МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по лабораторной работе №2
по дисциплине «Построение и анализ алгоритмов»
Тема: Алгоритм A*

Студент гр. 7383 _____ Васильев А.И.

Преподаватель _____ Жангиров Т.Р.

Санкт-Петербург
2019

Цель работы.

Исследовать и реализовывать задачу построения кратчайшего пути в ориентированном графе, используя алгоритм A*.

Формулировка задачи: необходимо разработать программу, которая решает задачу построения кратчайшего пути в ориентированном графе методом A* до любой из представленных вершин. Каждая вершина в графе имеет буквенное обозначение («a», «b», «c» …), каждое ребро имеет неотрицательный вес.

Вариант 2м: граф представлен в виде матрицы смежности, эвристическая функция для каждой вершины задаётся положительным числом во входных данных.

Входные данные: в первой строке указываются начальная и конечная вершины. Затем количество вершин и их эвристические значения. Далее идут данные о рёбрах графа и их весе. Выходные данные: кратчайший путь из стартовой вершины в конечную.

Реализация задачи.

В работе был использован класс `Astar` и структуры данных.

Параметры, хранящиеся в классе:

`graph` – двумерный массив, в котором хранится граф в виде матрицы смежностей;

`result` – строка результата;

`str` – строка для хранения пути до текущей вершины.

Методы класса:

`insert_paths` – функция заполнения матрицы весами рёбер;

`print` – функция вывода результата на консоль;

`algorithm` – функция поиска кратчайшего пути методом A*.

Класс `Astar` отвечает за хранение графа в виде матрицы смежности, а так же содержит функцию `algorithm`, которая осуществляет поиск кратчайшего пути из начальной вершины в конечную, используя A*.

Структура `Priority`:

Параметры структуры:

`path` – путь до данной вершины;

`prior_vertex` – значение эвристической функции, по которому строится приоритетная очередь;

`characteristic` – эвристика вершины.

Данная структура используется для хранения данных в очереди с приоритетами.

Параметры, передаваемые в `bool operator < (const Priority &comp_var_1, const Priority &comp_var_2)` являются элементами структуры. Данная функция используется для того, чтобы в приоритетной очереди сначала шли элементы с меньшим значением эвристической функции.

В функции `main()` считываются начальная и конечная вершины, между которыми нужно найти кратчайший путь. Затем записываются в массив значения эвристики для каждой из вершин графа. Далее в цикле начинается считывание рёбер графа и их вес. Запускается алгоритм A*, который рассматривает всех соседей текущей вершины, помещает пути до них в очередь, в зависимости от значения эвристической функции, и находит кратчайший путь для вершины, которая на данном шаге находится ближе. Пока очередь не пуста алгоритм выполняет свою работу.

Тестирование.

Тестовые случаи не выявили не правильного поведения программы, следовательно можно судить о том, что поставленная задача была выполнена. Тестовые случаи приведены в табл. 1.

Таблица 1 – Тестовые случаи

Входные данные	Вывод	Верно?
a e 5 a 3 b 1 c 9 d 1 e 4 a b 3 b c 1 c d 1 a d 5 d e 1	ade	Да
b g 5 b 3 c 1 d 7 e 2 g 8 b c 3 c d 1 d e 1 b e 5 e g 1	beg	Да

Исследование.

Сложность алгоритма составляет O(|V|•|E|), где |V| – количество вершин в графе, |E| – количество ребер в графе. На каждом шаге работы программы просматриваются всевозможные пути из искомой вершины. В худшем случае могут быть просмотрены все пути данного графа. Тогда сложность зависит от количества вершин.

Выводы.

В ходе лабораторной работы был изучен алгоритм поиска кратчайшего пути A*. Был написан код на языке программирования C++, который применял этот метод для поставленной задачи. Сложность реализованного алгоритма составляет O(|V|•|E|).