СПб НИУ ИТМО

кафедра ИПМ

Алгоритмы и структуры данных

Лабораторная работа № 5

Задачи на графы

Вариант 9

Работу выполнил:

Студент II курса

Группы № 2120

Журавлев Виталий

Санкт-Петербург

2014 г.

**Цель работы:**

Написать программу, являющуюся решением заданной по условию задачи с вводом входных данных и выводом результата.

**Условие задачи:**

Винни-Пух и Пятачок нанялись защищать компьютерную сеть от хакеров, которые выкачивали из компьютеров секретную информацию. Компьютерная сеть Винни-Пуха и Пятачка состояла из связанных между собой больших ЭВМ, к каждой из которых подключалось несколько терминалов. Подключение к одной из больших ЭВМ позволяло получить информацию, содержащуюся в памяти этой ЭВМ, а также всю информацию, доступную для ЭВМ, к которым данная ЭВМ могла направлять запросы. Хаккеры и раньше нападали на подобные компьютерные сети и их тактика была известна. Поэтому Винни-Пух и Пятачок разработали специальную программу, которая помогла принять меры против готовившегося нападения.
Тактика хакеров.
При нападениях хакеры всегда получают доступ к информации всех ЭВМ сети. Добиваются они этого, захватывая некоторые ЭВМ сети, так чтобы от них можно было запросить информацию у оставшихся ЭВМ. Вариантов захвата существует множество.
Например, захватить все ЭВМ. Но хакеры всегда выбирают такой вариант, при котором суммарное количество терминалов у захваченных ЭВМ минимально.
Примечание.
В сети Винни-Пуха и Пятачка ни у каких 2-х ЭВМ количество терминалов не совпадает.
Техническое задание.
Вам необходимо написать программу, входными данными которой было бы описание сети, а выходными - список номеров ЭВМ, которые могут быть выбраны хаккерами для захвата сети согласно их тактике. Ввод осуществляется из файла с именем INPUT.TXT.

Формат ввода.
Количество ЭВМ в сети : N
ЭВМ #1 имеет терминалов : T[1]
ЭВМ #2 имеет терминалов : T[2]
...
ЭВМ #N имеет терминалов : T[N]
Права на запрос :
A[1] B[1]
A[2] B[2]
...
A[K] B[K]
0 0
A[i] и В[i] - номера ЭВМ, последняя строка '0 0' обозначает конец списка прав на запрос, каждая пара A[i] B[i] обозначает, что ЭВМ с номеров A[i] имеет право запрашивать информацию у ЭВМ с номером B[i] (A[i] не равно B[i]).
При вводе числа N и T[i] - натуральные, T[i] <=1000, N<=50, K<=2450.
Входные данные соответствуют приведенным условиям.
Формат вывода.
Номера захватываемых ЭВМ : С[1] C[2] ... С[M].
Количество захватываемых ЭВМ : <M>

|  |  |
| --- | --- |
| Input.txt 5 100 2 1 3 10 1 3 3 2 4 3 4 5 5 4 0 0  | Output.txt 1 4 2  |

**Описание алгоритма:**

**Код программы:**

#include <iostream>

#include <algorithm>

#include <fstream>

#include <vector>

using namespace std;

int findPath(vector<vector<int> > &G, vector <bool> is\_use, int from, int to, int f) {

 if (from == to) return f;

 is\_use[to] = true;

 for (int v = 0; v < G.size(); v++) {

 if (G[to][v]>0 && !is\_use[v]) {

 int delta = findPath(G, is\_use, from, v, min(f, G[to][v]));

 if (delta > 0) {

 G[to][v] -= delta;

 G[v][to] += delta;

 return delta;

 }

 }

 }

 return 0;

}

int MaximumFlow(vector <vector<int> > G, int source, int drain) {

 int flow = 0, d = -1;

 while (d != 0) {

 vector <bool> is\_use(G.size());

 d = findPath(G, is\_use, source, drain, INT\_MAX);

 flow += d;

 }

 return flow;

}

void MinimumCut(vector<vector<int> > G, vector<int> &best\_cut, int &best\_cost) {

 for (int i = 0; i < (1 << (G.size() - 2)); i++) {

 int m = i, sum = 0;

 vector <bool> is\_use(G.size());

 for (int j = 1; j < G.size() - 1; j++) {

 is\_use[j] = m % 2;

 m /= 2;

 }

 is\_use[0] = true;

 is\_use[G.size() - 1] = false;

 for (int k = 0; k < G.size(); k++) {

 for (int l = 0; l<G.size(); l++) {

 if (is\_use[k] && !is\_use[l]) {

 sum += G[k][l];

 }

 }

 }

 if (best\_cost > sum) {

 best\_cost = sum;

 best\_cut.clear();

 for (int j = 0; j < G.size(); j++) {

 if (is\_use[j]) {

 best\_cut.push\_back(j);

 }

 }

 }

 }

}

int main() {

 int n, source, drain, flow, cut = INT\_MAX;

 vector<int> best\_cut;

 string in\_file\_str = "input.txt";

 ifstream in;

 in.open(in\_file\_str);

 in >> n >> drain >> source;

 vector<vector<int> > G(n);

 while (!in.eof()) {

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

 G[i].resize(n, 0);

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

 in >> G[i][j];

 }

 }

 }

 --source;

 --drain;

 flow = MaximumFlow(G, source, drain);

 MinimumCut(G, best\_cut, cut);

 cout << "Maximum flow network: " << flow << endl;

 cout << "Minimum cut: " << cut << endl;

 cout << "Check result: theorem is ";

 if (cut == flow) {

 cout << "true" << endl;

 }

 else

 {

 cout << "false" << endl;

 }

 return 0;

}

**Входные данные:**

input.txt:

6 1 6

0 11 5 7 6 0

0 0 0 4 0 0

0 0 0 0 3 0

0 4 0 0 10 8

0 0 3 10 0 13

0 0 0 0 0 0

 **Результат работы программы:**



**Вывод:**

В ходе выполнения лабораторной работы были исследованы построение максимального потока в сетях и нахождение минимального разреза этой сети.

Из результата вычислений видно, что теорема Форда-Фалкерсона о том, что величина максимального потока в сети равна величине минимального разреза этой сети верна.