Meno: ____________________

Úvod do distribuovaných algoritmov

Zimný semester 2003/2004

skúška 16.12.2003

Za každú otázku je za správnu odpoveď +1 bod, za nezodpovedanie -1 bod a za zlú odpoveď -2 body. Na úspešné absolvovanie skúšky treba kladný počet bodov.

Ak nie je povedané inak, vrcholy poznajú svoje (jednoznačné) identifikátory, vedia lokálne rozlíšiť svojich susedov (a nemajú žiadnu inú informáciu o topológii)a pracujú asynchrónne podľa deterministického algoritmu.

---

1. Podľa definície z prednášky, podmienka P je invariant prechodového systému $\mathcal{S} = (\mathcal{C}, \mathcal{I}, \rightarrow)$, ak $\forall \gamma \in \mathcal{I} : P(\gamma)$ a $\forall \gamma, \delta \in \mathcal{C}$ také, že $\gamma \rightarrow \delta$ platí $P(\gamma) \Rightarrow P(\delta)$.

Majme prechodový systém $\mathcal{S}$ a podmienku P, ktorá je invariantom $\mathcal{S}$. Potom:

□ P platí v každej konfigurácii $\mathcal{S}$.
□ existuje konfigurácia systému $\mathcal{S}$, v ktorej P neplatí.
□ ak je množina počiatočných konfigurácií $\mathcal{S}$ neprázdna, tak P platí v každej konfigurácii $\mathcal{S}$.
□ neplatí ani jedna z uvedených možností.

---

2. Majme prechodový systém $\mathcal{S}$ a podmienku P, ktorá platí v každej konfigurácii každého výpočtu. Potom:

□ P je invariant $\mathcal{S}$. □ P nie je invariant $\mathcal{S}$. □ P môže a nemusí byť invariant $\mathcal{S}$.
□ ak je množina počiatočných konfigurácií $\mathcal{S}$ neprázdna, tak P platí v každej konfigurácii $\mathcal{S}$.

---

3. Výpočet (ako postupnosť konfigurácií) určuje postupnosť udalostí, kde udalosť je prechod medzi dvoma konfiguráciami. Podľa definície relácie $\rightarrow$, udalosť zodpovedá *send*, *receive* alebo *internal* akcii niektorého procesora. Relácia závislosti udalostí $a \prec b$ je tranzitívny uzáver nasledujúcich pravidiel: 1. a je typu *send* a b je príslušný *receive*, 2. a a b sú udalosti na rovnakom procesore a a je vo výpočte pred b.
Podľa definície z prednášky, logické hodiny sú funkcia Θ, ktorá pre každý výpočet priradí udalostiam prvky z dobre usporiadanej množiny tak, že ak $a \prec b$ potom $\Theta(a) < \Theta(b)$.

Ktoré z nasledujúcich definícií **nie sú** logickými hodinami?

□ $\Theta(a) = |\{b : b \prec a\}|$ □ $\Theta(a) = \max\{n : a = a_0 \prec a_1 \prec \ldots \prec a_n\}$
□ $\Theta(a) = 47 + \max\{n : a_0 \prec a_1 \prec \ldots \prec a_n = a\}$
□ ani jedna z uvedených možností

---

4. Logické hodiny, pre ktoré platí $\Theta(a) <_d \Theta(b) \Rightarrow (a \prec b)$, kde Θ zobrazuje udalosti do množiny $(D, <_d)$,

□ neexistujú □ existujú, ale $(D, <_d)$ nemôže byť $(\mathbb{N}, <)$ □ sa dajú počítať Lamportovvým algoritmom
□ ani jedna z uvedených možností

---

5. Orientovaný graf, ktorého vrcholy sú udalosti daného výpočtu a orientovaná hrana spája udalosti a, b ak je splnené niektoré z pravidiel 1. a 2. z príkladu 3., je:

□ planárny □ súvislý □ acyklický □ ani jedna z uvedených možností

---

6. Majme daný graf s ohodnotenými hranami. Nasledujúci algoritmus (pre vrchol v) počíta v každom vrchole jeho vzdialenosť od pevne daného vrchola v_0:

```
Init:
  Dist_v := ∞
  if v = v_0 then
    forall v' ∈ Neigh_{v_0} do
      send ⟨0⟩ to v'
```

```
on receipt ⟨msg⟩ from u:
if Dist_v > msg + w_{u,v} then
  Dist_v := msg + w_{u,v}
  forall v' ∈ Neigh do
    send ⟨Dist_v⟩ to v'
```

Koľko správ sa najviac vykomunikuje na grafe s n vrcholmi, ak $\forall e \in E : w_e \in \{1, 4, 8\}$?

☐ $O(n)$ ☐ $\omega(n)$ a $O(n \log n)$ ☐ $\omega(n \log n)$ a $O(n^k)$ pre nejaké $k > 1$ ☐ $\Omega(2^n)$

---

7. Uvedený algoritmus počíta vzdialenosti medzi všetkými dvojicami vrcholov. Pri zobudení (spontánnom alebo správou) každý vrchol v nastaví $D_v[v] := 0$ a $D_v[w] := \infty$ pre všetky $v \neq w$ a pošle správu $\langle v, 0 \rangle$ všetkým susedom.
Ktorú z nasledujúcich možností treba doplniť namiesto ???? v spracovaní správy $\langle u, d \rangle$ vrcholom v, aby algoritmus korektne fungoval?

```
On receipt ⟨u, d⟩ from w

if d + 1 < D_v[u]
  D_v[u] := d + 1
  for all x ∈ Neigh_v do
    send ???? to x
```

☐ $\langle u, d+1 \rangle$ ☐ $\langle w, d+2 \rangle$ ☐ $\langle w, d+1 \rangle$ ☐ $\langle x, d+1 \rangle$ ☐ $\langle v, d+1 \rangle$

---

8. $f(x)$-traverzovací algoritmus je traverzovací algoritmus, ktorý na objavenie x vrcholov použije najviac $\min\{f(x), n\}$ krokov. Ktorý riadok treba doplniť do nasledujúceho algoritmu, aby výsledkom **nebol** x-traverzovací algoritmus orientovanej hyperkocky $Q(n)$ (t.j. predpokladáme, že každý vrchol má svojich susedov označených číslom dimenzie príslušnej hrany)? Na začiatku iniciátor pošle správu $\langle 1 \rangle$ po linke dimenzie n.

```
on receipt ⟨msg⟩ via u:
send ⟨msg + 1⟩ via f(u, msg), kde .........
```

☐ $f(u, k) = l$, ak $k = 2^l$ a $f(u, k) = f(u, k - 2^s)$, kde $s = \max\{x \mid 2^x < k\}$ inak
☐ $f(u, k) = (l + 47) \bmod n$, ak $k = 2^l$ a $f(u, k) = f(u, k - 2^s)$, kde $s = \max\{x \mid 2^x < k\}$ inak
☐ $f(u, k) = \max\{l \mid 2^l | k\}$
☐ $f(u, k) = \max\{l \mid 2^l | (2^n - k)\}$
☐ ani jedna z uvedených možností

---

9. Z nasledujúcich funkcií vyberte najmenšiu takú, pre ktorú existuje $f(x)$-traverzovací algoritmus v úplnom grafe:

☐ $f(x) = x$ ☐ $f(x) = 2x$ ☐ $f(x) = 4x$ ☐ $f(x) = x\lceil \log(x) \rceil$ ☐ $f(x) = 2x\lfloor \log(x) \rfloor$

---

10. Pre ktorú z uvedených topológií **so zmyslom pre orientáciu** neexistuje x-traverzovací algoritmus (binárny strom má zmysel pre orientáciu, ak každý vrchol vie, ktorý sused je jeho rodič, pravý a ľavý syn):

☐ binárne stromy ☐ hyperkocky ☐ úplné grafy ☐ neplatí ani jedna z uvedených možností

---

11. Uvažujme algoritmus Chang-Roberts, ktorý zvolí za šéfa vrchol s minimálnym ID, spustený na kruhu dĺžky 5 s identifikátormi $1 \to 2 \to 5 \to 3 \to 4 \to 1$. Ktorá z nasledujúcich postupností môže byť postupnosťou správ prijatých vrcholom 1?

☐ $\langle 3 \rangle \langle 2 \rangle \langle 1 \rangle$ ☐ $\langle 5 \rangle \langle 4 \rangle \langle 3 \rangle \langle 2 \rangle \langle 1 \rangle$ ☐ $\langle 2 \rangle \langle 3 \rangle \langle 4 \rangle$ ☐ $\langle 4 \rangle \langle 3 \rangle \langle 3 \rangle \langle 2 \rangle \langle 2 \rangle \langle 2 \rangle \langle 2 \rangle \langle 1 \rangle$ ☐ ani jedna z uvedených možností

---

12. Optimálny algoritmus voľby šéfa na trojrozmernej mriežke s rozmermi $n \times n \times n$ použije:

☐ $\Theta(3)$ správ ☐ $\Theta(n)$ správ ☐ $\Theta(n \log n)$ správ ☐ $\Theta(n^3 \log(n^3))$ správ ☐ $\Theta(n^6)$ správ

---

13. Algoritmus voľby šéfa na anonymnej trojrozmernej mriežke s rozmermi $n \times n \times n$:

☐ neexistuje ☐ existuje iba ak n je nepárne ☐ neexistuje, ak n je nepárne ☐ vyžaduje $\Omega(n^3 \log(n^3))$ správ

---

14. Algoritmus voľby šéfa na anonymnom synchrónnom kruhu

☐ neexistuje ☐ existuje iba ak n je nepárne ☐ neexistuje, ak n je nepárne ☐ vyžaduje $\Omega(n \log n)$ správ

---

15. Pre ktorú z nasledujúcich topológií existuje deterministický algoritmus voľby šéfa v anonymnom asynchrónnom modeli, ak z každej topológie uvažujeme všetky grafy s aspoň 5 vrcholmi. (Koleso je kružnica s pridaným centrálnym vrcholom, ktorý je incidentný so všetkými ostatnými).

☐ stromy ☐ hypekocky ☐ cesty ☐ kolesá

---

16. Optimálny algoritmus voľby šéfa na synchrónnom kruhu s celočíselnými identifikátormi:

☐ neexistuje ☐ použije $O(n)$ bitov ☐ vyžaduje $\Omega(n \log n)$ bitov
☐ použije $O(nb)$ bitov, kde $b \ll n$ je maximálna dĺžka (v počte bitov) identifikátora vrchola

---

17. Uvažujme algoritmus Korach-Kutten-Moran, ktorý pomocou algoritmu na traverzovanie rieši problém voľby šéfa. Aplikujme tento algoritmus na traverzovací algoritmus, ktorý po k presunoch tokenu navštívi $\Theta\left(k^{\frac{2}{3}}\right)$ vrcholov. Ktorý z nasledujúcich výrazov dáva najtesnejší horný odhad počtu správ výsledného algoritmu na voľbu šéfa?

☐ $O(N^{\frac{3}{2}} \log N)$ ☐ $O(N^{\frac{2}{3}} \log N)$ ☐ $O(\sqrt{N^3 \log N})$ ☐ $O(N^3)$ ☐ $O((\log N)^{\frac{2}{3}} N)$

---

18. Uvažujme algoritmus "shout-and-echo" spustený na synchrónnej sieti. Čo platí o strome, ktorý tento algoritmus vytvorí?

☐ je DFS-kostra ☐ je BFS-kostra ☐ môže byť ľubovoľný ☐ ani jedna z uvedených možností

---

19. Uvažujme algoritmus *GHS* na výpočet minimálnej kostry. Nech F je fragment s menom FN a úrovňou L. Nech l_F je hrana odchádzajúca z F s minimálnou váhou $\omega(l_F)$. Nech l_F vedie do fragmentu $F' = (FN', L')$, $L < L'$. Ktoré z nasledujúcich pravidiel sa použije?

☐ F a F' sa spoja do $(FN', L+1)$ ☐ F a F' sa spoja do (FN', L')
☐ F' čaká ☐ F a F' sa spoja do $(\omega(l_F), L+1)$ ☐ ani jedna z uvedených možností

---

20. Uvažujme algoritmus *GHS* na výpočet minimálnej kostry. Ktoré z nasledujúcich správ mohli byť prijaté spiacim procesom? Môže byť viac správnych odpovedí, v tom prípade treba označiť všetky.

☐ $\langle \textbf{connect}, 0 \rangle$ ☐ $\langle \textbf{connect}, l \rangle$, $l > 0$ ☐ $\langle \textbf{test}, FN, lev \rangle$ ☐ $\langle \textbf{initiate}, FN, lev, S \rangle$

---

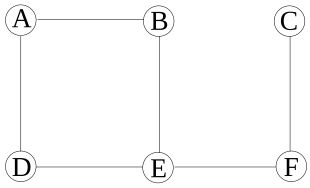
21. Uvedený program je zápisom procedúry *Recompute* z algoritmu *Netchange*, ktorá obnovuje routovacie tabuľky pri zmene topológie. Pri obdržaní správy $\langle \text{mydist}, v, d \rangle$ od w, si proces u nastaví $ndis_u[w, v] := d$ a zavolá *Recompute(v)*. Lokálne premenná $Nb_u[v]$ má obsahovať suseda, do ktorého sa posielajú správy určené pre v. Ktorý z nasledujúcich riadkov treba doplniť namiesto ????, aby algoritmus korektne fungoval?

Procedure *Recompute(v):*
if $v = u$
  $D_u[v] := 0; Nb_u[v] := local$
else
  ????
  if $d < N - 1$
    $D_u[v] := d + 1$
    $Nb_u[v] := w$, kde $ndis_u[w, v] = d$
  else
    $D_u[v] := N; Nb_u[v] := udef$
if $D_u[v]$ sa zmenilo
  forall $x \in Neigh_u$ **do**
    send$\langle \text{mydist}, v, D_u[v] \rangle$ **to** x

☐ $d := \max\{ndis_u[w, v] : w \in Neigh_u\}$ ☐ $d := 1 + \min\{ndis_u[w, v] : w \in Neigh_u\}$
☐ $d := \min\{ndis_u[v, w] : w \in Neigh_u\}$ ☐ $d := \min\{ndis_u[w, v] : w \in Neigh_u\}$
☐ $d := 1 + \min\{ndis_u[v, w] : w \in Neigh_u\}$

---

22. Algoritmus *Netchange* bol spustený na topológii na obrázku. Po dosiahnutí stabilnej konfigurácie sa pridala hrana A–F. Akú správu **nepošle** vrchol A v priebehu ďalšieho výpočtu?



☐ $\langle$**mydist** $, B, 1\rangle$ ☐ $\langle$**mydist** $, C, 2\rangle$ ☐ $\langle$**mydist** $, C, 4\rangle$ ☐ $\langle$**mydist** $, F, 1\rangle$ ☐ ani jedna z uvedených možností

---

23. Uvažujme cayleyho graf $G'(n) = CG(\mathbb{S}_{2n+1}, \{(1,2),(1,3),(4,5),(6,7),\ldots,(2n,2n+1)\})$ (t.j. G' je Cayleyho graf nad grupou permutácií $2n+1$ prvkov). Označme $G(n)$ komponent súvislosti $G'(n)$, ktorý obsahuje identickú permutáciu a N počet vrcholov G. Koľko hopov potrebuje optimálna routovacia schéma na doručenie ľubovoľnej správy v G?

☐ $n-2$ ☐ $n+2$ ☐ $\lfloor \log_2(N) \rfloor$ ☐ $\lfloor \log_3(N) \rfloor + 3$

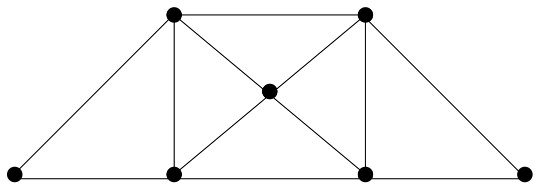
---

24. 8-smerná mriežka je graf, ktorý vznikne z mriežky $n \times n$ pridaním diagonálnych hrán (t.j. vnútorné vrcholy majú stupeň 8). Koľko najmenej intervalov musí mať optimánlna IRS (t.j. taká, ktorá doručuje správy po najkratších cestách) v 8-smernej mriežke?

☐ 1 ☐ 2 ☐ $\geq$3 ☐ závisí od parity n

---

25. V nasledujúcom grafe nájdite optimálnu 1-IRS (t.j. očíslujte vrcholy číslami z $\mathbb{Z}_7$ a ku každej orientovanej hrane pripíšte jeden cyklický interval tak, aby výsledná schéma doručovala správy po najkratších cestách).



---

26. Uvažujme mriežku $n \times n$ a v nej dva nezávisle náhodne vybrané vrcholy. Aká je stredná hodnota ich vzdialenosti?

☐ $\frac{2(n^2-1)}{3n}$ ☐ $\frac{(n-2)(n-1)}{6n}$ ☐ $\frac{n}{4}$ ☐ ani jedna z uvedených možností

---

27. Koľko minimálne bufferov treba na zabezpečenie vyhýbania sa uviaznutiu v mriežke $n \times n$?

☐ 1 ☐ 2 ☐ 3 ☐ $O(\log n)$ ☐ $O(n)$

---

28. Uvažujme problém dohody v synchrónnom úplnom grafe s chybnými hranami, pričom v chybná hrana môže v každom kroku stratiť správu, ktorá po nej bola poslaná. Koľko najviac hrán môže byť chybných, aby existoval deterministický algoritmus na n-vrcholovom ($n > 3$) grafe?

☐ 0 ☐ 1 ☐ $\lfloor n/3 \rfloor - 1$ ☐ $\lfloor n/2 \rfloor - 1$ ☐ $n-2$ ☐ n

---

29. Uvažujme algoritmus EIG, ktorý rieši problém dohody v prítpmnosti stop chýb. Algoritmus v každom vrchole vytvorí strom, v ktorom každý vrchol zodpovedá trase jednej správy a postupne ho zapĺňa hodnotami. Majme tri procesy 1,2,3, ktoré začnú so vstupnými hodnotami 0,0,1. Predpokladajme, že proces 3 pošle v prvom kole správu procesu 1 a havaruje. Doplňte označenia k vrcholom stromu v procese 2, ako bude vyzerať po skončení algoritmu.