Учреждение образования
«Полоцкий государственный университет»

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

к практической подготовке

по дисциплине «*Дискретная математика*»

для студентов заочного отделения
машиностроительного факультета

Кафедра «Высшей математики»

И.П.Кунцевич

Новополоцк
2013

Дисциплина «Дискретная математика» является спецкурсом, позволяющему ознакомится студентам машиностроительного факультета с математическими методами дискретного и дискретно-непрерывного характера, применяемых при организации и управлении современным технологическим производством. Данный курс позволяет глубже усвоить специальные и профилирующие дисциплины, такие, например, как «Теория резания», «Математическое моделирование и САПР», «Организация производства и менеджмент в машиностроении» и др.

Целью изучения дисциплины «Дискретная математика» является:

− овладение основами теоретических знаний по дискретной математике;

− ознакомление с основными прикладными задачами и методами дискретной математики;

− приобретение студентами навыков описания дискретных объектов с помощью математических моделей;

− развитие интеллектуального потенциала студентов и способностей их к логическому и алгоритмическому мышлению;

− обучение основным математическим методам научного познания.

РАЗДЕЛ 1

БУЛЕВЫ ФУНКЦИИ

В результате изучения раздела «Булевы функции» студент должен

знать:

− основные понятия: булевы функции, формулы и реализация функций формулами;

− применение булевых функций в технике;

уметь:

− для данной формулы строить таблицу истинности и определять, при каких наборах значений переменных формула принимает значения 0 или 1;

− по данной формуле строить переключательную схему и определять, при каких положениях переключателей ток в сети отсутствует.

1. КРАТКОЕ СОДЕРЖАНИЕ ТЕОРЕТИЧЕСКОГО МАТЕРИАЛА

1.1. Булевы функции. Способы задания

Определение 1. Функция $f(x_1, x_2, \ldots, x_n)$, определенная на множестве $B^n = \{0;1\}^n$ и принимающая значения из множества $B = \{0,1\}$, называется *булевой функцией* (*функцией алгебры логики, переключательной функцией*):

$$f : \underbrace{B \times B \times \ldots \times B}_{n \text{ раз}} \to B, \text{ где } B = \{0,1\}.$$

Множество булевых функций от n переменных обозначим через P_2:

$$P_2 = \left\{ f \,\middle|\, f : B^n \to B \right\}.$$

Для обозначения булевых функций и переменных используются те же обозначения, что были введены в первом разделе при изучении общей теории функций: $f, g, h, \ldots$ – обозначение функций, $x, y, z, \ldots$, x_1, x_2, $x_3, \ldots$ – обозначение переменных.

Переменные в булевых функциях могут принимать только два значения 0 и 1. Такие переменные называются *булевыми*. Переменным 0 и 1 можно придать и смысловую интерпретацию, объектами которой являются высказывания. Так, значению переменной «1» соответствует значение высказывания «*истинно*», а значению переменной «0» – «*ложно*».

Булевы функции названы в честь английского математика Джорджа Буля.

Способы задания булевых функций не отличаются от способов задания обычных функций анализа. К таковым способам задания относятся:

1) табличный, то есть с помощью так называемых *таблиц истинности*;
2) графический;
3) аналитический.

1.2. Аналитический способ задания булевых функций

Приведем обозначения и названия булевых функций от одной и двух переменных, записав в виде таблицы истинности.

Рассмотрим булевы функции от одной переменной, табл. 1.1.

Таблица 1.1

x	$f_1(x)=0$	$f_2(x)=1$	$f_3(x)=x$	$f_4(x)=\overline{x}$
0	0	1	0	1
1	0	1	1	0

В данном случае:

1) $f_1(x)=0$ – *константа* 0;
2) $f_2(x)=1$ – *константа* 1;

3) $f_3(x) = x$ – *тождественная функция*;

4) $f_4(x) = \overline{x}$ – *отрицание* x ($\overline{x}$ читается «не x», обозначение $\overline{x}$ или $\neg x$).

Две из указанных функций фактически являются константами, поскольку их значения не зависят от значений аргумента.

Рассмотрим булевы функции от двух переменных. В данном случае их общее число будет составлять 16, причем две из них являются константами, четыре – зависят от одной переменной и только десять – существенно зависят от обеих переменных. Так как из 16 булевых функций от двух переменных чаще рассматриваются только девять функций, которые имеют название и обозначение, то подробно остановимся на их описании, табл. 1.2, 1.3 и 1.4.

Таблица 1.2

x_1	x_2	$f_1(x_1, x_2) = 0$	$f_2(x_1, x_2) = 1$	$f_3(x_1, x_2) = x_1 \wedge x_2$	$f_4(x_1, x_2) = x_1 \vee x_2$
0	0	0	1	0	0
0	1	0	1	0	1
1	0	0	1	0	1
1	1	0	1	1	1

Таблица 1.3

x_1	x_2	$f_5(x_1, x_2) = x_1 \to x_2$	$f_6(x_1, x_2) = x_1 \leftrightarrow x_2$	$f_7(x_1, x_2) = x_1 \oplus x_2$
0	0	1	1	0
0	1	1	0	1
1	0	0	0	1
1	1	1	1	0

Таблица 1.4

x_1	x_2	$f_8(x_1, x_2) = x_1 \downarrow x_2$	$f_9(x_1, x_2) = x_1 \mid x_2$
0	0	1	1
0	1	0	1
1	0	0	1
1	1	0	0

В данном случае:

1) $f_1(x_1, x_2) = 0$ – *константа* 0;

2) $f_2(x_1, x_2) = 1$ – *константа* 1;

3) $f_3(x_1, x_2) = x_1 \wedge x_2$ – называется *конъюнкцией* x_1 и x_2 (читается «x_1 и x_2»). Эту функцию часто называют *логическим умножением.* Возможны следующие обозначения: $x_1 \wedge x_2$ или $x_1 \,\&\, x_2$, или $x_1 \cdot x_2$, или $\min(x_1, x_2)$;

4) $f_4(x_1, x_2) = x_1 \vee x_2$ – называют *дизъюнкцией* x_1 и x_2 (читается «x_1 или x_2»). Эту функцию часто называют *логическим сложением.* Возможны следующие обозначения: $x_1 \vee x_2$ или $\max(x_1, x_2)$;

5) $f_5(x_1, x_2) = x_1 \to x_2$ – называется *импликацией* x_1 и x_2 (читается «x_1 имплицирует x_2» или «из x_1 следует x_2»). Эту функцию часто называют *логическим следованием*. Возможны следующие обозначения: $x_1 \to x_2$ или $x_1 \supset x_2$;

6) $f_6(x_1, x_2) = x_1 \leftrightarrow x_2$ – называется *эквиваленцией* (или *эквивалентностью*) x_1 и x_2 (читается «x_1 эквивалентно x_2»). Обозначается: $x_1 \leftrightarrow x_2$;

7) $f_7(x_1, x_2) = x_1 \oplus x_2$ – называется *сложением* x_1 и x_2 *по* $\mathrm{mod}\,2$ (читается «x_1 плюс x_2»). Обозначается: $x_1 \oplus x_2$;

8) $f_8(x_1, x_2) = x_1 \downarrow x_2$ – называется *стрелкой Пирса* x_1 и x_2 (читается «ни x_1, ни x_2» или «не x_1 и не x_2»). В технической литературе данная функция обычно называется *антидизъюнкцией* или «*не – или*», а также *функцией Даггера* или *функцией Вебба*. Обозначается: $x_1 \downarrow x_2$;

9) $f_9(x_1, x_2) = x_1 \mid x_2$ – называется *штрихом Шеффера* x_1 и x_2 (читается «не x_1 или не x_2» или «x_1 и x_2 не совместны»). В технической литературе данная функция называется обычно *антиконъюнкцией* или «*не – и*». Обозначается: $x_1 \mid x_2$.

Данные функции часто употребляют в математической логике и кибернетике и играют такую же роль, как, например, x^n или $\sin x$ в математическом анализе, поэтому их можно считать «элементарными». Символы $\neg, \wedge, \vee, \to, \leftrightarrow, \oplus, \mid, \downarrow$, участвующие в обозначениях элементарных функций, называются *логическими связками* (*операциями*) или *функциональными символами*.

1.3. Формулы. Реализация функций формулами

Из курса математического анализа известно понятие сложной функции. Поэтому рассмотрим задачу построения сложной функции из булевых.

Пусть X – некоторый фиксированный *алфавит переменных*, $\sigma = \{\neg, \wedge, \vee, \to, \leftrightarrow, \oplus, \mid, \downarrow\}$ – множество *функциональных символов* (*базис*) и $\mathcal{F}$ – множество булевых функций, соответствующих функциональным символам σ, т.е. являющиеся подмножеством множества функций P_2.

Определение 2. *Формулой над* σ называется всякое, и только такое, выражение, которое имеет вид:

1) x – любая переменная из множества X;

2) $(\neg A),(A\wedge B),(A\vee B),(A|B),(A\downarrow B),(A\leftrightarrow B),(A\to B),(A\oplus B)$, где A, B – это формулы над σ.

В дальнейшем будем обозначать формулы прописными буквами латинского алфавита. В тех случаях, когда нужно обратить внимание на множество тех переменных, которые участвуют в построении формулы, записывают $A(x_1,\ldots,x_n)$. Если A – произвольная формула над σ, то формулы, которые использовались для ее построения, будем называть *подформулами* формулы A

Для формул над множеством функциональных символов (логических связок) σ принимаются некоторые соглашения:

1) внешние скобки у формул опускаются;

2) формула $(A\wedge B)$ записывается в виде $(A\cdot B)$ или (AB);

3) считается, что операция «$\neg$» сильнее любой двуместной операции из множества σ;

4) связка «$\wedge$» считается сильнее, чем любая другая двуместная связка из множества σ.

Сопоставим теперь каждой формуле G некоторую функцию f_G. Понятие булевой функции f_G, реализуемой формулой G, вводится рекурсивно по следующим правилам:

1) Формуле $G=x$, где $x\in X$, сопоставляется функция $f_G(x)=x$;

2) если G равна одной из формул $(\neg A),(A\wedge B),(A\vee B),(A|B),(A\downarrow B),(A\leftrightarrow B),(A\to B),(A\oplus B)$, где A, B – это формулы над σ, то f_G равно соответствующей элементарной булевой функции $\neg f_A$, $f_A\wedge f_B$, $f_A\vee f_B$, $f_A|f_B$, $f_A\downarrow f_B$, $f_A\leftrightarrow f_B$, $f_A\to f_B$, $f_A\oplus f_B$. Если $f_G(x_1,x_2,\ldots,x_n)$, $x_i\in X$, то значение ее на произвольном наборе $(a_1,a_2,\ldots,a_n)$, $a_i\in\{0,1\}$, совпадает со значением на этом наборе для соответствующей ей элементарной булевой функции.

Таким образом, зная таблицы истинности элементарных функций (*функций базиса*), можно вычислить и таблицу истинности функции f_G, которую реализует формула G.

1.4. Применение булевых функций в технике

Булевы функции нашли применение в теории функциональных систем, описывающие работу дискретных (прерывистых) преобразователей. Всякое дискретное устройство имеет n входов $\{x_1,x_2,\ldots,x_n\}$ и k выходов

$\{y_1, y_2, \ldots, y_k\}$. Отвлекаясь от конкретного внутреннего строения устройства, его можно считать черным ящиком, рис. 1.1.

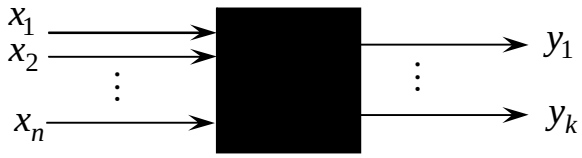


Рис. 1.1

Значения на входе и выходе устройства считаются бинарными 0 или 1, т.е. такие устройства работают в двузначном режиме (по принципу «да – нет»): «замкнуто – разомкнуто», «есть импульс – нет импульса», «есть напряжение – нет напряжения», «состояние T – состояние не-T» и т.д. Предполагается, что дискретное устройство – функциональная система, то есть каждому конкретному набору значений на своем входе устройство вырабатывает единственный набор значений на своем выходе.

Дискретные устройства могут быть реализованы из разных элементов. Простейшими дискретными устройствами можно считать устройства, собранные из *контактных схем* (*переключательных схем*).

Каждому контакту X электрической сети можно поставить в соответствие некоторую булеву переменную x, которая принимает значение 1, если контакт замкнут, и значение 0, если контакт разомкнут.

1) Дизъюнкции $p \vee q$ ставится в соответствие схема (рис. 1.2), состоящая из параллельного соединения контактов P и Q, которая замкнута тогда и только тогда, когда хотя бы один из контактов P или Q замкнут.

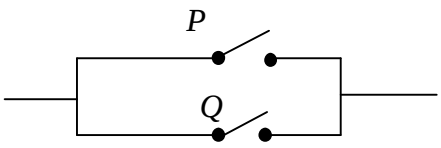


Рис. 1.2

Таблица 1.5

p	q	$p \vee q$
1	1	1
1	0	1
0	1	1
0	0	0

2) Конъюнкции $p \wedge q$ ставится в соответствие схема (рис. 1.3), состоящая из последовательного соединения контактов P и Q, которая замкнута тогда и только тогда, когда оба контакта P и Q замкнуты.

Таблица 1.6

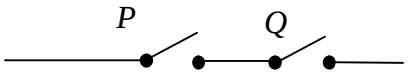


Рис. 1.3

p	q	$p \wedge q$
1	1	1
1	0	0
0	1	0
0	0	0

3) Отрицанию $\overline{p}$ может быть поставлен в соответствие контакт $\overline{P}$, который замкнут, если контакт P разомкнут, и разомкнут, когда контакт P замкнут.

Так как любая электрическая сеть состоит из замыкающих и размыкающих контактов, то ее можно выразить с помощью булевых функций, используя лишь символы конъюнкции, дизъюнкции и отрицания. Верно и обратное − каждой формуле ставится в соответствие некоторая схема, которая называется *последовательно-параллельной* или *переключательной*, или *контактной схемой*.

Первая задача, решаемая в технике с помощью булевых функций, – определение структуры дискретного устройства по заданным условиям его работы. Это задача *синтеза* структуры устройства. Вторая, обратная задача, связана с ее *анализом*, т.е. с определением условий работы устройства по уже известной ее логической структуре. Нередко разработчики и эксплуатационники сталкиваются с задачей упрощения структурной схемы устройства.

2. ПРАКТИЧЕСКИЕ ЗАДАНИЯ

Пример 1. Построить таблицу истинности для данной формулы. Определить, при каких значениях переменных формула принимает значения, равные 0:

$$F(x; y; z) = (x \downarrow \overline{y}) \vee \left(y \to \overline{(x \wedge z)} \right).$$

Решение.

Составляем комбинируемую таблицу истинности, где в каждом столбце будем находить значения булевых функций в отдельности.

x	y	z	$\overline{y}$	$x \downarrow \overline{y}$	$x \wedge z$	$\overline{(x \wedge z)}$	$y \to \overline{(x \wedge z)}$	$F(x; y; z) = (1) \vee (2)$
0	0	0	1	0	0	1	1	1
0	0	1	1	0	0	1	1	1
0	1	0	0	1	0	1	1	1
0	1	1	0	1	0	1	1	1
1	0	0	1	0	0	1	1	1
1	0	1	1	0	1	0	1	1
1	1	0	0	0	0	1	1	1
1	1	1	0	0	1	0	0	0
				(1)			(2)	

Вывод: данная формула принимает значение 0, если $x = 1,\ y = 1,\ z = 1$.

□

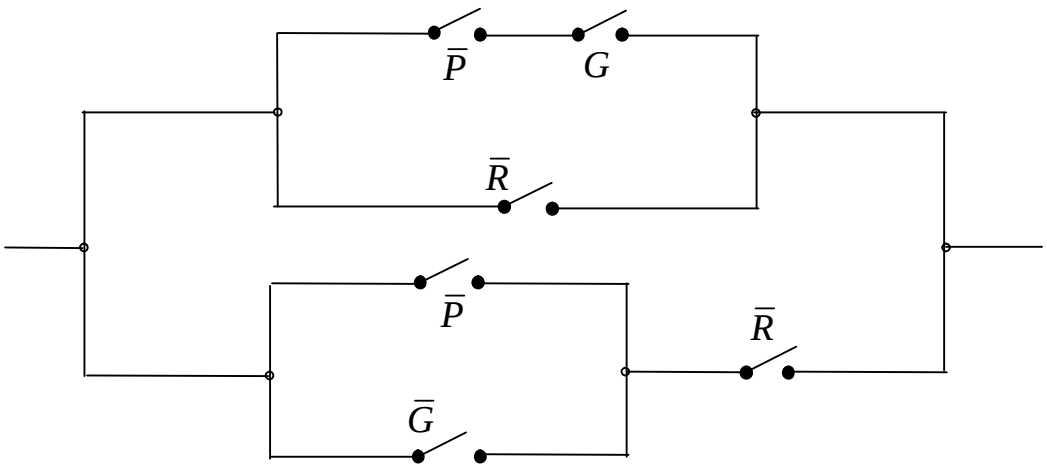
Пример 2. По данной формуле

$$f(p; g; r) = ((\overline{p} \wedge g) \vee \overline{r}) \vee ((\overline{p} \vee \overline{g}) \wedge \overline{r})$$

построить переключательную схему с помощью трёх переключателей P, G, R. Определить, при каких положениях переключателей ток в сети отсутствует.

Решение.

Для данной формулы логики высказываний соответствует следующая переключательная схема.



Определим, при каких положениях переключателей ток в сети на последнем рисунке отсутствует. В таблицу запишем все возможные наборы значений переменных p, g и r, и найдем для них соответствующие значения формулы логики высказываний.

p	g	r	$\overline{p}$	$\overline{p} \wedge g$	$\overline{r}$	$(\overline{p} \wedge g) \vee \overline{r}$	$\overline{g}$	$\overline{p} \vee \overline{g}$	$(\overline{p} \vee \overline{g}) \wedge \overline{r}$	$f(p;g;r)$
0	0	0	1	0	1	1	1	1	1	1
0	0	1	1	0	0	0	1	1	0	0
0	1	0	1	1	1	1	0	1	1	1
0	1	1	1	1	0	1	0	1	0	1
1	0	0	0	0	1	1	1	1	1	1
1	0	1	0	0	0	0	1	1	0	0
1	1	0	0	0	1	1	0	0	0	1
1	1	1	0	0	0	0	0	0	0	0

Вывод. Из последнего столбца таблицы следует, что ток в сети отсутствует в трех случаях:

1) переключатель R замкнут, а переключатели P и G разомкнуты;
2) переключатели P и R замкнуты, а переключатель G разомкнут;
3) все переключатели замкнуты;

□

РАЗДЕЛ 2

ЭЛЕМЕНТЫ ТЕОРИИ ГРАФОВ

В результате изучения раздела «Элементы теории графов» студент должен

знать:

− основные понятия и объекты теории графов;

− основы математического моделирования для описания дискретных процессов;

уметь:

− для неориентированного графа определять степень вершин; строить матрицы смежности вершин и ребер, матрицу инциденций;

− для взвешенного графа строить остовное дерево минимального веса, используя алгоритм Прима или Краскала;

− с помощью алгоритма Дейкстры в ориентированном графе находить кратчайший путь;

− с помощью алгоритма Литтла решать задачу о переналадке оборудования;

− решать задачу о формировании на сети потока максимальной мощности.

1. КРАТКОЕ СОДЕРЖАНИЕ ТЕОРЕТИЧЕСКОГО МАТЕРИАЛА

1.1. Графы и орграфы

Определение 1. *Графом* $G = G(V, E)$ называется совокупность двух множеств – непустого множества V (*множества вершин*) и множества E двухэлементных подмножеств множества V. Множество E называется *множеством ребер*.

Вершины v_i и v_j множества V называются *соединенными* ребром (v_i, v_j) или *инцидентными* к ребру (v_i, v_j), если $(v_i, v_j) \in E$. Если (v_i, v_j) – ребро, тогда вершины v_i и v_j называются *концами* ребра (v_i, v_j). Ребро (v_i, v_j) называется также *инцидентным* к вершинам v_i и v_j. Две вершины называются *смежными*, если они инцидентны к одному ребру. Два ребра называются *смежными,* если они инцидентны к общей вершине.

Число вершин графа G обозначим n, а число ребер − m:

$$|V| = n, \ |E| = m.$$

Определение 2. *Ориентированный граф* или *орграф* $G = G(V, E)$ – это такой граф, который состоит из множества V вершин и множества E упорядоченных пар элементов из V. В этом случае элемент множества E называется *дугой*. Если $(v_i, v_j) \in E$, то v_i называется *начальной вершиной* дуги (v_i, v_j), а v_j называется *конечной вершиной*.

Геометрическое представление графов следующее: вершина графа – точка в пространстве (на плоскости), ребро неориентированного графа – отрезок, дуга ориентированного графа – направленный отрезок.

Ребро, которое соединяет вершину саму с собой, называется *петлей*. Если в графе допускается наличие петель, то он называется *графом с петлями* или *псевдографом*. Если в графе допускается наличие более одного ребра между двумя вершинами, то он называется *мультиграфом*. Если каждая вершина графа и (или) ребра помечена, то такой граф называется *помеченным* (или *нагруженным*). В качестве пометок обычно используются буквы или целые числа.

Если орграф содержит более чем одну дугу из одной вершины в другую, то называется *ориентированным мультиграфом*. Если каждая дуга помечена, то будем говорить, что это *помеченный орграф*.

Определение 3. Граф $G'(V', E')$ называется *подграфом* (или *частью*) графа $G(V, E)$, если $V' \subseteq V$, $E' \subseteq E$. Если $V' = V$, то G' называется *остовным подграфом* G.

Аналогично, как и для графа, для орграфа вводится понятие ориентированный подграф.

1.2. Связность графов

Пусть $G = G(V, E)$ – граф с вершинами $v_i \in V$ $(i = \overline{1,n})$ и ребрами $e_j \in E$ $(j = \overline{1,m})$.

Определение 4. *Маршрутом* (*путем*) *длины* k из вершины v_1 в вершину v_{k+1} (или между v_1 и v_{k+1}) называется последовательность $v_1 e_1 v_2 e_2 v_3 ... v_k e_k v_{k+1}$ такая, что $e_k = (v_k, v_{k+1})$.

Таким образом, путь длины k имеет k ребер. По причине избыточности обозначений в этом определении путь для графа в общем случае будет обозначаться как $v_1 v_2 v_3 ... v_k v_{k+1}$.

Если все ребра различны, то путь называется *цепью*. Если все вершины различны (а значит, и ребра), то путь называется *простой цепью*. Замкнутая цепь называется *циклом*. Замкнутая простая цепь называется *простым циклом*. Граф без циклов называется *ациклическим*.

Аналогично, как и для графа, для орграфа вводятся понятия ориентированный путь, ориентированный цикл.

Определение 5. Граф $G = G(V, E)$ называется *связным,* если имеется цепь между любыми двумя его различными вершинами.

Для заданного ориентированного графа G можно построить неориентированный граф G^S такой, что каждая ориентированная дуга G (исключая петли) станет неориентированным ребром графа G^S. В таком случае граф $G^S(V, E^S)$ называется *соотнесенным графом* орграфа $G(V, E)$.

Определение 6. Орграф $G = G(V, E)$ называется *связным*, если его соотнесенный граф является связным. Орграф называется *сильно связным*, если для любой пары вершин v_i, $v_j \in V$ существует ориентированный путь из v_i в v_j.

1.3. Степень вершин

Определение 7. *Степенью* вершины v для неориентированного графа называется количество ребер, инцидентных этой вершине. Вершина степени 0 называется *изолированной*. Вершина степени 1 называется *висячей*. Обозначается степень вершины: $d(v)$.

Определение 8. *Полустепенью исхода* вершины v для орграфа называется количество дуг, для которых v является начальной вершиной, обозначается: $d^-(v)$. *Полустепенью захода* вершины v называется количество дуг, для которых v является конечной вершиной, обозначается: $d^+(v)$. Если $d^+(v)=0$, то вершина v называется *истоком*. Если $d^-(v)=0$, то вершина v называется *стоком*.

Теорема 1 (*Эйлера*). *Сумма степеней вершин графа равна удвоенному количеству ребер*:

$$\sum_{v\in V} d(v) = 2m .$$

Данную теорему примем без доказательства.

1.4. Представление графов в компьютере

Известны различные способы представления графов в памяти компьютера, которые различаются объемом занимаемой памяти и скорости выполнения операций над графами. Преставление выбирается исходя из потребностей конкретной задачи. В подавляющем большинстве случаев граф задается либо матрицей, либо матрицей инциденций.

Определение 9. *Матрица смежности вершин орграфа* G, содержащего n вершин, − это квадратная матрица $A_{n\times n}=\left[a_{ij}\right]_{n\times n}$ у которой строки и столбцы матрицы соответствуют вершинам орграфа. Элементы a_{ij} матрицы A равны числу дуг, направленных из i-ой вершины в j-ую. Если орграф состоит из однократных дуг, то элементы матрицы равны либо 0, либо 1.

Матрицей смежности вершин неориентированного графа G, содержащего n вершин, называют квадратную матрицу $A_{n\times n}=\left[a_{ij}\right]_{n\times n}$ у которой строки и столбцы матрицы соответствуют вершинам неориентированного графа. Элементы a_{ij} матрицы A равны числу ребер, направленных из i-ой вершины в j-ую. В случае неориентированного графа G ему вместе с ребром (v_i,v_j) принадлежит и ребро (v_j,v_i), поэтому матрица смежности вершин $A_{n\times n}=\left[a_{ij}\right]_{n\times n}$ будет симметрична относительно главной диагонали.

Определение 10. *Матрица смежности дуг орграфа* − это квадратная матрица $B_{m\times m}=\left[b_{ij}\right]_{m\times m}$ у которой строки и столбцы матрицы соответствуют дугам орграфа. Элементы b_{ij} матрицы B равны 1, если дуга e_i непосредственно предшествует дуге e_j и 0 в остальных случаях.

Матрицей смежности ребер неориентированного графа является матрица $B_{m\times m}=\left[b_{ij}\right]_{m\times m}$ у которой строки и столбцы матрицы соответствуют ребрам графа. Элементы b_{ij} матрицы B равны 1, если ребра e_i и e_j имеют общую вершину и 0 в остальных случаях.

Определение 11. *Матрицей инциденций* (*инцидентности*) *неориентированного помеченного графа* с n вершинами и m ребрами называется матрица $C_{n\times m}=\left[c_{ij}\right]_{n\times m}$, строки которой соответствуют вершинам, а столбцы – ребрам. Элементы c_{ij} матрицы инциденций неориентированного графа равны 1, если вершина v_i инцидентна ребру e_j, и 0 в противном случае.

Матрицей инциденций (инцидентности) орграфа с n вершинами и m дугами называется матрица $C_{n\times m}=\left[c_{ij}\right]_{n\times m}$, строки которой соответствуют вершинам, а столбцы – дугам орграфа:

$$C_{n\times m}=\left[c_{ij}\right]_{n\times m}=\begin{cases}1, & \text{если вершина } v_i \text{ инцидентна дуге } e_j \\ & \text{и является ее началом,}\\ 0, & \text{если вершина } v_i \text{ и дуга } e_j \text{ не инцидентны,}\\ -1, & \text{если вершина } v_i \text{ инцидентна дуге } e_j \\ & \text{и является ее концом.}\end{cases}$$

Если каждому ребру графа приписано некоторое положительное число, то такое число называется весом, а сам граф называется взвешенным графом. Простой взвешенный граф (сеть) может быть представлен также своей *матрицей весов* $\Omega=\left[\omega_{ij}\right]$, где ω_{ij} – вес ребра, соединяющего вершины v_i и v_j. Весы несуществующих ребер (дуг) полагают равными нулю или бесконечности в зависимости от приложений.

1.5. Деревья

Существует один простой и важный тип графов, которому разные авторы дали одинаковое название – *деревья*. Для них выполняются многие утверждения, которые не всегда выполняются для графов в общем случае. Деревья являются самым распространенным классом графов, применяемых в программировании, причем в самых разных ситуациях.

Определение 12. *Деревом* называется граф $T(V,E)$ без циклов. *Лес* – это граф, компоненты которого являются деревьями.

Ориентированное дерево T' представляет собой свободный от петель ориентированный граф, соотнесенный граф которого является деревом.

Если дерево имеет хотя бы одно ребро, оно имеет хотя бы две вершины со степенью 1. Вершины степени 1 называются *листьями*. Другие вершины называются *внутренними вершинами*.

Если дерево изображено таким образом, что в верхней части помещена вершина, а остальные находятся ниже его, то вершина в самой верхней части называется *корнем* дерева. Если корень дерева определен, то дерево называется *корневым деревом*.

Если корень выбран, *уровень вершины* v определяется длиной единственной цепи из корня в вершину v.

Определение 13. *Высотой* дерева называется длина самой длинной цепи от корня до листа, обозначается: $h(T)$.

1.6. Остовные деревья

Определение 14. Дерево T называется *остовным деревом* графа G, если T – подграф графа G и каждая вершина графа G является вершиной в T.

Теорема 2. *У каждого связного графа существует подграф, который является остовным деревом.*

Данную теорему примем без доказательства.

Определение 15. *Вес* остовного дерева взвешенного графа G равен сумме весов, приписанных ребрам остовного дерева. Обозначение: $\omega(T)$.

Минимальным остовным деревом называется такое остовное дерево графа G, что вес дерева T меньше или равен весу любого другого остовного дерева графа G. Вес минимального остовного дерева будем обозначать $\omega_{\min}(T)$.

Построение остова графа G, имеющего наименьший вес, имеет широкое применение при решении некоторого класса задач прикладного характера.

Рассмотрим два способа построения минимального остовного дерева взвешенного графа: алгоритм Краскала и алгоритм Прима.

Идея метода Краскала состоит в том, чтобы формировать дерево $T(V, E)$, выбирая ребра с наименьшим весом так, чтобы не возникал цикл.

Алгоритм Краскала

1. Выбрать в графе G ребро m минимального веса, не принадлежащее множеству E и такое, чтобы его добавление в множество E не создавало в дереве T цикл.

2. Добавить это ребро в множество ребер E.

3. Продолжать действия первого и второго шагов до тех пор, пока имеются ребра, обладающие указанными свойствами.

Принципиальное отличие алгоритма Прима состоит в том, что всегда имеется дерево, к которому ребра добавляют до тех пор, пока не получится остовное дерево.

Алгоритм Прима

1. Выбрать вершину v_1 графа G и ребро с наименьшим весом e_1, для которого v_1 – одна из вершин, и сформировать дерево T_1.

2. Для заданного дерева T_k с ребрами e_1, e_2, e_3,..., e_k, при наличии вершины, не принадлежащей T_k, следует выбрать ребро с наименьшим весом, смежное с ребром дерева T_k и имеющее вершину вне дерева T_k. Добавить новую вершину в дерево T_k, формируя дерево T_{k+1}.

3. Продолжать, пока имеются вершины графа G, не принадлежащие дереву.

Построение минимального остовного дерева можно проводить двумя способами: 1) остов графа строится непосредственно на самом графе, выделяя ребра утолщенной линией, которые входят в остовное дерево; 2) отдельно строится корневое дерево, которое будет минимальным остовным деревом. Второй случай используется, если требуется определить высоту построенного дерева.

1.7. Нахождение кратчайших путей. Алгоритм Дейкстры

Остановимся на вопросе перемещения из одной вершины в другую с позиции «наилучшего способа» достижения оптимального значения определенного критерия. Детализируем это требование: это может быть как самый дешевый путь по стоимости, так и самый кратчайший путь по протяженности, как самый безопасный путь, так и наименее энергопотребляющий, и т.д.

Для решения такого класса задач рассматриваются взвешенные графы (орграфы), ребрам (дугам) которых приписан некоторый вес. Общий подход к решению задачи о кратчайшем пути был развит американским математиком Ричардом Беллманом (1920 – 1984), который предложил для этого вида задач название *динамическое программирование*. Задача о кратчайшем пути – это частный случай задачи, которую можно сформулировать следующим образом: *найти в заданном графе пути, соединяющие две заданные вершины и доставляющие минимум или максимум некоторой аддитивной функции, оп-*

ределенной как критерий эффективности множества путей. Чаще всего эта функция трактуется как длина пути, и задача называется *задачей о кратчайшем пути*.

Алгоритм Дейкстры (Едсгер Дейкстра, нидерландский математик, 1930 – 2002) является одной из реализаций этой задачи. Данный алгоритм часто еще называют алгоритмом расстановки меток.

Пусть $G = G(V, E, \Omega)$ – ориентированный граф с взвешенными дугами. Обозначим через s (s-вершина) – вершину начала пути и через t (t-вершину) – вершину конца пути. В процессе работы алгоритма Дейкстры вершинам орграфа $x_i \in V$ приписываются числа (метки) $d(x_i)$, которые служат оценкой длины (веса) кратчайшего пути от s-вершины к вершине x_i. Если вершина x_i получила на некотором шаге метку $d(x_i)$, это означает, что в графе G существует путь из s-вершины в x_i, имеющий вес $d(x_i)$. Метки могут находиться в двух состояниях – быть временными или постоянными. Превращение временной метки в постоянную означает, что кратчайшее расстояние от вершины s до соответствующей вершины найдено. Алгоритм Дейкстры содержит одно ограничение – веса дуг должны быть положительными. Сам алгоритм состоит из двух этапов. На первом этапе находится длина кратчайшего пути, на втором − строится сам путь от вершины s к вершине t.

Алгоритм Дейкстры

Этап 1. Нахождение длины кратчайшего пути.

Шаг 1. *Присвоение вершинам начальных меток.*

Полагаем $d(s) = 0^*$, и считаем эту метку постоянной (постоянные метки помечаем сверху звездочкой). Для остальных вершин $x_i \in V$, $x_i \neq s$ полагаем $d(x_i) = \infty$ и считаем эти метки временными. Обозначим через $\tilde{x}$ текущую вершину и примем первоначально $\tilde{x} = s$.

Шаг 2. *Изменение меток.*

Для каждой вершины x_i с временной меткой, непосредственно следующей за вершиной $\tilde{x}$, меняем ее метку в соответствии со следующим правилом:

$$d_{нов}(x_i) = \min\left\{d_{стар}(x_i); d(\tilde{x}) + \omega(\tilde{x}; x_i)\right\}.$$

Шаг 3. *Превращение временной метки в постоянную.*

Из всех вершин с временными метками выбираем вершину x_j^* с наименьшим значением метки

$$d(x_j^*) = \min\left\{d(x_j)\middle| x_j \in V\right\},$$ где $d(x_j)$ − временная метка.

Превращаем эту метку в постоянную и полагаем $\tilde{x} = x_j^*$.

Шаг 4. *Проверка на завершение первого этапа.*

Если $\tilde{x} = t$, то $d(\tilde{x})$ – длина кратчайшего пути от вершины s до t. В противном случае происходит возвращение ко второму шагу.

Этап 2. Построение кратчайшего пути.

Шаг 5. *Последовательный поиск дуг кратчайшего пути.*

Среди вершин, непосредственно предшествующих вершине $\tilde{x}$ с постоянными метками, находим вершину x_i, удовлетворяющую соотношению

$$d(\tilde{x}) = d(x_i) + \omega(x_i; \tilde{x}).$$

Включаем дугу $(x_i; \tilde{x})$ в искомый путь и полагаем $\tilde{x} = x_i$.

Шаг 6. *Проверка на завершение второго этапа.*

Если $\tilde{x} = s$, то кратчайший путь найден – его образует последовательность дуг, полученных на пятом шаге и выстроенных в обратном порядке. В противном случае возвращаемся к пятому шагу.

1.8. Пути и циклы Гамильтона. Алгоритм Литтла

Пусть $G(V, E)$ – граф.

Определение 16. *Гамильтонов путь* – это простой путь, который проходит через каждую вершину графа G. *Гамильтонов цикл* – это простой цикл, который проходит через каждую вершину графа G.

Рассмотрим одну из задач, в которой требуется отыскать во взвешенном графе гамильтонов цикл минимальной длины. Приведем формулировку задачи и алгоритм ее решения.

Задача коммивояжера (или бродячего торговца)

Постановка задачи: коммивояжер должен так составить свой маршрут движения, чтобы посетить один и только один раз каждый из n городов и вернуться в исходный пункт. Его маршрут должен минимизировать суммарную длину пройденного пути.

После текстовой постановки задачи с помощью математического языка (символов, функций, уравнений или неравенств и т.д.) построим *математическую модель задачи*.

Итак, пусть $C = [c_{ij}]$ – матрица расстояний между городами. Для составления математической модели задачи обозначим через переменные x_{ij} факт переезда коммивояжером из города i в город j. Поскольку переезд из

одного города в другой может осуществляться только один раз, то переменные x_{ij} должны принимать только два значения: 1 или 0, т.е. булевы значения. Таким образом:

$$x_{ij} = \begin{cases} 1, & \text{если коммивояжер из города } i \\ & \text{переезжает непосредственно в город } j; \\ 0, & \text{в противном случае.} \end{cases}$$

Математическая модель задачи имеет следующий вид:

$$Z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \to \min \tag{1}$$

$$\sum_{i=1}^{n} x_{ij} = 1, \quad (j = \overline{1, n}), \tag{2}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad (i = \overline{1, n}), \tag{3}$$

$$x_{ij} \in \{0;1\}.$$

Система ограничений (2) обеспечивает построение маршрута, при котором коммивояжер въезжает в каждый город только один раз, а система (3) – маршрута, при котором он выезжает из каждого города только один раз. Устранение подциклов и получение маршрута, образующего полный цикл, включающий все города, достигается при добавлении системы ограничений:

$$u_i - u_j + n x_{ij} \le n - 1 \quad (i = \overline{1, n}, \ j = \overline{1, n}, \ i \ne j), \tag{4}$$

где все переменные u_i и u_j могут принимать произвольные действительные значения.

Решение задачи коммивояжера методом ветвей и границ (алгоритм Литтла)

Если считать города вершинами графа, а коммуникации $(i; j)$ – его дугами, то нахождение минимального пути, проходящего один и только один раз через каждый город с возвращением в исходную точку можно рассматривать как нахождение на графе гамильтонова цикла минимальной длины.

Рассмотрим алгоритм поиска гамильтонова цикла минимальной длины на графе с n вершинами (алгоритм Литтла). Если между вершинами i и j нет дуги, то ставится символ ∞. Этот же символ ставится на главной диагонали, что означает запрет на возвращение в вершину, через которую уже проходил цикл. Основная идея метода состоит в том, что сначала строят нижнюю границу длин $\varphi_0(\Omega^0)$ множества гамильтоновых циклов Ω^0. Затем множество циклов Ω^0 разбивается на два подмножества таким образом, что

первое подмножество Ω_{ij}^1 состояло из гамильтоновых циклов, содержащих некоторую дугу $(i;j)$, а другое подмножество $\tilde{\Omega}_{ij}^1$ не содержало эту дугу. Для каждого из подмножеств определяются нижние границы по тому же правилу, что и для первоначального множества гамильтоновых циклов. Полученные нижние границы подмножеств Ω_{ij}^1 и $\tilde{\Omega}_{ij}^1$ оказываются не меньше нижней границы для всего множества гамильтоновых циклов, то есть

$$\varphi_0(\Omega^0) \le \varphi(\Omega_{ij}^1) \equiv \varphi_{ij}^1, \quad \varphi_0(\Omega^0) \le \varphi(\tilde{\Omega}_{ij}^1) \equiv \tilde{\varphi}_{ij}^1 .$$

Сравнивая нижние границы φ_{ij}^1 и $\tilde{\varphi}_{ij}^1$ подмножеств Ω_{ij}^1 и $\tilde{\Omega}_{ij}^1$, можно выделить среди них то, которое с большей вероятностью содержит гамильтонов цикл минимальной длины. Затем одно из подмножеств Ω_{ij}^1 или $\tilde{\Omega}_{ij}^1$ по аналогичному правилу разбивается на два новых Ω_{ij}^2 и $\tilde{\Omega}_{ij}^2$. Для них снова определяются нижние границы φ_{ij}^2 и $\tilde{\varphi}_{ij}^2$ и т.д., рис. 1.

Процесс ветвления продолжается до тех пор, пока не отыщется единственный гамильтонов цикл. Его называют *первым рекордом*. Затем просматривают оборванные ветви. Если их нижние границы больше длины первого рекорда, то задача решена. Если же найдутся такие, для которых нижние границы меньше, чем длина первого рекорда, то подмножество с наименьшей нижней границей подвергают дальнейшему ветвлению, пока не убеждаются, что оно не содержит лучшего гамильтонова цикла. Если же такой найдется, то анализ оборванных ветвей продолжается относительно нового значения длины цикла. Его называют *вторым рекордом*. Процесс решения заканчивается тогда, когда будут проанализированы все подмножества.

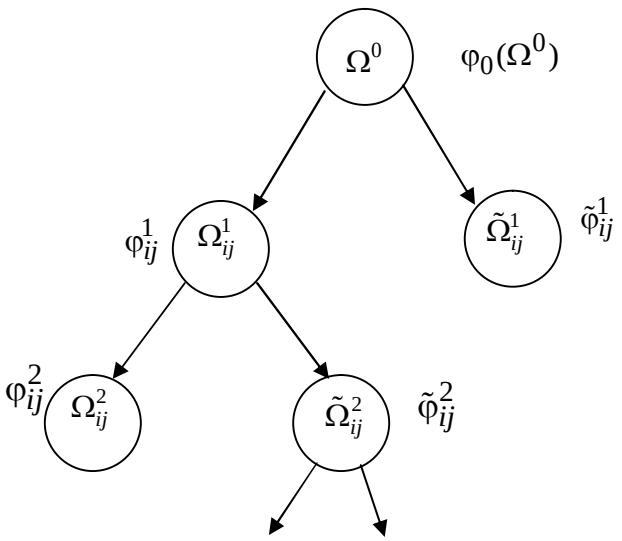


Рис. 1

Алгоритм Литтла

Пусть известна матрица весов некоторого орграфа, вершины которого занумеруем числами от 1 до n:

$$\Omega = \left(\begin{array}{c|cccc} & 1 & 2 & \dots & n \\ \hline 1 & \infty & \omega_{12} & \dots & \omega_{1n} \\ 2 & \omega_{21} & \infty & \dots & \omega_{2n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ n & \omega_{n1} & \omega_{n2} & \dots & \infty \end{array}\right),$$

где ω_{ij} – вес дуги, соединяющей вершины i и j, $i = \overline{1, n}$, $j = \overline{1, n}$.

Символ ∞ ставится для блокировки дуг графа, которые явно не включаются в гамильтонов цикл, т.е. в расчет не берутся дуги l_{ii} и $\omega_{ij} = \infty$. Можно показать, что оптимальность решения не изменяется при прибавления некоторого числа к строке или столбцу матрицы Ω.

Алгоритм Литтла разбивается на несколько шагов.

Шаг 1. *Приведение исходной матрицы.*

В каждом столбце и в каждой строке матрицы Ω нужно получить хотя бы один нуль. Для этого, например, в первом столбце выберем минимальный элемент и вычтем его из всех элементов первого столбца. Аналогично поступим с остальными столбцами. Если при этом в некоторых строках не появляются нули, то для них осуществим ту же процедуру.

После этого вычислим константу приведения φ_0 – сумму минимальных элементов столбцов и строк, использованных для преобразования матрицы Ω. Получим приведенную матрицу Ω_0 с константой приведения φ_0.

Шаг 2. *Определение степеней нулей.*

Найдем степени каждого нуля – сумму минимальных элементов строки i_k и столбца j_m, $k = \overline{1, n}$, $m = \overline{1, n}$, в которых стоит этот нуль, без учета самого нуля. Нуль с максимальной степенью определяет дугу $l(i_1; j_1)$, которая вероятнее всего войдет в гамильтонов цикл. Например, если нуль с максимальной степенью находится на пересечении второй строки и третьего столбца, то дуга $l(2;3)$, вероятнее всего, войдет в гамильтонов цикл.

Шаг 3. *Ветвление.*

На самом деле, дуга, соответствующая максимальной степени нуля, может, как входить в гамильтонов цикл, так и не входить в него. Поэтому следует рассмотреть сразу два случая.

Первый случай. Возможно, дуга $l(i_1; j_1)$ вошла в гамильтонов цикл. Блокируем ее, полагая $\omega_{ji} = \infty$. Строку i_1 и столбец j_1 вычеркиваем. Если требуется, приводим полученную матрицу меньшего порядка $\Omega_1(i_1; j_1)$ с константой приведения $\varphi_1 = \varphi_0 + \Delta_1$.

Второй случай. Дуга $l(i_1; j_1)$ не вошла в гамильтонов цикл. Полагаем $\omega_{ij} = \infty$. Если требуется, приводим полученную матрицу $\tilde{\Omega}_1(i_1; j_1)$ с константой приведения $\tilde{\varphi}_1 = \varphi_0 + \tilde{\Delta}_1$.

Если $\varphi_1 < \tilde{\varphi}_1$, то шаги 2, 3 повторяем с матрицей $\Omega_1(i_1; j_1)$.

Если $\varphi_1 > \tilde{\varphi}_1$, то шаги 2, 3 повторяем с матрицей $\tilde{\Omega}_1(i_1; j_1)$. И так до тех пор, пока не дойдем до матрицы второго порядка, содержащей два нуля:

$$\Omega_{n-2}(i_{n-2}; j_{n-2}) = \left(\begin{array}{c|cc} & j_{n-1} & j_n \\ \hline i_{n-1} & 0 & A \\ i_n & B & 0 \end{array}\right),$$

где A и B – некоторые числа или ∞. Нули соответствуют двум последним дугам гамильтонова цикла: $l(i_{n-1}; j_{n-1})$, $l(i_n; j_n)$, при этом $\varphi = \varphi_{n-2}$.

Если $\varphi_{n-2} < \tilde{\varphi}_k$, где $k = 1, 2, \ldots, n-2$, то задача решена. Если же для некоторого k_0 получается $\varphi_{n-2} > \tilde{\varphi}_k$, то всю процедуру следует провести с матрицей $\tilde{\Omega}_k(i_k; j_k)$. На последнем этапе получим новое значение функции φ: φ'_{n-2}. Это значение сравниваем с φ_{n-2}, то есть процесс продолжается до тех пор, пока новые значения φ'_k не станут меньше φ_{n-2}.

Классическая задача коммивояжера, только в несколько иной формулировке, нашла применение и при организации производства в машиностроении – задача о переналадке оборудования при переходе линии на обработку деталей, схожих по конструкции и габаритам.

1.9. Сети. Потоки в сетях

Функциональное назначение большинства физически реализованных сетей состоит в том, что они служат носителями систем потоков, т.е. систем, в которых некоторые объекты текут, движутся или транспонируются по системе каналов (дуг сети) ограниченной пропускной способности.

Изучение данных практических задач и многочисленных им подобных приводит к теории потоков в сетях. Данная теория разрабатывает решения общей задачи, которая называется *задача об оптимальном потоке*. Рассмотрим частный случай этой задачи, а именно, *задачу определения максимальной величины потока*.

Определение 17. *Сетью* называется связный ориентированный граф $G(V,E)$ без петель с выделенными вершинами I – *истоком* и S – *стоком*, причем каждой дуге поставлено в соответствие некоторое натуральное число $c(v_i,v_j)$ – *пропускная способность дуги*.

Напомним, что истоком называется вершина, у которой $d^+(v)=0$, т.е. из которой дуги только выходят; а стоком – вершина, у которой $d^-(v)=0$, т.е. вершина, в которую дуги только входят.

Пропускная способность дуги характеризует максимальное количество вещества, которое может пропустить за единицу времени дуга (v_i,v_j). На сети пропускную способность дуги будем записывать в круглых скобках.

Поток в сети определяет способ пересылки некоторых объектов из одной вершины графа в другую по направлению дуги. Число объектов (количество вещества) $f(v_i,v_j)$, пересылаемых вдоль дуги (v_i,v_j), не может превышать пропускной способности $c(v_i,v_j)$ этой дуги:

$$0 \le f(v_i,v_j) \le c(v_i,v_j).$$

Будем считать, что если существует дуга из вершины v_i в вершину v_j, то нет дуги из вершины v_j в вершину v_i. Таким образом, рассматривается поток вещества только в одну сторону.

Постановка задачи о максимальном потоке

На сети с заданными пропускными способностями дуг требуется сформировать максимальный по величине поток $F_{\max}$ между ее истоком и стоком. Этот поток обеспечивается назначением в каждой дуге (v_i,v_j) величины $f(v_i,v_j)$ передаваемого ею потока.

Задача о максимальном потоке в сети должна удовлетворять следующим условиям:

– сумма потоков дуг, выходящих из истока сети, должна быть равна сумме потоков дуг, входящих в сток сети:

$$\sum_{(I,v_i)\in E} f(I,v_i) = \sum_{(v_j,S)\in E} f(v_j,S);$$

– для вершины v, не являющейся стоком или истоком, т.е. $v \ne I$, $v \ne S$, количество единиц потока, входящего в вершину, должно быть равно количеству единиц потока, выходящего из нее (т.е. требуется сохранение потока):

$$\sum_{(v_i,v)\in E} f(v_i,v) = \sum_{(v,v_j)\in E} f(v,v_j);$$

– максимальный поток на пути от истока I к стоку S определяется той дугой (v_i, v_j), которая имеет минимальную пропускную способность из всех дуг, принадлежащих этому пути.

Если пропускная способность $c(v_i, v_j)$ дуги (v_i, v_j) равна идущему через нее потоку $f(v_i, v_j)$ (на сети значение $f(v_i, v_j)$ обозначается числом без скобок), то такая дуга называется *насыщенной*, а любой путь, в который она включена, называется *насыщенным путем*. Поток называется *насыщенным*, если любой путь из истока I в сток S содержит дугу (v_i, v_j), для которой $f(v_i, v_j) = c(v_i, v_j)$. Первая часть решения задачи о максимальном потоке как раз и состоит в нахождении насыщенного потока. Но насыщенный поток не всегда является максимальным.

По определению, поток в сети будет максимальным, если величина этого потока $F_{\max}$ будет больше величины любого другого потока в этой сети.

Разрез на сети

Максимальный поток определяется с помощью одного из основных понятий теории сетей – разреза.

Разрез может быть представлен как множество дуг, исключение которых из сети сделало бы орграф несвязным.

Предположим, что дана некоторая сеть. Разобьем множество вершин V этой сети на два непересекающихся подмножества A и B $(A \cup B = V,\ A \cap B = \varnothing)$ так, чтобы исток I попал в подмножество A, а сток S – в подмножество B, т.е. $I \in A, S \in B$. В этом случае говорят, что *на сети произведен разрез*, отделяющий исток I от стока S.

Пусть $R(A/B)$ − разрез на сети, представляющий совокупность дуг, которые связывают подмножества вершин A и B. В разрез входят дуги, начальные вершины которых принадлежат подмножеству A, а конечные – подмножеству B, обозначим их через R^+, т.е. $R^+ = \left\{(v_i, v_j) \middle| v_i \in A, v_j \in B\right\}$. Также в разрез входят дуги, начальные вершины которых принадлежат подмножеству B, а конечные – подмножеству A, обозначим их через R^-, т.е. $R^- = \left\{(v_i, v_j) \middle| v_i \in B, v_j \in A\right\}$.

Пропускной способностью или *величиной разреза* $R(A/B)$ называется величина $C(A/B)$, которая определяется следующей формулой:

$$C(A/B) = C(R^+) - C(R^-) = \sum_{v_i \in A,\ v_j \in B} c(v_i, v_j) - \sum_{v_i \in B,\ v_j \in A} c(v_i, v_j).$$

Потоком через разрез $R(A/B)$ называется величина $F(A/B)$, которая определяется следующей формулой:

$$F(A/B) = F(R^+) - F(R^-) = \sum_{v_i \in A,\ v_j \in B} f(v_i, v_j) - \sum_{v_i \in B,\ v_j \in A} f(v_i, v_j). \quad (6.2)$$

Алгоритм нахождения максимального потока на сети

Теорема 3 (Форда-Фалкерсона).

Максимальный поток в сети равен минимальной пропускной способности разреза:

$$F_{\max} = C_{\min}(A/B).$$

Доказательство теоремы представляет алгоритм определения максимального потока по сети.

Алгоритм нахождения максимального потока на сети:

Этап 1. *Насыщение потока.*

Шаг 1. Сформировать произвольный начальный поток.

Шаг 2. Найти оставшиеся возможные пути из истока I в сток S, имеющие только ненасыщенные дуги. Если такой путь найден, то переходим к шагу 3. Если путь не найден, то переходим к шагу 4.

Шаг 3. Увеличить поток по найденному пути таким образом, чтобы, по крайней мере, одна из дуг стала насыщенной.

Шаг 4. Получившийся поток насыщен.

Этап 2. *Пометка вершин сети* (*перераспределение потока*).

Шаг 5. Вершину I пометить $-I$.

Шаг 6. Пусть m – любая из уже помеченных вершин, n – произвольная непомеченная вершина, смежная с вершиной m. Вершину n помечаем $+m$, если данные вершины соединены *ненасыщенной* дугой $m \to n(+m)$, и помечаем $-m$, если соединены *непустой* дугой $m \leftarrow n(-m)$.

После пометки вершин возможны два случая: вершина S оказалась либо помеченной, либо непомеченной.

Шаг 7. Вершина S оказалась помеченной. Значит, существует последовательность помеченных вершин от истока I к стоку S. В этой последовательности каждая последующая вершина помечена буквой предыдущей вершины. На дугах последовательности определяем новый поток. Увеличиваем на δ единиц поток на дугах, имеющих направление от I к S и уменьшаем на

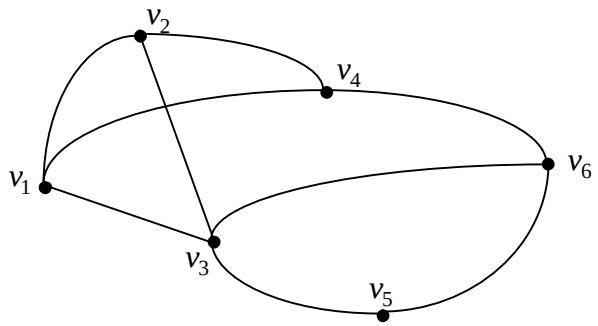
δ единиц поток на дугах, имеющих обратное направление. Число δ равно наименьшей разнице между пропускной способностью и потоком дуг, входящих в последовательность. Заметим, что поток можно увеличивать (уменьшать) на прямых (обратных) дугах до тех пор, пока одна из дуг не станет насыщенной (пустой). Далее вновь переходим к пометке вершин (шаг 5). Перераспределение потока сохраняет все его свойства и увеличивает поток на δ единиц в вершину S.

Этап 3. *Определение максимального потока.*

Шаг 8. Вершина S осталась непомеченной. Пусть A – множество всех помеченных вершин, B – множество всех непомеченных вершин. Тогда дуги, связывающие два подмножества вершин A и B, определяют разрез $R(A/B)$. Таким образом, найден поток F и разрез $R(A/B)$, для которого выполняется условие $F_{\max} = C_{\min}(A/B)$.

2. ПРАКТИЧЕСКИЕ ЗАДАНИЯ

Пример 1. Для данного неориентированного графа:



1) определить степень каждой вершины;

2) построить матрицу смежности вершин, матрицу смежности ребер, матрицу инциденций.

Решение.

1) Используя определение степени вершины графа, находим:

$$d(v_1)=3,\ d(v_2)=3,\ d(v_3)=4,\ \ d(v_4)=3,\ \ d(v_5)=2,\ \ d(v_6)=3.$$

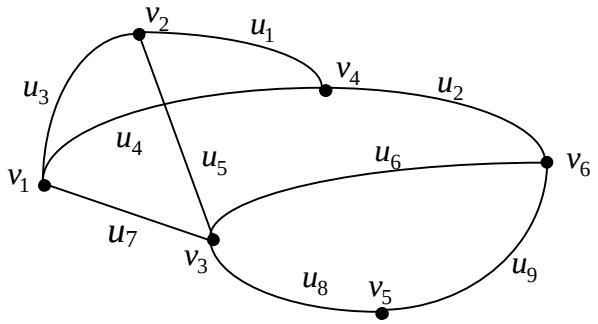
2) Построим матрицу смежности вершин. Данный граф имеет шесть вершин, поэтому матрица смежности вершин A является квадратной матрицей шестого порядка. Для заполнения первой строки матрицы A найдем число ребер, соединяющих первую вершину с каждой из остальных вершин.

Поскольку количество ребер, соединяющих первую вершину с первой равно нулю (в графе нет петель, соответствующих первой вершине), количество ребер соединяющих первую со второй равно одному, первую с третьей − одному, первую с четвертой − одному, первую с пятой − нулю, первую с

шестой − нулю, то первая строка матрицы A имеет вид $(0\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0)$. Заполняя аналогично остальные строки, получим матрицу A вида:

$$A=\left(\begin{array}{c|cccccc} & v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \\ \hline v_1 & 0 & 1 & 1 & 1 & 0 & 0 \\ v_2 & 1 & 0 & 1 & 1 & 0 & 0 \\ v_3 & 1 & 1 & 0 & 0 & 1 & 1 \\ v_4 & 1 & 1 & 0 & 0 & 0 & 1 \\ v_5 & 0 & 0 & 1 & 0 & 0 & 1 \\ v_6 & 0 & 0 & 1 & 1 & 1 & 0 \end{array}\right).$$

Для построения матрицы смежности ребер и матрицы инциденций рассмотрим не взвешенный граф, предварительно занумеровав ребра графа произвольным образом.



Так как граф имеет девять ребер, то матрица смежности ребер B является квадратной матрицей девятого порядка. Ребро u_1 имеет общую вершину с ребрами u_3 и u_5, а также общую вершину с ребрами u_2 и u_4. Поэтому первая строка матрицы B имеет вид $(0\ \ 1\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0)$. Заполняя аналогично остальные строки, получим матрицу B вида:

$$B=\left(\begin{array}{c|ccccccccc} & u_1 & u_2 & u_3 & u_4 & u_5 & u_6 & u_7 & u_8 & u_9 \\ \hline u_1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ u_2 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ u_3 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ u_4 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ u_5 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ u_6 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ u_7 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 \\ u_8 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ u_9 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{array}\right).$$

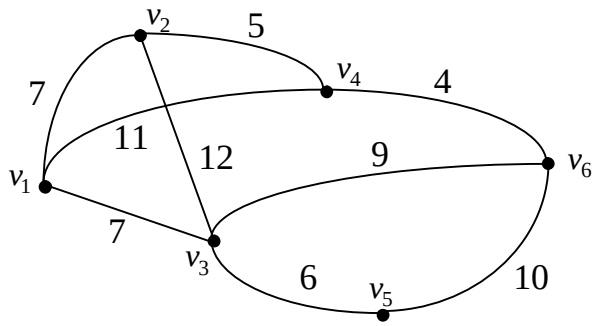
Граф имеет шесть вершин и девять ребер, поэтому матрица инциденций C имеет размерность 6×9. Поскольку вершина v_1 инцидентна ребрам u_3, u_4 и u_7, и не инцидентна остальным ребрам, то первая строка матрицы C имеет вид $(0\ \ 0\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0)$. Остальные строки матрицы заполняются аналогично. Следовательно, матрица C будет иметь вид:

$$C=\left(\begin{array}{c|ccccccccc} & u_1 & u_2 & u_3 & u_4 & u_5 & u_6 & u_7 & u_8 & u_9 \\ \hline v_1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ v_2 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ v_3 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ v_4 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ v_5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ v_6 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{array}\right).$$

Ответ: $d(v_1)=3,\ d(v_2)=3,\ d(v_3)=4,\ d(v_4)=3,\ d(v_5)=2,\ d(v_6)=3.$

□

Пример 1. Для данного взвешенного графа найти минимальное корневое остовное дерево, используя алгоритм Краскала или алгоритм Прима.



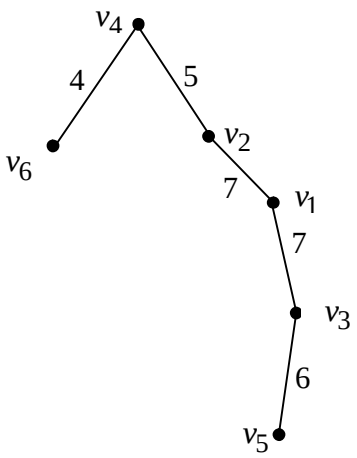
Решение.

I способ.

Используем для построения остовного дерева алгоритм Краскала.

В графе выберем ребро с минимальным весом. В нашем случае это ребро, соединяющее вершины v_4 и v_6 с весом, равным 4. Пусть, например, вершина v_4 будет корнем дерева. Далее выбираем ребра, инцидентные вершинам v_4, v_6 и имеющие минимальный вес. Таким будет ребро с весом 5, соединяющее вершины v_4 и v_2. Включим его в строящееся дерево. Затем к вершине v_2 присоединим ребро с весом 7, соединяющее вершины v_2 и v_1. К вершине v_1 присоединим ребро с весом 7, соединяющее вершины v_1 и v_3. И, в заключение, к вершине v_3 присоединим ребро с весом 6, соединяющее

вершины v_3 и v_5. Таким образом, получаем минимальное остовное дерево. Минимальный вес построенного дерева равен: $\omega_{\min}(T) = 4 + 5 + 7 + 7 + 6 = 29$.



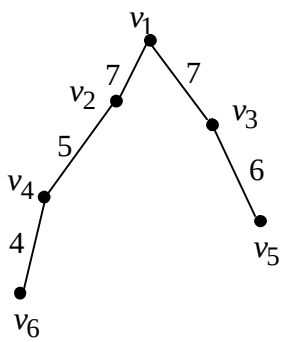
Так как мы в качестве корня дерева выбрали вершину v_4, то высота дерева будет равна $h(T) = 4$.

Заметим, что если в качестве корня выбрать вершину v_6, то высота дерева будет равна $h(T) = 5$.

II способ.

Рассмотрим построение минимального корневого остовного дерева данного взвешенного графа с помощью алгоритма Прима.

Выберем вершину v_1, которая будет корнем дерева. Из трех ребер, которые инцидентны вершине v_1, выберем те, что имеют наименьший вес. Два ребра с весом, равным 7, инцидентны вершине v_1. Присоединим эти ребра к выбранной вершине. К вершине v_1 присоединим ребро с весом 5, соединяющее вершины v_2 и v_4. К вершине v_4 присоединим ребро с весом 4, соединяющее вершины v_4 и v_6. К вершине v_3 присоединим ребро с весом 6, соединяющее вершины v_3 и v_5. Таким образом, получаем следующее минимальное корневое дерево с весом, равным $\omega_{\min}(T) = 7 + 7 + 6 + 5 + 4 = 29$. Заметим, что высота построенного дерева равна $h(T) = 3$.



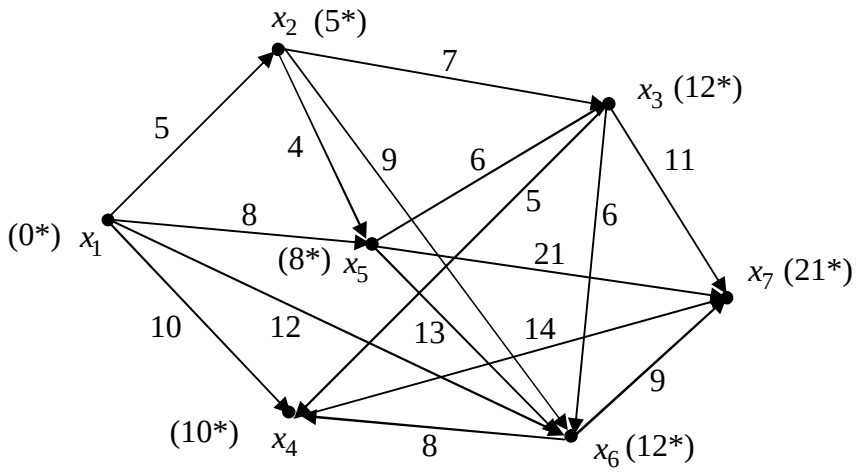
Ответ: минимальный вес дерева равен $\omega_{\min}(T) = 29$. □

Пример 3. По заданной матрице весов найти величину минимального пути и сам путь от вершины $s = x_1$ до вершины $t = x_7$ с помощью алгоритма Дейкстры.

$$\Omega = \left[\omega_{ij} \right] = \left(\begin{array}{c|ccccccc} & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 \\ \hline x_1 & - & 5 & \infty & 10 & 8 & 12 & \infty \\ x_2 & \infty & - & 7 & \infty & 4 & 9 & \infty \\ x_3 & \infty & \infty & - & 5 & \infty & 6 & 11 \\ x_4 & \infty & \infty & \infty & - & \infty & \infty & 14 \\ x_5 & \infty & \infty & 6 & \infty & - & 13 & 21 \\ x_6 & \infty & \infty & \infty & 8 & \infty & - & 9 \\ x_7 & \infty & \infty & \infty & \infty & \infty & \infty & - \end{array} \right)$$

Решение.

Сначала построим орграф, у которого к каждой дуге припишем ее вес.



Этап 1.

Шаг 1. Полагаем $d(x_1) = 0^*$,

$d(x_2) = d(x_3) = d(x_4) = d(x_5) = d(x_6) = d(x_7) = \infty$.

Шаг 2. За вершиной $\tilde{x} = x_1$ следуют вершины, которые образуют множество $\tilde{s} = \{x_2, x_4, x_5, x_6\}$.

Пересчитаем временные метки:

$d(x_2) = \min\{\infty; 0^* + 5\} = 5$, $\quad d(x_4) = \min\{\infty; 0^* + 10\} = 10$,

$d(x_5) = \min\{\infty; 0^* + 8\} = 8$, $\quad d(x_6) = \min\{\infty; 0^* + 12\} = 12$.

Получаем $d(x_2) < d(x_5) < d(x_4) < d(x_6)$. Следовательно, вершине x_2 присваивается постоянная метка: $d(x_2) = 5^*$, $\tilde{x} = x_2$.

Шаг 3. $\tilde{s} = \{ x_4,\ x_5,\ x_6,\ x_3 \}$. Пересчитаем временные метки:

$d(x_4) = \min\{\infty; 0^* + 10\} = 10$, $d(x_5) = \min\{\infty; \underline{0^* + 8}; 5^* + 4\} = 8$,

$d(x_6) = \min\{\infty; \underline{0^* + 12}, 5^* + 9\} = 12$, $d(x_3) = \min\{\infty; 5^* + 7\} = 12$.

Получаем $d(x_5) < d(x_4) < d(x_3) = d(x_6)$. Следовательно, вершине x_5 присваивается постоянная метка: $d(x_5) = 8^*$, $\tilde{x} = x_5$.

Шаг 4. $\tilde{s} = \{ x_4,\ x_3,\ x_6,\ x_7 \}$. Пересчитаем временные метки:

$d(x_4) = \min\{\infty; 0^* + 10\} = 10$, $d(x_3) = \min\{\infty; \underline{5^* + 7}, 8^* + 6\} = 12$,

$d(x_6) = \min\{\infty; \underline{0^* + 12}; 5^* + 9; 8^* + 13\} = 12$, $d(x_7) = \min\{\infty; 8^* + 21\} = 29$.

Получаем $d(x_4) < d(x_3) < d(x_6) = d(x_7)$. Следовательно, вершине x_4 присваивается постоянная метка: $d(x_4) = 10^*$, $\tilde{x} = x_4$.

Шаг 5. $\tilde{s} = \{ x_3,\ x_6,\ x_7 \}$. Пересчитаем временные метки:

$d(x_3) = \min\{\infty; \underline{5^* + 7}, 8^* + 6\} = 12$,

$d(x_6) = \min\{\infty; \underline{0^* + 12}, 5^* + 9; 8^* + 13\} = 12$,

$d(x_7) = \min\{\infty; 8^* + 21; \underline{10^* + 14}\} = 24$.

Получаем $d(x_3) = d(x_6) < d(x_7)$. Следовательно, вершинам x_3 и x_6 присваиваются постоянные метки: $d(x_3) = 12^*$, $d(x_6) = 12^*$, $\tilde{x} = x_3$, $\tilde{x} = x_6$.

Шаг 6. $d(x_7) = \min\{\infty; 12^* + 11; 10^* + 14; 8^* + 21; \underline{12^* + 9}\} = 21$. Вершине x_7 присваивается постоянная метка $d(x_7) = 21^*$, $\tilde{x} = x_7$.

Этап 2.

Проводим последовательный поиск дуг кратчайшего пути.

Вершине $\tilde{x} = x_7$ предшествуют вершины $\tilde{s} = \{x_3, x_4, x_5, x_6\}$. Кратчайшее расстояние получим при прохождении по дуге $(x_6; x_7)$.

Вершине $\tilde{x} = x_6$ предшествуют вершины $\tilde{s} = \{x_1, x_2, x_3, x_5\}$. Кратчайшее расстояние получим при прохождении по дуге $(x_1; x_6)$.

Таким образом, кратчайший путь от вершины x_1 до вершины x_7 построен. Его длина (вес) составляет 21, т.е. $d_{\min} = 21$, сам путь задает следующую последовательность дуг $(x_1; x_6) - (x_6; x_7)$.

Ответ: $d_{\min} = 21$, $(x_1; x_6) - (x_6; x_7)$. □

Пример 4. На одной и той же поточной линии предприятие может обрабатывать пять видов деталей. Время наладки при переходе линии на обработку от одного вида деталей к другому представлена матрицей $A=\left[a_{ij}\right]$, где a_{ij} – затраты времени (часы) на наладку переменно-поточной линии при переходе к обработке деталей от i-го вида к деталям j-го вида. С помощью алгоритма Литтла найти последовательность перестройки линии с одной детали на другую, при которой должны быть обеспечены минимальные общие потери рабочего времени на ее переналадки.

$$A=\left(\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & \infty & 10 & 8 & 11 & 10 \\ 2 & 10 & \infty & 13 & 9 & 10 \\ 3 & 8 & 13 & \infty & 9 & 11 \\ 4 & 11 & 9 & 9 & \infty & 12 \\ 5 & 10 & 10 & 11 & 12 & \infty \end{array}\right).$$

Решение.

Сначала получим приведенный вид данной матрицы. В каждом столбце определим минимальный элемент и запишем его в нижней строке:

$$A=\left(\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & \infty & 10 & 8 & 11 & 10 \\ 2 & 10 & \infty & 13 & 9 & 10 \\ 3 & 8 & 13 & \infty & 9 & 11 \\ 4 & 11 & 9 & 9 & \infty & 12 \\ 5 & 10 & 10 & 11 & 12 & \infty \\ \hline & 8 & 9 & 8 & 9 & 10 \end{array}\right).$$

Из каждого элемента столбца вычтем соответствующий минимальный элемент и получим матрицу A_0:

$$A_0=\left(\begin{array}{c|ccccc|c} & 1 & 2 & 3 & 4 & 5 & \\ \hline 1 & \infty & 1 & 0 & 2 & 0 & 0 \\ 2 & 2 & \infty & 5 & 0 & 0 & 0 \\ 3 & 0 & 4 & \infty & 0 & 1 & 0 \\ 4 & 3 & 0 & 1 & \infty & 2 & 0 \\ 5 & 2 & 1 & 3 & 3 & \infty & 1 \end{array}\right).$$

Матрица A_0 не является приведенной, поэтому определим минимальный элемент в каждой строке и вычтем его из всех элементов соответствующей строки. В результате получим приведенную матрицу A_0':

$$A_0' = \left(\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & \infty & 1 & 0 & 2 & 0 \\ 2 & 2 & \infty & 5 & 0 & 0 \\ 3 & 0 & 4 & \infty & 0 & 1 \\ 4 & 3 & 0 & 1 & \infty & 2 \\ 5 & 1 & 0 & 2 & 2 & \infty \end{array}\right).$$

Вычислим константу приведения φ_0:

$$\varphi_0 = 8 + 9 + 8 + 9 + 10 + 0 + 0 + 0 + 0 + 1 = 45.$$

Найдем степени каждого нуля – сумму минимальных элементов строки и столбца, в которых стоит нуль. К каждому нулю припишем вверху его степень:

$$A_0' = \left(\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & \infty & 1 & 0^{(1)} & 2 & 0^{(0)} \\ 2 & 2 & \infty & 5 & 0^{(0)} & 0^{(0)} \\ 3 & 0^{(1)} & 4 & \infty & 0^{(0)} & 1 \\ 4 & 3 & 0^{(1)} & 1 & \infty & 2 \\ 5 & 1 & 0^{(1)} & 2 & 2 & \infty \end{array}\right).$$

Максимальная степень равна 1. Нули с максимальной степенью определяют дуги, которые вероятнее всего войдут в гамильтонов цикл. В нашем случае наиболее вероятными дугами гамильтонова цикла являются $l(1;3)$, $l(3;1)$, $l(4;2)$, $l(5;2)$.

Выберем дугу $l(1;3)$. В связи с этим рассмотрим две матрицы: $A_1(1;3)$ и $\tilde{A}_1(1;3)$. В матрице $A_1(1;3)$ уберем первую строку и третий столбец, элемент a_{31} заменим на ∞. В матрице $\tilde{A}_1(1;3)$ элемент a_{13} заменим на ∞. Получим:

$$A_1(1;3) = \left(\begin{array}{c|cccc} & 1 & 2 & 4 & 5 \\ \hline 2 & 2 & \infty & 0 & 0 \\ 3 & \infty & 4 & 0 & 1 \\ 4 & 3 & 0 & \infty & 2 \\ 5 & 1 & 0 & 2 & \infty \end{array}\right), \qquad \tilde{A}_1(1;3) = \left(\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & \infty & 1 & \infty & 2 & 0 \\ 2 & 2 & \infty & 5 & 0 & 0 \\ 3 & 0 & 4 & \infty & 0 & 1 \\ 4 & 3 & 0 & 1 & \infty & 2 \\ 5 & 1 & 0 & 2 & 2 & \infty \end{array}\right).$$

Обе матрицы не являются приведенными. Для их приведения определим минимальные элементы столбцов и выполним их вычитание из элементов соответствующих столбцов:

$$A_1(1;3)=\left(\begin{array}{c|cccc} & 1 & 2 & 4 & 5 \\ \hline 2 & 2 & \infty & 0 & 0 \\ 3 & \infty & 4 & 0 & 1 \\ 4 & 3 & 0 & \infty & 2 \\ 5 & 1 & 0 & 2 & \infty \\ \hline & 1 & 0 & 0 & 0 \end{array}\right), \qquad \tilde{A}_1(1;3)=\left(\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & \infty & 1 & \infty & 2 & 0 \\ 2 & 2 & \infty & 5 & 0 & 0 \\ 3 & 0 & 4 & \infty & 0 & 1 \\ 4 & 3 & 0 & 1 & \infty & 2 \\ 5 & 1 & 0 & 2 & 2 & \infty \\ \hline & 0 & 0 & 1 & 0 & 0 \end{array}\right).$$

В результате получим следующие приведенные матрицы $A_1(1;3)$ и $\tilde{A}_1(1;3)$.

$$A_1(1;3)=\left(\begin{array}{c|cccc} & 1 & 2 & 4 & 5 \\ \hline 2 & 1 & \infty & 0 & 0 \\ 3 & \infty & 4 & 0 & 1 \\ 4 & 2 & 0 & \infty & 2 \\ 5 & 0 & 0 & 2 & \infty \end{array}\right), \qquad \tilde{A}_1(1;3)=\left(\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & \infty & 1 & \infty & 2 & 0 \\ 2 & 2 & \infty & 4 & 0 & 0 \\ 3 & 0 & 4 & \infty & 0 & 1 \\ 4 & 3 & 0 & 0 & \infty & 2 \\ 5 & 1 & 0 & 1 & 2 & \infty \end{array}\right).$$

Вычислим константы приведения:

$$\varphi_1(1;3)=\varphi_0+\Delta_1(1;3)=45+1=46, \qquad \tilde{\varphi}_1(1;3)=\varphi_0+\tilde{\Delta}_1(1;3)=45+1=46,$$

где $\Delta_1(1;3)$ и $\tilde{\Delta}_1(1;3)$ – суммы минимальных элементов строк и столбцов матриц $A_1(1;3)$ и $\tilde{A}_1(1;3)$.

Так как $\varphi_1(1;3)=\tilde{\varphi}_1(1;3)$, то далее будем рассматривать матрицу $A_1(1;3)$, поскольку ее размерность меньше, чем размерность матрицы $\tilde{A}_1(1;3)$. Определим степени нулей матрицы $A_1(1;3)$:

$$A_1(1;3)=\left(\begin{array}{c|cccc} & 1 & 2 & 4 & 5 \\ \hline 2 & 1 & \infty & 0^{(0)} & 0^{(1)} \\ 3 & \infty & 4 & 0^{(1)} & 1 \\ 4 & 2 & 0^{(2)} & \infty & 2 \\ 5 & 0^{(1)} & 0^{(0)} & 2 & \infty \end{array}\right).$$

Максимальная степень – число 2. Наиболее вероятной дугой гамильтонова цикла является дуга $l(4;2)$.

В связи с этим рассмотрим две матрицы: $A_2(4;2)$ и $\tilde{A}_2(4;2)$. В матрице $A_2(4;2)$ уберем строку под номером четыре и столбец под номером два, элемент a_{24} заменим на ∞. В матрице $\tilde{A}_2(4;2)$ элемент a_{42} заменим на ∞. Получим:

$$A_2(4;2)=\left(\begin{array}{c|ccc} & 1 & 4 & 5 \\ \hline 2 & 1 & \infty & 0 \\ 3 & \infty & 0 & 1 \\ 5 & 0 & 2 & \infty \end{array}\right), \qquad \tilde{A}_2(4;2)=\left(\begin{array}{c|cccc} & 1 & 2 & 4 & 5 \\ \hline 2 & 1 & \infty & 0 & 0 \\ 3 & \infty & 4 & 0 & 1 \\ 4 & 2 & \infty & \infty & 2 \\ 5 & 0 & 0 & 2 & \infty \end{array}\right).$$

Матрица $A_2(4;2)$ получена приведенной. Матрица $\tilde{A}_2(4;2)$ не является приведенной. Определим минимальные элементы строк матрицы $\tilde{A}_2(4;2)$ и вычтем их из элементов соответствующих строк:

$$\tilde{A}_2(4;2)=\left(\begin{array}{c|cccc|c} & 1 & 2 & 4 & 5 & \\ \hline 2 & 1 & \infty & 0 & 0 & 0 \\ 3 & \infty & 4 & 0 & 1 & 0 \\ 4 & 2 & \infty & \infty & 2 & 2 \\ 5 & 0 & 0 & 2 & \infty & 0 \end{array}\right) \Rightarrow \quad \tilde{A}_2(4;2)=\left(\begin{array}{c|cccc} & 1 & 2 & 4 & 5 \\ \hline 2 & 1 & \infty & 0 & 0 \\ 3 & \infty & 4 & 0 & 1 \\ 4 & 0 & \infty & \infty & 0 \\ 5 & 0 & 0 & 2 & \infty \end{array}\right).$$

Вычислим константы приведения:

$$\varphi_2(4;2)=\varphi_1(1;3)+\Delta_2(4;2)=46+0=46,$$

$$\tilde{\varphi}_2(4;2)=\varphi_1(1;3)+\tilde{\Delta}_2(4;2)=46+2=48.$$

Так как $\varphi_2(4;2)<\tilde{\varphi}_2(4;2)$, то далее будем рассматривать матрицу $A_2(4;2)$, для которой определим степени каждого нуля:

$$A_2(4;2)=\left(\begin{array}{c|ccc} & 1 & 4 & 5 \\ \hline 2 & 1 & \infty & 0^{(2)} \\ 3 & \infty & 0^{(3)} & 1 \\ 5 & 0^{(3)} & 2 & \infty \end{array}\right).$$

Максимальная степень – число 3. Наиболее вероятными дугами гамильтонова цикла являются дуги $l(3;4)$ и $l(5;1)$.

Выберем дугу $l(3;4)$. В связи с этим рассмотрим две матрицы: $A_3(3;4)$ и $\tilde{A}_3(3;4)$. В матрице $A_3(3;4)$ уберем строку под номером три и столбец под номером четыре. В матрице $\tilde{A}_3(3;4)$ элемент a_{34} заменим на ∞. Получим:

$$A_3(3;4)=\left(\begin{array}{c|cc} & 1 & 5 \\ \hline 2 & 1 & 0 \\ 5 & 0 & \infty \end{array}\right), \qquad \tilde{A}_3(3;4)=\left(\begin{array}{c|ccc} & 1 & 4 & 5 \\ \hline 2 & 1 & \infty & 0 \\ 3 & \infty & \infty & 1 \\ 5 & 0 & 2 & \infty \end{array}\right).$$

Матрица $A_3(3;4)$ – приведенная. Матрица $\tilde{A}_3(3;4)$ не является приведенной. Определим минимальные элементы строк и столбцов матрицы $\tilde{A}_3(3;4)$ и выполним приведение:

$$\tilde{A}_3(3;4)=\left(\begin{array}{c|ccc|c} & 1 & 4 & 5 & \\ \hline 2 & 1 & \infty & 0 & 0 \\ 3 & \infty & \infty & 1 & 1 \\ 5 & 0 & 2 & \infty & 0 \\ \hline & 0 & 2 & 0 & \end{array}\right) \Rightarrow \quad \tilde{A}_3(3;4)=\left(\begin{array}{c|ccc} & 1 & 4 & 5 \\ \hline 2 & 1 & \infty & 0 \\ 3 & \infty & \infty & 0 \\ 5 & 0 & 0 & \infty \end{array}\right).$$

Вычислим константы приведения:

$$\varphi_3(3;4)=\varphi_2(4;2)+\Delta_3(3;4)=46+0=46,$$
$$\tilde{\varphi}_3(3;4)=\varphi_2(4;2)+\tilde{\Delta}_3(3;4)=46+1+2=49.$$

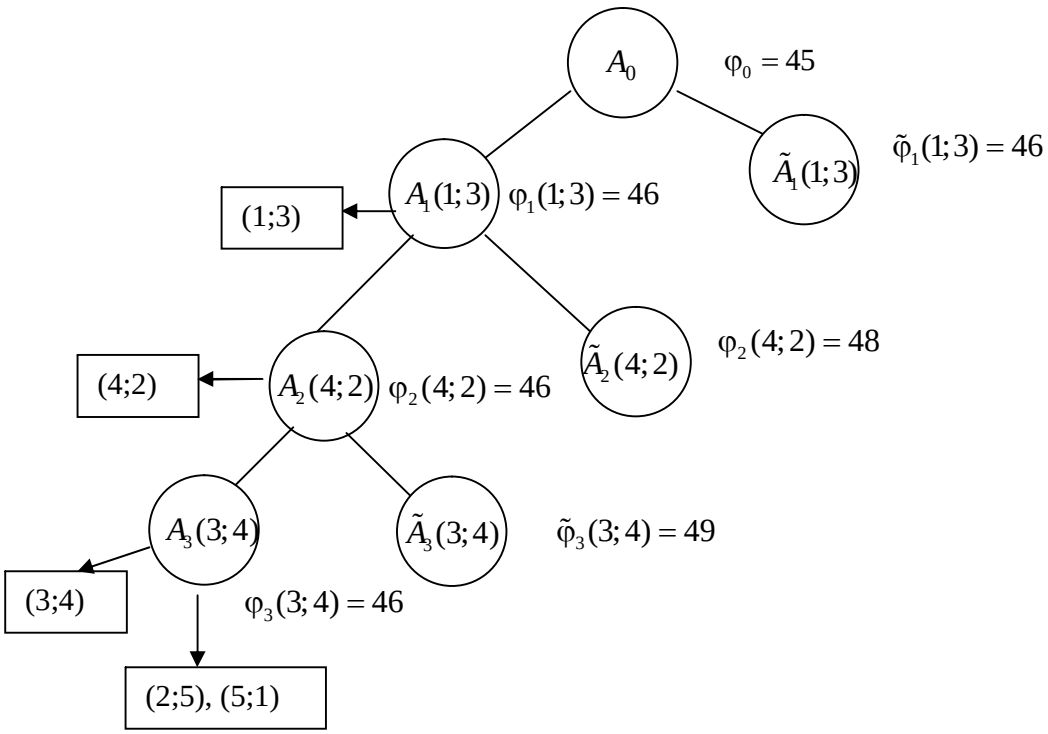
Так как $\varphi_3(3;4)<\tilde{\varphi}_3(3;4)$, то рассмотрим матрицу $A_3(3;4)$. Последние две дуги гамильтонова цикла определим из этой матрицы: $l(2;5)$ и $l(5;1)$.

Сравним константу приведения $\varphi_3(3;4)$ с константами приведения $\tilde{\varphi}_k$ альтернативных вариантов ($k=\overline{1,3}$):

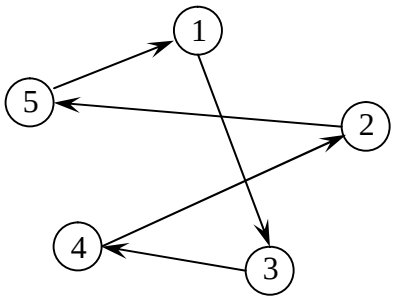
$$\varphi_3(3;4)=\tilde{\varphi}_1(1;3): \quad 46=46;$$
$$\varphi_3(3;4)<\tilde{\varphi}_2(4;2): \quad 46<48;$$
$$\varphi_3(3;4)<\tilde{\varphi}_3(3;4): \quad 46<49.$$

Так как $\varphi_3(3;4)\le\tilde{\varphi}_k$ ($k=\overline{1,3}$), то полученное решение, состоящее из дуг $(1;3)$, $(4;2)$, $(3;4)$, $(2;5)$, $(5;1)$ является оптимальным, но не единственным.

Весь процесс поиска оптимального плана можно изобразить в виде дерева ветвления, которое изображено на рисунке:



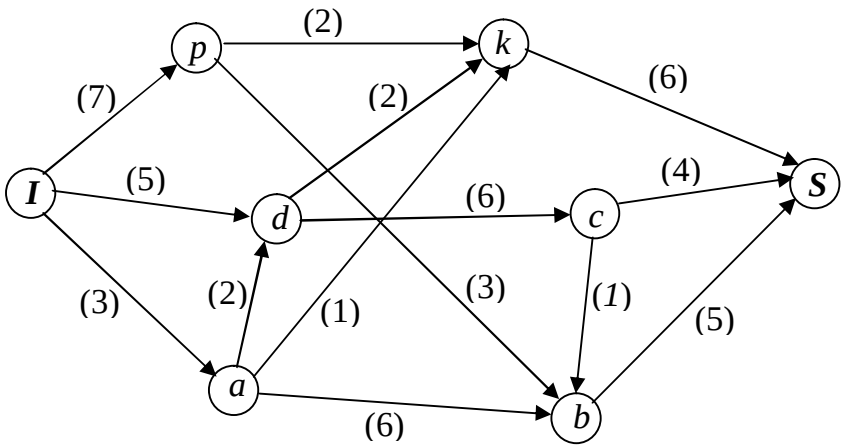
Из полученных дуг составим замкнутый гамильтонов цикл: $1 \to 3 \to 4 \to 2 \to 5 \to 1$, который можно представить в виде орграфа:



Построенный гамильтонов цикл соответствует последовательности перестройки линии с обработки одной детали на другую, что означает следующее: после обработки деталей первого вида следует наладить обработку деталей третьего вида: после третьего – четвертого вида; после четвертого – второго вида; после второго – пятого вида; после пятого – снова вернуться к обработке первого вида деталей. При этом минимальные общие потери рабочего времени составят $\varphi = \varphi_3(3;4) = 46$ часов.

Ответ: последовательность перестройки линии: $1 \to 3 \to 4 \to 2 \to 5 \to 1$; минимальные общие потери рабочего времени составят 46 часов. □

Пример 5. На заданной сети в скобках указаны пропускные способности дуг. Требуется сформировать на сети максимальный поток, направленный из истока I в сток S, и выписать ребра, образующие на сети разрез минимальной пропускной способности.



Решение.

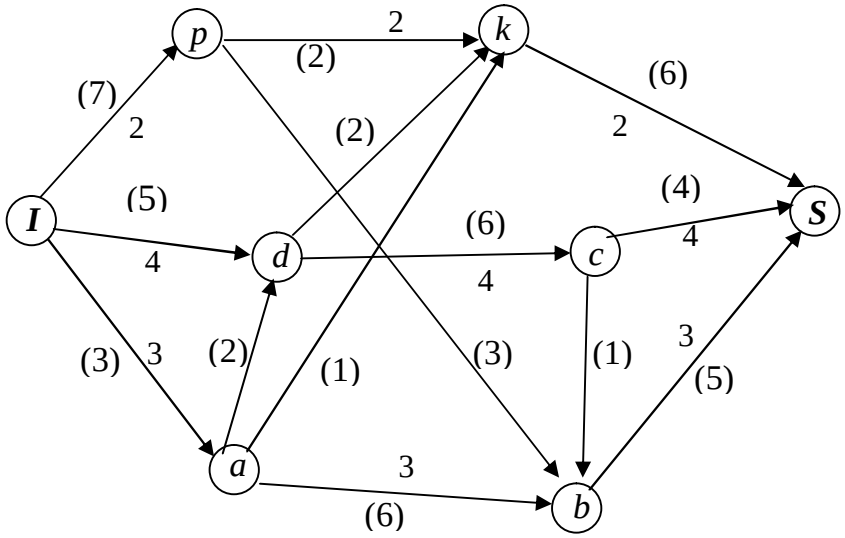
Этап 1.

Сформируем на сети начальный поток. Рассмотрим путь $I \to p \to k \to S$. Поскольку $\min\{7; 2; 6\} = 2$, по этому пути пропускаем поток в 2 единицы. На сети значение потока обозначим числами без скобок. По пу-

ти $I \to d \to c \to S$ пропускаем поток в 4 единицы, так как $\min\{5; 6; 4\} = 4$. По пути $I \to a \to b \to S$ пропускаем поток в 3 единицы, так как $\min\{3; 6; 5\} = 3$. Таким образом, начальный поток имеет вид:

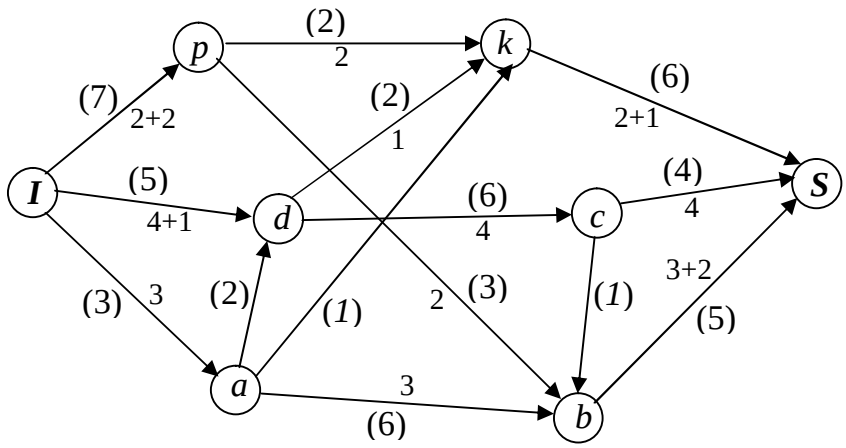
$$I \xrightarrow{2} p \xrightarrow{2} k \xrightarrow{2} S,$$
$$I \xrightarrow{4} d \xrightarrow{4} c \xrightarrow{4} S,$$
$$I \xrightarrow{3} a \xrightarrow{3} b \xrightarrow{3} S.$$

Начальный поток изображен на следующем рисунке.

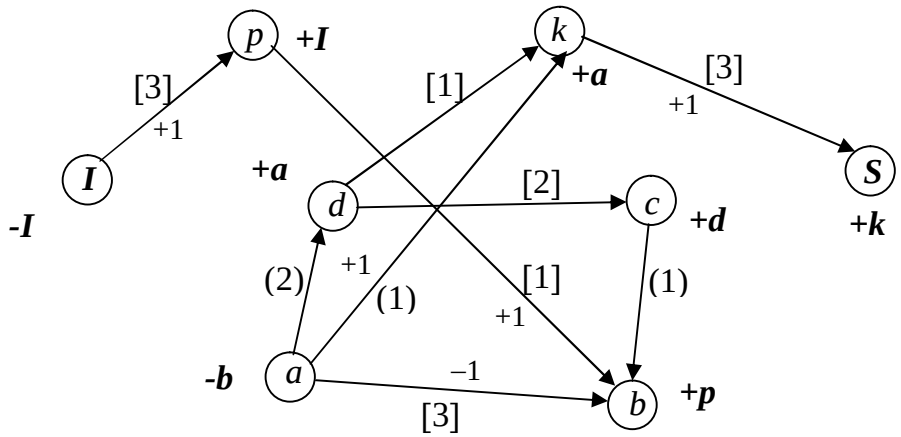


Каждый из рассмотренных путей содержит насыщенную дугу, поэтому эти пути насыщенные. На сети есть еще пути, которые содержат ненасыщенные дуги, а именно: $I \to p \to b \to S$, $I \to d \to k \to S$ и $I \to d \to c \to b \to S$. Для первого пути дополнительно увеличим поток на 2 единицы, так как $\min\{7-2; 3; 5-3\} = 2$. Второй и третий пути содержат одну и ту же дугу $(I; d)$ с минимальной для них оставшейся разностью $q(I; d) = 1$. Поэтому для увеличения потока на 1 единицу выберем, например, путь $I \to d \to k \to S$.

Теперь каждый из этих путей содержит насыщенную дугу, следовательно, полученный поток – насыщенный, (рисунок).



Выясним, является ли построенный поток максимальным. Изобразим сеть, на которой отметим все вершины и ненасыщенные дуги. На этой сети разность пропускной способности дуги и проходящего по ней потока обозначим числом в квадратных скобках. Пропускную способность дуги, по которой поток не проходит, оставим в круглых скобках, (рисунок).



Согласно рисунку на сети исток I и сток S связаны дугами. Значит, можно добавить какое-то количество потока по ненасыщенным дугам, при этом придется перераспределить поток.

Этап 2.

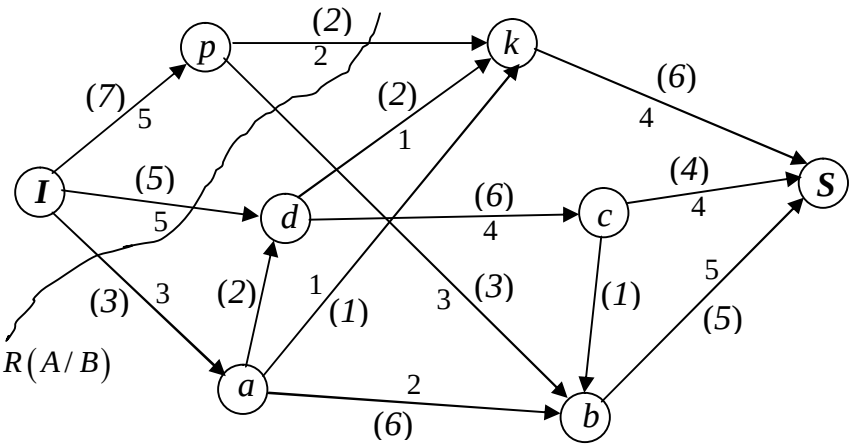
На построенной сети помечаем вершины. Вершину I пометим $-I$. Смежную ей вершину p пометим $+I$, так как эти вершины соединяет ненасыщенная дуга $I \to p$. Вершину b помечаем $+p$, так как вершины p и b соединяет ненасыщенная дуга $p \to b$. Вершину a помечаем $-b$, так как вершины b и a соединяет непустая дуга $b \leftarrow a$. Вершины d и k помечаем $+a$, так как они соединены с вершиной a ненасыщенными дугами $a \to d$ и $a \to k$. Вершина c смежная вершине d, и эти вершины соединены ненасыщенной дугой $d \to c$, поэтому вершину c помечаем $+d$. Вершина S смежная вершине k, и эти вершины соединены ненасыщенной дугой $k \to S$, поэтому вершину S помечаем $+k$.

Вершина S оказалась помеченной. Значит, существует последовательность помеченных вершин от I к S: $I \to p \to b \leftarrow a \to k \to S$. В этой последовательности каждая последующая вершина помечена буквой предыдущей вершины. Перераспределим поток на этом пути. Определим число δ:

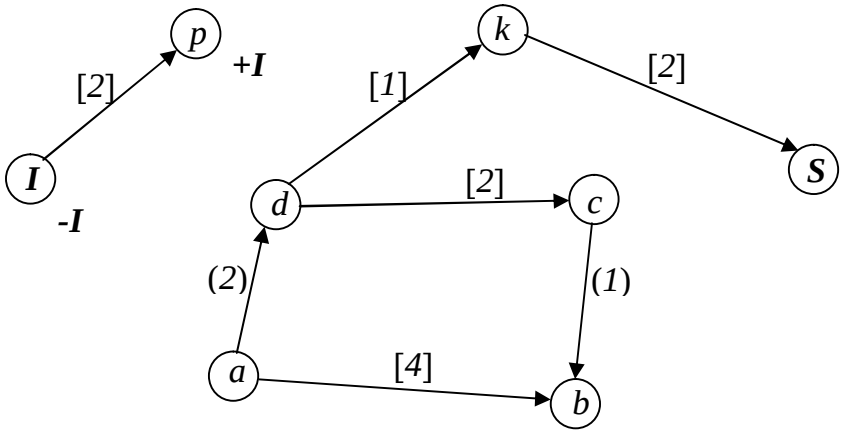
$$\delta = \min\{7-4;\ 3-2;\ 6-3;\ 1;\ 6-3\} = \min\{3;1;3;1;3\} = 1.$$

Увеличим на 1 единицу поток на дугах, имеющих направление от I к S: $(I;p)$, $(p;b)$, $(a;k)$, $(k;S)$. Уменьшим на 1 единицу поток на дугах, имею-

щих обратное направление: $(a;b)$. Получим следующую сеть с новым сформированным потоком, который изображен на рисунке.



Проверим, будет ли построенный поток максимальным. Изобразим сеть, на которой отметим все вершины и ненасыщенные дуги, следующий рисунок.



Вновь пометим вершины. Вершину I пометим $-I$. Смежную ей вершину p пометим $+I$, так как эти вершины соединяет ненасыщенная дуга $I \to p$. Все остальные вершины, в том числе и вершина S, остаются непомеченными. Значит, поток на сети максимальный.

Этап 3.

Согласно последнему рисунку, на последней сети исток I и сток S не связаны дугами. Значит поток, изображенный на предпоследнем рисунке, является максимальным.

Строим разрез на сети. Разбиваем множество вершин на два подмножества: A и B. Помеченные вершины образуют множество $A=\{I;p\}$, непоме-

ченные – множество $B=\{a;d;k;c;b;S\}$. Проводим разрез $R(A/B)$, который состоит из дуг $(p;k)$, $(I;d)$, $(I;a)$, $(p;b)$:

$$R(A/B)=\{(I;a),(I;d),(p;k),(p;b)\}.$$

Определим величину максимального потока $F_{\max}$:

$$F_{\max}=C_{\min}(A/B)=2+3+5+3=13 \text{ (ед.)}.$$

Ответ: мощность максимального потока составляет 13 ед., разрез минимальной пропускной способности образуют ребра $(I;a),(I;d),(p;k),(p;b)$.

□