Министерство образования и науки Российской Федерации

Севастопольский государственный университет

Кафедра ИС

Отчет

по лабораторной работе № 5

«Поиск кратчайших путей на графах»

Выполнил: ст. гр. И12д

Серегин А.В.

Проверил:

ст.пр. Шишкевич В.Е.

Севастополь

2015

1. Цель работы

Целью работы является изучение алгоритмов поиска кратчайших путей на графах на примере метода динамического программирования.

2. Варианты задания

Сравнить алгоритмы поиска кратчайшего пути в графе, а именно алгоритм Дейкстры и метод динамического программирования.

3. Текст программы

#include <iostream>

#include <algorithm>

#include <vector>

#include <Windows.h>

using namespace std;

void msInit(vector<vector<int>>& g, const int n)

{

g.clear();

g.resize(n);

for (int i(0); i < n; i++)

g[i].resize(n);

}

void pushArc(vector<vector<int>>& g, const int first, const int second, const int weight)

{

g[first][second] = weight;

}

void dfs(const vector<vector<int>> g, vector<int>& top, vector<bool>& used, const int v, const int n)

{

if (used[v])

return;

used[v] = true;

for (int to = 0; to < n; to++)

if (g[v][to])

dfs(g, top,used, to, n);

top.push\_back(v);

}

void topSort(const vector<vector<int>> g, vector<int>& top, const int n)

{

vector<bool> used(n, false);

for (int i = 0; i < n; i++)

dfs(g, top, used, i, n);

reverse(top.begin(), top.end());

}

int solve(const vector<vector<int>> g, const int n, const int stPoint, const int endPoint)

{

vector<int> d(n, INT\_MAX);

vector<int> top;

topSort(g, top, n);

d[stPoint] = 0;

for (int i = 1; i < n; i++)

for (int j = 0; j < i; j++)

if (g[top[j]][top[i]])

d[top[i]] = min(d[top[i]], d[top[j]] + g[top[j]][top[i]]);

return d[endPoint];

}

int Deixtra(const vector<vector<int>> g, const int n, const int stPoint, const int endPoint) {

vector<int> d(n, INT\_MAX);

vector<bool> used(n, false);

int v;

d[stPoint] = 0;

for (int i = 0; i < n; i++)

{

v = -1;

for (int j = 0; j < n; j++)

if (!used[j] && (v == -1 || d[j] < d[v]))

v = j;

used[v] = true;

for (int to = 0; to < n; to++)

{

if (g[v][to])

{

int len = g[v][to];

if (d[v] + len < d[to])

d[to] = d[v] + len;

}

}

}

return d[endPoint];

}

int main()

{

setlocale(LC\_ALL, "Russian");

vector<vector<int>> g;

int stPoint;

int endPoint;

ULONGLONG dpTime;

ULONGLONG deixtraTime;

int n;

cout << "Введите кол-во узлов:" << endl;

cin >> n;

msInit(g, n);

{

cout << "Введите кол-во дуг:" << endl;

int n,first,second,weight;

cin >> n;

cout << "Введиете дуги:" << endl;

for (int i(0); i < n; i++)

{

cin >> first >> second >> weight;

pushArc(g, first, second, weight);

}

}

cout << "Введите начальную и конечные точки:" << endl;

cin >> stPoint >> endPoint;

cout << "Кратчайший путь S = " << Deixtra(g, n, stPoint, endPoint) << endl;

dpTime = GetTickCount64();

for (int i(0); i < 1000; i++)

solve(g, n, 0, n-1);

dpTime = GetTickCount64()-dpTime;

deixtraTime = GetTickCount64();

for (int i(0); i < 1000; i++)

Deixtra(g, n, 0, n-1);

deixtraTime = GetTickCount64() - deixtraTime;

char\* str = nullptr;

if (deixtraTime < dpTime) str = "Дейкстры";

else str = "динамического программирования";

cout << "При 1000 прогонов алгоритм " << str << " оказался быстрее на " << abs(long(dpTime) - long(deixtraTime)) << "ms." << endl;

system("pause");

return 0;

}

4. Результаты работы

На рисунке ниже представлены результаты работы программы.

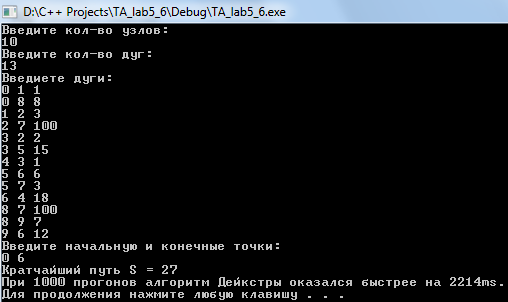


Рисунок 1 – Результат работы программы.

Выводы

В ходе лабораторной работы мы изучили алгоритм Дейкстры и метод динамического программирования для нахождения кратчайшего пути в графе. Опытным путем убедились что алгоритм Дейкстры работает быстрее.