МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ
УНИВЕРСИТЕТ имени академика С. П. КОРОЛЕВА
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»

Н.А. КАЛУГИН, А.Н. КАЛУГИН

ЭЛЕМЕНТЫ ТЕОРИИ ГРАФОВ

Рекомендовано к изданию редакционно-издательским советом федерального государственного бюджетного образовательного учреждения высшего профессионального образования «Самарский государственный аэрокосмический университет имени академика С.П. Королева (национальный исследовательский университет) в качестве учебного пособия

САМАРА
Издательство СГАУ
2013

УДК 519.2(075)
ББК 22.176я7
К176

Рецензенты: канд. физ.-мат. наук, доц. Е. Я. Горелова,
канд. техн. наук, доц. Ю.С. Горшков

Калугин Н.А.
К176 **Элементы теории графов:** учеб. пособие / *Н.А.Калугин, А.Н.Калугин.* – Самара: Изд-во Самар. гос. аэрокосм. ун-та, 2013. – 48с.: ил.

ISBN 978-5-7883-0950-7

В пособии содержатся основные сведения о теории графов, приводятся примеры задач, решаемых методами теории графов, описываются алгоритмы их решения. Изложение материала иллюстрируется примерами. Приведен вариант расчетной работы по теории графов.

Рассчитано на студентов экономических специальностей, но будет полезно и студентам других специальностей, изучающим высшую математику.

УДК 519.2(075)
ББК 22.176я7

ISBN 978-5-7883-0950-7

© Самарский государственный
аэрокосмический университет, 2013

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Теория графов является важным разделом современной математики. Практическая ее роль особенно возросла за последнее время в связи с распространением вычислительной техники. Успех применения графов можно объяснить тем, что они являются удобным языком для формулировки и эффективным инструментом для решения задач, относящихся к широкому кругу проблем, таких, например, как теория игр, программирование, конструирование электрических и контактных цепей, задачи экономики, управления. Поэтому владение методами теории графов является необходимой составной частью образования специалистов, занимающихся вопросами прикладной математики, экономики и управления.

Цель настоящего пособия – дать читателю представление о графах и научить его самостоятельно решать некоторые задачи. Помимо этого студентам предлагается выполнить расчетно-графическую работу, основанную на материалах пособия.

Работа состоит из двух частей, в первой из которых проводятся операции с неориентированным графом, а во второй - с ориентированным.

Первая часть включает следующие задания:

1. Составить матрицу смежности неориентированного графа.
2. С помощью волнового метода определить условные радиусы графа относительно всех вершин, радиус, диаметр и центры графа.
3. Методом редукции индекса найти расстояние от вершины 1 до всех остальных вершин графа и от всех вершин – до вершины с максимальным номером N. Указать цепи минимальной длины $1-N$ и $2-N$.
4. С помощью алгоритма Флери построить, если возможно, эйлеров цикл (эйлерову цепь) в графе.
5. Используя алгоритм Тэрри, построить в графе двойной эйлеров цикл, начиная с вершины 1.
6. С помощью алгоритма Краскала построить остов графа наименьшего веса и определить его вес.

Во второй части требуется выполнить следующие задания:

1. Построить матрицу смежности ориентированного графа.

2. Методом редукции индекса определить расстояние от вершины 1 до всех остальных вершин графа и от всех вершин до вершины N. Указать цепи минимальной длины $1-N$ и $2-N$.

3. Решить задачу сетевого планирования:
   - составить план работ, соответствующий сетевому графику G;
   - перенумеровать вершины с целью получения правильной нумерации;
   - определить ранние сроки наступления событий T_i^P;
   - определить ранние сроки окончания работ t_{ij}^{PO};
   - указать критическое время и критический путь;
   - найти поздние сроки наступления событий T_j^{Π};
   - определить поздние сроки начала работ $t_{ij}^{\Pi H}$;
   - найти полный R_{ij} и свободный r_{ij} резервы времени работ.
4. С помощью алгоритма Форда-Фалкерсона найти максимальный поток $\Phi_{\max}$ на транспортной сети и указать критическое сечение.

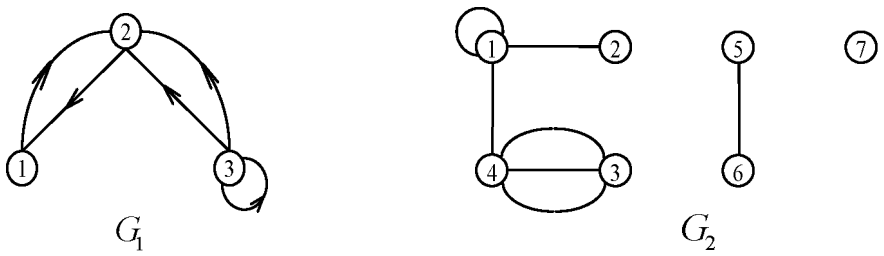
Исходным для выполнения работы является рисунок графа, который, в зависимости от выполняемого задания, модифицируется соответствующим образом (например, в первой части не учитывается ориентация ребер графа).

1. ОСНОВНЫЕ ПОНЯТИЯ ТЕОРИИ ГРАФОВ

Граф G – это совокупность двух конечных множеств: множества точек, которые называются вершинами, и множества ребер. Число вершин графа не может быть равным нулю.

Каждое ребро можно рассматривать как пару точек. Будем считать, что ребро $i-j$ соединяет вершины i и j. Если на ребре задано направление (пара $i-j$ упорядочена), то оно называется ориентированным ребром или дугой. Будем считать, что дуга $i-j$ начинается в вершине i и заканчивается в вершине j. Ясно, что для неориентированных ребер порядок вершин в паре $i-j$ безразличен.

Граф G, содержащий хотя бы одну дугу, называется ориентированным графом (орграфом). В противном случае G - неориентированный граф. На рис. 1.1 изображены орграф G_1, содержащий 3 вершины и 5 ребер, и неориентированный граф G_2, имеющий 7 вершин и 7 ребер.



G_1 G_2

Рис 1.1. Примеры графов

Ребра, начинающиеся и заканчивающиеся в одних и тех же вершинах, называются кратными. Например, в графе G_1 (рис 1.1) - два кратных ребра (дуги) $3-2$, в графе G_2 кратными являются 3 ребра $4-3$. Ребра $1-2$ и $2-1$ в графе G_1 не являются кратными!

Ребро, у которого начало и конец совпадают (ребра $3-3$ в графе G_1 и $1-1$ в графе G_2), называется петлей.

Граф G, не содержащий петель и кратных ребер, называется простым, в противном случае – общим.

Две вершины, связанные ребром, называются смежными. Матрицей смежности графа G (имеющего n вершин) называется квадратная матрица $[A]$ порядка n, каждый элемент a_{ij} которой равен количеству ребер $i-j$ графа. Например, для графов G_1 и G_2 (рис 1.1) матрицы смежности $[A_1]$ и $[A_2]$ будут, соответственно, иметь следующий вид:

$$[A_1]=\begin{pmatrix}0&1&0\\1&0&0\\0&2&1\end{pmatrix};\quad [A_2]=\begin{pmatrix}1&1&0&1&0&0&0\\1&0&0&0&0&0&0\\0&0&0&3&0&0&0\\1&0&3&0&0&0&0\\0&0&0&0&0&1&0\\0&0&0&0&1&0&0\\0&0&0&0&0&0&0\end{pmatrix}.$$

Матрица смежности – удобный способ задания графа при вычислениях на компьютере.

Маршрутом из вершины i_0 в вершину i_n в графе G называется последовательность вершин $i_0-i_1-i_2-\ldots-i_n$, в которой для любой пары соседних вершин i_k, i_{k+1} существует ребро i_k-i_{k+1}. Можно определить маршрут и как последовательность ребер, в которой конец предыдущего ребра является началом последующего.

Маршрут, все ребра которого различны, называется цепью (путем).

Цепь называется простой цепью, если все промежуточные вершины цепи i_k $(k=1,\ldots,n-1)$ различны.

Цепь называется замкнутой цепью, если начало и конец ее совпадают $(i_0 = i_n)$.

Замкнутая простая цепь называется циклом. В частности, любая петля – цикл.

Граф G называется связным, если для любых двух его вершин существует цепь, их соединяющая. В противном случае G – несвязный граф.

Граф G_1 называется подграфом графа G, если все вершины G_1 являются вершинами G и все ребра G_1 принадлежат множеству ребер G.

Всякий граф является совокупностью связных графов (подграфов). Эти графы обладают тем свойством, что никакая вершина одного их них не связана цепью с любой вершиной другого. Каждый из этих подграфов называется компонентой связности графа G. Например, у связного графа G_1 (рис 1.1) – одна, а у несвязного графа G_2 – три компоненты связности.

Ребро $i - j$ называется мостом графа G, если граф, получившийся из G после удаления ребра $i - j$, содержит больше компонент связности, чем граф G. Например, ребра $1-4, 1-2, 5-6$ – мосты графа G_2 (рис 1.1). Граф G_1 мостов не имеет.

Степенью $\rho(i)$ вершины i неориентированного графа называется число выходящих из нее ребер (в графе G_2: $\rho(1) = 4$, $\rho(2) = \rho(5) = \rho(6) = 1$, $\rho(3) = 3$, $\rho(4) = 4, \rho(7) = 0$).

Положительной степенью $\rho^+(i)$ вершины i ориентированного графа называется число выходящих из нее ребер (в графе G_1: $\rho^+(1) = 1$, $\rho^+(2) = 1$, $\rho^+(3) = 3$).

Отрицательной степенью $\rho^-(i)$ вершины i ориентированного графа называется число входящих в нее ребер (в графе G_1: $\rho^-(1)=1$, $\rho^-(2)=3$, $\rho^-(3)=1$).

Вершина k ориентированного графа называется источником, если $\rho^-(k)=0$, $\rho^+(k)>0$.

Вершина k ориентированного графа называется стоком, если $\rho^+(k)=0$, $\rho^-(k)>0$.

2. РАССТОЯНИЯ В ГРАФАХ

Расстоянием $d(m-n)$ от вершины m до вершины n в связном графе G называется длина кратчайшей простой цепи из m в n. В зависимости от рассматриваемой задачи под длиной цепи могут пониматься различные величины.

2.1 Определение расстояний в графах без учета длин ребер. Волновой метод

Если длина ребер не учитывается, то при определении расстояния под длиной цепи понимается число ребер в ней.

Условным радиусом $r_i(G)$ неориентированного графа G относительно вершины i называется максимальное расстояние от вершины i до других вершин графа, т.е.

$$r_i(G) = \max_j d(i-j). \qquad (2.1)$$

Наибольший из условных радиусов графа называется диаметром графа $\delta(G)$, т.е.

$$\delta(G) = \max_i r_i(G). \qquad (2.2)$$

Наименьший из условных радиусов называется радиусом графа $r(G)$, т.е.

$$r(G) = \min_i r_i(G). \qquad (2.3)$$

Вершина k, для которой условный радиус равен радиусу графа, т.е. $r_i(G) = r(G)$, называется центром графа. Ясно, что у графа может быть несколько центров.

Расстояние от вершины m до вершины n в графе G определяется с помощью волнового метода, алгоритм которого описан ниже.

1. Пометим вершину m меткой (индексом) 0.

2. Пусть на некотором этапе алгоритма присвоены метки от 0 до k включительно. На очередном шаге индекс $k+1$ присваивается всем ранее не помеченным вершинам j, если существует ребро $i-j$, связывающее вершину i, имеющую некоторую метку k, с вершиной j.
3. Пусть на некотором шаге вершина n помечена меткой N. Процесс индексации останавливаем. Расстояние от m до n равно N, т.е.

$$d(m-n)=N. \qquad (2.4)$$

4. Определим цепь кратчайшей длины N из m в n. По построению можно найти вершину i_{N-1}, смежную с n и помеченную индексом $N-1$; по тем же соображениям существует вершина i_{N-2}, смежная с i_{N-1} и помеченная меткой $N-2$, и т. д. до вершины m, смежной с вершиной i_1.

Искомая цепь с наименьшим числом ребер, равным N, представляет собой построенную последовательность $m-i_1-i_2-i_3-\ldots-i_{N-1}-n$. Очевидно, что таких цепей может быть несколько.

Замечания.

1. Если на некотором этапе присвоить очередную метку невозможно, а вершина n еще не помечена, то не существует маршрута из m в n.
2. Если не останавливать процесс индексации в п.3, то через некоторое количество шагов все вершины в связном графе будут помечены. При этом наибольший индекс равен максимальному расстоянию от вершины m до других вершин графа (условному радиусу $r_m(G)$ в неориентированном графе).

Таким образом, волновым методом можно найти условные радиусы неориентированного графа относительно всех его вершин. Сравнивая их, легко определить радиус, диаметр и центры графа.

Пример 2.1. Определить радиус, диаметр, центры графа G (рис.. 2.1). Указать цепь минимальной длины $1-6$.

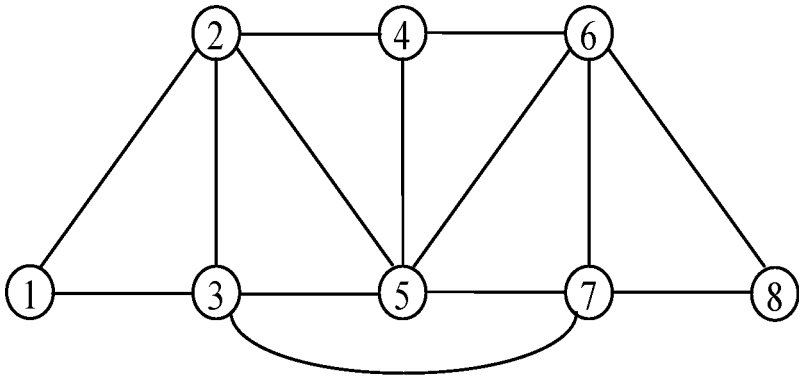


Рис. 2.1. Исходный граф

Построим волну из вершины 1 (рис. 2.2)

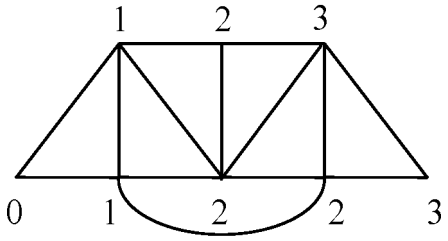


Рис. 2.2. Определение $r_1(G)$

Расстояние $d(1-6)=3$. Цепи из 1 в 6 длины 3: **1-2-4-6**; **1-2-5-6**; **1-3-7-6**. Условный радиус $r_1(G)=3$. Из соображений симметрии ясно, что $r_8(G)=r_1(G)=3$.

Процедура определения остальных условных радиусов представлена на рис. 2.3.

Сравнивая относительные радиусы, можно найти диаметр ($\delta(G)=3$) и радиус графа ($r(G)=2$). Центрами графа являются вершины **3, 4, 5, 7** (те, для которых условный радиус равен 2).

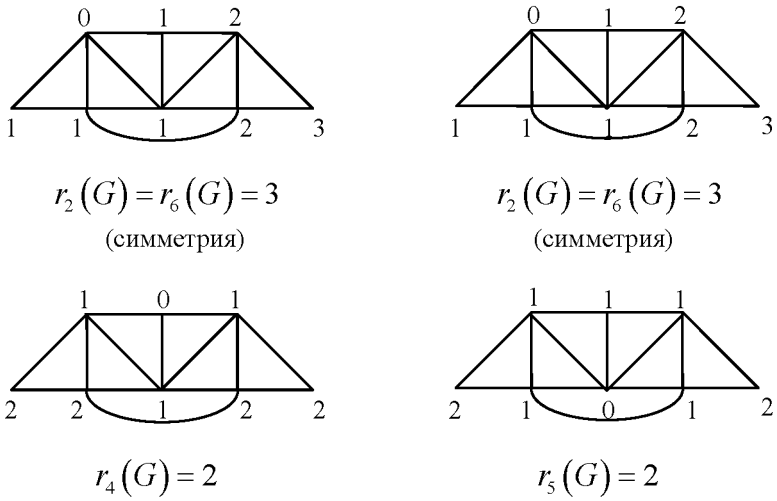


Рис 2.3. Условные радиусы графа

2.2 Определение расстояний в графах с учетом длин ребер. Метод редукции индекса

Пусть каждому ребру $i-j$ графа G поставлено в соответствие некоторое положительное число $\mu(i-j)$, которое называется длиной ребра. При определении расстояния в этом случае под длиной цепи понимается сумма длин входящих в нее ребер.

Введем в рассмотрение числа

$$\mu_{ij} = \begin{cases} 0, \text{ если } i = j; \\ \infty, \text{ если в графе нет ребра } i-j; \\ \mu(i-j), \text{ если в графе есть ребро } i-j. \end{cases} \quad (2.5)$$

Расстояния от вершины m до других вершин графа определяется с помощью метода редукции индекса, алгоритм которого приведен ниже.

1. Все вершины заносятся в список, начиная с вершины m, и им присваиваются индексы

$$M_m = 0;\ M_j = \infty\ (j \neq m). \quad (2.6)$$

2. На каждом основном шаге в порядке очередности списка просматриваются все вершины графа кроме m и вычисляются новые индексы по формуле

$$M_j = \min_i \left[M_i + \mu_{ij} \right]. \qquad (2.7)$$

Рядом с каждым индексом вершины j в скобках отмечаем номер вершины, для которой достигается указанный минимум. Если таких вершин несколько, то указывается номер любой из них.

3. Пункт 2 повторяется до тех пор, пока на очередном шаге индексы всех вершин будут равны индексам, вычисленным на предыдущем шаге. Последние индексы вершин и представляют собой расстояния от вершины m до других вершин графа, т.е.

$$d(m-j) = M_j. \qquad (2.8)$$

4. Цепь кратчайшей длины, связывающая вершину m с произвольной вершиной j, определяется следующим образом.

Для удобства обозначим число, стоящее в скобках рядом с последним индексом вершины k через $N(k)$. Тогда можно построить последовательность

$$j \Rightarrow N(j) = i_1 \Rightarrow N(i_1) = i_2 \Rightarrow \ldots \Rightarrow N(i_n) = m. \qquad (2.9)$$

Искомая кратчайшая цепь из m в j представляет собой последовательность (2.9), взятую в обратном порядке, т.е. последовательность

$$m - i_n - i_{n-1} - \ldots - i_2 - i_1 - j. \qquad (2.10)$$

Замечания.

1. Вычисления обычно оформляются в виде таблицы (см. пример ниже).

2. В формуле (2.7) следует учитывать только вершины i, связанные ребром $i-j$ с вершиной j, индекс которых M_i в данный момент меньше предыдущего значения индекса M_j.

3. Для неориентированного графа $d(m-j) = d(j-m) = M_j$. Кратчайшая цепь из j в m строится по номерам (2.9) в прямом порядке, т.е.

$$j - i_1 - i_2 - \ldots - i_{n-1} - i_n - m. \qquad (2.11)$$

4. Для ориентированных графов описанный выше алгоритм позволяет получить кратчайший путь из m во все другие вершины графа G. Для того, чтобы определить кратчайший путь из других вершин в m, надо снова использовать метод редукции индекса, предварительно видоизменив формулу (2.5):

$$\mu_{ij} = \begin{cases} 0, \text{ если } i = j; \\ \infty, \text{ если в графе нет ребра } j - i; \\ \mu(i - j), \text{ если в графе есть ребро } j - i. \end{cases} \qquad (2.12)$$

Практически это означает, что в формуле (2.7) следует рассматривать только вершины i, связанные ребром $j - i$ с вершиной j, индекс которых M_i в данный момент меньше предыдущего значения индекса M_j.

Пример 2.2. В неориентированном графе (рис 2.4) определить расстояния от вершины 3 до других вершин графа. Указать кратчайшую цепь из вершины 3 в вершину 2 и цепь **5-3**.

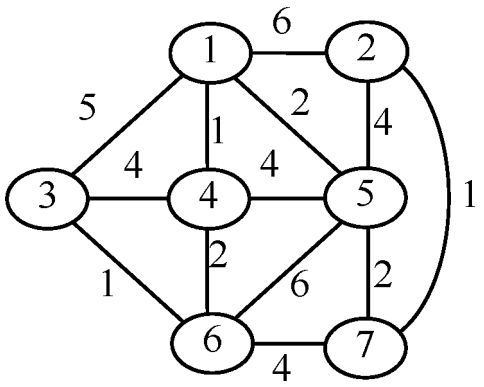


Рис. 2.4. Исходный неориентированный граф

Операции метода оформим в виде табл. 2.1, причем вычисление новых индексов в каждой строке будем производить по порядку столбцов.

Таблица 2.1. Расстояния от вершины 3

№ верш. / № шага	3	1	4	6	2	5	7
1	0	5(3)	4(3)	1(3)	11(1)	7(6)	5(6)
2	0	5(3)	3(6)	1(3)	6(7)	7(6)	5(6)
3	0	4(4)	3(6)	1(3)	6(7)	6(1)	5(6)
4	0	4(4)	3(6)	1(3)	6(7)	6(1)	5(6)

Индексы в строках 3 и 4 полностью совпадают, следовательно, индексацию можно остановить. Последняя строка представляет собой расстояния между вершиной 3 и остальными вершинами графа. Имеем

$$d(3-1)=d(1-3)=4;\ d(3-4)=d(4-3)=3;$$

$$d(3-6)=d(6-3)=1;\ d(3-2)=d(2-3)=6;$$

$$d(3-5)=d(5-3)=6;\ d(3-7)=d(7-3)=5.$$

Цепь кратчайшей длины из 3 в 2 имеет вид **3-6-7-2**; из 5 в 3: **5-1-4-6-3**.

Пример 2.3. В ориентированном графе (рис. 2.5) определить расстояния от вершины 1 до всех других и от всех вершин до вершины 1. Указать кратчайшие цепи 1-5 и 5-1.

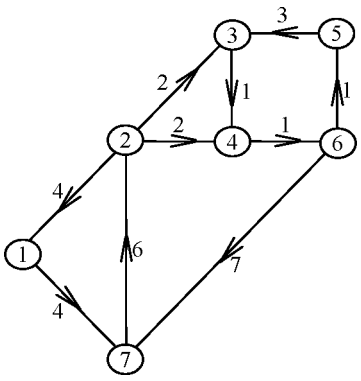


Рис. 2.5. Исходный ориентированный граф

Определим расстояния от вершины 1 до всех других вершин.

Таблица 2.2. Расстояния от вершины 1

№ верш. / № шага	1	2	3	4	5	6	7
1	0	-	-	-	-	-	4(1)
2	0	10(7)	12(2)	12(2)	-	13(4)	4(1)
3	0	10(7)	12(2)	12(2)	14(6)	13(4)	4(1)
4	0	10(7)	12(2)	12(2)	14(6)	13(4)	4(1)

Расстояния $1-j$ - в последней строке таблицы. $d(1-5)=14$. Кратчайшая цепь: **1-7-2-4-6-5**.

Теперь определим расстояния от всех вершин до вершины 1.

Таблица 2.3. Расстояния до вершины 1

№ верш. / № шага	1	2	3	4	5	6	7
1	0	4(1)	-	-	-	-	10(2)
2	0	4(1)	-	-	-	17(7)	10(2)
3	0	4(1)	-	18(6)	-	17(7)	10(2)
4	0	4(1)	19(4)	18(6)	22(3)	17(7)	10(2)
5	0	4(1)	19(4)	18(6)	22(3)	17(7)	10(2)

$d(5-1)=22$. Кратчайшая цепь: **5-3-4-6-7-2-1**.

3. ПОСТРОЕНИЕ ЭЙЛЕРОВОЙ ЦЕПИ В ГРАФЕ. АЛГОРИТМ ФЛЕРИ

Цепь, проходящая через каждое ребро графа G, называется эйлеровой цепью. Замкнутая эйлерова цепь называется эйлеровым циклом.

Связный граф, в котором существует эйлеров цикл, называется эйлеровым. Если в связном графе существует эйлерова цепь, но не существует эйлерова цикла, то он называется полуэйлеровым.

Теорема 1. Связный неориентированный граф является эйлеровым тогда и только тогда, когда каждая вершина его имеет четную степень.

Теорема 2. Связный неориентированный граф является полуэйлеровым тогда и только тогда, когда в нем ровно две вершины с нечетными степенями. Одна из этих вершин является началом эйлеровой цепи, а другая - концом (или наоборот).

Теорема 3. Связный ориентированный граф является эйлеровым тогда и только тогда, когда для каждой его вершины положительная степень равна отрицательной.

Теорема 4. Связный ориентированный граф является полуэйлеровым тогда и только тогда, когда он удовлетворяет следующим условиям.

1. В одной его вершине $\rho^+(k) - \rho^-(k) = 1$ (эта вершина - начало эйлеровой цепи).

2. В другой его вершине $\rho^-(m) - \rho^+(m) = 1$ (эта вершина является концом эйлеровой цепи).

3. Для остальных вершин $\rho^+(i) = \rho^-(i)$.

Алгоритм Флери, предназначенный для построения эйлеровой цепи в графе, имеет следующий вид:

1. Проверить возможность построения цепи: необходимо проверить, удовлетворяет ли граф теоремам 1-4.

2. Выбрать начало и конец цепи. В эйлеровом графе в качестве начала можно принять произвольную вершину, она же будет и концом цепи. В полуэйлеровом неориентированном графе в качестве начала выбирается одна из двух точек с нечетной степенью, другая будет концом маршрута. В ориентированном полуэйлеровом графе начало и конец цепи определяются однозначно (теорема 4).

3. Находясь в некоторой вершине, выбрать очередное ребро цепи, удовлетворяя следующим правилам:

- если есть другие возможности, нельзя выбирать ребро, являющееся мостом подграфа, образованного из исходного графа удалением (зачеркиванием) уже пройденных ребер;
- если есть другие возможности, нельзя выбирать ребро, ведущее в вершину, являющуюся концом маршрута.

4. Проходим выбранное ребро, заносим его в список и удаляем из графа (вычеркиваем).

5. Если пройдены не все ребра, возвращаемся к п.3.

Пример 3.1. Построить эйлерову цепь в графе (рис. 3.1).

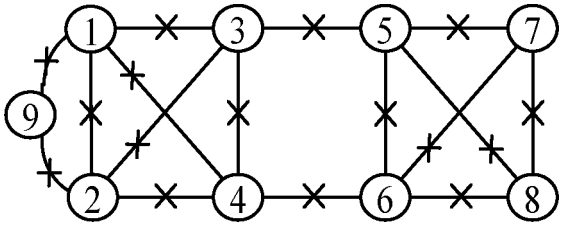


Рис. 3.1. Построение эйлеровой цепи

В этом графе степени всех вершин кроме 7 и 8 четны. Граф - полуэйлеров. Выберем в качестве начала маршрута вершину 7. Тогда концом маршрута будет вершина 8. После применения алгоритма получим эйлерову цепь **7-5-6-7-8-5-3-4-2-1-9-2-3-1-4-6-8.**

4. ПОСТРОЕНИЕ ДВОЙНОГО ЭЙЛЕРОВА ЦИКЛА В НЕОРИЕНТИРОВАННОМ ГРАФЕ. АЛГОРИТМ ТЭРРИ

Двойным эйлеровым циклом в неориентированном связном графе G называется замкнутый маршрут, проходящий по всем ребрам G дважды, причем один раз – в прямом, а другой – в противоположном направлении.

Если трансформировать каждое ребро неориентированного графа G в пару противоположных дуг, то получится ориентированный граф G_1, удовлетворяющий условиям теоремы 3 предыдущего раздела. Следовательно, в G_1 можно построить эйлеров цикл, который и будет являться двойным эйлеровым циклом исходного графа G. Вывод: *двойной эйлеров цикл можно построить в любом связном неориентированном графе*.

Для построения такого цикла применяется **алгоритм Тэрри**.

1. Начинаем движение в произвольной вершине графа.
2. Находясь в произвольной вершине, выбираем направление дальнейшего движения, исходя из двух правил запрета:
   - нельзя проходить по ребру дважды в одном направлении;
   - если есть другие возможности, нельзя выбирать ребро, которое привело в данную вершину в первый раз.
3. Проходя выбранное ребро, фиксируем (например, стрелкой) направление, по которому движемся, и заносим его в список.
4. Попав в вершину первый раз, отмечаем (например, крестом) ребро, по которому пришли.
5. Если пройдены дважды не все ребра, возвращаемся к п.2.

Практически это означает, что построение будет продолжаться до тех пор, пока имеется возможность движения, и закончится в начальной вершине.

Пример 4.1. Построить двойной эйлеров цикл, начиная с вершины 1, в графе G (рис. 4.1).

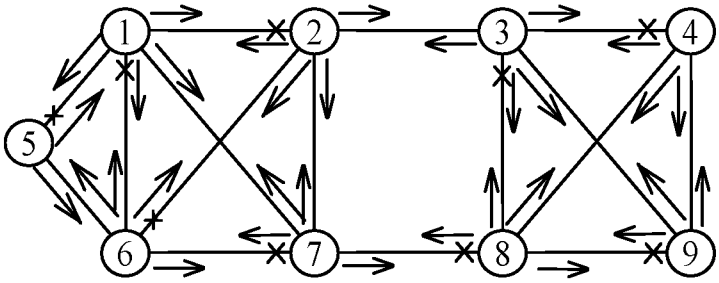


Рис. 4.1. Построение двойного эйлерова цикла

Искомый цикл имеет вид: **1-2-6-1-5-6-7-1-7-8-3-2-7-2-3-4-8-9-4-9-3-9-8-4-3-8-7-6-5-1-6-2-1**.

5. ПОСТРОЕНИЕ ОСТОВА ГРАФА НАИМЕНЬШЕГО ВЕСА. АЛГОРИТМ КРАСКАЛА

Рассматривается связный неориентированный граф G, причем каждому ребру $i-j$ приписан вес μ_{ij} (его можно понимать как длину, время, стоимость и т.д. дороги из пункта i в пункт j). Нужно выбрать из всех ребер графа такую совокупность, чтобы по этим ребрам можно было пройти из любого пункта в любой другой и чтобы общий вес этих ребер был минимальным.

Граф, не содержащий циклов, называется лесом. Связный лес называется деревом. Легко понять, что деревья – компоненты связности леса.

Деревом графа G называется его связный подграф без циклов. Дерево T связного графа G, включающее все его вершины, называется остовом графа G или его каркасом.

Если в связном графе N вершин, то остов содержит ровно $N-1$ ребро.

В связи с данными определениями описанная выше задача представляет задачу построения остова наименьшего веса (встречаются также следующие формулировки: *построение каркаса наименьшего веса, связного суграфа наименьшего веса*).

Для ее решения применяется **алгоритм Краскала**.

1. Выбираем в графе самое короткое ребро (ребро с наименьшим весом).
2. Если часть ребер уже выбрана, то из оставшихся выбираем то, которое не образует с уже выбранными ребрами цикла и имеет среди всех таких ребер наименьший вес.
3. Построение заканчивается тогда, когда добавление любого из оставшихся ребер приводит к образованию цикла.

Выбираемые ребра и их вес удобно заносить в таблицу, с помощью которой после окончания построения легко определяется вес остова $M(T)$.

Пример 5.1. Построить остов наименьшего веса для графа, изображенного на рис. 5.1.

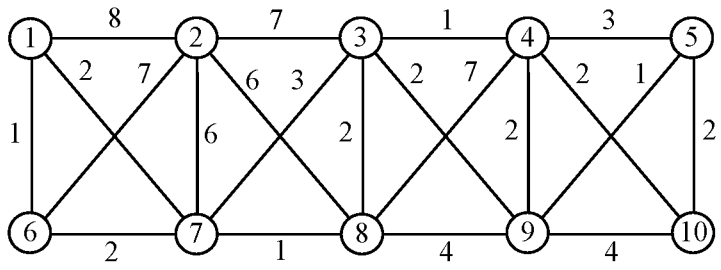


Рис. 5.1. Исходный граф

Порядок выбора ребер представлен в табл. 5.1, каркас наименьшего веса изображен на рис. 5.2.

Таблица 5.1. Построение остова наименьшего веса

№ п/п	1	2	3	4	5	6	7	8	9
Ребро	1-6	7-8	3-4	5-9	1-7	3-9	3-8	4-10	2-7
Вес	1	1	1	1	2	2	2	2	6

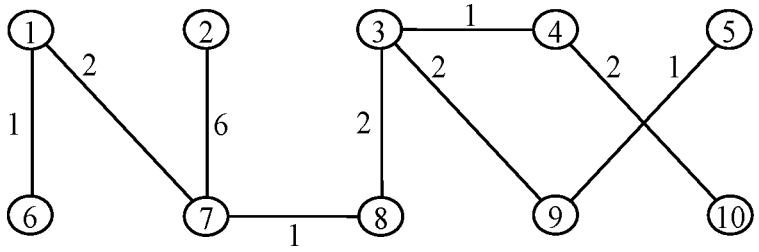


Рис. 5.2. Остов наименьшего веса

Вес каркаса $M(T)$ равен $1+1+1+1+2+2+2+2+6=18$.

Замечания.

1. В графе может быть несколько каркасов одного и того же минимального веса. Например, в данном случае можно было выбрать вместо ребра 4-10 ребро 5-10.

2. Для удобства построения можно после выбора каждого очередного ребра удалять ребра, образующие цикл с ранее выбранными ребрами. Это упрощает процесс выбора в сложных графах.

3. Если рассматривается граф G с k компонентами связности и m вершинами, то алгоритм Краскала позволяет построить остовный лес этого графа, содержащий $m-k$ ребер.

6. СЕТЕВОЕ ПЛАНИРОВАНИЕ

6.1 Построение сетевого графика

Пусть имеется план реализации некоторого проекта, т.е. имеется перечень операций (работ) проекта, и для каждой работы известны время, необходимое для ее выполнения, и перечень предшествующих операций (тех работ, которые должны быть завершены к началу данной). Представим план в виде таблицы.

Таблица 6.1. План реализации проекта

Операция	Предшествующая операция	Время выполнения t_{ij}	Дуга $i-j$	t_{ij}^{PO}	$t_{ij}^{ПН}$
a_1	-	2	1-2	2	0
a_2	-	6	1-3	6	0
a_3	a_1	4	2-3	6	2
a_4	a_1	5	2-4	7	7
a_5	a_2, a_3	7	3-5	13	6
a_6	a_4	1	4-5	8	12

В первом столбце таблицы названия операций (например, возведение стены) заменены символами a_i. Три последних столбца заполняются в процессе решения задачи сетевого планирования и объединены в одну таблицу с исходными данными исключительно из соображений удобства. Время выполнения каждой операции может измеряться различными единицами, важно лишь, чтобы выбранная единица измерения была одной и той же для всех операций.

Операции a_1, a_2 в табл. 6.1 не имеют предшествующих, поэтому называются начальными операциями. Операции a_5 и a_6 называются завершающими, т.к. не предшествуют ни одной другой операции проекта.

Сетевым графиком называется ориентированный граф, представляющий взаимосвязь отдельных операций проекта. С другой стороны, ориентированный граф может рассматриваться в качестве сетевого графика, если в нем:

- отсутствуют циклы и кратные дуги;
- ровно один источник и ровно один сток;
- каждой дуге $i-j$ сопоставлено число t_{ij} (длина дуги), равное продолжительности работы, образом которой является дуга.

Для построения сетевого графика произвольного проекта удобно следовать следующим правилам:

- каждая операция изображается в виде отдельной дуги;
- начала дуг начальных операций объединяются в одной вершине (источнике графа);
- концы дуг завершающих операций объединяют в одной вершине (стоке графа);
- связи между операциями изображаются пунктирными дугами, идущими из конца предшествующей операции в начало последующей.

В рассматриваемом случае можно прийти к следующей картине:

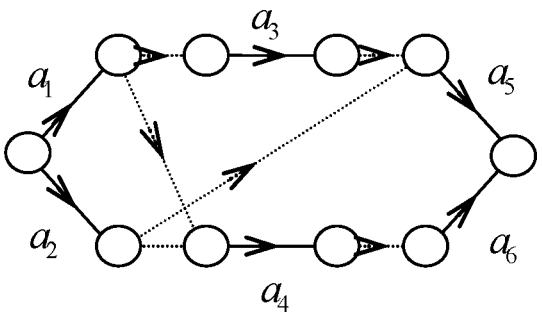


Рис 6.1. Исходный сетевой график

Вообще говоря, можно рассматривать дугу-связь как фиктивную операцию со временем выполнения, равным нулю, и дальнейшие процедуры сетевого планирования применять непосредственно к графу на

рис. 6.1. Однако для уменьшения объема вычислений желательно упростить граф. Для этого везде, где возможно, выбрасывают дуги-связи, объединяя конец предшествующей и начало последующей операции в одну вершину. В процессе упрощения следует избегать появления кратных дуг, поэтому в отдельных случаях приходится оставлять некоторые дуги-связи. Кроме того, после упрощения графа необходимо проверить, чтобы не была нарушена связь операций, заданная планом проекта.

В рассматриваемом случае упрощенный граф приобретет вид, представленный на рис. 6.2.

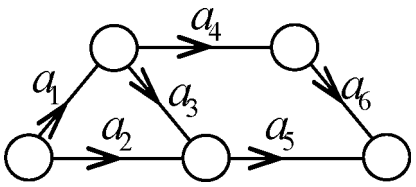


Рис. 6.2. Упрощенный сетевой график

Перенумеруем вершины графа так, чтобы для любой дуги номер начала был меньше номера конца. Такая нумерация называется правильной. Получить правильную нумерацию можно с помощью следующего алгоритма:

1. Источнику присваивается номер 1.
2. Вычеркиваются все дуги, выходящие из нумерованных вершин.
3. Нумеруются получившиеся после удаления дуг новые источники. Если источник один, то он получает следующий по порядку номер. Если источников несколько, то им присваивают возрастающие номера в произвольном порядке.
4. Пункты 2 и 3 повторяются до тех пор, пока не будут пронумерованы все вершины графа. Очевидно, что сток получит наибольший номер.

После нумерации в таблицу заносят дуги, соответствующие каждой операции проекта. На рисунке графа после этого можно заменить обозначение каждой операции длиной соответствующей дуги (временем выполнения операции). Окончательный вид сетевого графика рассматриваемого проекта представлен на рис. 6.3.

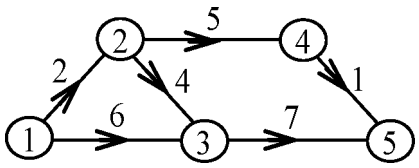


Рис 6.3. Окончательный вид сетевого графика

Замечания.

1. В дальнейшем будем считать, что нумерация вершин сетевого графика - правильная. Если в исходном графе это не так, то перед решением задачи сетевого планирования следует перенумеровать вершины.

2. Для удобства дальнейших рассуждений отождествим работу a_m и соответствующую ей дугу $i-j$, т.е. будем говорить о “работе $i-j$”, “операции $i-j$” и т.д.

6.2 Решение задачи сетевого планирования

Назовем событием i завершение всех работ, входящих в вершину i (начало всех работ, выходящих из вершины i).

Ранним сроком начала работы t_{ij}^{PH} называют наименьшее возможное время, когда работа $i-j$ может быть начата.

Если из вершины i выходят несколько работ, то ранние сроки начал этих работ совпадают, и этот общий срок называют ранним сроком наступления события i и обозначают T_i^P.

Если работа $i-j$ начата в момент времени t_{ij}^{PH}, то наименьшее возможное время ее окончания называется ранним сроком окончания работы и обозначается t_{ij}^{PO}.

Связь между величинами T_i^P, t_{ij}^{PH}, t_{ij}^{PO} определяется соотношениями

$$t_{ij}^{PH} = T_i^P; \qquad t_{ij}^{PO} = T_i^P + t_{ij}. \tag{6.1}$$

Назовем путем в сетевом графике простую цепь, связывающую источник (вершину 1) со стоком (вершиной N).

Критический путь – это путь наибольшей длины в сетевом графике (таких путей может быть несколько). Длина критического пути называется критическим временем и обозначается T_{KP}. Проект не может быть завершен за время, меньшее T_{KP}. Очевидно, что T_{KP} совпадает с ранним сроком наступления конечного события N.

Поздним сроком окончания работы $t_{ij}^{ПО}$ называют наибольшее допустимое время окончания работы $i-j$ (завершение работы $i-j$ в более поздний срок отодвигает срок реализации проекта в целом).

Если в вершину j входят несколько работ, то поздние сроки их окончания совпадают, и этот общий срок называется поздним сроком наступления события j и обозначается $T_j^{П}$.

Если работа $i-j$ окончена в момент времени $t_{ij}^{ПО}$, то наибольшее допустимое время ее начала называется поздним сроком начала работы и обозначается $t_{ij}^{ПН}$.

Связь между величинами $T_j^{П}$, $t_{ij}^{ПО}$, $t_{ij}^{ПН}$ определяется формулами

$$t_{ij}^{n\,м} = T_j^n; \qquad t_{ij}^{n\,Н} = T_j^n - t_{ij}. \tag{6.2}$$

Любая работа $i-j$ по определению не может быть начата ранее момента времени T_i^P и закончена позднее $T_j^{П}$. По плану эта работа может быть выполнена за время t_{ij}.

Максимально допустимое время, на которое можно увеличить продолжительность работы $i-j$ (по сравнению с планом) так, что это не вызовет задержки выполнения проекта в целом, называется полным резервом времени работы $i-j$ и обозначается R_{ij}.

Полный резерв определяется по формуле

$$R_{ij} = T_j^n - T_i^P - t_{ij}. \tag{6.3}$$

Если на некоторой работе $i-j$ использовать ее полный резерв, то время выполнения других работ, принадлежащих любому пути, содержащему дугу $i-j$, уже нельзя будет увеличивать без удлинения срока выполнения проекта в целом. Максимально допустимое время, на которое можно увеличить продолжительность работы $i-j$ так, что это не повлияет на резервы времени остальных работ проекта, называется свободным резервом времени работы $i-j$ и обозначается r_{ij}

Этот резерв определяется по формуле

$$r_{ij} = T_j^P - T_i^P - t_{ij}. \qquad (6.4)$$

Задача сетевого планирования включает вычисление величин t_{ij}^{PH}, t_{ij}^{PO}, $t_{ij}^{ПН}$, $t_{ij}^{ПО}$, R_{ij}, r_{ij} для всех работ, вычисление величин T_i^P, $T_j^П$ для всех вершин графа, определение критического времени T_{KP} и критических путей проекта.

Помимо формул (6.1)-(6.4) решение задачи требует использования следующих алгоритмов.

6.2.1 Алгоритм вычисления ранних сроков наступления событий T_i^P и критического времени

1. Полагают $T_1^P = 0$.

2. Просматривают остальные вершины в порядке возрастания номеров $(i = 2, 3...N)$ и вычисляют сроки

$$T_i^P = \max_{k\,(k<i)} \left[T_k^P + t_{ki} \right]. \qquad (6.5)$$

В формуле (6.5) участвуют только те вершины k, для которых существует дуга $k-i$. Номера всех вершин, для которых достигается указанный максимум, фиксируется для каждой вершины i. Они понадобятся для построения критических путей.

3. После определения T_N^P вычисляется T_{KP} по формуле

$$T_{KP} = T_N^P. \qquad (6.6)$$

6.2.2 Построение критического пути

Обозначим через $M(i)$ номер вершины, для которой достигается максимум в формуле (6.5) для вершины i. Алгоритм построения критического пути будет иметь следующий вид:

1. Фиксируется конец пути $i_0 = N$.

2. Вычисляется номер очередной вершины

$$i_k = M(i_{k-1}), \qquad (k = 1,2.....).$$

3. Пункт 2 повторяется до тех пор, пока на некотором шаге будет иметь место равенство $i_m = 1$.

4. Искомый критический путь будет представлять собой последовательность $1 - i_{m-1} - i_{m-2} - ... - i_1 - N$.

Замечание. В сетевом планировании (в отличие от задачи определения расстояний) строятся все имеющиеся критические пути.

6.2.3 Алгоритм вычисления поздних сроков наступления событий $T_j^{П}$

1. Полагают $T_j^n = T \geq T_{KP}$ (обычно считают $T = T_{KP}$).

2. Просматривают остальные вершины в порядке убывания номеров ($j = N-1, N-2, ..., 1$) и вычисляют значения

$$T_j^n = \min_{k\,(k>j)} (T_k^n - t_{jk}). \qquad (6.7)$$

В формуле (6.7) участвуют только те вершины k, для которых существует дуга $j - k$.

6.2.4 Общий план решения

Общий план решения задачи сетевого планирования имеет следующий вид:

1. Расчет ранних сроков наступления событий T_i^P и T_{KP}.

2. Расчет ранних сроков окончания работ t_{ij}^{PO}.

3. Построение критических путей.

4. Вычисление поздних сроков наступления событий $T_j^{П}$.

5. Вычисление поздних сроков начала работ $t_{ij}^{ПН}$.

6. Определение резервов времени работ R_{ij}, r_{ij}.

Замечание. Значения $t_{ij}^{РО}$ и $t_{ij}^{ПН}$ обычно вносят в таблицу вместе с исходными данными. Результаты других вычислений удобно записывать непосредственно на сетевом графике по следующему образцу (рис. 6.4).

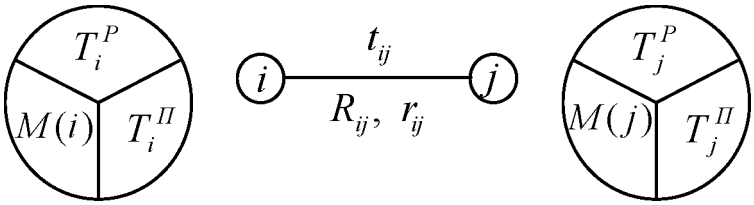


Рис.6.4. Образец оформления вычислений

Если номеров $M(i)$ несколько, то все они записываются в левой трети круга (через запятую). Ясно, что если во всех вершинах $M(i)$ - единственный, то критический путь также будет единственным.

Результаты решения рассматриваемой задачи представлены на рис. 6.5 и в табл. 6.1.

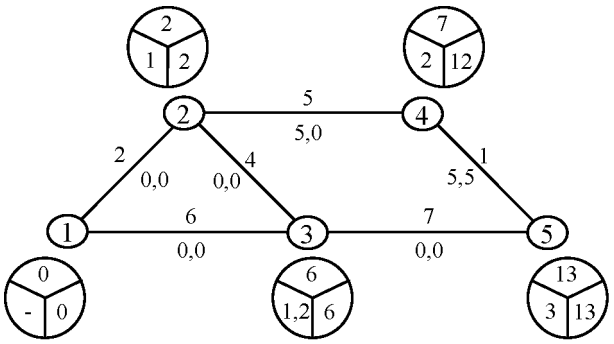


Рис 6.5. Задача сетевого планирования

7. ТРАНСПОРТНЫЕ СЕТИ И ПОТОКИ. АЛГОРИТМ ФОРДА - ФАЛКЕРСОНА

7.1. Основные понятия

Транспортной сетью называется ориентированный граф, в котором:

- отсутствуют петли;
- имеется ровно один источник и ровно один сток;
- каждой дуге $i-j$ поставлено в соответствие положительное число C_{ij}, называемое пропускной способностью дуги.

Физическим аналогом транспортной сети может служить система трубопроводов разной пропускной способности, по которой течет несжимаемая жидкость, причем жидкость подводится к источнику, а отводится – от стока.

Поток по дуге φ_{ij} - это реальное количество вещества, пропускаемое по дуге $i-j$ в единицу времени. Если $\varphi_{ij}=C_{ij}$, то дуга называется насыщенной.

Потоком на транспортной сети называется функция φ, определенная на множестве дуг графа и удовлетворяющая следующим условиям:

- для любой вершины, кроме источника и стока, сумма потоков по дугам, входящим в нее, равна сумме потоков по дугам, выходящим из вершины;
- поток по любой дуге не превосходит ее пропускной способности, т.е.

$$0 \le \varphi_{ij} \le C_{ij}. \tag{7.1}$$

Из условия (7.1) следует, что сумма Φ потоков по дугам, выходящим из источника, равна сумме потоков по всем дугам, входящим в сток. Величина Φ называется величиной потока на транспортной сети.

Назовем путем в транспортной сети любую цепь, связывающую источник со стоком.

Поток называется полным, если любой путь в сети содержит хотя бы одну насыщенную дугу.

Поток называется неполным, если существует хотя бы один путь, состоящий только из ненасыщенных дуг. Самый простой вариант неполного потока - так называемый нулевой поток, в котором для любой дуги $\varphi_{ij} = 0$ (жидкость вообще не течет).

Особый интерес представляет задача нахождения потока наибольшей величины $\Phi_{\max}$.

На рис. 7.1 изображены транспортная сеть и четыре потока на ней.

Замечания.

- Поток наибольшей величины – всегда полный, но не всякий полный поток является потоком наибольшей величины.
- На транспортной сети может быть несколько потоков наибольшей величины, однако для всяких таких потоков $\Phi_{\max}$ – величина постоянная, определяемая только графом сети.

Обозначим для определенности источник через i_S, а сток через i_F. Разобьем вершины транспортной сети на два подмножества S и F, причем к S отнесем источник и любые другие вершины кроме стока, а к F - вершины сети, не вошедшие в S.

Разрезом транспортной сети назовем набор дуг, начинающихся в вершинах множества S и заканчивающихся в вершинах множества F. Например, если на рис. 7.1 к S отнести вершины 1, 3 и 5, то в разрез войдут дуги 1-2, 1-4, 3-4, 5-4, 5-6.

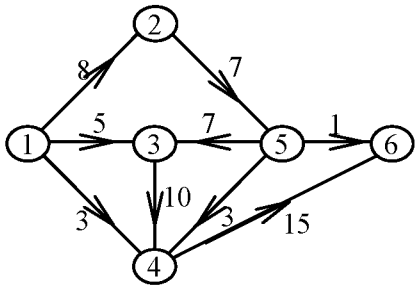
Назовем пропускной способностью разреза сумму пропускных способностей входящих в него дуг.

Разрезы, пропускная способность которых минимальна, называются критическими разрезами.

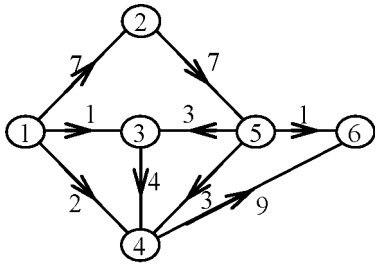
Теорема Форда-Фалкерсона

Наибольшая величина потока на транспортной сети равна пропускной способности критического разреза.

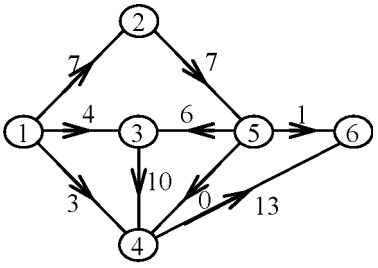
Отыскание максимального потока проводится с помощью алгоритма, предложенного Фордом и Фалкерсоном. Он состоит из двух частей: алгоритма построения полного потока и алгоритма его максимизации.



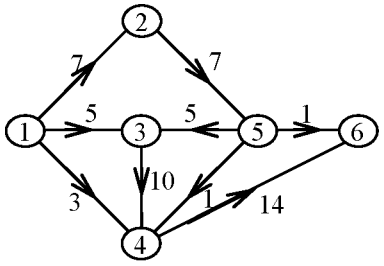
Транспортная сеть



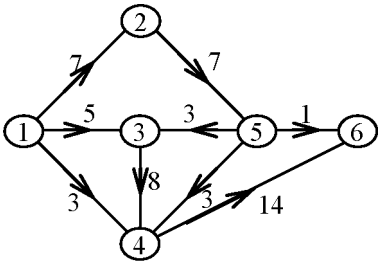
Неполный поток $\Phi = 10$



Полный поток $\Phi = 14$



Полный поток $\Phi_{\max} = 15$



Полный поток $\Phi_{\max} = 15$

Рис 7.1. Примеры потоков на транспортной цепи

7.2. Алгоритм построения полного потока

1. Составить список всех путей транспортной сети.

2. Построить начальный поток $\varphi_{\text{нач}}$ (в качестве начального удобно принять нулевой поток φ_O).
3. Просмотреть очередной путь. Для всех его дуг по очереди вычисляется величина

$$\delta_{ij} = C_{ij} - \varphi_{ij}. \tag{7.2}$$

Здесь через φ_{ij} обозначен поток по дуге $i-j$, вычисленный при рассмотрении предыдущих путей, включающих эту дугу.

4. Пусть при просмотре пути обнаружилась дуга, для которой $\delta_{ij} = 0$. В этом случае можно перейти к просмотру очередного пути, не вычисляя δ_{ij} для остальных дуг данного пути.
5. Пусть для данного пути все $\delta_{ij} > 0$. В этом случае проводятся следующие вычисления для дуг пути:
   - определяется величина поправки

$$\Delta = \min \delta_{ij}; \tag{7.3}$$

   - вычисляется новый поток по дугам просматриваемого пути

$$\varphi_{ij}^{(\text{нов})} = \varphi_{ij} + \Delta. \tag{7.4}$$

6. Пункты 3,4,5 повторяются до тех пор, пока не будет просмотрен последний путь списка. Окончательные значения φ_{ij} определяют полный поток на транспортной сети.

Вычисления удобно оформлять в виде таблицы, причем значения δ_{ij} для дуг пути обычно указываются в скобках, а значения φ_{ij} - вне скобок. Тогда достигнутое в любой момент значение φ_{ij} для дуги $i-j$ представляет собой последнее число без скобок в строке, соответствующей дуге $i-j$.

В табл. 7.1 представлен процесс вычисления полного потока для транспортной сети, изображенной на рис. 7.1.

Таблица 7.1. Определение полного потока

Дуги $i-j$	C_{ij}	φ_{ij}^{O}	Пути					$\varphi_{ij}^{П}$
			1-2-5-6	1-2-5-3--4-6	1-2-5-4--6	1-3-4-6	1-4-6	
1-2	8	0	(8)1	(7)7	(1)			7
1-3	5	0				(5)4		4
1-4	3	0					(3)3	3
2-5	7	0	(7)1	(6)7	(0)			7
3-4	10	0		(10)6		(4)10		10
4-6	15	0		(15)6		(9)10	(5)13	13
5-3	7	0		(7)6				6
5-4	3	0						0
5-6	1	0	(1)1					1
Δ			1	6		4	3	

Величина потока

$$\Phi = \varphi_{12} + \varphi_{13} + \varphi_{14} = \varphi_{56} + \varphi_{46} = 1 + 13 = 14.$$

Замечание. Для того чтобы упростить процесс вычислений во второй части алгоритма, желательно в список вносить сначала простые пути и лишь затем – более сложные. В данном случае это правило нарушено, чтобы лучше проиллюстрировать вторую часть алгоритма.

7.3. Алгоритм получения потока наибольшей величины (метод индексации)

1. Пометим источник i_S индексом (меткой) 0.
2. Пусть на некотором шаге присвоены метки от 0 до k включительно.

Индекс "$k+1$" присваивается всем ранее не помеченным вершинам, для которых выполняется одно из двух следующих условий:

- Существует ненасыщенная дуга $i-j$ ($\varphi_{ij} < C_{ij}$), связывающая вершину i с меткой "k" с данной вершиной j. Дуга помечается знаком "+".
- Существует дуга $j-i$ с ненулевым потоком $\varphi_{ji} > 0$, связывающая данную (еще не помеченную) вершину j с вершиной i, помеченной индексом "k". Дуга помечается знаком "–".

3. Процесс индексации продолжается до автоматической остановки. При этом возможны два случая.

Случай I: *процесс индексации не дошел до стока (сток i_F остался непомеченным).* В этом случае имеющийся полный поток является наибольшим. Кроме того, включив во множество S все помеченные вершины, получим критический разрез. Значения $\Phi_{\max}$ можно вычислить как суммируя пропускные способности дуг разреза, так и по изложенному выше правилу (потоки по дугам, выходящим из источника и входящим в сток). Алгоритм на этом заканчивается.

Случай II: *процесс индексации дошел до стока (сток i_F оказался помеченным индексом "N").* В этом случае поток не является максимальным и его можно увеличить. Для этого проводятся следующие операции.

4. Выпишем последовательность вершин с последовательно возрастающим метками от "0" до "N", все дуги которой помечены только знаком "+" либо знаком "–". Принцип построения тот же, что и в волновом методе, т.е. в обратном порядке, начиная с N.
5. Просматривая дуги последовательности, помеченные знаком "–", вычисляем по ним значение
$$\varepsilon_1 = \min \varphi_{ij}.$$
6. Просматривая дуги последовательности, помеченные знаком "+", вычисляем по ним значение
$$\varepsilon_2 = \min(C_{ij} - \varphi_{ij}).$$
7. Определяется величина поправки

$$\varepsilon = \min(\varepsilon_1, \varepsilon_2).$$

8. Изменяется поток на дугах последовательности:

- для дуг, помеченных знаком "+" поток увеличивается на величину ε:

$$\varphi_{ij}^{(\text{нов})} = \varphi_{ij} + \varepsilon;$$

- для дуг, помеченных знаком "−" поток уменьшается на величину ε:

$$\varphi_{ij}^{(\text{нов})} = \varphi_{ij} - \varepsilon.$$

Величина потока в сети при этом увеличивается на то же значение ε.

9. После изменения потока повторяем процесс индексации, т.е. возвращаемся к пункту 1.

Определим поток наибольшей величины в рассматриваемой транспортной сети. Для удобства на рисунке графа укажем для каждой дуги как поток, так и пропускную способность (в скобках). Результаты первой индексации представлены на рис. 7.2.

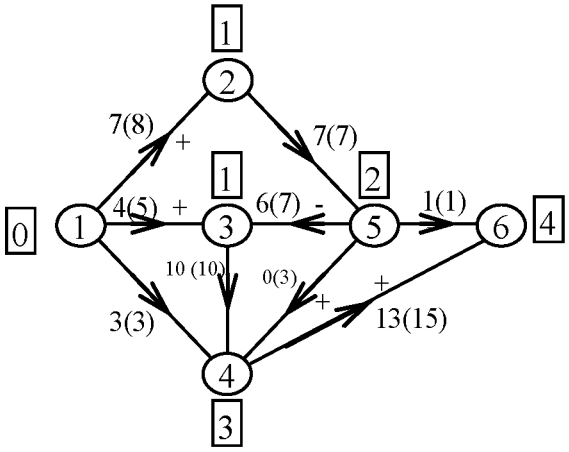


Рис. 7.2. Результаты индексации

Процесс индексации дошел до стока. Выпишем модифицируемую последовательность (рис 7.3).

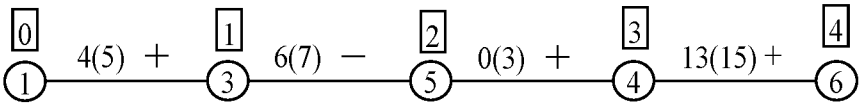


Рис. 7.3. Модифицируемая последовательность

$$\varepsilon_1 = 6;$$
$$\varepsilon_2 = \min\left[5-4;\ 3-0;\ 15-13\right] = 1;$$
$$\varepsilon = \min\left(6;1\right) = 1.$$

Новые значения потоков:

$$\varphi_{13} = 4+1 = 5;$$
$$\varphi_{53} = 6-1 = 5;$$
$$\varphi_{54} = 0+1 = 1;$$
$$\varphi_{46} = 13+1 = 14.$$

Новый поток и результаты повторной индексации представлены на рис. 7.4.

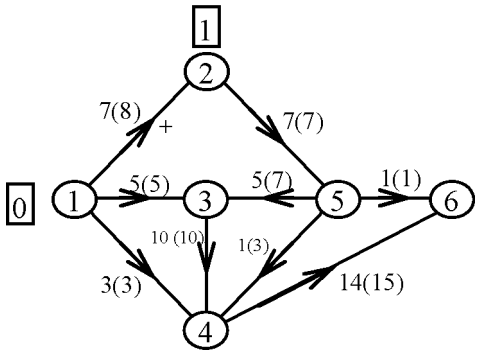


Рис. 7.4. Максимальный поток на транспортной сети

Процесс индексации до стока не дошел, следовательно, поток является наибольшим: $\Phi_{\max} = \varphi_{56} + \varphi_{46} = 1+14 = 15$ (ед.). В критический разрез входят ребра **1-3; 1-4;2-5**. Легко проверить, что $C_{13} + C_{14} + C_{25} = 5+3+7 = 15 = \Phi_{\max}$.

На этом решение задачи заканчивается.

ЗАКЛЮЧЕНИЕ

Настоящее издание ни в коей мере не претендует на то, чтобы заменить фундаментальные труды по теории графов.

Авторы сочтут свою задачу выполненной, если читатели получат некоторое представление об алгоритмах теории графов и научатся решать задачи, аналогичные рассмотренным в пособии.

Авторы также надеются, что учебное пособие послужит читателю базой для дальнейшего изучения теории графов и окажет помощь в решении практических задач.

СПИСОК ЛИТЕРАТУРЫ

1. Лекции по дискретной математике [Текст] / *Ю.В. Капитонова, С.Л. Кривой* [и др.]. – СПб.: БХВ-Петербург, 2004. – 624 с.
2. *Оре, О.* Теория графов [Текст] / *О. Оре*. – М.: Мир, 1980. – 336 с.
3. Теория графов [Текст] / [В.В. Белов и др.]. – М.: Высшая школа, 1976. – 392с.
4. *Уилсон, Р.* Введение в теорию графов [Текст] / *Р. Уилсон*. – М.: Мир, 1977. – 288 с.
5. Справочник по математике для экономистов [Текст] / *В.Е. Барбаумов, В.И. Ермаков* [и др.]. – М.: Высшая школа, 1987. – 336 с.
6. *Калугин, Н.А.* Основы теории графов [Текст]: учеб. пособие / *Н.А. Калугин, А.Н. Калугин*. – Самара: Изд-во СГАУ, 2003. – 40 с.

Учебное издание

Калугин Николай Александрович,
Калугин Александр Николаевич

ЭЛЕМЕНТЫ ТЕОРИИ ГРАФОВ

Учебное пособие

Редактор Н.С. Куприянова
Доверстка А.В. Ярославцева

Подписано в печать 01.07.2013 г. Формат 60x84 1/16.
Бумага офсетная. Печать офсетная. Печ. л. 2,75.
Тираж 100 экз. Заказ. Арт. – 23/2013.

Самарский государственный аэрокосмический университет.
443086, Самара, Московское шоссе, 34.

Изд-во Самарского государственного аэрокосмического университета.
443086 Самара, Московское шоссе, 34.