

# Algoritmusok és adatszerkezetek II.

## Augmentált keresőfák

Szegedi Tudományegyetem



# Bináris keresőfák műveletei

- $h$  magas és  $n$  csúcsból álló bináris keresőfa esetén a Keres, Beszúr, Töröl műveletek  $O(h)$  időben végrehajthatók
- $x$  csúcs rangja: a fában tárolt kulcsok között  $x$  hanyadik helyen áll  $<$  szerint
- Hogy keresnénk meg egy bináris keresőfa  $r$  rangú elemét?
- Hogyan határoznánk meg egy bináris keresőfa  $x$  csúcsának rangját?



# Bináris keresőfák műveletei

- $h$  magas és  $n$  csúcsból álló bináris keresőfa esetén a Keres, Beszúr, Töröl műveletek  $O(h)$  időben végrehajthatók
- $x$  csúcs rangja: a fában tárolt kulcsok között  $x$  hanyadik helyen áll  $<$  szerint
- Hogy keresnénk meg egy bináris keresőfa  $r$  rangú elemét?
- Hogyan határoznánk meg egy bináris keresőfa  $x$  csúcsának rangját?

## Általános ötlet

A fapontokban tárolt extra adattagok hozzásegíthetnek minket új műveletek hatékony ( $O(h)$  idejű) végrehajtásához



# Új műveletek – adott rangú kulcs meghatározása

- Cél:  $r \leq n$  ranggal rendelkező elem megtalálása a fában
- Naiv (de működő) elgondolás: elkezdem bejárni a fát  $<$  szerinti sorrendben, és megállok az  $r$ -ediknek érintett csúcsnál



# Új műveletek – adott rangú kulcs meghatározása

- Cél:  $r \leq n$  ranggal rendelkező elem megtalálása a fában
- Naiv (de működő) elgondolás: elkezdem bejárni a fát  $<$  szerinti sorrendben, és megállok az  $r$ -ediknek érintett csúcsnál
- $O(h)$  helyett  $\Theta(r)$  idejű algoritmus
  - Kiegyensúlyozott fa és kellően nagy  $r$  estében pedig  $r \gg h$



# Új műveletek – adott kulcs rangjának meghatározása

- Naiv elgondolás: elkezdem bejárni a fát  $<$  szerinti sorrendben, és megállok, ha  $x$  kulcsot érintem
- A válasz az  $x$  megtalálásáig érintett kulcsok száma
- $O(h)$  helyett  $O(n)$  idejű algoritmus



# Új műveletek – adott kulcs rangjának meghatározása

- Naiv elgondolás: elkezdem bejárni a fát  $<$  szerinti sorrendben, és megállok, ha  $x$  kulcsot érintem
- A válasz az  $x$  megtalálásáig érintett kulcsok száma
- $O(h)$  helyett  $O(n)$  idejű algoritmus

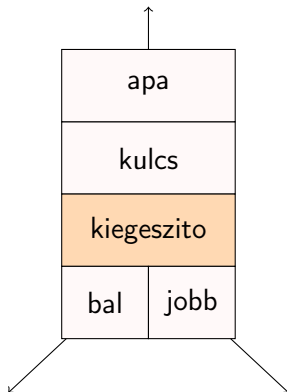
## A megoldás

Minden csúcs tároljon el kiegészítő információt magáról!



# Rendezett-minta fa implementációja

```
class Node {  
    Object kulcs;  
    int kiegészito;  
    Node *apa;  
    Node *bal;  
    Node *jobb;  
}
```



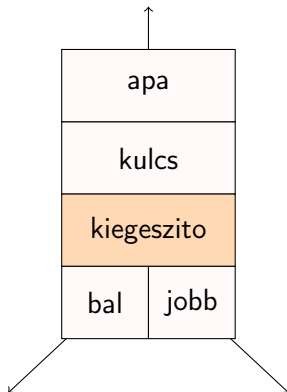


# Rendezett-minta fa implementációja

```
class Node {  
    Object kulcs;  
    int kiegészito;  
    Node *apa;  
    Node *bal;  
    Node *jobb;  
}
```

## Megjegyzés

A kiegészítő információ legyen az adott gyökerű részfa mérete (benne található kulcsok száma). Üres fa kiegészítő információja 0.



## Adott rangú kulcs keresése

```
RANGKERES(x, i) {  
    r = x.bal.kiegeszito + 1  
  
    if (i < r) {  
        RANGKERES(x.bal, i)  
    } elseif (i > r) {  
        RANGKERES(x.jobb, i - r)  
    } else {  
        return x  
    }  
}
```

$h$  magas fa esetén  $O(h)$



## Adott kulcs rangjának meghatározása

```
RANGMEGHATÁROZ(x) {  
    r = 0  
    y = x  
    while (y != nil) {  
        if (x.kulcs >= y.kulcs) {  
            r = r + y.bal.kiegeszito + 1  
        }  
        y = y.apa  
    }  
    return r  
}
```

## Szövegesen

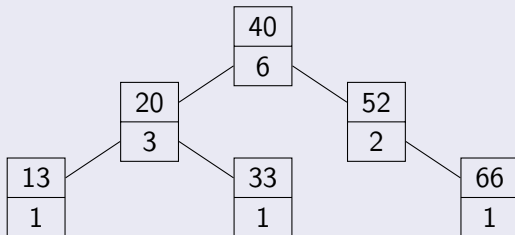
$x$ -től a gyökérig lépkedve azon  $y$  gyökerű részfák gyökérelemeinek rangjait összegezzük, melyekre  $x.kulcs \geq y.kulcs$



# A kiegészítő információk fenntartása

- A csúcsokban tárolt kiegészítő információknak mindig naprakészeknek kell legyenek

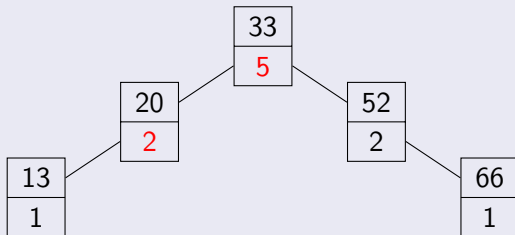
Példa: Töröl(40)



# A kiegészítő információk fenntartása

- A csúcsokban tárolt kiegészítő információknak mindig naprakészeknek kell legyenek

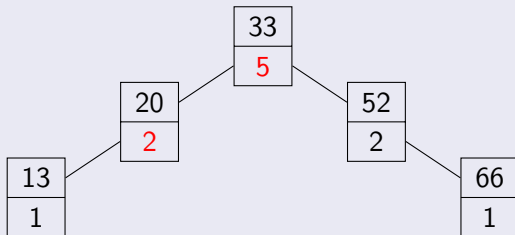
Példa: Töröl(40)



# A kiegészítő információk fenntartása

- A csúcsokban tárolt kiegészítő információknak mindig naprakészeknek kell legyenek

Példa: Töröl(40)



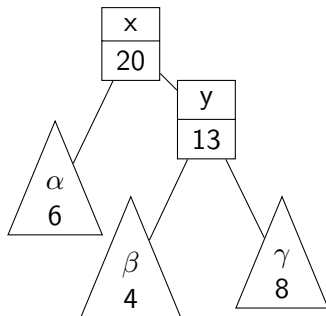
## Észrevétel

Épp ezért nem a csúcsok rangjára tekintünk közvetlenül kiegészítő információként (hiszen azt költséges lehet aktualizálni)



# Új művelet – Forgatás

- A keresőfákat forgatásokkal fogjuk tudni kiegyensúlyozni

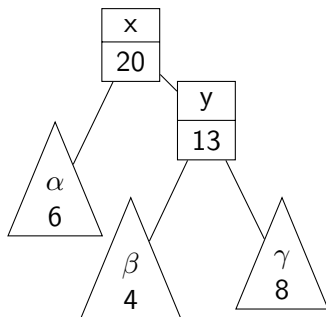
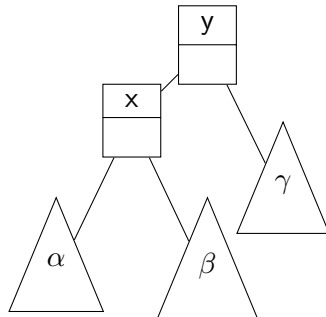


(a)  $x$  körüli balra forgatás előtt



## Új művelet – Forgatás

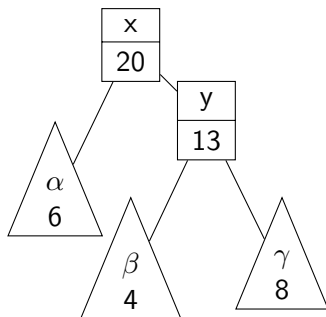
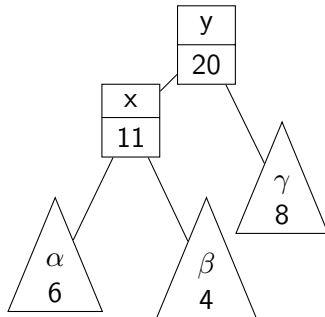
- A keresőfákat forgatásokkal fogjuk tudni kiegyensúlyozni

(a)  $x$  körüli balra forgatás előtt(b)  $x$  körüli balra forgatás után



## Új művelet – Forgatás

- A keresőfákat forgatásokkal fogjuk tudni kiegyensúlyozni

(a)  $x$  körüli balra forgatás előtt(b)  $x$  körüli balra forgatás után

# A kiegészítő információ fenntartása

- Beszúrás esetén: a beszúrás helyétől a gyökéig menően a kiegészítő információk inkrementálása ( $O(h)$ )
- Törlés esetén: a kieső csúcstól a gyökéig menően a kiegészítő információk dekrementálása ( $O(h)$ )
- $x$  körüli forgatás esetén ( $O(1)$ )
  - $y.kiegeszito = x.kiegeszito$
  - $x.kiegeszito = x.bal.kiegeszito + y.bal.kiegeszito + 1$ , azaz  $\alpha$ -beli és  $\beta$ -beli csúcsok száma  $+1$



# Intervallumfák

## Az intervallumfa sajátosságai

- A csúcsok  $[a; f]$  intervallumokat tárolnak
  - $a$  és  $f$  az intervallum alsó,-és felső végpontjait jelöli
  - $a \leq f$  feltehető
- $<$  rendezést az intervallumok kezdőpontja szerint értelmezzük
  - $[1; 4] < [5; 6]$
  - $[1; 4] < [3; 5]$
  - $[1; 4] < [2; 3]$



# Intervallumfák

## Az intervallumfa sajátosságai

- A csúcsok  $[a; f]$  intervallumokat tárolnak
  - $a$  és  $f$  az intervallum alsó,-és felső végpontjait jelöli
  - $a \leq f$  feltehető
- $<$  rendezést az intervallumok kezdőpontja szerint értelmezzük
  - $[1; 4] < [5; 6]$
  - $[1; 4] < [3; 5]$
  - $[1; 4] < [2; 3]$

## Speciális művelet: átfedő intervallum keresése

El akarjuk tudni dönteni, hogy a fa tartalmaz-e valamely  $I = [i_a; i_f]$  intervallummal átfedő intervallumot.

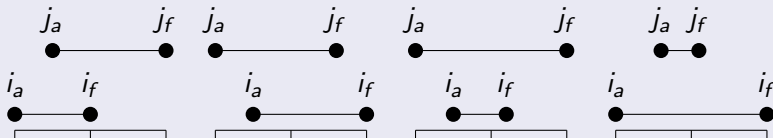
$I$  és  $J$  intervallumok átfednek  $\Leftrightarrow i_a \leq j_f$  és  $i_f \geq j_a$



# Intervallum trichotómia

- Az alábbi állítások közül pontosan egy teljesül bármely  $(I, J)$  intervallumpárra
  - a  $I \cap J \neq \emptyset$ , azaz  $I$  és  $J$  intervallumok átfedik egymást
  - b  $i_a > j_f$
  - c  $i_f < j_a$

## Illusztráció – 1. eset



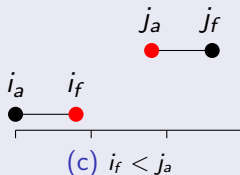
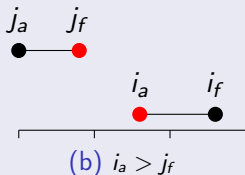
(a) Átfedő  $I$  és  $J$



## Intervallum trichotómia

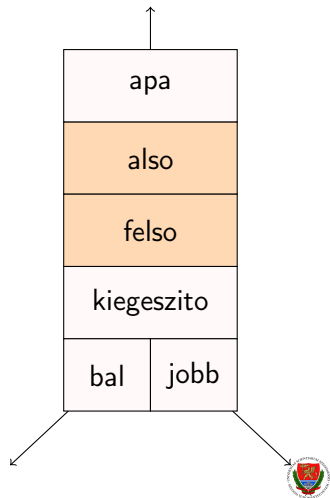
- Az alábbi állítások közül pontosan egy teljesül bármely  $(I, J)$  intervallumpárra
  - a  $I \cap J \neq \emptyset$ , azaz  $I$  és  $J$  intervallumok átfedik egymást
  - b  $i_a > j_f$
  - c  $i_f < j_a$

## Illusztráció – 2. és 3. eset



# Intervallumfa implementációja

```
class Node {  
    int also;  
    int felso;  
    int kiegészito;  
    Node *apa;  
    Node *bal;  
    Node *jobb;  
}
```

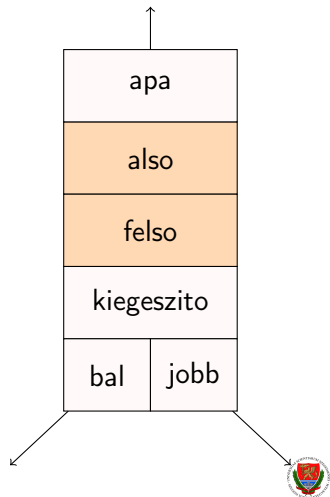


# Intervallumfa implementációja

```
class Node {  
    int also;  
    int felso;  
    int kiegészito;  
    Node *apa;  
    Node *bal;  
    Node *jobb;  
}
```

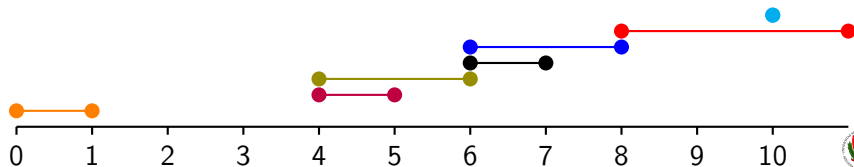
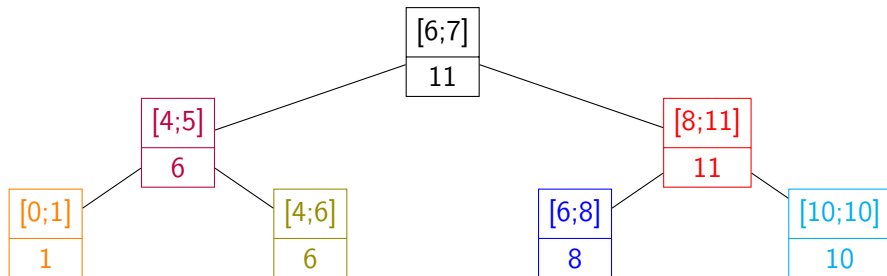
## Ötlet

A kiegészítő információ legyen az adott gyökerű részfában található maximális felső végpont





## Intervallumfa – példa



Átfedő intervallum keresése  $x$  gyökerű részében

```
ÁTFEDŐKERES(x, i) {  
    while (x != nil) {  
        if (i.also <= x.felso és i.felso >= x.also) {  
            return x    // átfedést találtunk  
        }  
        if (x.bal != nil és x.bal.kiegeszito >= i.also) {  
            x = x.bal    // folytassuk balra a keresést  
        } else {  
            x = x.jobb    // folytassuk jobbra a keresést  
        }  
    }  
    return nil    // nem találtunk átfedő intervallumot  
}
```



# Átfedő intervallum keresésének helyessége

## Tétel

*Az ÁtfedőKeres( $x, i$ ) minden végrehajtása csak abban az esetben tér vissza  $nil$ -l-lel, ha az  $x$  gyökerű intervallumfában nem található  $i$ -vel átfedő intervallum.*



# Átfedő intervallum keresésének helyessége – jobbra lépés

- Akkor lépünk jobbra, ha
  - Nincs bal részfa, vagy
  - Van bal részfa, de még a legnagyobb felső végpont is elmarad  $i$  intervallum alsó végpontjától (trichotómia b) esete)



# Átfedő intervallum keresésének helyessége – jobbra lépés

- Akkor lépünk jobbra, ha
    - Nincs bal részfa, vagy
    - Van bal részfa, de még a legnagyobb felső végpont is elmarad  $i$  intervallum alsó végpontjától (trichotómia b) esete)
- ⇒ jobbra lépéskor a bal részfában biztosan nincs átfedés



# Átfedő intervallum keresésének helyessége – jobbra lépés

- Akkor lépünk jobbra, ha
    - Nincs bal részfa, vagy
    - Van bal részfa, de még a legnagyobb felső végpont is elmarad  $i$  intervallum alsó végpontjától (trichotómia b) esete)
- ⇒ jobbra lépéskor a bal részében biztosan nincs átfedés

## Összegezve

Vagy fogunk jobbra lépve átfedő intervallumot találni, vagy balra lépve se találtunk volna



# Átfedő intervallum keresésének helyessége – balra lépés

- Előfordulhat-e, hogy  $x$ -nél balra menve nem találunk  $i$ -vel átfedő intervallumot, jobbra menve azonban találhatnánk?

---

<sup>1</sup> $x.bal$  az  $x$  csúcs bal fiát a gyökeréül tudó részfát jelöli



# Átfedő intervallum keresésének helyessége – balra lépés

- Előfordulhat-e, hogy  $x$ -nél balra menve nem találunk  $i$ -vel átfedő intervallumot, jobbra menve azonban találhatnánk?
  - Indirekt bizonyítás: tegyük fel, hogy előfordulhat ilyen
    - Balra mentünk, tehát  $\exists y \in x.bal^1$ , melyre  $y.felso > i.also$

---

<sup>1</sup> $x.bal$  az  $x$  csúcs bal fiát a gyökeréül tudó részfat jelöli





# Átfedő intervallum keresésének helyessége – balra lépés

- Előfordulhat-e, hogy  $x$ -nél balra menve nem találunk  $i$ -vel átfedő intervallumot, jobbra menve azonban találhatnánk?
  - Indirekt bizonyítás: tegyük fel, hogy előfordulhat ilyen
    - Balra mentünk, tehát  $\exists y \in x.bal^1$ , melyre  $y.felso > i.also$
    - Feltevésünk szerint  $x.bal$ -ban nincs  $i$ -vel átfedő intervallum  $\rightarrow$  trichotómia c) esete

---

<sup>1</sup> $x.bal$  az  $x$  csúcs bal fiát a gyökeréül tudó részfat jelöli



# Átfedő intervallum keresésének helyessége – balra lépés

- Előfordulhat-e, hogy  $x$ -nél balra menve nem találunk  $i$ -vel átfedő intervallumot, jobbra menve azonban találhatnánk?
  - Indirekt bizonyítás: tegyük fel, hogy előfordulhat ilyen
    - Balra mentünk, tehát  $\exists y \in x.bal^1$ , melyre  $y.felso > i.also$
    - Feltevésünk szerint  $x.bal$ -ban nincs  $i$ -vel átfedő intervallum  $\rightarrow$  trichotómia c) esete  $\rightarrow y.also > i.felso$
    - A keresőfa-tulajdonságból adódóan pedig  $\forall y' \in x.jobb$  esetében  $y'.also > y.also$  teljesül

---

<sup>1</sup> $x.bal$  az  $x$  csúcs bal fiát a gyökeréül tudó részfat jelöli



# Átfedő intervallum keresésének helyessége – balra lépés

- Előfordulhat-e, hogy  $x$ -nél balra menve nem találunk  $i$ -vel átfedő intervallumot, jobbra menve azonban találhatnánk?
  - Indirekt bizonyítás: tegyük fel, hogy előfordulhat ilyen
    - Balra mentünk, tehát  $\exists y \in x.bal^1$ , melyre  $y.felso > i.also$
    - Feltevésünk szerint  $x.bal$ -ban nincs  $i$ -vel átfedő intervallum  $\rightarrow$  trichotómia c) esete  $\rightarrow y.also > i.felso$
    - A keresőfa-tulajdonságból adódóan pedig  $\forall y' \in x.jobb$  esetében  $y'.also > y.also$  teljesül
    - Tehát  $\forall y' \in x.jobb$  az  $i$  intervallumra nézve c)-típusú (vele át nem fedő) lehet csupán

## Összegezve

Ha balra lépve nem sikerül átfedő intervallumot találjunk, akkor jobbra lépve se találhattunk volna

<sup>1</sup> $x.bal$  az  $x$  csúcs bal fiát a gyökeréül tudó részfat jelöli



# Összegzés

- Adicionális adattagok bevezetésével újfajta kérdések hatékony megválaszolása válhat lehetővé
- Adatszerkezetek kibővítésének fő lépései
  - 1 Alap-adatszerkezet megválasztása (nem csak bináris keresőfa)
  - 2 Alap-adatszerkezet kiegészítő információjának kijelölése
  - 3 Kiegészítő információjának **hatékony** fenntarthatóságának igazolása
  - 4 Új hatékony műveletek kifejlesztése

