## 58131 Tietorakenteet ja algoritmit (kevät 2015)

### Harjoitus 7 (23.–27.2.2015)

**1.** **(a)** Simuloi AVL-puun *insert*-algoritmia, kun alunperin tyhjään AVL-puuhun lisätään avaimia seuraavassa järjestyksessä: 80, 65, 38, 53, 22, 44, 71, 85, 56, 68, 54. (Simulointi tarkoittaa tässä, että piirrät kuvan puusta aina, kun se on oleellisesti muuttunut.) Simuloi edelleen, miten edellä saadusta AVL-puusta poistetaan avaimia järjestyksessä 22, 68, 38, 80, 71.

**(b)** Kuten kohta (a), mutta lisäykset tehdään järjestyksessä 54, 68, 56, 85, 71, 44, 22, 53, 38, 65, 80, ja poistettavat avaimet ovat 54, 44, 38, 53.

**2.** **(a)** Piirrä mahdollisimman vähän solmuja sisältävä AVL-puu, jonka korkeus on neljä. Piirrä mahdollisimman monta solmua sisältävä AVL-puu, jonka korkeus on neljä.

**(b)** Oletetaan, että AVL-puun kunkin solmun *height*-kentälle varataan kahdeksan bittiä (eli suurin luku on 255). Kuinka monta solmua puussa pitäisi olla, että juuren *height* vuotaisi yli? (Suuruusluokka-arvio riittää.)

**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)** Mikä on suurin mahdollinen määrä solmuja 2-3-puussa, jonka korkeus on $h$? Entä pienin mahdollinen?

**(c)** Mikä on suurin mahdollinen korkeus 2-3-puulle, jossa on $n$ solmua? Entä pienin mahdollinen?

Vihje: Geometrisen sarjan summakaava $\sum_{i=0}^{k} a^k = \frac{a^{k+1}-1}{a-1}$, kun $a \neq 1$.

**4.** Binäärihakupuun operaatioiden *succ* ja *pred* aikavaativuus on $\mathcal{O}(h)$, jossa $h$ on puun korkeus. Esitä pseudokoodin tasolla, miten binäärihakupuun toteutusta voidaan muuttaa niin, että että operaatioiden *succ* ja *pred* aikavaativuudeksi tulee $\mathcal{O}(1)$ ja muiden operaatioiden aikavaativuus säilyy ennallaan.

**5.** Osoita, että mikä tahansa binäärihakupuu voidaan muuntaa miksi tahansa toiseksi samat avaimet sisältäväksi binäärihakupuuksi suorittamalla sopiva jono kiertoja. Arvioi tarvittavien kiertojen lukumäärää. (Tässä puiden siis ei tarvitse olla tasapainoisia.)

*Vihje:* osoita ensin, että mikä tahansa puu voidaan muuntaa oikealle suuntautuvaksi ketjuksi (eli sellaiseksi, että millään solmulla ei ole vasenta lasta).

**6.** *Osavälikyselyssä* halutaan järjestetty lista kaikista niistä binäärihakupuun $T$ avaimista $x$, jotka ovat annetulla osavälillä $p \ldots q$ (eli siis $p \leq x \leq q$). Huomaa, että osavälin päätepisteiden $p$ ja $q$ ei tarvitse kuulua puuhun $T$. Esitä osavälikyselylle tehokas toteutus pseudokoodina. Mikä on algoritmin aikavaativuus? Onko algoritmisi yhtä tehokas myös, jos puu $T$ onkin toteutettu ilman *parent*-viitteitä?