Itä-Suomen yliopisto — Tietorakenteet ja algoritmit 1

Tietojenkäsittelytieteen laitos — Tentti 28.10.2010

Tehtävissä joissa pyydetään kirjoittamaan pseudokoodi, kelpaa myös esim. Java tai C.

---

1. [Yhteensä **24p**] Vastaa lyhyesti seuraaviin kysymyksiin:

   (a) Anna käsitteen "iso-O" täsmällinen määritelmä.

   (b) Kurssilla käsiteltiin linkitetyistä listoista useita eri variaatioita. Mainitse nimeltä kaksi eri tyyppistä listaa, ja luettele joukko-operaatioiden aikavaativuudet, kun joukko on toteutettu käyttämällä kyseisiä listatyyppejä. Valintasi on oltava sellainen, että ainakin joidenkin operaatioiden aikavaativuuksissa näiden kahden valinnan välillä on selkeä ero.

   (c) Mikä on n solmuisen binäärihakupuun minimikorkeus? Entä maksimikorkeus? Kerro lisäksi mitä merkitystä puun korkeudella on.

   (d) Vertaile ketjutusta ja avointa hajautusta käsiteltäessä yhteentörmäyksiä hajautuksessa. Luettele näiden hyviä ja huonoja puolia.

   (e) Mitkä ovat binäärikeon tukemat operaatiot ja niiden aikavaativuudet? Mikä on kekoehto?

   (f) Kurssilla käsiteltiin (1) lisäysjärjestäminen (insertion sort), (2) kekojärjestäminen (heapsort), (3) lomitusjärjestäminen (mergesort) ja (4) pikajärjestäminen (quicksort). Mainitse jokaiselle algoritmille vähintään yksi hyvä puoli, jonka suhteen se eroaa edukseen kolmesta muusta.

2. [Yhteensä **12p**] Erääseen sovellukseen tarvitaan tietorakenne R, jonka hakuavaimet ovat positiivisia kokonaislukuja. Operaatioina tarvitaan $Search(R, k)$, joka kertoo esiintyykö avain k rakenteessa R; ja $Insert(R, k)$, joka lisää avaimen k tietorakenteeseen, jos se ei siellä jo ole. Ehdota sopiva toteutus tietorakenteelle (sanallinen selostus, operaatioiden pseudokoodia ei tarvita), kun talletettavien avainten lukumäärä n on noin 1000000, ja:

   (a) Avaimet ovat välillä $0 \ldots 2^{64} - 1$, ja niiden jakauma on tasainen (kaikkien mahdollisten avainten esiintymistodennäköisyys on sama).

   (b) Avaimet ovat välillä $0 \ldots 2^{64} - 1$, mutta jakaumasta ei tiedetä mitään, ja halutaan toteuttaa myös operaatio $Delete(R, k)$, joka poistaa avaimen k tietorakenteesta.

   (c) Avainten arvoalue on suhteellisen pieni (esim. välillä $0 \ldots m - 1$, missä m on paljon pienenpi kuin n), jakauma on tasainen, ja sallitaan että sama avain voidaan lisätä rakenteeseen useaan kertaan. Myös poistot on sallittu.

   (d) Kuten (b), mutta talletettavien avainten lukumäärää n ei tiedetä etukäteen.

   Perustele valintasi huolellisesti. Hyvä valinta on sellainen, että operaatiot ovat tehokkaita (pahimmassa tapauksessa tai keskimäärin), tilaa tarvitaan mielellään vähän, ja ei käytetä tarpeettoman monimutkaista tietorakennetta. Jos valitsemastasi tietorakenteesta on useita erilaisia muunnelmia, niin kerro täsmällisesti mitä tarkoitat.

3. [Yhteensä **12p**] On annettu kokonaisluku k ja taulukko $A[1 \ldots n]$, johon on talletettu (satunnaiseen järjestykseen) n eri suurta kokonaislukua. Tehtävänä on etsiä taulukosta k pienintä lukua, ja palauttaa ne kasvavassa suuruusjärjestyksessä. Anna ongelmalle kolme *eri* ratkaisua, joiden aikavaativuudet ovat:

   (a) $O(nk)$; (b) $O(n \log n)$ ja (c) $O(n + k \log n)$.

   Kahteen ensimmäiseen riittää lyhyt mutta *täsmällinen* sanallinen selostus. Anna täsmällinen pseudokoodi viimeiseen kohtaan. Saat käyttää apuna kurssilla käsiteltyjä algoritmeja ja tietorakenteita kirjoittamatta niiden pseudokoodia.

   Anna lyhyt perustelu miksi algoritmisi toimivat vaaditussa ajassa. Mikä algoritmeista on mielestäsi paras? Riipuuko valintasi jotenkin parametrin k arvosta? Perustele lyhyesti.

4. [Yhteensä **12p**] Halutaan liittää AVL-puun jokaiseen solmuun x kokonaisluku $x.size$, joka on solmun x jälkeläisten lukumäärä.

   (a) Kirjoita algoritmi (pseudokoodi), jolla *size*-arvot saadaan lasketuksi, kun annetussa AVL-puussa nämä kentät ovat aluksi tyhjiä. Analysoi algoritmisi aikavaativuus.

   (b) Oletetaan, että edellisen kohdan mukaiset *size*-arvot ovat nyt solmuissa. Kirjoita tehokas algoritmi (pseudokoodi), joka annetulla avaimella k palauttaa puussa olevien avainta k pienempien avainten lukumäärän. Avain k ei välttämättä ole itse puussa. Mikään avain ei esiinny puussa useammin kuin kerran. Aikavaativuus?

   (c) Kuvaa lyhyesti (pseudokoodia ei tarvita), miten *size*-arvojen ylläpitäminen voitaisiin ottaa huomioon AVL-puun lisäys- ja poisto-operaatioissa, siten että näiden operaatioiden aikavaativuus ei muutu.

   Arvosteluun vaikuttaa paitsi algoritmin toimivuus, myös tehokkuus.