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: Меджидли И. И. о
Преподаватель: Михайлова С. А
Группа: М8О-201Б-21

Дата: 26.02.2024
Оценка:
Подпись:

Москва, 2024

Лабораторная работа №9

Задача: Задан взвешенный ориентированный граф, состоящий из n вершин и m ребер. Вершины пронумерованы целыми числами от 1 до n. Необходимо найти величину максимального потока в графе при помощи алгоритма Форда-Фалкерсона. Для достижения приемлемой производительности в алгоритме рекомендуется использовать поиск в ширину, а не в глубину. Истоком является вершина с номером 1, стоком – вершина с номером n. Вес ребра равен его пропускной способности. Граф не содержит петель и кратных ребер.

1 Описание

Давай разберем, что делают основные части кода, решающего данную задачу.

addtoGraph - добавляет ребро с заданной пропускной способностью в граф, представленный матрицей (вектором векторов).

bfs - ищет увеличивающий путь в остаточной сети, используя поиск в ширину, и возвращает минимальную пропускную способность по этому пути. Если такого пути нет, возвращает 0.

$FFFunction$ - основная функция, которая реализует алгоритм Форда-Фалкерсона. Она создает копию исходного графа, которая будет использоваться как остаточная сеть, и инициализирует поток нулем. Затем она повторяет поиск увеличивающего пути с помощью bfs и увеличивает поток на минимальную пропускную способность по этому пути, пока такой путь существует. Она также обновляет остаточную сеть, уменьшая пропускные способности по прямым ребрам и увеличивая по обратным. В конце она возвращает максимальный поток.

main - считывает входные данные, содержащие количество вершин и ребер в графе, а также пропускные способности каждого ребра. Затем она заполняет граф с помощью использования *addtoGraph* и вызывает функцию $FFFunction$ для нахождения максимального потока из истока (индекс 0) в сток (индекс n-1). Выводит результат.

2 Исходный код

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

using namespace std;

void addtoGraph(vector<vector<long long>> &graph, long long a, long long b, long long
    value){
    graph[a][b] = value;
}

long long bfs(vector<vector<long long>> &rgraph, vector<long long> &parent, long long
    s, long long t){
    fill(parent.begin(), parent.end(), -1);
    parent[s] = -2;
    long long sizer = rgraph.size();
    queue<pair<long long, long long>> q;
    q.push({s, 1000000000});
    while(!q.empty()){
        long long u = q.front().first;
        long long imedy = q.front().second;
        q.pop();
        for (long long jj = 0; jj < sizer; ++jj){
            if(parent[jj] == -1 && rgraph[u][jj] > 0 && u != jj ){
                parent[jj] = u;
                long long min_edge = min(imedy, rgraph[u][jj]);
                if(jj == t) return min_edge;
                q.push({jj, min_edge});
            }
        }
    }
    return 0;
}

long long FFFunction(vector<vector<long long>> &graph, long long s, long long t){
    vector<vector<long long>> rgraph = graph;
    vector <long long> parent(graph.size(), -1);
    long long min_edge = 0, max_flow = 0;
    while(min_edge = bfs(rgraph, parent, s, t)){
        max_flow += min_edge;
        long long savet = t;
        while(savet != s){
            long long v = parent[savet];
            rgraph[v][savet] -= min_edge;
            rgraph[savet][v] += min_edge;
            savet = v;
        }
```

```
46      }
47      return max_flow;
48  }
49
50  int main()
51  {
52      ios::sync_with_stdio(false); cin.tie(0);
53      long long n, m, a, b, v;
54      cin >> n >> m;
55      vector<vector<long long>> graph(n, vector<long long>(n));
56      for(long long i = 0; i < m; ++i){
57          cin >> a >> b >> v;
58          addtoGraph(graph, a-1, b-1, v);
59      }
60      long long res = FFFunction(graph, 0, n-1);
61      cout << res;
62      return 0;
63  }
```

3 Тесты

Тестировать программу буду ручным способом.

```
imedzhidli@imedzhidli:~/Desktop/DA/LABA9$ ./lab9
5 6
1 2 4
1 3 3
1 4 1
2 5 3
3 5 3
4 5 10
7
imedzhidli@imedzhidli:~/Desktop/DA/LABA9$ ./lab9
5 6
1 2 1000000000
1 3 1000000000
1 4 1000000000
2 5 1000000000
3 5 1000000000
4 5 1000000000
3000000000
imedzhidli@imedzhidli:~/Desktop/DA/LABA9$ ./lab9
4 3
1 2 2
2 3 3
3 4 1
1imedzhidli@imedzhidli:~/Desktop/DA/LABA9$
```

Как можно заметить, все работает верно.

4 Выводы

При выполнении последней - девятой лабораторной работы по курсу «Дискретный анализ» я познокомился с графами, способам их обхода, в том числе поиском в ширину (bfs), разобрался и смог реализовать алгоритм Форда-Фалкерсона для нахождения максимального потока в графе.

Я рад, что в ходе курса смог научиться многому новому. Благодарен каждому причастному к созданию этого курса за возможность получить необходимые в дальнейшем знания.