OBLIKOVANJE I ANALIZA ALGORITAMA — popravni kolokvij

23. 1. 2013.

1. (10) Neka su $f, g : \mathbf{N} \to \mathbf{N}$ dvije funkcije. Napišite preciznu definiciju asimptotske relacije ponašanja

$$f(n) \in \Omega(g(n)) \quad (n \to \infty).$$

Koja svojstva ima relacija Ω (u smislu relacije uređaja)? Navedite jedan primjer izražavanja složenosti nekog problema ili algoritma relacijom Ω i opišite riječima odgovarajući zaključak.

2. (10) Zadana je rekurzivna relacija

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

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

$$T(n) = 2\,T(\lfloor n/3 \rfloor) + T(\lceil n/3 \rceil) + n, \quad \text{za } n \geq 2,$$

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

3. (35) Banka je zaplijenila n bankovnih kartica, sumnjajući na pokušaj prijevare. Svaka kartica pripada točno jednom bankovnom računu i sadrži kôdirani zapis podataka o tom računu na čipu i magnetskoj traci. Dvije kartice su **ekvivalentne** ako pripadaju istom računu. Banka treba provjeriti postoji li među zaplijenjenim karticama podskup od strogo **preko** $n/2$ kartica koje su sve međusobno ekvivalentne, tj. pripadaju istom računu.

Izravno dekôdiranje zapisa na kartici je vrlo skupo. Zato banka ima visokotehnološki uređaj koji prima dvije kartice i samo **provjerava** jesu li te dvije kartice ekvivalentne ili ne. Algoritamski rečeno, za kartice s rednim brojevima i, j, ova provjera odgovara pozivu funkcije $iste(i, j)$, koja vraća logičku vrijednost — 0 (laž), ili 1 (istina).

Sastavite algoritam koji, za zadani broj kartica n, vraća odgovor na postavljeno pitanje koje zanima banku. Složenost algoritma mjerimo brojem poziva funkcije *iste*.

Red veličine složenosti algoritma mora biti $O(n \log n)$. Analizirajte složenost vašeg algoritma i pokažite da ona zadovoljava ovaj uvjet. Uputa: iskoristite princip "podijeli, pa vladaj" na "polovinama" zadanog niza kartica (od 1 do $n/2$, i preostalim).

OKRENITE!

4. (30) Poznata web-tražilica El Goog troši veliku količinu vremena svaki puta kad slaže svoj indeks stranica. Tvrtka ima na raspolaganju samo **jedno** super-računalo i gotovo neograničen broj osobnih računala.

Cjelokupno slaganje indeksa sastoji se od n poslova koji se mogu obaviti potpuno nezavisno jedan od drugog. Svaki pojedini posao P_i ima dva dijela: prvo treba obaviti **pripremu** na super-računalu i ta faza traje s_i jedinica vremena, a zatim se posao **završava** na nekom osobnom računalu i ta faza traje t_i jedinica vremena.

Pripremna faza je toliko složena da super-računalo može obavljati samo **jedan** posao u danom trenutku. Čim je taj dio posla gotov, posao se trenutno prebacuje na neko osobno računalo. Istog trena, bez prekida, počinje izvršavanje sljedećeg posla na super-računalu. Osobnih računala ima barem n, tako da se završna faza svih poslova može obaviti potpuno paralelno — svi poslovi se mogu istovremeno završavati na raznim osobnim računalima.

El Goog treba naći **redosljed** kojim treba izvršiti poslove na super-računalu, tako da **ukupno** trajanje slaganja cijelog indeksa — od početka prvog posla na super-računalu, do završetka zadnjeg posla na nekom osobnom računalu, bude **najkraće** moguće.

(a) Analizirajte slučaj $n = 2$ i nađite **pohlepni** kriterij koji daje najmanje trajanje slaganja cijelog indeksa. Dokažite optimalnost tog kriterija za bilo koji n.

(b) Sastavite algoritam koji nalazi optimalni redosljed poslova na super-računalu i nađite njegovu složenost. Ulazni argumenti su broj poslova n i polja vremena s, t. Izlaz algoritma su permutacija p brojeva od 1 do n, koja sadrži optimalni redosljed poslova ($p[j] = i$ znači da posao s indeksom i treba obaviti kao j-ti po redu), i pripadno najmanje trajanje slaganja indeksa. Složenost algoritma mora biti $O(n^2)$.

5. (35) Zadana su dva polinoma A i B s kompleksnim koeficijentima.

(a) Opišite osnovne korake **brzog** algoritma za računanje produkta $C = A \cdot B$ zadanih polinoma, korištenjem brze diskretne Fourierove transformacije (FFT) vektora čija duljina je potencija broja 2. Kolika je složenost tog algoritma?

(b) Ukratko opišite što radi diskretna Fourierova transformacija vektora duljine $n = 2^k$. Skicirajte rekurzivni algoritam za **brzu** diskretnu Fourierovu transformaciju (FFT) i izvedite njegovu složenost, uz pretpostavku sekvencijalnog izvršavanja operacija.

(c) Ukratko opišite kako se dobiva algoritam za inverznu Fourierovu transformaciju.

Napomena: odgovori na pojedine dijelove zadatka vrednuju se nezavisno, tj. ako ne znate odgovoriti na (a) dio, smijete odgovoriti na ostale dijelove.

REZULTATI: petak, 25. 1. 2013. u 12 sati. Saša Singer