**Tema 1**

**Algoritmica Grafurilor**

Problema 3

1. Presupunem că G=(V,E) este un graf.

R-COV(G,k) returnează ”Yes” dacă ∃ o mulțime T⊆V(E), T-vertex cover pentru G, |T|≤k≤n.

1. If E(G)= Ø then return (”Yes”, Ø);

În această secvență se verifica dacă G nu are muchii, iar în cazul în care E(G)= Ø, răspunsul primit va fi că exitsa o mulțime T⊆V(G), T-vertex cover.

1. If|E(G)| > k(|V(G)|-1) then return ”No”;

Considerăm o mulțime de k noduri, fiecare nod din această mulțime poate fi adiacent cu cel mult n-1 noduri (poate fi adiacent cu oricare din celelalte noduri), de unde rezultă că fiecare nod poate parcurge cel mult k(n-1) muchii. Așadar, răspunsul trebuie să fie ”No” dacă |E(G)|>k(n-1).

1. Let {u,v} ∈ E(G);
2. If R-COV(G-u,k-1) return (”Yes”,T) then return (”Yes”,T∪{u})
3. Else if R-COV(G-v,k-1) return (”Yes”,T) then return (”Yes”,T∪{v})

Considerăm acum orice muchie (u,v) ∈ E(G).

Considerăm o mulțime de noduri T⊆V(G).

Dacă T-vertex cover pentru G, atunci înseamna că măcar o extremitate a muchiei (u,v) aparține lui T, adică măcar nodul u sau nodul v este din mulțimea T.

Dacă eliminăm nodul u sau nodul v din G, trebuie să eliminăm si toate muchiile incidente cu acest nod. Obținem în felul acesta un subgraf G’ al grafului G. Facem aceeași eliminare și din T⊆V(G) obținând astfel o mulțime T’⊆V(G’), T’-vertex cover pentru G’.

6. else return (”No”);

De aceea R-COV(G,k) returnează ”Yes” dacă R-COV(G-u,k-1) sau R-COV(G-v,k-1) returnează ”Yes”. Altfel, returnează ”No”, adică nici u si nici v nu aparțin lui T.

1. Presupunem k-constantă.

Atunci când alegem o muchie (u,v) din E(G), există cel mult iteratii ( => complexitate O()).

Pentru fiecare muchie (u,v) ∈ E(G) ∃ cel mult două apeluri recursive (unul pentru nodul u si altul pentru nodul v). Acest apel recursiv formează de fapt un arbore binar de adâncime maximă k-1. Numărul maxim de noduri într-un arbore binar de adâncime cel mult k-1 este -1=>

* T(n,k)= O ()

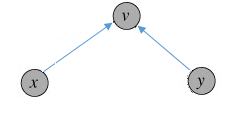
k-constantă => T(n,k)= O().

Problema 1

1. Vom arăta că {B1+, B2+ ,…,Bp+} este o partiție a lui V(D).

Dacă un vertex v ∈ unor componente Bi+ , Bj+⊆{B1+, B2+ ,…,Bp+}, i,j ∈ {1,2,…,p} spunem că v ∈ Bi+ ∩ Bj+ , atunci ∃ un arc de la x ∈ Bi spre v și un alt arc de la y ∈ Bj spre v.

( Observatie! Două noduri a și b au un ”common prey” dacă a+ ∩ b+ ≠ Ø).



Atunci x și y împart același ”common prey” – nodul v, deci este o muchie în Gcp de la x la y, ceea ce contrazice faptul că x și y fac parte din componente diferite ale lui Gcp. Astfel, componentele B1+, B2+ ,…,Bp+ sunt separate.

∀ Bz+ , z ∈ {1,2,…,p} pentru că în orice nod w ∈ Bz+ , z ∈ {1,2,…,p} un arc cu extremitatea în w (adică pleacă din nodul w).

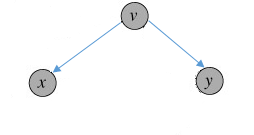
Din ipoteza => faptul că în orice vertex din B1+, B2+ ,…,Bp+ există un arc care intră în nodul w ∈ Bz+ , z ∈ {1,2,…,p}. În concluzie, {B1+, B2+ ,…,Bp+} este o partiție a lui V(D).

Analog pentru {A1-, A2- ,…,Ak-}

Vom arăta că {A1-, A2- ,…,Ak-} este o partiție a lui V(D).

Dacă un vertex v ∈ unor componente Ai- , Aj-⊆{A1-, A2- ,…,Ak-}, i,j ∈ {1,2,…,k} spunem că v ∈ Ai- ∩ Aj- , atunci ∃ un arc de la v spre x ∈ Ai și un alt arc de la v spre y ∈ Aj.

( Observatie! Două noduri a și b au un ”common enemy” dacă a- ∩ b- ≠ Ø).



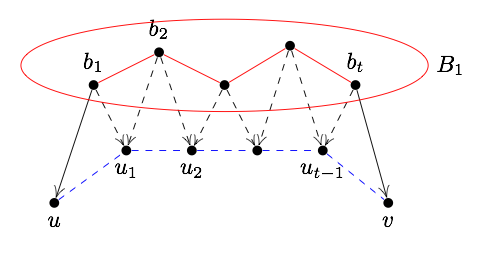
Atunci x și y împart același ”common enemy” – nodul v, deci este o muchie în Gce de la x la y, ceea ce contrazice faptul că x și y fac parte din componente diferite ale lui Gce. Astfel, componentele A1-, A2- ,…,Ak- sunt separate.

∀ Az- , z ∈ {1,2,…,k} pentru că în orice nod w ∈ Az- , z ∈ {1,2,…,k} un arc cu vârful în w (adică intră din nodul w).

Din ipoteza => faptul că în orice vertex din A1-, A2- ,…,Ak- există un arc care iese în nodul w ∈ Az- , z ∈ {1,2,…,k}. În concluzie, {A1-, A2- ,…,Ak-} este o partiție a lui V(D).

1. Presupunem că ∃ două vârfuri u și v ∈ B1+. Dacă u și v împart același ”common prey” w ∈ B1, atunci ele împart si același ”common enemy”, de unde obținem că sunt adiacente în Gce.

Pe de altă parte avem b1, bt ∈ B1 în care există un arc [b1, u] și un arc [bt, v]=> ∃ un drum de lungime minimă de la b1 la bt în Gcp.



Roșu pentru Gcp si albastru pentru Gce .

Pentru că b1 și b2 sunt adiacente în Gcp ele împart același ”common prey” în u1, apoi u si u1 împart același ”common enemy” prin b1 => u și u1 sunt adiacente în Gce => ∃ (u, u1) ∈ E(Gce).

Generalizare:

Pentru că bi și bi+1 sunt adiacente în Gcp ele împart un ”common prey” în ui. Apoi ui-1 și ui împart un ”common enemy” în Bi => ui-1 și ui sunt adiacente în Gce. Așadar, u și v fac parte din aceeași componentă convexă din Gce => componentele conexe din Bi+, i ∈ {1,2,…,p}, aparțin unei componente din Aj +, j ∈ {1,2,…,k}=> Bi și Aj sunt bijective și p=k.

Analog pentru Aj si Bi.

Presupunem că ∃ două vârfuri u și v ∈ A1-. Dacă u și v împart același ”common enemy” w ∈ A1, atunci ele împart si același ”common prey”, de unde obținem că sunt adiacente în Gcp.

Pe de altă parte avem a1, at ∈ A1 în care există un arc [a1, u] și un arc [at, v]=> ∃ un drum de lungime minimă de la a1 la at în Gce.

Pentru că a1 și a2 sunt adiacente în Gce ele împart același ”common enemy” în u1, apoi u si u1 împart același ”common prey” prin a1 => u și u1 sunt adiacente în Gcp => ∃ (u, u1) ∈ E(Gce).

Generalizare:

Pentru că ai și ai+1 sunt adiacente în Gce ele împart un ”common enemy” în ui. Apoi ui-1 și ui împart un ”common prey” în Aj => ui-1 și ui sunt adiacente în Gcp. Așadar, u și v fac parte din aceeași componentă convexă din Gcp => componentele conexe din Aj-, i ∈ {1,2,…,k}, aparțin unei componente din Bi +, i ∈ {1,2,…,p}=> Aj și Bi sunt bijective și p=k.

Din cele două cazuri => că Gcp și Gce au același număr de componente conexe.

Problema 2

1. D=(V,E) are proprietatea că ∀ nod v ∈ V(D), d+ (v) = d-(v)=1.

Din această proprietate deducem faptul că pentru ∀ nod v ∈ V(D), ∃ x,y ∈ V(D) astfel încat arcele (x,v) și (v,y) ∈ E(D).

Tot din proprietatea d+ (v) = d-(v)=1 deducem faptul că în digraful cu cel puțin 3 noduri se formează un circuit.

Presupunem că pentru ∀ i<n (n=numărul de vârfuri), alegând orice mulțime S ⊂ V, cu |S|=k, ∃ un nod v din V(D) astfel încât |S ∩ v+ |≡ 1 mod 2.

P(1): pentru k=1, |S|=1, adică conține un nod v, ∃ un nod y ∈ V(D) și arcul (y,v) ∈ E(D) astfel încât |y+ ∩ S| ≡ 1 mod 2=> S conține un singur vârf => ”A”

Vom considera P(k) adevărată și demonstrăm că P(k+1) adevărată.

P(k): pentru ∀ mulțime S ⊂ V(D), |S|=k. ∃ un vârf v ∈ V(D) astfel încât |v+ ∩ S| ≡ 1 mod 2. Vom nota această mulțime S cu A.

P(k+1): ∀ S ⊂ V, |S| =k+1=>∃ un vârf w ∈ V(D) astfel încât |w+ ∩ S| ≡ 1 mod 2. Vom nota această mulțime S cu B.

Vom considera că {B}={A}∩{v}, v ∈ V(D), dar nu ∈ A, adică v ∈ V(D)\A.

∀ nodul v ∈ V(D)\A, ∃ un nod i astfel încât (y,v) ∈ E(D) => y+={v}, adică nu ∃ un alt nod care să plece din y=> | y+|=1.

| y+∩{v}|=| y+∩{A∪ v}| ≡ 1 mod 2=>P(k+1) ”A”. Deci, conform principiului inducției matematice => în digraful D nu ∃ mulțimi pare.



x este inițial ”solo-prey” pentru u;

z este inițial ”solo-prey” pentru w;

y este ”common-prey” pentru u și w;

După ce facem modificările necesare obținem D u ͦ w.

Acum y o să fie ”solo-prey” pentru w, iar z o să fie ”common-prey” pentru u și w.

Vom demonstra că digraful inițial D are o mulțime pară < = > D u ͦ w  are o mulțime pară.

|u+|=|x|+|xi|+|c|+|ci|

|w+|=|y|+|yi|+|c|+|ci|, unde |x|,|y|-toate ”solo-prey” ale lui u,w care nu sunt în S

|xi|,|yi|-toate ”solo-prey” ale lui u,w care sunt în S

|c|-toate ”common-prey” ale lui u și w care nu sunt în S

|ci|-toate ”common-prey” ale lui u și w care sunt în S

|xi+ci| ≡ 0 mod 2

|yi+ci| ≡ 0 mod 2 , unde xi,yi,ci au aceeași paritate

|u+|=|x|+|xi|+|y|+|yi|

|w+|=|c|+|ci|+|y|+|yi|

Mulțimea de ”solo-prey” pentru w devine mulțimea de ”common-prey” pentru u și w, iar ”common-prey” devine ”solo-prey” pentru w.

|u+ ∩ S|=xi+ui ≡ 0 mod 2

Pentru că ∃ o mulțime pară în D u ͦ w  => ∃ o mulțime pară și în digraful inițial.

1. Fie matricea de adiacență A a lui D cu elemente din corpul GF(2).

Presupunem că există i= Ø mulțime S pară. Fie nodul v ∈ lui S. Adunăm toate coloanele corespunzătoare nodurilor din S pe coloana v.

for (int i=0; i<nr\_linii;i++)

{

for(int j=0;j<nr\_coloane;j++)

{

a[i][v]+=a[i][j];

}

}

Dacă obținem pe coloana v doar 0 (a[i][j]=0 pentru orice i=0,i<nr\_linii)

= > det(A)=0 = > mulțimea S este mulțime pară de vârfuri

a11 a12 ... a1v ... a1m

a21 a22 ... a2v ....a2m

.......................

an1 an2 ... anv ... anm

m=nr\_de\_linii

n=nr\_de\_coloane