Министерство цифрового развития, связи

и массовых коммуникаций Российской Федерации

Сибирский Государственный Университет Телекоммуникаций и Информатики

СибГУТИ

Кафедра прикладной математики и кибернетики

РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА

Выполнил: студент гр. ИП-014

Михеев Д.М.

Преподаватель: Авдеев К.И.

Новосибирск 2022

Оглавление

Постановка задачи

Задание:

Нахождение кратчайшего расстояния (Форд-Беллман).

Формат входных данных:

Матрица весов, список ребер либо другой альтернативный вариант. В лабораторной разумно делать ввод не с клавиатуры, а из файла.

Формат выходных данных:

Сами кратчайшие пути и их длины.

Теоретическая часть

Формируем массив $D^{[k]}(i)$ – минимально возможная стоимость переезда (перехода, перевозки) из вершины v_0в v_i на каждом этапе работы нашего алгоритма.

Первоначально он задается как $D^{[0]}(i)$ = (0,∞, ∞,…,∞).

Затем пересчитываем стоимости всех вершин по формуле

$$D^{[k+1]}(i) = \min_j \left(D^{[k]}(j) + C(j,i) \right)$$

до тех пор, пока система не стабилизируется (так называемое *транзитивное замыкание*). В результате мы получим стоимости переезда из каждой вершины графа до заданной v_0 и эти стоимости будут минимально возможными.

Код программы

```
#include <iostream>
#include <fstream>

#define inf 100000
using namespace std;
struct Edges {
  int u, v, w;
};
const int SIZE=6;
 int a[SIZE][SIZE] = {
            {0, -3, -6, -7, 0, -4},
            {-2, 0, -7, -4, -3, -2},
            {-8, -7, 0, -5, -4, -3},
            {-2, -8, -4, 0, -6, 0},
            {0, -2, -4, -8, 0, -7 },
            {-3, -5, -6, -4, -1, 0},

 };
const int Vmax = 1000;
const int Emax = Vmax * (Vmax - 1) / 2;
int i, j, n, e, start;
Edges edge[Emax];
int d[Vmax];

void bellman_ford(int n, int s) {
 int i, j;

 for (i = 0; i < n; i++)
   d[i] = inf;
 d[s] = 0;

 for (i = 0; i < n - 1; i++)
   for (j = 0; j < e; j++)
     if (d[edge[j].v] + edge[j].w < d[edge[j].u])
       d[edge[j].u] = d[edge[j].v] + edge[j].w;

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

```
38.    if (d[i] == inf)
39.      cout << endl
40.           << start << "->" << i + 1 << "="
41.           << "Not";
42.    else
43.      cout << endl << start << "->" << i + 1 << "=" << d[i];
44.}
45.int main() {
46.  setlocale(LC_ALL, "Rus");
47.  int w;
48.  n=SIZE;
49.  e = 0;
50.  for (i = 0; i < n; i++)
51.    for (j = 0; j < n; j++) {
52.      cout << "Вес " << i + 1 << "->" << j + 1 << " > " << a[i][j] << endl;
53.      if (a[i][j] != 0) {
54.        edge[e].v = i;
55.        edge[e].u = j;
56.        edge[e].w = a[i][j];
57.        e++;
58.      }
59.    }
60.
61.  cout << "Стартовая вершина > ";
62.  cin >> start;
63.  cout << "Список кратчайших путей:";
64.  bellman_ford(n, start - 1);
65.}
```

Результаты тестирования

Стоимости вершин на каждом шаге:

```
Вес 1->1 > 0
Вес 1->2 > -3
Вес 1->3 > -6
Вес 1->4 > -7
Вес 1->5 > 0
Вес 1->6 > -4
Вес 2->1 > -2
Вес 2->2 > 0
Вес 2->3 > -7
Вес 2->4 > -4
Вес 2->5 > -3
Вес 2->6 > -2
Вес 3->1 > -8
Вес 3->2 > -7
Вес 3->3 > 0
Вес 3->4 > -5
Вес 3->5 > -4
Вес 3->6 > -3
Вес 4->1 > -2
Вес 4->2 > -8
Вес 4->3 > -4
Вес 4->4 > 0
Вес 4->5 > -6
Вес 4->6 > 0
Вес 5->1 > 0
Вес 5->2 > -2
Вес 5->3 > -4
Вес 5->4 > -8
Вес 5->5 > 0
Вес 5->6 > -7
Вес 6->1 > -3
Вес 6->2 > -5
Вес 6->3 > -6
Вес 6->4 > -4
Вес 6->5 > -1
Вес 6->6 > 0
```

Кратчайшие расстояния:

```
Стартовая вершина > 1
Список кратчайших путей:
1->1=-155
1->2=-157
1->3=-158
1->4=-156
1->5=-153
1->6=-152
```