Tietorakenteet ja algoritmit (syksy 2012)

Harjoitus 6 (12.10.2012)

1. Voitat television tietokilpailussa päävoiton. Voittona on n:stä elektroniikkatuotteesta niin monta kuin jaksat kantaa. Jaksat kantaa korkeintaan k kg. Jokaisella laitteella on arvo a_i euroa ja paino p_i kg, $i = 1, 2, \dots, n$. Anna pseudokoodilla algoritmi, joka maksimoi voittosi arvon.

   Vihje: Branch-and-bound (katso luennoista kalvot kauppamatkustajan ongelmasta).

2. **Kiertopeli:** (Tämä oikeuttaa kahteen laskuharjoituspisteeseen)

   - Pelilautana on 3×3 -ruudukko, joka sisältää luvut $1 - 9$.
   - Pelin alussa luvut ovat ruudukossa satunnaisessa järjestyksessä, esim:

   ```
   5 7 1
   9 3 6
   8 2 4
   ```

   - Pelaajan on tarkoitus järjestää luvut suuruusjärjestykseen:

   ```
   1 2 3
   4 5 6
   7 8 9
   ```

   - Peli etenee *kiertoja* tekemällä, esim:
     kierretään oikeassa alakulmassa olevaa 2 x 2 aliruudukkoa

   ```
   5 7 1 =kierto=> 5 7 1
   9 3 6           9 2 3
   8 2 4           8 4 6
   ```

   - Kierto-operaatio siis liikauttaa neljää lukua yhden askeleen "myötäpäivään"
   - Kierto voidaan suorittaa missä kohtaan pelilautaa tahansa
   - Kierto on ainoa tapa millä lukuja voidaan liikutella

   Suunnittele algoritmi joka saa syötteekseen pelilaudan ja kertoo mitkä kierrot suorittamalla peli ratkeaa. Arvioi algoritmisi aika- ja tilavaativuutta.

   Onko varmaa että ratkaisu löytyy aina? Löytääkö algoritmisi ratkaisun, joka sisältää pienimmän mahdollisen määrän kiertoja? Jos ei niin, miten algoritmiasi tulisi muuttaa jotta pienimmän kiertomäärän ratkaisu löytyisi?

3. 2-3-puu on yksi variaatio tasapainotetusta puusta. Puu on 2-3-puu jos seuraavat ehdot ovat voimassa:

- Jokaisella solmulla lehtisolmuja lukuunottamatta on 2 tai 3 lasta.
- Kaikki lehtisolmut ovat samalla tasolla

Tällä kertaa emme välitä puuhun talletetuista avaimista, vaan päähuomio on puun muodolla ja erityisesti sen korkeuden ja solmumäärän suhteella.

(a) Piirrä suurin ja pienin mahdollinen 2-3-puu, jonka korkeus on 1, 2 ja 3 (eli siis kaikille korkeuksille omaa suurin ja pienin puunsa).

(b) Oletetaan, että 2-3-puun korkeus on h. Nyt mikä on solmumäärän ala- ja yläraja korkeuden suhteen.

(c) Oletetaan, ett 2-3-puussa on n solmua. Nyt mikä on puun korkeuden ala- ja yläraja solmumäärän suhteen.

4. Puun tehokkuustestaus. Tässä tehtävässä molemmat kohdat ovat yhden laskaripisteen arvoisia.

(a) Tavallisen binäärihakupuun heikkous on, että puun korkeus voi kasvaa suureksi. Jos puuhun tulevat avaimet voidaan lisätä siihen yhdessä erässä, yksi ratkaisu ongelmaan on sekoittaa avaimet ensin satunnaiseen järjestykseen.

Käytetään edellisen viikon binäärihakupuuta ja tutkitaan puiden korkeutta eri avainten määrällä, kun avaimet sekoitetaan ennen lisäystä.

Alkioiden sekoittaminen on Javassa helpoa:

- laita puuhun laitettavat avaimet ArrayList:iin
- sekoita lista kutsu metodia `Collections.shuffle(lista)`

Mittaa useilla eri lisättyjen avainten määrillä miten lähellä puun korkeus on optimaalista korkeutta (mikä olikaan optimaalinen korkeus? kenties nyt on aika kerrata monistetta).

Puusi korkeuden mittaamista varten tarvitset tietysti sopivan rekursiivisen metodin, pseudokoodi löytyy myös luentomateriaalista.

(b) Tutustu Javan valmiiseen tasapainoiseen binäärihakupuutoteutukseen, eli luokkaan TreeSet.

Vertaa empiirisesti TreeSet-puun ja oman puusi tehokkuutta. Tee kolme mittaussarjaa:

- Lisää molempiin puihin n kpl satunnaisia avaimia. Mittaa lisäämiseen kuluva aika.
- Lisää molempiin puihin satunnaisia solmuja m kpl ja suorita n kpl satunnaisia search-operaatiota. Mittaa ainoastaan searcheihin kuluva aika.
- Lisää molempiin puihin solmut $1, 2, 3, \ldots, m$ suuruusjärjestyksessä ja suorita n kpl satunnaisia search-operaatiota. Mittaa ainoastaan searcheihin kuluva aika.

Kaikissa mittaussarjoissa kokeillaan useita m:n ja n:n arvokombinaatioita ja jälleen kiinnostuksen kohteena ovat lähinnä isot syötteet. Algoritmin suoritusaikaa voit mitata samaan tapaan kuin viikon neljä mallivastauksissa.

Mitä opimme mittaustuloksista? Miten tulokset suhtautuvat O-analyysiin?