.

Поток f в сети $G = (V, E)$ называется *целочисленным* , если все значения $f(u, v)$ — целые.

Лемма 8

Пусть $G = (V, E)$ — двудольный граф с долями L и R, и $G' = (V, E')$ — соответствующая сеть. Пусть M — паросочетание в G.

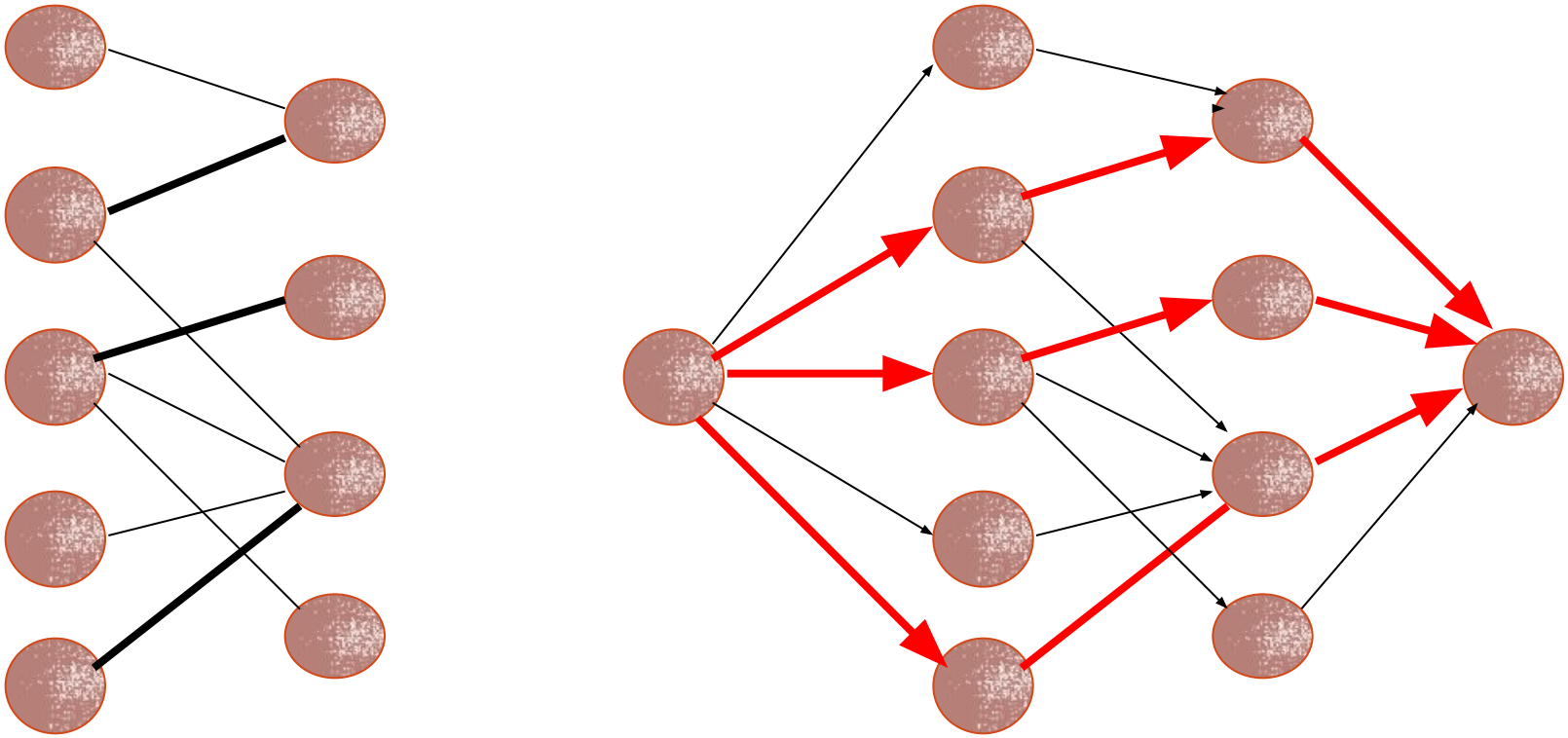
Тогда существует целочисленный поток в G' со значением $|f| = |M|$. Обратно, если f — целочисленный поток в G', то в G' найдется паросочетание из $|f|$ элементов.

Теорема 9 о целочисленном потоке

Если пропускные способности всех рёбер — целые числа, то максимальный поток, найденный алгоритмом Форда-Фалкерсона, будет целочисленным.

Следствие 10 Число рёбер в максимальном паросочетания M в двудольном графе G равно значению максимального потока в сети G'.

ПРИМЕР



Соответствующая сеть *G′* и максимальный поток в ней. Пропускная способность любого ребра равна единице; поток по выделенным рёбрам равен единице, по остальным — нулю. Выделенные рёбра, соединяющие вершины из *L* с вершинами из *R*, соответствуют максимальному паросочетанию в двудольном графе.

АНАЛИЗ

Таким образом, чтобы найти максимальное паросочетание в двудольном графе G, нам достаточно применить метод Форда-Фалкерсона и найти максимальный поток в соответствующей сети G'.

Оценка времени работы такого алгоритма.

Никакое паросочетание в двудольном графе не может содержать более $\min(L,R) = O(V)$ рёбер, поэтому значение максимального потока в G' равно $O(V)$.

Следовательно, время работы алгоритма Форда-Фалкерсона равно $O(VE)$.

АЛГОРИТМ ПРОТАЛКИВАНИЯ ПРЕДПОТОКА

Не просматриваем всю всю остаточную сеть на каждом шаге, а действуем локально в окрестности одной вершины. Кроме того, не требуем, выполнения закона сохранения потока в процессе работы алгоритма.

Определим *предпоток* как функцию $f : V \times V \longrightarrow R$, которая кососимметрична, удовлетворяет ограничениям, связанным с пропускными способностями, а также ослабленному закону сохранения: $f(V, u) \geq 0$ для всех вершин $u \in V \setminus \{s\}$.

Таким образом, в каждой вершине u (кроме истока) есть некоторый неотрицательный *избыток* $e(u) = f(V,u)$.

Вершину, отличную от s, с положительным избытком назовём *переполненной.*

МОТИВИРОВКА

В алгоритмах проталкивания предпотока избыток, например, жидкости в каждой вершине сливается. Важную роль играет целочисленный параметр — *высота* вершины.

Мы будем воображать, что в процессе работы алгоритма вершина может подниматься вверх. Высота вершины определяет, куда мы стараемся направить избыток жидкости.

Высота истока всегда равна $|V|$, а стока — нулю. Все остальные вершины изначально находятся на высоте 0, и со временем поднимаются.

Для начала мы отправляем из истока вниз столько жидкости, сколько нам позволяют пропускные способности выходящих из истока труб (это количество равно пропускной способности разреза $(s, V \setminus s)$).

Возникающий в соседних с истоком вершинах избыток жидкости сначала просто выливается, но затем будет направлен дальше.

Рассматривая какую-либо вершину *и* в ходе работы алгоритма, мы можем обнаружить, что в ней есть избыток жидкости, но все трубы, по которым ещё можно отправить жидкость из *и* куда-то (в *ненасыщенные* трубы) ведут в вершины той же или большей высоты. В этом случае мы можем выполнить операцию, называемую «*подъёмом*» вершины *и.*

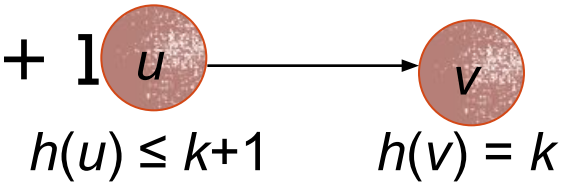
После этого вершина *и* становится на единицу выше самого низкого из тех её соседей, в которого ведёт ненасыщенная труба, т.е., мы поднимаем *и* ровно настолько, чтобы появилась ненасыщенная труба, ведущая вниз.

В конце концов мы добьёмся того, что в сток приходит максимально возможное количество жидкости. При этом предпоток может ещё не быть потоком (избыток жидкости сливается). Продолжая подъём вершин, которые могут стать выше истока, мы постепенно отправим избыток обратно в исток, что означает сокращение потока жидкости от истока, — и превратим предпоток в поток, который окажется максимальным.

ВЫСОТНАЯ ФУНКЦИЯ

Пусть $G = (V, E)$ — сеть с истоком s и стоком t, а f — предпоток в G. Функция $h : V \to \mathbb{N}$ называется *высотной* функцией для предпотока f, если:

$h(s) = |V|, h(t) = 0, h(u) \leq h(v) + 1$ для любого ребра $(u, v) \in E_f$.



Лемма 11

Пусть f — предпоток в сети $G = (V, E)$ и h — высотная функция. Тогда если для вершин $u, v \in V$, выполнено $h(u) > h(v) + 1$, то остаточная сеть не содержит ребра (u, v) («по круто идущим вниз трубам идёт максимально возможный поток».)

ОПЕРАЦИЯ ПРОТАЛКИВАНИЯ PUSH (*И, V*)

применима, если:

1) вершина u переполнена: $e(u) > 0$;
2) ребро $(и, v)$ не насыщено: $c_f(и, v) > 0$;
3) $h(u) = h(v) + 1$.

Push(u,v) {

$d_f(u,v) = \min(e(u), c_f(u,v))$;

$f[u,v] = f[u,v] + d_f(u,v)$;

$f[v,u] = -f[u,v]$

$e[u] = e[u] - d_f(u,v)$;

$e[v] = e[v] + d_f(u,v)$;

}

Условие $h(u) = h(v) + 1$ гарантирует, что мы направляем дополнительный поток лишь по рёбрам, идущим вниз с единичной разницей высот.

Проталкивание называется *насыщающим*, если в результате ребро (u,v) становится *насыщенным*, то есть если $c_f(u,v)$ обращается в нуль (ребро исчезает из остаточной сети);
в противном случае проталкивание считают *ненасыщающим.*

Если есть сосед, который на единицу ниже, то можно выполнить
проталкивание, но нельзя выполнить подъём.
Если все соседи не ниже, то проталкивание выполнить нельзя,
а подъём — можно, после чего возможно проталкивание.

ОПЕРАЦИЯ ПОДНЯТИЯ ВЕРШИНЫ LIFT (*U*)

применима, если:

1) вершина u переполнена: $e(u) > 0$;

2) для любого ребра $(u,v) \in E_f$ выполнено: $h(u) \le h(v)$.

Lift (и) поднимает переполненную вершину *и* на максимальную

высоту, которая допустима по определению высотной функции:

высота вершины превосходит высоту соседа в остаточной сети

не более чем на 1.

Lift(u) {

$h[u] = 1 + \min\{ h[v]: (u,v) \in E_f\}$;

}

Заметим, если вершина *и* переполнена, то в E_f найдётся по крайней мере одно ребро, выходящее из *и*:

так как $f[V, и] = e[и] > 0$, поэтому существует по крайней мере одна такая вершина v, для которой $f(v, и) > 0$.

$\Rightarrow c_f(u, v) = c(u, v) - f[u, v] = c(u, v) + f[v, u] > 0$, что означает, что $(и, v) \in E_f$.

ОБЩАЯ СХЕМА АЛГОРИТМА ПРОТАЛКИВАНИЯ ПРЕДПОТОКА

Начальный предпоток:

$$f_p(u,v) = \begin{cases} C(u,v), & \text{если } u=s \\ -C(v,u), & \text{если } u=s \\ 0, & \text{в остальных случаях.} \end{cases}$$

```
Initialize(G, s) {
    for (для каждой вершины u ∈ V) {
     h[u] = 0; e[u] = 0;
  }
    for (для каждого ребра (u,v) ∈ E) {
         f[u,v] = 0; f[v,u] = 0; h[s] = |V|;
  }
   for (для каждой вершины u ∈ Adj [s]) {
        f[s,u] = c(s,u); f[u,s] = -c(s,u); e[u] = c(s,u);
  }
}
```

ПРОДОЛЖЕНИЕ СХЕМЫ АЛГОРИТМА

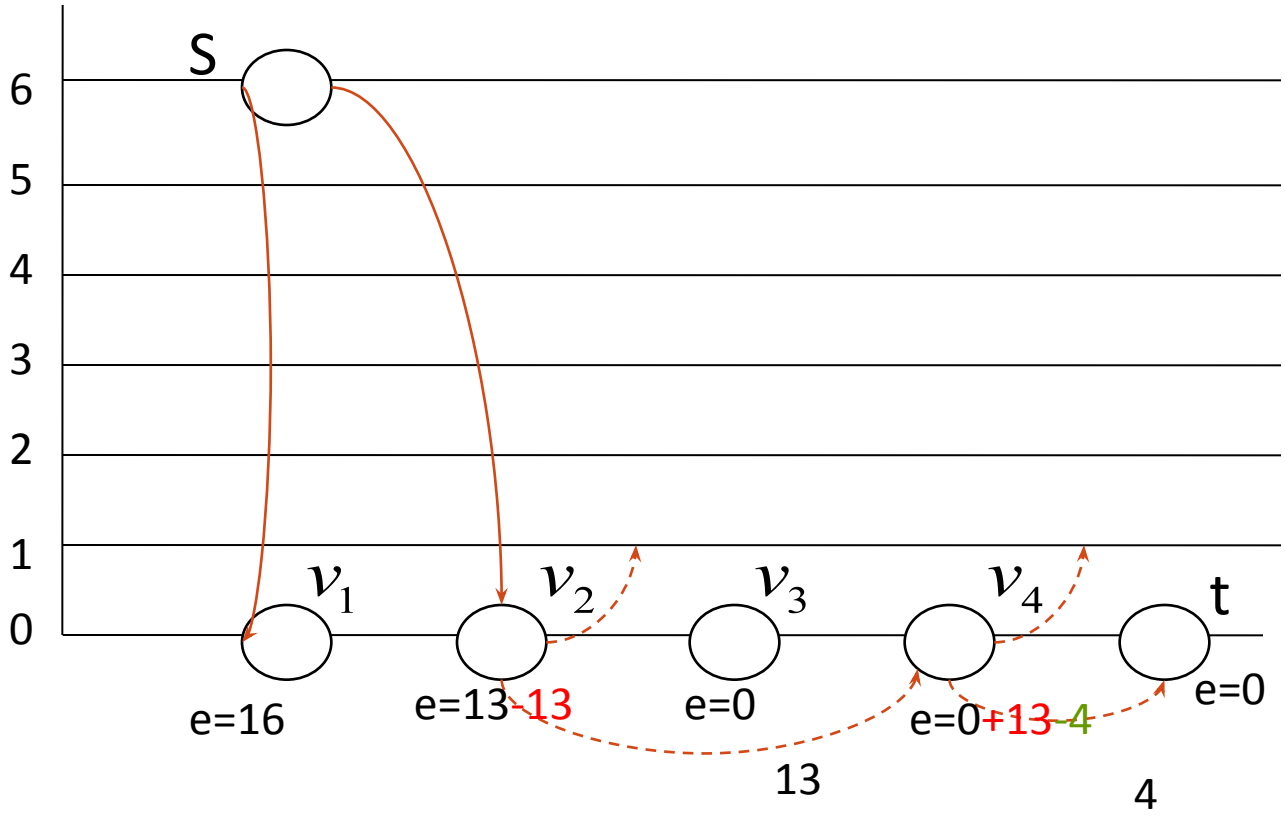
Поток по каждому ребру, выходящему из истока *s*, становится равным пропускной способности этого ребра. По остальным рёбрам поток равен 0.

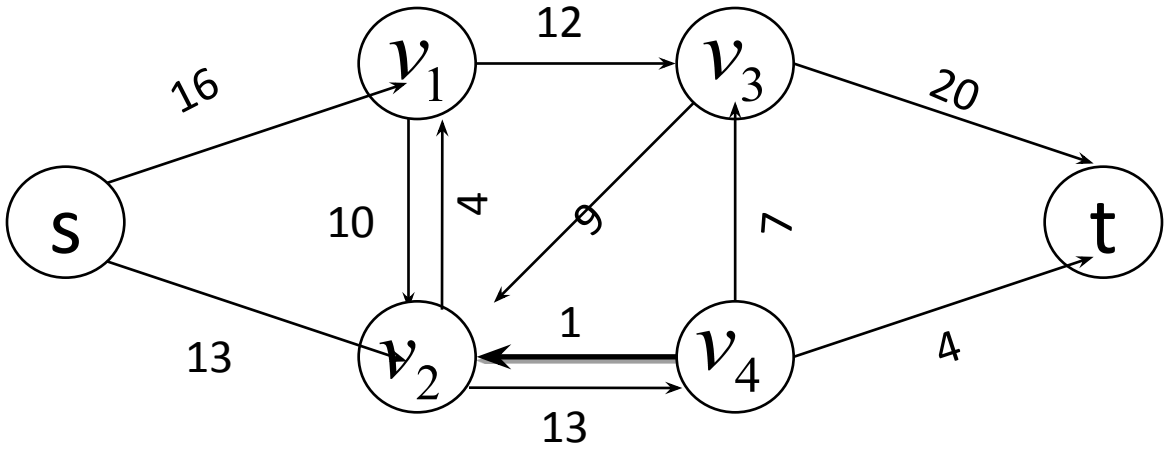
В каждой смежной с истоком вершине *v* появляется избыток $e[v] = c(s, v)$.

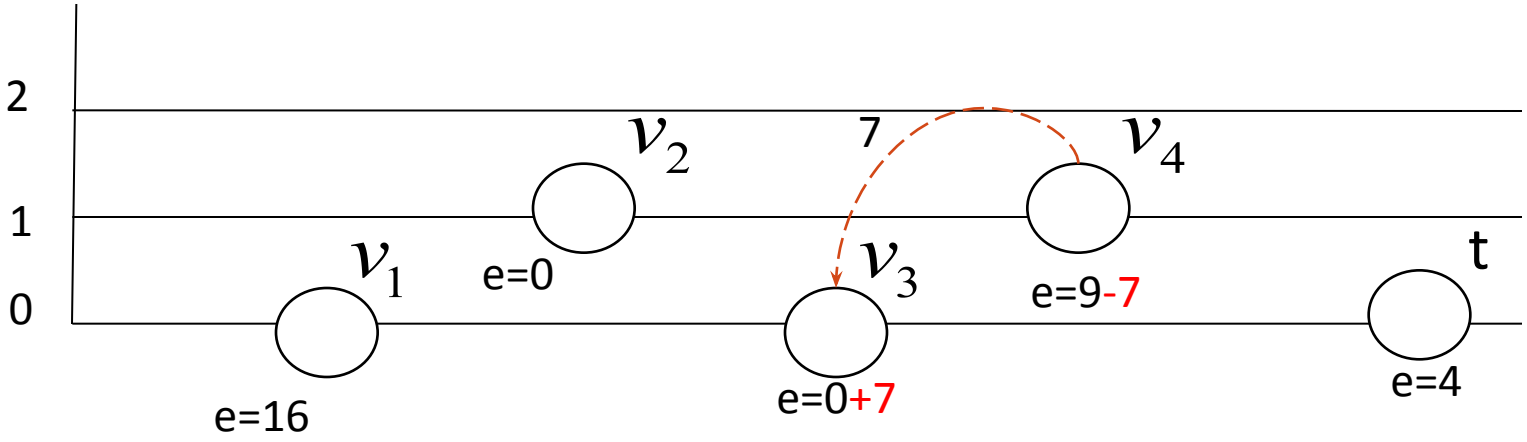
h – высотная функция, так как рёбра (*и, v*), для которых

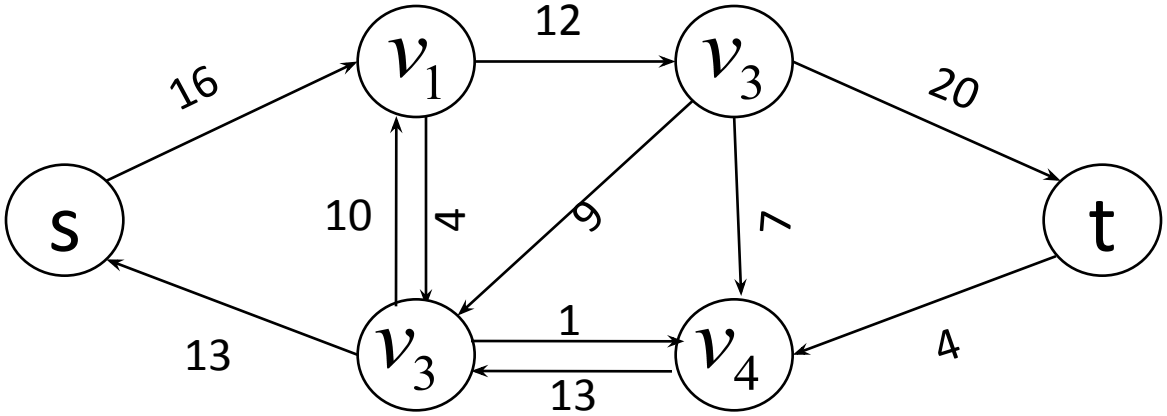
$h[u] > h[v] + 1$, выходят только из истока, но они насыщены и их нет в остаточной сети.

```
Generic-Preflow-Push {
    Initialize(G,s);
    while (возможны операции подъема или проталкивания)
            выполнить одну из этих операций;
}
```


S
6
5
4
3
2
1
0
v_1
v_2
v_3
v_4
t
e=16
e=13-13
e=0
e=0+13-4
e=0
13
4


v_1
v_3
S
v_2
v_4
t
16
12
20
10
4
9
7
1
13
4
13


2
1
0
v_2
7
v_4
v_1
e=0
v_3
e=9-7
t
e=16
e=0+7
e=4


v_1
12
v_3
16
20
s
10
4
9
7
t
13
v_3
1
v_4
4
13

АНАЛИЗ

Лемма 12 Пусть f — предпоток в сети $G = (V,E)$. Пусть h — высотная функция для f и вершина $и$ переполнена. Тогда в $и$ возможно либо проталкивание, либо подъём.

Доказательство

Поскольку h — высотная функция, то $h(u) \leq h(y) + 1$ для любого остаточного ребра (u,v). Если в $и$ невозможно проталкивание, то для всех остаточных рёбер $(и, v)$ выполнено неравенство

$h(u) < h(v) + 1$, из чего следует, что $h(u) \leq h(v)$, и в вершине $и$ возможен подъём.

Корректность метода. 1) Если алгоритм остановится, то предпоток f в этот момент будет максимальным потоком. 2) Алгоритм действительно остановится.

Лемма 13 ри исполнении программы Generic-Preflow-Push высота $h[u]$ любой вершины $и \in V$ может только возрастать.

Лемма 14 Во время выполнения программы Generic-Preflow-Push функция h остаётся высотной функцией.

Лемма 15 Пусть $G = (V, E)$ — сеть с истоком s и стоком t. Пусть f— предпоток в G, а h — высотная для f. Тогда в остаточной сети G_f не существует пути из истока в сток.

Теорема Если программа Generic-Preflow-Push, применённая к сети $G = (V, E)$ с истоком s и стоком t останавливается, то получающийся предпоток f, будет максимальным потоком для G.

Пояснение. В момент остановки переполненных вершин в сети нет. Значит, в этот момент предпоток является потоком. Функция h будет высотной функцией, и потому в остаточной сети G_f нет пути из s в t. По теореме о максимальном потоке и минимальном разрезе поток f максимален.

Лемма 16 Пусть $G = (V, E)$ — сеть с истоком s и стоком t, а f — предпоток в G. Тогда для любой переполненной вершины u найдется простой путь из u в s в остаточной сети G_f.

Лемма 17 При исполнении программы Generic-Preflow-Push высота любой вершины $v \in V$ не превосходит $2|V| - 1$.

Следствие При исполнении программы Generic-Preflow-Push общее число операций подъёма не превосходит $2|V|^2$.

Лемма 18 При исполнении программы Generic-Preflow-Push количество насыщающих проталкиваний не превосходит $2|V||E|$.

Теорема

Общее число операций подъёма и проталкивания при исполнении программы Generic-Preflow-Push на сети $G = (V, E)$ равно $O(V^2E)$.

ЗАДАЧА О ВЫХОДЕ

Имеется, граф вершины которого образуют решётку из n строк и n столбцов. Обозначим через (i, j) вершину на пересечении i-го столбца и j-ой строки. Все вершины такой решётки, не считая граничных имеют четырёх соседей.

Требуется выяснить, существуют ли τ попарно непересекающихся (не имеющих общих вершин) путей от данных $\tau \leq n^2$ начальных точек решётки (x_1,y_1), (x_2,y_2), … , (x_τ,y_τ) к τ различным граничным точкам.

ПОДСКАЗКА К РЕШЕНИЮ

Рассмотрим сеть, в которой каждая внутренняя вершина имеет дубль. Все инцидентные ребра каждой внутренней вершины сделайте входящими. Добавьте выходящую дугу из каждой внутренней вершины в вершину-дубль. Все инцидентные ребра каждой внутренней вершины сделайте выходящими из вершины-дубля.

Покажите, что задача о максимальном потоке в такой сети сводится к обычной задаче о максимальном потоке для построенной таким образом сети.

МИНИМАЛЬНОЕ ПОКРЫТИЕ ПУТЯМИ

Покрытием путями ориентированного графа $G = (V, E)$ назовется множество путей P с таким свойством: каждая вершина из V принадлежит ровно одному пути из P. Пути могут начинаться и заканчиваться где угодно, и иметь любую длину, в том числе нулевую. Покрытие наименьшим возможным числом путей называется *минимальным покрытием путями*.

Решение

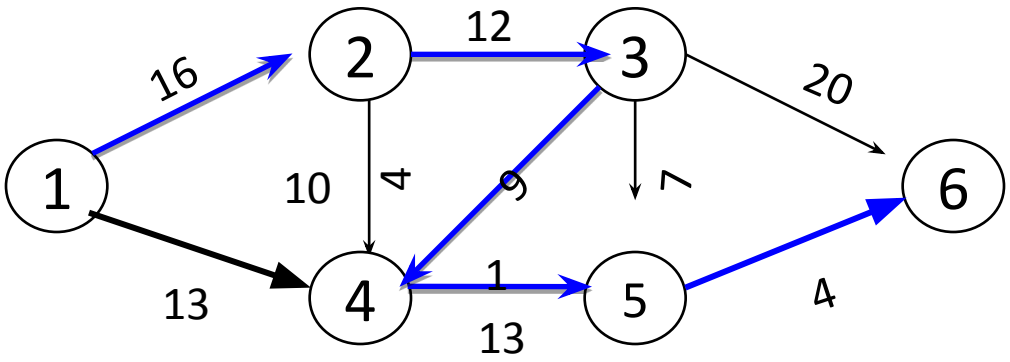
Построим граф $G' = (V', E')$, где

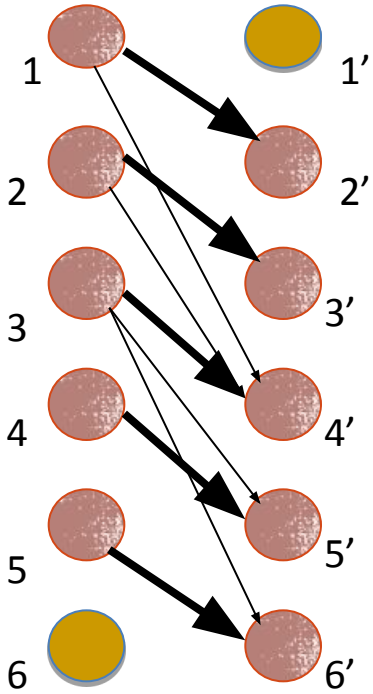
$V' = \{x_0, x_1, \ldots, x_n\} \cup \{y_0, y_1, \ldots, y_n\}$,

$E' = \{(x_0, x_i): x_i \in V\} \cup \{(y_i, y_0): y_i \in V\} \cup \{(x_i, y_j\}: (i, j) \in E\}$

Решим для этого графа задачу о максимальном потоке.

ПРИМЕР





Также нужно достроить исток и сток.

Исток соединить с левой долей, сток – с правой.