Banda szétválasztása

Innen: Algowiki

Tartalomjegyzék

Feladat

Egy bandában N $(1 \leq N \leq 10000)$ bűnöző szerepel, de nem mindenki ismer közvetlenül mindenkit (közvetve viszont igen). Az ismeretségek számát jelölje M $(1 \leq M \leq 500000)$. A rendőrség szeretné egyetlen bandatag letartóztatásával a bandát a lehető legtöbb olyan részre vágni, amely részek között semmilyen kapcsolat nincs.

Készíts programot, amely megadja, hogy melyik bandatagot tartóztassák le, hogy a banda lehető legtöbb független bandára essen szét!

Több megoldás esetén azt kell kiírni, amelyiknél a keletkező bandák legnagyobbika a lehető legkisebb! Ha ezek is egyformák, akkor közülük a legkisebb sorszámú tagot kell kiírni!

Az eredeti feladat

Az eredeti feladat elérhető a Mesteren (https://mester.inf.elte.hu/) az alábbi kategória alatt: *Mester / NT, OKTV, IOI válogató / IOI Válogató 2019 / 5. Banda szétválasztása*.

Megoldási ötletek

Helyes, de lassú megoldások

Brute force: Az ismeretségi gráf **minden pontjára** nézzük meg, hogyha kivennénk őt a gráfból, akkor a gráf hány komponensre esne szét, és a komponensek közül melyik a legnagyobb. Ezen értékek alapján pedig tudnánk is megoldást mondani, viszont ha egy ilyen szimulációt például mélységi bejárással oldunk meg, akkor a költség $O(N+M) * O(N)$, azaz $O(N^2 + M * N)$ adódik. Ez viszont a feladatban foglalt korlátok mellett lassú megoldás lesz.

Segítségek

1. Ábrázoljuk az ismeretségeket gráfként. Legyenek a csúcsok az emberek, él pedig két csúcs közt akkor fut, hogyha az emberek ismerik egymást.
2. Tényleg minden pont kivételét érdemes leszimulálni? Vagy csak elég pár "speciális" pontét.
3. Mik ezek a "speciális" pontok?

Megoldás

Fontos gondolatok

A feladat szövege szerint ábrázoljuk az ismeretségeket gráfként, ahol egy **csúcs felel meg egy embernek**, és **két csúcs között pontosan akkor fut él, hogyha az emberek ismerik egymást**. Továbbá mivel közvetve mindenki ismer mindenkit, megállapítható, hogy a gráf **összefüggő**. Az ismeretség reláció tulajdonságaiból pedig következik a gráf **egyszerűsége** és az **irányítatlansága** is.

Tekintsük a gráfban az **elvágó pontokat**, azaz az olyan pontokat, amelyek kivételével a gráf 2, vagy annál több komponensre esik szét. Nyílván hogyha van megoldás, akkor az egy elvágó pont lesz, hiszen a feladat azt szeretné, hogy egy csúcs és a belőle induló élek kivételével (egy bandatag letartóztatásával) a lehető legtöbb komponensre essen a gráf. Ha nincs a gráfban elvágópont, akkor technikailag bármelyik pont kivétele után a komponensek száma nem fog változni, az $N-1$ csúcs közül pedig biztosan minden csúcsból minden csúcsba el lehet jutni valamilyen úton, mivel a gráf összefüggő. Viszont ekkor válasszuk az 1-es csúcsot, hiszen ez a legkisebb sorszámú.

Az elvágó pontokra pedig nézzük meg, hány komponensre vágnák a gráfot, hogyha kikerülnének, és ezek közül melyik lenne a legnagyobb. Ezek szerint pedig válasszuk azt az elvágó pontot, amelyik a **legtöbb komponensre** vágná a gráfot, és ezek közül pedig azt, amelyiknél a legnagyobb komponens a **legkisebb**. Több ilyen pont esetén pedig a megoldás a **legkisebb sorszámú** lesz.

Részletes megoldás

Tarjan-algoritmussalsal keressük meg az elvágó pontokat, és az eredeti algoritmust picit módosítva, egy adott pont vizsgálatánál **tartsuk nyílván a pont kivételével keletkező komponensek számát, és közülük a legnagyobb csúcsszámút**.

Külön érdemes talán megemlíteni, hogy az éppen vizsgált pont őseinek komponensét speciálisan kell kezelni, hiszen annak csúcsszáma egyenlő a **pont előtt bejárt pontok számával** (ezt megkapjuk ha a "mikor értünk az adott pontba" változóból kivonunk 1-et) + az olyan, ebből a pontból indított mélységi bejárás során bejárt pontok számával, amik az **ősökkel egy komponensben vannak**. Ez az összefüggőség miatt egyenlő: a gráf csúcsszámából kivonva azon komponensek csúcsszámának összegét, akik nem egyenlőek a szülő komponenssel.

Így mivel rendelkezésre állnak a szükséges értékek, a feladat szövegének elegettéve válasszuk ebből a legmegfelelőbbet.

Komplexitás

Műveletigény

A megoldásban használt Tarjan-algoritmus csak egy mélységi bejárás, tehát ezért a műveletigény $O(N+M)$.

Memóriaigény

A gráfot éllistás formában tároljuk, tehát a memóriaigény $O(N+M)$.

Implementáció

[becsuk]

Implementáció C++ nyelven:

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

#define MAXN 10000
#define MAXM 500000

using namespace std;

vector<int> pontok[MAXN+1];

int N, M;

int tin[MAXN+1];
int low[MAXN+1];
int cnt = 1;

int valasz_komponens = 0, valasz_maxelottunk = 0, valasz = 1;

int
tarjan(int current, int parent)
{
    low[current] = tin[current] = cnt++;
    int komponensek = 1;    /* Hány komponensre szakítaná szét a gráfot a pont kivétele */
    int mi = 1;             /* Mennyi pont van előttünk + mi */
    int maxelottunk = 0;    /* Legnagyobb előttünk lévő komponens csúcsszáma */
    int mogottunk = N - 1;  /* A szülőnkkel egy komponensben hányan vannak */

    bool elvagopont = false;
    for(const auto next : pontok[current]) {
        if(next == parent) continue;

        if(!tin[next]) {
            int gyerekek = tarjan(next, current);

            /* Ha a pont elvágó pont, és kivételével létrejönne az a komponens, aminek pontja a 'next' azonosítójú */
            if(tin[current] <= low[next]) {
                elvagopont = true;
                maxelottunk = max(maxelottunk, gyerekek);
                mogottunk -= gyerekek;
                komponensek++;
            }
            mi += gyerekek;

            low[current] = min(low[current], low[next]);
        }else{
            low[current] = min(low[current], tin[next]);
        }
    }

    /* Ha elvágó pont volt + speciális vizsgálat a legelső pontra. */
    if(elvagopont && (current != 1 || (--komponensek >= 2))) {
        /* Lehet hogy a szülőnk komponense nagyobb, hiszen ezt az előbb nem vizsgáltuk. */
        maxelottunk = max(maxelottunk, mogottunk);

        /* Ha ez a pont a feladat szövege szerint jobb mint az eddigi, jegyezzük meg. */
        if(komponensek > valasz_komponens || (komponensek == valasz_komponens && maxelottunk <= valasz_maxelottunk)
           ){
            /* Ha a legkisebb sorszámú */
            if(komponensek != valasz_komponens || maxelottunk != valasz_maxelottunk || current < valasz)
                valasz = current;
```

```
61              valasz_komponens = komponensek;
62              valasz_maxelottunk = maxelottunk;
63
64          }
65      }
66
67      return mi;
68  }
69
70  int
71  main()
72  {
73      ios_base::sync_with_stdio(0);
74
75      /* Beolvasás */
76      cin >> N >> M;
77      for(int i = 0; i < M; i++) {
78          int a, b;
79          cin >> a >> b;
80          pontok[a].push_back(b);
81          pontok[b].push_back(a);
82      }
83
84      /* Megoldás */
85      tarjan(1, 1);
86
87      /* Kiírás */
88      cout << valasz << endl;
89
90      return 0;
91  }
```

A lap eredeti címe: „https://algowiki.miraheze.org/w/index.php?title=Banda_szétválasztása&oldid=1400”