Министерствонауки и высшего образования РФ

Севастопольский государственный университет

Кафедра информатики и управления в технических системах

ОТЧЕТ

по лабораторной работе №5

ПОИСК КРАТЧАЙШИХ ПУТЕЙ НА ГРАФАХ

по дисциплине «Основы теории алгоритмов»

Выполнил:

Студент группы ИВТ/б 22-о

Черняев Н.Г.

Проверил:

Абрамович А.Ю.

г. Севастополь 2019

**Цель работы**

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

**Вариант №4**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Куда | | | | | | | | |
| Откуда |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 1 |  | 1 | 10 |  |  |  |  |  |  |
| 2 |  |  | 3 | 3 |  |  | 7 |  |  |
| 3 |  |  |  |  | 2 | 7 |  | 5 |  |
| 4 |  |  | 2 |  | 4 |  |  |  |  |
| 5 |  |  |  |  |  |  | 3 |  |  |
| 6 |  |  |  |  | 2 |  |  |  |  |
| 7 |  |  |  |  |  | 5 |  |  | 7 |
| 8 |  |  |  |  |  |  |  |  | 7 |
| 9 |  |  |  |  |  |  |  |  | 7 |

1

10

3

3

7

2

7

5

2

4

3

2

5

7

7

Составить структурную схему программы, определяющей кратчайший путь на графе от заданной начальной вершины *s* до заданной конечной вершины *t* с помощью метода динамического программирования.

Составить структурную схему программы, реализующей алгоритм топологической сортировки с произвольной нумерацией вершин графа

Создать программу, реализующую метод динамического программирования и алгоритм топологической сортировки вершин. Исходный граф задается в виде матрицы смежности, вводимой построчно с помощью консоли.

Создать программу, которая использует приведенный в даннойработе алгоритм Дейкстры для заданного графа.

**Текст программы**

#include "pch.h"

#include <iostream>

#include <fstream>

#include <algorithm>

#include <time.h>

#include <chrono>

using namespace std;

const int INF = 1000000000;

bool used[9] = { 0 };

int top[9] = { 0 };

int g[9][9] = { 0 };

int n = 9;

int l;

int s = 0;

int f = 8;

int d[9] = { 0, INF, INF, INF, INF, INF, INF, INF, INF };

int dfs(int v) {

if (used[v])

return 0;

used[v] = true;

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

if (g[v][to])

dfs(to);

top[l++] = v;

}

int topSort() {

l = 0;

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

dfs(i);

reverse(top, top + l);

return 0;

}

int solve() {

int i, j;

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

d[i] = INF;

d[s] = 0;

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

for (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 0;

}

void solve\_d() {

int i, j,

v,

to,

len;

d[s] = 0;

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

v = -1;

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

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

v = j;

used[v] = true;

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

if (g[v][to]) {

len = g[v][to];

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

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

}

}

}

}

}

int main()

{

ifstream myFile("F:\\file.txt");

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

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

myFile >> g[i][j];

if (g[i][j] == -1) g[i][j] = INF;

}

}

auto start = chrono::steady\_clock::now();

//topSort();

//solve();

solve\_d();

auto end = chrono::steady\_clock::now();

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

cout << d[i] << " ";

}

cout << endl

<< "Time taken in microseconds: "

<< chrono::duration\_cast<chrono::microseconds>(end - start).count()

<< endl;

cout << endl

<< "Time taken in nanoseconds: "

<< chrono::duration\_cast<chrono::nanoseconds>(end - start).count()

<< endl;

}

**Результаты**

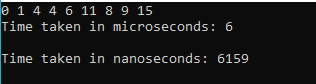


рис. 1 – пример работы программы, реализующей метод динамического программирования

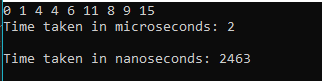


рис. 2 – пример работы программы, реализующей алгоритм Дейкстры

Как видно из результатов программы алгоритм Дейкстры находит решение в 3 раза быстрее, чем метод динамического программирования. Это означает, что алгоритм Дейкстры более эффективен.

**Вывод**

В данной лабораторной работе были изучены алгоритмы поиска кратчайших путей на графах на примере метода динамического программирования.