

(Huom: viisi tehtävää)

1. Kun on annettuna lukuja sisältävä taulukko $A[1..n]$ sekä luku s , kysytään onko taulukossa kahta lukua, joiden summa on s (s.o. $A[i] + A[j] = s$, missä $1 \leq i < j \leq n$). Esimerkiksi syötteellä $A = [-1, 3, 6, 2, -3]$ ja $s = 5$ vastaus olisi "kyllä", mutta summaa $s = 4$ tuottavaa lukuparia taulukosta A ei löydy. Esitä tehtävän ratkaiseva algoritmi ja arvioi sen aikakompleksisuus suhteessa taulukon alkioden lukumäärään n .

("Brute-force" 3 p, kertaluokkaa tehokkaampi 6 p.)

2. On valittava tehokkain kolmesta saman ongelman ratkaisevasta hajota-ja-hallitsetyypisistä algoritmeista A , B ja C , jotka ratkaisevat ongelman (n -kokoiset tapaukset) seuraavasti:

- Algoritmi A ratkaisee viisi kpl puolta pienempiä osaongelmia rekursiivisesti ja yhdistää niiden ratkaisut lineaarisessa ajassa.
- Algoritmi B ratkaisee kaksi $(n - 1)$ -kokoista osaongelmaa rekursiivisesti ja yhdistää niiden ratkaisut vakioajassa.
- Algoritmi C ratkaisee yhdeksän kpl $n/3$ -kokoisia osaongelmia rekursiivisesti ja yhdistää niiden ratkaisut neliöllisessä ($O(n^2)$) ajassa.

Mikä on kunkin algoritmin aikakompleksisuus asympotoottisen kertaluokan tarkkuudella, ja mikä algoritmeista on tämän perusteella tehokkain? (6 p.)

3. Merkkijonon $A = a_1 a_2 \dots a_m$ alijono syntyy poistamalla siitä nolla tai useampia merkkejä a_i . Jonojen *pisin yhteinen alijono* on jokin mahdollisimman pitkä niille yhteinen alijono. Esimerkiksi "aula" on merkkijonojen "kaatuilla" ja "naulita" pisin yhteinen alijono. Halutaan laskea jonojen $A = a_1 a_2 \dots a_m$ ja $B = b_1 b_2 \dots b_n$ pisimmän yhteisen alijonon *pituus*; merkitään sitä $L(m, n)$. Sen arvo voidaan laskea seuraavilla, jonojen A ja B alkuosille $A_i = a_1 a_2 \dots a_i$ ($0 \leq i \leq m$) ja $B_j = b_1 b_2 \dots b_j$ ($0 \leq j \leq n$) johdettavissa olevilla palautuskaavoilla:

$$\begin{aligned} L(i, j) &= 0, \text{ kun } i = 0 \text{ tai } j = 0. \text{ Muulloin } \dots \\ L(i, j) &= L(i - 1, j - 1) + 1, \text{ jos } a_i = b_j, \text{ ja muuten} \\ L(i, j) &= \max\{L(i - 1, j), L(i, j - 1)\}. \end{aligned}$$

Esitä polynomisessa ajassa toimiva algoritmi jonojen A ja B pisimmän yhteisen alijonon pituuden $L(m, n)$ laskemiseksi, ja arvioi sen aikakompleksisuus. (6 p.)

4. Selitä lyhyesti mutta täsmällisesti

- (a) prioriteettijono
- (b) polynominen palautus
- (c) branch-and-bound
- (d) ϵ -approksimointialgoritmi

(6 p.)

5. Valitse, mihin seuraavista luokista kukin allaolevista väitteistä sijoittuu:

K: Kyllä, väite pitää paikkansa;

LK: Luultasti kyllä – ei ole todistettu, mutta yleisesti oletetaan näin olevan;

LE: Luultavasti ei – tätä ei ole kumottu, mutta yleisesti oletetaan, ettei näin ole;

E: Ei, väite ei pidä paikkaansa.

- (a) Jokin luokan \mathcal{NP} ongelma ratkeaa polynomisessa ajassa.
- (b) Jokainen NP-täydellinen ongelma vaatii pahimmassa tapauksessa ylipolynomisen ratkaisuaajan.
- (c) Jokainen luokan \mathcal{P} ongelma ratkeaa polynomisessa ajassa.
- (d) Jokin NP-täydellinen ongelma ratkeaa polynomisessa ajassa.
- (e) Jos $\mathcal{P} = \mathcal{NP}$, jokainen NP-täydellinen ongelma vaatii pahimmassa tapauksessa ylipolynomisen ratkaisuaajan.
- (f) Jos jokin NP-täydellinen ongelma ratkeaa polynomisessa ajassa, niin $\mathcal{P} = \mathcal{NP}$.

(6 p.)

(Yht. max. 30 p.)