|  |  |
| --- | --- |
|  | **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»

**ОТЧЕТ К ЛАБОРАТОРНОЙ РАБОТЕ №8 ПО «ТИПАМ И СТРУКТУРАМ ДАННЫХ»**

**Тема: «Графы»**

Группа: ИУ7-32Б

|  |  |  |
| --- | --- | --- |
| Студент: |  | Сидоров Максим Михайлович |

Преподаватель: Никульшина Татьяна Александровна

**Цель работы:** научиться работать с графовыми структурами.

**Условие задачи:** проверить связность графа. Найти все вершины графа, к которым от заданной вершины можно добраться по пути не длиннее А.

**Входные данные:** номер команды из меню, текстовый файл и терминальный ввод.

Команды:

1. Найти вершины, в которые можно добраться из данной
2. Выйти

**Выходные данные:**

- файл с графом в формате DOT

- анализ связности

- вершины, в которые можно попасть

**Аварийные ситуации:**

- ввод неправильной команды

- некорректный файл

**Описание СД:**

Граф хранится в матрице смежности и было выбрано следующее представление:

typedef struct

{

int \*\*mat;

int n;

} matrix;

*Листинг 1. Представление матицы смежности.*

**Алгоритм поиска в графе:** для поиска вершин я использовал поиск в глубину. Прекращение поиска – приход в вершину, в котором уже побывал или длина пути превысила максимально допустимый.

**Алгоритм проверки связности графа:** поиск в ширину. Прекращение поиска – приход в узел, в котором уже побывали. Если не из каждого узла можно попасть в каждый, то граф несвязный. Иначе – связный.

**Тестовые данные:**

|  |  |  |
| --- | --- | --- |
| **Описание** | **Вход** | **Выход** |
| Некорректная команда | А, -1, 15 | Некорректная команда |
| Запуск программы с некорректным файлом | Dsfsj.chh | Некорректный файл |
| Номер некорректной вершины | -2 | Некорректная вершина |
| Некорректная длина пути | -2 | Некорректная длина |
| Поиск вершин, куда можно попасть из текущей | - | Перечисление вершин, в которые можно попасть |

**Функции:**

int read\_table(matrix \*a, const char \*file\_name) – считывание таблицы из файла

int \*\*alloc\_matrix(int n, int m) – выделение памяти под матрицу

void free\_matrix(matrix \*a) – освобождение памяти из-под матрицы

void check\_connection(matrix \*con, matrix \*a, int from, int cur) – проверка связности для вершины

int check(matrix \*a) – проверка связности графа

void find(matrix \*a, int \*points, int p, int cur\_len, int max\_len) – поиск связанных с данной вершиной вершин

void find\_points(matrix \*a, int point, int max\_len) – функция-обертка для поиска вершин

**Вывод:** сложность алгоритма поиска: O(V + E), где V – количество вершин, E – количество ребер.

Реальная задача: навигатор, поиск пути из указанной точки до ближайшего ресторана, заправки и т.д.

Матрица смежности: использует О(V^2) памяти; быстрые поиск наличия ребра между вершинами O(1); медленный перебор всех ребер O(V^2); быстрое добавление нового ребра О(1); добавление или удаление вершины происходит медленно O(V^2).

Список смежности: объем памяти зависит от количества ребер и узлов O(V+E); добавление нового ребра О(1); добавление и удаление узла происходит максимум за O(2k), где k – количество соседних вершин; быстрый перебор элементов; так как можно напрямую обратиться к соседним узлам; поиск наличия ребра между вершинами происходит за O(k), где k – количество соседних вершин.

**Контрольные вопросы:**

1. **Что такое граф?**

Граф – конечное множество вершин и ребер, соединяющих их.

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

Могут быть представлены в виде матрицы смежности, где ячейка a[i][j]=0 указывает об отсутствии пути из i в j, иначе о существовании пути.

Могут быть представлены в виде списка смежностей, в котором для каждой вершины есть список тех вершин, с которыми он связан.

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

Обход вершин, исключение вершин, включение вершин, поиск путей.

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

- обход в глубину

- обход в ширину

1. **Где используются графовые структуры?**

В базах данных, при оптимизации и моделировании маршрутов, поиск информации и т.д.

То есть в задачах, где между элементами могут быть установлены произвольные связи, необязательно иерархические.

1. **Какие пути в графе Вы знаете?**

Эйлеров путь, непростой путь, гамильтонов путь.

1. **Что такое каркасы графа?**

Каркас графа – дерево, в которое входят все вершины графа и некоторые его ребра.