ADATSZERKEZETEK ÉS ALGORITMUSOK

Piros-Fekete Fa Törlés

„Hierarchikus adatszerkezetek, keresési fák”

Piros-Fekete fák – törlés

Egy z csúcs törlése egy bináris keresőfából:

1. eset: z-nek nincs nem-nil gyereke
   - töröljük a csúcsot (a szülőt a nil-elemmel kötjük össze)
2. eset: z-nek egy nem-nil gyereke van
   - töröljük a csúcsot (a szülőt a gyerek-elemmel kötjük össze)
3. eset: z-nek két nem-nil gyereke van
   - megkeressük a közvetlen rákövetkezőjét
   - átmásoljuk belőle az adatot z-be (z színe változatlan marad)
   - töröljük a rákövetkezőt (ennek csak 0 v. 1 gyereke lehet)

- Jelöljük y-nal a ténylegesen kitörölt elemet.

Piros-Fekete fák – törlés

Ha a kitörölt elem (y) piros volt

Tulajdonság és leírása	Igaz?
1. Minden csúcs piros vagy fekete	igen
2. A gyökér fekete	igen
3. Minden levél (nil[T]) fekete	igen
4. Ha egy csúcs piros, mindkét gyerek fekete	igen
5. Minden út egy csúcstól a leszármazott levelekig ugyanannyi fekete csúcsot tartalmaz	igen

Ebben az esetben nincs tennivaló.

Piros-Fekete fák – törlés

Ha a kitörölt elem (y) fekete volt

Tulajdonság és leírása	Igaz?
1. Minden csúcs piros vagy fekete	igen
2. A gyökér fekete	???
3. Minden levél (nil[T]) fekete	igen
4. Ha egy csúcs piros, mindkét gyerek fekete	???
5. Minden út egy csúcstól a leszármazott levelekig ugyanannyi fekete csúcsot tartalmaz	???

Az „elvesztett” fekete csúcsot kell „pótolni”.

Piros-Fekete fák – törlés

- Ahhoz, hogy helyreállítsuk a fekete csúcsok számát minden úton, a gyerekének adjuk a kitörölt csúcs fekete színét
- Úgy képzeljük, hogy a gyerekének most van egy extra feketéje
  - Ha a gyerek színe fekete, akkor most dupla-fekete,
  - Ha a gyerek színe piros, akkor most piros-fekete
    - Ténylegesen nem változtatjuk a csúcsot a kódban, csak úgy vesszük, mintha…
- Ezzel a legnehezebb, 5. tulajdonságot teljesítjük
  - Cserébe az 1., 2. tulajdonságot nem
  - A gyerek se nem piros, se nem fekete.

Piros-Fekete fák – törlés

- Az 1. tulajdonság helyreállítására ezt az „extra” feketét visszük felfelé a fában, míg a következők egyike igaz nem lesz:
  - x egy piros-fekete csúcsra mutat
    - Ekkor feketére színezzük, és kész
  - x a fa gyökerére mutat
    - Ha dupla-fekete, eltávolítjuk az egyik feketéjét, és kész, mert ha a gyökérből viszünk el egy extra feketét, akkor egyforma marad a levelekhez vezető utakon a feketék száma
  - Olyan ponthoz érünk, ahol forgatásokkal és átszínezésekkel el tudjuk távolítani az extra feketét úgy, hogy nem sértjük meg a piros-fekete tulajdonságokat többé

Piros-Fekete fák – törlés

```
PF-FÁBÓL-TÖRÖL(T,z) -- törlés, mint bináris keresőfából, majd p-f helyreállítás
  if bal[z]=nil[T] or jobb[z]=nil[T]   -- jelölő strázsa
    then y←z                           -- 0 vagy 1 gyerek
    else y←FÁBAN-KÖVETKEZŐ(T,z)        -- 2 gyerek
  if bal[y]≠nil[T]
    then x←bal[y]                      -- x az y 0 vagy 1 gyerekére mutat
    else x←jobb[y]
  szülő[x] ←szülő[y]                   -- ha volt (egy) gyereke, befűzzük
  if szülő[y]= nil[T]
    then gyökér[T]←x
    else if y = bal[szülő[y]]
      then bal[szülő[y]]←x
      else jobb[szülő[y]]←x
  if y≠z
  then kulcs[z]←kulcs[y]               -- ha két gyerek volt (ha van egyéb mező, azt is)
```

Piros-Fekete fák – törlés

```
PF-FÁBÓL-TÖRÖL(T,z)  -- folytatás

if szín[y]=FEKETE    -- ha y PIROS, akkor PF tulajdonság OK
                        (fm nem változott sehol,piros-piros szülő-gyerek
                        kapcsolat nem keletkezett.)
then PF-FÁBÓL-TÖRÖL-JAVÍT(T,x)
                     -- az x ténylegesen a törlendőnek a gyereke!

return y
```

Piros-Fekete fák – törlés

- Mi sérül?
  - Ha az y fekete volt, akkor most minden út, ami az y-n keresztül haladt, eggyel kevesebb fekete pontot fog tartalmazni
    - y őseire nem teljesül a 5. tulajdonság
- Amikor y-t eltávolítjuk, fekete értékét „továbbadjuk” a gyerekének
  - Ha x is fekete volt, akkor most „kétszeresen fekete” lesz
    - Ez sérti az 1. tulajdonságot

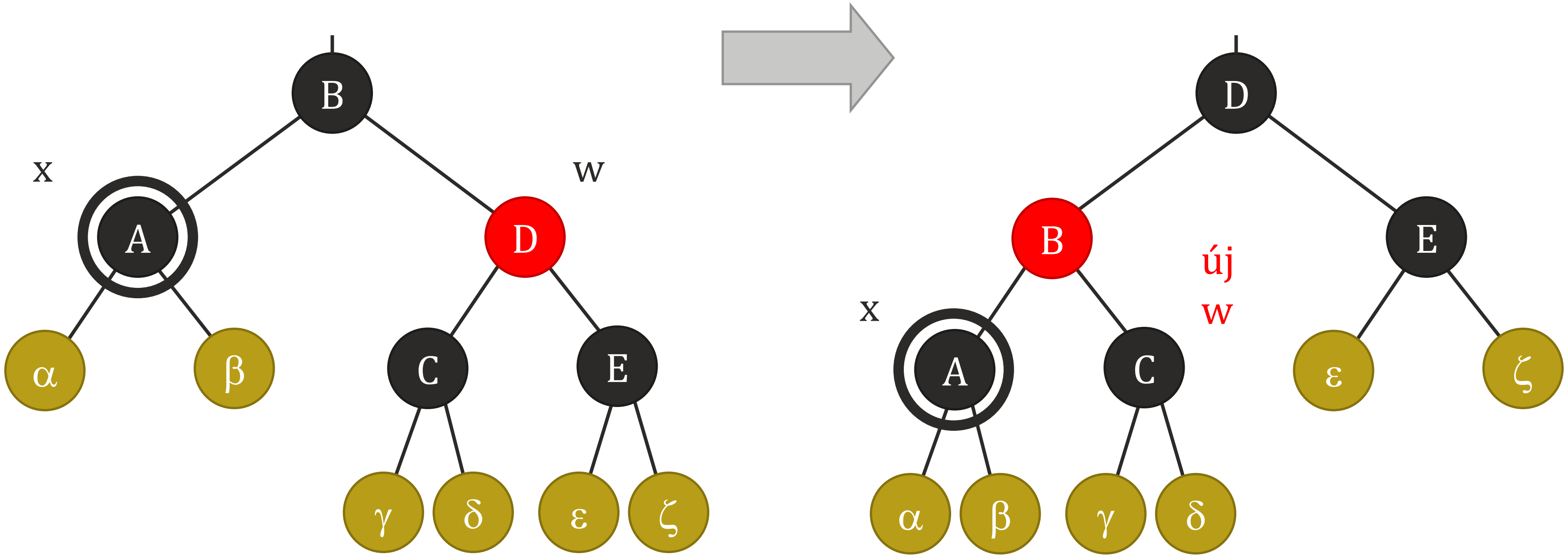
Piros-Fekete fák – törlés

- Lehetséges esetek:
    - Csak azokat vesszük figyelembe, ahol x balgyerek, ahol jobb, az szimmetrikusan adódik.
    - w jelöli x testvérét (w <- jobb[szülő[x]])

1. eset: x testvére w piros
2. eset: x testvére w fekete és w mindkét gyereke fekete
3. eset: x testvére w fekete, w balgyereke piros, jobbgyereke fekete
4. eset: x testvére w fekete, w jobbgyereke piros

1. eset

- x duplán fekete, és a testvére (w) piros
  - w-nek van fekete fia
  - így w és szülő[x] színét felcserélve és balra forgatva a szülő[x]-t, az x új testvére fekete lesz
  - 2.,3., vagy 4. eset



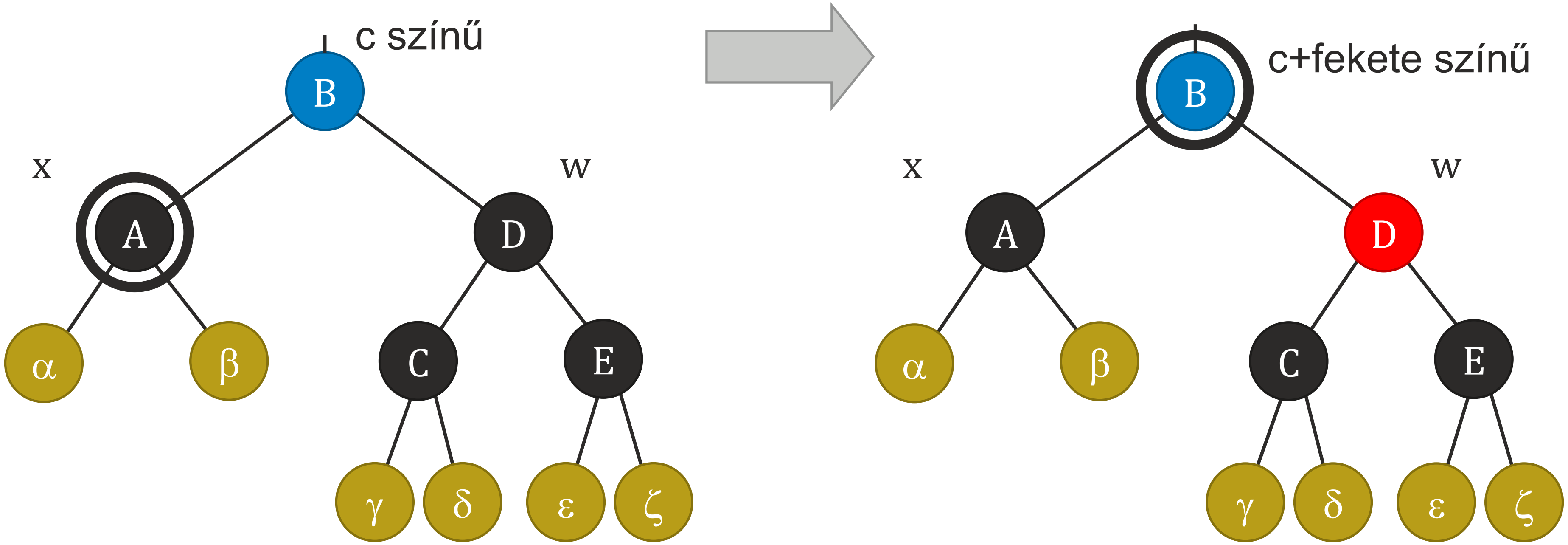
Piros-Fekete fák - törlés

- 1. eset: x testvére w piros
  - Pszeudokód:

```
if szín[w]=PIROS
then szín[w]←FEKETE
   szín[szülő[x]]←PIROS
   BALRA-FORGAT (T, szülő[x])
   w←jobb[szülő[x]]
```

2. eset

- A w is fekete, így nézzük az ő gyerekeinek a színét
  - ha mindkét fia fekete: elvehetünk egy feketét x-től és w-től így x egyszer fekete, w piros lesz, és szülő[x] kap extra feketét,
  - Ezután folytatjuk a helyreállítási ciklust a szülő[x]-re



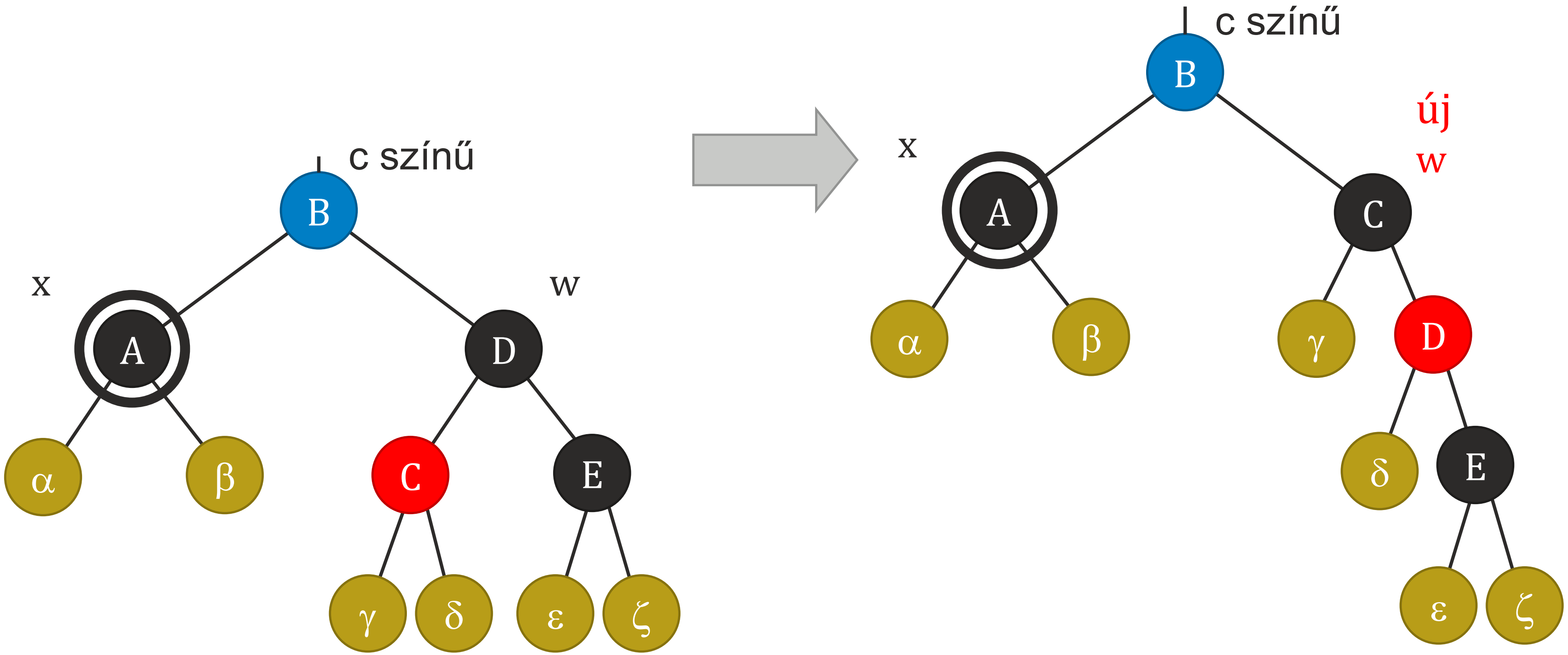
Piros-Fekete fák - törlés

- 2. eset: x testvére w fekete és w mindkét gyereke fekete
  - Pszeudokód:

```
if szín[bal[w]]=FEKETE and
  szín[jobb[w]]=FEKETE
  then szín[w]←PIROS
     x←szülő[x]
```

3. eset

- w fekete, bal fia piros, jobb fia fekete
  - w és bal[w] színét cseréljük, majd jobbforgatás w-re, az új w fekete, és jobb fia piros
  - A 4. esettel folytatjuk



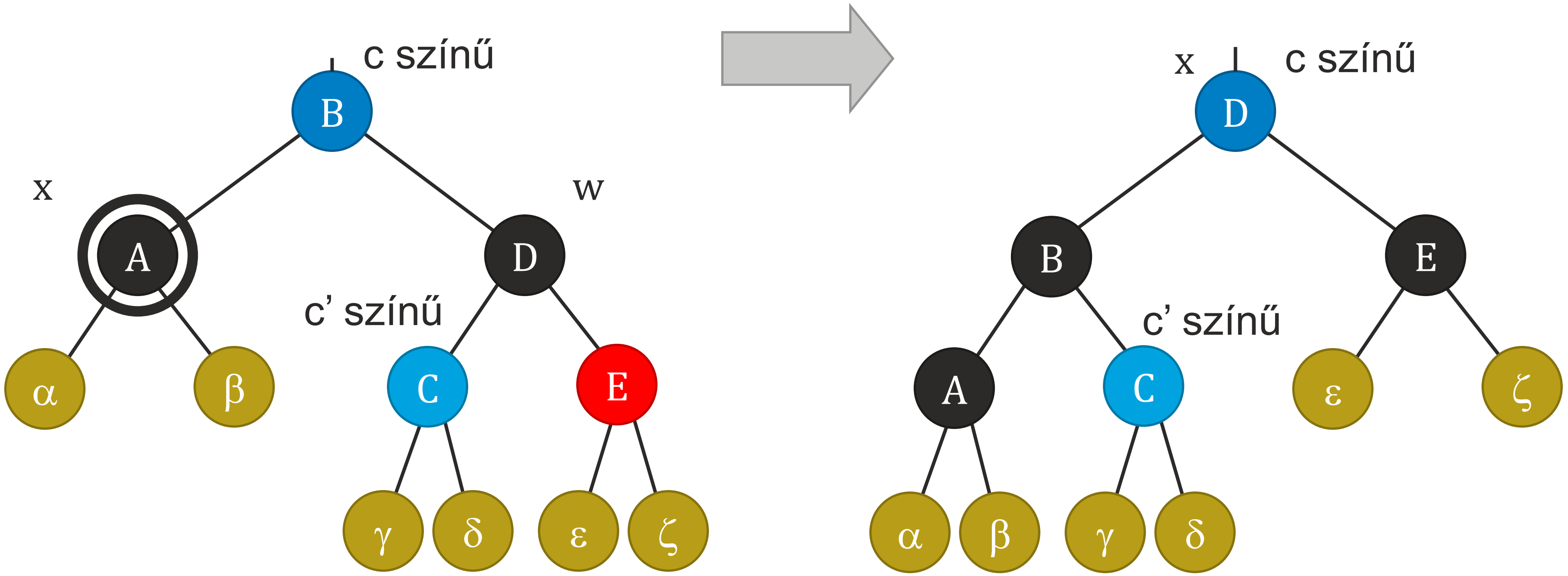
Piros-Fekete fák - törlés

- 3. eset: x testvére w fekete, w balgyereke piros, jobbgyereke fekete
  - Pszeudokód:

```
else if szín[jobb[w]]=FEKETE
     then szín[bal[w]]←FEKETE;
          szín[w]←PIROS,
          JOBBRA-FORGAT(T,w);
          w←jobb[szülő[x]]
```

4. eset

- w fekete és jobb fia piros
  - w, szülő[x] és jobb[w] színét váltjuk, majd szülő[x] körül balraforgatva úgy törölhetjük le az x extra feketéjét, hogy a PF tulajdonság marad
  - Ezután kész (a fa gyökerét feketére színezzük)



Piros-Fekete fák - törlés

- 4. eset: x testvére w fekete, w jobbgyereke piros
  - Pszeudokód:

```
szín[w]←szín[szülő[x]]
szín[szülő[x]]←FEKETE;
szín[jobb[w]]←FEKETE
BALRA-FORGAT(T, szülő[x])
x ← gyökér[T]
```

Piros-Fekete fák – algoritmusok

```
PF-FÁBÓL-TÖRÖL-JAVÍT(T,x)
while x ≠ gyökér[T] and szín[x]= FEKETE
   do if x = bal[szülő[x]]
      then w←jobb[szülő[x]]
           if szín[w]=PIROS
           then szín[w]←FEKETE; szín[szülő[x]]←PIROS
                 BALRA-FORGAT(T,szülő[x]); w←jobb[szülő[x]]
           if szín[bal[w]] = FEKETE and szín[jobb[w]] = FEKETE
              then szín[w]← PIROS; x←szülő[x]
              else if szín[jobb[w]]= FEKETE
                   then szín[bal[w]]←FEKETE; szín[w]←PIROS
                        JOBBRA-FORGAT(T,w); w←jobb[szülő[x]]
                   szín[w]← szín[szülő[x]]
                   szín[szülő[x]]←FEKETE; szín[jobb[w]]←FEKETE
                   BALRA-FORGAT(T, szülő[x])
                   x←gyökér[T]
       else ua., mint a then, csak a bal és jobb felcserélve
szín[gyökér[T]]←FEKETE
```

Összegzés

Piros-Fekete fák – Elemzés

- Hozzáadás
  - Beszúrás
    - Összehasonlítások $\mathcal{O}(\log_2 n)$
    - PF-tulajdonsághoz $\mathcal{O}(\log_2 n)$
      - Minden lépésnél x felfelé mozog a fában legalább egy szintet
  - Összesen $\mathcal{O}(\log_2 n)$
- Törlés
  - Összesen szintén $\mathcal{O}(\log_2 n)$
- Bonyolultabb, de $\mathcal{O}(\log_2 n)$ viselkedést ad dinamikus esetekben is!

Dinamikus fák: PF vagy AVL?

- Beszúrás
  - AVL: két menet a fán keresztül
    - lefelé a csúcs beszúrásához
    - felfelé az újrakiegyensúlyozáshoz
  - Piros-Fekete: két menet a fán keresztül
    - lefelé a csúcs beszúrásához
    - felfelé az újrakiegyensúlyozáshoz
- A Piros-Fekete népszerűbb?

Dinamikus fák: PF vagy AVL?

- Beszúrás
  - Ha a Cormen et al. könyvet olvassuk,
    - nem indokolja, hogy miért részesíti előnyben a Piros-Fekete fákat
  - Weiss könyvében szerepel, hogy egy Piros-Fekete fát *ki lehet egyensúlyozni egy menetben*
    - M A Weiss, Algorithms, Data Structures and Problem Solving with C++, Addison-Wesley, 1996
    - Így a Piros-Fekete fák hatékonyabbak lesznek, mint az AVL fák!
- Fontos olvasni a szakirodalmat

Dinamikus fák

- Beszúrás egy menetben
  - Ahogy megyünk lefelé a fán, ha találunk egy csúcsot
    két piros gyerekkel, változtassuk a színét pirosra és a gyerekekét feketére
  - Ez nem változtatja a fekete csúcsok számát egyetlen úton sem
  - Ha ennek a csúcsnak a szülője piros volt, akkor forgatás kell ...
    - A forgatás lehet sima, vagy dupla