

Algoritmuselmélet

Keresőfák, piros-fekete fák

Katona Gyula Y.

Számítástudományi és Információelméleti Tanszék
Budapesti Műszaki és Gazdaságtudományi Egyetem

Keresőfák

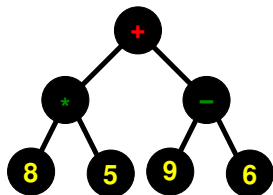
Tároljuk az U rendezett halmaz elemeit, hogy **BESZÚR**, **TÖRÖL**, **KERES**, **MIN**, (**MAX**, **TÓLIG**) hatékonyak legyenek.

Bináris fa bejárása

teljes fa: az alsó szint is tele van $\implies l$ szintű, teljes fának $2^l - 1$ csúcsa van.

Fa csúcsai \rightarrow $elem(x)$, $bal(x)$, $jobb(x)$, $apa(x)$ és esetleg $reszfa(x)$

Ha x a gyökér, y pedig a 9-es csúcs, akkor



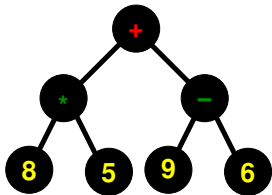
$$\begin{aligned} bal(jobb(x)) &= y, \\ apa(apa(y)) &= x, \\ elem(bal(x)) &= *, \\ reszfa(x) &= 7. \end{aligned}$$

PREORDER, INORDER, POSTORDER

```
pre(x)
begin
  látogat(x);
  pre(bal(x));
  pre(jobb(x))
end
```

```
in(x)
begin
  in(bal(x));
  látogat(x);
  in(jobb(x))
end
```

```
post(x)
begin
  post(bal(x));
  post(jobb(x));
  látogat(x)
end
```



PREORDER: + * 8 5 - 9 6

INORDER: 8 * 5 + 9 - 6

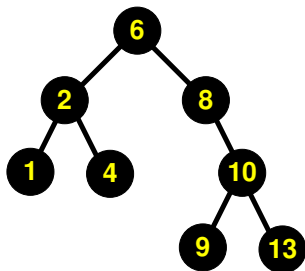
POSTORDER: 8 5 * 9 6 - +

Lépésszám: $O(n)$

Bináris keresőfa

Definíció (Keresőfa-tulajdonság)

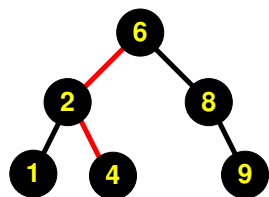
Tetszőleges x csúcsra és az x baloldali részfájában levő y csúcsra igaz, hogy $\text{elem}(y) \leq \text{elem}(x)$. Hasonlóan, ha z egy csúcs az x jobb részfájából, akkor $\text{elem}(x) \leq \text{elem}(z)$.



Házi feladat: Igazoljuk, hogy egy bináris keresőfa elemeit a fa inorder bejárása *nemcsökkenő sorrendben* látogatja meg.

Egy kényelmes megállapodás: a továbbiakban feltesszük, hogy nincsenek ismétlődő elemek a keresőfában.

Naiv algoritmusok



KERES(4, S)

KERES(s,S): Összehasonlítjuk s -et S gyökerében tárolt s' elemmel.

- Ha $s = s'$, akkor megtaláltuk.
- Ha $s < s'$, akkor balra megyünk tovább.
- Ha $s > s'$, akkor jobbra megyünk.

Ugyanezt az utat járjuk be a KERES(5, S) kapcsán, de azt nem találjuk meg.

Lépésszám: $O(\ell)$, ahol ℓ a fa mélysége

MIN: mindig balra lépünk, amíg lehet

MAX: mindig jobbra lépünk, amíg lehet

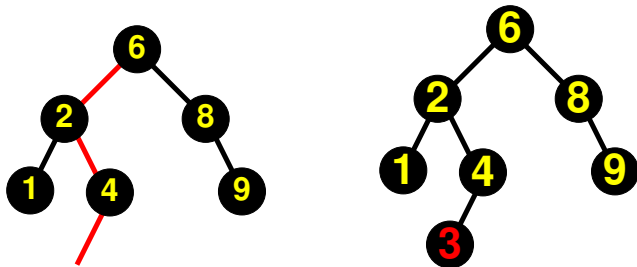
Lépésszám: $O(\ell)$

TÓLIG(a, b, S): KERES(a, S) \rightarrow INORDER a -tól b -ig

Lépésszám: $O(\ell + k)$, ahol k az a és b között levő elemek száma

Naiv BESZÚR

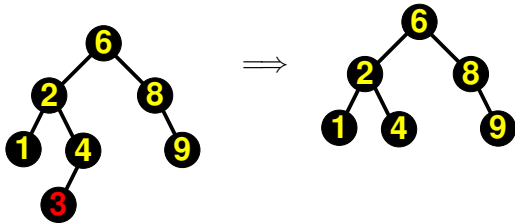
BESZÚR(s, S): KERES(s, S)-sel megkeressük, hova kerülne, és új levelet adunk hozzá, pl. **BESZÚR**(3, S):



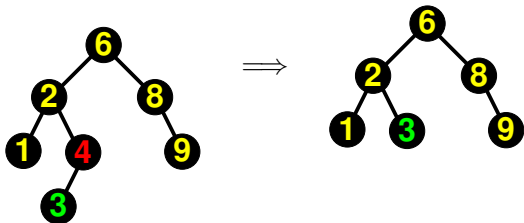
Lépésszám: $O(\ell)$

Naiv TÖRÖL

- $TÖRÖL(s, S)$: Ha s levél, akkor triviális, pl. $TÖRÖL(3, S)$:

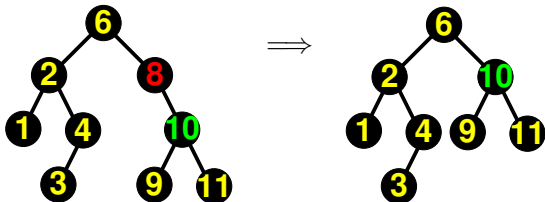


- $TÖRÖL(s, S)$: Ha s -nek egy fia van, akkor: $s \leftarrow \text{fiú}(s)$, pl. $TÖRÖL(4, S)$:

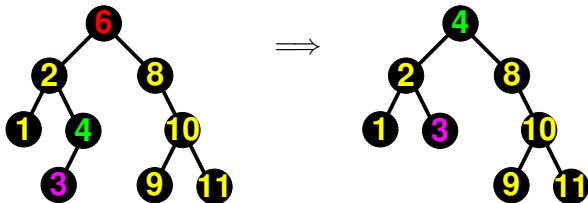


Naiv TÖRÖL

- Vagy pl. TÖRÖL(8, S')



- TÖRÖL(s, S): Ha s-nek két fia van, akkor visszavezetjük az előző esetre. s helyére tegyük $y := \text{MAX}(\text{bal}(s))$ -t és töröljük y-t. Pl. TÖRÖL(6, S')



Naiv TÖRÖL

Állítás

$y := \text{MAX}(\text{bal}(s))$ csúcsnak nem lehet két fia.

Bizonyítás.

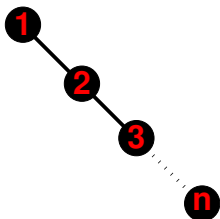
Ha lenne két fia, akkor lenne egy y' jobb fia is. De ekkor $y' > y$.



Lépésszám: $O(\ell)$

Faépítés naiv beszúrásokkal

Ha pl. az $1, 2, \dots, n$ sorozatból építünk fát így, akkor ezt kapjuk:



Az építés költsége: $2 + 3 + \dots + (n - 1) = O(n^2)$

Tétel

Ha egy véletlen sorozatból építünk fát naiv beszúrásokkal, akkor az építés költsége átlagosan $O(n \log_2 n)$. A kapott fa mélysége átlagosan $O(\log_2 n)$.

Animáció: Bejárások, bináris keresőfa

Piros-fekete fák

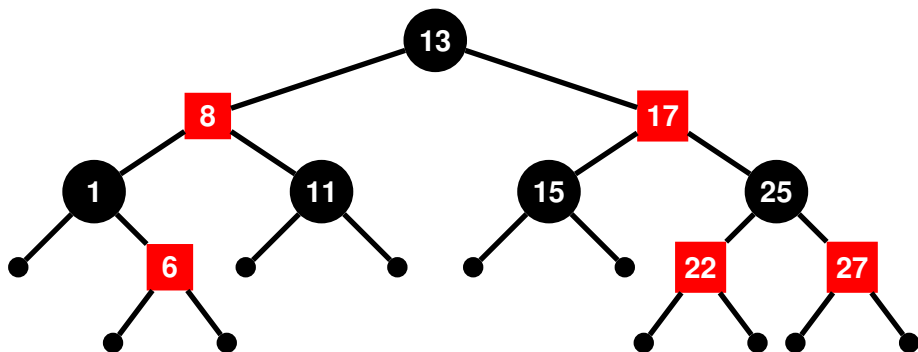
Olyan bináris keresőfa, melynek mélysége nem lehet nagy.
BESZÚR, TÖRÖL, KERES, MIN, (MAX, TÓLIG) hatékonyak.

Definíció

A **piros-fekete fa** egy bináris keresőfa, melyre teljesülnek a következők:

- 1 Minden nem levél csúcsnak 2 fia van.
- 2 Elemeket belső csúcsokban tárolunk.
- 3 Teljesül a keresőfa tulajdonság.
- 4 A fa minden csúcsa **piros** vagy fekete.
- 5 A gyökér fekete.
- 6 A levelek feketék.
- 7 Minden **piros** csúcs mindkét gyereke fekete.
- 8 Minden v csúcsra igaz, hogy az összes v -ből levélbe vezető úton ugyanannyi fekete csúcs van.


Példa



Megj.: A szokásos bináris fát kiegészítjük üres levelekkel.

Piros-fekete fák

Jelölések

- F_v : v gyökerű részfa
- $m(v)$: v magassága, a leghosszabb v -ből levélbe vezető út éleinek száma
- $fm(v)$: v fekete-magassága, a v -ből levélbe vezető összes úton a fekete csúcsok száma, v -t nem számolva.
(Ez minden úton egyforma a  tulajdonság miatt.)

Állítás

Egy *piros*-fekete fa minden v csúcsára teljesül

$$\frac{m(v)}{2} \leq fm(v) \leq m(v).$$

Bizonyítás.

A leghosszab levélbe vezető úton a feketék száma nem lehet több az élek számánál $\implies fm(v) \leq m(v)$. ✓

7. pont miatt a leghosszabb úton a pontoknak legalább a fele fekete $\implies \frac{m(v)}{2} \leq fm(v)$. ✓ □

Állítás

F_v belső csúcsainak száma $b_v \geq 2^{fm(v)} - 1$.

Bizonyítás.

Indukcióval $m(v)$ -re: $m(v) = 0 \implies fm(v) = 0, b_v \geq 2^0 - 1 \quad \checkmark$

Ha $m(v) > 0$, akkor legyen x, y a két fia.

$\implies m(x) < m(v)$ és $m(y) < m(v)$

$fm(v) - 1 \leq fm(x) \leq fm(v)$ és $fm(v) - 1 \leq fm(y) \leq fm(v)$

$b_v = b_x + b_y + 1 \implies$

$b_v \geq (2^{fm(x)} - 1) + (2^{fm(y)} - 1) + 1 \geq 2 \cdot (2^{fm(v)-1} - 1) + 1 = 2^{fm(v)} - 1. \quad \square$

Tulajdonságok

Állítás

Ha egy *piros*-fekete fában n elemet tárolunk, akkor a fa magassága $\leq 2 \log(n + 1)$.

Bizonyítás.

Ha r a gyökér $\implies b_r = n$.

$$n = b_r \geq 2^{fm(r)} - 1 \implies \log(n + 1) \geq fm(r) \geq \frac{m(r)}{2} \quad \checkmark \quad \square$$

Tétel

KERES, *MAX*, *MIN* lépésszáma *piros*-fekete fában $O(\log n)$.

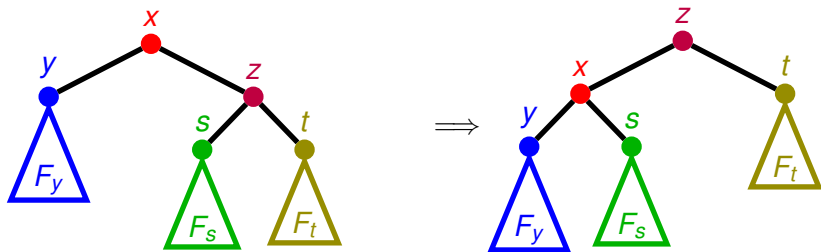
Bizonyítás.

Általában minden keresőfában a lépésszám a fa magasságával arányos $\implies O(l) = O(\log n)$. □

BESZÚR lépésszáma

Ha a keresőfáknál használatos beszúrást használnánk, akkor megsérülhetne a **piros-fekete** tulajdonság.

Forgatás



Megj.: Ez a művelet megtartja a keresőfa tulajdonságot.

BESZÚR

Szúrjuk be az új elemet a keresőfáknál megismert módon. \implies
Új belső csúcs keletkezik (gyerekei csak üres fekete levelek): z

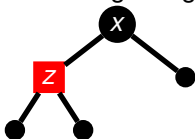
- Ha z a gyökér, akkor legyen fekete \implies



- Ha z nem gyökér, akkor legyen az apja x , \implies
 z legyen piros.

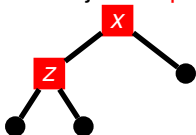
(1) Ha x fekete \implies fekete-magasságok sehol nem

változnak $\checkmark \implies$



(2) Ha x piros \implies nem teljesül a piros-fekete

tulajdonság \implies

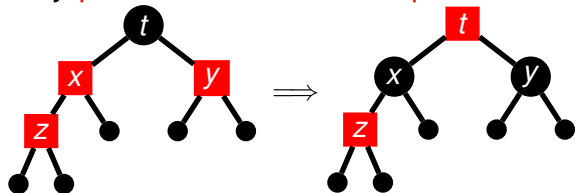


\implies további lépések kellenek.

BESZÚR

(2) Mivel x piros, nem gyökér \implies
legyen x apja t (fekete), x testvére y .

(2.1) Ha y piros \implies átszínezzük t -t pirosra



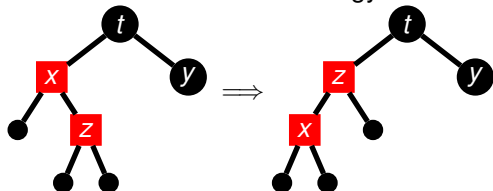
Ezzel a problémát két szinttel feljebb toltuk, ott folytatjuk a fa rendbetételét.

Kivéve, ha t a gyökér $\implies t$ marad fekete $\implies fm(t)$ eggyel nagyobb lesz.

BESZÚR

(2.2) Ha y fekete:

(2.2.1) Ha z és x nem azonos oldali gyerek \implies forgatunk x körül.



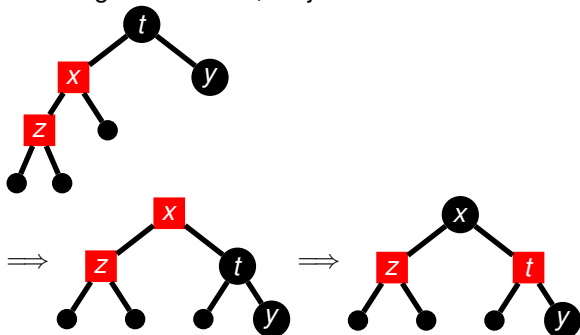
Ezzel a következő esetre vezettük a problémát.

BESZÚR

(2.2) Ha y fekete:

(2.2.2) Ha z és x azonos oldali gyerek

\implies forgatunk t körül, majd átszínezzük.



Ezzel a gyökér fekete-magassága nem változik, és teljesül a piros-fekete tulajdonság. ✓

BESZÚR

Tétel

A *BESZÚR* során

- (a) a lépésszám $O(\log n)$,
- (b) legfeljebb 2 forgatás történik.

Bizonyítás.

- (a) *y* piros esetben a (2.1) pontban 2 szinttel feljebb kerül a baj \implies szintenként konstans lépés $\implies O(\log n)$. ✓
- (b) Forgatás csak a (2.2) esetben történik, de ekkor nincs felgyűrűzés, rögtön kijavítjuk a fát. ✓



TÖRÖL

Hasonló módszerek, de bonyolultabb.

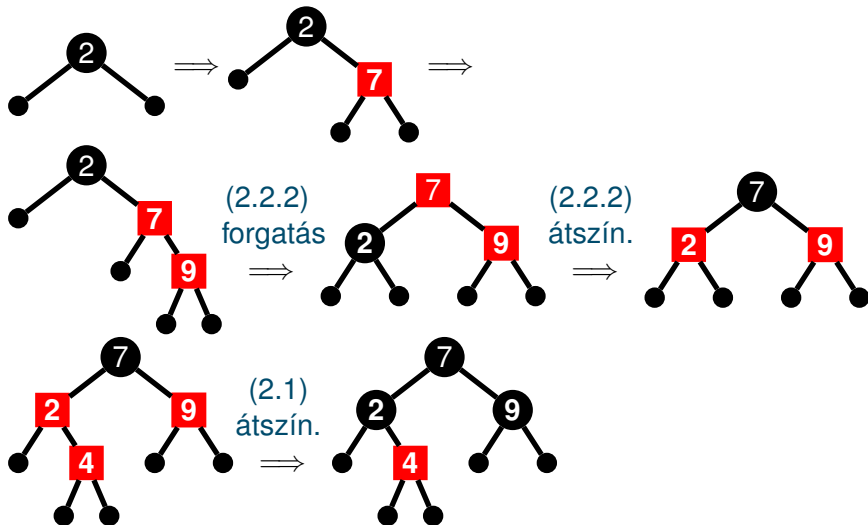
Tétel

A TÖRÖL során

- (a) a lépésszám $O(\log n)$,*
- (b) legfeljebb 3 forgatás történik.*

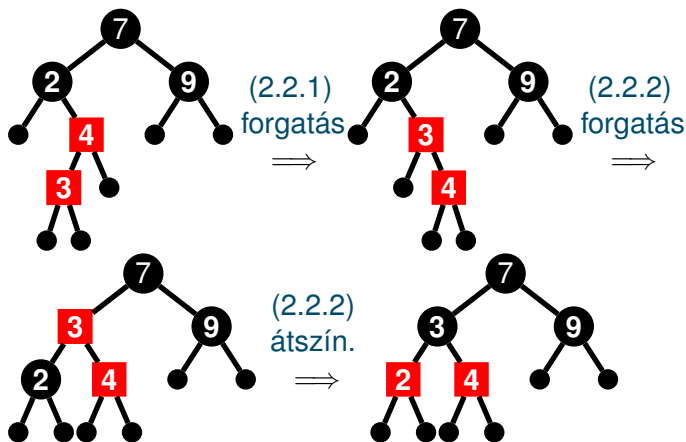
Példa BESZÚRÁSokra

Szűrjük be egy üres fába sorban a 2, 7, 9, 4, 3, 1 elemeket.



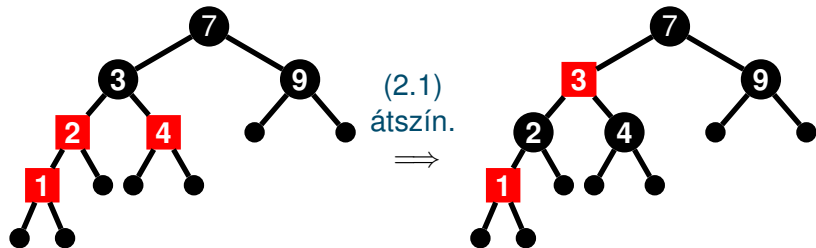
Példa BESZÚRÁSokra

Szűrjük be egy üres fába sorban a 2, 7, 9, 4, 3, 1 elemeket.



Példa BESZÚRÁSokra

Szűrjük be egy üres fába sorban a 2, 7, 9, 4, 3, 1 elemeket.



Java animáció: Piros-fekete fa