OBLIKOVANJE I ANALIZA ALGORITAMA — 1. kolokvij

27. 10. 2010.

1. (10) Između ponuđenih odgovora

$$\Theta(1),\quad \Theta(\lg n),\quad \Theta(n),\quad \Theta(n \lg n),\quad \Theta(n^2),\quad \Theta(n^2 \lg n),\quad \Theta(n^3),\quad \Theta(2^n),$$

nađite točan red veličine za broj koliko puta se izvršava naredba `x = x + 1` u svakom od sljedećih dijelova programa (`/` je operator cjelobrojnog dijeljenja, kao u C-u):

(a)
```
for i = 1 to n / 2
  for j = 1 to 2 * i
    for k = 1 to i
      x = x + 1;
```

(b)
```
for i = 1 to n {
  j = 1;
  while (j <= i) {
    for k = 1 to n
      x = x + 1;
    j = j * 2;
  }
}
```

Ukratko **argumentirajte** odgovore!

2. (10) Zadana je rekurzivna relacija

$$T(n) = 2\,T(n/3) + f(n), \quad f(n) = n,$$

uz početni uvjet $T(1) = d > 0$. Nađite uvjetno asimptotsko ponašanje relacijom Θ za rješenje $T(n)$, ako je n potencija od 3. Može li se dobiveno rješenje proširiti tako da asimptotsko ponašanje vrijedi bezuvjetno, za svaki dovoljno veliki $n \in \mathbf{N}$, za rekurziju

$$T(n) = T(\lfloor n/3 \rfloor) + T(\lceil n/3 \rceil) + n, \quad \text{za } n \geq 3,$$

uz početne uvjete $T(1) = T(2) = d > 0$?

3. (20) Neka je $a[1..n]$ uzlazno sortirano polje od n međusobno **različitih** cijelih brojeva. Brojevi $a[i]$ mogu biti i negativni. Kažemo da je indeks i **fiksna točka** polja a, ako vrijedi $a[i] = i$.

(a) Sastavite algoritam koji provjerava postoji li fiksna točka polja a. Ulazni argumenti algoritma su polje a i broj elemenata n. Algoritam treba vratiti 1 (istina) ako postoji fiksna točka polja a. U protivnom, treba vratiti 0 (laž). Red veličine složenosti algoritma mora biti $O(\log n)$ u najgorem slučaju. Analizirajte složenost vašeg algoritma i pokažite da ona zadovoljava ovaj uvjet.

Zamislite da, umjesto fiksne točke, provjeravamo postoji li indeks i takav da je

(b) $a[i] = 2i$, (c) $a[i] = \lfloor i/2 \rfloor$.

Hoće li odgovarajuća modifikacija algoritma iz (a) raditi korektno u ova dva slučaja? Ako da, argumentirajte. U protivnom, pokažite primjerom što se događa.

OKRENITE!

4. (20) Zadano je potpuno binarno stablo T s n čvorova, gdje je $n = 2^d - 1$, za neki $d \geq 1$. Svaki čvor v tog stabla označen je nekim realnim brojem x_v. Možete pretpostaviti da su sve ove oznake međusobno **različiti** realni brojevi. Kažemo da je čvor v **lokalni minimum** u T, ako je njegova oznaka x_v manja od oznake x_u, za sve čvorove u s kojima je v izravno pozvezan bridom (granom) u T. Uočite da takvih susjednih čvorova u može biti najviše 3 (roditelj, lijevo i desno dijete od v). Treba naći neki (bilo koji) lokalni minimum od T.

(a) Pokažite da svako potpuno binarno stablo T ima lokalni minimum. U trivijalnom binarnom stablu ($n = 1$), korijen je, po definiciji, lokalni minimum.

Oznake čvorova nisu posebno popisane pa im ne možemo izravno pristupati, već su "ugrađene" u sam čvor. To znači da, za bilo koji čvor v, oznaku x_v možemo očitati samo tako da "ispitamo" ili posjetimo taj čvor, po pravilima obilaska binarnog stabla — obilazak počinje u nekom čvoru, a pomak iz čvora dozvoljen je samo u neki susjedni čvor.

Možete uzeti da je svaki čvor stabla opisan strukturom tipa `node`, koja sadrži oznaku čvora x_v i dvije adrese, `left` i `right`, lijevog i desnog djeteta. Listovi stabla prepoznaju se po posebnoj adresi `NULL`, kao signal da nemaju djece. Po potrebi, možete u strukturu `node` dodati i treću adresu `parent`, za roditelja svakog čvora (korijen se onda prepoznaje po `NULL` adresi roditelja).

(b) Pretpostavimo da obilazak stabla T počinje u **korijenu** stabla, zadanom kao ulazni argument `root`. Sastavite algoritam koji nalazi i vraća neki lokalni minimum tog binarnog stabla. Složenost algoritma mjerimo brojem "ispitivanja" (ili posjeta) čvorova. Početni čvor se, također, broji. Red veličine složenosti algoritma mora biti $O(n)$ u najgorem slučaju. Analizirajte složenost vašeg algoritma i pokažite da ona zadovoljava ovaj uvjet. Napomena i uputa: pripazite na to da algoritam radi za mala stabla, kad je $d = 1, 2, 3$.

(c) Uzmimo da obilazak može početi u **bilo kojem** čvoru stabla T, a ne samo u korijenu, s tim da u svakom čvoru pamtimo i adresu roditelja. Početni čvor zadaje se kao ulazni argument algoritma. Može li se algoritam iz (b) jednostavno modificirati tako da radi i u ovom slučaju, a da za složenost i dalje vrijedi ista ocjena u najgorem slučaju? Ako da, pokažite kako. U protivnom, argumentirajte primjerom što se može dogoditi.

Napomena: Broj bodova ovisi o složenosti algoritma. Složenost $O(n)$ vrijedi najviše 10 bodova. **Bonus**: složenost $O(\log n)$ vrijedi 10 bodova više!

REZULTATI: petak, 29. 10. 2010. u 12 sati. Saša Singer