Adatszerkezetek és algoritmusok

Horváth Géza

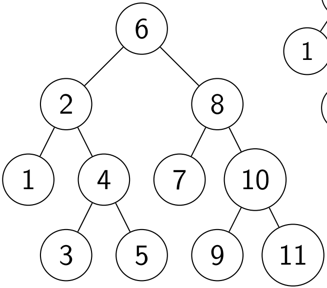
hetedik előadás

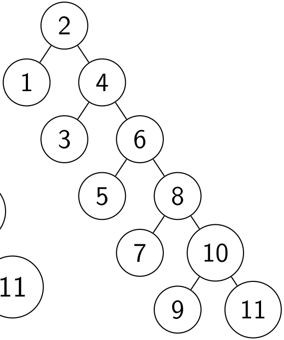
Előadások témái

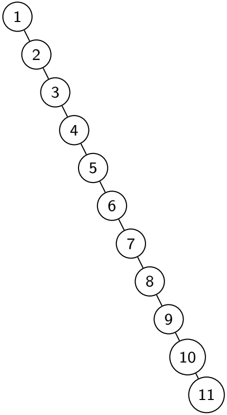
1. Az algoritmusokkal kapcsolatos alapfogalmak bevezetése egyszerű példákon keresztül.
2. Az algoritmusok futási idejének aszimptotikus korlátai.
3. Az adatszerkezetekkel kapcsolatos alapfogalmak. A halmaz, a multihalmaz és a tömb adatszerkezet bemutatása.
4. Az adatszerkezetek folytonos és szétszórt reprezentációja. A verem, a sor és a lista.
5. Táblázatok, önátrendező táblázatok, hash függvények és hash táblák, ütközéskezelés.
6. Fák, bináris fák, bináris keresőfák, bejárás, keresés, beszúrás, törlés.
7. Kiegyensúlyozott bináris keresőfák: AVL fák.
8. Piros-fekete fák.
9. B-fák.
10. Gráfok, bejárás, legrövidebb út megkeresése.
11. Párhuzamos algoritmusok.
12. Eldönthetőség és bonyolultság, a P és az NP problémaosztályok.
13. Lineáris idejű rendezés. Összefoglalás.

Keresőfák

Mennyi idő a kulcs megtalálása?

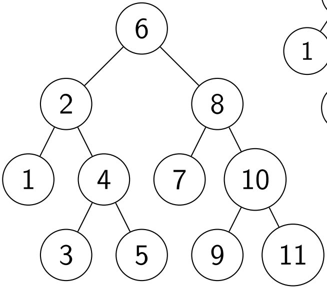




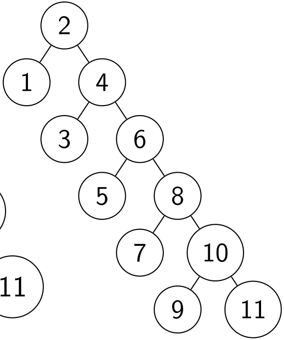


Keresőfák

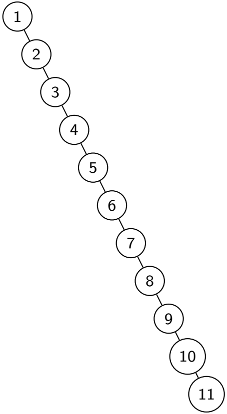
Mennyi idő a kulcs megtalálása?



Logaritmus!

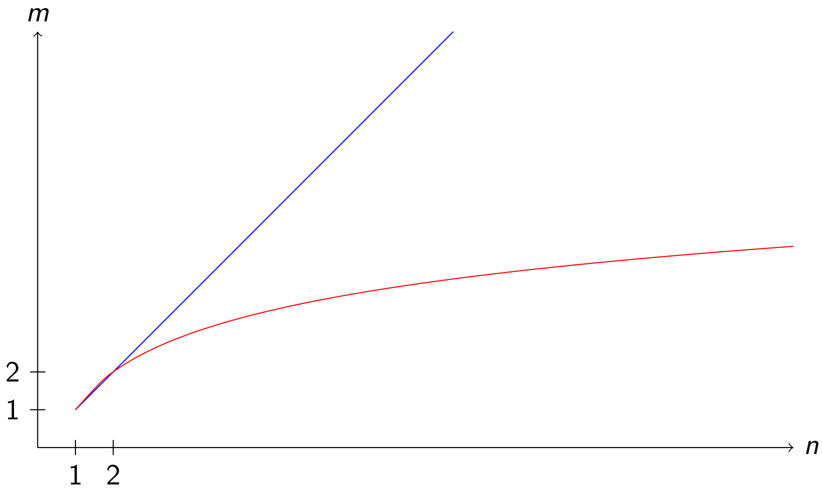


???



Lineáris.

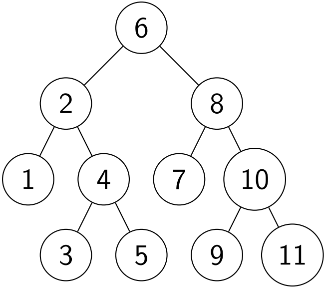
A lineáris és a logaritmus idejű keresés összehasonlítása



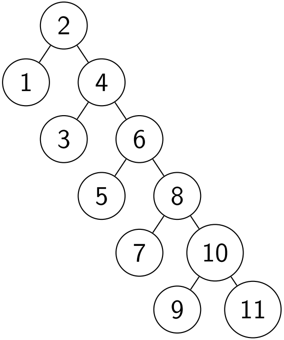
n – **bemenet mérete**

m – **futási idő**

Keresőfák



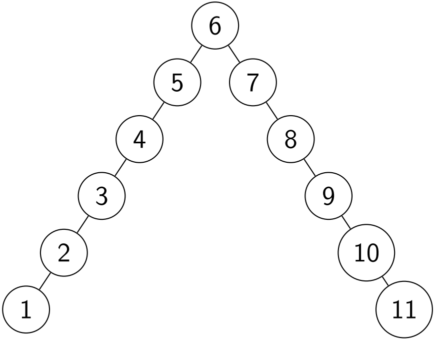
Kiegyensúlyozott bináris keresőfa.

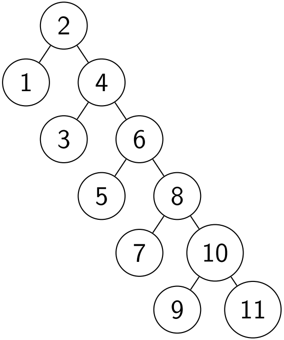


- TREE-SEARCH – $\mathcal{O}(log_2 n)$
- TREE-INSERT – $\mathcal{O}(log_2 n)$
- TREE-DELETE – $\mathcal{O}(log_2 n)$

- TREE-SEARCH – $\mathcal{O}(n)$
- TREE-INSERT – $\mathcal{O}(n)$
- TREE-DELETE – $\mathcal{O}(n)$

Keresőfák





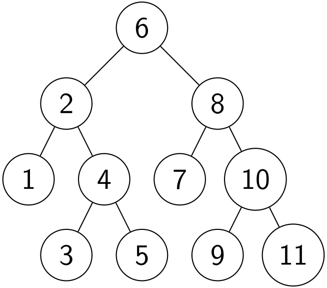
- TREE-SEARCH – $\mathcal{O}(n)$
- TREE-INSERT – $\mathcal{O}(n)$
- TREE-DELETE – $\mathcal{O}(n)$

- TREE-SEARCH – $\mathcal{O}(n)$
- TREE-INSERT – $\mathcal{O}(n)$
- TREE-DELETE – $\mathcal{O}(n)$

Kiegyensúlyozott bináris keresőfa

Definíció

Egy bináris fa kiegyensúlyozott, ha bármely csúcs esetén a baloldali és a jobboldali részfa magassága közötti különbség maximum 1.

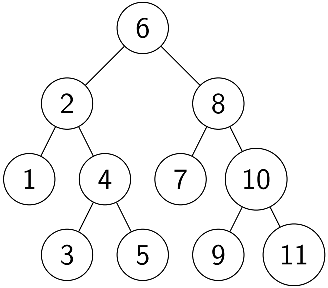


További példák a táblán!

Kiegyensúlyozott bináris keresőfa

Tétel

Kiegyensúlyozott bináris keresőfa esetén a keresés, a beszúrás és a törlés művelete is logaritmus időben – $\mathcal{O}(log_2 n)$ *– megvalósítható.*



Mely műveletek 'ronthatják el' a kiegyensúlyozottságot?

Az AVL fa

Az AVL fát 1962-ben írták le először, nevét felfedezőiről (Georgy Adelson–Velsky és Evgenii Landis) kapta. Az AVL fa olyan kiegyensúlyozott bináris keresőfa, mely elemek hozzáadása vagy törlése után, – szükség esetén, – visszaállítja a kiegyensúlyozottságot.

Műveletek AVL fákkal, mint adatszerkezetekkel

Műveletek:

- adatszerkezetek **létrehozása**: folytonos vagy láncolt reprezentációval
- adatszerkezetek **módosítása**
  - elem hozáadása: TREE-INSERT + <u>REBALANCE</u>
  - elem törlése: TREE-DELETE + <u>REBALANCE</u>
  - elem cseréje: nincs
- elem **elérése**: ITERATIVE-TREE-SEARCH

AVL fa – egyensúly-faktor

Definíció

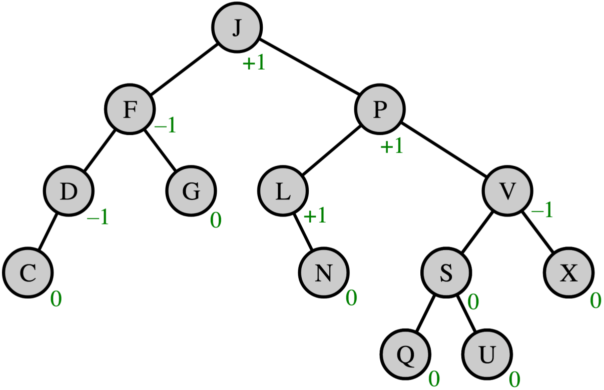
Az AVL fa minden csúcsához tartozik egy szám, ami a jobboldali és a baloldali részfa magasságának a különbsége. Ezt a számot egyensúly-faktornak hívjuk.
BalanceFactor(N)=Height(RightSubtree(N))–Height(LeftSubtree(N)).

Megjegyzés: Azoknak a fáknak a magassága, melyek mindössze egy csúcsot tartalmaznak, definíció szerint 0.

AVL fa – egyensúly-faktor

Definíció

BalanceFactor(N)=Height(RightSubtree(N))–Height(LeftSubtree(N)).



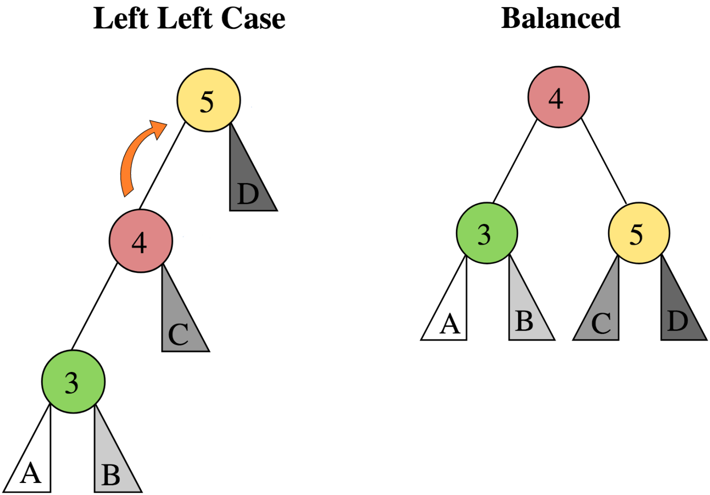
Az AVL fa – forgatás

Az AVL fa módosítása – elem hozzáadása vagy törlése – esetén néha forgatásra van szükség ahhoz, hogy a fa visszanyerje kiegyensúlyozott alakját. Ez nem mindig szükséges, csak abban az esetben, ha a módosítás következtében valamely csúcsának egyensúly-faktora kilép a $[-1, 0, 1]$ intervallumból.

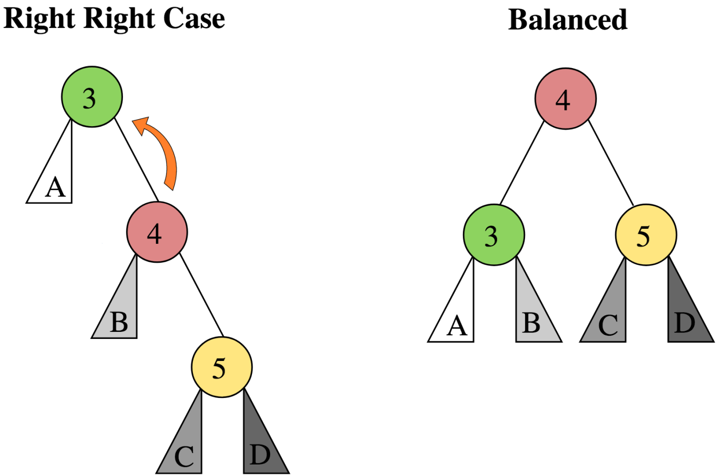
Alapvetően 4 különböző eset fordulhat elő, amikor forgatásra van szükség:

- Left-Left Case (LL)
- Left-Right Case (LR)
- Right-Left Case (RL)
- Right-Right Case (RR)

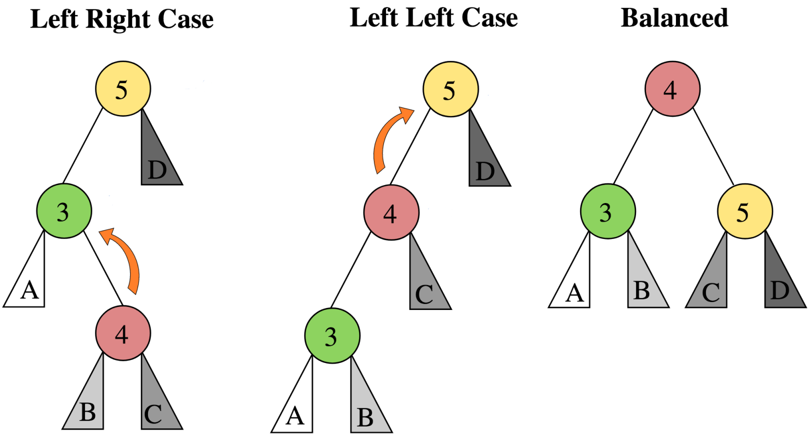
Az AVL fa – forgatás: LL eset



Az AVL fa – forgatás: RR eset



Az AVL fa – forgatás: LR eset



Az AVL fa – forgatás: RL eset