|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии

**ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №7**

**ГРАФЫ**

Название предмета: Типы и структуры данных

Студент: Варламова Екатерина Алексеевнa

Группа: ИУ7-31Б

*2020г.*

1. **Описание условия задачи**

Найти все вершины графа, к которым от заданной вершины можно добраться по пути не длиннее А.

1. **Техническое задание**
2. ***Описание исходных данных***

Программа ожидает:

- название файла с количеством ребер и их перечислением (читается до первого неверного ребра)

- максимальная длина пути

- вершина, из которой осуществляется поиск пути

Формат ввода:

Формат файла:

<количество вершин>

<вершина-источник> <вершина место назначения> <вес ребра>

…

<вершина-источник> <вершина место назначения> <вес ребра>

Все числа являются целыми. При первой ошибке ввод заканчивается и начинается обработка правильно введённых вершин. Вершины нумеруются с единицы.

1. ***Описание результата программы***

Результатом работы программы являются:

1. Граф в графическом виде
2. Массив расстояний во все вершины от заданной (с указанием, подходит ли расстояние под условие задачи)

Формат вывода:

1. Массив расстояний

Вид:

from <num of vertex> to <num of vertex> distance is <distance>

* если не достижима, то inf
* указание (<- less than or equal to max distance)

Пример:

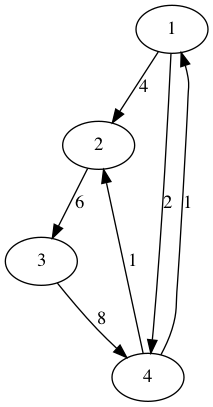
from 1 to 1 distance is 0 <- less than or equal to max distance

from 1 to 2 distance is 3 <- less than or equal to max distance

from 1 to 3 distance is inf

from 1 to 4 distance is 2 <- less than or equal to max distance

1. Вывод графа



1. ***Описание задачи, реализуемой программой***

Программа находит все вершины графа, к которым от заданной вершины можно добраться по пути не длиннее А.

1. ***Способ обращения к программе***

Способ обращения к программе - консольный. Дальнейшие инструкции будут выведены после запуска.

1. ***Описание возможных аварийных ситуаций и ошибок пользователя***

* ошибка в названии файла или его отсутствие (физическое и/или в аргументах командной строки);
* отказ операционной системы выделить память

Во всех указанных случаях программа сообщит об ошибке

1. **Описание внутренних структур данных**

В программе есть 1 основная структура данных:

**typedef struct**

**{**

**int src, dst, cost;**

**} elem\_t;**

**typedef struct**

**{**

**elem\_t \*\*front;**

**size\_t size;**

**size\_t alloc\_size;**

**} vector\_t;**

То есть граф хранится в виде динамического массива рёбер. Такое представление наиболее эффективно по времени и по памяти по сравнению, например, с таблицей смежности, ведь обычно граф не является полным, а значит таблица смежности стала бы сильно разреженной.

1. **Описание алгоритма**

Воспользуемся алгоритмом Форда-Беллмана для поиска кратчайших путей из заданной вершины. Выбор обоснован тем, что вес рёбер может быть в том числе отрицательным, при этом, например, в алгоритме Дейкстры это не разрешено. Кроме того, алгоритм обнаруживает отрицательный цикл. Ниже описан подробный алгоритм:

Пусть массив кратчайших путей d, вершина, из которой осуществляется поиск путей v. Количество вершин n.

Изначально: d[v] = 0; d[i] = inf, для любого 0 < i < n, i v

1. Осуществляется проход по всем рёбрам: для каждого ребра src => dst алгоритм пытается улучшить значение d[dst] значением d[src] + c, где c – стоимость прохода из src в dst.
2. Предыдущий шаг повторяется n – 1 раз (доказывается, что столько раз достаточно).

Замечание: на самом деле, в программе проход осуществляется n раз, для того чтобы обнаружить отрицательный цикл. Очевидно, что если за n-ый проход хотя бы одно значение в массиве d улучшится, то в графе есть отрицательный цикл.

***Оценка по времени***

Исходя из описания алгоритма, мы можем сделать вывод, что сложность алгоритма составляет: O(m \* n).

***Оценка по памяти***

Память в данном алгоритме затрачивается только на хранение рёбер в виде 3 целых чисел и на хранение массива с ответом.

1. **Выводы по проделанной работе**

Для решения поставленной задачи был реализован алгоритм Форда-Беллмана для поиска кратчайших путей в графе. Выбор обоснован тем, что вес рёбер может быть в том числе отрицательным, при этом, например, в алгоритме Дейкстры это не разрешено. Кроме того, алгоритм обнаруживает отрицательный цикл. Этот алгоритм может быть использован, например, в системе навигации для таксистов: вершина, из которой осуществляется поиск представляет собой местоположение водителя, а остальные вершины - местоположения клиентов и промежуточные пункты, тогда программа строит таблицу кратчайших путей до клиентов и выдаёт водителю (ребрами служат реальные дороги).

1. **Ответы на вопросы**
2. Что такое граф?

Граф - это конечное множество вершин и ребер, соединяющих их.

1. Как представляются графы в памяти?

Графы могут быть представлены в виде таблицы смежности, списков достижимых вершин из каждой вершины, массива ребёр.

1. Какие операции возможны над графами?

- поиск вершин в графе

- поиск кратчайших путей от Vk до Vm

- поиск Эйлерова пути

- поиск Гамильтонова пути

- поиск кратчайших путей между всеми вершинами

1. Какие способы обхода графов существуют?

Поиск (обход) в глубину, в ширину.

1. Где используются графовые структуры?

Схемы авиалиний, схемы дорог.

1. Какие пути в графе Вы знаете?

Простой путь, гамильтонов, эйлеров, замкнутый.

1. Что такое каркасы графа?

Каркас графа - подграф данного графа, с тем же числом вершин, что и у исходного графа. Он получается из исходного графа удалением максимального числа рёбер, входящих в циклы, но без нарушения связности графа.