Setul de probleme 1

soluţiile se primesc

miercuri 5 noiembrie între orele 14 şi 16, la cabinetul C-402

29 octombrie 2014

Problema 1.

a) Arătaţi că dacă M este un **mers închis impar** într-un digraf D, atunci M conţine un **circuit impar** în D.

b) Demonstraţi că graful suport al unui digraf tare conex care nu are circuite impare este un graf bipartit.

(**2+2 = 4 puncte**)

Problema 2. S-a propus următorul algoritm (cu timp de lucru polinomial) pentru determinarea numărului de stabilitate $\alpha(G)$ al unui graf G:

GreedyStab(G)
determină un vârf de grad minim v_0 în G;
if $V(G) = \{v_0\} \cup N_G(v_0)$ **then return** 1
else return $1 + \mathbf{GreedyStab}\big(G - (\{v_0\} \cup N_G(v_0))\big)$

a) Arătaţi că algoritmul propus poate greşi oricât de mult: $\forall n \in \mathbf{N}$ există un graf G astfel încât $\alpha(G) - \mathbf{GreedyStab}(G) = n$.

b) Demonstraţi că $\mathbf{GreedyStab}(G) \geq \sum_{v \in V(G)} \frac{1}{1 + d_G(v)}$

şi că dacă avem egalitate atunci $\mathbf{GreedyStab}(G) = \alpha(G)$.

(**2+2= 4 puncte**)

Problema 3. Fie $G = (V, E)$ un digraf, X o multime finită şi $c : E \to 2^X$ o funcţie care asociază fiecărui arc $e = vw \in E$ o submulţime a lui X: $c(vw) \subseteq X$. Funcţia c poate fi extinsă la drumurile lui G, considerând pentru orice drum P al lui G, $c(P) = \emptyset \cup \bigcup_{e \in E(P)} c(e)$ (în particular, dacă $E(P) = \emptyset$ – adică P este un vârf – avem $c(P) = \emptyset$). Pentru orice $v, w \in V$, notăm $\mathcal{P}_{v,w} = \{P | P \text{ drum în } G \text{ de la } v \text{ la } w\}$. Notăm cu $\big|A\big|$ numărul de elemente ale mulţimii A, iar $\#$ este un element care nu aparţine mulţimii X. Considerăm următoarea problemă:

P: *Dat digraful $G = (V, E)$, funcţia c şi $s \in V$, să se determine pentru fiecare $v \in V$, un drum P^*_{sv} astfel încât $\big|c(P^*_{sv})\big| = \min\{\, \big|c(P)\big| : P \in \mathcal{P}_{s,v}\}$.*

Adevărat sau Fals? *"Următorul algoritm rezolvă problema* **P***".*

1. $u[s] \leftarrow \emptyset$; $parent[s] \leftarrow 0$; $S \leftarrow \{s\}$;
for $v \in V - S$ **do**
if $sv \in E$ **then** { $u[v] \leftarrow c(sv)$; $parent[v] \leftarrow s$ }
else { $u[v] \leftarrow X \cup \{\#\}$; $parent[v] \leftarrow -1$ } ;
2. **while** $S \neq V$ **do**
{ find $v^* \in V - S$ s.t. $\big|u[v^*]\big| = \min\{\big|u[v]\big| : v \in V - S\}$;
$S \leftarrow S \cup \{v^*\}$;
for $v \in V - S$ **do**
if $v^*v \in E \,\&\, \big|u[v]\big| > \big|u[v^*] \cup c(v^*v)\big|$ **then**
{ $u[v] \leftarrow u[v^*] \cup c(v^*v)$; $parent[v] \leftarrow v^*$ }
}

(dacă răspunsul este **Adevărat** se va da o demonstraţie; dacă răspunsul este **Fals** se va da un contraexemplu) (**3 puncte**)

Problema 4. Fie $G = (V, E)$ un graf cu $|V| = n$, $|E| = m$, reprezentat cu ajutorul listelor de adiacenţă. O *ordonare* a vârfurilor lui G este o aplicaţie injectivă $\pi : V \to \{1, 2, \ldots, n\}$ ($\pi(v) = i$ are semnificaţia că vârful v se află pe locul i în ordonarea π). π este o *ordonare lexicografică* dacă pentru orice două vârfuri $x, y \in V$ cu $\pi(x) < \pi(y)$, dacă mulţimea

$$D_{\{x,y\}} = \{z \in V | \pi(z) < \pi(x) \text{ şi } z \text{ este adiacent cu exact unul dintre } x \text{ şi } y\}$$

este nevidă, atunci $z_0 \in D_{\{x,y\}}$ cu $\pi(z_0) = \min_{z \in D_{\{x,y\}}} \pi(z)$ satisface $z_0x \in E$ şi $z_0y \notin E$.

a) Arătaţi că algoritmul de mai jos construieşte o ordonare lexicografică:

Lexicographic(G)
iniţializează lista $\mathcal{L}$ de mulţimi având o singură mulţime: V ;
(fiecare mulţime din L se reprezintă ca o listă dublu înlănţuită)
for $i := 1$ to n **do** {
$v :=$ primul vârf al primei mulţimi din $\mathcal{L}$;
şterge v din acea mulţime ;
$\pi(v) := i$;
for each $L_j \in \mathcal{L}$ **do**
înlocuieşte L_j cu $L_j \cap N_G(v)$ urmată de $L_j - N_G(v)$;
şterge din $\mathcal{L}$ mulţimile vide }

b) Argumentaţi că algoritmul se poate implementa în timpul $O(n + m)$.
(**1 +2 puncte**)

Precizări

1. Este încurajată asocierea în echipe formate din 2 studenţi care să realizeze în

comun tema.
2. Depistarea unor soluţii copiate între echipe diferite conduce la anularea punctajelor tuturor acestor echipe.
3. Nu e nevoie să se rescrie enunţul problemelor. Nu uitaţi să treceţi numele şi grupele din care fac parte membrii echipei la începutul lucrarii.
4. Este încurajată redactarea latex a soluţiilor.
5. Nu se primesc soluţii prin e-mail.