OBLIKOVANJE I ANALIZA ALGORITAMA — 2. kolokvij

27. 1. 2016.

1. (20) Zadano je n poslova. Svaki posao je zadan kao vremenski interval realnih brojeva, $P_i = [p_i, k_i]$, za $i = 1, \ldots, n$, gdje je p_i početak, a k_i kraj i-tog posla. Pretpostavljamo da svi poslovi imaju pozitivno trajanje, tj. da je $p_i < k_i$, za $i = 1, \ldots, n$. Različiti poslovi smiju imati presjek pozitivne duljine (trajanja).

Ove poslove treba rasporediti na m izvršitelja, i to tako da se poslovi koje radi svaki **pojedini** izvršitelj smiju preklapati u samo **jednoj** točki (kraj jednog posla smije biti početak drugog posla). Možete zamisliti da jedan posao predstavlja nastavu iz nekog kolegija (zadanu početkom i krajem), a izvršitelji su predavaonice u koje treba rasporediti svu zadanu nastavu, tako da nema preklapanja nastave ni u jednoj predavaonici.

(a) Koliki je **najmanji** broj izvršitelja m_P potreban da se korektno izvrše svi poslovi, uz zadani uvjet? Precizno argumentirajte! Sastavite algoritam koji, za zadani niz poslova P, nalazi i vraća najmanji potreban broj izvršitelja m_P. Nađite složenost tog algoritma u ovisnosti o n.

Zadan je niz poslova P i raspoloživi broj izvršitelja m. Izvršitelje numeriramo brojevima od 1 do m, a posao P_i dodjeljujemo izvršitelju j, tako da u izlaznom polju *radi* postavimo $radi[i] = j$.

(b) Sastavite algoritam koji raspoređuje zadane poslove na zadani broj izvršitelja, tako da vraća polje *radi* od n elemenata. Ako je $m \geq m_P$, algoritam treba iskoristiti samo prvih m_P izvršitelja. U protivnom, za $m < m_P$, onim poslovima P_i koje nije moguće izvršiti treba postaviti $radi[i] = 0$. Nije potrebno minimizirati broj ili trajanje poslova koje je nemoguće izvršiti (samo ih označite). Nađite složenost tog algoritma u ovisnosti o n.

Napomena: oba algoritma moraju imati složenost $O(n^2)$ u ovisnosti o n. Dozvoljeno je koristiti iste pomoćne algoritme (funkcije) u oba algoritma (ne treba ih dva puta pisati, jednom je dovoljno).

2. (30) Promatramo problem "segmentnih najmanjih kvadrata": zadan je niz od n točaka u ravnini, $P = (P_1, \ldots, P_n)$, gdje je $P_i = (x_i, y_i)$, za $i = 1, \ldots, n$, s tim da je $x_1 < x_2 < \cdots < x_n$. Ovaj niz treba podijeliti (particionirati) u neki broj segmenata. Svaki **segment** je neki podniz uzastopnih točaka iz P (gledano po indeksima ili x-koordinatama), tj. dovoljno je gledati prvu i zadnju točku

$$S_{i,j} = (P_i, P_{i+1}, \ldots, P_{j-1}, P_j), \quad i \leq j.$$

Segment smije biti i jednočlan.

Ideja je naći takvu particiju zadanih točaka da točke u svakom pojedinom segmentu približno leže na istom pravcu, a ti pravci se smiju razlikovati za razne segmente.

OKRENITE!

Svakom segmentu $S_{i,j}$ **pridružen** je realni broj $e_{i,j} \geq 0$, kojeg možemo interpretirati kao **grešku** najbolje aproksimacije pripadnih točaka iz $S_{i,j}$ — na primjer, pravcem po metodi najmanjih kvadrata. Grešku $e_{i,j}$ dobivamo pozivom funkcije *err* na sljedeći način

$$e_{i,j} = err(i, j, P), \quad i \leq j.$$

Funkciju **ne** treba napisati! Pretpostavljamo da je složenost ovog poziva linearna u broju točaka u segmentu, tj. $\Theta(j - i + 1)$.

Segmenti u nekoj **particiji** od P moraju sadržavati sve točke iz P i ne smiju se sjeći. Za bilo koju particiju od P, **kaznena** funkcija te particije definira se kao **zbroj** grešaka po svim segmentima u toj particiji i tom zbroju još dodamo "kazneni" član = **broj** segmenata u particiji, pomnožen s unaprijed zadanom realnom konstantom $C > 0$. Broj C je kazna za "lomljenje" jednog komada pravca u dva komada (bez te kazne, segmentima duljine najviše 2, dobivamo ukupnu grešku nula). Sastavite algoritme koji nalaze:

(a) **najmanju** vrijednost kaznene funkcije po svim particijama od P,

(b) neku particiju koja daje najmanju vrijednost kaznene funkcije (dovoljno je vratiti broj segmenata u toj particiji i indekse početnih točaka tih segmenata, uzlazno ili silazno sortirano).

Složenost ovih algoritama **mora** biti polinomna u n. Analizirajte složenost algoritama i pokažite da zadovoljavaju ovaj uvjet.

Uputa: Iskoristite dinamičko programiranje, tj. nađite rekurziju za minimalnu vrijednost kaznene funkcije na odgovarajućim potproblemima. U algoritmima iskoristite "memoizaciju" = pamćenje poznatih najmanjih vrijednosti kaznene funkcije za potprobleme.

3. (20) Definirajte što je diskretna Fourierova transformacija (DFT) kompleksnog vektora duljine n i što je inverzna transformacija.

(a) Skicirajte rekurzivni algoritam za **brzu** diskretnu Fourierovu transformaciju (FFT) vektora duljine $n = 2^k$ i izvedite njegovu složenost, uz pretpostavku sekvencijalnog izvršavanja operacija. Ukratko opišite kako se iz ovog algoritma dobiva algoritam za inverznu Fourierovu transformaciju.

Zadana su dva polinoma A i B, stupnja najviše $n-1$, s kompleksnim koeficijentima. Koeficijente svakog polinoma prikazujemo vektorom duljine n.

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

Napomena: odgovori na pojedine dijelove zadatka vrednuju se nezavisno.

REZULTATI: ponedjeljak, 1. 2. 2016. u 12 sati. Saša Singer