Eksamensdisposition - Balancerede binære søgetræer

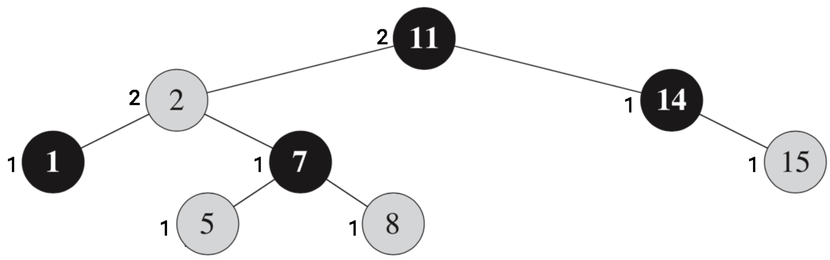
Søren Mulvad, rbn601

3. april 2019

- **Motivation**
- **Struktur for binære søgetræer**
- **Struktur for rød-sorte træer**
- **Bevis for at højden er $O(\lg n)$**
- **Opretholdelse af egenskaber**

1 Balancerede binære søgetræer

- **Motivation**
  - En effektiv datastruktur, som tilbyder worst-case garantier for ting som `Insertion`, `Deletion` og `Search`.
  - Bliver f.eks. brugt i plane-sweep algoritmen (der ser om linjesegmenter skærer hinanden), samt i Linuxkernen og til at understøtte forskellige operationer i flere programmeringssprog.
- **Struktur for binære søgetræer**
  - Attributer for hver knude: *key* samt $left$, *right* og p.
  - Opsat så vi altid har $\leq$ elementer i venstre deltræ og $>$ elementer på højre deltræ.
  - Mange operationer, inklusiv `Insert`, `Search`, `Delete` samt `Max` og `Min` kan nu udføres i $O(h)$ tid. Ved tilfældige indsættelser kan man vise at højden vil blive $O(\lg n)$, men vi er ikke altid garanteret at indsættelserne er tilfældige.
- **Struktur for rød-sorte træer**
  - Ligesom binære søgetræer med den ene ekstra attribut *color*, som kan være rød eller sort. Løser problemet med at garantere en højde $h = O(\lg n)$.
  - Opfylder 5 rød-sorte egenskaber:
    1. Hver knude er enten rød eller sort
    2. Roden er sort
    3. Alle blade er $T.nil$, som er sort
    4. Hvis en knude er rød, så er begge dens børn sorte
    5. For hver eneste knude gælder, at alle simple veje fra knuden til dens efterkommere har det samme antal sorte knuder. Denne kalder vi $bh(x)$.
  - Tegn figuren. Knudernes black-height $bh(x)$ er skrevet til venstre for dem:



- **Bevis for at højden $h = O(\lg n)$ (s. 309)**
  - Jeg vil bevise, at et rødt-sort træ x med n interne knuder har en højde

$$h = O(\lg n)$$

  - Starter med Lemma. Lad $s(x)$ betegne antallet af interne knuder i et deltræ for knuden x. Vi starter så med ved stærk induktion at vise, at dette deltræ har mindst $2^{bh(x)} - 1$ interne knuder.

$$s(x) \geq 2^{bh(x)} - 1 \tag{1}$$

  - Base case (x har en højde $h = 0$):
    * $s(x) = 0$, da x med den højde så må være et eksternt blad.
    * $bh(x) = 0$ og derfor $2^{bh(x)} - 1 = 0$.
    * Vi ser at Eq. (1) er overholdt.
  - Induktionsstep:

* Lad os nu sige knude x har højde $h = k$. Så vil dens to børn $x.left$ og $x.right$ have en højde der højest er $k - 1$. Pga. vores induktionsantagelse ved vi, at Eq. (1) gælder for dem. Vi får nu:

$$
\begin{aligned}
s(x) &= s(x.left) + s(x.right) + 1 && (2)\\
&\geq 2^{bh(x.left)} - 1 + 2^{bh(x.right)} + 1 && (3)\\
&\geq 2^{bh(x)-1} - 1 + 2^{bh(x)-1} - 1 + 1 && (4)\\
&= 2^{bh(x)} - 1 && (5)
\end{aligned}
$$

  I Eq. (3) benytter vi vores induktionsantagelse.
  I Eq. (4) benytter vi, at et barn har en sort-højde $bh(x)$ (rød) eller $bh(x) - 1$ (sort).
  I Eq. (5) lader vi to 1-taller gå ud med hinanden og sætter de potenser sammen.

- Hermed har vi bevist vores Lemma. Nu skal bevise vores påstand. Vi benytter nu egenskab nr. 4, nemlig at hver rød knude har to sorte børn. Dvs. antallet af røde knuder på en vej fra roden ikke kan være med end $h/2$. Derfor må antallet af sorte knuder på sådan en vej minimum være $h/2$. Derfor får vi, at $bh(r) \geq h/2$.
- Nu samler vi disse uligheder hvorved vi får:

$$
\begin{gathered}
n = s(r) \geq 2^{bh(r)} - 1 \geq 2^{h/2} - 1\\
\Updownarrow\\
\begin{aligned}
n + 1 &\geq 2^{h/2}\\
\lg(n+1) &\geq h/2\\
2\lg(n+1) &\geq h\\
h &= O(\lg n)
\end{aligned}
\end{gathered}
$$

- Det er hermed bevist.

Opretholdelse af egenskaber:
Jeg har hermed bevist, at højden på et RB-søgetræ er $O(\lg n)$. For at det skal gøre sig gældende er det meget vigtigt at vi overholder de fem egenskaber. Det kan vi gøre ved:

1. LR (`Left-Rotation`)
2. RR (`Righ-Rotation`)
3. Farve en knude sort eller rød

Vi kan have to cases når vi indsætter eller sletter en knude, det er:

1. Vi overskrider ingen egenskaber og kan bare fortsætte
2. Vi overskrider egenskaber, og skal rette op på det.

Hvis det er case 2, så kan vi løse problemet lokalt, og hvis problemet bliver ved med at probagere op ad træet, helt op til roden er rød, så maler vi den bare sort.