Министерство образования и науки Украины

Севастопольский национальный технический университет


# ПРИМЕНЕНИЕ ГРАФОВЫХ МОДЕЛЕЙ В ПРОЕКТИРОВАНИИ КОМПЬЮТЕРНЫХ СИСТЕМ

Методические указания
к практическому и лабораторному
занятиям по дисциплине
" **ТЕХНОЛОГИЯ ПРОЕКТИРОВАНИЯ КОМПЬЮТЕРНЫХ СИСТЕМ**"
для студентов дневной формы обучения
направления 6.050102 “Компьютерная инженерия”

Севастополь
2014

УДК 004.53.06

Применение графовых моделей в проектировании компьютерных систем. Методические указания к выполнению практического и лабораторного занятий по дисциплине "Технология проектирования компьютерных систем" для студентов дневной формы обучения направления 6.050102 “Компьютерная инженерия”/ составители А.И. Тертычный, В.И.Шевченко.- Севастополь: Изд-во СевНТУ, 2014.- 28 с.

В методических указаниях кратко изложены основные положения теории графов, способы их представления и применения при проектировании компьютерных систем. Рассматриваются примеры представления графа и выполнения операций с графами. Приведены варианты заданий для выполнения лабораторной работы студентов.

Методические указания рассмотрены и утверждены на заседании кафедры кибернетики и вычислительной техники, протокол № 11 от 25 июня 2014 г.

Рецензент:
Мащенко Е.Н., к.т.н., доцент кафедры кибернетики и вычислительной техники.

Допущено учебно-методическим центром СевНТУ в качестве методических указаний.

# СОДЕРЖАНИЕ

# 1. ОБЩИЕ СВЕДЕНИЯ О ГРАФАХ

Множество вершин ***V,*** связи между которыми определены множеством ребер ***E,*** называют *графом* и обозначают ***G = (V, E)***. Ребро указывает связь между вершинами, не имеющую направления. Такой граф называется *неориентированным* [1]. Пример неориентированного графа дан на рис.1.

В качестве примера неориентированного графа можно привести сеть связи компьютеров – сеть передачи данных (СПД).

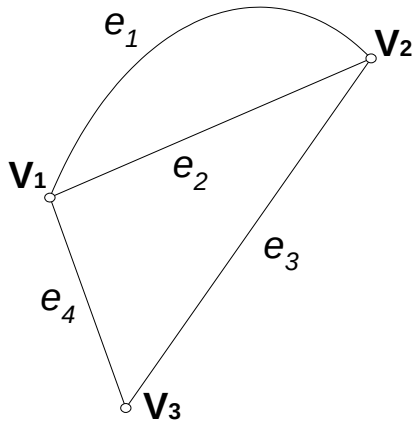



Рисунок 1 – Неориентированный граф

## 1.1. Ориентированные графы

Часто связи между объектами характеризуются вполне определенной ориентацией. Например, на некоторых улицах допускается только одностороннее автомобильное движение, в соединительных проводах электрической цепи задаются положительные направления токов, отношения между людьми могут определяться подчиненностью или старшинством. Ориентированные связи характеризуют переход системы из одного состояния в другое, различные отношения между числами (неравенство, делимость).

Для указания направления связи между вершинами графа соответствующее ребро отмечается стрелкой. Ориентированное таким образом ребро называют дугой, а граф с ориентированными ребрами – *ориентированным графом* или *орграфом* (рис. 2, *а).*

Если пара вершин соединяется двумя или большим числом дуг, то такие дуги называют ***параллельными.*** При этом две дуги, одинаково направленные по отношению к данной вершине, называют ***строго параллельными,*** а различно направленные ***нестрого параллельными***[1]. Ясно, что нестрого параллельные дуги, отображающие ориентацию связи в обоих направлениях, по существу равноценны неориентированной связи и могут быть заменены ребром. Произведя такую замену в

орграфе, придем к *смешанному* графу, который содержит ребра и дуги (рис. 2, *б)*. И наоборот, любой неориентированный или смешанный граф можно преобразовать в ориентированный заменой каждого ребра парой нестрого параллельных дуг.

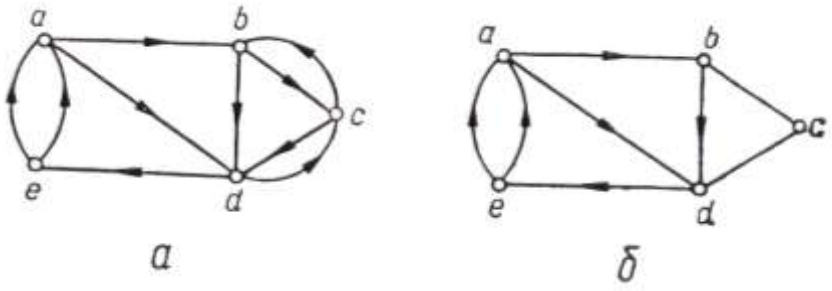



Рисунок 2 – Пример ориентированного (а) и смешанного (б) графов

Изменив направления всех дуг орграфа на противоположные, получаем орграф, ***обратный*** исходному. Если направления дуг орграфа не учитываются и каждая дуга рассматривается как неориентированное ребро, то он называется *соотнесенным* (неориентированным) графом.

## 1.2 Взвешенные графы

Дальнейшее обобщение отображения связей между объектами с помощью графов состоит в приписывании ребрам и дугам некоторых количественных значений, качественных признаков или характерных свойств, называемых ***весами.*** В простейшем случае это может быть порядковая нумерация ребер и дуг, указывающая на очередность при их рассмотрении (приоритет или иерархия). Вес ребра или дуги может означать длину (пути сообщения), пропускную способность (линии связи), напряжение или ток (электрические цепи), количество набранных очков (турниры), валентность связей (химические формулы), количество рядов движения (автомобильные дороги), цвет проводника (монтажная схема электронного устройства), характер отношений между людьми (сын, брат, отец, подчиненный, учитель) и т. п.

Вес можно приписывать не только ребрам и дугам, но и вершинам. Например, вершины, соответствующие населенным пунктам на карте автомобильных дорог, могут характеризоваться количеством мест в кемпингах, пропускной способностью станций техобслуживания. Вообще, вес вершины означает любую характеристику соответствующего ей объекта (атомный вес элемента в структурной формуле, цвет изображаемого вершиной предмета, возраст человека и т. п.).

Особое значение для моделирования физических систем приобрели взвешенные ориентированные графы, названные ***графами потоков сигналов*** или ***сигнальными графами.*** Вершины сигнального графа отождествляются с некоторыми переменными, характеризующими состояние системы, а вес каждой вершины означает функцию времени или некоторые величины, характеризующие соответствующую переменную ***(сигнал вершины).*** Дуги отображают связи между переменными, и вес каждой дуги представляет собой численное или функциональное отно-

шение, характеризующее передачу сигнала от одной вершины к другой *(передача дуги)*. Сигнальные графы находят широкое применение в теории цепей и систем, а также во многих других областях науки и техники.

## 1.3 Типы конечных графов

Если множество вершин графа конечно, то он называется ***конечным графом***. В математике рассматриваются и бесконечные графы, но мы заниматься ими не будем, так как в практических приложениях они встречаются редко. Конечный граф $G = (V, E)$, содержащий $p$ вершин и $q$ ребер, называется *(p,q)-графом.*

Пусть $V = \{v_1, v_2, ..., v_p\}$ и $E = \{e_1, e_2, .... e_q\}$ – соответственно множества вершин и ребер (p, q)-графа. Каждое ребро $e_k \in E$ соединяет пару вершин $v_i\ v_j \in V$, являющихся его *концами (граничными вершинами)*. Для ориентированного ребра (дуги) различают *начальную вершину,* из которой дуга исходит, и *конечную вершину,* в которую дуга заходит. Ребро, граничными вершинами которого является одна и та же вершина, называется ***петлей.*** Ребра с одинаковыми граничными вершинами являются параллельными и называются ***кратными.*** В общем случае граф может содержать и ***изолированные вершины,*** которые не являются концами ребер и не связаны ни между собой, ни с другими вершинами [3]. Например, для (5, 6)-графа на рис. 3, *а* $V = \{v_1, v_2, v_3, v_4, v_5\}$; $E = \{e_1, e_2, e_3, e_4, e_5, e_6\}$; ребра $e_2$ и $e_3$ параллельны, ребро $e_6$ является петлей, а $v_4$ – изолированная вершина.

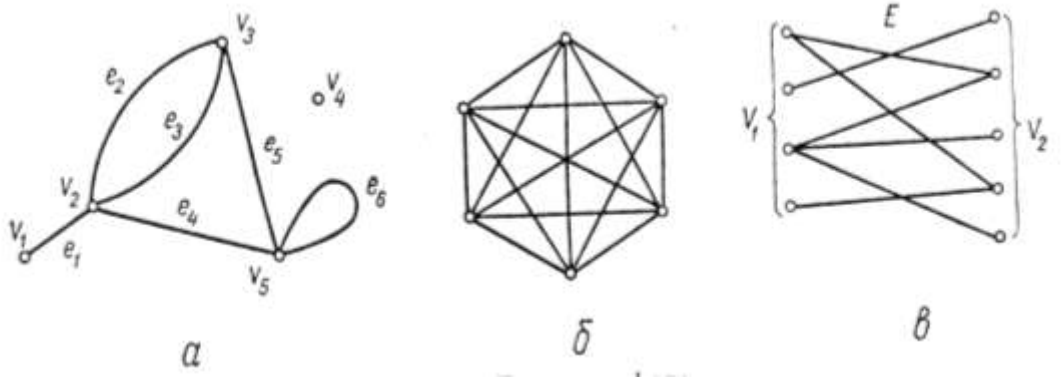



Рисунок 3 – Примеры графов

Число ребер, связанных с вершиной $v_i$ (петля учитывается дважды), называют ***степенью вершины*** и обозначают через $\delta(v_i)$ или $\deg(v_i)$. Так, для графа на рис. 3а $\delta(v_1) = 1$, $\delta(v_2) = 4$ и т. д. Очевидно, степень изолированной вершины равна нулю ($\delta(v_4) = 0$). Вершина степени единицы называется ***концевой или висячей вершиной*** ($\delta(v_1) = 1$).

Граф без петель и кратных ребер называют ***простым*** **или** ***обыкновенным.*** Граф без петель, но с кратными ребрами называют ***мультиграфом.*** Наиболее общий случай графа, когда допускаются петли и кратные ребра, называют ***псевдографом.*** Так, граф на рис. 1 – это мультиграф, а на рис. 3а – псевдограф. Если граф не имеет ребер ($E = 0$), то все его вершины изолированы ($V \neq 0$), и он называется ***пустым*** или ***нуль-графом.*** Простой граф, в котором любые две вершины соединены ребром, называет-

ся ***полным*** (на рис. 3б приведен пример полного графа с шестью вершинами). Если множество вершин *V* простого графа допускает такое разбиение на два непересекающихся подмножества $V_1$ и $V_2$ ($V_1 \cap V_2 = 0$), что не существует ребер, соединяющих вершины одного и того же подмножества, то он называется ***двудольным*** или ***биграфом*** (рис. 3в). Ориентированный граф считается ***простым***, если он не имеет строго параллельных дуг и петель [3].

Граф, степени всех вершин которого одинаковы и равны ***r***, называется ***однородным (регулярным) r-й степени***. Полный граф с ***n*** вершинами всегда однородный степени ***n - 1***, а пустой граф — однородный степени 0. Граф третьей степени называют ***кубическим***. Он обладает рядом интересных свойств и, в частности, всегда имеет четное число вершин.

## 1.4 Смежность

Две вершины $v_i$ и $v_j \in V$ графа $G = (V, E)$ называются ***смежными***, если они являются граничными вершинами ребра $e_k \in E$. Отношение смежности на множестве вершин графа можно определить, представив каждое ребро как пару смежных вершин, т.е. $e_k = (v_i\ v_j)$, k = 1, 2, ..., q. Для неориентированных графов такие пары неупорядочены, так что $e_k = (v_i, v_j) = (v_j, v_i)$, а для орграфов — упорядочены, причем $v_i$ и $v_j$ - означают соответственно начальную и конечную вершины дуги $e_k$. Петля при вершине $v_i$ в обоих случаях представляется неупорядоченной парой $(v_i, v_i)$. Ясно, что множество вершин V вместе с определенным на нем отношением смежности полностью определяет граф.

Граф можно представить также ***матрицей смежности.*** Строки и столбцы этой матрицы соответствуют вершинам графа, а ее (*ij*)-элемент равен числу кратных ребер, связывающих вершины $v_i$ и $v_j$ (или направленных от вершины $v_i$ к вершине $v_j$ для орграфа). Например, для графов, приведенных на рис. 2,*а* и 3,а, имеем соответственно следующие матрицы смежности:

$\boldsymbol{V_1} =$

| | *a* | *b* | *c* | *d* | *e* |
|---|---|---|---|---|---|
| *a* | | 1 | | 1 | |
| *b* | | | 1 | 1 | |
| *c* | | 1 | | 1 | |
| *d* | | | 1 | | 1 |
| *e* | 2 | | | | |

;

$\boldsymbol{V_2} =$

| | $v_1$ | $v_2$ | $v_3$ | $v_4$ | $v_5$ |
|---|---|---|---|---|---|
| $v_1$ | | 1 | | | |
| $v_2$ | 1 | | 2 | | 1 |
| $v_3$ | | 2 | | | 1 |
| $v_4$ | | | | | |
| $v_5$ | | 1 | 1 | | 1 |

.

Матрица смежности неориентированного графа всегда симметрична, а орграфа − в общем случае несимметрична. Неориентированным ребрам соответствуют пары ненулевых элементов, симметричных относительно главной диагонали матрицы, дугам − ненулевые элементы матрицы, а петлям − ненулевые элементы главной диагонали. В столбцах и строках, соответствующих изолированным вершинам, все элементы равны нулю. Элементы матрицы простого графа равны 0 или I, причем все элементы главной диагонали нулевые.

Для взвешенного графа, не содержащего кратных ребер, можно обобщить матрицу смежности так, что каждый ее ненулевой элемент равняется весу соответствующего ребра или дуги. Матрица смежности для неориентированного графа всегда симметрична относительно главной диагонали. И наоборот, любая квадратная матрица n-го порядка может быть представлена орграфом с *n* вершинами, дуги которого соединяют смежные вершины и имеют веса, равные соответствующим элементам матрицы. Если матрица симметрична, то она представима неориентированным графом.

Если граф взвешенный, т.е. каждая дуга графа имеет вес, элементы матрицы смежности содержат значения весов.

## 1.5 Инцидентность

Если вершина $v_i$ является концом ребра $e_k$, то говорят, что они *инцидентны:* вершина $v_i$ инцидентна ребру $e_k$ и ребро $e_k$ инцидентно вершине $v_i$ . В то время как смежность представляет собой отношение между однородными объектами (вершинами), инцидентность — это отношение между разнородными объектами (вершинами и ребрами). При рассмотрении орграфов различают ***положительную*** *инцидентность* (**дуга исходит** из вершины) и ***отрицательную*** *инцидентность* (**дуга заходит** в вершину).

Рассматривая инцидентность вершин и ребер (p, q)-графа, можно представить его ***матрицей инцидентности*** размера $p \times q$. Для неориентированного графа элементы этой матрицы определяются по следующему правилу: **ij-элемент** равен 1, если вершина $v_i$, **инцидентна** ребру $e_j$, и равен нулю, если $v_i$ и $e_j$ не инцидентны. В случае орграфа ненулевой **ij-элемент** равен 1, если $v_i$, начальная вершина дуги $e_j$, и равен −1, если $v_i$ − конечная вершина дуги $e_j$.

Например, матрица инцидентности неориентированного графа, приведенного на рис. 3, *а,* имеет вид:

$A=$

| $e_1$ | $e_2$ | $e_3$ | $e_4$ | $e_5$ | $e_6$ | |
|---|---|---|---|---|---|---|
| 1 | | | | | | $v_1$ |
| 1 | 1 | 1 | 1 | | | $v_2$ |
| | 1 | 1 | | 1 | | $v_3$ |
| | | | | | | $v_4$ |
| | | | 1 | 1 | | $v_5$ |

Каждый столбец матрицы инцидентности содержит обязательно два единичных элемента (для орграфа эти элементы всегда имеют различные знаки и равны соответственно «1» и «-1»). Количество единиц в строке равно степени соответствующей вершины (для орграфа количество положительных единиц определяет положитель-

ную степень, а количество отрицательных единиц − отрицательную степень). Нулевая строка соответствует изолированной вершине, а нулевой столбец − петле.

Следует иметь в виду, что нулевой столбец матрицы инцидентности лишь указывает на наличие петли, но не содержит сведений о том, с какой вершиной эта петля связана (в практических приложениях это может быть несущественно).

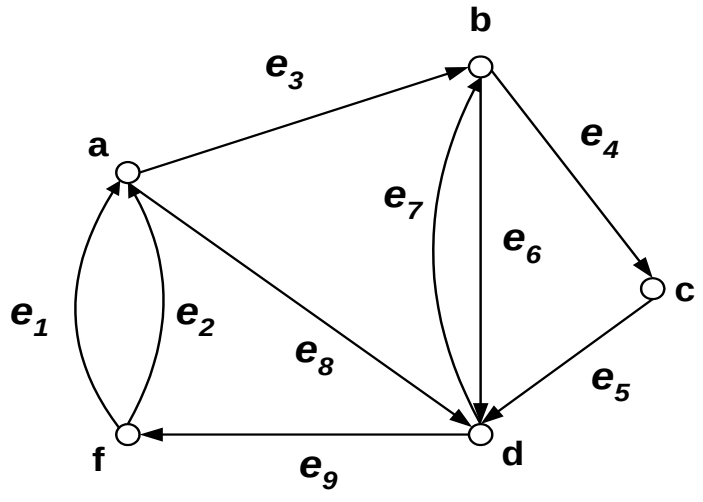
Ориентированный граф, изображенный на рисунке 4, может быть представлен матрицей инцидентности A1

***A1*** =

| | $e_1$ | $e_2$ | $e_3$ | $e_4$ | $e_5$ | $e_6$ | $e_7$ | $e_8$ | $e_9$ |
|---|---|---|---|---|---|---|---|---|---|
| **a** | -1 | -1 | 1 | | | | | 1 | |
| **b** | | | -1 | 1 | | 1 | -1 | | |
| **c** | | | | -1 | 1 | | | | |
| **d** | | | | | -1 | -1 | 1 | -1 | 1 |
| **f** | 1 | 1 | | | | | | | -1 |

## 1.6. Список дуг

Представление с помощью списка дуг применяется в тех случаях, когда необходимо иметь отдельную, независимую нумерацию дуг. При этом способе каждой дуге сопоставляется тройка ($e$, $v_i$, $v_j$), где $e$ – дуга, $v_i$ – ее начало, $v_j$ – ее конец. Этот способ представления легко обобщается на случай взвешенных графов. Более того, он наиболее приспособлен для хранения различной информации о дугах.




| e | $v_i$ | $v_j$ |
|---|---|---|

Рисунок 4 − Представление графа в виде списка дуг

В случае взвешенного графа к описанию дуги добавляется поле – значение веса.

## 1.7. Изоморфизм

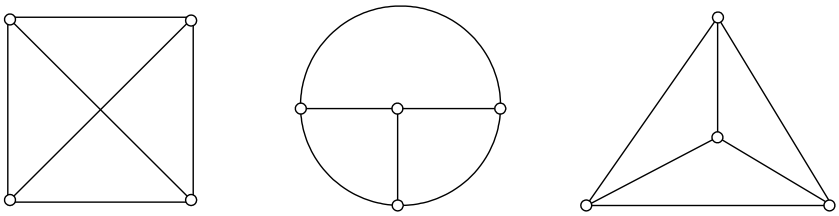


Рисунок 5 – Изоморфные графы

На рисунке 5 изображены три графа, которые с геометрической точки зрения совершенно различны (пересечение ребер, если оно не отмечено точкой, не является вершиной). Но по существу они различаются лишь начертанием, а отношения инцидентности (при соответствующем обозначении вершин и ребер) для них одинаковы. Графы, для которых сохраняется отношение инцидентности, называются ***изоморфными.***

# 2. СТРУКТУРЫ ПРЕДСТАВЛЕНИЯ ГРАФОВ В ПАМЯТИ ЭВМ

При изучении взаимосвязи между объектами (взаимоотношений людей в коллективах, связей в ЭВА (электронно-вычислительной аппаратуры), в молекулах, микросхемах и т.п.) исследуемый объект обозначается точками и соединяющими их линиями. Такое изображение объектов ввиду наглядности позволяет определять наиболее существенные связи, находить оптимальное решение задачи, определять ошибки в рассуждениях и т. д. Объект, состоящий из точек и соединяющих их линий, оказывается удобной моделью схем ЭВА, и в этой связи представляет интерес исследование самого этого объекта.

Теория графов оперирует формальными моделями объектов, имеет дело со свойствами самих графов независимо от того, какова природа объектов, описываемых графами. Использование аппарата теории графов для разработки алгоритмов конструкторского и технологического проектирования схем ЭВА приводит к повышению эффективности и качества создаваемых объектов.

Остановимся на представлении моделей графов.

Представление графов в памяти ЭВМ существенно зависит от типов структур данных, допускаемых используемым алгоритмическим языком и типов ЭВМ [2].

## 2.1. Представление графа с помощью матрицы смежности

Любая программная система оценивается, прежде всего, по двум критериям:

а) быстродействию, величина которого должна стремиться к увеличению;

б) объему памяти, величина которого должна стремиться к уменьшению.

Задача, решаемая в рамках САПР – найти оптимальное соответствие между объемом памяти и быстродействием.

Одно из самых широко распространенных и наиболее удобное для обработки представление графа в виде матрицы смежности.

Матрица смежности представляет собой математическую модель. Для графов с большим числом дуг это достаточно компактное представление.

На рисунке 6 изображен ориентированный граф. Представим этот граф матрицей смежности.

Строки матрицы соответствуют исходящим вершинам, а столбцы – входящим. Значение элемента матрицы (ячейки) соответствует весу дуги.

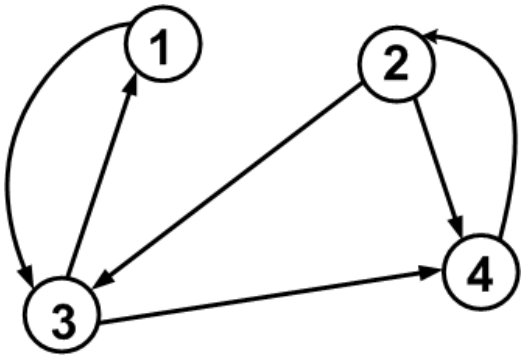



Рисунок 6 – Ориентированный граф

Граф невзвешенный, поэтому, каждая ячейка матрицы должна отражать только наличие или отсутствие дуги и может принимать два значения: например, 0 или 1. Примем $M_{i,j} = 1$, если между вершинами $i$ и $j$ дуга существует, и $M_{i,j} = 0$, если между этими вершинами дуги нет.

Количество ячеек или элементов в матрице смежности равно $n^2$.

Размерность матрицы определяется по формуле

$$R = n^2 \cdot L, \qquad (1)$$

где $n$ – число вершин в графе;

$L$ – размер одного элемента матрицы.

**M**

| *i* \ | *1* | *2* | *3* | *4* |
|---|---|---|---|---|
| *1* | **0** | | **1** | |
| *2* | | **0** | **1** | **1** |
| *3* | **1** | | | **1** |
| *4* | | **1** | | |

Минимальная ячейка памяти, с которой компьютер выполняет операцию – 1 байт, поэтому размер одного элемента матрицы в оперативной памяти будет равен 1 байту, ($L$=1байт).

Тогда размер матрицы будет составлять $n^2$ байт.

Для графа, представленного на рисунке 6, $n = 4$ и, соответственно $R = 16$ байт.

Если учесть, что для кодирования 0 и 1 достаточно всего 1 бита, то длина каждого элемента матрицы в оперативной памяти *можно выделить 1 бит* ($L'$=1 бит), тогда размерность матрицы в битах $R'$*=16 бит = 2 байта.* Сравнивая две величины $R$ и $R'$, характеризующие одну и ту же математическую модель, делаем вывод, что в первом случае оперативная память используется нерационально. Так как, выделив байт (т.е. 8 бит) для одного элемента матрицы, реально используется только один бит.

В реальных задачах, когда графы имеют 1000 и более вершин, для хранения информации в памяти вычислительного устройства, нужно представить ее в виде одномерного битового вектора.

Подробнее рассмотрим битовую форму представления графа на конкретном примере для нашего графа.

Представление может быть либо по строкам, либо по столбцам. Мы будем представлять по строкам. Строим одномерный битовый вектор из 16 ячеек. Запишем информацию построчно.

Одномерный битовый вектор

| стр. 1 | | | | стр. 2 | | | | стр. 3 | | | | стр. 4 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *10* | *11* | *12* | *13* | *14* | *15* | *16* |
| **0** | **0** | **1** | **0** | **0** | **0** | **1** | **1** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **0** |
| 4 бит | | | | 4 бит | | | | 4 бит | | | | 4 бит | | | |
| один байт | | | | | | | | один байт | | | | | | | |

В общем случае позиция элемента в векторе определяется по формуле

$$P_{M_{i,j}} = (n \cdot (i-1) + j - 1) \cdot L + 1, \qquad (2)$$

где $P_{M_{i,j}}$ – позиция элемента $M_{i,j}$ в векторе;

$i$ – номер строки матрицы смежности;

$j$ – номер столбца матрицы смежности

$L$ – длина элемента в оперативной памяти.

Например, определим позицию для элемента $M_{2,3}$

$$P_{M_{2,3}} = (4 \cdot (2-1) + 3 - 1) \cdot 1 + 1 = 7$$

Седьмой элемент вектора равен 1, что указывает наличие дуги из вершины 2 в вершину 3.

Эта формула справедлива в том случае, если нумерация бит в векторе и строк и столбцов *в матрице* начинается с 1. Если биты строчки и столбцы нумеровать, начиная с 0, то величина $P_{M_{i,j}}$ будет вычисляться по формуле

$$P_{M_{i,j}} = (n \cdot i + j) \cdot L \qquad (3)$$

Итак, применяя формулу (3), для элемента матрицы $M_{1,2}$ , получим:

$$P_{M_{1,2}} = 4 \times 1 + 2 = 6$$

$$P_{M_{1,2}} = 6$$

Следовательно, хранит информацию шестой бит.

| *0* | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *1* | *11* | *12* | *13* | *14* | *15* |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | **0** | **1** | **0** | **0** | **0** | **1** | **1** | **1** | **0** | **0** | **1** | **0** | **1** | **0** | **0** |
| *0 байт* | | | | | | | | 1 байт | | | | | | | |

Так как компьютер выполняет операции с байтом, то для выделения требуемого элемента необходимо определить номер байта, в котором он находится, и позицию бита в байте.

Номер байта, в котором находится заданный бит, определяется по формуле

$$N_{\text{байт}} = \left(P_{M_{i,j}} - 1\right) \; div \; 8 + 1 \qquad (4),$$

а номер бита в байте определяется как

$$N_{\text{бит}} = P_{M_{i,j}} - \left(N_{\text{байт}} - 1\right) \cdot 8, \qquad (5),$$

или

$$N_{\text{бит}} = \text{mod}\left[\left(P_{M_{i,j}} - 1\right), 8\right] + 1. \qquad (6)$$

при условии, что строки, столбцы, байты и биты в байте нумеруются с 1, используются формулы 4 – 6.

При нумерации строк, столбцов, байт и бит в байте с 0 формулы 4 – 6 будут иметь следующий вид

$$N_{\text{байт}} = P_{M_{i,j}} \; div \; 8 \qquad (7)$$

$$\text{N}_{\text{бит}} = P_{M_{i,j}} - N_{\text{байт}} \cdot 8 \qquad (8)$$

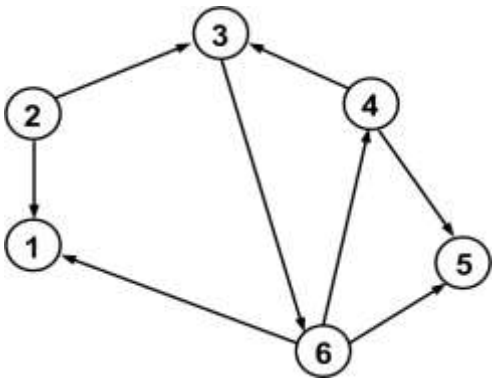
$$N_{\text{бит}} = P_{M_{i,j}} \mod 8 \qquad (9)$$

Предлагаемая выше модель имеет свои недостатки. Так, при условии, когда *n* не кратно 8, построчная информация матрицы смежности будет занимать в одномерном битовом векторе не целое число байт, в результате чего происходит смещение начала строки. В одном байте будет *находиться* конец одной строки и начало другой, строки будут начинаться с разных бит в байте и занимать разное число байт.

В качестве примера рассмотрим ориентированный граф с шестью вершинами (см. рисунок 7).

Представим граф матрицей смежности А.

Обозначим *n* – число вершин в графе. *n = 6*




A =

| | 1 | 2 | 3 | 4 | 5 | 6 | |
|---|---|---|---|---|---|---|---|
| | | | | | | | 1 |
| | 1 | | 1 | | | | 2 |
| | | | | | | 1 | 3 |
| | | | 1 | | 1 | | 4 |
| | | | | | | | 5 |
| | 1 | | | 1 | 1 | | 6 |

Рисунок 7 – Пример ориентированного графа

Одномерный битовый вектор будет записан так:

| *стр. 1* | | | | | | *стр. 2* | | *стр. 3* | | | | | | | | *стр. 4* | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** | | | |
| *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *1* | *2* | *3* | *4* | *5* |
| *1 байт* | | | | | | | | 2 байт | | | | | | | | 3 байт | | | | |

Для удобства представления и обработки матрицы в битовой форме применяют способ с выравниванием строки на границу байта или слова. Таким образом, представляем матрицу набором одномерных битовых векторов, где каждый вектор представляет одну строку матрицы и занимает целое число байт.

Если $n \le 8$, то на одну строку выделяется один байт. Если $8<n<16$, выделяется уже два байта и т.д.


При этом способе наглядно видно не только соответствие набора векторов матрице смежности, но и не максимально эффективное использование памяти, т.к. последние два бита в каждом байте не используются.

С увеличением размерности матрицы отношение неиспользуемой части памяти к выделенной стремится к нулю, а одинаковая позиция колонок в строках, и возможность выборки строки из матрицы как последовательности байт, существенно повышают эффективность обработки матрицы. Поэтому в САПР при битовом представлении матрицы используется выравнивание на границу элемента массива, представляющего матрицу: байта, слова (2 байта), двойного слова (4 байта).

Так как каждая строка располагается с начала элемента массива, то для расчета номера элемента в строке и номера бита в элементе используются формулы 4 – 9, где $P_{M_{i,j}}$ заменяется на $P_j$ ($j$ – номер столбца элемента), константа 8 используется для элемента байт, 16 – для слова, 32 – для двойного слова.

## 2.2. Операция анализа элемента матрицы в битовой форме

Наиболее простой способ выполнения операции анализа при условии, если матрица представлена в байтовой форме, т.е. каждый элемент занимает 1 байт. В этом случае для проверки наличия дуги достаточно сравнить значения соответствующего элемента массива с 0 или 1, например: *If M[i,j] =’1’ then оператор 1 (*Если элемент матрицы $M_{i,j}$ равен «1», то выполняет оператор 1).

Для установки значения элемента матрицы достаточно выполнить оператор присвоения для соответствующего элемента массива M[i,j ]:= ′0′.

В случае битового представления матрицы смежности в 1 байте хранятся значения нескольких элементов матрицы.

Компьютер не имеет операции обработки отдельных битов, поэтому для определения значения анализируемых битов необходимо выполнить логические операции, с помощью которых некоторая переменная имела бы различные значения в зависимости от значения анализируемого бита.

Также для установки битов соответствующих элементов матрицы необходимо использовать логические операции.

Рассмотрим анализ матрицы смежности, представленной в битовой форме на наличие дуги между двумя вершинами. Обратимся к графу на рисунке 7. Представим его набором одномерных векторов

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| *1 строка* | 0 | 0 | 0 | 0 | 0 | 0 | | |
| *2 строка* | 1 | 0 | 1 | 0 | 0 | 0 | | |
| *3 строка* | 0 | 0 | 0 | 0 | 0 | 1 | | |
| *4 строка* | 0 | 0 | 1 | 0 | 1 | 0 | | |
| *5 строка* | 0 | 0 | 0 | 0 | 0 | 0 | | |
| *6 строка* | 1 | 0 | 0 | 1 | 1 | 0 | | |

Проверим, действительно ли существует дуга между 2 и 3 вершинами.

Если связь есть – $M_{2,3} = 1$; если дуги нет, то $M_{2,3} = 0$

Для анализа элемента используется нулевой битовый вектор с единичными значениями в позиции анализируемого элемента (иногда его называют маской). При анализе выполняется операция «логическое И», или конъюнкция, а затем сравнение результата с нулевым вектором, т.е. нулем.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| *2 строка* | 1 | 0 | 1 | 0 | 0 | 0 | | |
| ***and*** | | | | | | | | |
| *маска* | 0 | 0 | 1 | 0 | 0 | 0 | | |
| ***R*** | 0 | 0 | 1 | 0 | 0 | 0 | | |

В векторе-маске позиция «1» совпадает с проверяемой позицией «1» в одномерном битовом векторе строки матрицы смежности, все остальные биты равны 0. В векторе ***R***, полученном в результате конъюнкции, проверяемая позиция равна значению анализируемого элемента, следовательно, дуга существует, если в векторе ***R*** имеется единичный бит, и дуги нет, если все биты нулевые.

*Если* $\boldsymbol{R} = 0$, *то* $M_{i,j} = 0$. *Если* $\boldsymbol{R} \neq 0$, *то* $M_{i,j} = 1$

## 2.3. Формирование матрицы

Задаемся, что из заданной вершины никуда переходов нет, т.е. обнуляем строку (присвоение 0).

Затем элементы матрицы, соответствующие дугам, необходимо установить в единичное значение. Для этого определяются байт и бит, в котором находится требуемый элемент.

Например, для установки дуги 4,5 в 4 байте необходимо установить 1 в 5-й бит. Для этого формируется нулевой битовый вектор (маска) с 1 в 5-м разряде и выполняется логическая операция «ИЛИ» («OR») *со значением 4-го байта* и результат записывается в 4-й байт.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| *маска* | 0 | 0 | 0 | 0 | **1** | 0 | | |
| ***or*** | | | | | | | | |
| *4 строка* | 0 | 0 | 1 | 0 | **0** | 0 | | |
| *4 строка* | 0 | 0 | 1 | 0 | **1** | 0 | | |

После выполнения таких же действий для дуги 4.3 получим 1 в 3-м бите.

Следует отметить, что задание графа с помощью матрицы смежности удобно и в том случае, когда граф – взвешенный и элементами матрицы являются не нули и единицы, а веса дуг.

Недостатком представления графа с помощью матрицы смежности является большой расход памяти при работе с графами, имеющими небольшое число дуг (матрица смежности при этом получается весьма разреженной).

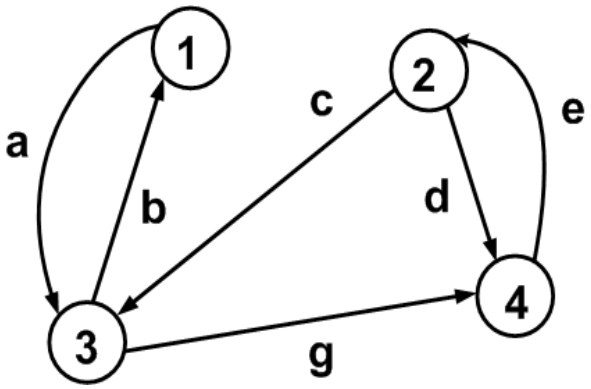
## 2.4. Представление с помощью матрицы инцидентности

Воспользуемся ориентированным графом, представленным на рисунке 6, и обозначим буквами его дуги (рисунок 8). Матрица инцидентности графа – прямоугольник размером $n \times m$, где $m$ и $n$ – число дуг и вершин соответственно.

Каждая строка матрицы инцидентности соответствует дуге представляемого графа, а столбцы его вершинам (рисунок 8).

Приняты обозначения:

- вершина не используется – (0);
- вершина исходящая, т.е. начало дуги – (1);
- вершина входящая, т.е. конец дуги – (-1).




M=

| | *1* | *2* | *3* | *4* | |
|---|---|---|---|---|---|
| | **1** | | **-1** | | *a* |
| | **-1** | | **1** | | *b* |
| | | **1** | **-1** | | *c* |
| | | **1** | | **-1** | *d* |
| | | **-1** | | **1** | *e* |
| | | | **1** | **-1** | *g* |

Рисунок 8 − Представление графа в виде матрицы инцидентности

Матрица инцидентности в памяти компьютера может представляться в байтовом виде (двумерный байтовый массив) или в битовой форме (битовые векторы). Требуемый объем оперативной памяти в этих случаях будет составлять

$$R_{\text{байт}} = n \times m \ \ (байт);$$

$$R_{\text{бит}} = n \times m \times 2 \ \ (бит).$$

В случае битовой формы, как и для матрицы смежности, используется представление строки битовым вектором с выравниванием по границе элемента базового массива. Формулы для расчета позиции элемента аналогичны формулам для матрицы смежности, но необходимо учитывать, что элемент матрицы занимает два бита.

Соответственно один элемент матрицы инцидентности может кодировать три вышеуказанные состояния взаимосвязи дуги и вершины. Для кодирования трех состояний необходимо два бита. Для примера примем следующую кодировку состояний для битовой формы:

$$элемент \Rightarrow \begin{cases} 0 \rightarrow 00 \\ -1 \rightarrow 01 \\ 1 \rightarrow 10 \end{cases}$$

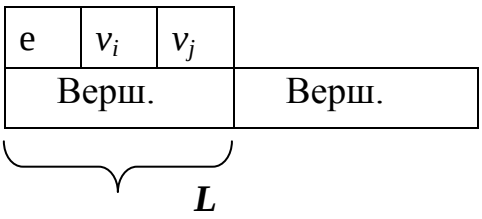
Представление матрицы инцидентности в виде двумерного массива или в текстовом файле позволяет использовать больше число вариантов кодирования элементов:

(0, 1, -1); (0, 1, 2); (‘ – ‘, ‘*и*’, ‘*в*’) и т. д.

Представление с помощью ***матрицы инцидентности*** (*или инциденций*) определяет граф однозначно, но применяется крайне редко в силу большой разреженности матрицы и практического отсутствия алгоритмов, работающих на такой структуре данных.

## 2.5. Представление с помощью списка дуг

При этом способе для каждой дуги рассматривается три параметра ($e$, $v_i$, $v_j$), где $e$ – дуга, $v_i$ – ее начало, $v_j$ – ее конец.

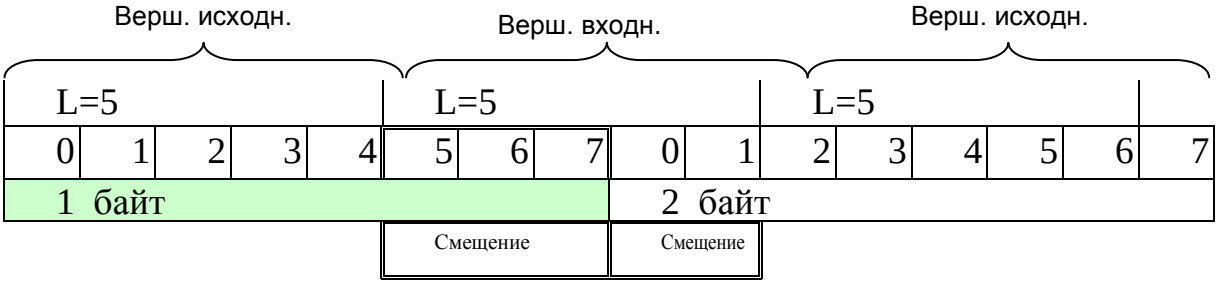



Размер поля кодирования $L$ вычисляется по формуле

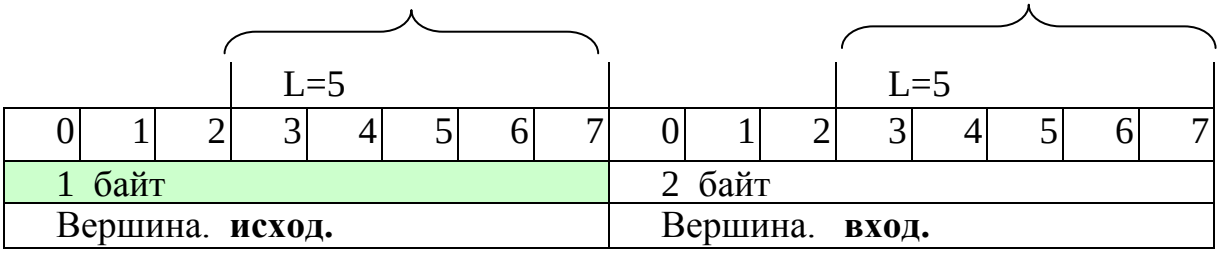
$$L = log_2 n \qquad (бит),$$

где $n$ – число вершин (максимальный код вершины), (полученная величина $L$ округляется до большего целого).

Так, например, при $n$ =30 $L$=5




Чтобы избежать смещения, поступают следующим образом:




При $L \leq 8$ под информацию отводят целый байт и занимают, начиная с последнего бита. Такая процедура называется **выравниванием границ байта,** т.е. информация заполняется, выравниваясь по правой границе байта, что соответствует представлению целых чисел в компьютере.

При больших значениях L соответственно выделяется 2, 3 и т.д. байт, обязательно выравнивается по правой границе байта.

Вернемся к нашему графу (пусть вершин будет больше), как изображено на рисунке 9.

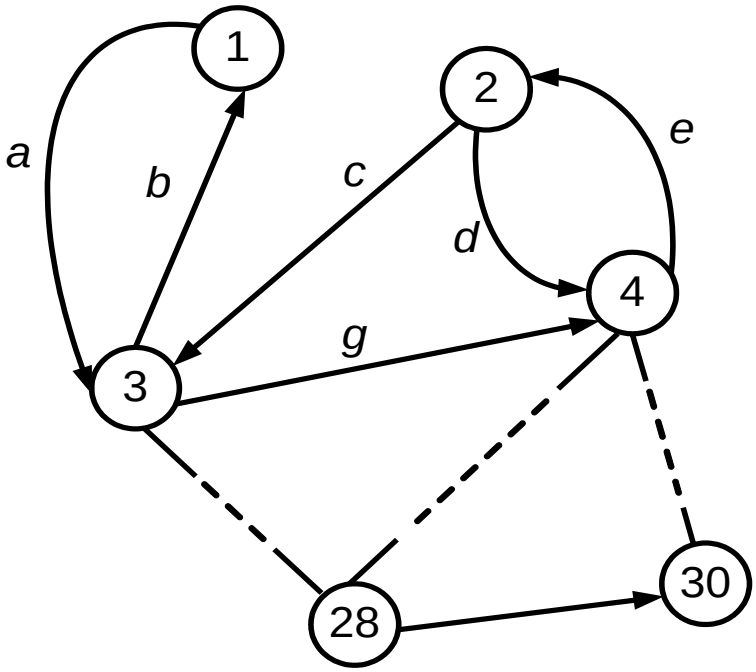



Рисунок 9 – Пример графа

Список дуг имеет вид:

| Исх. верш. | Вход. верш. |
|---|---|
| 1 | 3 |
| 3 | 1 |
| … | … |

Представим кодирование дуги в упакованном формате

| Исходн. вер. =1 | | | | | Входн. верш. =3 | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | **0** | **0** | **0** | **1** | **0** | **0** | **0** | **1** | **1** | | | | | | |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 1 байт | | | | | | | | 2 байт | | | | | | | |

При выравнивании значения на границу байта номер дуги будет представлен целым байтом

| Исходн. вер. =1 | | | | | | | |
|---|---|---|---|---|---|---|---|
| **0** | **0** | **0** | **0** | **0** | **0** | **0** | **1** |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

| Входн. верш. =3 | | | | | | | |
|---|---|---|---|---|---|---|---|
| **0** | **0** | **0** | **0** | **0** | **0** | **1** | **1** |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

Элементы, описывающие дуги, выстраиваются в таблицу или двумерным массивом

| Исход. | Вход |
|---|---|
| 1 | 3 |
| 3 | 1 |
| 2 | 3 |
| …… | …… |
| 1 байт | 2 байт |

**Подводя итог выше сказанному, можно сделать вывод:**

− Различные способы представления графов требуют различных объемов оперативной памяти и разного времени обработки графа. При этом в зависимости от самого графа эффективней может быть та или иная модель. Например, для графа с малым числом вершин наиболее эффективной является список дуг, а для графа с большим числом дуг – матрица смежности.

− Фактором, влияющим на выбор модели, является множество выполняемых операций с моделью, их трудоемкость для разных форм представления графа.

− Во многих случаях наиболее эффективным (по критерию минимального объема оперативной памяти) способом кодирования графа является битовая форма представления информации, но в этом случае значительно усложняется реализация операции над данными, что приводит к увеличению времени обработки данных.

− В зависимости от объема представления данных, требования к размеру оперативной памяти, скорости решения задачи выбирают ту или иную модель и форму представления данных (описания графа).

# 3. ПРАКТИЧЕСКОЕ ЗАНЯТИЕ. СПОСОБЫ ПРЕДСТАВЛЕНИЯ ГРАФОВЫХ МОДЕЛЕЙ В ПАМЯТИ ЭВМ. ОПЕРАЦИИ ОБРАБОТКИ ГРАФОВ

**Цель:**

- изучить способы хранения ориентированных графов в программах;
- определить структуры данных для выполнения лабораторной работы «Разработка технологических операций обработки графов».
- научиться разрабатывать процедуры для выполнения простейших операции над графами с использованием средств объектного программирования;

**Порядок проведения занятия:**

1. Представить графическое изображение ориентированного графа.
2. Рассмотреть способы описания графа в программе с помощью: матрицы смежности и инцидентности (байтовая и битовая формы представления), списка дуг.
3. Выполнить кодировку матрицы смежности и матрицы инцидентности для заданного графа.
4. Разработать процедуры для выполнения операций над графами (удаление вершины или дуги, добавление вершины или дуги) для различных форм представления графа.

# 4. ЛАБОРАТОРНАЯ РАБОТА. РАЗРАБОТКА ТЕХНОЛОГИЧЕСКИХ ОПЕРАЦИЙ ОБРАБОТКИ ГРАФОВ

**Цель работы:**

- изучить основные способы представления и обработки графов,
- приобрести навыки разработки процедур обработки графовых моделей.

**Задание на лабораторную работу:** Разработать процедуры, выполняющие ввод и преобразование данных, описывающих граф и вывод полученного результата преобразования. Создать программу, которая, загружая данные из файла, производила бы их преобразование из исходной формы в заданную, и вывод результатов преобразования в файл. При этом на этапах ввода, преобразования и вывода данных должен осуществляться контроль корректности информации и производиться обработка ошибок. В оперативной памяти (ОП) граф должен быть представлен в битовой форме. Ограничение: максимальный размер графа – 16 вершин.

Структура разрабатываемой программы изображена на рисунке 10.

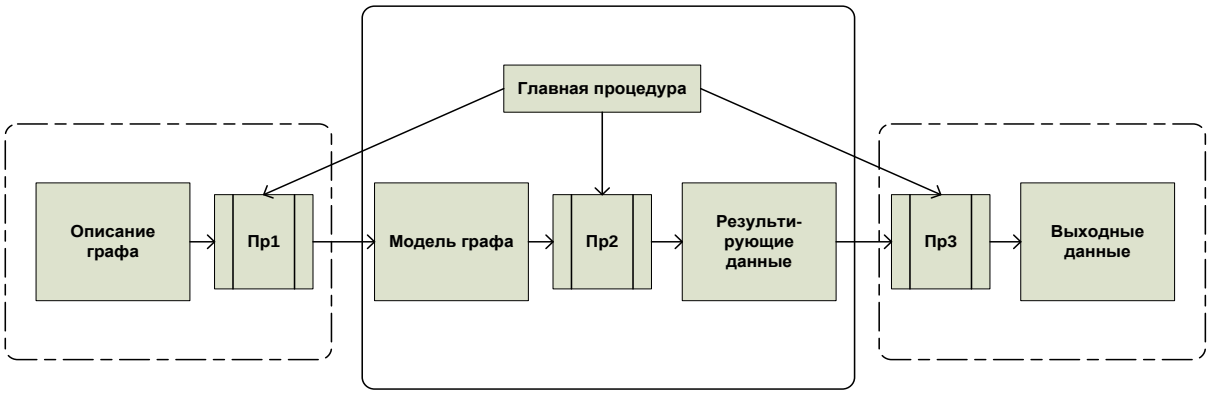



Рисунок 10 – Структурная схема программы

Программа состоит из четырех процедур:

- Главная процедура, предназначена для обеспечения функционирования процедур обработки графа и взаимодействия с пользователем;
- Процедура 1 (Пр1), предназначена для ввода описания графа из файла исходных данных и формирования модели графа в ОП;
- Процедура 2 (Пр2), предназначена для реализации функции обработки графа и формирования результирующих данных;
- Процедура 3 (Пр3), предназначена для вывода результирующих данных в текстовый файл.

Описание графа и выходные данные представляются в текстовых файлах, а модель графа и результирующие данные в ОП.

Программа разрабатывается в составе бригады из двух студентов:

- Первый определяет структуры данных для представления модели графа и результирующих данных. Разрабатывает главную процедуру и процедуру 2;
- Второй определяет форматы представления описания графа и выходных данных. Разрабатывает процедуры 1 и 3.

При выполнении лабораторной работы необходимо провести отладку и тестирование программы.

В таблице 1 приняты следующие обозначения:

МС – матрица смежности;
МИ – матрица инцидентности.

Таблица 1 – Варианты заданий

| № | Способ представления описания графа | Способ представления модели графа | Способ представления результирующих и выходных данных |
|---|---|---|---|
| 1 | МИ | МИ | МС |
| 2 | МС | МС | МИ |
| 3 | МИ | МИ | Получить список четных вершин, имеющих связь только с нечетными вершинами |
| 4 | МС | МС | Получить список нечетных вершин, имеющих связь только с четными вершинами |
| 5 | Список дуг | МИ | Получить список вершин, у которых число входов меньше, чем число выходов |
| 6 | Список дуг | МС | Получить список вершин, у которых число входов больше, чем число выходов |
| 7 | МИ | МИ | Список дуг |
| 8 | МС | МС | Список дуг |
| 9 | МИ | МИ | Список вершин, у которых число дуг больше заданного N |
| 10 | МС | МС | Список вершин, у которых число дуг меньше заданного N |
| 11 | Список дуг | МИ | Получить список вершин, у которых число входов равно числу выходов |
| 12 | Список дуг | МС | Получить список вершин, у которых четное число входов |

## Требования к выполнению лабораторной работы

При выполнении лабораторной работы необходимо выполнить требования, предъявляемые к процессу разработки САПР и его составных частей. К таким требованиям относятся:

- модульный принцип построения программных систем (ПС), то есть выделение в ПС фрагментов, предназначенных для решения некоторой задачи или преобразования и оформление их в виде процедур или функций;
- единство данных – определение множества данных функционально-полно описывающих некоторый объект и объединение их в один класс или запись;
- все входные данные в процедуру или функцию, так же как и выходные данные из процедуры или функции передаются только через параметры, без использования механизма внешних переменных (глобальных переменных);
- процедуры и функции, для которых существуют ситуации, не позволяющие выполнить требуемое действие, должны возвращать признак завершения операции, указывающий на нормальную реализацию действия или на причину его невыполнения.

Все указанные требования должны быть учтены при разработке программной части лабораторной работы.

## Контрольные вопросы

1. Дать определение графа.
2. Объяснить разницу между ориентированным и неориентированным графом.
3. Представить граф с помощью:
   а) матрицы смежности;
   б) матрицы инцидентности.
4. Привести формулу расчета объема оперативной памяти для представления графа, состоящего из $n$ вершин и $m$ дуг:
   а) с помощью матрицы смежности;
   б) с помощью матрицы инцидентности;
   в) с помощью списка дуг.

## Содержание отчета

1. Постановка задачи.
2. Краткое описание способа решения задачи.
3. Описание структур данных.
4. Спецификация на процедуры и функции.
5. Описание алгоритмов решения задачи.
6. Описание процедур и функций.
7. Текст программы.
8. Результаты тестирования.
9. Заключение (выводы).
10. Приложение.

# БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Евстигнеев В.А. Применение теории графов в программировании / В.А. Евстигнеев – М.: Наука, 1991. – 318 с.
2. Норенков И.П. Основы автоматизированного проектирования: Учеб. для вузов / И.П. Норенков – М.: Изд-во МГУ им. Н.Э.Баумана, 2002. – 336 с.
3. Сигорский В.П. Математический аппарат инженера / В.П. Сигорский – Киев: Техніка, 1997. – 462 с.

ДЛЯ ЗАМЕТОК

Заказ №_____от «___»_____2014 Тираж_____экз.
Изд-во СевНТУ