AVL-puut

- eräs tapa tasapainottaa binäärihakupuu siten, että korkeus on $O(\log n)$ kun puussa on n avainta
- pohjana jo esitetyt binäärihakupuiden operaatiot
- tasapainotus vie pahimmillaan lisäajan lisäys- ja poisto-operaatioissa
- siis kaikki operaatiot pahimmassa tapauksessa ajassa $O(\log n)$
- monimutkaisin tällä kurssilla esitettävä tietorakenne
- pitää ymmärtää, mihin ja miten näitä voi käyttää
- pitää ymmärtää tasapainotuksen perusajatus ja menetelmät
- tavoite ei ole kurssin jälkeen muistaa pseudokoodin yksityiskohtia tai osata koodata AVL-puut

Tasapainotuksen perusajatus

- pidetään puu matalana jakamalla avaimet mahdollisimman tasaisesti vasemmalle ja oikealle
- huom. pelkästään juuren tarkasteleminen ei riitä, saman pitää toteutua myös alipuissa
- ei kuitenkaan kannata yrittää pitää puuta esim. melkein täydellisenä, koska rakenteen korjailemiseen menisi aivan liikaa aikaa
- AVL-puissa tasapainotus perustuu alipuiden korkeuserojen minimoimiseen
- kun tasapainoehto rikkoutuu, se voidaan palauttaa suorittamalla pari yksinkertaista operaatiota epätasapainoisen solmun lähiympäristössä

AVL-puissa tasapainotus perustuu alipuiden korkeuteen

- muistetaan, että solmun korkeus on kaarten lukumäärä solmusta syvimpään lehteen sen alla
- puun (tai alipuun) korkeus on sen juuren korkeus
- erityisesti yksisolmuisen puun korkeus on nolla
- tämän kanssa yhteensopivasti sovimme, että tyhjän puun korkeus on -1
- liitetään solmuun x kenttä $x.\,height$
- siis $x.\,height = \max(x.\,left\,.\,height, x.\,right\,.\,height) + 1$

Puu toteuttaa AVL-tasapainoehdon jos kaikilla solmuilla vasemman ja oikean alipuun korkeuksien ero on korkeintaan yksi.

Sanomme puuta AVL-puuksi, jos se toteuttaa AVL-tasapainoehdon. Tällöin sanomme myös, että puu on tasapainossa.

Lause 6.3 Jos AVL-puun korkeus on h, niin siinä on korkeintaan $F_{h+3} - 1$ solmua, missä F_n on n:s Fibonaccin luku.

Korollaari n-solmuisen AVL-puun korkeus on $O(\log n)$.

Tässä siis käytetään Fibonaccin lukuja

$$\begin{aligned} F_0 &= 0 \\ F_1 &= 1 \\ F_n &= F_{n-1} + F_{n-2} \qquad \text{kun } n \geq 2. \end{aligned}$$

Kaava

$$F_n = \frac{1}{\sqrt{5}} \left(\left(\frac{1+\sqrt{5}}{2} \right)^n - \left(\frac{1-\sqrt{5}}{2} \right)^n \right)$$

on helppo todistaa oikeaksi induktiolla. Sen johtaminen sujuu helpoimmin käyttämällä generoivaa funktiota joka ei kuulu tämän kurssi asioihin.

Mielikuvan saamiseksi todetaan, että

- $1/\sqrt{5} \approx 0{,}447$
- $(1+\sqrt{5})/2 \approx 1{,}618$
- $(1-\sqrt{5})/2 \approx$ -0,618 < 1

joten karkeana approksimaationa voidaan todeta

$$F_n \approx 0{,}447 \cdot (1{,}618)^n.$$

Korollaarin todistushahmotelma Olkoon AVL-puun korkeus h ja solmujen lukumäärä n. Lauseen perusteella siis $n \geq F_{h+3} - 1$.

Tehdään loput laskut vain likimääräisesti. Siis suunnilleen pätee

$$\begin{aligned} n &\geq 0{,}447 \cdot 1{,}618^{h+3} - 1 \\ 2{,}236 \cdot (n+1) &\geq 1{,}618^{h+3} \\ \log_2(2{,}236) + \log_2(n+1) &\geq (h+3)\log_2 1{,}618 \\ &\approx 0{,}694 \cdot (h+3) \\ h &\leq 1{,}44 \cdot (\log_2(n+1) + 1{,}16) - 3 \\ &= O(\log n). \end{aligned}$$

Lauseen 6.3 todistus: Induktio puun korkeuden suhteen. Olkoon $S(h)$ pienin solmujen lukumäärä AVL-puussa, jonka korkeus on h.

Määritelmien mukaan $S(-1) = 0 = F_{3+(-1)} - 1$ ja $S(0) = 1 = F_{3+0} - 1$.

Olkoon nyt T AVL-puu, jonka korkeus on $h \geq 1$. Oletetaan, että väite pätee pienemmillä korkeuksilla kuin h.

Puussa T on ainakin

- juuri
- yksi alipuu jonka korkeus $h - 1$
- toinen alipuu jonka korkeus $h - 1$ tai $h - 2$.

Solmuja on siis vähimmillään $1 + S(h-1) + S(h-2)$.

Induktio-oletuksen perusteella

$$\begin{aligned} 1 + S(h-1) + S(h-2) &\geq 1 + (F_{h-1+3} - 1) + (F_{h-2+3} - 1) \\ &= F_{h+2} + F_{h+1} - 1 \\ &= F_{h+3} - 1. \end{aligned}$$

□

AVL-puun insert

Perusidea: oletetaan, että ennen lisäystä AVL-tasapainoehto on voimassa

- suorita lisäys kuten binäärihakupuun perusversiossa
- korjaa muuttuneet $height$-arvot
- jos jossain solmussa tasapainoehto rikkoutui, korjaa tilanne järjestelemällä alipuita

Huomioita:

- $height$-arvot voivat muuttua vain lisätystä solmusta juureen johtavalla polulla
- siis tasapainoehtokaan ei voi mennä rikki muuten kuin tämän polun solmuilla

Osoittautuu että

- tasapaino saadaan kaikissa mahdollisissa tapauksissa korjatuksi suorittamalla yksi tai kaksi kiertoa
- kierto on vakioaikainen toimenpide, jossa yhden solmun, sen lapsen ja lapsen alipuiden linkitystä hieman muutetaan niin että hakupuuominaisuus säilyy mutta korkeudet hieman muuttuvat

Vaikea kohta algoritmissa on sen analysoiminen, millä eri tavoilla puuhun on voinut syntyä epätasapainoinen solmu

- jos epätasapainoisia solmuja on useita, olkoon p niistä lähinnä lisättyä solmua
- ennen lisäystä p, $p.\mathit{left}$ ja $p.\mathit{right}$ olivat kaikki tasapainossa
- oletuksen nojalla $p.\mathit{left}$ ja $p.\mathit{right}$ ovat tasapainossa vielä lisäyksen jälkeenkin, mutta p ei ole
- lisäyksen seuraksena jollain alipuista
  - $p.\mathit{left}\,.\,\mathit{left}$
  - $p.\mathit{left}\,.\,\mathit{right}$
  - $p.\mathit{right}\,.\,\mathit{left}$
  - $p.\mathit{right}\,.\,\mathit{right}$

  korkeus on kasvanut yhdellä
- osoittautuu, että kutakin neljää tapausta varten on oma melko yksinkertainen menettelynsä jolla tasapaino taatusti palautuu (käydään läpi taululla; yksityiskohdat varsinaisessa luentomonisteessa)

AVL-puun insert: yhteenveto

- lisää solmu kuten binäärihakupuun perusversioon
- palaa lisätystä solmusta juurta kohti päivittäen $height$-arvoja
- kun löytyy solmu p jossa tasapainoehto ei päde, tasapainota:
  - jos lisäys alipuussa $p.left.left$ niin tee oikea kierto
  - jos lisäys alipuussa $p.left.right$ niin tee vasen-oikea kaksoiskierto
  - jos lisäys alipuussa $p.right.left$ niin tee oikea-vasen kaksoiskierto
  - jos lisäys alipuussa $p.right.right$ niin tee vasen kierto
- nyt operaation on valmis!

AVL-puun delete

Perusajatus: oletetaan taas, että ennen poistoa puu on tasapainossa

- tee poisto kuten binäärihakupuun perusversiossa
- samaan tapaan kuin insertissä seuraamme polkua kohti juurta ja korjaamme epätasapainot kierroilla
- liikkeelle lähdetään puusta poistetun solmun vanhemmasta
  - huomaa että poistettu solmu ei aina ole sama solmu jossa poistettava avain alunperin sijaitsi (kaksilapsisen solmun tapaus deletessä)
- toisin kuin insertissä, tasapainotusta voidaan joutua tekemään monessa eri solmussa

- tilanne voidaan taas jakaa erikoistapauksiin sen mukaan, mihin neljästä lapsen alipuusta poisto on kohdistunut
- oikean tasapainotustoimenpiteen löytämiseksi voi olla hyödyllistä miettiä tilannetta siltä kannalta, mikä neljästä alipuusta on liian korkea, ja purkaa tilanne ikään kuin se olisi syntynyt lisäyksen tuloksena
- osoittautuu taas, että samat neljä operaatiota kuin insertin tapauksessa riittävät korjaamaan kaikki mahdolliset tilanteet
- toisin kuin insertin tapauksessa, tasapainotetun solmun korkeus voi muuttua
- voi siis syntyä uusia epätasapainotilanteita puun ylempiin solmuihin, joten pitää jatkaa kohti juurta ja tarvittaessa tasapainottaa uudelleen
- joka tapauksessa kiertoja tulee enintään $O(h) = O(\log n)$ kappaletta

AVL-puut: yhteenveto

- AVL-tasapainoehto takaa, että puun korkeus on $O(\log n)$
- siis kaikki joukko-operaatiot toimivat pahimmassa tapauksessa ajassa $O(\log n)$
- tasapainoehdon pitämiseksi voimassa insert- ja delete-operaatioiden yhteydessä tehdään lisäksi kiertoja
- kierto on vakioajassa tapahtuva toimenpide joka korjaa tasapainoa yhdessä solmussa ja sen lapsissa
- kiertoja tehdään pahimmassa tapauksessa jokaisessa solmussa muuttuneen kohdan ja puun juuren välillä
- siis myös insert ja delete tasapainotuksineen toimivat ajassa $O(\log n)$