Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

# Сравнение некоторых алгоритмов поиска максимального потока

КУРСОВАЯ РАБОТА

студента 4 курса 431 группы
специальности 10.05.01 «Компьютерная безопасность»
факультета компьютерных наук и информационных технологий
Енца Михаила Владимировича

Научный руководитель
ассистент _____ Е.Н.Новокшонова
подпись, дата

Заведующий кафедрой
д.ф.-м.н., доцент _____ М. Б. Абросимов
подпись, дата

Саратов 2018

**СОДЕРЖАНИЕ**

# ВВЕДЕНИЕ

Модель транспортной сети и нахождение максимального потока имеет широкое применение во многих областях науки и жизни. Например, логистические схемы транспортных компаний, в науке – множество теоретических задач, таких как поиск минимального разреза, поиск максимального паросочетания и другие.

Конечно, при использовании алгоритмов важно, чтобы их скорость была высокой, поэтому для различных задач, одной природы, порой применяются различные алгоритмы, оптимизированные под определенные входные данные.

Задача о максимальном потоке является одной из прикладных задач в теории графов и оптимизации.

В разделе «Потоки на графах» даются основные определения транспортной сети, а также их основные свойства.

В разделе «Алгоритмы поиска максимального потока» рассматриваются две известные концепции поиска максимального потока, а также 4 алгоритма построенные на этих концепциях.

В разделе «Сравнение некоторых алгоритмов поиска максимального потока» показаны результаты сравнения двух алгоритмов поиска максимального потока на графах.

В этой работе я разберу основные теоретические и практические идеи транспортных сетей, а также сравню время работы двух известных алгоритмов. [тавтология с предыдущим абзацем]

## ОПРЕДЕЛЕНИЯ И ОБОЗНАЧЕНИЯ

*Ориентированный граф* (или *орграф*) – пара $\vec{G} = (V, \alpha)$, где $V$ – конечное непустое множество (вершины графа), а $\alpha \subseteq V \times V$ – отношение на множестве $V$. Пара $(u, v) \in \alpha$ называется *дугой* орграфа с началом $u$ концом $v$, где $u, v \in V$.

Две вершины $u$, $v$ называются *смежными*, если они соединены дугой, то есть $\{u, v\} \in \alpha$. [2].

Последовательность вида $v_0\{v_0, v_1\}v_1\{v_1, v_2\}v_2 \dots v_{l-1}\{v_{l-1}, v_l\}v_l$, где $v_0, v_1, v_2, \dots, v_{l-1}, v_l \in V$, а $\{v_0, v_1\}, \{v_1 v_2\}, \dots, \{v_{l-1} v_l\} \in \alpha$, называется *маршрутом* (или *путём*) длины $l$ из вершины $v_0$ (начало маршрута) в вершину $v_l$ (конец маршрута).

Путь, каждая вершина которого принадлежит не более чем двум его дугам, является *простым*.

Говорят, что вершина $v_l$ *достижима* из вершины $v_0$, если существует маршрут с началом в $v_0$ и концом в $v_l$.[2].

В задаче о кратчайшем пути в ориентированном графе дается взвешенный ориентированный граф. К дуге приписан вещественный вес $w(u, v)$. Весом пути называется сумма весов дуг, входящих в этот путь. Весом кратчайшего пути из вершины $u$ в $v$ называется минимальный из весов путей $u$ в $v$, а если таких путей нет, то вес кратчайшего пути считается равным $\infty$.

*Кратчайшим путем* из u в v называется всякий путь из u в v, вес которого равен весу кратчайшего пути из u в v [4].

*Петлёй* называется дуга с совпадающим началом и концом [2].

## 1 Потоки на графах

*Транспортная сеть* $\vec{G} = (V, E)$ представляет собой ориентированный граф, в котором каждая дуга $(u, v) \in E$ имеет неотрицательную пропускную способность $c(u, v) > 0$. Если $(u, v) \notin E$, предполагается, что $c(u, v) = 0$. В транспортной сети выделяются две вершины: источник $s$ и сток $t$. Для удобства предполагается, что каждая вершина лежит на неком пути из источника к стоку, т.е. для любой вершины $v \in V$ существует путь $s \to v \to t$. Таким образом, граф является связным и $|E| > |V| - 1$.

Пусть $\vec{G} = (V, E)$ – транспортная сеть с функцией пропускной способности $c$. Пусть $s$ – источник, а $t$ – сток. Потоком в $\vec{G}$ является действительная функция $f: V\ x\ V \to R$, удовлетворяющая следующим трем условиям.

- Ограничение пропускной способности: $f(u, v) \leq c(u, v)$ для всех $u, v \in V$.
- Антисимметричность: $f(u, v) = -f(u, v)$ для всех $u, v \in V$.
- Сохранение потока: для всех $u \in V - \{s, t\}$

$$\sum_{v \in V} f(u, v) = 0$$

Количество $f(u, v)$, которое может быть положительным, нулевым или отрицательным, называется *потоком* из вершины $u$ в вершину $v$. Величина потока $f$ определяется как

$$|f| = \sum_{v \in V} f(s, v)$$

Т.е. как суммарный поток, выходящий из источника. В задаче о максимальном потоке дана некоторая транспортная сеть $\vec{G}$ с источником $s$ и стоком $t$, и необходимо найти поток максимальной величины.

Ограничение пропускной способности предполагает, чтобы поток из одной вершины в другую не превышал заданную пропускную способность дуги. Антисимметричность введена для удобства обозначения и заключается в том, что поток из вершины $u$ в вершину $v$ противоположен потоку в обратном направлении. Свойство сохранения потока утверждает, что суммарный поток, выходящий из вершины, не являющийся источником или стоком, равен нулю. Используя антисимметричность, можно записать свойство сохранения потока как

$$\sum_{u \in V} f(u, v) = 0$$

Для всех $v \in V - \{s, t\}$, т.е. суммарный поток, входящий в вершину, отличную от источника и стока равен 0.

Если в $E$ не присутствуют ни $(u, v)$, ни $(v, u)$, между вершинами $u$ и $v$ нет потока, и $f(u, v) = f(v, u) = 0$.

Суммарный положительный поток, входящий в вершину $v$, задается выражением

$$\sum_{\substack{u \in V \\ f(u,v)>0}} f(u, v)$$

Суммарный положительный поток, выходящий из некоторой вершины, определяется симметрично. Суммарный чистый поток в некоторой вершине равен разности суммарного положительного потока, выходящего из данной вершины, и суммарного положительного потока, входящего в нее. Одна из интерпретаций свойства сохранения потока состоит в том, что для отличной от источника и стока вершины, входящей в нее суммарный положительный поток должен быть равен выходящему суммарному положительному потоку. Свойство, что суммарный чистый поток в транзитной вершине должен быть равен 0, часто нестрого формулируют как «входящий поток равен выходящему потоку».

## 2 Алгоритмы поиска максимального потока

Алгоритмов поиска максимального потока много, но большинство являются модификацией алгоритмов, которые представлены ниже.

### 2.1 Алгоритм Форда-Фалкерсона

Алгоритм Форда-Фалкерсона решает задачу поиска максимального потока. Алгоритм является итеративным. Вначале величине потока присваивается значение 0: $f(u,v) = 0$ для всех $u, v \in V$. На каждой итерации величина потока увеличивается посредством «увеличивающего пути» и последующего увеличения потока. Этот процесс повторяется до тех пор, пока уже невозможно отыскать увеличивающий путь.

*Остаточная сеть* – это сеть, состоящая из дуг, допускающих увеличение потока. Пусть задана транспортная сеть $\vec{G} = (V, E)$ с источником $s$ и стоком $t$. Пусть $f$ – некоторый поток в $\vec{G}$. Рассмотрим пару вершин $u, v \in V$. Величина дополнительного потока, который можно направить из $u$ в $v$, не превысив пропускную способность $c(u,v)$, и задается формулой:

$$c_f(u,v) = c(u,v) - f(u,v) \quad (1)$$

Когда поток $f(u,v)$ отрицателен, остаточная пропускная способность $c_f(u,v)$ больше, чем пропускная способность $c(u,v)$.

Для заданной транспортной сети $\vec{G} = (V, E)$ и потока $f$, остаточной сетью в $\vec{G}$, порожденной потоком $f$, является сеть $\vec{G}_f = (V, E_f)$, где $E_f = \{(u,v) \in VxV : c_f(u,v) > 0\}$.

Таким образом по каждой дуге остаточной сети, или остаточной дуге, можно направить поток, больший 0.

Дугами $E_f$ являются или дуги $E$, или обратные им. Если $f(u,v) < c(u,v)$ для некоторой дуги $(u,v) \in E$, то $c_f(u,v) = c(u,v) - f(u,v) > 0$ и $(u,v) \in E_f$.

Если $f(u,v) > 0$ для некоторой дуги $(u,v) \in E$, то $f(v,u) < 0$. В таком случае $c_f(v,u) = c(v,u) - f(v,u) > 0$ и, следовательно, $(v,u) \in E_f$. Если в исходной сети нет ни дуги $(u,v)$, ни $(v,u)$, то $c(u,v) = c(v,u) = 0, f(u,v) = f(v,u) = 0$, и $c_f(u,v) = c_f(v,u) = 0$. Таким образом, можно сделать вывод, что дуга $(u,v)$ может оказаться в остаточной сети только в том случае, если хотя бы одна из дуг $(u,v)$ или $(v,u)$ присутствует в исходной транспортной сети, поэтому $|E_f| \leq 2|E|$.

Остаточная сеть $\vec{G}_f$ является транспортной сетью со значениями пропускных способностей, заданными $c_f$.

Следующая лемма показывает, как поток в остаточной сети связан с потоком в исходной транспортной сети.

Лемма 1. Пусть $\vec{G} = (V,E)$ – транспортная сеть с источником $s$ и стоком $t$, а $f$ – поток в $\vec{G}$. Пусть $\vec{G}_f$ – остаточная сеть в $\vec{G}$, построенная потоком $f$, а $f'$ – поток в $\vec{G}_f$. Тогда сумма потоков $f + f'$, является потоком в $\vec{G}$, и величина этого потока равна $|f + f'| = |f| + |f'|$.

Доказательство. Необходимо проверить, выполняется ли ограничения антисимметричности, пропускной способности и сохранения потока. Для подтверждения антисимметричности заметим, что для всех $u, v \in V$, справедливо $(f + f')(u,v) = f(u,v) + f'(u,v) = -f(v,u) - f'(v,u) = -\big(f(v,u) + f'(v,u)\big) = -(f + f')(v,u)$.

Покажем соблюдение ограничения пропускной способности. Заметим, что $f'(u,v) \leq c_f(u,v)$ для всех $u, v \in V$. Поэтому $(f + f')(u,v) = f(u,v) + f'(u,v) \leq f(u,v) + \big(c(u,v) - f(u,v)\big) = c(u,v)$.

Покажем соблюдение сохранения потока, заметим, что для всех $u \in V - \{s,t\}$ справедливо равенство

$$\sum_{v\in V}(f+f')(u,v)$$

$$=\sum_{v\in V}\big(f(u,v)+f'(u,v)\big)=\sum_{v\in V}f(u,v)+\sum_{v\in V}f'(u,v)=0+0=0$$

И наконец,

$$|f+f'|=\sum_{v\in V}(f+f')(s,v)$$

$$=\sum_{v\in V}\big(f(s,v)+f'(s,v)\big)=\sum_{v\in V}f(s,v)+\sum_{v\in V}f'(s,v)=|f|+|f'|$$

■

Для заданной транспортной сети $\vec{G}=(V,E)$ и потока $f$ увеличивающим путем $p$ является простой путь из $s$ в $t$ в остаточной сети $\vec{G}_f$.

Согласно определению остаточной сети, каждая дуга $(u,v)$ увеличивающего пути допускает некоторый дополнительный положительный поток из $u$ в $v$ без нарушения ограничения пропускной способности для данной дуги.

Максимальная величина, на которую можно увеличить поток вдоль каждой дуги увеличивающего пути $p$, называется *остаточной пропускной способностью* $p$ и задается формулой

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

Лемма 2. Пусть $\mathrm{G}=(\mathrm{V},\mathrm{E})$ – транспортная сеть, а $f$ – некоторый поток в $\vec{G}$, и пусть $p$ – некоторый увеличивающий путь в $\vec{G}_f$. Определим функцию $f_p: VxV\to R$ следующим образом:

$$f_p(u,v)=\begin{cases}c_f(p),\text{если }(u,v)\in p,\\-c_f(p),\text{если }(v,u)\in p,\\0,\text{в противном случае.}\end{cases}$$

Тогда $f_p$ является потоком в $\vec{G}$ и его величина составляет $|f_p| = c_f(p) > 0$. ■

Следствие 3. Пусть $\vec{G} = (V, E)$ – транспортная сеть, а $f$ – некоторый поток в $\vec{G}$, и пусть $p$ – некоторый увеличивающий путь в $\vec{G}_f$. Пусть $f_p$ определен в соответствии с уравнением представленном в лемме 2. Определим функцию $f': VxV \to R$ как $f' = f + f_p$. Тогда $f'$ является потоком в $\vec{G}$ и имеет величину $|f'| = |f| + |f_p| > |f|$.

Доказательство. Непосредственно вытекает из лемм 1 и 2. ■

*Разрезом* $(S, T)$ транспортной сети $\vec{G} = (V, E)$ называется разбиение множества $S$ и $T = V - S$, такие что $s \in S$, а $t \in T$. Если $f$ – поток, то *чистый поток* через разрез $(S, T)$ по определению равен $f(S, T)$. Пропускной способностью разреза $(S, T)$ является $c(S, T)$. Минимальным разрезом сети является разрез, пропускная способность котрого среди всех разрезов сети минимальна.

Лемма 4. Пусть $f$ – некоторый поток в транспортной сети $\vec{G}$ с источником $s$ и стоком $t$, и пусть $(S, T)$ – разрез $\vec{G}$. Тогда чистый поток через $(S, T)$ равен $f(S, T) = |f|$.

Доказательство. Заметим, что согласно свойству сохранения потока $f(S - s, V) = 0$, так что

$$f(S,T) = f(S,V) - f(S,S) = f(S,V) = f(s,V) + f(S-s,V) = f(s,V) = |f|$$

■

Следствие 5. Величина любого потока $f$ в транспортной сети $\vec{G}$ не превышает пропускную способность произвольно разреза $\vec{G}$ [5].

Доказательство. Пусть $(S, T)$ – произвольный разрез $\vec{G}$, а $f$ – некоторый поток. Согласно лемме 4 и ограничениям пропускной способности,

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

■

Теорема 6 (О максимальном потоке и минимальном разрезе). Если $f$ – некоторый поток в транспортной сети $\vec{G} = (V,E)$ с источником $s$ и стоком $t$, то следующие утверждения эквивалентны.

1. $f$ – максимальный поток в $\vec{G}$.
2. Остаточная сеть $\vec{G}_f$ не содержит увеличивающих путей.
3. $|f| = c(S,T)$ для некоторого разреза $(S,T)$ сети $\vec{G}$.

Доказательство. (1) → (2): Предположим противное: пусть $f$ является максимальным потоком в $\vec{G}$, но $\vec{G}_f$ содержит увеличивающий путь $p$. Согласно следствию 3, сумма поток $f + f_p$ задается уравнением из леммы 2, является потоком в $\vec{G}$, величина которого строго больше, чем $|f|$, что противоречит предположению, что $f$ – максимальны поток.

(2) → (3): предположим, что $\vec{G}_f$ не содержит увеличивающего пути, т.е. $\vec{G}_f$ не содержит пути из $s$ в $t$. Определим

$$S = \{v \in V : \text{в } \vec{G}_f \text{ существует путь из } s \text{ в } v\}$$

И $T = V - S$. Разбиение $(S,T)$ является разрезом: очевидно, что $s \in S$, а $t \notin S$, поскольку в $\vec{G}_f$ не существует пути из $s$ в $t$. Для каждой пары вершин $u \in S, v \in T$ справедливо соотношение $f(u,v) = c(u,v)$, поскольку в противном случае $(u,v) \in E_f$ и $v$ следует поместить во множество $S$. Следовательно, согласно лемме 4. $|f| = f(S,T) = c(S,T)$.

(3) → (1): Согласно следствию 5, $|f| \leq c(S,T)$ для всех разрезов $(S,T)$, поэтому из условия $|f| = c(S,T)$ следует, что $f$ – максимальный поток.

При выполнении каждой итерации метода Форда-Фалкерсона находим некоторый увеличивающий путь $p$, и поток $f$ вдоль каждой дуги данного пути увеличивается на величину остаточной пропускной способности $c_f(p)$. Простая реализация алгоритма вычисляет максимальный поток в графе $\vec{G} = (V, E)$ путем обновления потока $f[u, v]$ между каждой парой вершин $u$ и $v$, соединенных дугой. Если вершины $u$ и $v$ не связаны дугой ни в одном направлении, неявно предполагается, что $f[u, v] = 0$. Предполагается, что значения пропускных способностей задаются вместе с графом и $c(u, v) = 0$, если $(u, v) \notin E$. Остаточная пропускная способность $c_f(u, v)$ вычисляется по формуле (1). В коде процедуры $c_f(p)$ в действительности является просто временной переменной, в которой хранится остаточная пропускная способность пути $p$.

Алгоритм Форда-Фалкерсона

Шаг 1 для каждой дуги $(u, v) \in E[\vec{G}]$: $f[v, u] = 0,\ f[u, v] = 0$

Шаг 2 пока существует путь $p$ из $s$ в $t$ в остаточной сети $\vec{G}_f$: выполнять шаги 3-5

Шаг 3 выбираем $c_f(p)$ как минимум из $c_f(u, v)$ и $(u, v)$, $(u, v) \in p$

Шаг 4 для каждой дуги $(u, v) \in p$: выполнять шаг 5

Шаг 5 $f[u, v] = f[u, v] + c_f(p), f[v, u] = -f[u, v]$

Строка 1 инициализирует поток $f$ значением 0. В строках 2-5 выполняется неоднократный поиск увеличивающего пути $p$ в $\vec{G}_f$, и поток $f$ вдоль пути $p$ увеличивается на остаточную пропускную способность $c_f(p)$. Когда увеличивающих путей больше нет, поток $f$ является максимальным.

Время выполнения алгоритма, указанного выше, зависит от того, как именно выполняется поиск увеличивающего пути $p$.

На практике задача поиска максимального потока чаще всего возникает в целочисленной постановке. Если пропускные способности – рациональные числа, можно использовать соответствующее масштабирование, которое сделает их целыми. Тогда реализация алгоритма имеет время работы $O(E|f^*|)$, где $f^*$ -- максимальны поток, найденный данным алгоритмом. Анализ производится следующим образом. Выполнение 1 строки занимает время $O(E)$. Цикл в строках 2-5 выполняется не более $|f^*|$ раз, поскольку величина потока за каждую итерацию увеличивается по крайней мере на одну единицу.

### 2.2. Алгоритм Диница

Если в качестве увеличивающего пути $p$ выбирается кратчайший путь из $s$ в $t$ в остаточной сети, где каждая дуга имеет единичную длину (вес). Такая реализация алгоритма Форда-Фалкерсона называется алгоритмом Диница. Докажем, что время выполнения алгоритма Диница составляет $O(VE^2)$.

Анализ зависит от расстояний между вершинами остаточной сети $G_f$. В следующей лемме длина кратчайшего пути из вершины $u$ в $v$ в остаточной сети $G_f$, где каждая дуга имеет единичную длину, обозначается как $\delta_f(u, v)$.

Лемма 7. Если для некоторой транспортной сети $\vec{G} = (V, E)$ с источником $s$ и стоком $t$ выполняется алгоритм Диница, то для всех вершин $v \in V - \{s, t\}$ длина кратчайшего пути $\delta_f(s, v)$ в остаточной сети $\vec{G}_f$ монотонно возрастает с каждым увеличением потока.

Доказательство. Предположим, что для некоторой вершины $v \in V - \{s, t\}$ существует такое увеличение потока, котрое приводит к уменьшению длины кратчайшего пути из $s$ в $v$, и покажем, что это предположение приводит к противоречию. Пусть $f$ – поток, который был непосредственно перед первым увеличением, приведшим к уменьшению длины некого кратчайшего пути, а $f'$ -- поток сразу после этого увеличения. Пусть $v$ – вершина с минимальной длиной кратчайшего пути $\delta'_f(s, v)$, которая уменьшилась в результате увеличения

потока, т.е. $\delta'_f(s,v) < \delta_f(s,v)$. Пусть $p = s \dashrightarrow u \longrightarrow v$ – кратчайший путь от $s$ к $v$ в $\vec{G}'_f$, такой что $(u,v) \in E'_f$ и

$$\delta'_f(s,u) = \delta'_f(s,v) - 1$$

Исходя из того, как мы выбирали $v$, можно утверждать, что длина пути до вершины $u$ не уменьшилась, т.е.

$$\delta'_f(s,u) \geq \delta_f(s,u).$$

Утверждаем, что в таком случае $(u,v) \notin E_f$. Если $(u,v) \in E_f$, тогда справедливо следующее:

$$\delta_f(s,v) \leq \delta_f(s,u) + 1 \leq \delta'_f(s,u) + 1 = \delta'_f(s,v)$$

Что противоречит предположению $\delta'_f(s,v) \leq \delta_f(s,v)$.

Рассмотрим, как может получиться, что $(u,v) \notin E_f$, но $(u,v) \in E'_f$. Увеличение должно привести к возрастанию потока из $v$ в $u$. Алгоритм Диница всегда увеличивает поток вдоль кратчайших путей, поэтому последней дугой кратчайшего пути из $s$ в u в $\vec{G}_f$ является дуга $(v,u)$. Следовательно, $\delta_f(s,v) = \delta_f(s,u) - 1 \leq \delta'_f(s,u) - 1 = \delta'_f(s,v) - 2$, что противоречит предположению $\delta'_f(s,v) < \delta_f(s,v)$, а значит, наше предположение о существовании вершины $v$ не верно. ■

Теорема 8. Если для некоторой транспортной сети $\vec{G} = (V,E)$ с источником $s$ и стоком $t$ выполняется алгоритм Диница, то общее число увеличений потока, выполняемое данным алгоритмом, составляет $O(VE)$.

Доказательство. Назовем дугу $(u,v)$ остаточной сети $\vec{G}_f$ критической для увеличивающего пути $p$, если остаточная пропускная способность $p$ равна остаточной пропускной способности дуги $(u,v)$, т.е. если $c_f(p) = c_f(u,v)$. После увеличения потока вдоль некоего увеличивающего пути, все критические дуги пути исчезают из остаточной сети. Кроме того, по крайней мере одна дуга

любого увеличивающего пути должна быть критической. Теперь покажем, что каждая дуга из $|E|$ дуг может становиться критической не более $\frac{|V|}{2} - 1$ раз.

Пусть $u$ и $v$ – вершины из множества вершин $V$, соединенные некоторой дугой из множества $E$. Поскольку увеличивающие пути – это кратчайшие пути, то тогда дуга $(u, v)$ становится критической первый раз, справедливо равенство

$$\delta_f(s, v) = \delta_f(s, u) + 1.$$

После того, как поток увеличен, дуга $(u, v)$ исчезает из остаточной сети. Она не может появиться в другом увеличивающем пути, пока не будет уменьшен поток из $u$ в $v$, а это может произойти только в том случае, если на некотором увеличивающем пути встретится дуга $(v, u)$. Если в этот момент поток в сети $\vec{G}$ составляет $f'$, справедливо следующее равенство:

$\delta'_f(s, u) = \delta'_f(s, v) + 1$. Поскольку, согласно лемме 7, $\delta_f(s, v) \leq \delta'_f(s, v)$, получаем: $\delta'_f(s, u) = \delta'_f(s, v) + 1 \geq \delta_f(s, v) + 1 = \delta_f(s, u) + 2$.

Следовательно, за время, прошедшее с момента, когда дуга $(u, v)$ была критической, до момента, когда она становится критической в следующий раз, расстояние до $u$ от источника увеличивается не менее чем на 2. Расстояние до $u$ от источника в начальный момент было не меньше 0. Среди промежуточных вершин на кратчайшем пути из $s$ в $u$ не могут находиться $s$, $u$ или $t$ (поскольку наличие дуги $(u, v)$ в кратчайшем пути подразумевает, что $u \neq t$). Следовательно, к тому моменту, когда вершина $u$ станет недостижимой из источника, расстояние до нее будет не более $|V| - 2$. Таким образом, дуга $(u, v)$ может стать критической не более $O(E)$ пар вершин, которые могут быть соединены дугами, общее количество критических дуг в ходе выполнения алгоритма Диница равно $O(VE)$. Каждый увеличивающий путь содержит по крайней мере одну критическую дугу, следовательно, теорема доказана. ■

Если увеличивающий путь находится посредством поиска в ширину, каждую итерацию алгоритм Форда-Фалкерсона можно выполнить за время

$O(E)$, следовательно, суммарное время выполнения алгоритма Диница составляет $O(VE^2)$.

### 2.3 Алгоритм проталкивания предпотока

Многие наиболее асимптотически быстрые алгоритмы поиска максимального потока принадлежат данному классу, и на этом методе основаны реальные реализации алгоритма поиска максимального потока. С помощью методов проталкивания предпотока можно решать и другие связанные с потоками задачи, например, задачу поиска потока с минимальными затратами. Время работы простой реализации $O(V^2E)$, также существует усовершенствованная версия алгоритма, работающая за время $O(V^3)$.

Алгоритм проталкивания предпотока обрабатывают вершины по одной, рассматривая только соседей данной вершины в остаточной сети. Алгоритмы проталкивания предпотока не обеспечивают в ходе своего выполнения свойство сохранения потока. При этом они поддерживают предпоток, который представляет собой функцию $f: VxV \to R$, обладающую свойством антисимметричности, удовлетворяющую ограничениям пропускной способности и следующему ослабленному условию сохранения потока: $f(V,u) \geq 0$ для всех вершин $u \in V - \{s\}$. Это количество называется избыточным потоком, входящим в вершину $u$, и обозначается

$$e(u) = f(V,u)$$

Вершина $u \in V - \{s,t\}$, называется переполненной, если $e(u) > 0$.

#### 2.2.1 Основные операции

В алгоритме проталкивая предпотока выполняются две основные операции: проталкивание избытка потока от вершины к одной из соседних с ней вершин и подъем вершины.

Пусть $\vec{G} = (V, E)$ транспортная сеть с источником $s$ и стоком $t$, а $f$ – некоторый предпоток в $\vec{G}$. Функция$h: V \to N$ является функцией высоты, если $h(s) = |V|, h(t) = 0$ и

$$h(u) \leq h(v) + 1$$

для любой остаточной дуги $(u, v) \in E_f$. Сразу же можно сформулировать следующую лемму.

Лемма 9. Пусть $\vec{G} = (V, E)$ – транспортная сеть, а $f$ – некоторый предпоток в $\vec{G}$, и пусть $h$ – функция высоты, заданная на множестве $V$. Для любых двух вершин $u, v \in V$ справедливо следующее утверждение: $h(u) > h(v) + 1$, то $(u, v)$ не является дугой остаточного графа.■

Основная операция проталкивания может применяться тогда, когда u является переполненной вершиной, $c_f(u, v) > 0$ и $h(u) = h(v) + 1$. Предполагается, что остаточные пропускные способности при заданных $f$ и $c$ можно вычислить за фиксированное время. Излишний поток, хранящийся в вершине $u$, поддерживается в виде атрибута $e[u]$, а высота вершины $u$ – в виде атрибута $h[u]$. Выражение $d_f(u, v)$ – это временная переменная, в которой хранится количество потока, которое можно протолкнуть из $u$ в $v$.

Операция проталкивания

Условия применения: u переполнена, $c_f(u, v) > 0$, и $h[u] = h[v] + 1$.

Шаг 1. $d_f(u, v) = \min(e[u], c_f(u, v))$

Шаг 2. $f[u, v] = f[u, v] + d_f(u, v)$

Шаг 3. $f[v, u] = -f[u, v]$

Шаг 4. $e[u] = e[u] - d_f(u, v)$

Шаг 5. $e[v] = e[v] + d_f(u, v)$

Операция проталкивания работает следующим образом. Предполагается, что вершина $u$ имеет положительный избыток $e[u]$ и остаточная пропускная способность дуги $(u, v)$ положительна. Тогда можно увеличить поток из $u$ в $v$ на величину $d_f(u, v) = \min(e[u], c_f(u, v))$, при этом избыток $e[u]$ не становится отрицательным и не будет превышена пропускная способность $c(u, v)$. Если функция $f$ является предпотоком перед применением операции проталкивания, она останется предпотоком и после ее применения.

Операция проталкивания называется *проталкиванием* из $u$ к $v$. Если операция проталкивания применяется к некоторой дуге $(u, v)$, выходящему из вершины $u$, будем говорить, что операция проталкивания применяется к $u$. Если в результате дуга $(u, v)$ становится насыщенной (после проталкивания $c_f(u, v) = 0$), то это *насыщающее проталкивание*, в противном случае это *ненасыщающее проталкивание*. Если дуга насыщена, она не входит в остаточную сеть. Один из результатов ненасыщающего проталкивания характеризует следующая лемма.

Лемма 10. После ненасыщающего проталкивания из $u$ в $v$ вершина $u$ более не является переполненной.

Доказательство. Поскольку проталкивание ненасыщающее, количество посланного потока должно быть равно величине $e[u]$ непосредственно перед проталкиванием. Поскольку избыток $e[u]$ уменьшается на эту величину, после проталкивания он становится равным 0. ■

Основная операция подъема $Relabel(u)$ применяется, если вершина $u$ переполнена и $h[u] \leq h[v]$ для всех дуг $(u, v) \in E_f$. Иными словами, переполненную вершину u можно подвергнуть подъему, если все вершины $v$, для которых имеется остаточная пропускная способность от $u$ к $v$, расположены не ниже $u$, так что протолкнуть поток из $u$ нельзя. Ни источник $s$, ни сток $t$ нельзя подвергать подъему.

Операция подъема

Условие применения: $u$ переполнена и для всех $v \in V$, таких что $(u, v) \in E_f, h[u] \leq h[v]$.

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

Когда вызывается операция подъема, говорим, что вершина $u$ подвергается подъему. Заметим, что когда производится подъем $u$, остаточная сеть $E_f$ должна содержать хотя бы одну дугу, выходящую из $u$, чтобы минимизация в коде операции производилась по непустому множеству. Это свойство вытекает из предположения, что вершина $u$ переполнена. Поскольку $e[u] > 0$, имеем $e[u] = f(V, u) > 0$ и, следовательно, должна существовать по крайней мере одна вершина $v$, такая что $f[v, u] > 0$. Но тогда

$$c_f(u, v) = c(u, v) - f[u, v] = c(u, v) + f[v, u] > 0,$$

Откуда вытекает, что $(u, v) \in E_f$. Таким образом, операция подъема дает $u$ наибольшую высоту, допускаемую наложенными на функцию высоты ограничениями.

### 2.2.2. Универсальный алгоритм.

Универсальный алгоритм проталкивания предпотока использует следующую процедуру для создания начального предпотока в транспортной сети:

Алгоритм создания начального потока в транспортной сети

Шаг 1. для каждой вершины $u \in V[G]$: выполнять шаги 2-3

Шаг 2. $h[u] = 0$

Шаг 3. $e[u] = 0$

Шаг 4. для каждой дуги $(u, v) \in E|G|$: выполнять шаги 5-6

Шаг 5. $f[u, v] = 0$

Шаг 6. $f[v,u] = 0$

Шаг 7. $h[s] = |V[G]|$

Шаг 8. для каждой вершины $u \in Adj[s]$: выполнять шаги 9-12

Шаг 9. $f[s,u] = c(s,u)$

Шаг 10. $f[u,s] = -c(s,u)$

Шаг 11. $e[u] = c(s,u)$

Шаг 12. $e[s] = e[s] - c(s,u)$

Алгоритм, указанный выше, создает начальный предпоток $f$, определяемый формулой

$$f[u,v] = \begin{cases} c(u,v), \text{если } u = s \\ -c(v,u), \text{если } v = s \\ 0, \text{в противном случае.} \end{cases}$$

Это действительно функция высоты, поскольку единственной дугой $(u,v)$, для которых $h[u] > h[v] + 1$, являются дуги, для которых $u = s$, и эти дуги заполнены, а это значит, что их нет в остаточной сети.

Инициализация, за которой следует ряд операций проталкивания и подъема, выполняемых без определенного порядка, образует алгоритм:

Алгоритм

Шаг 1. создать начальный предпоток алгоритмом инициализации

Шаг 2. пока можно выполнить одну из операций проталкивания или подъема: выполнять шаг 3.

Шаг 3. выбрать операцию проталкивания или подъема и выполнить её

Следующая лемма утверждает, что до тех пор, пока существует хотя бы одна переполненная вершина, применима хотя бы одна из этих операций.

Лемма 11 (Для переполненной вершины можно выполнить либо проталкивание, либо подъем).

Пусть $\vec{G} = (V, E)$ – транспортная сеть с источником $s$ и стоком $t$, $f$ – предпоток, а $h$ – некоторая функция высоты для $f$. Если $u$ – некоторая переполненная вершина, то к ней можно применить или операцию проталкивания, или операцию подъема.

Доказательство. Для любой остаточной дуги $(u, v)$ выполняется соотношение $h(u) \leq h(v) + 1$, поскольку $h$ – функция высоты. Если к $u$ не применима операция проталкивания, то для всех остаточных дуг $(u, v)$ должно выполняться условие $h(u) < h(v) + 1$, откуда следует, что $h(u) \leq h(v)$. В этом случае к $u$ можно применить операцию подъема. ■

Чтобы показать, что универсальный алгоритм проталкивания предпотока позволяет решить задачу максимального потока, сначала докажем, что после его завершения предпоток $f$ является максимальным потоком. Затем докажем, что алгоритм завершается. Начнем с рассмотрения некоторых свойств функции высоты $h$.

Лемма 12 (Высота вершины никогда не уменьшается). При выполнении алгоритма проталкивания предпотока над транспортной сетью $\vec{G} = (V, E)$, для любой вершины $u \in V$ ее высота $h[u]$ никогда не уменьшается. Более того, всякий раз, когда к вершине $u$ применяется операция подъема, ее высота $h[u]$ увеличивается как минимум на 1.

Доказательство. Поскольку высота вершины меняется только при выполнении операции подъема, достаточно доказать второе утверждение леммы. Если вершина $u$ должна подвергнуться подъему, то для всех вершин $v$, таких что $(u, v) \in E_f$, выполняется условие $h[u] \leq h[v]$. Таким образом, $h[u] < 1 + \min\{h[v]: (u, v) \in E_f\}$, и операция должна увеличить значение $h[u]$. ■

Лемма 13. Пусть $\vec{G} = (V, E)$ – транспортная суть с источником $s$ и стоком $t$. Во время выполнения алгоритма проталкивания предпотока над сетью $\vec{G}$ атрибут $h$ сохраняет свойство функции высоты.

Доказательство. Доказательство проводится индукцией по числу выполненных основных операций. $h$ является функцией высоты.

Утверждается, что если $h$ – функция высоты, то после выполнения операции подъема она останется функцией высоты. Если посмотреть на остаточную дугу $(u, v) \in E_f$, выходящее из $u$, то операция подъема гарантирует, что после ее выполнения $h[u] \leq h[v] + 1$. Рассмотрим теперь некоторую остаточную дугу $(w, u)$, входящее в $u$. Согласно лемме 12, $h[w] \leq h[u] + 1$ перед выполнением операции подъема; следовательно, после ее выполнения $h[w] < h[u] + 1$. Таким образом, операция подъема оставляет $h$ функцией высоты.

Теперь рассмотрим операцию проталкивания. Данная операция может добавить дугу $(v, u)$ к $E_f$ или удалить дугу $(u, v)$ из $E_f$. В первом случае имеем $h[v] = h[u] - 1 < h[u] + 1$, так что $h$ остается функцией высоты. Во втором случае удаление дуги $(u, v)$ из остаточной сети приводит к удалению соответствующего ограничения, так что $h$ по-прежнему остается функцией высоты. ■

Следующая лемма характеризует важное свойство функции высоты.

Лемма 14. Пусть $\vec{G} = (V, E)$ – транспортная сеть с источником $s$ и стоком $t$, $f$ – предпоток в $\vec{G}$, а $h$ – функция высоты, определенная на множестве $V$. Тогда не существует пути из источника $s$ к стоку $t$ в остаточной сети $\vec{G}_f$.

Доказательство. Предположим, что в $\vec{G}_f$ существует некоторый путь $p = \{v_0, v_1, \dots, v_k\}$ из $s$ в $t$, где $v_0 = s$, а $v_k = t$, и покажем, что это приводит к противоречию. Без потери общности можно считать, что $p$ – простой путь, так что $k < |V|$. Для $i = 0,1, \dots, k-1$, дуги $(v_i, v_{i+1}) \in E_f$. Поскольку $h$ – функция высоты, для $i = 0,1, \dots, k-1$ справедливы соотношения $h(v_i) \leq h(v_{i+1})$.

Объединяя эти неравенства вдоль пути $p$, получим, что $h(s) \leq h(t) + k$. Но поскольку $h(t) = 0$, получаем $h(s) \leq k < |V|$, что противоречит требованию $h(s) = |V|$ к функции высоты. ■

Покажем, что после завершения универсального алгоритма проталкивания предпотока вычисленный алгоритмом предпоток является максимальным потоком.

Теорема 15 (О корректности универсального алгоритма проталкивания предпотока).

Если алгоритм проталкивания предпотока, выполняемый над сетью $\vec{G} = (V < E)$ с источником $s$ и стоком $t$, завершается, то вычисленный им предпоток $f$ является максимальным потоком в $\vec{G}$.

Доказательство. Используем следующий инвариант цикла: всякий раз, когда производится проверка условия цикла на 2 шаге алгоритма проталкивания предпотока, $f$ является предпотоком.

Инициализация. Алгоритм создания начального потока делает $f$ предпотоком.

Сохранение. Внутри цикла на шаге 2 выполняется только операции проталкивания и подъема. Операции подъема влияют только на атрибуты высоты, но не на величину потока, следовательно, от них не зависит, будет ли $f$ предпотоком. Анализируя работу операции проталкивания, мы доказали, что, если $f$ является предпотоком перед выполнением операции проталкивания, он останется предпотоком и после ее выполнения.

Завершение. По завершению процедуры каждая вершина из множества $V - \{s, t\}$ должна иметь избыток, развый 0, поскольку из лемм 11 и 13 и инварианта, что $f$ всегда останется предпотоком, вытекает, что переполненных вершин нет. Следовательно, $f$ является потоком. Поскольку $h$ – функция высоты, согласно лемме 14 не существует пути из $s$ в $t$ в остаточной сети $\vec{G}_f$. Согласно

теореме 6 о максимальном потоке и минимальном разрезе, $f$ является максимальным потоком. ■

Лемма 16. Пусть $\vec{G} = (V, E)$ – транспортная сеть с источником $s$ и стоком $t$, а $f$ – предпоток в $\vec{G}$. Тогда для любой переполненной вершины $u$ существует простой путь из $u$ в $s$ в остаточной сети $\vec{G}_f$.

Доказательство. Пусть $u$ – некоторая переполненная вершина, и пусть $U = \{v$: в $\vec{G}_f$ существует простой путь из $u$ в $v\}$. Предположим, что $s \notin U$, и покажем, что это приведет к противоречию. Обозначим $\overline{U} = V - U$.

Утверждение, что для каждой пары вершин $w \in \overline{U}$ и $v \in U$ выполняется соотношение $f(w, v) \leq 0$. Если $f(w, v) > 0$, то $f(v, w) < 0$, откуда в свою очередь вытекает, что $c_f(v, w) = c(v, w) - f(v, w) > 0$. Следовательно, существует дуга $(v, w) \in E_f$ и существует простой путь вида $u \dashrightarrow v \rightarrow w$ в остаточной сети $G_f$, что противоречит тому, как мы выбирали $w$.

Таким образом, должно выполняться неравенство $f(\overline{U}, U) \leq 0$, поскольку каждое слагаемое в этом неявном суммировании неположительное, и, следовательно, $e(U) = f(V, U) = f(\overline{U}, U) + f(U, \overline{U}) = f(\overline{U}, U) \leq 0$.

Излишки неотрицательны для всех вершин из множества $V - \{s\}$, поскольку предположили, что $U \subseteq V - \{s\}$, то для всех вершин $v \in U$ должно выполняться $e(v) = 0$. В частности, $e(u) = 0$, что противоречит предположению о том, что $u$ переполнена. ■

Следующая лемма устанавливает границы высот вершин, а вытекающее из нее следствие устанавливает предел общего числа выполненных операций подъема.

Лемма 17. Пусть $G = (V, E)$ – транспортная сеть с источником $s$ и стоком $t$. В любой момент в процессе выполнения алгоритма проталкивания предпотока в сети $\vec{G}$ для всех вершин $u \in V$ выполняется соотношение $h[u] \leq 2|V| - 1$.

Доказательство. Высота источника $s$ и стока $t$ никогда не изменяется, поскольку эти вершины по определению не переполняются. Таким образом, всегда $h[s] = |V|$ и $h[t] = 0$, и обе не превышают $2|V| - 1$.

Рассмотрим теперь произвольную вершину $u \in V - \{s, t\}$. Изначально $h[u] = 0 \leq 2|V| - 1$. Покажем, что после каждого подъема неравенство $h[u] \leq 2|V| - 1$ остается справедливым. При подъеме вершины $u$ она является переполненной и, согласно лемме 16, имеется простой путь $p$ из $u$ в $s$ в $\vec{G}_f$. Пусть $p = \{v_0, \dots, v_k\}$, где $v_0 = u$, а $v_k = s$, и $k \leq |V| - 1$, поскольку $p$ – простой путь. Для $i = 0, \dots, k - 1$ имеем $(v_i, v_{i+1}) \in E_f$, следовательно, $h[v_i] \leq h[v_{i+1}] + 1$ согласно лемме 13. Расписав неравенства для всех составляющих пути $p$, получаем $h[u] = h[v_0] \leq h[v_k] + k \leq h[s] + (|V| - 1) = 2|V| - 1$. ■

Следствие 18. (верхний предел числа подъемов). Пусть $\vec{G} = (V, E)$ – транспортная сеть с источником $s$ и стоком $t$. Тогда в процессе выполнения алгоритма проталкивания предпотока в $\vec{G}$ число подъемов не превышает $2|V| - 1$ для одной вершины, а их общее количество не более $(2|V| - 1)(|V| - 2) < 2|V|^2$.

Доказательство. Во множестве $V - \{s, t\}$ только $|V| - 2$ вершин могут быть подняты. Пусть $u \in V - \{s, t\}$. Операция подъема увеличивает высоту $h[u]$. Значение $h[u]$ первоначально равно 0 и, согласно лемме 17, возрастает не более чем на $2|V| - 1$. Таким образом, каждая вершина $u \in V - \{s, t\}$ подвергается подъему не более $2|V| - 1$ раз, а общее число выполненных подъемов не превышает $(2|V| - 1)(|V| - 2) < 2|V|^2$. ■

Лемма 17 также помогает определить границу количества насыщающих проталкиваний.

Лемма 19. (Граница количества насыщающих проталкиваний). В процессе выполнения алгоритма проталкивания предпотока для любой

транспортной сети $\vec{G} = (V, E)$ число насыщающих проталкиваний меньше, чем $2|V||E|$.

Доказательство. Для любой пары вершин $u, v \in V$ рассмотрим насыщающие проталкивания от $u$ к $v$ и от $v$ к $u$, и назовем их насыщающими проталкиванием между $u$ и $v$. Если есть хотя бы одно такое проталкивание, то хотя бы одна из дуг $(u, v)$ и $(v, u)$ является дугой в $E$. Теперь предположим, что произошло насыщающее проталкивание из $u$ в $v$. В этот момент $h[v] = h[u] - 1$. Чтобы позднее могло произойти еще одно проталкивание из $u$ в $v$, алгоритм сначала должен протолкнуть поток из $v$ в $u$, что невозможно до тех пока не будет выполнено условие $h[v] = h[u] + 1$. Поскольку $h[u]$ никгда не уменьшается, для того чтобы выполнялось условие $h[v] = h[u] + 1$, значение $h[v]$ должно увеличиться по меньшей мере на 2. Аналогично, $h[u]$ должно увеличиться между последовательными насыщающими проталкиваниями из $v$ в $u$ как минимум на 2. Высота изначально принимает значение 0 и, согласно лемме 17, никогда не превышает $2|V| - 1$, откуда следует, что количество раз, когда высота вершины может увеличить на 2, меньше $2|V|$ насыщающих проталкиваний между $u$ и $v$. Умножив это число на число дуг, получим, что общее число насыщающих проталкиваний меньше, чем $2|V||E|$. ■

Следующая лемма устанавливает границу числа ненасыщающих проталкиваний в обобщенном алгоритме проталкивания предпотока.

Лемма 20. (Граница количества ненасыщающих проталкиваний)

В процессе выполнения алгоритма проталкивания предпотока для любой транспортной сети $\vec{G} = (V, E)$ число ненасыщающих проталкиваний меньше $4|V|^2(|V| + |E|)$.

Доказательство. Определим потенциальную функцию $\Phi = \sum_{v:e(v)>0} h[v]$. Изначально $\Phi = 0$ и значение $\Phi$ может изменяться после каждого подъема, насыщающего и ненасыщающего проталкивания. Найдем предел величины, на которую насыщающие проталкивания и подъемы могут увеличивать $\Phi$. Затем

покажем, что каждое ненасыщающее проталкивание должно увеличивать Ф. Затем покажем, что каждое ненасыщающее проталкивание должно уменьшать Ф как минимум на 1, и используем эти оценки для определения верхней границы числа ненасыщающих проталкиваний.

Рассмотрим два пути увеличения Ф. Во-первых, подъем вершины u увеличивает Ф менее чем на $2|V|$, поскольку множество, для которого вычисляется сумма, остается прежним, а подъем не может увеличить высоту вершины u больше, чем ее максимально возможная высота, которая составляет не более $2|V| - 1$ согласно лемме 17. Во-вторых, насыщающее проталкивание из вершины $u$ в вершину $v$ увеличивает Ф менее чем на $2|V|$, поскольку никаких изменений высот при этом не происходит, и только вершина $v$, высота которой не более $2|V| - 1$, может стать переполненной.

Теперь покажем, что ненасыщающее проталкивание из $u$ в $v$ уменьшает Ф не менее чем на 1. Перед ненасыщающим проталкиванием вершина $u$ была переполненной, а $v$ могла быть переполненной или непереполненной. Согласно лемме 10, после этого проталкивания $u$ больше не является переполненной. Кроме того, после данного проталкивания $v$ должна быть переполненной, если только она не является источником. Следовательно, потенциальная функция Ф уменьшилась ровно на $h[u]$, а увеличилась на 0 или на $h[v]$. Поскольку $h[u] - h[v] = 1$, в итоге потенциальная функция уменьшается как минимум на 1.

Таким образом, в ходе выполнения алгоритма увеличение Ф просиходит благодаря подъемам и насыщающим проталкиваниям; согласно следствию 18 и лемме 19, это увеличение ограничено, и составляет менее $(2|V|)(2|V|^2) + (2|V|)(2|V||E|) = 4|V|^2(|V| + |E|)$. Поскольку $\Phi \geq 0$, суммарная величина уменьшения и, следовательно, общее число ненасыщающих проталкиваний меньше, чем $4|V|^2(|V| + |E|)$. ■

Определив границу числа подъемов, насыщающего проталкивания и ненасыщающего проталкивания, заложили основу дальнейшего анализа

алгоритма проталкивания предпотока, а также любых других алгоритмов, основанных на методе проталкивания предпотока.

Теорема 21. При выполнении алгоритма проталкивания предпотока для любой транспортной сети $\vec{G} = (V, E)$ число основных операций составляет $O(V^2E)$.

Доказательство. Непосредственно вытекает из уже доказанных следствия 18 и лемм 19 и 20. ■

### 2.4 Алгоритм «поднять в начало»

Метод проталкивания предпотока позволяет применять основные операции в произвольном порядке. Однако после тщательного выбора порядка их выполнения и при эффективном управлении структурой сетевых данных, можно решить задачу поиска максимально потока быстрее, чем за предельное время $O(V^2E)$, а для плотных сетей – лучше.

Алгоритм «поднять-в-начало» поддерживает список вершин сети. Алгоритм сканирует список с самого начала, выбирает некоторую переполненную вершину $u$, а затем «разгружает» ее, т.е. выполняет операции проталкивания и подъема до тех пор, пока избыток в $u$ не станет равен 0. Если выполнялось поднятие вершины, то она переносится в начало списка, и алгоритм начинает очередное сканирование списка.

Для исследования корректности и временных характеристик данного алгоритма используется понятие «допустимых» дуг: это дуги остаточной сети, состоящей из допустимых дуг.

#### 2.3.1 Допустимые дуги и сети

Пусть $\vec{G} = (V, E)$ – некоторая транспортная сеть с источником $s$ и стоком $t, f$ – предпоток в $G$, а $h$ – функция высоты. Дуга $(u, v)$ называется *допустимой дугой*, если $c_f(u, v) > 0$ и $h(u) = h(v) + 1$. В противном случае дуга $(u, v)$

называется *недопустимой*. *Допустимой сетью* является сеть $\vec{G}_{f,h} = (V, E_{f,h})$, где $E_{f,h}$ – множество допустимых дуг.

Допустимая сеть состоит из тех дуг, через которые можно протолкнуть поток. Следующая лемма показывает, что такая сеть является ориентированным ациклическим графом.

Лемма 22. (Допустимая сеть является ациклической). Если $\vec{G} = (V, E)$ – некоторая транспортная сеть с источником $s$ и стоком $t$, $f$ – предпоток в $\vec{G}$, а $h$ – функция высоты, тогда допустимая сеть $\vec{G}_{f,h} = (V, E_{f,h})$ является ациклической.

Доказательство. Доказательство проводится методом от противного. Предположим, что $\vec{G}_{f,h}$ содержит некоторый циклический путь $p = (v_0, \dots, v_k)$, где $v_0 = v_k$ и $k > 0$. Поскольку каждая дуга пути $p$ является допустимой, справедливо равенство $h(v_{i-1}) = h(v_i) + 1$ для $i = 1, \dots, k$. Просуммировав эти равенства вдоль циклического пути, получаем:

$$\sum_{i=1}^{k} h(v_{i-1}) = \sum_{i=1}^{k} (h(v_i) + 1) = \sum_{i=1}^{k} h(v_i) + k$$

Поскольку каждая вершина циклического пути $p$ встречается при суммировании по одному разу, приходим к выводу, что $k = 0$, что противоречит первоначальному предположению. ■

Слушающая лемма показывает, как операции проталкивания и подъема изменяют допустимую сеть.

Лемма 23. Пусть $\vec{G} = (V, E)$ – транспортная сеть с источником $s$ и стоком $t$, $f$ – предпоток в $\vec{G}$, и предположим, что $h$ – функция высоты. Если вершина u переполнена и дуга $(u, v)$ является допустимой, то применяется операция проталкивания. Данная операция не создает новые допустимые дуги, однако она может привести к тому, что дуга $(u, v)$ станет недоступной.

Доказательство. По определению допустимой дуги, из $u$ в $v$ можно протолкнуть поток. Поскольку вершина $u$ переполнена, применяется операция проталкивания. В результате проталкивания потока из $u$ в $v$ может быть создана только одна новая остаточная дуга $(v, u)$. Поскольку $h[v] = h[u] - 1$, дуга $(v, u)$ не может стать допустимой. Если примененная операция является насыщающим проталкиванием, то после ее выполнения $c_f(u, v) = 0$ и дуга $(u, v)$ становится недопустимой. ■

Переполненная вершина $u$ *разгружается* путем проталкивания всего ее избыточного потока через допустимые дуги в смежные вершины, при этом, если необходимо, производится подъем вершины $u$, чтобы дуги, выходящие из вершины $u$, стали допустимыми.

В алгоритме «поднять-в-начало» поддерживается связанный список $L$, состоящий из всех вершин $V - \{s, t\}$. Ключевым свойством данного списка является то, что вершины в нем топологически отсортированы в соответствии с допустимой сетью.

## 3 Сравнение некоторых алгоритмов поиска максимального потока

Для сравнения некоторых алгоритмов поиска максимально потока были реализованы указанные выше алгоритмы: алгоритм Диница и алгоритм проталкивания предпотока, реализован генератор графов для проведения серии экспериментов на различных графах.

Программа была написана на языке Python 3.6 и имеет графический интерфейс, для построения которого использованы модули PyQt5, matplotlib и networkx.

Генератор графов

Генератор имеет 5 возможных параметров, которые соответственно задаются в поле «параметры» интерфейса программы.

Параметр «-n» отвечает за количество вершин в сгенерированном графе, если параметр не указывается, то он выбирается случайно в интервале $[2, 20]$. Параметр «-m» отвечает за количество дуг, если параметр не указывается, то он выбирается случайно в интервале $[n, n * (n - 1)]$. Параметр «-cup» отвечает за установку фиксированной пропускной способности, если параметр не указывается, то пропускная способность для каждой дуги выбирается случайно в интервале $[1, max\_cup]$. Параметр «-max_cup» отвечает за максимально возможную пропускную способность, если параметр не указывается, то max_cup имеет значение по умолчанию равное 20. Параметр «-file» отвечает за считывание графа из файла, если параметр указан, то все остальные параметры игнорируются и граф считывается из указанного файла. На рисунке 1 представлен пример установки параметров.

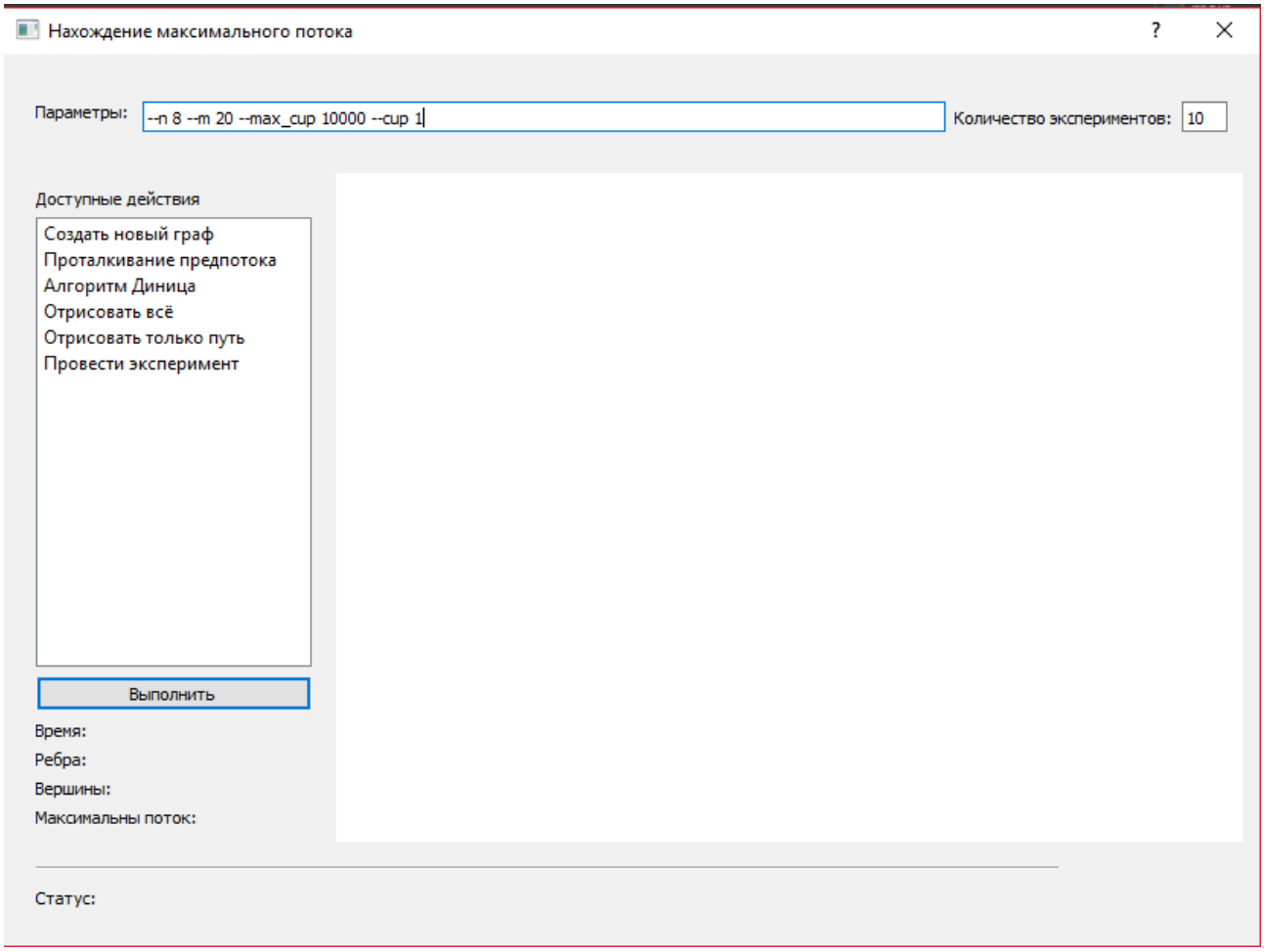


Рисунок 1

Генератор графов создает граф случайным образом, имеет $n$ вершин и $m$ дуг, также полученный граф не имеет петель. Гарантируется, что путь из 0 вершины в $n - 1$ вершину будет существовать в созданном графе.

Нахождение максимального потока

Для того, чтобы найти максимальный поток в сгенерированном или считанном из файла графе, необходимо

Шаг 1. По необходимости задать параметры для генератора графов

Шаг 2. Выбрать поле «Создать новый граф» в списке слева

Шаг 3. Нажать на кнопку «Выполнить».

Шаг 4. Выбирать один из двух алгоритмов: «Алгоритм Диница» или «Проталкивание предпотока» в списке слева

Шаг 5. Нажать на кнопку «Выполнить».

Алгоритм исполнится на графе, полученном на шагах 1-3. Время выполнения и найденный максимальный поток отобразятся под списком слева.

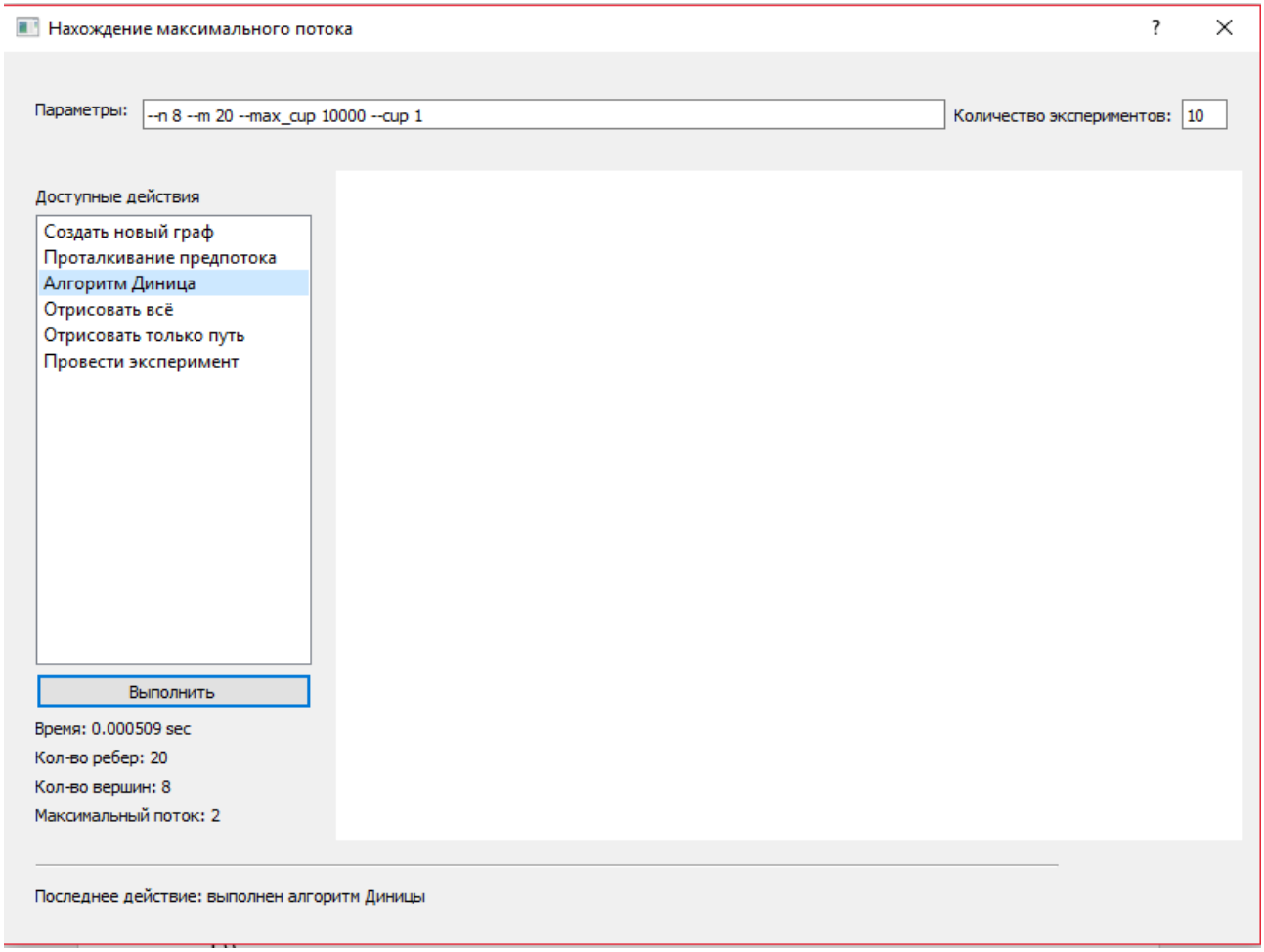


Рисунок 2

На рисунке 2 показан результат работы программы используя алгоритм Диница на графе, созданном случайно.

На рисунке 3 приведен листинг алгоритма Диница. (спорный момент)

(Рисунок)

Для того, чтобы отобразить граф, необходимо

Шаг 1. Выбрать один из режимов отображения: «Отобразить всё» или «Отобразить только путь»

Шаг 2. Нажать на кнопку «Выполнить»

Рисунок 3

На рисунке 3 показан результат отображения графа с найденным на нем максимальным потоком.

Сравнение скорости работы алгоритмов

Для того, чтобы узнать среднюю скорость работы алгоритмов на серии экспериментов, необходимо

Шаг 1. Задать число экспериментов (сколько раз сгенерировать случайный граф) в поле «Количество экспериментов»

Шаг 2. Выбрать поле «Провести эксперимент» в списке слева.

Шаг 3. Нажать на кнопку «Выполнить».

Результат эксперимента будет отображен в поле снизу.

ЭКСПЕРИМЕНТЫ☺

Была проведена серия экспериментов с параметрами генератора

Количество сгенерированных графов 3000

- $-n\ 8 - m\ 15 - cup\ 1$
- $-n\ 8 - m\ 15 - \max\_cup\ 1000000$
- $-n\ 8 - m\ 15$
- $-n\ 9 - m\ 20 - cup\ 1$
- $-n\ 9 - m\ 20\ - \max\_cup\ 1000000$
- $-n\ 9 - m\ 20$
- $-n\ 15 - m\ 30$
- $-n\ 15 - m\ 30$
- $-n\ 15 - m\ 30$
- $-n\ 30 - m\ 100$
- $-n\ 30 - m\ 100$

## ЗАКЛЮЧЕНИЕ

Я научился вставлять формулу в ворд.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Богомолов, А.М. Алгебраические основы теории дискретных систем [Текст] / А.М. Богомолов, В.Н. Салий – Москва: «Физико-математическая литература» РАН, 1997. – 367с.

2. Кормен, Томас Х., Алгоритмы: построение и анализ, 2-е издание. [Текст] / Кормен, Томас Х., Лейзернон, Чарльз И., Ривест, Рональд Л., Штайн, Клиффорд – Издательский дом “Вильямс”, 2005. – 1296с.

3. Зыков, А.А. Основы теории графов [Текст] / А.А. Зыков –Москва: «Наука», 1987. –382с.

4. Абросимов М.Б. Практические задания по графам [Текст]: учебное пособие / М.Б. Абросимов, А.А. Долгов –Саратов: «Научная книга», 2016. –82с.

5. википедия)) https://ru.wikipedia.org/wiki/Транспортная_сеть

# ПРИЛОЖЕНИЕ А

да