ФГАОУ ВПО «УрФУ имени первого президента России Б.Н.Ельцина»
Институт математики и компьютерных наук

Кафедра математической экономики

Эйлеровы графы

Курсовая работа
студента 2 курса группы ФИИТ-201
Садовничей Евгении Александровны

Научный руководитель
Асанов Магаз Оразкимович

Екатеринбург
2014

Теоретическая часть

Задача

Разработать алгоритм разбиения множества ребер графа на наименьшее число различных цепей и циклов.

Эйлеровы графы

Сначала рассмотрим общие сведения об эйлеровых графах, для того чтобы в дальнейшем их использовать при построении алгоритма и доказательстве его корректности.

Лемма о рукопожатиях
Любой конечный неориентированный граф имеет чётное число вершин нечётных степеней.

Эйлеров цикл - это цикл, который содержит каждое ребро графа ровно один раз.

Эйлерова цепь - это маршрут в графе, проходящий по каждому ребру ровно один раз.

Граф эйлеров, если в нем существует эйлеров цикл.

Теорема Эйлера
Пусть G - связный граф, тогда следующие условия эквивалентны.

1) G - эйлеров граф
2) Степень каждой вершины четна
3) Множество ребер графа можно разбить на циклы

Алгоритм

Теперь рассмотрим алгоритм разбиения графа на минимальное число цепей и циклов.
Начнем с важного частного случая, когда граф эйлеров. В этом случае граф разбивается на единственный эйлеров цикл.

Приведем алгоритм нахождения этого цикла:

S – стек вершин.
P – массив вершин, входящих в эйлеров цикл.

```
S.add(v)
while not S.isEmpty():
  w := S.top()
  if E contains(w, u):
    S.add(u)
    remove(w, u)
  else:
    S.pop()
    P.add(w)
```

Докажем, что данный алгоритм корректен:
Заметим, что первой в S помещается вершина v, и она будет последней перемещена из S в P. Следовательно, она будет последней вершиной в P. Далее, первый раз, когда обнаружится, что все инцидентные активной вершине ребра пройдены, активной будет стартовая вершина v (Так как

степени всех вершин четны). Значит, эта вершина будет первой перемещена из S в P. Итак, по окончании работы алгоритма в начале и в конце последовательности вершин, содержащейся в P, находится вершина v. Иначе говоря, если эта последовательность представляет маршрут, то этот маршрут замкнут.
Покажем, что P это маршрут содержащий все ребра.
Допустим, что в момент окончания работы алгоритма имеются еще не пройденные ребра. Поскольку граф связен, должно существовать хотя бы одно непройденное ребро, инцидентное посещенной вершине. Но тогда эта вершина не могла быть удалена из S, и S не мог стать пустым.
Будем говорить, что ребро (w,u) представлено в S или P, если в какой-то момент работы алгоритма вершины w и u находятся рядом. Каждое ребро графа представлено в S. Допустим в какой-то момент из S в P перемещена вершина w, а следующей в S лежит u. Возможны 2 варианта:
1) На следующем шаге u перемещена в S. Тогда (w,u) представлено в P.
2) Сначала будет пройдена некоторая последовательность ребер, начинающаяся в вершине u. Ввиду четности степеней эта последовательность может закончиться только в вершине u, а значит она следующей попадет в P и (w,u) будет представлено в P.
Отсюда понятно, что последовательность вершин в P является маршрутом и что каждое ребро графа в конечном итоге будет содержаться в этом маршруте, причем один раз.

Теперь решим исходную задачу. Для этого сведем ее к выше описанному случаю:
Пусть G = (V, E) - связный граф, тогда по лемме о рукопожатиях у него четное количество вершин нечетной степени, обозначим количество этих вершин 2l, если l=0, то мы возвращаемся к ранее рассмотренному случаю. Иначе добавим к графу G вершину z, которую соединим ребрами со всеми вершинами нечетной степени. По теореме Эйлера полученный граф является эйлеровым. В следствии чего в графе имеется эйлеров цикл. Найдем эйлеров цикл по приведенному выше алгоритму.
Начнем обход этого цикла с верины z. Следующая за z вершина в этом обходе будет являться началом первой цепи. Далее в течении обхода мы вернемся в вершину z. Последняя перед z вершина будет окончанием первой цепи. Таким образом будем выделять цепи, пока не совершим полный обход по циклу. Т.к при выделении каждой новой цепи мы проходим по двум ребрам связанным с z, всего таких пар ребер l, значит цепей тоже l. По построению вершины z ясно, что все цепи начинаются и заканчиваются вершинами нечетной степени.

В получившемся цикле цепи, начинающиеся и заканчивающиеся на z, являются разбиением графа на l цепей.

Докажем, что это разбиение минимально:
Допустим мы разбили граф на какое-то количество цепей. Тогда, если мы возьмем какую-нибудь цепь и удалим все ребра содержащиеся в ней, то изменится четность только первой и последней вершин в цепи. Заметим также, что если мы так удалим все цепи из разбиения, то степень каждой вершины в графе станет равной 0. Т.о. нам нужно изменить четность всех нечетных вершин, а для этого каждая нечетная вершина должна являться либо концом, либо началом какой-то цепи. Значит меньше чем 2l/2, т.е. l, цепей быть не может.

Следовательно, это искомое разбиение графа на минимальное количество цепей и циклов.

Практическая часть

Задача

Гоблинский банк "Гринготтс" по праву считается одним из самых надёжных банков в мире волшебников. Он находится в сотнях километров под Лондоном, и охраняется мощными заклинаниями и драконами. А знаете ли вы, как был построен этот банк? Гоблины ведь не такие дураки, как гномы — они не станут копать многокилометровые подземные ходы киркой и лопатой. Как они проложили все туннели без лопат и кирок? Очень просто: чтобы прорыть туннель, гоблины с помощью магии вызывали гипервяка. Гипервяк — это нечто вроде большущей змеи мощностью более тысячи мегачервей. Двигаясь, он поедает всё, что находится на его пути, оставляя за собой удобный, ровный и прочный коридор. Гипервяком легко управлять с помощью магии, заставляя его рыть туннели в нужном направлении.

А знаете ли вы, почему гномы и другие подземные создания не используют гипервяков? Потому, что гипервяки вообще не могут двигаться, не поедая землю или камни. Значит, если гипервяк докопал туннель до нужного вам места, и новые туннели из этого места не нужны, то единственный способ остановить гипервяка — это, … хм…, скажем так, дематериализовать бедную зверушку. Что конечно, строжайше запрещено, ведь эти создания внесены в оранжевую книгу редких магических существ, и за каждого положено платить большой штраф. Просто под землей, где никто не видит, гоблины не очень заботятся о законности. Зато они заботятся о надежности банка, поэтому никогда не копают больше одного туннеля между двумя комнатами. И уж конечно, гоблинам и в голову не придет копать туннель, не соединяющий две разные комнаты.

К сожалению, министру магии попала в руки полная схема туннелей между помещениями банка. Он хочет потребовать штраф за каждый прорытый туннель. Гоблины, конечно, могут заявить, что если два туннеля приходят в одну комнату, то они обошлись всего одним гипервяком для прокладывания обоих туннелей, проведя гипервяка по краю комнаты. И вообще, для любой цепочки туннелей, где конечная комната очередного туннеля совпадает с начальной комнатой следующего, хватит одного гипервяка. Но, разумеется, никто не поверит, что гоблины перетаскивали гипервяка из одного помещения в другое, а расширять существующие туннели они не могли из соображений безопасности.
Помогите гоблинам представить министру магии такой план рытья туннелей, при котором погибло бы минимальное количество гипервяков.

Исходные данные

В первой стоке входа находятся числа N и M ($2 \le N \le 20000$; $1 \le M \le 20000$) — количество помещений банка и туннелей между помещениями. В следующих M строках указаны пары чисел (каждое число от 1 до N) — номера помещений, которые соединяет очередной туннель. От каждого помещения банка можно добраться до другого, пользуясь только прогрызенными гипервяками туннелями.

Результат

В первой строке вывести минимальное количество гипервяков K, необходимое для прокладывания всех туннелей. В последующих K строках через пробел последовательно перечислите номера вершин, через которые прогрызал ходы соответствующий гипервяк согласно вашей схеме.

Решение

Пусть помещение банка - это вершина графа, а туннели - это ребра. (v, w) - ребро, если между банками v и w есть туннель.

Получился связный граф, который нужно разбить на минимальное количество цепей.

Код решения

```
#include <iostream>
#include <fstream>
#include <vector>
#include <stack>
using namespace std;

vector<vector<int>> arr;
vector<int> ans;
stack<int> s;
int n, m;
int k; // количество вершин с нечетной степенью

void printAns() {
	if (k == 0) {
		cout << 1 << endl;
	}
	else {
		cout << k / 2 << endl;
	}
	for (int i = 0; i < ans.size(); i++) {
		if (ans[i] != n) {
			cout << ans[i] + 1 << " ";
		}
		else {
			cout << endl;
		}
	}
}
void removeEdge(int x, int y) {
	for (int j = 0; j < arr[y].size(); j++) {
		if (arr[y][j] == x) {
			arr[y][j] = -1;
			break;
		}
	}
}
void buildCycle(int v) {
	s.push(v);
	while (!s.empty()) {
		int x = s.top();
		bool flag = false;
		for (int i = 0; i < arr[x].size(); i++) {
			if (arr[x][i] == -1) continue;
			flag = true;
			s.push(arr[x][i]);
```

```
                removeEdge(x, arr[x][i]);
                arr[x][i] = -1;
                break;
            }
            if (flag == false) {
                ans.push_back(s.top());
                s.pop();
            }
        }
}
int main()
{
        cin >> n >> m;
        arr.resize(n + 1, vector<int>(0));
        for (int i = 0; i < m; i++) {
            int x, y;
            cin >> x >> y;
            arr[x - 1].push_back(y - 1);
            arr[y - 1].push_back(x - 1);
        }
        for (int i = 0; i < n; i++) {
            if (arr[i].size() % 2 == 1) {
                k++;
                arr[n].push_back(i);
                arr[i].push_back(n);
            }
        }
        if (arr[n].size() == 0) {
            buildCycle(0);
        }
        else {
            buildCycle(n);
        }
        printAns();
}
```