Уважаемые студенты!

Задание:

1. Прочтите приведенный ниже конспект лекции.

2. Напишите конспект лекции в тетрадь в соответствии с планов лекции.

3. Конспект лекции и творческое задание предоставить на электронную почту в течение трех дней.

ЛЕКЦИЯ 18. Некоторые классы графов и их частей.

ПЛАН

1. Некоторые классы графов и их частей. Деревья.
2. Ориентированные графы.
3. Взвешенные графы
4: Планарные графы .
5: Задача поиска маршрутов (путей) в графе (орграфе)
6: Поиск путей (маршрутов) с минимальным числом дуг (ребер)
7: Минимальный путь (маршрут) во взвешенном орграфе (графе)

1. Некоторые классы графов и их частей. Деревья.

Определение. Связный неориентированный граф без циклов называется *неориентированным деревом* или просто *деревом*.

Из определения следует, что дерево не может содержать ни петель, ни кратных рёбер.

Определение. Несвязный неориентированный граф без циклов называется *лесом*; связные компоненты леса являются деревьями.

Очевидно, что люба часть дерева или леса также не имеет циклов. В таком графе любая цепь является простой – в противном случае, она содержала бы цикл.

Теорема 1. Любые две вершины дерева связаны одной и только одной цепью. Обратно, если две любые вершины графа можно связать только одной цепью, то он является деревом.

Определение. Вершина v_0 называется концевой или висячей вершиной графа G, если её степень равна единице. Ребро, инцидентное концевой вершине, также называется концевым.

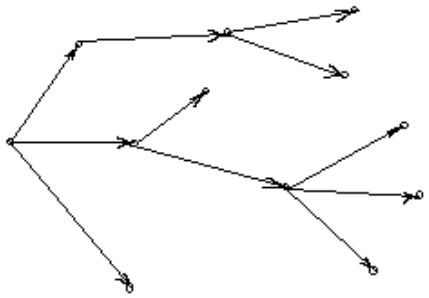
Если конечное дерево состоит более чем из одной вершины, оно имеет хотя бы две концевые вершины и хотя бы одно концевое ребро.

Пусть в дереве G отмечена некоторая вершина v_0. Эту вершину называют *корнем* дерева G, а само дерево – *деревом с корнем*. В таком дереве можно естественным образом ориентировать рёбра. Любую вершину v_m ребра (v_m, v_n) можно соединить с корнем единственной простой цепью. Если эта цепь не содержит ребра (v_m, v_n), то вводится ориентация от v_n к v_m; если цепь содержит

данное ребро, то вводится ориентация от v_m к v_n. Ориентированное таким образом дерево называется *ориентированным деревом с корнем*.

В нём все рёбра имеют направление от корня (см. рисунок 1).

Рисунок 1.



Если же изменить направления всех рёбер ориентированного дерева на противоположные (к корню), то получится ориентированный граф, который называется *сетью сборки*. В общем случае, такой граф тоже является ориентированным деревом. В каждую вершину ориентированного дерева, за исключением корня, входит только одно ребро. Иначе говоря, эта вершина является концом только одного ребра. Отсюда прямо следует, что в конечном дереве число вершин на один превышает число рёбер.

Замечание. Любое дерево можно ориентировать, выбрав в качестве корня любую его вершину.

Пусть дано конечное дерево G. Назовём его концевые вершины вершинами типа 1. Отметим, что если дерево имеет более двух вершин, то среди них есть неконцевые.

Далее удалим из дерева все вершины типа 1 и инцидентные им рёбра. Останется связный граф G', также являющийся деревом. Дерево G' также имеет концевые вершины, которые будем называть вершинами типа 2 в дереве G. Аналогичным образом определяются вершины типа 3 и так далее.

Легко видеть, что в конечном дереве имеются лишь вершины конечного числа типов, причём число вершин максимального типа равно одному или двум, так как в соответствующем дереве каждая вершина является концевой.

Теорема 2. Центрами дерева являются вершины максимального типа и только они.

Из данной теоремы прямо следует, что дерево имеет либо один, либо два центра. Диаметральные цепи в деревьях проходят через центр дерева, либо, если их два, через оба центра. В первом случае длина диаметральной цепи равна $2k-2$, во втором - $2k-1$, где k – максимальный тип дерева.

Определение. Цикломатическим числом конечного неориентированного графа G называется число, равное $\gamma(G) = n_G + n_E - n_V$. Здесь n_G – количество связных компонентов графа, n_E – количество рёбер, n_V – количество вершин.

Цикломатическое число дерева равно нулю. Цикломатические числа остальных конечных графов положительны.

2. Ориентированные графы.

Понятие ориентированного графа (орграфа) было определено ранее. Сейчас рассмотрим подробнее, как выглядят в таком графе пути и циклы.

Пусть дан ориентированный граф $G(V,E)$. Каждое ребро имеет начало $v_m \in V$ и конец $v_n \in V$; также говорят, что данное ребро выходит из вершины v_m и входит в вершину v_n.

Дадим определение пути в ориентированном графе. Сразу оговоримся, что это понятие можно определять различными способами; мы приводим только один.

<u>Определение.</u> Путь из вершин и рёбер – это такая последовательность рёбер и вершин графа $L=(v_1,e_1,v_2,e_2,\ldots,e_k,v_{k+1})$, в которой вершина v_i является началом $i-$ го ребра, а вершина $v_{i+1}-$ его концом. Вершина v_0 называется началом пути L, вершина $v_{k+1}-$ его концом, число рёбер $k-$длиной пути.

Путь, состоящий из одной вершины, имеет нулевую длину. Каждому пути L ненулевой длины взаимно однозначно соответствует последовательность $\tilde{L}=(e_1,\ldots,e_k)$ рёбер этого пути. Её называют *путём из рёбер*. Такое понятие пути – аналог соответствующего понятия для неориентированного графа. Наконец, для графа, не содержащего кратных рёбер, можно указать взаимнооднозначное соответствие с последовательностью $\hat{L}=(v_1,\ldots,v_{k+1})$ вершин пути. В зависимости от ситуации удобнее использовать тот или иной способ обозначения пути.

<u>Определение.</u> Путь $L=(v_1,e_1,v_2,e_2,\ldots,e_k,v_{k+1})$ называется *ориентированным циклом*, если состоит более, чем из одного элемента, и его начало совпадает с его концом.

Начало цикла обычно не фиксируется, иначе говоря, все пути, получающиеся друг из друга циклическими сдвигами – это один и тот же цикл. Определение простого ориентированного цикла аналогично соответствующему определению для неориентированного цикла – это цикл, в котором каждая вершина инцидентна ровно двум его рёбрам. Любой граф, содержащий циклы, можно “укоротить” до простого. Граф, не содержащий циклов, называется ациклическим.

<u>Определение.</u> Вершина ориентированного графа называется начальной, если в неё ни входит ни одно ребро и конечной, если из неё не выходит ни одно ребро.

Во всяком ациклическом графе есть хотя бы одна начальная и хотя бы одна конечная вершина. Максимальным рангом $R(v_0)$ вершины ориентированного графа называется максимальная из длин путей этого графа с концом в вершине v_0. Ранг вершины равен нулю тогда и только тогда, когда вершина является начальной. Если же через вершину проходит какой-нибудь цикл, то $R(v_0)=+\infty$.

Пусть вершины конечного ориентированного графа пронумерованы от 1 до n. Нумерация вершин называется правильной на ребре $e(v_i,v_j)$, если $i<j$ и правильной на графе G, если она правильна на всех его рёбрах. Правильная нумерация вершин графа возможна только в том случае, если он ациклический.

Понятия длины путей, протяжённости и расстояния между вершинами определяются для ориентированного графа так же, как для неориентированного графа.

3. Взвешенные графы

Определим понятие взвешенного (помеченного) графа. При этом сопоставим каждой вершине $v_i \in V$ графа $G = \langle V,E \rangle$ некоторый вес $w_i \in W$, а каждому ребру $e_i \in E$ также сопоставим некоторый вес $p_i \in P$.

Т.о., получим граф, содержащий взвешенные вершины (v_i, w_i) и взвешенные ребра (e_i, p_i), т.е. граф $G = \langle (V,W),(E,P) \rangle$.

Строго говоря, G представляет собой уже не граф, а функцию, определенную на вершинах и ребрах графа, т.е. если $f: V \to W$, $g: E \to P$, то $G = \langle f,g \rangle$.

Пример. В графе помеченными могут быть только вершины, или только ребра, или и вершины и ребра (Рис. 1).

Взвешенный граф полностью определяется матрицей инцидентности матрицами весов:

$$S_v = \begin{vmatrix} w_1 \\ w_2 \\ \dots \\ w_{|V|} \end{vmatrix} \quad \text{и } S_E = \begin{vmatrix} p_1 000 \dots 0 \\ 0 p_2 00 \dots 0 \\ \dots\dots \\ 0000 \dots p_{|E|} \end{vmatrix}$$

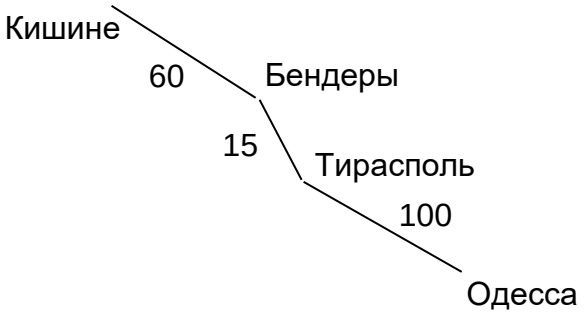


Рис. 1. Пример взвешенного графа.

Поиск остовного дерева минимальной длины во взвешенном графе.

Данный алгоритм (Алгоритм Крускаля) применяется при разработке различных коммуникационных сетей. Суть его состоит в том, чтобы сумма весов ребер полученного дерева была минимальной.

Пусть дан граф G с помеченными ребрами.

1. Выбирают ребро с минимальным весом.
2. Из оставшихся выбирают ребро с минимальным весом.
3. Проверяют, не образует ли выбранное ребро цикла с выбранными ранее ребрами. Если да, то отбрасывают его и переход к п 4 . Если нет, то включают его в отобранную совокупность ребер.

4. Если не все вершины графа еще связаны, то повторяем все действия, начиная с п.2. В противном случае – конец алгоритма.

Пример: В заданном взвешенном графе (Рис. 2) выделить остовное дерево минимальной длины.

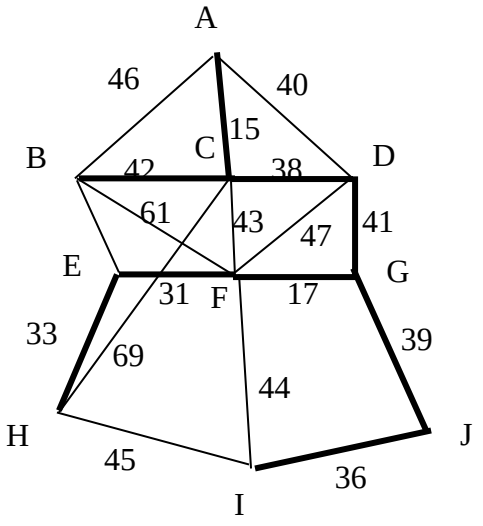


Рис..2.

Ребра расположены в порядке возрастания следующим образом: AC, FG, EF, EH, IJ, CD, GJ, AD, DG, BC, CF, FI, HI, AB, BE, FD, BF, CH. Алгоритм Крускаля позволяет последовательно сохранять ребра AC, FG, EF, EH, IJ, CD,CJ, а ребро AD исключается, т.к. оно образовывало бы цикл с уже включенными в дерево ребрами AC и CD. Затем добавляются ребра DG и BC. Как можно заметить, все вершины уже связаны и минимальное дерево получено.

4: Планарные графы

Определение:. Граф G называется *планарным,* если он может быть изображен на плоскости таким образом, чтобы его ребра не пересекались. Такой рисунок называется *картой графа* (Рис.3)

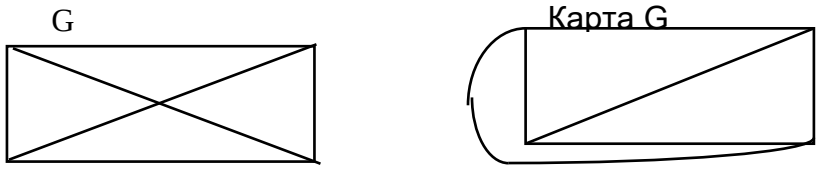


Рис.3.

Графы K_5 и $K_{3,3}$ не являются планарными (Рис.4).

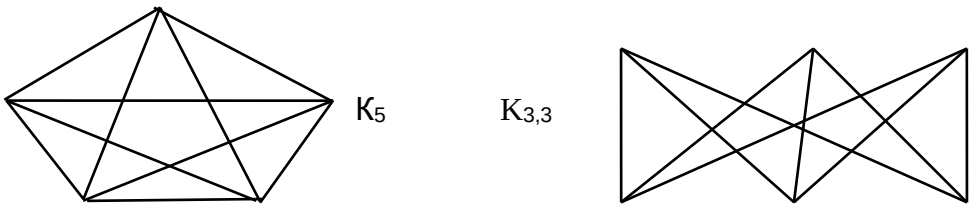


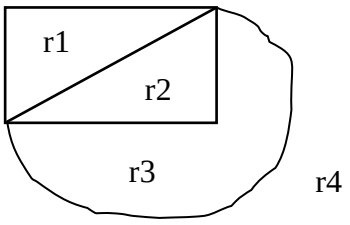
Рис 4..

Карта G называется *связной*, если граф G связный.

Каждая картина делит плоскость R^2 на области, ограниченные и неограниченные.

Множество таких областей обозначаются через R.

Теорема (Эйлера): Для произвольной связной карты выполняется равенство : IVI+IRI - IEI = 2 (Рис.5).



4 + 4 - 6 =2

Рис.5.

Каждая область карты ограничена замкнутым маршрутом .

Определение : Пусть G=<V,E> - планарный граф. Для карты G определим Δr - степень области r, как длину замкнутого маршрута, ограничивающего область r.

Для графа, изображенного на рис. 1, имеем:

$\Delta r_1=3$ $\Delta r_3=3$

$\Delta r_2=3$ $\Delta r_4=3$

Имеют место следующие соотношения:

$$\sum_{r\in G} \Delta r=2\,|E|$$

если $|V|\geq 3$, то $|E|\leq 3|V| - 6$

$6\leq 3*4-6$

$6\leq 6$

Рассмотрим операцию *стягивания* графа G как слияние двух смежных вершин после удаления ребра между ними.

Граф называется стягиваемым к G', если G' может быть получен из G путем последовательного выполнения элементарных стягиваний.

Теорема (Куратовского):. Граф является планарным, тогда и только тогда, когда он не содержит подграфов, стягиваемых к K_5 и $K_{3,3}$.

Определение : *Раскраской* графа G называется задание цвета вершинам таким образом, что если вершины v и w- смежные, $v,w \in V$, то они имеют различные цвета.

Величина $\chi(G)$ называется *хроматическим* числом графа G и равна минимальному числу цветов, необходимых для раскраски графа.

Граф G называется *p-хроматическим*, если его вершины можно раскрасить p различными цветами.

Одно-хроматический граф – тривиальный граф. Дву-хромотический граф тогда и только тогда, когда он не содержит циклов нечетной длины.

Теорема: Для любого планарного графа G хроматическое число не более 4.

Пусть M - карта графа G. Определим карту M’ следующим образом: внутри каждой области карты M выберем внутреннюю точку. Если две области имеют общее ребро, то между выбранными внутренними точками проведем дугу. Этот процесс определяет карту M’, называемую *двойственной* карте M’ (Рис. 6).

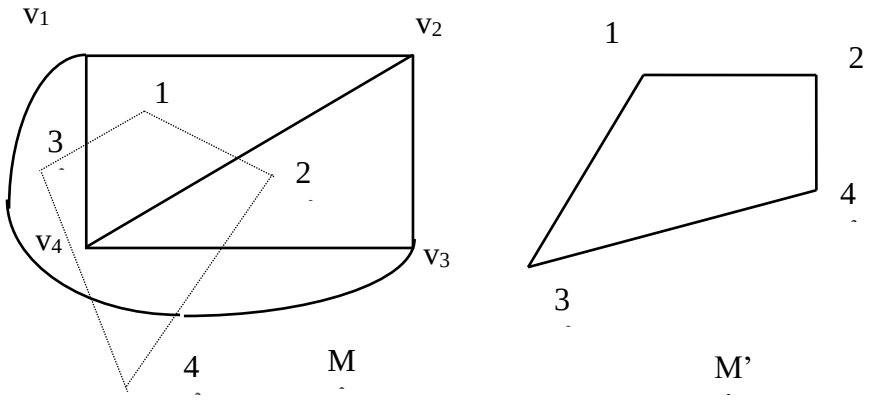


Рис. 6.

Раскраска карты M’ соответствует раскраске областей карты M т.о., что области, имеющие общее ребро, имеют разные цвета.

Теорема: Если области карты M необходимо раскрасить таким образом, чтобы смежные области имели различные цвета, то необходимо не более 4-х цветов.

Толщиной графа G называется наименьшее число планарных графов, объединение которых дает G. Толщина планарного графа равна 1. Нижняя оценка толщины t(G) графа G=<V,E> определяется неравенством:

$$t(G) \geq 1 + \left\lceil \frac{\sum_{i=1}^{n} \rho_i - 2}{6(n-2)} \right\rceil$$

, где ⌉ ⌊ -целая часть, |V|=n, ρ_i – степень i-ой вершины.

5: Задача поиска маршрутов (путей) в графе (орграфе)

Рассмотрим алгоритм поиска маршрута в связном графе G=<V, E>, соединяющего вершины $v,w \in V$ причем $v \neq w$.

Алгоритм Тэрри:

Если, исходя из вершины v и осуществляя последующий переход от каждой доступной вершины к смежной ей вершине, следовать следующим правилам 1-4, то всегда можно найти маршрут в связанном графе, соединяющим две заданные вершины.

Правило 1: Идя по произвольному ребру, отмечать направление, в котором оно было пройдено.

Правило 2: Исходя из некоторой вершины v' всегда следовать только по тому ребру, которое не было пройдено или пройдено в противоположном направлении.

Правило 3: Для всякой вершины $v' \neq v$ отмечать первое заходящее в v' ребро, если v′ встречается в первый раз.

Правило 4: Исходя из некоторой вершины $v' \neq v$ по первому заходящему в v' ребру идти лишь тогда, когда нет другой возможности.

Пример:

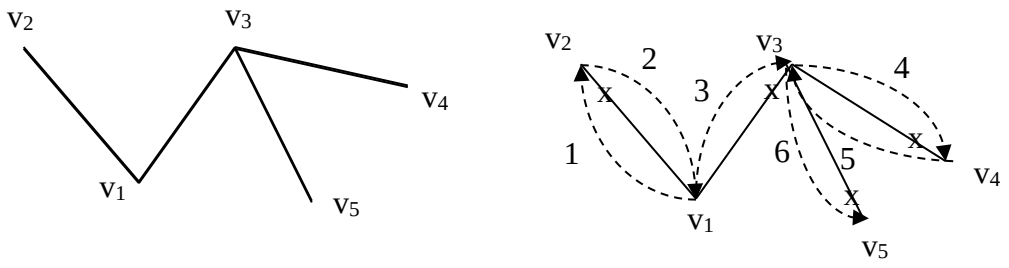


Рис. 7. а) б)

Найти маршрут из v_1 в v_5 (рис.7)

Знаком × помечены первые заходящие в вершины ребра возле той вершины, в которую это ребро заходит. По рисунку видно, что маршрут имеет вид: $v_1\ v_2\ v_1\ v_3\ v_4\ v_3\ v_5$.

После того как из v_1 зашли в v_3 по правилу 4 в v_1 вернуться не можем, т. к. имеются другие возможности, а ребро v_1v_3 является первым заходящим в v_3 ребром. После того как из v_4 зашли в v_3 по дуге 5 по правилу 2 в v_4 вернуться не можем, а в силу правила 4 не можем идти и к v_1, остается идти к v_5.

Замечание 1: Алгоритм Тэрри остается в силе, когда G связный граф.

Замечание 2: Если G не связный псевдограф, то с помощью алгоритма Тэрри, исходя из любой вершины v и помечая пройденные вершины и ребра, можно выделить компоненту связанности псевдографа, содержащую вершину v. Алгоритм закончит свою работу, когда в первый раз невозможно будет выполнить правило 2, т. е. пришли в вершину u такую, что все инцидентные ей ребра пройдены в направлении из u, при этом получим, что u = v.

Замечание 3: Из полученного с помощью алгоритма Тэрри маршрута всегда можно выделить простую цепь, соединяющую v и w.

6: Поиск путей (маршрутов) с минимальным числом дуг (ребер)

Путь в орграфе D из вершины v в вершину w, где $v \neq w$ называется *минимальным,* если он имеет минимальную длину среди всех путей в орграфе из v в w.

Рассмотрим свойства минимального пути(маршрута):

Утверждение 1: Любой минимальный путь (маршрут) является простой цепью.

Утверждение 2: (о минимальности подпути минимальность пути) Пусть $M=v_1, ..., v_k$ - минимальный путь (маршрут) из v_1 в v_k, где $v_1 \neq v_k$, тогда для любых номеров i, j таких, что $1 \leq i < j \leq k$ путь (маршрут) $M_0=v_i, v_{i+1}, ..., v_j$ также является минимальным .

Пусть $D=\langle V, E\rangle$ орграф, где $v \in V$, $V_1 \subseteq V$.

Множество $D(v)=\{w|\ (v, w) \in E\}$ называется образом вершины v.

Множество $D^{-1}(v)=\{w|\ (w, v) \in E\}$ прообраз вершины v.

$D(V_1)=\bigcup_{v\in V_1} D(v)$ – образ множества вершин V_1.

$D^{-1}(V_1)=\bigcup_{v\in V_1} D^{-1}(v)$ прообраз множества вершин V_1.

Пусть $G=<V, E>$ граф, где $v\in V$, $V_1\subseteq V$.

Множество $G(v)=\{w|\ (v, w)\in E\}$ называется образом вершины v.

$G(V_1)=\bigcup_{v\in V_1} G(v)$ – образ множества вершин V_1.

Рассмотрим алгоритм поиска минимального пути в орграфе с $n \geq 2$ вершинами из v в w, где $v\neq w$.

Алгоритм фронта волны:

Шаг 1: Помечаем вершину v индексом 0, затем помечаем вершины принадлежащие образу вершины v индексом 1. Множество вершин с индексом 1 обозначаем $FW_1(v)$ – фронт волны 1-го уровня. Полагаем k=1.

Шаг 2: Если $FW_k(v)=\varnothing$ или $k= n-1$ и $w\notin FW_k(v)$, то вершина w недостижима из v и алгоритм прекращает работу, иначе шаг 3.

*Шаг 3:*Если $w\notin FW_k(v)$ перейти к шагу 4, иначе существует путь из v в w длиной k и этот путь является минимальным. При этом последовательность вершин v w_1 w_2 ... w_{k-1} w, где

$w_{k-1}\in FW_{k-1}(v)\cap D^{-1}(w)$

$w_{k-2}\in FW_{k-2}(v)\cap D^{-1}(w_{k-1})$

(1)

.....

$w_1\in FW_1(v)\cap D^{-1}(w_2)$

и является искомым минимальным путем из v в w. На этом алгоритм прекращает свою работу.

Шаг 4: Помечаем индексом k+1 все непомеченные вершины, которые принадлежат множеству вершин с индексом k. Множество вершин с индексом k+1 обозначим $FW_{k+1}(v)$, присваиваем k:= k+1 и переходим к шагу 2.

Замечание 4: Множество $FW_k(v)$ называют фронтом волны k-ого уровня.

Замечание 5: Вершины w_1 w_2 ... w_{k-1} из (6.1) могут быть выделены неоднозначно, т. е. могут существовать несколько минимальных путей из v в w.

Пример: Определить минимальный путь из v_1 в v_6 в орграфе, заданном матрицей смежности.

	v	v	v	v	v	v
v	0	0	0	1	1	0
v	1	0	0	1	1	1
v	1	1	0	1	1	1
v	0	1	1	0	1	0

v	1	1	1	1	0	0
v	1	1	1	1	1	0

Следуя данному алгоритму, последовательно получаем, что:

$FW_1(v)=\{v_4, v_5\}$

$FW_2(v)=D(FW_1(v))\backslash\{v_1, v_4, v_5\}=\{v_2, v_3\}$

$FW_3(v)=D(FW_2(v))\backslash\{v_1, v_2, v_3, v_4, v_5\}=\{v_6\}$

$v_6\in FW_3(v)$, т. е. существует путь из v_1 в v_6 длиной 3 этот путь минимальный.

Определим множество

$FW_2(v)\cap D^{-1}(v_6)=\{v_2, v_3\}\cap\{v_2, v_3\}=\{v_2, v_3\}$ Выберем любую вершину из найденного множества, например v_3.

Далее определяем множество

$FW_1(v) \cap D^{-1}(v_3)=\{ v_4, v_5\}\cap\{ v_4, v_5, v_6\}=\{ v_4, v_5\}$

Из полученного множества выбираем любую вершину, например v_5. Таким образом получим, что искомым минимальным путем будет v_1 v_5 v_3 v_6.

Замечание 6: Если выражения D( ), D^{-1}( ) заменить на G( ), G^{-1}( ), то данный алгоритм применим и к неориентированному графу.

7: Минимальный путь (маршрут) во взвешенном орграфе (графе)

Пусть каждой дуге взвешенного орграфа D=<V, E> приписан некоторый вес – число l(e), где $e\in E$.

Для любого пути M через l(M) обозначим сумму длин входящих в него дуг и назовем *длиной пути* M во взвешенном орграфе.

Любой не взвешенный граф можно считать взвешенным, длина каждой дуги которого равна 1.

Пусть во взвешенном орграфе существует путь из v в w, где $v\neq w$. Он называется минимальным, если имеет минимальную длину среди всех возможных путей из v в w.

Рассмотрим свойства минимального пути взвешенного орграфа.

1) Если для любой дуги $e\in E$ $l(e)>0$, то любой минимальный путь (маршрут) является цепью.

2) Если $v_1, ..., v_k$ минимальный путь (маршрут) из v_1 в v_k, то для любых i, j таких, что $1\leq i < j \leq k$ путь (маршрут) $v_i, v_{i+1}, ..., v_j$ также минимальный.

3) Если v, ..., u, w минимальный путь (маршрут) из v в w, содержащий не более k+1 дуг (ребер), то v, ..., u также минимальный путь (маршрут), содержащий не более k дуг (ребер).

Рассмотрим задачу поиска минимального пути (маршрута) во взвешенном орграфе (графе).

Если орграф содержит ребро с отрицательным весом, то минимального пути может и не существовать.

Замечание 7: Если необходимо найти максимальный путь, то достаточно будет поменять знаки длин дуг на противоположные.

Пусть D=<V, E> взвешенный орграф с n≥2 дугами. Введем величины $\lambda_i^{(k)}$, где i =1, 2, ..., n k=1, 2, ... – длина минимального пути из v_1 в v_i, содержащего не более k дуг. Если таких путей нет, то $\lambda_i^{(k)}=\infty$. Если произвольную вершину v считать путем нулевой длины из v в v, то можем ввести величины:

$\lambda_1^{(0)}=0$ и $\lambda_i^{(0)}=\infty$ для i =2, 3, ..., n.

(2)

Рассмотрим квадратную матрицу n×n каждый элемент которой

$$C_{ij}=\begin{cases}1, \text{ если}(v_i, v_j)\in E\\ \infty, \text{ если}(v_i, v_j)\notin E\end{cases} \qquad (3)$$

Такую матрицу назовем *матрицей длин дуг* взвешенного орграфа.

Утверждение 3: При i =2, ..., n, k≥0 выполняется равенство

$$\lambda_i^{(k+1)}=\min_{1\le j\le n}\{\lambda_j^{(k)}+C_{j\,i}\}, \qquad (4)$$

а при i =1, k≥0 справедливо равенство

$$\lambda_1^{(k+1)}=\min\{0, \min_{1\le j\le n}(\lambda_j^{(k)}+C_{j,1})\} \qquad (5)$$

Используя данное утверждение можно описать алгоритм нахождения таблицы значений величин $\lambda_i^{(k)}$ в виде матрицы из i строк и (k+1) столбцов, начиная с k=0 и увеличивая k по мере необходимости.

Предположим, что в орграфе отсутствуют простые циклы отрицательной длины.

Утверждение 4: Из всякого замкнутого пути отрицательной длины можно выделить простой цикл отрицательной длины.

Утверждение 5:. Если во взвешенном орграфе отсутствуют простые циклы отрицательной длины, то в нем нет и замкнутых путей отрицательной длины.

Утверждение 6: Во взвешенном орграфе, в котором нет простых циклов отрицательной длины, из всякого незамкнутого пути можно выделить простую цепь с теми же началом и концом, длина которой не превосходит длину исходного пути.

Рассмотрим алгоритм, который позволяет по таблице $\lambda_i^{(k)}$, где i =2, ..., n k=0, 1,...,n-1, определить минимальный путь во взвешенном орграфе из v_1 в любую достижимую вершину, причем из всех возможных путей выбирается путь с минимальным числом дуг.

Алгоритм Форда-Беллмана нахождения пути во взвешенном орграфе из v_1 в v_{i_1} $i_1\neq1$

Шаг 1: Пусть таблица величин $\lambda_i^{(k)}$ где i =2, ..., n k=0, 1, ... уже составлена. Если $\lambda_{i_1}^{(n-1)}=\infty$, то вершина v_i не достижима из v_1 и алгоритм заканчивает свою работу. В данном алгоритме принимают, что l(e), e∈E – конечное число.

Шаг 2: Пусть $\lambda_{i_1}^{(n-1)}<\infty$, тогда $\lambda_{i_1}^{(n-1)}$ выражает длину любого минимального пути из v_1 в v_{i1}. Определим минимальное число $k_1\geq 1$, при котором выполняется равенство: $\lambda_{i_1}^{(k_1)}=\lambda_{i_1}^{(n-1)}$. По определению чисел $\lambda_i^{(k)}$ получаем, что k_1 - это минимальное число дуг в пути среди всех минимальных путей из v_1 в v_{i_1}.

Шаг 3: Последовательно определяем номера $i_2, i_3, ..., i_{k_1+1}$

$$\begin{aligned}
&\lambda_{i_2}^{(k_1-1)}+C_{i_2,\,I^I}=\lambda_{i_1}^{(k_1)}\\
&\lambda_{i_3}^{(k_1-2)}+C_{i_3,\,i_2}=\lambda_{i_2}^{(k_1-1)} \qquad (6)\\
&\cdots\cdots\cdots\cdots\cdots\cdots\\
&\lambda_{i_{(k_1+1)}}^{(0)}+C_{i_{(k_1+1)},i_{k_1}}=\lambda_{i_{k_1}}^{(1)}
\end{aligned}$$

эти номера находим в соответствии с равенством (2). С учетом того, что $\lambda_{i_1}^{(n-1)}=\lambda_{i_1}^{(k_1)}<\infty$ имеем, что $C_{i_2,i_1}<\infty$, ..., $C_{i_{(k_1+1)},i_{k_1}}<\infty$, $\lambda_{i_{(k_1+1)}}^{(0)}<\infty$, откуда, используя (2) получаем ребро(v_{i_2},v_{i_1}),...,$(v_{i_{(k_1+1)}},v_{i_{(k_1)}})\in E$, $l(v_{i_2},v_{i_1})=C_{i_2,i_1}$,..., $l(v_{i_{(k_1+1)}},v_{i_{(k_1)}})=C_{i_{(k_1+1)},i_{k_1}}$,

$$\lambda_{i_{(k_1+1)}}^{(0)}=0,\ i_{k_1+1}=1,\ v_{i_{(k_1+1)}}=v_1. \qquad (7)$$

Складывая (6) и учитывая (7) имеем, что длина маршрута $l(v_1, v_{i_{k_1}}, ..., v_{i_2}, v_{i_1})=\lambda_{i_1}^{(k_1)}$, т. е. $v_1, v_{i_{k_1}}, ..., v_{i_2}, v_{i_1}$ – искомый минимальный путь из v_1 в v_{i_1} в орграфе D с k_1 дугами.

Замечание 8: Номера $i_2, i_3,...i_{k_1}$, удовлетворяющие (6) могут быть выделены неоднозначно, т. е. может существовать несколько минимальных путей из v_1 в v_{i_1}.

Замечание 9:Данный алгоритм можно модифицировать т.о., чтобы определить минимальный путь из v_1 в v_{i_1}, содержащий не более k_0 дуг, где $k_0\geq 1$ – заданное число. Для этого в алгоритме вместо $\lambda_{i_1}^{(n-1)}$ необходимо рассматривать $\lambda_{i_1}^{(k_0)}$. При этом, если в орграфе D имеются простые циклы отрицательной длины, может выполняться неравенство $\lambda_1^{(k_0)}<0$.

Замечание 10: Алгоритм Форда-Беллмана и его модификация, описанная выше, после соответствующего изменения в терминологии и обозначениях применимы и к неориентированному графу.

Пример: Определим минимальный путь из v_1 в v_6 во взвешенном орграфе (рис.8), матрица длин дуг которого имеет вид:

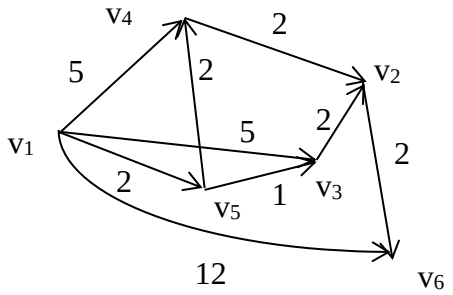


Рис.8.

	v_1	v_2	v_3	v_4	v_5	v_6	$\lambda_i^{(0)}$	$\lambda_i^{(1)}$	$\lambda_i^{(2)}$	$\lambda_i^{(3)}$	$\lambda_i^{(4)}$	$\lambda_i^{(5)}$
v_1	∞	∞	5	5	2	12	0	0	0	0	0	0
v_2	∞	∞	∞	∞	∞	2	∞	∞	7	5	5	5
v_3	∞	2	∞	∞	∞	∞	∞	5	3	3	3	3
v_4	∞	2	∞	∞	∞	∞	∞	5	4	4	4	4
v_5	∞	∞	1	2	∞	∞	∞	2	2	2	2	2
v_6	∞	∞	∞	∞	∞	∞	∞	12	12	9	7	7

Величина $\lambda_6^{(5)}$ выражает длину минимального пути из v_1 в v_6. Найдем минимальное число $k_1 \geq 1$ дуг, при котором выполняется равенство $\lambda_6^{(5)} = \lambda_6^{(k_1)}$ Из таблицы получаем $k_1=4$, значит минимальный маршрут будет содержать самое меньшее 4 дуги. Определим теперь последовательность номеров $i_1, i_2, ..., i_5$, где $i_1=6$, в минимальном пути, удовлетворяющую (6). Из таблицы получаем, что это последовательность номеров 6,2,3,5,1, так как:

$\lambda_2^{(3)}+C_{2,6}=5+2=7=\lambda_6^{(4)}$

$\lambda_3^{(2)}+C_{3,2}=3+2=5=\lambda_2^{(3)}$

$\lambda_5^{(1)}+C_{5,3}=2+1=3=\lambda_3^{(2)}$

$\lambda_1^{(0)}+C_{1,5}=0+2=2=\lambda_5^{(1)}$.

Тогда v_1 v_5 v_3 v_2 v_6 – минимальный путь из v_1 в v_6, содержащий наименьшее число дуг.