Tema 2 Algoritmica Grafurilor

Hălăucă Andrei IIA1, Belu Cătălin-Cosmin IIA6

November 23, 2018

1 PROBLEMA 1

Algorithm 1 BFS(Breadth-first search)

Require: $G(V, E)$
Ensure: BFS
  for $v \in V$ **do**
    $label(u) \leftarrow -1$;
    $parent(u) \leftarrow -1$;
    $n \leftarrow cuplajPerfect(G)$;
  end for
  $label(s) \leftarrow 0$;
  $parent(s) \leftarrow 0$;
  creeaza coada Q;
  $push(Q, s)$;
  while $Q \neq \emptyset$ **do**
    $u \leftarrow pop(Q)$;
    for $v \in A(u)$ **do**
      if $label(v) < 0$ **then**
        $label(v) \leftarrow label(u) + 1$;
        $parent(v) \leftarrow u$;
        $push(Q, v)$;
        $v' \leftarrow getPartener(v, M)$;
        $label(v') \leftarrow label(v)$;
        $parent(v') \leftarrow parent(v)$;
        $push(Q, v')$;
      end if
    end for
  end while

Algorithm 2 cuplajPerfect(V,E)

Require: $G(V, E)$
Ensure: Cuplajul maximal
  $M \leftarrow \emptyset$;
  repeat
    P $\leftarrow$ Familie maximală de drumuri de creștere minime relativ la M, disjuncte pe noduri;
    for $p \in P$ **do**
      $M \leftarrow M \triangle P$;
    end for
  until $P \leftarrow \emptyset$

2 PROBLEMA 2

Fie G=(V,E) un graf ce conține un cuplaj de
cardinal p dacă și numai dacă $q(G-S) \leq |S| + |G| - 2p, \forall S \subseteq V(G)$.

2.1 $''\Rightarrow''$

Știm că graful G=(V,E) conține un cuplaj de cardinal p. Definim operația $''*''$ care, pentru orice două grafuri G_1 și G_2, adaugă muchie de la orice nod din G_1 la orice nod din G_2.

Astfel, avem $H = G*K^{|G|-2p}$, reprezentând graful format prin adăugarea a $|G| - 2p$ noduri la G, astfel încât nodurile adăugate la graful G să fie adiacente cu toate nodurile grafului G.
Știm, din Teorema lui Tutte, că avem $q(H-S') \leq |S'|, \forall S' \subseteq V(H)$.**(1)**

Fie $|S'| = |S| + (|G| - 2p), \forall S \subseteq V(G)$. Atunci, are loc $G - S = H - S'$, deci putem avea relația $q(G-S) = q(H-S')$.**(2)**
Din **(1)** și **(2)** $\Rightarrow q(G-S) = q(H-S') \leq |S'| = |S| + |G| - 2p$ (♣)

2.2 $''\Leftarrow$"

Știm că $q(G-S) \leq |S| + |G| - 2p, \forall S \subseteq V(G)$.$(\star)$

La fel ca la implicația directă definim $H = G * K^{|G|-2p}$.

Deci $|H| = |G| + |G| - 2p = 2 \cdot |G| - 2p = 2 \cdot (|G| - p)$, adică H are număr par de noduri $\Rightarrow q(H - \emptyset) = 0$. Astfel, pentru a deconecta graful H trebuie să eliminăm toate nodurile adăugate.

De asemenea, $\forall S'$ ce nu conține toate nodurile adăugate vom avea $q(H - S') \leq 1$, pentru că $H - S'$ este conex și, deci, poate avea componentă pară sau componentă impară, în funcție de paritatea $|S'|$.

Definim $S', S' \subseteq V(H)$ astfel încât S' nu conține toate nodurile adăugate în graful $H(|G|-2p)$ și $S'', \forall S'' \subseteq V(H)$ astfel încât S'' conține toate nodurile adăugate în graful $H(|G| - 2p)$.

$\Rightarrow S'' = G \cap S' \Rightarrow H - S' = (G \cup K^{|G|-2p}) - S' = G - S'' \Rightarrow H - S' = G - S'$, deci avem și $q(H-S') = q(G-S'')$ $(\star\star)$.

Din $(\star)$ și $(\star\star) \Rightarrow q(H-S') = q(G-S'') \leq |S''| + |G| - 2p = |S'|$. Avem $q(H-S') \leq |S'|$, deci H satisface Teorema lui Tutte, atunci graful G conține un cuplaj de cardinal p. (♣♣)

(♣) + (♣♣) $\Rightarrow$ **Graful G conține un cuplaj de cardinal p dacă și numai dacă** $q(G-S) \leq |S| + |G| - 2p, \forall S \subseteq V(G)$**.**

3 PROBLEMA 3

Fie $G = (V, E)$ un graf conex și c funcția de cost injectivă pe muchiile sale.

3.1 (a)

$''\Rightarrow''$

Fie graful $G = (V, E)$ și o muchie $e_1 \in E$ care aparține arborelui parțial de cost minim al grafului G, cunoscând funcția de cost c din ipoteză. Vom nota arborele parțial de cost minim al grafului G cu T_G^{min}, deci vom avea $e_1 \in T_G^{min}$. Funcțiia c, fiind injectivă, implică faptul că, costul arborelui parțial de cost minim va fi unic.

Presupunem, prin reducere la absurd, că avem costul curent al grafului notat cu R și că există o muchie care nu este de cost minim într-o tăietură. Dacă muchia e' nu e de cost minim în tăietura făcută, atunci știm sigur că există o muchie pe care o notăm cu e'' care este de cost minim. Deci vom avea $c(e') > c(e'')$, dar cum știm că $e_1 \in T_G^{min}$ și costul arborelui este R $\Rightarrow$ costul arborelui fără e' îl vom nota cu $R' = R - e'$, iar costul arborelui fără e' este $R' + e'$.

Dar, cum costul arborelui cu e'' este $R' + e''$ vom avea $R' + e'' < R' + e'$, deoarece $c(e') > c(e'')$, ceea ce înseamnă că muchia e'' conduce la un arbore parțial de cost minim mai mic. Cum ceea ce reiese de aici este în contradicție cu ipoteza deoarece am plecat de la ideea că acest T_G^{min} este de cost minim, dar am găsit un alt arbore ce are costul și mai mic, ceea ce înseamnă că presupunerea făcută, cum că e' este o muchie de cost minim este falsă.

$\Rightarrow$ Dacă o muchie aparține unui arbore parțial de cost minim, atunci ea este de cost minim într-o tăietură.(1)

$''\Leftarrow''$

Fie T o tăietură și o muchie e' de cost minim x. Fie costul curent al arborelui notat cu R.

Presupunem, prin reducere la absurd, că nu voi alege muchia e' ca și componentă în arbore. De asemenea, știu că trebuie să aleg o muchie din tăietură deoarece dacă nu aș face-o aș pierde proprietatea de conexitate. Deci, voi alege muchia notată e'', de cost $x + 1$. Știm până acum că muchia e'' nu este de cost minim, deci muchia e' este de cost minim. Dacă aleg muchia e', atunci costul va fi: $R + x$, iar dacă aleg muchia e'', costul va fi: $R + x + 1$.Având ca scop principal minimizarea costului total al arborelui și știind că $R + x < R + x + 1$, voi alege muchia e' $\Rightarrow$ presupunerea făcută este falsă întrucât voi alege mereu muchia de cost minim(în cazul acesta e')

pentru a minimiza mereu costul total al arborelui parțial de cost minim.

⇒ **Dacă o muchie este de cost minim într-o tăietură, atunci ea aparține unui arbore parțial de cost minim.(2)**

Din **(1) și (2)**⇒ O muchie aparține unui arbore parțial de cost minim dacă și numai dacă este de cost minim într-o tăietură.

3.2 (b)

3.2.1 $''\Leftarrow''$

Fie graful $G = (V, E)$ care conține un circuit. Fie o muchie e' de cost maxim, ceea ce înseamnă că $e \notin T_G^{min}$.
Considerăm T un arbore parțial de cost minim obținut din graful G prin eliminarea unei muchii e, cu $e' \neq e$, din circuitul lui G(practic, eliminăm posibilitatea de a mai avea circuite).

Presupunem, prin reducere la absurd, că T este un arbore parțial de cost minim cu costul x și $e' \in E(T)$ și că T' este arborele obținut prin eliminarea muchiei de cost maxim din graful G(adică eliminarea muchiei e'). De asemenea, costul lui T' este x', iar $e' \notin E(T')$. Știind că $c(e') > c(e)$, vom avea $x > x'$.
Ținând cont că am plecat de la premisa că T este arbore parțial de cost minim și am găsit T'($e' \notin T'$) arbore parțial de cost mai mic decât T($e' \in T$), putem afirma că presupunerea făcută este falsă ⇒ Un arbore parțial nu poate conține muchia de cost maxim dintr-un circuit întrucât costul acesteia va duce la minimizarea costului arborelui parțial.

Deci, putem afirma că **o muchie de cost maxim într-un circuit nu aparține nici unui arbore parțial de cost minim.**

3.3 (c)

3.4 (d)

4 PROBLEMA 4

4.1 (a)

Fie $T = (V, E)$ un arbore cu $V = \{1, 2, 3, ..., p\}$ cu $p \geq 2$. Fie n, numarul de frunze ale lui T $\Rightarrow n \geq 2$.

Astfel, la fiecare pas i, $i < p - 1$, vom avea minim două frunze, din care vom alege frunza minimă, k, și o vom elimina din arbore, iar x_i va primi valoarea cu care este etichetat tatăl frunzei eliminate.
La pasul $i = p - 1$, în arbore vom avea doar două noduri rămase, ambele fiind frunze. Aici o vom elimina pe cea minimală, deci vom rămâne cu un singur nod, cu etichetă maximă.

La pașii $i \in \{1, 2, ..., p - 2\}$ avem nodul p(cel cu etichetă maximă), astfel:

-p este frunză la pasul i și p are etichetă maximă⇒există cel puțin o altă frunză care va avea etichetă minimală și va fi eliminată⇒p nu se elimină la pașii $1, 2, ..., p - 2$ **(1)**

SAU

-p nu este frunză la pasul i⇒p nu poate fi eliminată **(1)**
La pasul $p - 1$ va fi eliminată frunza minimală. **(2)**

Din **(1)** și **(2)**⇒nodul p nu va fi eliminat în urma pașilor $i \in \{1, 2, ..., p-1\}$, acesta reprezentând ultima componentă a emblemei arborelui de ordinul p după executarea pasului $p - 1$. **(A)**

De asemenea, x_i este, după execuția pasului i, vecinul frunzei șterse. La pasul x_{p-1} se elimină frunza minimă a cărui vecin este p(frunză maximă)$\Rightarrow x_{p-1} = p$. **(B)**

Din (A) și (B)⇒ultima componentă a emblemei unui arbore de ordin p este $x_{p-1} = p$.

4.2 (b)

(b1)
Algoritmul dat returnează un arbore $T = (V, E')$ construit după vectorul x, arborele $T = (V, E')$ fiind identic cu arborele inițial $T = (V, E)$.
Cum $i_1 = min(V \setminus \{x_1, x_2, ..., x_{p-1}\})$ și
$i_k = min(V \setminus \{i_1, i_2, ..., i_{k-1}, i_k, i_{k+1}, ..., x_{p-1}\})$.
Vom avea că $V(T_k) = \{i_k, i_{k+1}, ..., p\}$ și
$E'(T_k) = \{x_k i_k, x_{k+1} i_{k+1}, ..., x_{p-1} i_{p-1}\}, \forall k \leq p - 1,$

deci $E'(T_k) + 1 = V(T_k) \geq 2$(avem cel puțin nodurile i_k și p în arborele T_k).

Cum $i_k \neq x_k, x_{k+1}, ..., x_{p-1}, x_{k+1}, x_{k+2}, ..., i_{p-1} \Rightarrow$ nodul i_k are maxim un vecin în graful T_k. Cum în $E'(T_k)$ avem muchia $x_k i_k \Rightarrow$nodul i_k are cel puțin un vecin în T_k

$\Rightarrow i_k$ **are un singur vecin în** T_k**, deci** i_k **este frunză.**

(b2)

Știm de la punctul **(b1)** că i_k este frunză în T_k. Conform algoritmului dat și a ipotezei $T_k = T \setminus \{i_1, i_2, ..., i_{k-1}\}$ vom deduce că arborele T_{k+1} se va construi de la arborele T_k eliminând nodul cu eticheta i_k. Deoarece nodul i_k este nodul eliminat la pasul k în crearea emblemei lui T, acesta este frunză minimă a lui T la pasul k(conform ipotezei din cerință).**(1)**

Algoritmul dat construiește arborele $T(V, E')$ după vectorul x adăugând nodurile exact în ordinea în care acestea au fost eliminate din $T(V, E)$ pentru a fi construită emblema acestuia(vectorul x). $\Rightarrow$la pasul k din algoritm(primul for), nodul adăugat la arbore este același nod eliminat la pasul k în construcția emblemei. **(2)**

Din (1) și (2) $\Rightarrow$ **nodul** i_k **este frunză minimă a arborelui.**

(b3)

Dacă $k = 1$ avem $T_k = T \setminus \{i_1, i_2, ..., i_{1-1}\} <=> T_k = T \setminus \emptyset <=> T_k = T$. Cum $T_1 = T \Rightarrow$ pentru a demonstra că T este un arbore cu emblema x este suficient să demonstrăm că T_1 este un arbore cu emblema x.

Avem i_1 cea mai mică frunză din T_1(știm asta din (b2)). Demonstrăm că ultima componentă a emblemei lui T_1 este p. Dacă T_{p-1} are 2 noduri(p și p-1), eliminând frunza minimală din $T_{p-1} \Rightarrow$ ultima componentă a emblemei lui T_{p-1} este p. **(1)**

T_k se obține din T_{k-1} eliminând frunza minimală, $\forall k \in \{1, 2, ..., p-1\}$. **(2)**

Din **(1)** și **(2)** $\Rightarrow$ ultima componentă a emblemei lui T_1 este p.

Rămâne să demonstrăm că întreg vectorul x este emblema lui T(până acum știm doar de ultimul element).

Presupunem, prin reducere la absurd, că x nu este emblema lui T_1 astfel încât să existe minim un element $z_i, z_i \neq p$, cu proprietatea că $z_i \neq x_i$, unde $z_i \in Z$, iar $Z \in N^{p-1}$ este presupusa emblemă a lui T_1. Știm că T_k se obține din T_{k-1} eliminând frunza minimală, k. Nodurile sunt eliminate pentru obținerea lui T_k în aceeași ordine în care acestea s-au adăugat în arbore la construirea acestuia pornind de la emblema x.**(*)**

Cum $x \in \{1, 2, ..., p-1\} \Rightarrow$ Afirmația **(*)** se contrazice cu ipoteza noastră $\Rightarrow$ x este emblema lui T_1, dar T_1=T $\Rightarrow$

$\Rightarrow$ **x este emblema lui T.**