Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Кафедра информатики и прикладной математики

Алгоритмы и структуры данных

Экзамен(Лабораторная работа 5)

“Задача на взлом сети”

Вариант 9

Проверил: **Зинчик А.А.**

Старался: **Шкаруба Н.Е.**

Группа **P3218**

2016г

Описание окружения

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 других ЭВМ, к которым данная могла направлять запросы. Хакеры 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 такова: при нападениях они 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.

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.

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

Формат вывода:

Номера захватываемых ЭВМ : C[1] C[2] ... C[M]. Количество захватываемых ЭВМ : <M>

Описание решения

Суть задачи состоит в том, чтобы найти в графе максимальные компоненты связности. Сделать это можно, если найти поиском в ширину все компоненты, а затем руками посмотреть максимальные из них. Если обе компоненты включены друг в друга, то берём ту, у которой количество “терминалов” меньше.

Код

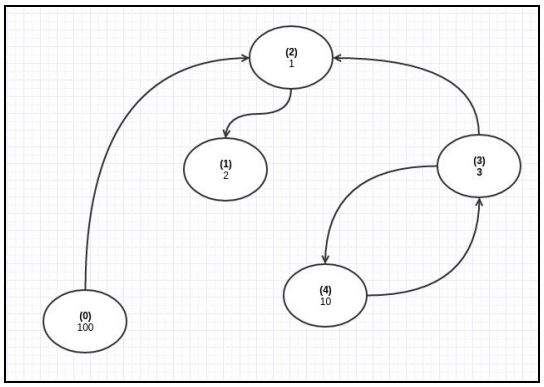
Весь код лежит по адресу:
https://github.com/SigmaOne/ITMO/tree/master/Course%202/Algorithms_and_Data_Structures

Основная функция, ищущая наибольшие связанные компоненты.

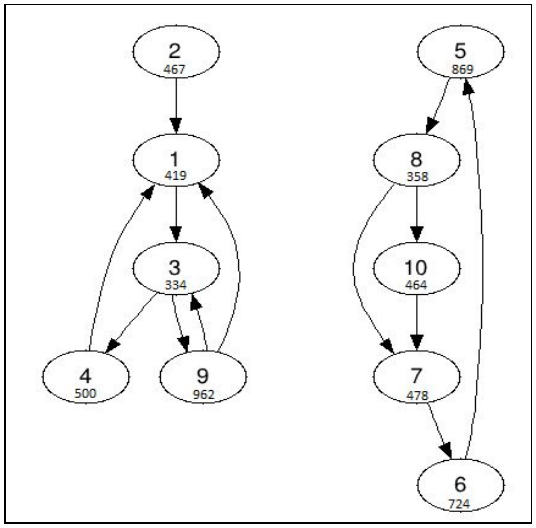
```
vector<Vertex*> findVulnerabilitiesInNetwork(Graph &graph) {
    vector<Vertex*> allVertices = graph.getAllVertices();
    map<Vertex*, vector<Vertex*>> connectedComponents;

    // Get all connected vertices
    for(vector<Vertex*>::iterator it = allVertices.begin(); it != allVertices.end(); it++)
        connectedComponents[*it] = depthFirstSearch(graph, *it);

    // Get min ones
    for (vector<Vertex *>::iterator it1 = allVertices.begin(); it1 != allVertices.end(); it1++)
        for (vector<Vertex *>::iterator it2 = allVertices.begin(); it2 != allVertices.end(); it2++)
            if (*it1 != *it2 && connectedComponents.count(*it1) != 0)
                if (contains(connectedComponents[*it1], *it2) &&
!contains(connectedComponents[*it2], *it1))
                    connectedComponents.erase(*it2);
                else if (contains(connectedComponents[*it1], *it2) &&
contains(connectedComponents[*it2], *it1))
                    if ((*it1)->terminalsAmount > (*it2)->terminalsAmount)
                        connectedComponents.erase(*it1);
                    else
                        connectedComponents.erase(*it2);

    // Shape result
    vector<Vertex*> result;
    for(map<Vertex*, std::vector<Vertex *>>::iterator it = connectedComponents.begin(); it !=
connectedComponents.end(); it++)
        result.push_back(it->first);

    return result;
}
```

Тесты

Алгоритм был протестирован на 2х графах:




```
----------Running GraphTest1----------
Expected computers to hack first: 0 3
Computers to hack first: 0 3
Expected computers amount: 2
Computers amount: 2
```




```
----------Running GraphTest2----------
Expected computers to hack first: 1 7
Computers to hack first: 1 7
Expected computers amount: 2
Computers amount: 2
```

Также, была реализована возможность ввода графа руками:

```
----------Running custom test----------
Input Computers count: 5
Input amount of connected Terminals to computer[0] : 13
Input amount of connected Terminals to computer[1] : 234
Input amount of connected Terminals to computer[2] : 123
Input amount of connected Terminals to computer[3] : 432
Input amount of connected Terminals to computer[4] : 123
Input connections:
Format: {fromId} {toId}
Input "0 0" to find vulnerabilities1 2
3 2
1 2
3 2
0 0
Computers to hack first: 0 1 3 4
Computers amount: 4
```