Tema Analiza Algoritmilor Partea 1

Constantin Ioan, 321CC

November 29, 2017

1 Problema 1

(A)
Diagrama nu infirma informatiile prezentate la curs:
1. $P \subseteq NP$ si $NPC \subseteq NP$
2. Teorema conform careia $P = NP \Rightarrow P = NP = NPC$
3. Teorema 3 conform careia $NPC \cap P \neq \emptyset \Rightarrow P = NP$
(In mod evident daca $NPC = P \Rightarrow NPC \cap P = P$ (= NPC) ($\neq \emptyset$))
4. Corolarul teoremei conform caruia $NPC \cap P \neq \emptyset \Rightarrow NPC \subseteq P$

(B)
Diagrama infirma informatiile prezentate la curs, deoarece contrazice teorema conform careia $P = NP \Rightarrow P = NP = NPC$. In cazul de fata, desi P = NP, exista elemente din multimea P care nu apartin multimii NPC, deci $NPC \subset P$, dar $NPC \neq P$.

(C)
Diagrama nu infirma informatiile prezentate la curs:
1. $P \subseteq NP$ si $NPC \subseteq NP$
2. $P \subset NP$, dar $P \neq NP$, deci nu se respecta ipoteza
teoremei conform careia $P = NP \Rightarrow P = NP = NPC$
3. $NPC \cap P = \emptyset$, asadar nu se respecta ipoteza:
- teoremei 3 conform careia $NPC \cap P \neq \emptyset \Rightarrow P = NP$
- corolarului teoremei 3 conform caruia $NPC \cap P \neq \emptyset \Rightarrow NPC \subseteq P$

(D)
Diagrama infirma informatiile prezentate la curs, deoarece, conform teoremei 3, daca $NPC \cap P \neq \emptyset$ atunci P = NP, iar in cazul de fata desi, in mod evident $NPC \cap P \neq \emptyset$, se observa ca exista elemente din multimea
P care nu apartin multimii NP si reciproc, asadar rezulta $P \neq NP$, ajungandu-se la contradictie cu teorema 3.

2 Problema 3

(A)

```
ShortPath(G,u,v,k)
   for m=1:n
      for i=1:n
         for j=1:n
            if (cost[i][m] ≠ infinit ∧ cost[m][j] ≠ infinit ∧ cost[i][m] + cost[m][j] < cost[i][j])
               cost[i][j]=cost[i][m]+cost[m][j]
            endif
         endfor
      endfor
   endfor

   if(cost[u][v] < k)
        return 1
    else
        return 0
   endif
endfunction
```

Complexitatea lui ShortPath este data de cele 3 for-uri care au cate n pasi, deci este egala cu $\theta(n^3)$.
Algoritmul are complexitate polinomiala $\Rightarrow$ algoritmul este tractabil
Algoritmul este determinist, intrucat fiecare operatie de control sau prelucrare a datelor are rezultat unic determinat.
Algoritmul este determinist tractabil $\Rightarrow$ ShortPath $\in$ P

(B)

```
Parcurgere(i,vizitat,max,numarnatural,lung,final)
   if(lung <= n)
      if(i==final)
         if(max > numarnatural)
            success
         else
            fail
         endif
      else
         j=choice(1,2,...,n)
         if(vizitat[j] == 0 ∧ arc[i][j])
            vizitat[j]=1
            ok=Parcurgere(j,vizitat,max+cost[i][j],numarnatural,lung+1,final)
            vizitat[j]=0
            if(ok==1)
               success
            endif
         endif
      endif
   endif
endfunction

LongPath(G,u,v,k)
        Parcurgere(u,vizitat,0,k,1,v)
endfunction
```

Algoritmul este nedeterminist intrucat contine operatia choice(), care nu are rezultat unic definit(rezultatul este o valoare din multimea finita (1,2,...,n)).
Calculam complexitatea algoritmului folosind recursivitatea:
$T(n)=T(n-1)+\theta(1),T(n-1)=T(n-2)+\theta(1)$, ...
Adunam aceste relatii, se vor simplifica T(n-1), T(n-2), ... si rezulta:
$T(n)=n*\theta(1)=\theta(n)$
Algoritmul are complexitate polinomiala $\Rightarrow$ algoritmul este tractabil
Algoritmul este nedeterminist tractabil $\Rightarrow$ LongPath $\in$ NP

Precizari:
   Functia Parcurgere:
      i - nodul curent

j - nodul urmator

max - suma costurilor muchiilor pana in punctul curent

numarnatural - numarul k din enunt

lung - lungimea curenta

final - ultimul nod din drum (nodul v din enunt)

cost[][] - matricea costurilor

arc[][] - matricea care are 1 pe pozitia (i,j) daca exista muchia (i,j) si 0 altfel