Московский авиационный институт (национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №9 по курсу «Дискретный анализ»

Студент: Р. Р. Гаптулхаков
Преподаватель: Н. С. Капралов
Группа: М8О-308Б
Дата: 15.09.21
Оценка:
Подпись:

Москва, 2021

Лабораторная работа №5

Задача:

Вариант №5: Поиск кратчайшего пути между парой вершин алгоритмом Беллмана-Форда

Разработать программу на языке C или C++, реализующую указанный алгоритм согласно заданию:
Задан взвешенный ориентированный граф, состоящий из **n** вершин и **m** ребер. Вершины пронумерованы целыми числами **от 1 до n**. Необходимо найти длину кратчайшего пути из вершины с номером **start** в вершину с номером **finish** при помощи алгоритма Беллмана-Форда. Длина пути равна сумме весов ребер на этом пути. Обратите внимание, что в данном варианте веса ребер могут быть отрицательными, поскольку алгоритм умеет с ними работать. Граф не содержит петель, кратных ребер и циклов отрицательного веса.

Формат входных данных:
В первой строке заданы $1 \leq n \leq 105$, $1 \leq m \leq 3 * 10^5$, $1 \leq start \leq n$ и $1 \leq finish \leq n$. В следующих m строках записаны ребра. Каждая строка содержит три числа – номера вершин, соединенных ребром, и вес данного ребра. Вес ребра – целое число от -109 до 109.

Найти самую длинную общую подстроку двух строк.

Формат результата: Необходимо вывести одно число – длину кратчайшего пути между указанными вершинами. Если пути между указанными вершинами не существует, следует вывести строку **No solution**.

1 Описание

Пусть дан ориентированный взвешенный граф G с n вершинами и m рёбрами, и указана некоторая вершина v. Требуется найти длины кратчайших путей от вершины v до всех остальных вершин.

Мы считаем, что граф не содержит цикла отрицательного веса.
Заведём массив расстояний $d[0 \dots n-1]$, который после отработки алгоритма будет содержать ответ на задачу. В начале работы мы заполняем его следующим образом: $d[v] = 0$, а все остальные элементы d[] равны бесконечности ∞.
Сам алгоритм Форда-Беллмана представляет из себя несколько фаз. На каждой фазе просматриваются все рёбра графа, и алгоритм пытается произвести релаксацию (relax, ослабление) вдоль каждого ребра (a,b) стоимости c. Релаксация вдоль ребра — это попытка улучшить значение d[b] значением d[a] + c. Фактически это значит, что мы пытаемся улучшить ответ для вершины b, пользуясь ребром (a,b) и текущим ответом для вершины a.
Утверждается, что достаточно n-1 фазы алгоритма, чтобы корректно посчитать длины всех кратчайших путей в графе (повторимся, мы считаем, что циклы отрицательного веса отсутствуют). Для недостижимых вершин расстояние d[] останется равным бесконечности ∞.
Алгоритм работает за временную сложность $O(V * E)$, V - количество вершин, E - количество рёбер. Пространственная сложность $O(V)$.

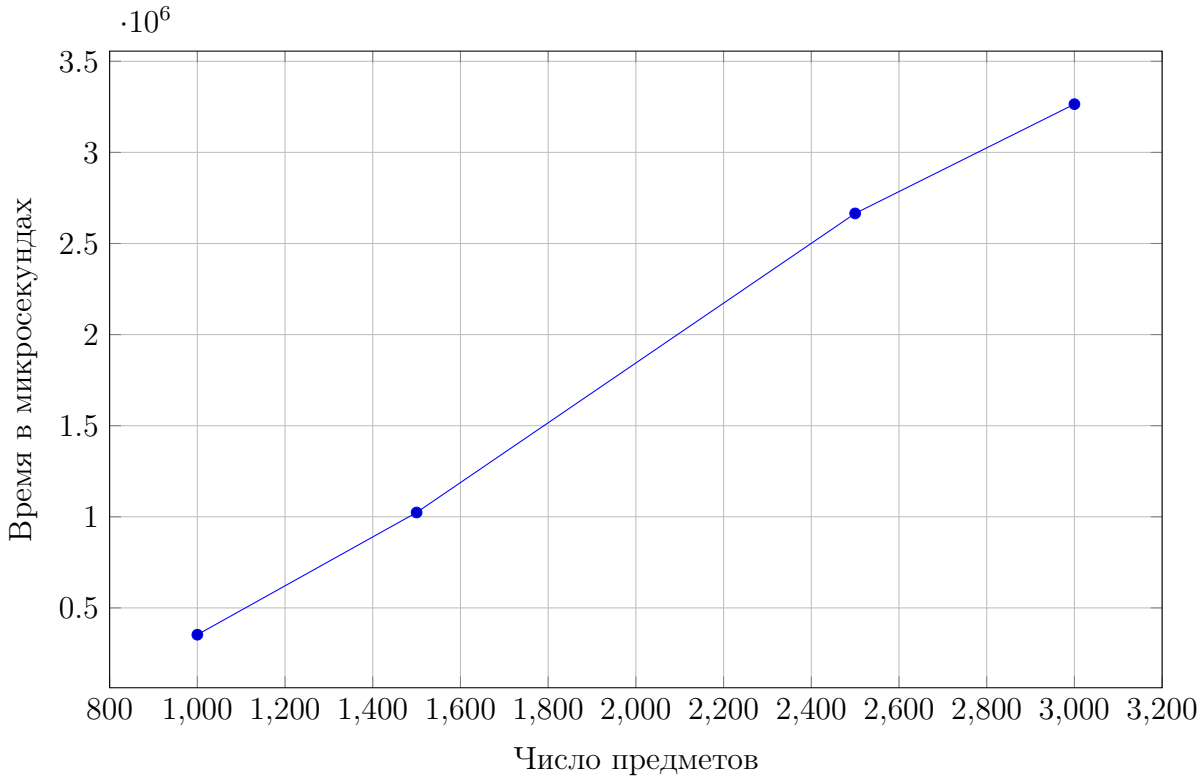
2 Исходный код

```
#include <iostream>
#include <vector>

const long long INF = 1000000001;

struct edge {
    int a;
    int b;
    long long int cost;
};

int main() {
    std::ios::sync_with_stdio(false);
    int n = 0, m = 0, start, finish;
    std::cin >> n >> m >> start >> finish;

    std::vector<long long int> d(n, INF);
    std::vector<edge> ed(m);
    d[start - 1] = 0;
    int a, b, cost;

    for(int i = 0; i < m; ++i) {
        std::cin >> a >> b >> cost;
        ed[i].a = a - 1;
        ed[i].b = b - 1;
        ed[i].cost = cost;
    }

    for(int j = 0; j < n - 1; ++j) {
        bool flag = false;
        for(int i = 0; i < m; ++i) {
            if(d[ed[i].a] < INF) {
                if(d[ed[i].b] > d[ed[i].a] + ed[i].cost) {
                    d[ed[i].b] = d[ed[i].a] + ed[i].cost;
                    flag = true;
                }
            }
        }
        if(!flag) break;
    }

    if(d[finish - 1] == INF) {
        std::cout << "No solution\n";
    } else {
        std::cout << d[finish - 1] << '\n';
    }
    return 0;}
```

3 Тест производительности

1 Графики

Синий: **line**



2 Консольный вывод

rusya@DESKTOP-93JGKCU:/mnt/c/Users/rusya/Desktop/lab9DA ./solution < test1000.txt
No solution
Time: 352944*10 mcseconds
rusya@DESKTOP-93JGKCU:/mnt/c/Users/rusya/Desktop/lab9DA ./solution < test1500.txt
No solution
Time: 1023695 mcseconds
rusya@DESKTOP-93JGKCU:/mnt/c/Users/rusya/Desktop/lab9DA ./solution < test2500.txt
No solution
Time: 2664946 mcseconds
rusya@DESKTOP-93JGKCU:/mnt/c/Users/rusya/Desktop/lab9DA ./solution < test3000.txt
No solution
Time: 3264539 mcseconds

4 Выводы

В отличие от алгоритма Дейкстры, алгоритм Форда-Беллмана применим также и к графам, содержащим рёбра отрицательного веса. Впрочем, если граф содержит отрицательный цикл, то, понятно, кратчайшего пути до некоторых вершин может не существовать (по причине того, что вес кратчайшего пути должен быть равен минус бесконечности); впрочем, этот алгоритм можно модифицировать, чтобы он сигнализировал о наличии цикла отрицательного веса, или даже выводил сам этот цикл.

Алгоритм носит имя двух американских учёных: Ричарда Беллмана (Richard Bellman) и Лестера Форда (Lester Ford). Форд фактически изобрёл этот алгоритм в 1956 г. при изучении другой математической задачи, подзадача которой свелась к поиску кратчайшего пути в графе, и Форд дал набросок решающего эту задачу алгоритма. Беллман в 1958 г. опубликовал статью, посвящённую конкретно задаче нахождения кратчайшего пути, и в этой статье он чётко сформулировал алгоритм в том виде, в котором он известен нам сейчас.

Список литературы

[1] *EMAX*
URL: `https://e-maxx.ru/algo/ford_bellman` (дата обращения: 04.05.21)