***Лабораторная работа***

***Алгоритмы на графах. Задачи о кратчайшем пути и максимальном потоке***

**Задание**

1. Каждый студент должен произвольно задать сеть с характеристиками, выбранными соответственно номеру в журнале.

Исходные данные:

G1=(6;12)

G2= (7;9)

Первый столбец − параметры графа G, второй − графа G. В паре (i,j) первое число − число вершин, второе число − число дуг.

G− сеть для задачи о максимальном потоке, G − для задачи о кратчайшем пути. Интервалы весов указывают границы изменения пропускных способностей и длин дуг.

1. Для построенной сети реализовать алгоритм Форда-Фалкерсона на языке программирования высокого уровня.
2. Для построенной сети реализовать программу на языке программирования высокого уровня, которая решает задачу о поиске кратчайшего пути в графе на основе алгоритма Дейкстры.
3. По результатам работы составить отчет

**Содержание отчета**

1. Граф, построенный на основе индивидуального задания.
2. Словесное описание и блок-схемы алгоритмов Форда-Фалкерсона и Дейкстры.
3. Коды программ, реализующие решение задачи.
4. Скриншоты с результатами работы программы.
5. Выводы по работе

**Краткие теоретические сведения**

*1 . Задача о максимальном потоке. Алгоритм Форда-Фалкерсона*

**Определения**

Орграф G=(V, E) называется **взвешенным**, если каждой его дуге приписано число , называемое весом дуги.

**Сетью** называется конечный связный взвешенный орграф G=(V, E) без петель, в котором выделены две вершины:

1. вершина , не имеющая входящих дуг и называемая **источником** или **началом сети**;
2. вершина , не имеющая выходящих дуг и называемая **стоком** или **концом сети**, здесь n − число вершин графа G**.**

Каждой дуге  орграфа поставлено в соответствие некоторое неотрицательное число , называемое **пропускной способностью** дуги (вес дуги). В дальнейшем предполагается, что  − целые числа.

**Потоком по сети** G=(V, E) из источника s в сток t называется неотрицательная функция f = , определенная на дугах сети и такая, что выполняется:

1)  для любой дуги ;

2)=

Числа  называются **дуговыми потоками,** а число r − **величиной потока**.

Первая группа неравенств означает, что дуговые потоки не могут превышать соответствующие пропускные способности дуг. Вторая группа неравенств описывает законы сохранения потока в сети и в вершинах: величина потока втекающего в сеть равна величине потока вытекающего из сети, а величина потока втекающего в любую вершину равна величине потока вытекающего из этой вершины.

Задача о максимальном потоке состоит в нахождении такого потока f = , при котором величина потока r максимальна.

Введем вспомогательные понятия. Пусть , а . **Разрезом** сети  называется множество дуг  таких, что  или . Разрез  называется **разделяющим** источник  от  стока, если .

**Пропускной способностью** или **величиной разреза**  называется величина .

Дуга  называется **насыщенной** потоком f, если 

**Теорема Форда-Фалкерсона.**

В любой сети величина максимального потока равна пропускной способности минимального разреза.

Из этой теоремы следует, что решив задачу о максимальном потоке мы одновременно решим и задачу о минимальном разрезе.

**2.Метод расстановки пометок нахождения максимального потока**

Алгоритм начинает работать с произвольного потока. В качестве начального потока можно взять, очевидно, нулевой поток, который будет последовательно увеличиваться до тех пор, пока это возможно. Алгоритм заключается в систематическом поиске путей, по которым можно увеличить поток. После того, как такой путь найден, увеличивают поток по сети.

Решать задачу будем методом расстановки пометок, используя две операции:

− расстановки пометок;

− изменения потока.

В процессе работы каждая вершина графа может находиться в одном из трех состояний:

1) **не помечена**;

2) **помечена, но не просмотрена**;

3) **помечена и просмотрена**.

В процессе работы алгоритма каждая вершина данной сети получает метку. Метка произвольной вершины j имеет вид:  или , где , а – натуральное число или бесконечность. Первая часть метки указывает на то, что можно увелить поток по дуге  на величину в случае «+», и уменьшить поток по дуге  − в случае «-». Помеченная вершина считается просмотренной, если все смежные с ней вершины обработаны, т.е. сделана попытка их пометить.

**Алгоритм Форда-Фалкерсона**

**Подготовительный этап.**

Выбираем произвольный поток. В качестве начального потока можно взять нулевой поток:  для любой дуги .

Помечаем источник s:  (эта метка означает, что мы пытаемся пропустить через сеть бесконечный по величине поток). Теперь источник помечен, но не просмотрен. Остальные вершины не помечены.

**Этап расстановки пометок.**

Выбираем произвольную помеченную и непросмотренную вершину i (например, вершину, имеющую минимальный номер) и пытаемся пометить все смежные с ней непомеченные вершины j:

все те вершины j, для которых , получают метку , где ; такие узлы j теперь помечены и непросмотрены;

все те вершины j, для которых , получают метку , где ; такие узлы j теперь помечены и непросмотрены.

После этой процедуры вершина i считается помеченной и просмотренной и больше не рассматривается на этом шаге даже в случае, если не все смежные с ней вершины удалось пометить.

Далее просматриваем следующую вершину, и так до тех пор, пока не пометим сток t или же пока нельзя будет больше пометить ни одной вершины, при этом сток останется не помеченным. Если сток окажется не помеченным, то процесс нахождения максимального потока в сети можно считать законченным, а если сток помечен, то переходим к следующему этапу.

Если максимальный поток найден, то все вершины, которые удалось пометить на этом этапе и вершина s образуют множество X и определяют минимальный разрез . величины . Отметим, что все дуги  разреза  такие,что , являются насыщенными, а по остальным дугам разреза «течет» нулевой поток.

**Этап изменения потока.**

Используя первую часть метки вершины, определяем путь, по которому мы пришли из вершины s в вершину t. Выделение этого пути начинаем с вершины t: если вершина t имеет метку , то вершина t помечена из вершины i и т.д. В результате мы получаем последовательность смежных вершин: . По всем дугам  этого пути , начальная вершина которых имеет пометку «+», увеличиваем поток на величину , а по всем остальным дугам этого пути , начальная вершина их имеет пометку «-», уменьшаем поток на эту же величину  «направление» дуги на этом пути совпадает с направлением пути. В результате − поток по сети увеличится на величину .

После изменения потока все метки вершин, кроме метки вершины s , удаляются и возвращаемся на этап расстановки пометок.

**Конец работы алгоритма.**

**3.Пример решения задачи о максимальном потоке и минимальном разрезе**

Рассмотрим конкретную задачу о нахождении максимального потока в сети.

Дана сеть G(V,E) (рис.1) с источником s и стоком t. Пропускные способности дуг указаны. Найти максимальный поток из s в t.

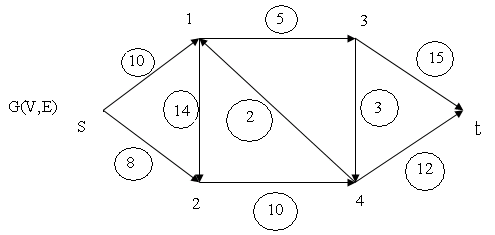


Рис.1

**Решение.**

Подготовительный этап.

М(s)=(s, +∞); все дуговые потоки нулевые −  для любой дуги .

Вершина s помечена и не просмотрена, остальные вершины не помечены.

Этап расстановки пометок

Рассматриваем вершину s:

М(1)=(,10); М(2)=(,8)

Вершина s помечена и просмотрена, вершины 1, 2 помечены и непросмотрены.

Рассматриваем вершину 1:

М(3)=(1, 5)

Вершина 1 помечена и просмотрена.

Рассматриваем вершину 2:

М(4)=(2, 8)

Вершина 2 помечена и просмотрена.

Рассматриваем вершину 3:

М(t)=(3, 5)

Помечена вершина t, переходим на следующий этап.

Этап изменения потока

=5

Путь, по которому мы пришли из вершины s в вершину t:

(,1,3, t)

Величина потока r = 5.

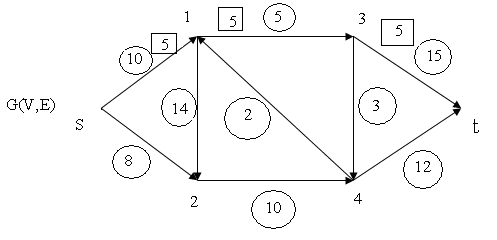


Рис.2

В прямоугольниках (рис.2) указаны дуговые потоки после этого этапа, все остальные дуговые потоки равны нулю.

Этап расстановки пометок

Рассматриваем вершину s:

М(1)=(,5); М(2)=(,8)

Вершина s помечена и просмотрена, вершины 1, 2 помечены и непросмотрены.

Рассматриваем вершину 1:

Вершину 3 из вершины 1 пометить нельзя.

Вершина 1 помечена и просмотрена.

Рассматриваем вершину 2:

М(4)=(2, 8)

Вершина 2 помечена и просмотрена.

Рассматриваем вершину 4:

М(t)=(4, 8)

Вершина 4 помечена и просмотрена. Пометили вершину t.

Этап изменения потока

=8

Путь, по которому мы пришли из вершины s в вершину t:

(,2,4, t)

Величина потока r = r+8=13.

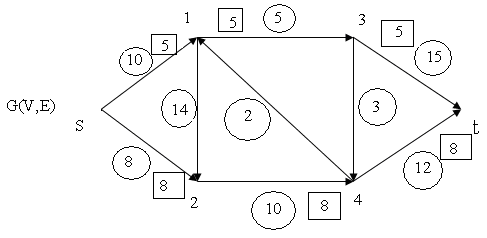


Рис.3

Дуговые потоки после этого этапа указаны на рис.3, все остальные дуговые потоки равны нулю.

Этап расстановки пометок

Рассматриваем вершину s:

М(1)=(,5)

Вершину 2 из вершины s пометить нельзя.

Вершина s помечена и просмотрена, вершина 1 помечена и непросмотрена.

Рассматриваем вершину 1.

М(2)=(,5)

Вершины 3 и 4 из вершины 1 пометить нельзя.

Вершина 1 помечена и просмотрена, вершина 2 помечена и непросмотрена.

Рассматриваем вершину 2.

М(4)=(2, 2)

Вершина 2 помечена и просмотрена, вершина 4 помечена и непросмотрена.

Рассматриваем вершину 4.

М(t)=(4, 2)

Вершину 3 из вершины 1 пометить нельзя.

Вершина 4 помечена и просмотрена. Пометили вершину t.

Этап изменения потока

=2

Путь, по которому мы пришли из вершины s в вершину t:

(, 1, 2,4, t)

Величина потока r = r+2=15.

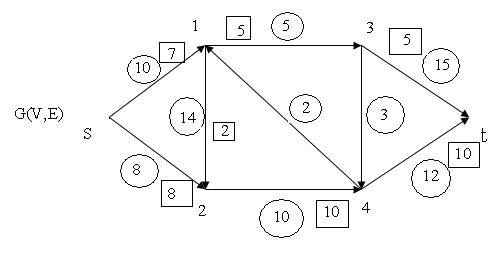


Рис.4

Дуговые потоки после этого этапа указаны на рис.4, все остальные дуговые потоки равны нулю.

Этап расстановки пометок

Рассматриваем вершину s:

М(1)=(,3)

Вершину 2 из вершины s пометить нельзя.

Вершина s помечена и просмотрена, вершина 1 помечена и непросмотрена.

Рассматриваем вершину 1.

М(2)=(, 3)

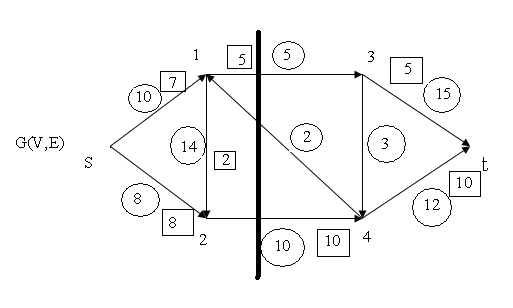
Вершины 3 и 4 из вершины 1 пометить нельзя.

Вершина 1 помечена и просмотрена, вершина 2 помечена и непросмотрена.

Рассматриваем вершину 2.

Из вершины 2 других вершин пометить не удается.

На этом этапе других вершин пометить не удалось, следовательно максимальный поток найден r=15, а минимальный разрез имеет следующий вид:



= 

**Задача о кратчайшем пути.**

Пусть задана **сеть** G =  с множеством вершин N, где , и множеством дуг U, т.е. задан ориентированный граф с n + 1 вершиной, в котором выделены две вершины – вход (нулевая вершина) и выход (вершина с номером n). Для каждой дуги (i; j) задано число , называемое длиной дуги. **Длиной пути** называется сумма длин дуг этого пути (если длины дуг не заданы, то длина пути определяется как число дуг пути, т.е. ). **Задача о кратчайшем пути** состоит в поиске минимального (кратчайшего) по длине пути из вершины с номером 0 до вершины с номером n.

В дальнейшем будем предполагать:1) сеть не имеет контуров; 2) из вершины с номером 0 можно попасть (по некоторому пути) в любую другую вершину сети, а из любой вершины сети можно попасть (по некоторому пути) в вершину с номером n; 3) нулевая вершина не имеет входящих дуг, а вершина n не имеет выходящих дуг.

Сеть G не имеет контуров, поэтому всегда можно пронумеровать вершины таким образом, что для любой дуги (i, j) имеет место соотношение: i<j. Такая нумерация вершин сети называется **правильной**.

Если сеть G имеет правильную нумерацию (вершин), то кратчайший путь можно найти алгоритмом 1. В процессе работы этого алгоритма каждая вершина получает **метку** − вершина i получает метку M(i) =(j,  ), где первая часть метки указывает номер вершины, из которой помечена вершина i, а величина  указывает длину кратчайшего пути из нулевой вершины в данную вершину.

**Алгоритм Дейкстры**

Шаг 0. Полагаем, что множество вершин Q − это пустое множество. Помечаем нулевую вершину: M(0) = (0,), где  = 0, т.е. M(0) = (0,0). Заносим нулевую вершину в множество Q. Все остальные вершины получают метку − M(i) =(0, ), т.е. = (это означает, что вершина i помечена из вершины 0 и, предположительно, находится от нее на бесконечном расстоянии).

Шаг 1. Для каждой вершины kQ вычисляем величину  (минимум берется по всем вершинам i таким, что iQ и в сети имеется дуга (i,k), и этот минимум достигается на вершине j). Среди всех таких вершин k выбираем ту, которая имеет минимальную величину , помечаем ее − M(k) =(j, ) и заносим в множество Q.

Подобную процедуру повторяем до тех пор, пока не будет помечена вершина n.

Длина кратчайшего пути равна . Используя первую часть метки вершины, находим сам кратчайший путь методом обратного хода.

Конец работы алгоритма.

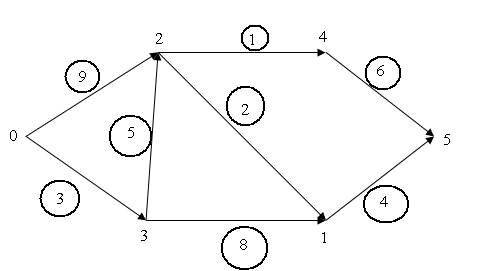


Рис.3.

Применим алгоритм Дейкстры к графу на рис. 3, числа в кружочках равны длинам дуг.

Шаг 0.

M(0) = (0,0),

M(i) =(0, ) для i = 1,2,…,n

Q = {0}

Шаг 1.

; ;

M(3) =(0, 3)

Q = {0, 3}

; ;

M(2) =(3, 8);

Q = {0, 3, 2}

; ;

Q = {0, 3, 2, 4}

; ;

Q = {0, 3, 2, 4, 1}

;

Q = {0, 3, 2, 4, 1, 5}

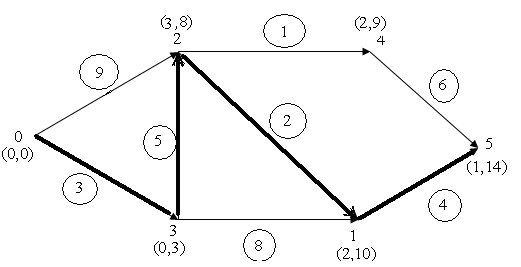


Рис.4

На рисунке 4 приведен пример применения алгоритма 1 для определения кратчайшего пути: числа у дуг равны длинам дуг, метки вершин помещены в круглые скобки, кратчайший путь выделен жирными линиями.