

Министерство науки и высшего образования Российской Федерации

Калужский филиал  
федерального государственного бюджетного образовательного  
учреждения высшего образования  
**«Московский государственный технический университет  
имени Н.Э. Баумана  
(национальный исследовательский университет)»**  
(КФ МГТУ им. Н.Э. Баумана)

**Ю.С. Белов, С.С. Гришунов**

**РЕШЕНИЕ ЗАДАЧ ОПТИМИЗАЦИИ ПРИ ПРИНЯТИИ РЕШЕНИЙ**  
Методические указания к выполнению домашней работы  
по курсу «Типы и структуры данных»

Калуга – 2019


УДК 004.62  
ББК 32.972.5  
Б435

Методические указания составлены в соответствии с учебным планом КФ МГТУ им. Н.Э. Баумана по направлению подготовки 09.03.04 «Программная инженерия» кафедры «Программного обеспечения ЭВМ, информационных технологий».

Методические указания рассмотрены и одобрены:


- Кафедрой «Программного обеспечения ЭВМ, информационных технологий» (ИУ4-КФ) протокол № 51.4/5 от «23» января 2019 г.

Зав. кафедрой ИУ4-КФ

 к.т.н., доцент Ю.Е. Гагарин

- Методической комиссией факультета ИУ-КФ протокол № 7 от «29» 01 2019 г.


Председатель методической  
комиссии факультета ИУ-КФ

 к.т.н., доцент М.Ю. Адкин

- Методической комиссией

КФ МГТУ им.Н.Э. Баумана протокол № 4 от «5» 02 2019 г.

Председатель методической комиссии  
КФ МГТУ им.Н.Э. Баумана

 д.э.н., профессор О.И. Перерва

Рецензент:

к.т.н., доцент кафедры ИУ6-КФ

 А.Б. Лачихина

Авторы

к.ф.-м.н., доцент кафедры ИУ4-КФ  
ассистент кафедры ИУ4-КФ

 Ю.С. Белов  
 С.С. Гришунов

#### Аннотация

Методические указания к выполнению домашнего задания по курсу «Типы и структуры данных» содержат общие сведения об алгоритмах поиска максимального потока и максимального паросочетания в графе.

Предназначены для студентов 2-го курса бакалавриата КФ МГТУ им. Н.Э. Баумана, обучающихся по направлению подготовки 09.03.04 «Программная инженерия».

© Калужский филиал МГТУ им. Н.Э. Баумана, 2019 г.  
© Ю.С. Белов, С.С. Гришунов, 2019 г.

## ОГЛАВЛЕНИЕ

|   |    |
|---|----|
| ВВЕДЕНИЕ .....  | 4  |
| ЦЕЛЬ И ЗАДАЧИ РАБОТЫ, ТРЕБОВАНИЯ К РЕЗУЛЬТАТАМ ЕЕ<br>ВЫПОЛНЕНИЯ ..... | 5  |
| ЗАДАЧА О МАКСИМАЛЬНОМ ПОТОКЕ .....                                    | 6  |
| ЗАДАЧА О МАКСИМАЛЬНОМ ПАРОСОЧЕТАНИИ .....                             | 25 |
| ЗАДАНИЕ ДЛЯ ДОМАШНЕЙ РАБОТЫ .....                                     | 30 |
| ВАРИАНТЫ ЗАДАНИЙ .....  | 30 |
| КОНТРОЛЬНЫЕ ВОПРОСЫ И ЗАДАНИЯ .....                                   | 31 |
| ФОРМА ОТЧЕТА ПО ДОМАШНЕЙ РАБОТЕ .....                                 | 31 |
| ОСНОВНАЯ ЛИТЕРАТУРА .....   | 32 |
| ДОПОЛНИТЕЛЬНАЯ ЛИТЕРАТУРА.....  | 32 |

## **ВВЕДЕНИЕ**

Настоящие методические указания составлены в соответствии с программой проведения домашних работ по курсу «Типы и структуры данных» на кафедре «Программное обеспечение ЭВМ, информационные технологии» факультета «Информатика и управление» Калужского филиала МГТУ им. Н.Э. Баумана.

Методические указания к выполнению домашней работы по курсу «Типы и структуры данных» содержат краткое описание алгоритмов поиска максимального потока и максимального паросочетания в графе.

Предназначены для студентов 2-го курса бакалавриата КФ МГТУ им. Н.Э. Баумана, обучающихся по направлению подготовки 09.03.04 «Программная инженерия».

## **ЦЕЛЬ И ЗАДАЧИ РАБОТЫ, ТРЕБОВАНИЯ К РЕЗУЛЬТАТАМ ЕЕ ВЫПОЛНЕНИЯ**

Целью выполнения домашней работы является формирование практических навыков создания алгоритмов решения оптимизационных задач.

Основными задачами выполнения домашней работы являются:

1. Изучить виды задач оптимизации при принятии решений.
2. Изучить основные алгоритмы для решения данных задач.
3. Реализовать алгоритм согласно варианту.

Результатами работы являются:

- Программа, реализующая индивидуальное задание
- Подготовленный отчет

## ЗАДАЧА О МАКСИМАЛЬНОМ ПОТОКЕ

Так же, как дорожную карту можно смоделировать ориентированным графом, чтобы найти кратчайший путь из одной точки в другую, ориентированным граф можно интерпретировать как некоторую транспортную сеть и использовать его для решения задач о потоках вещества в системе трубопроводов. Представим, что некоторый продукт передается по системе от источника, где данным продукт производится, к стоку, где он потребляется. Источник производит продукцию с некоторой постоянной скоростью, а сток с той же скоростью потребляет продукт. Интуитивно потоком продукта в любой точке системы является скорость движения продукта. С помощью транспортных сетей можно моделировать течение жидкостей по трубопроводам, движение деталей на сборочных линиях, передачу тока по электрическим сетям, информации — по информационным сетям и т.д.

Каждое ориентированное ребро сети можно рассматривать как канал, по которому движется продукт. Каждый канал имеет заданную пропускную способность, которая характеризует максимальную скорость перемещения продукта по каналу, например, 200 литров жидкости в минуту для трубопровода или 20 ампер для провода электрической цепи. Вершины являются точками пересечения каналов; через вершины, отличные от источника и стока, продукт проходит, не накапливаясь. Иными словами, скорость поступления продукта в вершину должна быть равна скорости его удаления из вершины. Это свойство называется свойством сохранения потока; в случае передачи тока по электрическим цепям ему соответствует закон Кирхгофа.

В задаче о максимальном потоке мы хотим найти максимальную скорость пересылки продукта от источника к стоку, при которой не будут нарушаться ограничения пропускной способности. Это одна из простейших задач, возникающих в транспортных сетях, и, как будет показано, существуют эффективные алгоритмы ее решения. Более того, основные методы, используемые в алгоритмах решения задач о максимальном потоке, можно применять для решения других задач, связанных с транспортными сетями.

## Транспортные сети и потоки

Транспортная сеть (flow network)  $G = (V, E)$  представляет собой ориентированный граф, в котором каждое ребро  $(u, v) \in E$  имеет неотрицательную пропускную способность (capacity)  $c(u, v) > 0$ . Если  $(u, v) \notin E$ , предполагается, что  $c(u, v) = 0$ . В транспортной сети выделяются две вершины: источник (source)  $s$  и сток (sink)  $t$ . Для удобства предполагается, что каждая вершина лежит на искомом пути из источника к стоку, т.е. для любой вершины  $v \in V$  существует путь  $s \rightarrow v \rightarrow t$ . Таким образом, граф является связным и  $|E| > |V| - 1$ . На рис. 1а показан пример транспортной сети  $G = (V, E)$  для задачи грузовых перевозок компании Lucky Truck. Источник  $s$  — это фабрика в Ванкувере, а сток  $t$  — склад в Виннипеге. Шайбы доставляются через промежуточные города, но за день из города и в город  $v$  можно отправить только  $c(u, v)$  ящиков. На рисунке приведена пропускная способность каждого ребра сети. На рис. 1б показаны потоки в транспортной сети. Поток  $f$  в сети  $G$  имеет значение  $|f| = 19$ . Показаны только положительные потоки. Если  $f(u, v) > 0$ , то ребро  $(u, v)$  снабжено меткой  $f(u, v)/c(u, v)$  (косая черта используется только для того, чтобы отделить транспортную сеть и ее потоки от пропускной способности и не обозначает деление).

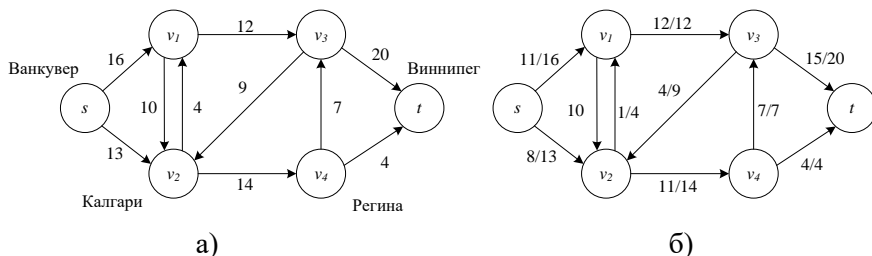


Рис. 1.

Если  $f(u, v) \leq 0$ , то у ребра  $(u, v)$  показана только его пропускная способность.

Теперь дадим формальное определение потоков. Пусть  $G = (V, E)$  — транспортная сеть с функцией пропускной способности  $c$ . Пусть  $s$  —

источник, а  $t$  — сток. **Потоком** (flow) в  $G$  является действительная функция  $f: V \times V \rightarrow R$ , удовлетворяющая следующим трем условиям.

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

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

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

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

т.е. как суммарный поток, выходящий из источника (в данном случае  $| \cdot |$  обозначает величину потока, а не абсолютную величину или мощность). В задаче о максимальном потоке (maximum flow problem) дана некоторая транспортная сеть  $G$  с источником  $s$  и стоком  $t$ , и необходимо найти поток максимальной величины.

Перед тем как рассматривать пример задачи о потоке в сети, кратко проанализируем три свойства потока. Ограничение пропускной способности предполагает, чтобы поток из одной вершины в другую не превышал заданную пропускную способность ребра. Антисимметричность введена для удобства обозначений 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 являющейся источником или стоком, равен нулю. Используя антисимметричность, можно записать свойство сохранения потока как

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

для всех  $u \in V - \{s, t\}$  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  $f(u, v) = f(v, u) = 0$ .

Последнее наблюдение касается свойств, связанных с положительными потоками. **Суммарный положительный поток** (total positive flow), входящий в вершину  $v$ , задается выражением

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

Суммарный положительный поток, выходящий из некоторой вершины, определяется симметрично. **Суммарный чистый поток** (total net flow) 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 входящий 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”.

### Пример потока

С помощью [транспортной сети](#) можно моделировать задачу о грузовых перевозках, представленную на рис. 1а. У компании Lucky Puck в Ванкувере есть фабрика (источник  $s$ ), производящая хоккейные шайбы, а в Виннипеге — склад (сток  $t$ ), где эти шайбы хранятся. Компания арендует места на грузовиках других фирм для доставки шайб с фабрики на склад. Поскольку грузовики ездят по определенным маршрутам (ребрам) между городами (вершинами) и имеют ограниченную грузоподъемность, компания Lucky Puck может перевозить не более  $c(u, v)$  ящиков в день между каждой парой городов  $u$  и  $v$ , как показано на рис. 1а. Компания Lucky Puck не может повлиять на маршруты и пропускную способность, т.е. она не может менять транспортную сеть, представленную на рис. 1а. Ее задача — определить, какое наибольшее количество  $p$  ящиков в день можно отгружать, и затем производить именно такое количество, поскольку не

имеет смысла производить шайб больше, чем можно отправить на склад. Для компании не важно, сколько времени займет доставка конкретной шайбы с фабрики на склад, она заботится только о том, чтобы  $p$  ящиков в день отправлялось с фабрики и  $p$  ящиков в день прибывало на склад.

На первый взгляд, вполне логично моделировать потоком в данной сети “поток” отгрузок, поскольку число ящиков, отгружаемых ежедневно из одного города в другой, подчиняется ограничению пропускной способности. Кроме того, должно соблюдаться условие сохранения потока, поскольку в стационарном состоянии скорость ввоза шайб в некоторый промежуточный город должна быть равна скорости их вывоза. В противном случае ящики станут накапливаться в промежуточных городах.

Однако между отгрузками и потоками существует одно отличие. Компания Lucky Puck может отправлять шайбы из Эдмонта в Калгари и одновременно из Калгари в Эдмонтон. Предположим, что 8 ящиков в день отгружается из Эдмонта ( $v_1$  на рис. 1) в Калгари ( $v_2$ ) и 3 ящика в день из Калгари в Эдмонтон. Как бы ни хотелось непосредственно представлять отгрузки потоками, мы не можем этого сделать. Согласно свойству антисимметричности, должно выполняться условие  $f(v_1, v_2) = -f(v_2, v_1)$  но это, очевидно, не так, если считать, что  $f(v_1, v_2) = 8$ , а  $f(v_2, v_1) = 3$ .

Компания Lucky Puck понимает, что бессмысленно отправлять 8 ящиков в день из Эдмонта в Калгари и 3 ящика из Калгари в Эдмонтон, если того же результата можно добиться, отгружая 5 ящиков из Эдмонта в Калгари и 0 ящиков из Калгари в Эдмонтон (что потребует к тому же меньших затрат). Этот последний сценарий мы и представим потоком:  $f(v_1, v_2) = 5$ , а  $f(v_2, v_1) = -5$ . По сути, 3 из 8 ящиков в день, отправляемые из  $v_1$  в  $v_2$ , взаимно уничтожаются (canceled) с 3 ящиками в день, отправляемыми из  $v_1$  в  $v_2$ .

В общем случае взаимное уничтожение позволяет нам представить отгрузки между двумя городами потоком, который положителен не более чем вдоль одного из двух ребер, соединяющих соответствующие вершины. Таким образом, любую ситуацию, когда ящики перевозятся

между двумя городами в обоих направлениях, можно привести с помощью взаимного уничтожения к эквивалентной ситуации, в которой ящики перевозятся только в одном направлении — направлении положительного потока.

По определенному таким образом потоку  $f$  нельзя восстановить, какими в действительности являются поставки. Если известно, что  $f(u, v) = 5$ , то это может быть как в случае, когда 5 единиц отправляется из  $u$  в  $v$ , так и в случае, когда 8 единиц отправляется из  $u$  в  $v$ , а 3 единицы из  $v$  в  $u$ . Обычно нас не будет интересовать, как в действительности организованы физические поставки, для любой пары вершин учитывается только чистое пересылаемое количество. Если важно знать объемы действительных поставок, следует использовать другую модель, в которой сохраняется информация о поставках в обоих направлениях.

Взаимное уничтожение неявно будет присутствовать в рассматриваемых алгоритмах. Предположим, что ребро  $(u, v)$  имеет величину потока  $f(u, v)$ . В процессе выполнения алгоритма мы можем увеличить поток вдоль ребра  $(v, u)$  на некоторую величину  $d$ . С математической точки зрения, эта операция должна уменьшить  $f(u, v)$  на  $d$ , следовательно, можно считать, что эти  $d$  единиц взаимно уничтожаются с  $d$  единицами потока, который уже существует вдоль ребра  $(u, v)$ .

### **Сети с несколькими источниками и стоками**

В задаче о максимальном потоке может быть несколько источников и стоков. Например, у компании Lucky Puck может быть  $m$  фабрик  $\{s_1, s_2, \dots, s_m\}$  и  $n$  складов  $\{t_1, t_2, \dots, t_n\}$  как показано на рис. 2а, где приведен пример транспортной сети с пятью источниками и тремя стоками. К счастью, эта задача не сложнее, чем обычная задача о максимальном потоке.

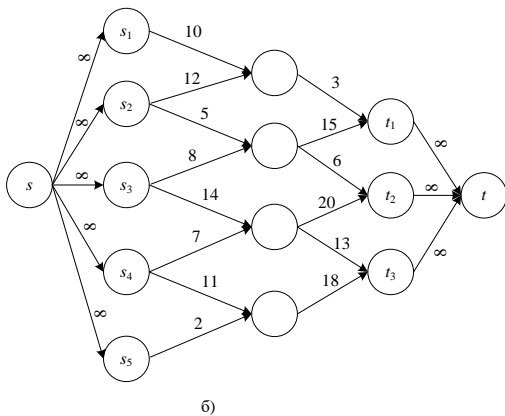
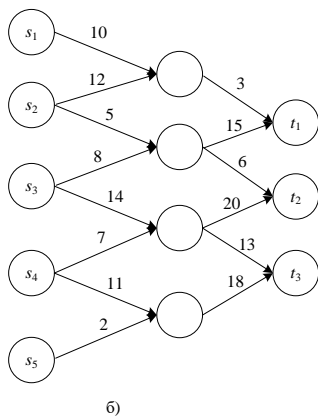


Рис. 2.

Задача определения максимального потока в сети с несколькими источниками и несколькими стоками сводится к обычной задаче о максимальном потоке. На рис. 26 показано, как сеть, представленную на рис. 2а, можно превратить в обычную транспортную сеть с одним источником и одним стоком. Для преобразования задачи о максимальном потоке с несколькими источниками и несколькими стоками к задаче с одним источником и одним стоком добавляется **фиктивный источник** (supersource)  $s$  и ориентированные ребра  $(s, s_i)$  с пропускной способностью  $c(s, s_i) = \infty$  для каждого  $i = 1, 2, \dots, m$ . Точно так же создается новый **фиктивный сток** (supresink)  $t$  и добавляются ориентированные ребра  $(t_i, t)$  с  $c(t_i, t) = \infty$  для каждого  $i = 1, 2, \dots, n$ . Интуитивно понятно, что любой поток в сети на рис. 2а соответствует потоку в сети на рис. 2б и наоборот. Единственный источник  $s$  просто обеспечивает поток любого требуемого объема к источникам  $s_i$ , а единственный сток  $t$  аналогичным образом потребляет поток любого желаемого объема от множественных стоков  $t_i$ .

### Как работать с потоками

Далее нам придется работать с функциями, подобными  $f$ , аргументами которых являются две вершины транспортной сети. Будем использовать **неявное обозначение суммирования** (implicit

summation notation), в котором один аргумент или оба могут представлять собой множество вершин, интерпретируя эту запись так, что указанное значение является суммой всех возможных способов замены аргументов их членами. Например, если  $X$  и  $Y$  — множеств вершин, то

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

Тогда условие сохранения потока можно записать как  $f(u, V) = 0$  для всех  $u \in V - \{s, t\}$ . Для удобства мы также обычно будем опускать фигурные скобки при использовании неявного обозначения суммирования: например, в уравнении  $f(s, V - s) = f(s, V)$  запись  $V - \{s\}$  обозначает множество  $V - \{s\}$ .

Использование неявного обозначения множеств обычно упрощает уравнения, описывающие потоки.

**Лемма 1.** Пусть  $G = (V, E)$  — транспортная сеть, и  $f$  — некоторый поток в сети  $G$ . Тогда справедливы следующие равенства.

Для всех  $X \subseteq V$   $f(X, X) = 0$

Для всех  $X, Y \subseteq V$   $f(X, Y) = -f(Y, X)$ .

Для всех  $X, Y, Z \subseteq V$ , таких что  $X \cap Y = \emptyset$ ,  
 $f(X \cup Y, Z) = f(X, Z) + f(Y, Z)$  и  $f(Z, X \cup Y) = f(Z, X) + f(Z, Y)$

Интуитивно мы ожидаем, что это свойство справедливо. Согласно условию сохранения потока, все вершины, отличные от источника и стока, имеют одинаковые величины входящего и выходящего положительного потока. Источник по определению имеет суммарный чистый поток, больший 0; т.е. из источника выходит больший положительный поток, чем входит в него. Симметрично, сток является единственной вершиной, которая может иметь суммарный чистый поток, меньший 0; т.е. в сток входит больший положительный поток, чем выходит из него.

### Метод Форда-Фалкерсона

Метод Форда-Фалкерсона 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 поиска “увеличивающего пути” (т.е. некоего пути от источника  $s$  к стоку  $t$ , вдоль которого можно послать больший поток) и последующего увеличения потока. Этот процесс повторяется до тех пор, пока уже невозможно отыскать увеличивающий путь.

`Ford_Fulkerson_Method(G,s,t)`

1. Задаем начальное значение потока  $f$  равным 0
2. While (пока) существует увеличивающий путь  $p$
3.     do увеличиваем поток  $f$  вдоль пути  $p$
4. return  $f$

### Остаточные сети

Интуитивно понятно, что если заданы некоторая транспортная сеть и поток, то остаточная сеть — это сеть, состоящая из ребер, допускающих увеличение потока. Более строго, пусть задана транспортная сеть  $G = (V, E)$  с источником  $s$  и стоком  $t$ . Пусть  $f$  — некоторый поток в  $G$ . Рассмотрим пару вершин  $u, v \in V$ . Величина дополнительного потока, который мы можем направить из  $u$  в  $v$ , не превысив пропускную способность  $c(u, v)$ , является остаточной пропускной способностью (residual capacity) ребра  $(u, v)$ , и задается формулой

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

Например, если  $c(u, v) = 16$  и  $f(u, v) = 11$ , то  $f(u, v)$  можно увеличить на  $c_f(u, v) = 5$ , не нарушив ограничение пропускной способности для ребра  $(u, v)$ . Когда поток  $f(u, v)$  отрицателен, остаточная пропускная способность  $c_f(u, v)$  больше, чем пропускная способность  $c(u, v)$ . Так, если  $c(u, v) = 16$  и  $f(u, v) = -4$ , то остаточная пропускная способность  $c_f(u, v) = 20$ . Эту ситуацию можно интерпретировать следующим образом: существует поток величиной 4 единицы из  $v$  в  $u$ , который можно аннулировать, направив 4 единицы потока из  $u$  в  $v$ . Затем можно направить еще 16 единиц из  $u$  в  $v$ , не нарушив ограничение пропускной способности для ребра  $(u, v)$ . Таким образом, начиная с потока  $f(u, v) = -4$ , мы направили дополнительные 20 единиц потока, прежде чем достигли ограничения пропускной

способности.

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

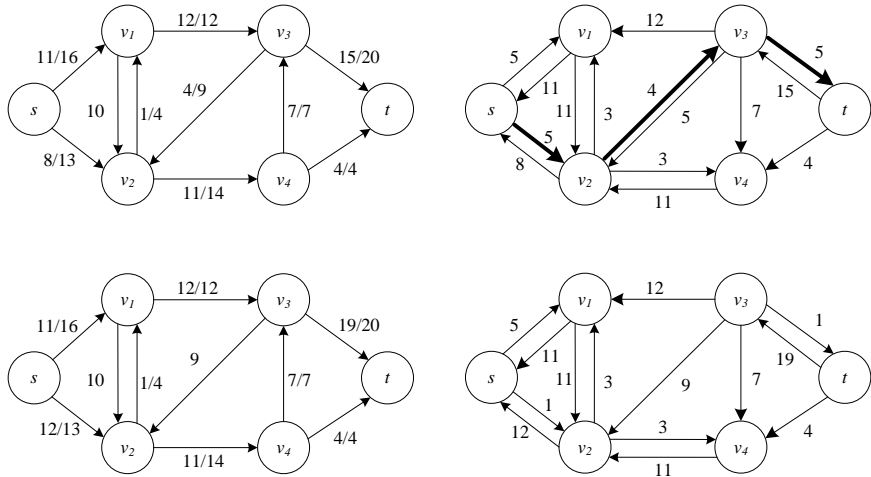


Рис. 3 а) Транспортная сеть  $G$  и поток  $f$ , представленные на рис. 1б  
 б) Остаточная сеть  $G_f$  с выделенным увеличивающим путем; его остаточная пропускная способность равна 4. в) Поток в сети  $G$ , полученный в результате увеличения потока вдоль пути  $p$  на величину его остаточной пропускной способности 4. г) Остаточная сеть, порожденная потоком, показанным в части в)

Таким образом, как и отмечалось выше, по каждому ребру остаточной сети, или **остаточному ребру** (residual edge), можно направить поток, больший 0. На рис. 3а воспроизведены транспортная сеть  $G$  и поток  $f$ , представленные на рис. 1б, а на рис. 3б показана соответствующая остаточная сеть  $G_f$ .

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

некоторого ребра  $(u, v) \in E$ , то  $f(u, v) < 0$ . В таком случае  $c_f(v, u) = c(v, u) - f(v, u) > 0$  и, следовательно,  $(u, v) \in E_f$ . Если в сети нет ни ребра  $(u, v)$ , ни  $(v, u)$ , то с  $(u, v) = c(v, u) = 0, f(u, v) = f(v, u) = 0$ , и  $c_f(u, v) = c_f(v, u)$ . Таким образом, можно сделать вывод, что ребро  $(u, v)$  может оказаться в остаточной сети только в том случае, если хотя бы одно из ребер  $(u, v)$  или  $(v, u)$  присутствует в исходной транспортной сети, поэтому

$$|E_f| \leq 2 |E|$$

Обратите внимание, что остаточная сеть  $G_f$  является транспортной сетью со значением пропускных способностей, заданными  $c_f$ . Следующая лемма показывает, как поток в остаточной сети связан с потоком в исходной транспортной сети.

**Лемма 2.** Пусть  $G = (V, E)$  - транспортная сеть с источником  $s$  и стоком  $t$ ,  $a, f$  - поток в  $G$ . Пусть  $G_f$  - остаточная сеть в  $G$ , порожденная потоком  $f$ ,  $a, f'$  - поток и  $G_f$ . Тогда сумма потоков  $f + f'$ , является потоком в  $G$ , и величина этого потока равна  $|f + f'| = |f| + |f'|$ .

### Увеличивающие пути

Для заданных транспортной сети  $G = (V, E)$  и потока  $f$  увеличивающим путем (augmenting path)  $p$  является простой путь из  $s$  в  $t$ ; в остаточной сети  $G_f$ .

Согласно определению остаточной сети, каждое ребро  $(u, v)$  увеличивающего пути допускает некоторый дополнительный положительный поток из  $u$  в  $v$  без нарушения ограничения пропускной способности для данного ребра.

Выделенный путь на рис. 3б является увеличивающим путем. Рассматривая представленную на рисунке остаточную сеть  $G_f$  как некоторую транспортную сеть, можно увеличивать поток вдоль каждого ребра данного пути вплоть до 4 единиц, не нарушая ограничений пропускной способности, поскольку наименьшая остаточная пропускная способность на данном пути составляет  $c_f(v_2, v_3) = 4$ . Максимальная величина, на которую можно увеличить поток вдоль каждого ребра увеличивающего пути  $p$ , называется



**остаточной пропускной способностью** (residual capacity)  $p$  и задается формулой

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

**Лемма 3.** Пусть  $G = (V, E)$  — транспортная сеть,  $f$  — некоторый поток в  $G$ , и пусть  $p$  — некоторый увеличивающий путь в  $G_f$ . Определим функцию  $f_p : V \times V \rightarrow R$  следующим образом:

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

Тогда  $f_p$  является потоком в  $G$  и его величина составляет  $|f_p| = c_f(p) > 0$ . Вытекающее из данной леммы следствие показывает, что если добавить  $f_p$  к  $f$ , то мы получим новый поток в  $G$ , величина которого ближе к максимальной. На рис. 3в показан результат добавления  $f_p$ , представленного на рис. 3б, к показанному на рис. 3а.

**Следствие.** Пусть  $G = (V, E)$  — транспортная сеть,  $f$  — некоторый поток в  $G$ , и пусть  $p$  — некоторый увеличивающий путь в  $G_f$ . Определим функцию  $f' : V \times V \rightarrow R$  как  $f' = f + f_p$ . Тогда  $f'$  является потоком в  $G$  и имеет величину  $|f'| = |f| + |f_p| > |f|$

### Разрезы транспортных сетей

В методе Форда-Фалкерсона производится неоднократное увеличение потока вдоль увеличивающих путей до тех пор, пока не будет найден максимальный поток. В теореме о максимальном потоке и минимальном разрезе, утверждается, что поток является максимальным тогда и только тогда, когда его остаточная сеть не содержит увеличивающих путей. Однако для доказательства данной теоремы нам понадобится ввести понятие разреза транспортных сетей.

**Разрезом** (cut)  $(S, T)$  транспортной сети  $G = (V, E)$  называется разбиение множеств вершин на множества  $S$  и  $T = V - S$ , такие что  $s \in S$ , а  $t \in T$ . Если  $f$  — поток, то **чистый поток** (net flow) через разрез  $(S, T)$  по определению равен  $f(S, T)$ . **Пропускной способностью** (capacity) разреза  $(S, T)$  является  $c(S, T)$ . **Минимальным разрезом** (minimum cut) сети является разрез, пропускная способность которого

среди всех разрезов сети минимальна.

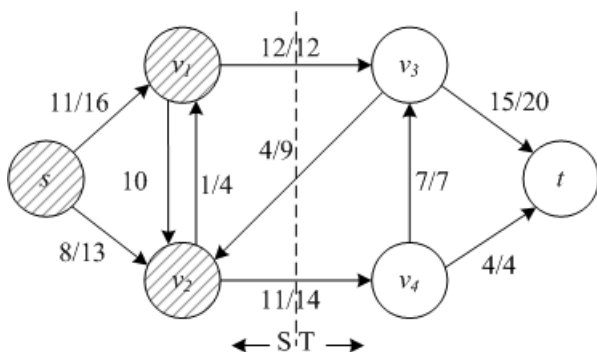
На рис. 4 показан разрез  $(\{s, v_1, v_2\}, \{v_3, v_4, t\})$  транспортной сети, представленной на рис. 16. Чистый поток через данный разрез равен

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

а пропускная способность этого разреза равна

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

Рис. 4. Разрез  $(S, T)$  транспортной сети, представленной на рис. 16,



$S = \{s, v_1, v_2\}, T = \{v_3, v_4, t\}$ . Вершины, принадлежащие  $S$ , отмечены черным цветом, а вершины  $T$  — белым

Обратите внимание, что чистый поток через разрез может включать в себя отрицательные потоки между вершинами, но пропускная способность разреза складывается исключительно из неотрицательных значений. Иными словами, чистый поток через разрез  $(S, T)$  составляется из положительных потоков в обоих направлениях: положительный поток из  $S$  в  $T$  прибавляется, а положительный поток из  $T$  в  $S$  вычитается. С другой стороны, пропускная способность разреза  $(S, T)$  вычисляется только по ребрам, идущим из  $S$  в  $T$ . Ребра, ведущие из  $T$  в  $S$ , не участвуют в вычислении  $c(S, T)$ .

Следующая лемма показывает, что чистый поток через любой разрез одинаков и равен величине потока.

**Лемма 4.** Пусть  $f$  — некоторый поток в транспортной сети  $G$  с источником  $s$  и стоком  $t$ , и пусть  $(S, T)$  — разрез  $G$ . Тогда чистый

поток через  $(S, T)$  равен  $f(S, T) = |f|$ .

Непосредственным следствием леммы 4 является доказанный ранее результат — величина потока равна суммарному потку, входящему в сток.

Другое следствие леммы 4 показывает, как пропускные способности разрезов можно использовать для определения границы величины потока.

**Следствие.** Величина любого потока  $f$  в транспортной сети  $G$  не превышает пропускную способность произвольного разреза  $G$ .

Непосредственно из следствия вытекает, что максимальный поток в сети не превышает пропускной способности минимального разреза. Сейчас мы сформулируем и докажем важную теорему о максимальном потоке и минимальном разрезе, в которой утверждается, что значение максимального потока равно пропускной способности минимального разреза.

**Теорема (О максимальном потоке и минимальном разрезе).** Если  $f$  - некоторый поток в транспортной сети  $G = (V, E)$  с источником  $s$  и стоком  $t$  следующие утверждения эквивалентны.

1.  $f$  — максимальный поток в  $G$
2. Остаточная сеть  $G_f$  не содержит увеличивающих путей
3.  $|f| = c(S, T)$  для некоторого разреза  $(S, T)$  сети  $G$

**Доказательство.** (1)  $\Rightarrow$  (2): Предположим противное: пусть  $f$  является максимальным потоком в  $G$ , но  $G_f$  содержит увеличивающий путь  $p$ . Согласно следствию, сумма потоков  $f + fp$ , является потоком в  $G$ , величина которого строго больше, чем  $|f|$ , что противоречит предположению, что  $f$  — максимальный поток.

(2)  $\Rightarrow$  (3): Предположим, что  $G_f$  не содержит увеличивающего пути, т.е.  $G_f$  не содержит пути из  $s$  в  $t$ . Определим

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

и  $T = V - S$ . Разбиение  $(S, T)$  является разрезом: очевидно, что  $s \in S$ ,  $t \in T$  поскольку в  $G_f$  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) $\Rightarrow$  (1): Согласно следствию,  $|f| \leq c(S, T)$  для всех разрезв  $(S, T)$ , поэтому из условия  $|f| = c(S, T)$  следует, что  $f$  — максимальный поток.

### **Базовый алгоритм Форда-Фалкерсона**

При выполнении каждой итерации [метода Форда-Фалкерсона](#) мы находим некоторый [увеличивающий путь](#)  $p$ , и поток  $f$  вдоль каждого ребра данного пути увеличивается на величину остаточной пропускной способности  $c_f(p)$ . Приведенная далее реализация данного метода вычисляет максимальный поток в графе  $G=(V, E)$  путем обновления потока  $f[u, v]$  между каждой парой вершин  $u$  и  $v$  соединенных ребром. Если вершины  $u$  и  $v$  не связаны ребром 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  $(u, v) \notin E$ . В коде процедуры  $c_f(p)$  в действительности является просто временной переменной, в которой хранится остаточная пропускная способность пути  $p$ .

Ford\_Fulkerson( $G, s, t$ )

1. for(для) каждого ребра  $(u, v) \in E[G]$
2.     do  $f[u, v] \leftarrow 0$
3.      $f[v, u] \leftarrow 0$
4. while(пока) существует путь  $p$  из  $s$  в  $t$  в остаточной сети  $G_f$
5.     do  $c_f(p) \leftarrow \min\{c_f(u, v) : (u, v) \text{ принадлежит } p\}$
6.     for(для) каждого ребра  $(u, v)$  in  $p$
7.         do  $f[u, v] \leftarrow f[u, v] + c_f(p)$
8.          $f[u, v] \leftarrow -f[u, v]$

Приведенный псевдокод алгоритма FORD\_FULKERSON является расширением приведенного ранее псевдокода FORD\_FULKERSON\_METHOD. На рис. 5 пока результаты каждой итерации при тестовом выполнении. Строки 1-3 инициализируют поток  $f$  значением 0. В цикле while в строках 4-8 выполняется неоднократный поиск увеличивающего пути  $p$  в  $G_f$  и поток  $f$  вдоль пути  $p$  увеличивается на остаточную пропускную способность  $c_f(p)$ . Когда

увеличивающих путей больше нет, поток  $f$  является максимальным. Остаточная сеть на рис. 5а — это исходная сеть  $G$ ; поток  $f$ , показанный на рис. 5д, является максимальным потоком.

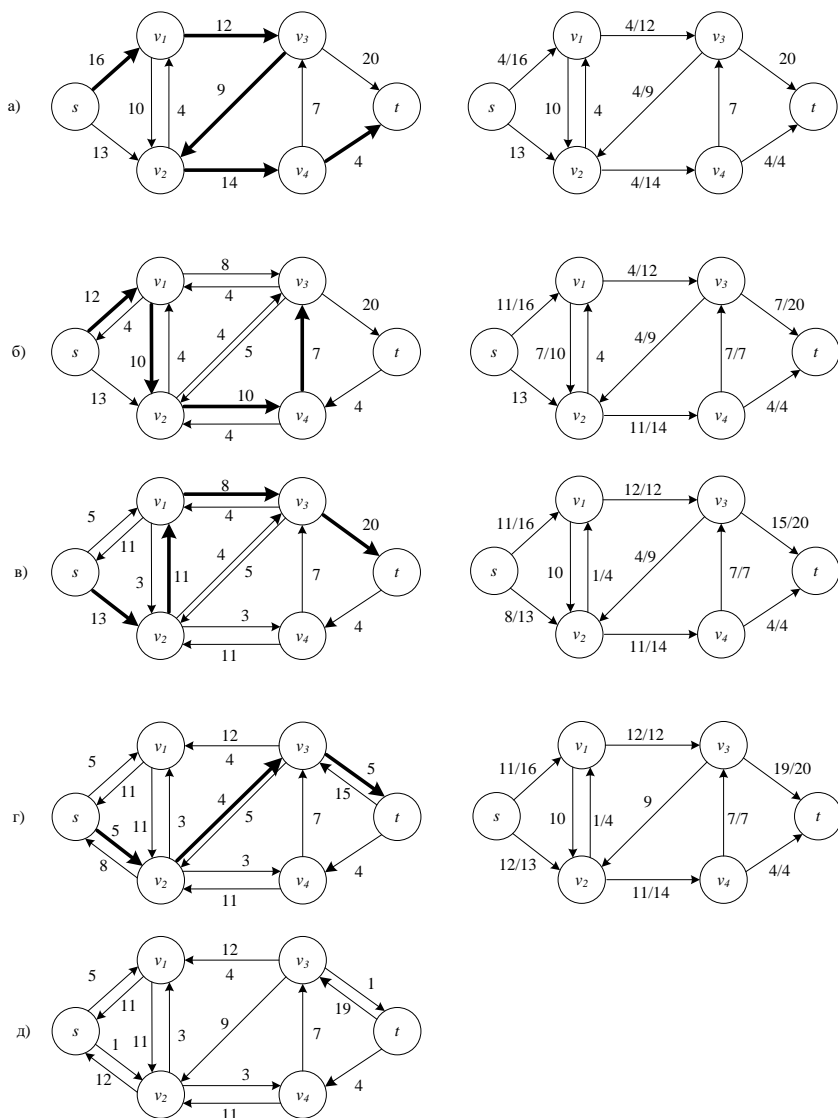


Рис. 5. Работа базового алгоритма Форда-Фалкерсона (последовательные итерации цикла while). В левой части каждого

рисунка показана остаточная сеть  $G_f$  с выделенным увеличивающим путем  $p$ : в правой части показан новый поток  $f$ , который получается в результате прибавления  $f_p$  к  $f$

### Анализ метода Форда-Фалкерсона

Время выполнения процедуры FORD\_FULKERSON зависит от того, как именно выполняется поиск увеличивающего пути  $p$  в строке 4. При неудачном методе поиска алгоритм может даже не завершиться: величина потока будет последовательно увеличиваться, но она не обязательно сходится к максимальному значению потока  $f$ . Если увеличивающий путь выбирается с использованием поиска в ширину, алгоритм выполняется за полиномиальное время.

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. В таком предположении непосредственная реализация процедуры FORD\_FULKERSON имеет время работы  $O(E |f^*|)$ , где  $f^*$  — максимальный поток, найденный данным алгоритмом. Анализ проводится следующим образом. Выполнение строк 1-3 занимает время  $O(E)$ . Цикл while в строках 4-8 выполняется не более  $|f^*|$  раз, поскольку величина потока за каждую итерацию увеличивается по крайней мере на одну единицу.

Работ внутри цикла while зависит от того, насколько эффективно организовано управление структурой данных, используемой для реализации сети  $G = (V, E)$ . Предположим, что мы поддерживаем структуру данных, соответствующую ориентированному графу  $G' = (V, E')$ , где  $G' = (V, E')$ , где  $E' = \{(u, v) : (u, v) \in E \text{ или } (v, u) \in E\}$ .

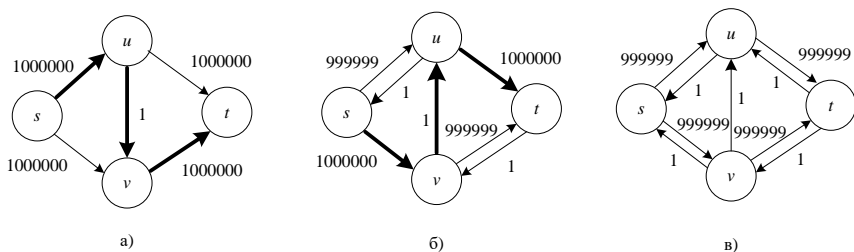


Рис. 6. Транспортная сеть, для которой выполнение процедуры FOKDJFULKERSON может занимать время  $O(E |f^*|)$ ,  $|f^*| = 2\,000\,000$

Ребра сети  $G$  являются также ребрами графа  $G'$ , поэтому в этой структуре данных можно довольно легко хранить пропускные способности и потоки. Для данного потока  $f$  в  $G$ , ребра остаточной сети  $Gf$  состоят из всех ребер  $(u, v)$  графа  $G'$ , таких что  $c(u, v) - f[u, v] \neq 0$ . Таким образом, время поиска пути в остаточной сети составляет  $O(V + E) = O(E)$ , если используется только поиск в глубину или поиск в ширину. Каждая итерация цикла `while` занимает время  $O(E)$ , так что в результате общее время выполнения процедуры FORD\_FULKERSON составляет  $O(E |f^*|)$ .

Когда значения пропускных способностей являются целыми числами и оптимальное значение потока  $|f^*|$  невелико, время выполнения алгоритма Форда-Фалкерсона достаточно неплохое. Но на рис. 6а показан пример того, что может произойти в простой транспортной сети, с большим значением  $|f^*|$ . Величина максимального потока в данной сети равна  $2\,000\,000$ :  $1\,000\,000$  единиц потока идет по пути  $s \rightarrow v \rightarrow t$ , а другие  $1\,000\,000$  единиц идут по пути  $s \rightarrow v \rightarrow t$ . Если первым увеличивающим путем, найденным процедурой FORD\_FULKERSON, является путь  $s \rightarrow u \rightarrow v \rightarrow t$ , как показано на рис. 6а, поток после первой итерации имеет значение 1. Полученная остаточная сеть показана на рис. 6б. Если в ходе выполнения второй итерации найден увеличивающий путь  $s \rightarrow v \rightarrow u \rightarrow t$ , как показано на рис. 6б, поток станет равным 2. На рис. 6в показана соответствующая остаточная сеть. Можно продолжать процедуру, выбирая увеличивающий путь  $s \rightarrow v \rightarrow u \rightarrow t$  для итераций с нечетным номером  $s \rightarrow v \rightarrow u \rightarrow t$  для итераций с четным номером. В таком случае нам придется выполнить  $2\,000\,000$  увеличений, при этом величина Потока на каждом шаге увеличивается всего на 1 единицу.

### **Алгоритм Эдмондса-Карпа**

Указанный недостаток метода Форда-Фалкерсона можно преодолеть, если реализовать вычисление увеличивающего пути  $p$  в строке 4 как поиск в ширину, т.е. если в качестве увеличивающего пути выбирается кратчайший путь из  $s$  в  $t$  в остаточной сети, где каждое ребро имеет единичную длину (вес). Такая реализация метода Форда-Фалкерсона называется **алгоритмом Эдмондса-Карпа** (Edmonds-Karp algorithm). Докажем, что время выполнения алгоритма Эдмондса-Карпа составляет  $O(V E^2)$ .



## ЗАДАЧА О МАКСИМАЛЬНОМ ПАРОСОЧЕТАНИИ

Пусть дан неориентированный граф  $G = (V, E)$ . Паросочетанием (matching) называется подмножество ребер  $M \subset E$ , такое что для всех вершин  $v \in V$  в  $M$  содержится не более одного ребра, инцидентного  $v$ . Мы говорим, что вершина  $v \in V$  является связанной (matched) паросочетанием  $M$ , если в  $M$  есть ребро, инцидентное  $v$ ; в противном случае вершина  $v$  называется открытой (unmatched). Максимальным паросочетанием называется паросочетание максимальной мощности, т.е. такое паросочетание  $M$ , что для любого паросочетания  $M' |M| \geq |M'|$ . Мы предполагаем, что множество вершин можно разбить на два подмножества  $V = L \cup R$ , где  $L$  и  $R$  не пересекаются, и все ребра из  $E$  проходят между  $L$  и  $R$ . Далее мы предполагаем, что каждая вершина из  $V$  имеет по крайней мере одно инцидентное ребро. Иллюстрация понятия паросочетания показана на рис. 7.

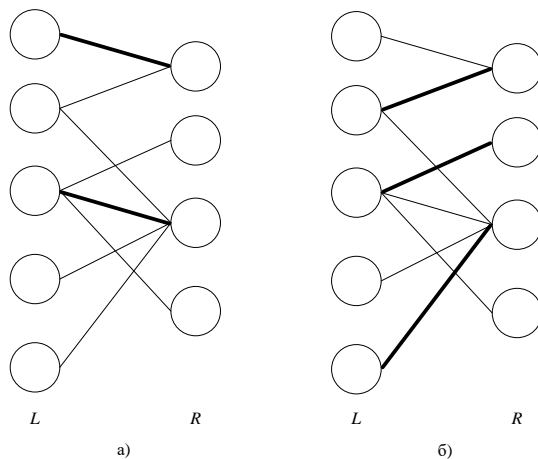


Рис. 7. Двудольный граф  $G = (V, E)$  с разбиением вершин  $V = L \cup R$ .  
а) Паросочетание с мощностью 2. б) Максимальное паросочетание с мощностью 3

Задача поиска максимального паросочетания в двудольном графе имеет множество практических приложений. В качестве примера можно рассмотреть паросочетание множества машин  $L$  и множества задач  $R$ ,

которые должны выполняться одновременно. Наличие в  $E$  ребра  $(u, v)$  означает, что машина  $u \in L$  может выполнять задачу  $v \in R$ . Максимальное паросочетание обеспечивает максимальную загрузку машин.

### Поиск максимального паросочетания в двудольном графе

С помощью [метода Форда-Фалкерсона](#) можно найти максимальное паросочетание в неориентированном двудольном графе  $G = (V, E)$  за время, полиномиально зависящее от  $|V|$  и  $|E|$ . Проблема состоит в том, чтобы построить транспортную сеть, потоки в которой соответствуют паросочетаниям, как показано на рис. 8. Определим для заданного двудольного графа  $G$  соответствующую транспортную сеть  $G' = (V', E')$  следующим образом. Возьмем в качестве источника  $s$  и стока  $t$  новые вершины, не входящие в  $V$ , и пусть  $V' = V \cup \{s, t\}$ . Если разбиение вершин в графе  $G$  задано как  $V = L \cup R$ , ориентированными ребрами  $G'$  будут ребра  $E$ , направленные из  $L$  в  $R$ , а также  $|V|$  новых ребер  $E' = \{(s, u) : u \in L\} \cup \{(u, v) : u \in L, v \in R, (u, v) \in E\} \cup \{(v, t) : v \in R\}$ . Чтобы завершить построение, присвоим каждому ребру  $E'$  единичную пропускную способность. Поскольку каждая вершина из множества вершин  $V$  имеет по крайней мере одно инцидентное ребро,  $|E| \geq |V|/2$ . Таким образом  $|E| \leq |E'| = |E| + |V| \leq 3|E|$ , так что  $|E'| = O(E)$ .

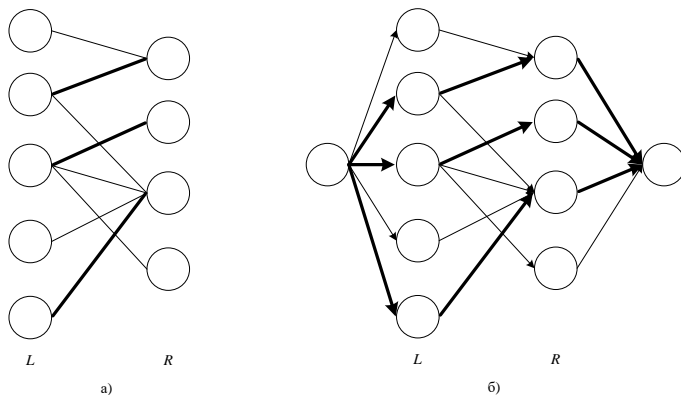


Рис. 8. Двудольный граф (а) и соответствующая ему транспортная сеть (б). Выделенные ребра обеспечивают максимальный поток и определяют максимальное паросочетание

Следующая лемма показывает, что паросочетание в  $G$  непосредственно соответствует некоторому потоку в соответствующей транспортной сети  $G'$ . Поток  $f$  в транспортной сети  $G = (V, E)$  называется целочисленным (integer-valued), если значения  $f(u, v)$  целые для всех  $(u, v) \in V \times V$ .

**Лемма 5.** Пусть  $G = (V, E)$  — двудольный граф с разбиением вершин  $V = L \cup R$ , и пусть  $G' = (V', E')$  — соответствующая ему транспортная сеть. Если  $M$  — паросочетание в  $G$ , то существует целочисленный поток  $f$  в  $G'$ , величина которого  $|f| = |M|$ . Справедливо и обратное утверждение: если  $f$  — целочисленный поток к  $G'$ , то в  $G$  существует паросочетание  $M$  с мощностью  $|M| = |f|$ .

**Доказательство.** Покажем сначала, что паросочетанию  $M$  в графе  $G$  соответствует некоторый целочисленный поток  $f$  в сети  $G'$ . Определим  $f$  следующим образом. Если  $(u, v) \in M$ , то  $f(s, u) = f(u, v) = f(v, t) = 1$  и  $f(u, s) = f(v, u) = f(t, v) = -1$ . Для всех остальных ребер  $(u, v) \in E'$  определим  $f(u, v) = 0$ . Нетрудно убедиться, что  $f$  обладает свойством антисимметричности, удовлетворяет ограничениям пропускной способности и сохранения потока.

Интуитивно понятно, что каждое ребро  $(u, v) \in M$  соответствует единице потока в (проходящего по маршруту  $s \rightarrow u \rightarrow v \rightarrow t$ ). Кроме того, пути, порожденные ребрами из  $M$ , представляют собой непересекающиеся множества, не считая  $s$  и  $t$ . Чистый поток через разрез  $(L \cup \{s\}, R \cup \{t\})$  равен  $|M|$ ; следовательно, величина потока равна  $|f| = |M|$ .

Чтобы доказать обратное, предположим, что  $f$  — некоторый целочисленный поток  $G'$ , и пусть  $M = \{(u, v) : u \in L, v \in R, f(u, v) > 0\}$

Каждая вершина  $u \in L$  имеет только одно входящее ребро, а именно  $(s, u)$ , и его пропускная способность равна 1. Следовательно, в каждую вершину  $u \in L$  входит не более одной единицы положительного потока, и если она действительно входит, то из нее должна также выходить одна единица положительного потока согласно свойству сохранения потока. Более того, поскольку  $f$  — целочисленный поток, для каждой вершины  $u \in L$  одна единица потока может входить

не более чем по одному ребру и выходить не более чем по одному ребру. Таким образом, одна единица положительного потока входит в  $u$  тогда и только тогда, когда существует в точности одна вершина  $v \in R$ , такая что  $f(u, v) = 1$ , и из каждой вершины  $u \in L$  выходит не более одного ребра, несущего положительный поток. Симметричные рассуждения применимы для каждой вершины  $u \in L$ . Следовательно,  $M$  является паросочетанием.

Чтобы показать, что  $|M| = |f|$  заметим, что  $f(s, u) = 1$  для каждой связанной вершины  $u \in L$ , и  $f(u, v) = 0$  для каждого ребра  $(u, v) \in E - M$ . Следовательно,  $|M| = f(L, R) = f(L, V') - f(L, L) - f(L, s) - f(L, t)$ .

Можно сделать вывод, что максимальное паросочетание в двудольном графе  $G$  соответствует максимальному потоку в соответствующей ему транспортной сети  $G'$ , следовательно, можно находить максимальное паросочетание в  $G$  с помощью алгоритма поиска максимального потока в  $G'$ . Единственной проблемой в данных рассуждениях является то, что алгоритм поиска максимального потока может вернуть такой поток в  $G'$ , в котором некоторое значение  $f(u, v)$  оказывается нецелым, несмотря на то, что величина  $|f|$  должна быть целой. Следующая теорема показывает, что такая проблема не может возникнуть при использовании метода Форда-Фалкерсона.

**Теорема (Теорема о целочисленности).** Если функция пропускной способности  $c$  принимает только целые значения, то максимальный поток  $f$ , полученный с помощью метода Форда-Фалкерсона, обладает тем свойством, что значение потока  $|f|$  является целочисленным. Более того, для всех вершин  $u$  и  $v$  величина  $f(u, v)$  является целой.

Доказательство проводится индукцией по числу итераций.

**Следствие.** Мощность максимального паросочетания  $M$  в двудольном графе  $G$  равна величине максимального потока  $f$  в соответствующей транспортной сети  $G'$ .

**Доказательство.** Предположим, что  $M$  - максимальное паросочетание в  $G$ , но соответствующий ему поток  $f$  в  $G'$  не максимален. Тогда в  $G'$  существует максимальный поток  $f'$ , такой что  $|f'| > |f|$ . Поскольку пропускные способности в  $G'$  являются целочисленными, теорема о целочисленности позволяет сделать вывод,

что поток  $f$  -целочисленный. Следовательно,  $f$  соответствует некоторому паросочетанию  $M'$  в  $G$  с мощностью  $|M'| = |f'| > |f| = |M|$ , что противоречит нашему предположению о том, что  $M$  — максимальное паросочетание. Аналогично можно показать, что если  $f$  - максимальный поток в  $G'$ , то соответствующее ему паросочетание является максимальным паросочетанием в  $G$ .

Таким образом, для заданного неориентированного двудольного графа  $G$  можно найти максимальное паросочетание путем создания транспортной сети  $G'$ , применения метода Форда-Фалкерсона и непосредственного получения максимального паросочетания  $M$  по найденному максимальному целочисленному потоку  $f$ , поскольку любое паросочетание в двудольном графе имеет мощность не более  $\min(L, R) = O(V)$ , величина максимального потока в  $G'$  составляет  $O(V)$ . Поэтому максимальное паросочетание в двудольном графе можно найти за время  $O(VE') = O(VE)$  поскольку  $|E'| = \Theta(E)$ .

## **ЗАДАНИЕ ДЛЯ ДОМАШНЕЙ РАБОТЫ**

Создать программу согласно полученному варианту. При выполнении домашней работы запрещается использовать сторонние классы и компоненты, реализующие заявленную функциональность.

### **ВАРИАНТЫ ЗАДАНИЙ**

1. Реализовать алгоритм поиска максимального потока с одним истоком и одним стоком на основе алгоритма поиска в ширину.
2. Реализовать алгоритм поиска максимального потока с одним истоком и одним стоком на основе алгоритма поиска в глубину.
3. Реализовать алгоритм поиска максимального потока с множеством истоков и одним стоком на основе алгоритма поиска в ширину.
4. Реализовать алгоритм поиска максимального потока с множеством истоков и одним стоком на основе алгоритма поиска в глубину.
5. Реализовать алгоритм поиска максимального потока с одним истоком и множеством стоков на основе алгоритма поиска в ширину.
6. Реализовать алгоритм поиска максимального потока с одним истоком и множеством стоков на основе алгоритма поиска в глубину.
7. Реализовать алгоритм поиска максимального потока с множеством истоков и множеством стоков на основе алгоритма поиска в ширину.
8. Реализовать алгоритм поиска максимального потока с множеством истоков и множеством стоков на основе алгоритма поиска в глубину.
9. Реализовать алгоритм поиска максимального паросочетания на основе алгоритма поиска в ширину.
10. Реализовать алгоритм поиска максимального паросочетания на основе алгоритма поиска в глубину.

## **КОНТРОЛЬНЫЕ ВОПРОСЫ И ЗАДАНИЯ**

1. Сформулируйте определение транспортной сети.
2. Сформулируйте определение разреза транспортной сети.
3. Сформулируйте определение потока в графе.
4. Раскройте понятия «сток» и «исток».
5. Опишите базовый алгоритм Форда-Фалкерсона.
6. Дайте определение увеличивающему пути.
7. Раскройте алгоритм поиска максимального потока с несколькими истоками и стоками.
8. Опишите алгоритм Эдмондса-Карпа.
9. Сформулируйте определение паросочетания в графе.
10. Раскройте алгоритм поиска максимального паросочетания.

## **ФОРМА ОТЧЕТА ПО ДОМАШНЕЙ РАБОТЕ**

На выполнение домашней работы отводится 6 занятий (12 академических часа: 11 часов на выполнение и сдачу практического задания и 1 час на подготовку отчета).

Номер варианта студенту выдается преподавателем.

Отчет на защиту предоставляется в печатном виде.

Структура отчета (на отдельном листе(-ах)): титульный лист, цель и задачи работы, формулировка задания, этапы выполнения работы, результаты выполнения, выводы.

## ОСНОВНАЯ ЛИТЕРАТУРА

1. Алексеев В.Е. Графы и алгоритмы. Структуры данных. Модели вычислений [Электронный ресурс]/ В.Е. Алексеев, В.А. Таланов. — Электрон. текстовые данные. — М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 153 с. — Режим доступа: <http://www.iprbookshop.ru/52186.html>
2. Вирт Никлаус. Алгоритмы и структуры данных [Электронный ресурс]/ Никлаус Вирт— Электрон. текстовые данные. — Саратов: Профобразование, 2017. — 272 с.— Режим доступа: <http://www.iprbookshop.ru/63821.html>
3. Самуйлов С.В. Алгоритмы и структуры обработки данных [Электронный ресурс]: учебное пособие/ С.В. Самуйлов. — Электрон. текстовые данные. — Саратов: Вузовское образование, 2016. — 132 с.— Режим доступа: <http://www.iprbookshop.ru/47275.html>

## ДОПОЛНИТЕЛЬНАЯ ЛИТЕРАТУРА

4. Костюкова Н.И. Графы и их применение [Электронный ресурс]/ Н.И. Костюкова. — Электрон. текстовые данные. — М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 147 с. — Режим доступа: <http://www.iprbookshop.ru/52185.html>
5. Сундукова Т.О. Структуры и алгоритмы компьютерной обработки данных [Электронный ресурс]/ Т.О. Сундукова, Г.В. Ваныкина. — Электрон. текстовые данные. — М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. — 749 с.— Режим доступа: <http://www.iprbookshop.ru/57384.html>



## Электронные ресурсы:

6. Научная электронная библиотека <http://elibrary.ru>
7. Электронно-библиотечная система «ЛАНЬ»  
<http://e.lanbook.com>
8. Электронно-библиотечная система «IPRbooks»  
<http://www.iprbookshop.ru>
9. Электронно-библиотечная система «Юрайт» <http://www.biblio-online.ru>
10. Электронно-библиотечная система «Университетская библиотека ONLINE» <http://www.biblioclub.ru>