Tema3

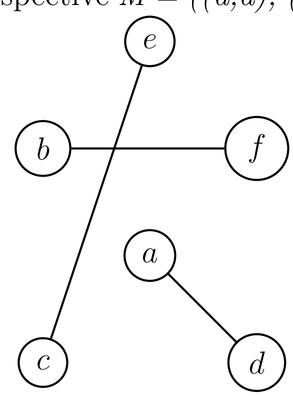
Tiganus Ionel, Grigore Valerian, grupa A3

January 2023

Exercitiul 1

a)Stiind ca G = *(S, T; E)* este un graf bipartit (din enunt) si ca cuplajele *M, M'* contin doar muchiile respective ce leaga nodurile S cu nodurile T (adica $M \subseteq E$, $M' \subseteq E$) deducem ca si graficul partial G' = *(S ∪T,M ∪M')* este un graf bipartit. Din faptul ca graful partial este bipartit inseamna ca si componentele conexe ale acestuia sunt bipartite (orice nod de tipul S se leaga cu macar un nod de tipul T).

Vom lua un exemplu pentru a ilustra ca afirmatia este adevarata: Fie S = *(a,b,c)*, T = *(d,e,f)*, A = *(a,b)*, B = *(e,f)* si cu cuplajele respective M = *((a,d), (b,f))*, M' = *(c,e)*, graful obtinut va fi acesta:



Exercitiul 2

a) Vom presupune ca clasele V_i nu sunt multimi stabile cu G. Stiind ca atunci cand exista o muchie dintre doua noduri dintr-o multime V_i in graful $\vec{G}$ va avea loc un arc dintre respectivele noduri (fiind o orientare ciclica se deduce acest lucru), dar intrucat exista arc, unul dintre noduri are gradul diferit de 0 ceea ce duce la contrariul enuntului. (1)

Abordarea pentru inegalitate va fi urmatoarea: pentru fiecare multime V_i vom prelua toate nodurile cu grad 0 si le vom elimina si totodata vom face o colorare pentru multimea respectiva. (2).

Exercitiul 3

b) Luam cazul cu o singura sursa si o singura destinatie: alegem *s-sursa* si *t-destinatia*.Deci s o sa apartina multimii X si t multimii Y.Pentru a determina un *m-flux* maxim putem face urmatoarele:

Pentru toate nodurile care ajung in X si pentru toate care pleaca din Y vom adauga cate un arc care va avea o capacitate foarte mare.

Acum, folosim algoritmul lui *FordFord-Fulkerson* si calculam fluxul maxim. Pentru a afla cat e un *m-flux* de valoare maxima putem scadea fluxul calculat anterior din fluxul fiecarui arc din E.

Algoritmul are o complexitate *O(E*mf)*, *mf-valoarea* maxima pt flux.

Exercitiul 4

a) Clasa *NP* este definita ca fiind clasa limbajelor, definind probleme de decizie, ce pot fi verificate in timp polinomial. Deci pentru a demonstra problema *FOREST* face parte din clasa de probleme *NP* trebuie sa aratam ca aceasta se poate rezolva intr-un timp polinomial. Algoritmul acesta se poate rezolva in timp polinomial deci problema *FOREST* face parte in categoria problemelor clasei *NP* intrucat pentru acesta se face parcurgerea tuturor posibilitatilor de partitionare a grafului aferent si verificarea fiecarea daca este o padure (ceea ce duce la un timp polinomial, evident).

b)Vom presupune prin absurd ca G este *3COL* insa H nu poate avea 3 paduri. Prin aceasta deducem ca $\exists$ *cel putin* x_1,x_2 care sa fie legate la un nod din G si sa aibe aceeasi culoare. Intrucat x_1 *si* x_2 au aceeasi culoare presupune ca si ca nodul din G sa aiba aceeasi culoare dar daca ar avea aceeasi culoare s-ar contrazice cu colorarea grafului G. (1)

Vom presupune prin absurd ca H poate avea 3 paduri insa G nu este *3COL*. Pentru a forma cele 3 paduri *H-ului* ne trebuie sa cautam pentru fiecare padure o culoare distincta astfel pentru cele 3 paduri ne trebuie 3 colorari deci ca graful G sa fie *3COL*. (2)

Din (1),(2): G este *3COL* daca si numai daca H poate fi partitionat in 3 paduri.

c)Pentru a verifica daca face parte problema *FOREST* la *NP-complete* prima oara trebuie sa demonstram ca face parte din clasa *NP* (am rezolvat la punctul *a)* aceasta). Problema *FOREST* face parte din *NP-complete* intrucat exista o problema X care se reduce la problema aceasta (orice instanta a problemei X se poate regasi si la problema *FOREST* iar solutionarea acestora sa se regaseasca). O astfel de problema este fix problema *3COL* astfel cele 3 colorari a grafului din problema *3COL* sa fie partitionarea in 3 paduri diferite pentru problema *FOREST* (determinarea celor 3 colorari se poate face prin determinarea celor 3 paduri, fiecare padure fiind reprezentata de o culoare anume).