Kombinatorikus optimalizálás 1. hét

Pusztai Pál
pusztai@sze.hu

Tartalom

- Az algoritmusok megadásánál alkalmazott jelölésmód
  - Szöveges megadás, pszeudokód
- Algoritmusok és hatékonyságuk
  - Függvények növekedése, aszimptotikus jelölések
  - NP-teljes problémák
- Gráfok és fák
  - Alapfogalmak
  - Gráfok és gyökeres fák ábrázolása
  - Bináris fák, bináris keresőfák
    - Bejárás, keresés, minimum, maximum
    - Piros-fekete fák

Jelölésmód

- Az algoritmusokat **szövegesen** vagy **pszeudokóddal** adjuk meg.
  - A megoldási lépéseket **sorszámozzuk**.
  - A jelölések követik a két ajánlott irodalom jelöléseit
    - A pszedokódos jelölésünk „egyszerűsített”.
- Szöveges megadás
  - **Matematikai** jelöléseket használunk
    - Pl. $\in$, $\notin$, $\subseteq$, $\cup$, $\cap$, $\setminus$, $|\ |$, $\emptyset$, …
    - A halmazokat a { } zárójelpárral jelöljük, pl. {1, 2, 3}
    - A vektorokat a ( ) zárójelpárral jelöljük, pl. (1, 0, 1, 1)
  - A **változókat** kisbetűvel vagy nagybetűvel, dőlt kiemeléssel írjuk
    - Pl. r, L
  - A **vektorváltozókat** kisbetűvel, a **mátrixváltozókat** nagybetűvel, félkövér kiemeléssel írjuk
    - Pl. $\mathbf{x}$, $\mathbf{b}$, $\mathbf{A}$, $\mathbf{C}^{(n \times n)}$
  - Az **értékadás** jelölése
    - Pl. $i = i + 1$

Jelölésmód

- Pszeudokódos megadás
  - Az algoritmusokat **szubrutinok** (eljárások, függvények) formájában adjuk meg.
  - A **tagolás** a blokkszerkezetet tükrözi.
  - Egy sorba legfeljebb egy utasítást írunk.
  - Az **értékadás**, többszörös értékadás jelölése
    - Pl. $i \leftarrow i+1$
    - Pl. $i \leftarrow j \leftarrow k$ értékadás ekvivalens a $j \leftarrow k$, és az ezt követő $i \leftarrow j$ értékadással.
  - Egy **tömb** elemeire [ ] zárójellel hivatkozunk.
    - Pl: $A[i]$ jelenti az A egydimenziós tömb i-edik elemét.
    - Pl: $B[2, j+1]$ jelenti a B kétdimenziós tömb második sorának $j+1$-edik elemét.
    - Pl: $A[1..j]$ jelenti az $A[1], A[2], \ldots, A[j]$ elemekből álló **résztömböt**.
  - Egy egydimenziós tömb megadása
    - Pl: $A = <5, 3, 8, 1, 9, 7, 2, 6, 4>$
  - Az összetett adatok (objektumok) esetén a **tulajdonságok** (mezők) elérése
    - Pl: *hossz*[A] jelenti az objektumként kezelt A tömb elemeinek számát.

Jelölésmód

- Pszeudokódos megadás (folyt.)
  - Programmá írás
    - A **paraméterátadás** az egyszerű adatoknál értékszerinti, az összetett adatoknál címszerinti.
    - A **változók** a szubrutinok lokális munkaváltozói.
  - A **mutatók** konstansa: NIL (az ábrákon / jelöli).
  - A logikai műveletek **gyors kiértékelésűek**.
  - A **megjegyzéseket** /* és */ jelek közé tesszük.

- A feladatokat tartalmazó diák jobb felső sarkában szereplő  ikon azt jelzi, hogy az a feladattípus szerepelni szokott a vizsgán.

Jelölésmód

- Példa szövegesen megadott algoritmusra (fák szélességi bejárása)
  - Adatszerkezet
    - Input: a fa a gyökérpontjával azonosítva.
    - Output: a fa, minden pontján az elérésre jellemző távolságokkal.
    - Segédszerkezet: sor.
  - Algoritmus

1. Tegyük a sor végére a gyökérpontot, a távolsága legyen 0.
2. Ha a sor üres, vége.
3. Vegyük ki a sor elejéről a soron következő pontot.
4. A kivett pont minden gyerekét tegyük a sor végére, a távolságuk legyen a kivett pont távolsága +1.
5. Újra a 2. lépés következzen.

Jelölésmód

- Példa pszeudokóddal megadott algoritmusra (fák inorder bejárása)

```
INORDER-FABEJÁRÁS(x)
1   if x ≠ NIL
2           INORDER-FABEJÁRÁS(bal[x])
3           Ki: kulcs[x]
4           INORDER-FABEJÁRÁS(jobb[x])
```

Vezérlőszerkezetek (pszeudokód)

- Szekvencia

T_1
T_2
…
T_n

- Szelekció

if f_1
    T_1
else if f_2
    T_2
...
else if f_n
    T_n

- Elöltesztelő iteráció

while f
    T

- Hátultesztelő iteráció

repeat
    T
until f

- Növekményes iteráció

for *cv* ← *ke*, *ve*, *le*
    T

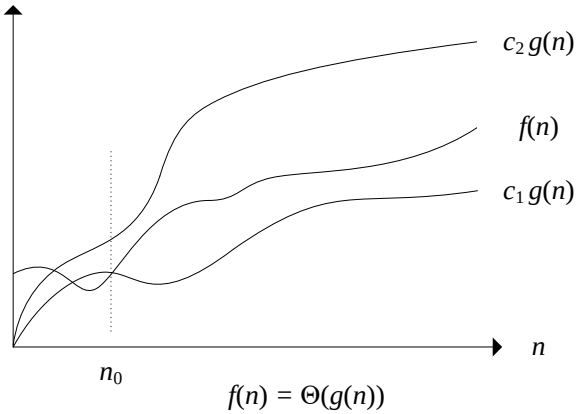
Feladatok

- Írjuk ki a páros számokat 1-től 10-ig mindhárom ciklusfajtával! A kiírást az alábbi utasítással végezzük!
  - Ki: kifejezéslista
  - Pl: Ki: ”Az eredmény:”, er

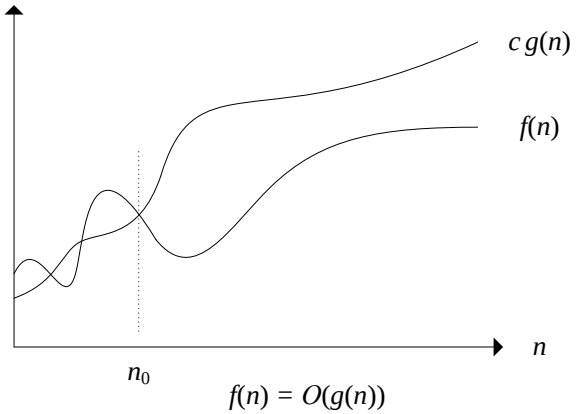
Algoritmusok hatékonysága

- Algoritmus
  - Fogalma
    - Olyan egyértelmű, véges számú tevékenységből álló műveletsor, amely az adott problémakör összes feladatát megoldja (teljes).
  - Megadása
    - Szöveges leírás, pszeudokód, folyamatábra, D-diagram, struktúra diagram, …
- Hatékonyság
  - Elemi művelet (utasítás, lépés)
    - Végrehajtási ideje nem függ a végrehajtásban szereplő adatok nagyságától.
    - Megadható egy olyan korlát, amely mindig nagyobb, mint a művelet végrehajtási ideje.
  - Egy algoritmus elemi műveleteinek száma
    - Becsülhető, az aktuális feladat méretétől (a feladatban résztvevő elemek számától) függően.
    - Megadása általában az algoritmusra jellemző *O* ((nagy) ordó) függvénnyel történik.
  - Hatékonyságok
    - Átlagos hatékonyság, legrosszabb eset hatékonyság, optimális eset hatékonyság

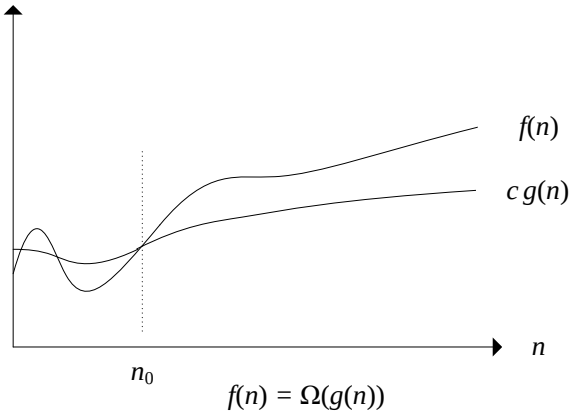
Függvények növekedése



$f(n) = \Theta(g(n))$



$f(n) = O(g(n))$



$f(n) = \Omega(g(n))$

Aszimptotikus jelölések

$\Theta(g(n))=\{f(n)$: léteznek c_1, c_2 pozitív állandók és n_0 úgy, hogy $n \geq n_0$ esetén $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\}$.
$O(g(n))=\{f(n)$: létezik c pozitív állandó és n_0 úgy, hogy $n \geq n_0$ esetén $0 \leq f(n) \leq cg(n)\}$.
$\Omega(g(n))=\{f(n)$: létezik c pozitív állandó és n_0 úgy, hogy $n \geq n_0$ esetén $0 \leq cg(n) \leq f(n)\}$.
$o(g(n))=\{f(n)$: minden c pozitív állandóhoz létezik $n_0>0$ úgy, hogy $n \geq n_0$ esetén $0 \leq f(n) < cg(n)\}$.
$\omega(g(n))=\{f(n)$: minden c pozitív állandóhoz létezik $n_0>0$ úgy, hogy $n \geq n_0$ esetén $0 \leq cg(n) < f(n)$.

Θ: aszimptotikusan **éles** korlát, o, ω: aszimptotikusan **nem éles** korlátok.
O, Ω: éles vagy nem éles korlátok.
$\Theta(1)$, $O(1)$ a konstans függvényeket jelölik.

Függvények növekedése

- Megjegyzés
  - Egy k-adfokú polinom $O(n^k)$ nagyságrendű.
  - Minden $c > 1$ és $k > 1$ egész esetén $c^n = \Omega(n^k)$.
  - $\lg(n!) = \Theta(n \lg n)$

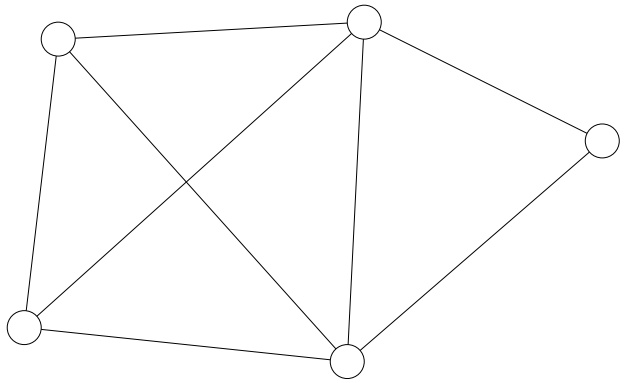
- NP-teljes problémák
  - Még nem ismerünk polinomiális hatékonyságú algoritmust a megoldásukra.
  - Ha bármelyik NP-teljes problémát sikerülne polinomiális időben megoldani, akkor az összes többi is megoldható lenne polinomiális időben.

Feladatok

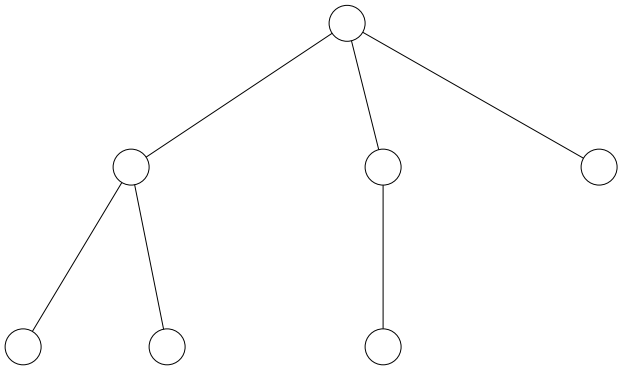
- Igazak-e vagy sem az alábbiak?
  - $0.1n^2-3n+1=\Theta(n^2)$
  - $2n=O(n^2)$
  - $2n=\Omega(n^2)$
  - $2n=o(n^2)$
  - $2n=o(n)$
  - $2n^2=\omega(n)$
  - $2^{n+1}=O(2^n)$
  - $2^{2n}=O(2^n)$
  - $n! = \Omega(n^n)$.
  - $n! = \Omega(2^n)$.
  - $f(n)=\Theta(g(n))$ akkor és csak akkor, ha $f(n)=O(g(n))$ és $f(n)=\Omega(g(n))$.
  - $f(n)=\Theta(g(n))$ akkor és csak akkor, ha $g(n)=\Theta(f(n))$.
  - $f(n)=O(g(n))$ akkor és csak akkor, ha $g(n)=\Omega(f(n))$.
  - $f(n)=o(g(n))$ akkor és csak akkor, ha $g(n)=\Omega(f(n))$.
  - $o(f(n)) \cap \omega(f(n))=\emptyset$

Gráfok

- **Gráfon** bizonyos elemek (**pontok**) és a köztük fennálló közvetlen kapcsolatok (**élek**) halmazát értjük.



Általános gráf



Fa

Gráfok

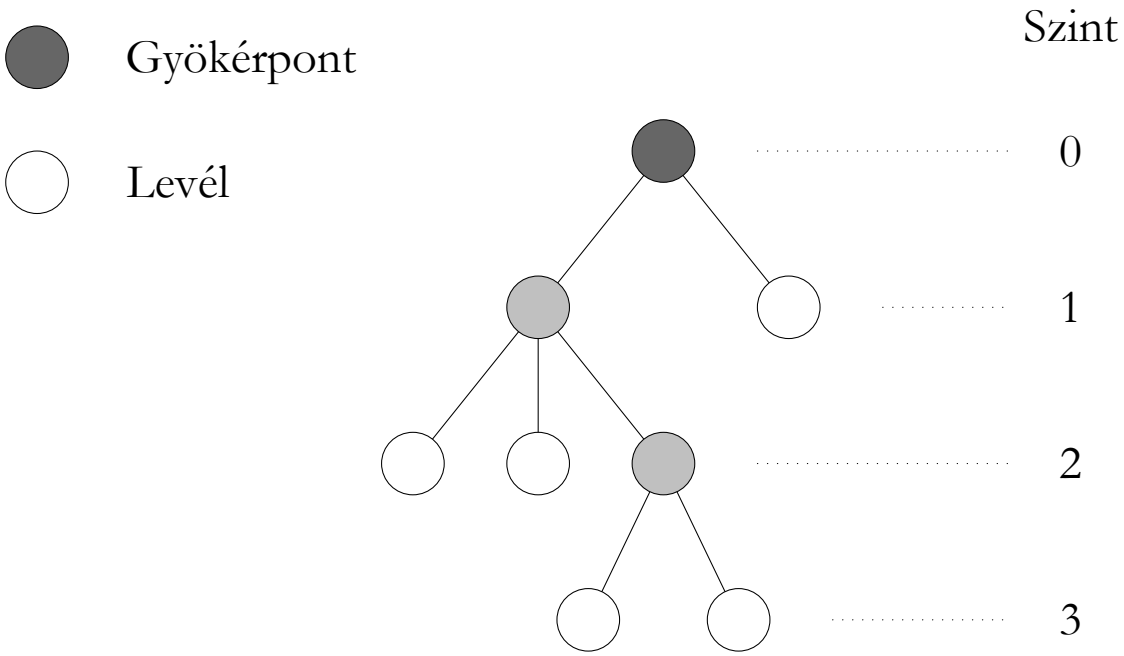
- Alapfogalmak
  - **Irányított gráfról** akkor beszélünk, ha az élekhez irányítást rendelünk.
  - **Egyszerű gráf** az olyan gráf, amelyben bármely két pont között egy irányban legfeljebb egy él van és nincs **hurokél** (egy pont önmagával vett kapcsolata).
  - **Teljes gráf** az olyan gráf, amelyben bármely két pont között van él.
  - **Út** a gráf egy olyan pontsorozata, amelyben a szomszédos pontok között van él.
  - **Körmentes út** egy olyan út, amelyben minden pont különböző, egyébként (ha van ismétlődő pont) az út **körös út**.
  - Egy irányítatlan gráf **összefüggő**, ha bármely két pontja között van út.
  - Egy gráf **részgráfján** értjük a pontok és a köztük lévő élek egy részhalmazát.
  - Egy gráfot **szimmetrikusnak** nevezünk, ha irányítatlan, vagy minden irányított élnek van párja.
  - Egy gráf **súlyozott**, ha az élekhez súlyt ($\in \mathbf{R}$) rendelünk.

Fák

- Fa struktúrájú gráf, fagráf, vagy röviden **fa** az olyan összefüggő, irányítatlan gráf, amelyben nem hozható létre körös út.
- Néhány **ekvivalens** állítás a fákra vonatkozóan
  - Bármely két pont között egyértelműen létezik egy őket összekötő út.
  - Éppen eggyel kevesebb él van, mint pont.
  - Akár csak egy élt hozzávéve az élekhez, a kapott gráf már tartalmaz kört, azaz nem lesz fa.
  - Akár csak egy élt is elhagyva az élek közül, a kapott gráf már nem lesz összefüggő, azaz már nem lesz fa.

Gyökeres fák

- **Gyökeres fa** az olyan fa, amelyben egy pontot kitüntetünk. Ezt a fa **gyökérpontjának** nevezzük.



Gyökeres fa

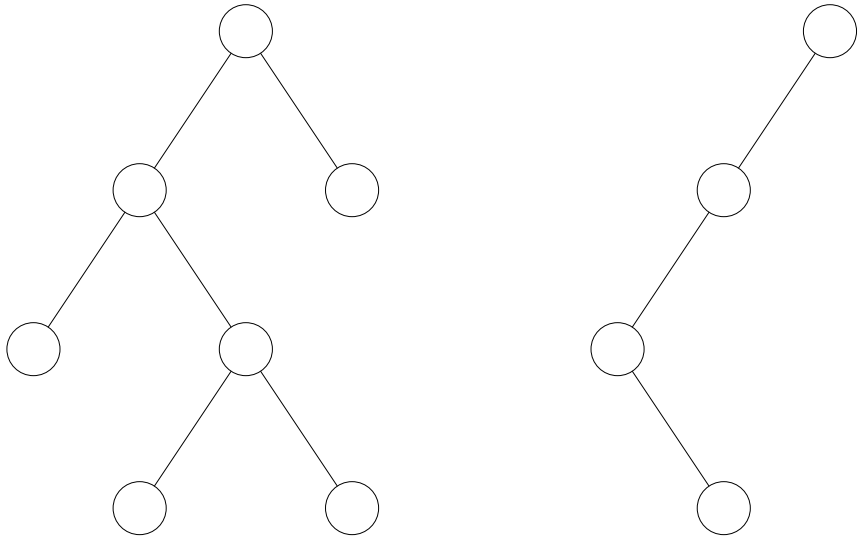
Gyökeres fák

- A gyökeres fák jellemzői
  - A gyökérpont kivételével minden pontnak egyértelműen definiálható a **megelőzője** (szülő, ős), a gyökérpontból hozzávezető úton, az őt megelőző pont.
  - Egy pont **követője** (gyermek, leszármazott) az a pont, amelynek ő a megelőzője.
  - Egy fa bármely pontja, ha őt gyökérpontnak tekintjük, szintén meghatároz egy fát. Ezt a fát az eredeti fa **részfájának** nevezzük.
  - **Rendezett fa** az olyan fa, amelyben a pontok leszármazottai között sorrendet értelmezünk, azaz beszélünk első, második, stb. leszármazottról.

- Megjegyzés
  - A továbbiakban fán gyökeres fát értünk.

Bináris fák

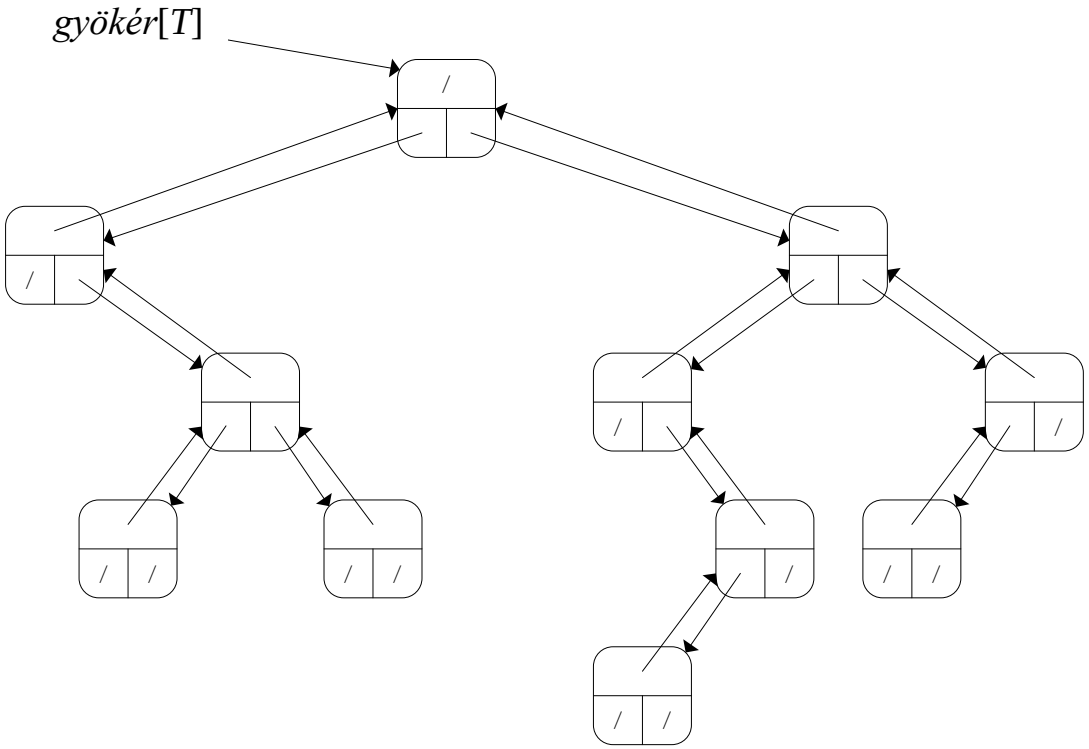
- **Bináris fa** az olyan rendezett fa, amelyben egy pontnak legfeljebb két leszármazottja van.
- A leszármazottakat **bal**, ill. **jobb leszármazottnak**, az általuk meghatározott részfát **bal**, ill. **jobb részfának** nevezzük.



Bináris fák

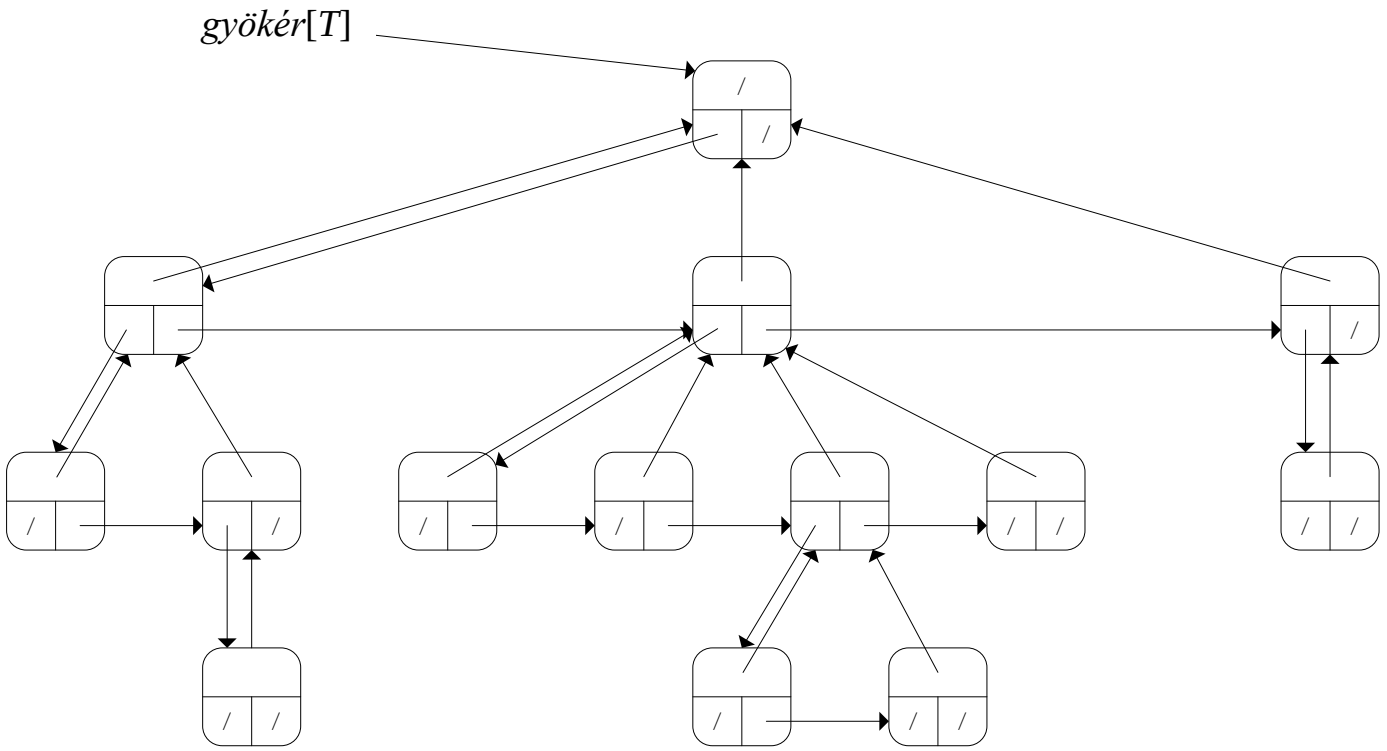
Bináris fák ábrázolása

- A bináris fákat láncolt struktúraként ábrázolhatjuk, ahol minden csúcs egy önálló objektum.
- A *kulcs* mező és a kísérő adatok mellett minden csúcs tartalmaz egy *bal*, egy *jobb* és egy *szülő* nevű mezőt is, amelyek rendre a csúcs bal- és jobboldali gyerekére, illetve a szülőjére mutatnak.



Bináris fa ábrázolása

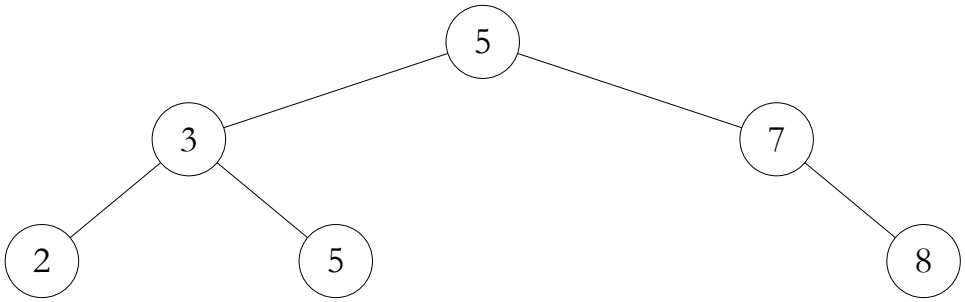
Gyökeres fák ábrázolása

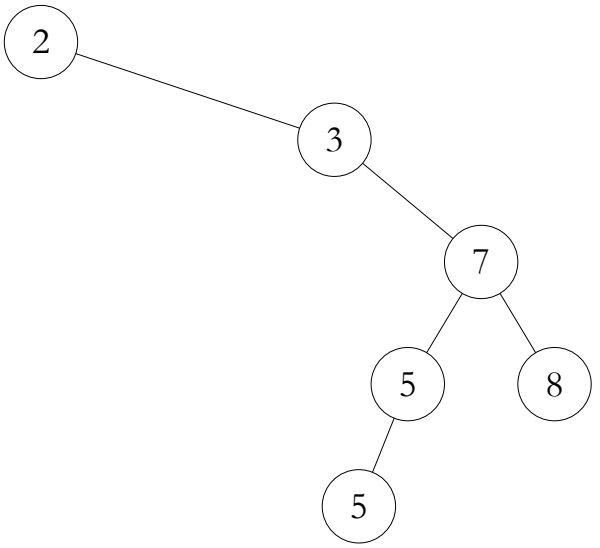


Bal-gyermek, jobb-testvér reprezentáció

Bináris keresőfák

- A **bináris-kereső-fa tulajdonság**: Legyen x az adott bináris keresőfa egy tetszőleges csúcsa. Ha y az x baloldali részfájának egy csúcsa, akkor $kulcs[y] \leq kulcs[x]$, ha pedig y az x jobboldalára esik, akkor $kulcs[x] \leq kulcs[y]$.
- A **keresőfák** olyan adatszerkezetek, amelyekre értelmezettek a dinamikus halmaz műveletek.
- Dinamikus halmazok
  - A számítástechnikában használatos, időben változó, véges halmazokat **dinamikus halmazoknak** nevezzük.
  - A dinamikus halmazokon értelmezett **műveletek**: KERES, BESZÚR, TÖRÖL MINIMUM, MAXIMUM, ELŐZŐ és KÖVETKEZŐ.





Bináris keresőfák

Bináris keresőfák

```
INORDER-FABEJÁRÁS(x)
1   if x ≠ NIL
2         INORDER-FABEJÁRÁS(bal[x])
3         Ki: kulcs[x]
4         INORDER-FABEJÁRÁS(jobb[x])
```

Megjegyzés:

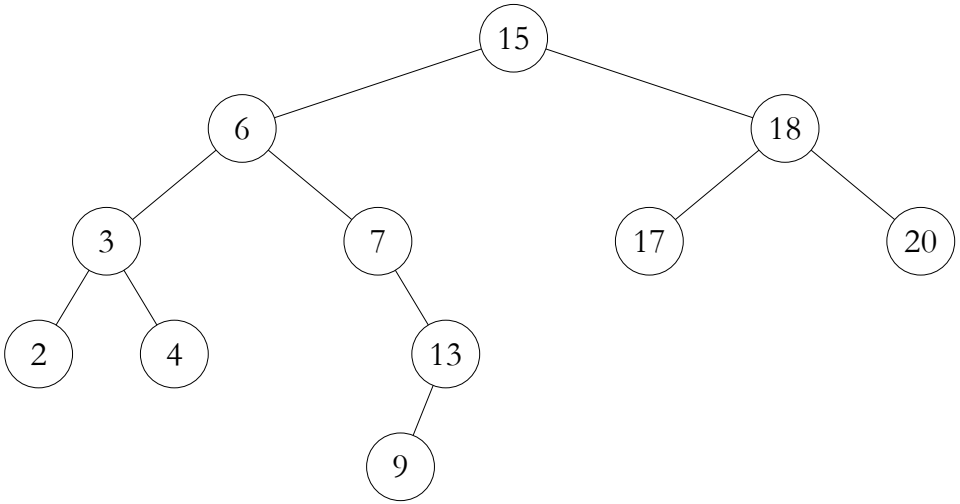
- Egy T bináris keresőfa összes elemének kiírása az INORDER-FABEJÁRÁS(*gyökér*[T]) hívással történhet.
- Az **inorder bejárás** esetén a fa gyökerében lévő kulcsot a baloldali részfájában lévő értékek után és a jobboldali részfájában lévő értékek előtt (azok között) írjuk ki.
- A **preorder bejárás** esetén a gyökér kulcsát a részfáinak kulcsai előtt, míg a **posztorder bejárás** esetében azok után írjuk ki.

Tétel: Ha x egy n csúcsú részfa gyökere, akkor az INORDER-FABEJÁRÁS(x) hívás $\Theta(n)$ ideig tart.

Hatékonyság: Egy n csúcsú bináris keresőfa bejárása $\Theta(n)$ ideig tart.

Feladatok

- Rajzoljunk 2, 3, 4, 5 és 6 magasságú bináris keresőfákat, amelyek az alábbi halmazban lévő kulcsokat tartalmazzák.
  - {1, 4, 5, 10, 16, 17, 21}
- Milyen kulcssorozatot írna ki az alábbi bináris keresőfa esetén egy preorder bejárást követő algoritmus, és milyet egy posztorder bejárást követő?



Bináris keresőfák

```
FÁBAN-KERES(x, k)
1   if x = NIL  vagy  k = kulcs[x]
2           return x
3   if k < kulcs[x]
4           return FÁBAN-KERES(bal[x], k)
5   else
6           return FÁBAN-KERES(jobb[x], k)
```

```
FÁBAN-ITERATÍVAN-KERES(x, k)
1   while x ≠ NIL  és  k ≠ kulcs[x]
2           if k < kulcs[x]
3                   x ← bal[x]
4           else
5                   x ← jobb[x]
6   return x
```

Hatékonyság: Egy h magasságú bináris keresőfában a keresés $O(h)$ ideig tart.

Feladatok

- Tegyük fel, hogy egy bináris keresőfában a kulcsok az 1 és 1000 közötti egész számok. A 363 értéket akarjuk megkeresni. Az alábbi sorozatok közül melyek nem lehetnek keresési sorozatok?
  - 2, 252, 401, 398, 330, 344, 397, 363
  - 924, 220, 911, 244, 898, 258, 362, 363
  - 925, 202, 911, 240, 912, 245, 363

Bináris keresőfák

FÁBAN-MINIMUM(x)
1 **while** $bal[x] \neq$ NIL
2 $x \leftarrow bal[x]$
3 **return** x

FÁBAN-MAXIMUM(x)
1 **while** $jobb[x] \neq$ NIL
2 $x \leftarrow jobb[x]$
3 **return** x

Hatékonyság: Egy h magasságú bináris keresőfában ezek a műveletek is elvégezhetők $O(h)$ idő alatt.

Piros-fekete fák

A **piros-fekete fa** olyan bináris keresőfa, amelynek minden csúcsa egy extra bit információt tartalmaz, ez a csúcs színe, amelynek értékei: PIROS, FEKETE.

A csúcsok színezésének korlátozásával biztosítható, hogy a piros-fekete fában bármely, a gyökértől levélig vezető út hossza nem lehet nagyobb, mint a legrövidebb ilyen út hosszának kétszerese.

Az ilyen fák megközelítőleg **kiegyensúlyozottak**, lehetővé teszik, hogy a dinamikus halmaz műveletek a legrosszabb esetben is $O(\lg n)$ idejűek legyenek.

Egy bináris keresőfa piros-fekete fa, ha rendelkezik a következő **piros-fekete tulajdonságokkal**:

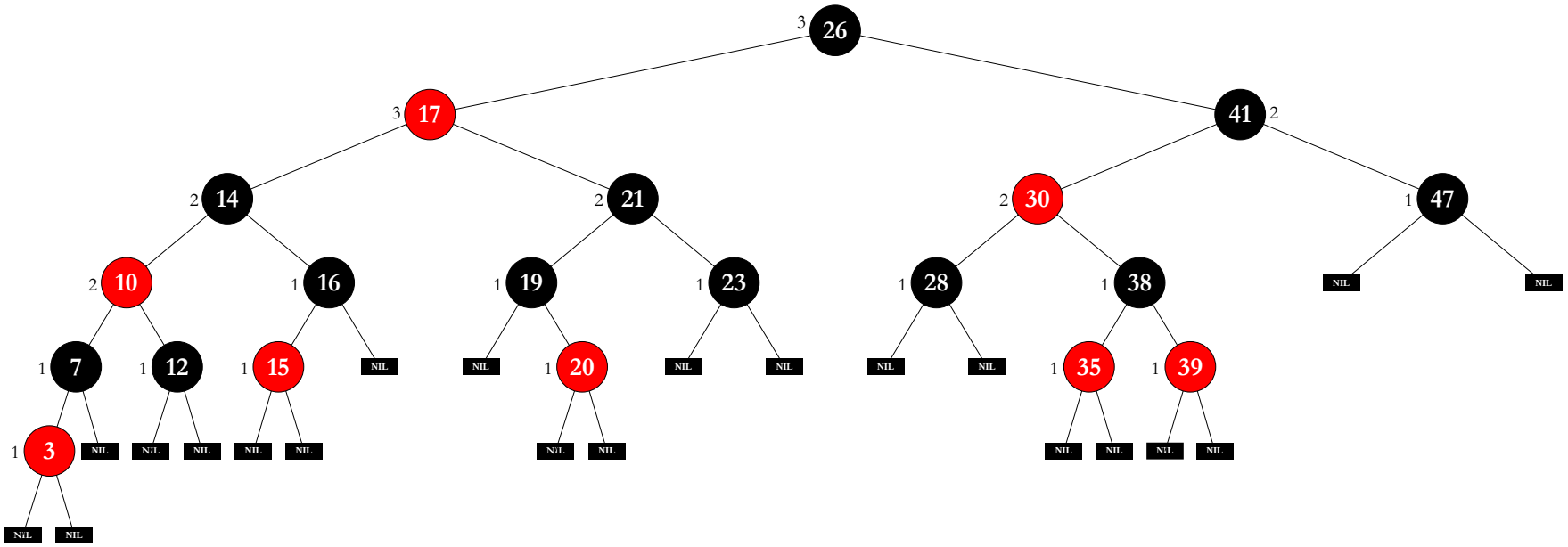
1. Minden csúcs színe vagy piros, vagy fekete.
2. A gyökércsúcs színe fekete.
3. Minden levél (NIL) színe fekete.
4. Minden piros csúcsnak mindkét gyereke fekete.
5. Minden csúcsra igaz, hogy a belőle kiinduló, levélig vezető utakon ugyanannyi fekete csúcs van.

Egy x csúcs **fekete-magasságának** nevezzük az x csúcsból induló, levélig vezető úton található, x-et nem tartalmazó fekete csúcsok számát.

Egy piros-fekete fa fekete-magassága a fa gyökércsúcsának fekete-magassága.

Tétel: Bármely n belső csúcsot tartalmazó piros-fekete fa magassága legfeljebb $2\lg(n+1)$.

Piros-fekete fák

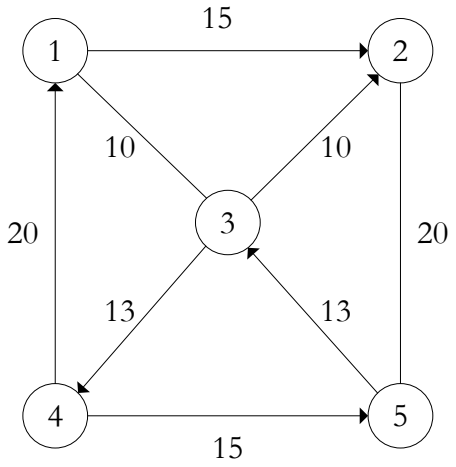


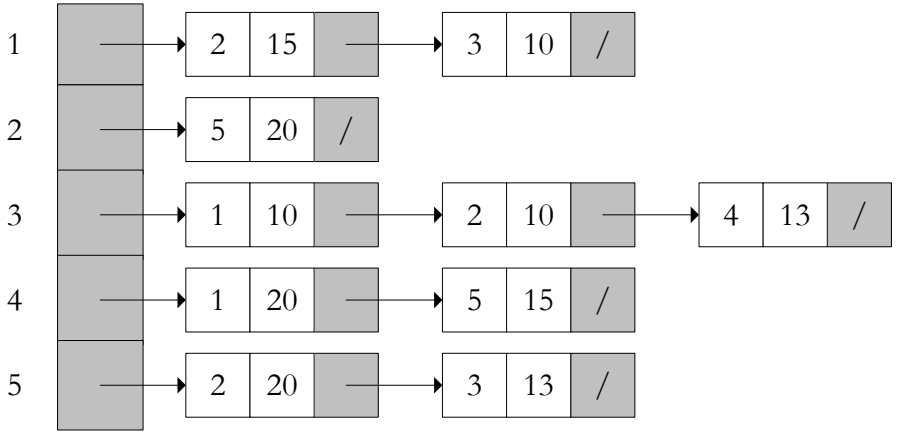
Egy piros-fekete fa a fekete magasságokkal

Feladatok

- Rajzoljuk meg az {1, …, 15} kulcsokat tartalmazó 3 magasságú teljes bináris keresőfát! Egészítsük ki NIL levelekkel és színezzük be a csúcsokat úgy, hogy a fa fekete-magassága 2, 3, ill. 4 legyen!
- Hogyan adjunk meg egy *n* csúcsú piros-fekete fát, hogy a lehető legnagyobb legyen a piros és a fekete belső csúcsok aránya? Milyen fa esetén lesz az arány a legkisebb, és mi lesz ez az érték?

Gráfok ábrázolása

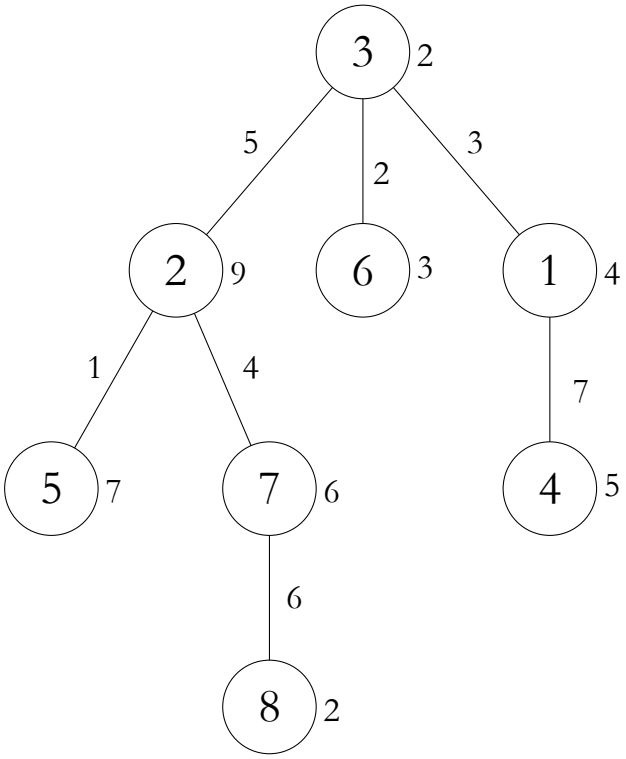




	1	2	3	4	5
1	0	15	10	0	0
2	0	0	0	0	20
3	10	10	0	13	0
4	20	0	0	0	15
5	0	20	13	0	0

Irányított gráf ábrázolása szomszédsági listákkal és csúcsmátrixszal

Gyökeres fák



Pont	1	2	3	4	5	6	7	8
Címke	3	3	-	1	2	3	2	7
Pontjellemző	4	9	2	5	7	3	6	2
Éljellemző	3	5	-	7	1	2	4	6

Gyökeres fa ábrázolása címketömbbel