Tema Analiza Algoritmilor Partea a 2-a

Constantin Ioan, 321CC

1 Problema 4

Notam problema propusa (sugerata in enunt) din NPD cu P1.

Presupunem ca nu sunt noduri izolate si graful G este conex,
iar toate muchiile grafului G au costul 1.

Algoritmul de conversie are ca intrari intrarile de la P1 si
ca iesiri intrarile de la Long Path.

```
Conversie(G)
   n=card(V)
   u=n+1
   v=n+2
   V'=V U {u, v}
   for each x ∈ V
      E'=E U {(u, x), (x, v)}
      (u,x)=1
      (x,v)=1
   endfor
   K=n+1
   return ((V',E'),K,u,v)
```

Complexitate temporala:
Alogritmul Conversie are complexitate polinomiala (O(n)) datorita
acelui for each

Precizari:

- Graful G' contine toate nodurile grafului G plus nodurile u si v
- Cream muchii de cost 1 intre nodul u si toate varfurile grafului G si intre nodul v si toate varfurile grafului G
- Stabilim K ca fiind suma costurilor celui mai costisitor drum de la u la v
- Cum toate muchiile sunt 1 cel mai costisitor drum este si cel mai lung
- Cel mai lung drum de la u la v uneste toate nodurile grafului G, deci are n-1 muchii din graful G plus prima muchie a drumului(care pleaca din u) plus ultima muchie a drumului(care ajunge in v), deci in total are n+1 muchii
- Returnam intrarile de la Long Path

Demonstram echivalenta urmatoare:
P1 returneaza 1 pentru graful G $\iff$ Long Path returneaza 1 pt. graful G'

$\Leftarrow$:
Fie (x1,x2),(x2,x3),...,(x(n-1),x(n)) un drum care trece prin fiecare nod al grafului G o singura data. Acest drum contine n-1 muchii.
Adaugam drumului o muchie (u,x1) la inceput si o muchie (x(n),v) la sfarsit.
Noul drum din graful G' de la u la v are n-1+2=n+1 muchii de cost 1, deci costul total este n+1.
K=n+1 $\Rightarrow$ costul total $\geq$ K, deci Long Path returneaza 1 pentru graful G'

$\Rightarrow$:
Drumul (u,x1),(x1,x2),(x2,x3),...,(x(i-1),x(i)),(x(i),v) din graful G', cu i in intervalul [1,n] are costul total $\geq$ K, deci costul total $\geq$ n+1.

Muchiile (u,x1) si (x(i),v) au costul 1, asadar drumul din graful G (x1,x2),(x2,x3),...,(x(i-1),x(i)) are costul total $\geq$ n+1-2=n-1.

Cum toate muchiile au costul 1 si nodurile nu se repeta inseamna ca drumul din graful G trece prin n noduri(i==n), asadar drumul trece prin fiecare nod al grafului G o singura data, deci P1 returneaza 1 pentru graful G.

—Din echivalenta de mai sus, complexitatea temporala polinomiala a algoritmului Conversie si faptul ca intrarile algoritmului Conversie sunt intrarile P1, iar iesirile algoritmului Conversie sunt intrarile Long Path$\Rightarrow$
$\Rightarrow$ P1 se reduce polinomial la Long Path

2 Problema 6

(A)

```
PozitieCorecta(A,x,y)
    for i=1:x
        if(A[i][y] == 1)
            return 0
        endif
    endfor

    i=x-1
    j=y-1
    while(i >= 1 ∧ j >= 1)
        if(A[i][j] == 1)
            return 0
        endif
        i −−
        j −−
    endwhile

    i=x-1
    j=y+1
    while(i >= 1 ∧ j <= n)
        if(A[i][j] == 1)
            return 0
        endif
        i −−
        j ++
    endwhile
    return 1
endfunction

Afis(A,nrsol)
    print nrsol
    for i=1:n
```

```
        for j=1:n
            if(A[i][j] == 1)
                      print 'R'
            else
                      print '0'
            endif
        endfor
    endfor
    nrsol++
endfunction

UmplereTabla(A,x)
    if(x ≠ n)
        for j=1:n
            if(PozitieCorecta(A, x, j) == 1)
                 A[x][j]=1
                 UmplereTabla(A,x+1)
                 A[x][j]=0
            endif
        endfor
    else
        Afis(A,nrsol)
    endif
return 0
endfunction

Main()
    for i=1:n
        for j=1:n
            A[i][j]=0
        endfor
    endfor
    nrsol=1
    UmplereTabla(A,1)
endfunction
```

Complexitate spatiala UmplereTabla:
$-$ >A si x sunt date de intrare, deci nu se iau in considerare
$-$ >j<=n $\Rightarrow$ O(log(n))
$-$ >Apel PozitieCorecta: 2*O(log(n))=O(2*log(n))=O(log(n))
$-$ >Apel recursiv UmplereTabla(A,x+1):
  - x<=n $\Rightarrow$ O(log(n))
  - x=1:n, j=1:n $\Rightarrow$ sunt n^2 iteratii

$\Rightarrow n^2 * O(log(n))$
$-$ >Apel Afis: 2*O(log(n))=O(2*log(n))=O(log(n))
$-$ >Asadar, complexitatea spatiala este
O(log(n))+O(log(n))+n^2*$O(log(n))$+O(log(n))=O(3*log(n)+n^2*log(n))=O(n^2*log(n))

(B)
Functia UmplereTabla foloseste functiile PozitieCorecta si Afis de la subpunctul anterior, iar functia Main() este aceeasi.

```
UmplereTabla(A,x)
  if(x ≠ n)
    j=choice({1, 2, ..., n})
      if(PozitieCorecta(A, x, j) == 1)
        A[x][j]=1
        UmplereTabla(A,x+1)
        A[x][j]=0
      endif
  else
    Afis(A,nrsol)
    success
  endif
fail
endfunction
```

Daca algoritmul nu s-ar opri la primul succes, acesta ar afisa toate solutiile problemei damelor.

Complexitate spatiala UmplereTabla:

$-$ >A si x sunt date de intrare, deci nu se iau in considerare

$-$ >j<=n $\Rightarrow$ O(log(n))

$-$ >Apel PozitieCorecta: 2*O(log(n))=O(2*log(n))=O(log(n))

$-$ >Apel recursiv UmplereTabla(A,x+1):

- x<=n $\Rightarrow$ O(log(n))
- x=1:n $\Rightarrow$ sunt n iteratii

$\Rightarrow n * O(log(n))$

$-$ >Apel Afis: 2*O(log(n))=O(2*log(n))=O(log(n))

$-$ >Asadar, complexitatea spatiala este

O(log(n))+O(log(n))+$n*O(log(n))$+O(log(n))=O(3*log(n)+n*log(n))=O(n*log(n))