Отчёт по лабораторной работе №7

“Обработка графов”

Вариант 12

Выполнил: Соколов Ефим Маркович (группа ИУ7-33Б)

### ***Цель работы:*** реализовать алгоритмы обработки графовых структур: поиск различных путей, проверка связности, построение остовых деревьев минимальной стоимости.

Техническое задание

Задана система двусторонних дорог. Найти множество городов, расстояние от которых до выделенного города ( столицы ) больше, чем Т.

Типы входных и выходных данных

***Входные данные:***

* Файл с расширением *.gv,* в котором хранится информация об исходном графе (вершины и ребра)
* Название “столицы” (строка)
* Расстояние T (целое неотрицательно число)

**Выходные данные:**

* Список городов, расстояние до которых от “столицы” больше T
* Графическая визуализация этих расстояний

***Возможные аварийные ситуации:***

* Некорректный ввод данных (пустой файл, файл не существует, несуществующая “столица”, некорректное число T)
* Городов, расстояние до которых от “столицы” больше T, не существует

Структуры данных

Для представления неориентированного графа и решения поставленной задачи используются следующие структуры данных:

#define MAX\_LEN 100 // Максимальное количество вершин в графе

int adjacency\_matrix[MAX\_LEN][MAX\_LEN]; // Матрица смежности

int path\_matrix[MAX\_LEN][MAX\_LEN]; // Матрица путей

int distance\_matrix[MAX\_LEN][MAX\_LEN]; // Матрица расстояний

Алгоритм

Из исходного файла загружается информация об исходном графе (заполняется матрица смежности *int adjacency\_matrix[MAX\_LEN][MAX\_LEN]*). Для решения поставленной задачи, формируется матрица расстояний (*int distance\_matrix[MAX\_LEN][MAX\_LEN]*) – матрица, в каждой ячейке[i, j] которой хранится расстояние между вершинами i-j – используя алгоритм Флойда-Уоршелла (сложность O(n^3)). Затем, проходя по всей матрице расстояний, отсеиваются вершины (города), расстояние до столицы от которых больше T. В конце в консоль печатается список городов, удовлетворяющих условию, и строится графическая визуализация результата, используя графический фреймворк *GraphViz.* Графическая визуализация рисуется на основе матрицы путей (*int path\_matrix[MAX\_LEN][MAX\_LEN]*), благодаря которой учитываются промежуточные вершины по пути в очередной город, удовлетворяющий условию.

Тесты

|  |  |  |  |
| --- | --- | --- | --- |
|  | Тест\* | Пользовательский ввод | Результат |
| 1 | Файл не существует |  | Сообщение “Файл не найден!” |
| 2 | Пустой файл |  | Сообщение “Пустой файл!” |
| 3 | Столица не существует | Khabarovsk | Сообщение “Такого города не существует” |
| 4 | Расстояние T отрицательно | Murmansk  -3000 | Сообщение “Введено некорректное число T!” |
| 5 | Нет городов на расстоянии > T от столицы | Ufa  12000 | Сообщение “Городов, удовлетворяющих условию нет!” |
| 6 | Тривиальный тест | Moscow  3000 | Moscow --- Novosibirsk [3361km]  Moscow --- Vladivostok [10350km]  + графическая интерпретация результата |

\* Для тестов 3-6 используется файл следующего содержания:

Moscow -- Saint\_Petersburg [702];

Moscow -- Novosibirsk [3380];

Moscow -- Krasnodar [1379];

Moscow -- Ufa [1369];

Saint\_Petersburg -- Murmansk [1347];

Saint\_Petersburg -- Krasnodar [2063];

Saint\_Petersburg -- Ufa [2057];

Murmansk -- Krasnodar [3293];

Saint\_Petersburg -- Vladivostok [9648];

Ufa -- Novosibirsk [1992];

Оценка эффективности

**Время выполнения:**

|  |  |
| --- | --- |
| Количество элементов | Время выполнения (т.п.\*) |
| 5 | 98304 |
| 10 | 206869 |
| 15 | 314472 |

\*такты процессора

**Занимаемая память:**

|  |  |
| --- | --- |
| Количество элементов | Занимаемая память (кбайт) |
| 5 | 30 |
| 10 | 30 |
| 15 | 30 |

Вывод

Реализация задачи состоит из двух этапов — составление матрицы расстояний (алгоритм Флойда-Уоршелла) и полный перебор (поиск вершин, удовлетворяющих условию). В худшем случае алгоритм Флойда-Уоршелла имеет асимптотическую сложность O(V^3), где V – число вершин графа, а полный перебор вершин – O(V), таким образом общая сложность всего алгоритма, будет определяться сложностью алгоритма Флойда-Уоршелла - O(V^3).

Контрольные вопросы

1. **Что такое граф?**

Граф – конечное множество вершин и соединяющих их ребер (**G = <V, E>)**. Если ребро имеет вес, то граф называется взвешенным; если пары **Е** (ребра графа) имеют направление, то граф называется ориентированным.

1. **Как представляются графы в памяти?**

С помощью матрицы смежности или списков смежности.

1. **Какие операции возможны над графами?**

Обход вершин, исключение и включение вершин; поиск различных путей.

1. **Какие способы обхода графов существуют?**

Обход в глубину **(DFS – Depth First Search)** и обход в ширину **(BFS – Breadth First Search).**

1. **Где используются графовые структуры?**

Графовые структуры могут использоваться в задачах, в которых между элементами могут быть установлены произвольные связи.

1. **Какие пути в графе Вы знаете?**

Простой путь, сложный путь, эйлеров путь, гамильтонов путь.

1. **Что такое каркасы графа?**

Каркас графа – дерево, в которое входят все вершины графа, и некоторые (необязательно все) его рёбра.