Tema 2
Seminar AF

Total: 10 puncte

1. (1 punct) Demonstrați că într-un graf orientat dacă ștergeți toate muchiile care intră în S, fluxul maxim de la S la T rămâne neschimbat.

   Rezolvare:

   Avem 2 cazuri:

   - In nodurile din care pleaca muchii spre S nu se poate ajunge plecand din S, atunci fluxul nu este acceptat.
   - In nodurile din care pleaca muchii se poate ajunge plecand din S. Atunci se va forma un ciclu si BFS-ul se va opri si nu va folosi acel drum.

   In concluzie, fluxul nu se modifica daca toate muchiile care intra in S se sterg.

2. (1 punct) Se dă un graf bipartit. Considerăm un algoritm greedy în care la fiecare pas alegem 2 noduri cu muchie între ele care nu sunt încă parte din cuplaj, și adăugăm muchia la cuplaj până când nu mai putem repetă. Demonstrați că în cel mai rău caz, acest cuplaj va avea măcar jumătate din numărul de muchii al cuplajului maxim și găsiți un exemplu pe care este fix jumătate.

   Rezolvare:

   Fie C cuplajul rezultat din rezolvarea Greedy si C’ cuplajul maxim.

   De fiecare data cand se adauga o muchie la cuplajul C, se elimina cel mult 2 muchii din cuplajul C’, adica maxim cate o muchie din nodul 1 al muchiei

adaugate si una din nodul doi. Din aceasta proprietate, rezulta ca cuplajul maxim al lui C >= cuplajul lui C' / 2.

Exemplu:

N1 = 2
N2=2
Muchiile: 1-1, 1-2, 2-1
Cuplaj Greedy: 1
Cuplaj Maxim: 2

3. (1 punct) Țara ta își pregătește tancuri pentru război. Ea are mai multe orașe cu drumuri orientate între unele perechi. Momentan toate tancurile sunt în orașul 1 și vrei să ajungă cât mai multe în orașul N. Dacă fiecare drum are o valuare de cât de multe tancuri se pot duce pe el până când se distruge, dar și fiecare oraș are o valoare de cât de multe tancuri pot trece prin el până când infrastructura se distruge, descrieți un algoritm care calculează numărul maxim de tancuri care putem duce din orașul 1 în N.

   Rezolvare:

   Pentru fiecare nod in afara de nodurile 1 si N facem un nod auxiliar.

   De exemplu, daca avem nodul X atunci acesta este legat de nodul X' de o muchie ce are capacitatea tancurilor din orasul X. In nodul X pastram doar muchiile care intra in X, in timp ce in X' vom pune toate muchiile care pleaca din nodul X.

   Astfel, problema se reduce la o problema de flux maxim si se poate rezolva cu algoritmul lui Edmonds-Karp.

4. (1.5 puncte) Primind un arbore neorientat cu N noduri, aplicați M operații de tipul "ștergeți muchia x-y" sau "răspundeți dacă există drum de la nodul x la nodul y" (răspunsul trebuie dat instant, nu se știu toate operațiile în avans).

Rezolvare:

Initializam un vector care sa mentina mereu radacina arborelui din care face parte nodul. Initial toate nodurile vor avea radacina arborelui initial.

Pentru fiecare operatie de stergere a muchiei x-y actualizam radacina pentru nodurile din noua componenta conexa. Astfel, presupunand ca y e fiul lui x, toate din subarborele cu radacina in y for avea radacina[i]=y.

Pentru fiecare operatie de cautare a drumului de la x la y, vedem daca radacina[x]==radacina[y]. In caz afirmativ, exista drum intre x si y, altfel nu.

5. (1.5 puncte) O școală cu N clase merge într-o excursie folosind M autocare. Clasa i are a[i] elevi și autocarul i are loc pentru b[i] elevi. Distribuiți elevii în autocare astfel în fiecare autocar să fie maxim K elevi din aceeași clasă.

   Rezolvare:

   Solutia 1 – Greedy: Luam fiecare elev din a[1] si punem cate k, elevi ramas daca sunt mai putini decat k sau locuri in autobuz daca sunt mai putine decat k ramase in autobuze si repetam algoritmul cu fiecare claasa.

   Solutia 2 – Grafuri: Facem un graf bipartit cu muchii orientate de la setul nodurilor cu clase la cele cu autobuze. Aceste muchii vor avea capacitatea k. Facem un nod de start si legam de nodurile de clase cu muchii de capacitatea a[i] si un nod de final de care legam nodurile de autobuze cu b[i].

   Astfel, problema se reduce la o problema de flux maxim si se poate rezolva cu algoritmul lui Edmonds-Karp.

6. (2 puncte) Ai N matematicieni și N informaticieni care participă la un concurs cu K runde. La fiecare rundă sunt N probleme care trebuie rezolvate de o pereche formată dintr-un matematician și un informatician. Fiecare persoană poate să lucreze la maxim o problema pe rundă, și aceeași pereche

de oameni nu poate să lucreze împreună în 2 runde diferite. Dându-se o listă de perechi de informaticieni și matematicieni care lucrează bine împreună, determinați dacă este posibil să participe la concurs.

Clarificare: Dacă lucrează bine rezolva o problema, dacă nu rezolva 0.

Rezolvare:

Folosim algoritmul lui Edmonds-Karp pentru a calcula cuplajul maxim si stocam muchiile. Daca acesta este mai mic decat N, atunci returnam -1, deoarece este clar ca nu pot fi rezolvate toate problemele.

Eliminam muchiile deja folosite si repetam algoritmul pana ajungem la K runde. Daca dupa K runde am avut mereu cuplaj maxim = N, atunci pot participa, daca functia a returnat -1 atunci nu pot participa.

7. (2 puncte) Ce complexitate are următorul algorithm, dacă graful este arbore, și invocăm dfs(rădăcina)? Demonstrați complexitatea.

```
dfs(nod):
        Pentru fiecare copil a lui nod
                dfs(copil)
        Dacă nod este frunza // (Nu are copii)
                Set[nod] = {nod}
        Dacă nu este frunza
                copil_max = copilul lui nod cu MĂRIME(Set[copil]) maxim //
O(1);
                Swap(Set[nod], Set[copil_max]) // O(1), Set[copil_max] va
deveni gol
                Set[nod] += {nod}
                Pentru fiecare copil a lui nod în afara de copil_max
                        Set[nod] += Set[copil] // O(MĂRIME(Set[copil]))
```

Rezolvare:

Stim ca DFS are complexitatea O(n) cand acesta este apelat pe un arbore.

Daca un nod este frunza atunci operatiile care se fac au complexitatea O(1).

Daca un nod nu este frunza atunci operatiile care se efectueaza au complexitatea O(1+size(Set[copil])). In cazul in care fiecare nod are maxim un fiu, atunci complexitatea ar fi O(1+n-1) = O(n).

Deci, complexitatea finala ar fi O(n*(1+n)) = O(n^2).