Tema 3 Algoritmica Grafurilor

Andrei Hălăucă — Grupa A1
Cătălin-Cosmin Belu — Grupa A6

January 11, 2019

Problema 1

a)

$x_{ij} = 0$;
$x = x_{ij}$;
$e(s) \leftarrow (\cdot, \cdot, \text{inf})$;
Etichetat[],Scanat[];
while $\exists inod | Etichetat[i] == true \&\& Scanat[i] == false$ **do**
  Scan(i);
  if $Etichetat[i] == true$ **then**
    modifica x pe drumul dat de etichete;
    $Etichetat[k] = false, \forall k$;
    $e(s) \leftarrow (\cdot, \cdot, \text{inf})$;
  end if
end while
$S \leftarrow \{i : i \in V, Etichetat[i] == true\}$;
$T \leftarrow V - S$;
$c(S, T) = NULL$;
for $i \in S$ **do**
  for $j \in T$ **do**
    $c(S, T) += c_{ij}$;
  end for
end for
$c(S', T') == NULL$;
for $i \in S'$ **do**
  for $j \in T'$ **do**
    $c(S', T') += c_{ij}$;
  end for

end for
if $c(S,T) == c(S',T')$ **then**
  return true;
end if
return false;

b)

i ← s;
$DFS(i)$
{
if $i == t$ **then**
  return false;//fluxul x nu este de valoare maximă
end if
$BeenThere[i] = true$;
for i=0;i<nrVertex;i++ **do**
  if forward(i,j)&&BeenThere[j]==false **then**
    if $x_{ij} < c_{ij}$ **then**
      $DFS(j)$;
    end if
  else if backward(i,j)&&BeenThere(j)==false **then**
    if $x_{ji} > 0$ **then**
      $DFS(j)$;
    end if
  end if
end for
return true;
}

c)

$DFS(i, DRUM)$
{
if $i == t$ **then**
  return false;//fluxul x nu este de valoare maximă
end if
$BeenThere[i] = true$;
$DRUM.push\_back(i)$;
for i=0;i<nrVertex;i++ **do**
  if forward(i,j)&&BeenThere[j]==false **then**
    if $x_{ij} < c_{ij}$ **then**
      $DFS(j, DRUM)$;
    end if
  else if backward(i,j)&&BeenThere(j)==false **then**
    if $x_{ji} > 0$ **then**
      $DFS(j, DRUM)$;
    end if
  end if
end for
return true;
}
$c(e_0) + +$;
$Flux\_maxim(R, x)$;
{
if $DFS(s) == false$ **then**
  $x^* = x \bigotimes r(DRUM)$;
end if
$returnx^*$;
}

Problema 2 x flux întreg într-o rețea R=(G,s,t,c),v(x)=v_1+v_2+...+v_n,$v_i \in \mathbb{N}^*, \forall i = 1, p, p \geq 1$.

Știm că o funcție $x : V \times V \to \mathbb{R}$ este flux dacă satisface următoarele:

(i) $0 \leq x_{ij} \leq c_{ij}, \forall (i,j) \in V \times V$;

(ii) $\sum_{j \in V} x_{ji} - \sum_{j \in V} x_{ij} = 0, \forall i \in V - \{s,t\}$

Din teorema fluxului întreg cunoaștem că dacă toate capacitățile din R sunt întregi atunci există un flux întreg, de valoare maximă,x($\forall x_{ij} \in \mathbb{Z}_+$). În cadrul problemei noastre, fluxul întreg de valoare maximă este chiar x. De asemenea, valoarea fluxului este $v(x) = \sum_{i \in V} x_{it} - \sum_{i \in V} x_{ti}$.

Din ipoteză cunoaștem $v_i \in \mathbb{N}^*, \forall i = 1, p, p \geq 1 \implies$ Fluxul $x > 0$,dar din teorema fluxului întreg știm că $x \in \mathbb{Z}$, deci $x \in \mathbb{N}^*$, adică x este flux natural strict pozitiv.

Presupunem, prin reducere la absurd, că ar exista un x^1 un flux $\leq x$(știm că x are valoarea maximă) astfel încât $v(x^1) = v(x) - 1, \forall x$ flux întreg. În continuare, x_1 trebuie să respecte proprietățile unui flux,deci le vom ilustra.

Știm că $0 \leq x_{ij} \leq c_{ij}, \forall (i,j) \in V \times V$, dar și că $x^1 \leq x$(din presupunerea făcută), deci $0 \leq {x^1}_{ij} \leq c_{ij}, \forall (i,j) \in V \times V$. De asemenea, $\sum_{j \in V} x_{ji} - \sum_{j \in V} x_{ij} = 0, \forall i \in V - \{s,t\}$ și $x^1 \leq x$(tot din presupunerea făcută), deci $\sum_{j \in V} {x^1}_{ji} - \sum_{j \in V} {x^1}_{ij} = 0, \forall i \in V - \{s,t\}$.

$\implies$ x^1 este flux.

Identic pentru $x^2, x^3, ..., x^p$, sugerând faptul că pot scade cu o unitate din valoarea fluxului, care să fie $\leq x$(deoarece x e de valoare maximă).

Problema 3

a) G nu este 2-ring $\Leftrightarrow$

(1) Toate gradele nodurilor sale nu sunt multipli de 2, dar numărul de muchii din G este multiplu de 2, adică vorbim despre un graf eulerian

SAU

(2) Există noduri cu grad care nu este multiplu de 2, deci nu avem graf eulerian.

Dacă G nu este 2-ring atunci el admite o 2-colorare-fair.

(1) G graf eulerian $\Rightarrow$Avem C-circuit eulerian în G(V,E) astfel încât$\forall e \in E, C \bigcap e \neq \emptyset$, și C-circuit eulerian, deci avem $\forall v \in V, C \bigcap v \neq \emptyset$ $\Rightarrow$Avem nevoie doar de două culori pentru a colora întregul graf,$\forall e_1, e_2 \in E$, unde e_1 și e_2 au un mod comun și sunt diferite(sunt muchii alăturate), e_1 și e_2 vor fi colorate diferit.

În aceste condiții vom avea numărul de muchii colorate cu culoarea a egal cu numărul de muchii colorate cu culoarea b în graful nostru, deci în fiecare nod vor intra un număr de muchii de culoare a egal cu numărul de muchii de culoare b, deci G admite o 2-colorare-fair.

(2) Avem graful G=(V,E).

Aici avem suma gradelor tuturor nodurilor=$2|E|$ $\Rightarrow$numărul de noduri de grad impar într-un graf este par.($\star$)

Știind acestea vom adăuga un nou nod la graful nostru și vom adăuga câte o muchie între nodul nou adăugat și fiecare nod ce are gradul impar. Astfel toate nodurile ce aveau grad impar în G vor avea grad par.($\star\star$).

Din ($\star$) și ($\star\star$) $\Rightarrow$ gradul nodului adăugat în graf este par $\Rightarrow$ Toate nodurile din noul graf G' au grad par.

Din punctul **(1)** știm că un grad cu toate nodurile de grad par admite o 2-colorare-fair, deci G' admite o 2-colorare-fair. Acum dacă vom scoate muchiile adăugate, pentru nodurile care în G aveau grad impar vom avea ${c_v}^{-1}(h) = {c_v}^{-1}(k) = 1$, ceea ce satisface proprietatea din enunț, deci graful G admite o 2-colorare-fair.

(1) + (2) $\Longrightarrow$ Graful G admite o 2-colorare-fair, dacă G nu este 2-ring.

b) Dacă G este p-ring $\Longrightarrow$ G nu admite o p-colorare-fair.
Din ipoteză știm că graful G este p-ring, deci știm că toate nodurile sale sunt multipli de p, dar numărul de muchii din graf nu este multiplu de p($\forall p \in \mathbb{N}, p \geq 2$).

Presupunem, prin reducere la absurd, că graful G admite o p-colorare,$\forall p \in \mathbb{N}, p \geq 2$. Din ipoteză știm că G este p-ring, deci toate nodurile sale sunt de forma $p \cdot k$,$\forall p, k \in \mathbb{N}, p \geq 2$.

Fie M_i-mulțimea nodurilor colorate cu aceeași culoare i,$\forall i \in C$,C=mulțimea culorilor.
Putem afirma că $|M_{i_1}| = |M_{i_2}| = |M_{i_3}| = ... = |M_{i_n}|$, $\forall i_1, i_2, ..., i_n \in C$, ceea ce ne permite să constatăm că graful G are $|V| = p \cdot k, \forall p, k \in \mathbb{N}, p \geq 2$.

Dacă $|V| = p \cdot k, \forall p, k \in \mathbb{N}, p \geq 2 \implies$ este imposibil să avem toate gradele din G multipli de p dacă numărul de noduri este multiplu de p.

$\Longrightarrow$ CONTRADICȚIE cu presupunerea făcută, deci **dacă graful G este p-ring, atunci acesta nu admite o p-colorare-fair**.

Problema 4

a)

```
iterație=0;
while iterație<500 do
  i=1;
  A'=A;
  while ∃x ∈ A' do
    choose x : A';
    A_i.push_back(x);
    A'.delete(x);
    if i == p then
      i=1;
    else
      i++;
    end if
  end while
  j=1;
  exit=false;
  while j ≤ p do
    nrElem=A_i.dimensiune();
    maxim=-1;
    for s = 1; s ≤ nrElem; s++ do
      for t = 1; t ≤ nrElem; t++ do
        if α(A_i[s], A_i[t]) > maxim then
          maxim = α(A_i[s], A_i[t]);
        end if
      end for
    end for
    if maxim > s then
      exit=true;
      break;
    end if
    j++;
  end while
  if exit == true then
    break;
    iterație++;
  end if
end while
```

Algoritmul rezolvă polinomial problema CLUST-P. În algoritmul prezentat avel un while ce se execută de maxim 500 de ori, acesta conținând alte două bucle while:

-prima buclă are complexitatea O(n), n=nr de elemente ale mulțimii A

-în cea de-a doua buclă se parcurg mulțimile clasterizate și pentru fiecare mulțime se va calcula perechea de noduri pentru care $\alpha(u, v)$ are valoarea maximă în mulțimea respectivă; complexitatea acestei bucle este $O(m \cdot {k_{A_i}}^2)$, unde m=numărul de mulțimi clasterizate, k_{Ai}=dimensiunea fiecărei mulțimi. Astfel complexitatea totală este de $O(500(n + m\cdot)) = O(n^3)$.

Deci, problema noastră este în clasa NP. Mai departe vom arăta că o problemă NP-Hard se reduce polinomial la CLUST-P. Vom alege problema NP-Hard K-STABLE_SET.

instanță:G graf, k număr natural

întrebare:$\exists$ k mulțime stabilă în G astfel încât $\bigcup_k A_k = V(G)$

și $A_k \bigcap A_l = \emptyset, \forall k \neq l, \forall 1 \leq k \leq p, 1 \leq l \leq p, l \neq k$.

Aici noi vom uni nodurile care nu respectă proprietatea precizată în enunț($\alpha(u, v) \leq s$). După ce vom uni aceste noduri, vom avea îndeplinită proprietatea $\max_{u,v \in A_i} \leq s$ pentru orice mulțime de noduri.

b)

MatAdiac[nrElem][nrElem];
while $k < nrElem$ **do**
  for $l = 0; l < nrElem; l++$ **do**
    if $\alpha(k, l) > s$ **then**
      $A[k][l] \leftarrow 1$;
    else
      $A[k][l] \leftarrow 0$;
    end if
  end for
  k++;
end while
Bipartit=true;
for $i = 0; i < nrElem; i++$ **do**
  $stableSet[i] \leftarrow -1$;
end for
Coada C;
$C.push(0)$;
while Coada C nu este goala **do**
  $elem = pop(C)$;
  while $j < nrElem$ **do**
    $C.push(j)$;
    if $A[elem][j] == true \&\& stableSet[j] == -1$ **then**
      $stableSet[j] \leftarrow 1 - stableSet[j]$;
    else if $A[elem][j] == true \&\& stableSet[elem] == stableSet[j]$ **then**
      Bipartit=false;
    end if
    j++;
  end while
end while
if Bipartit==true **then**
  return true;
end if
return false;

Cum algoritmul de mai sus este de complexitate polinomială O(n^2), problema din cerință este polinomial rezolvabilă pentru p=2 fixat.