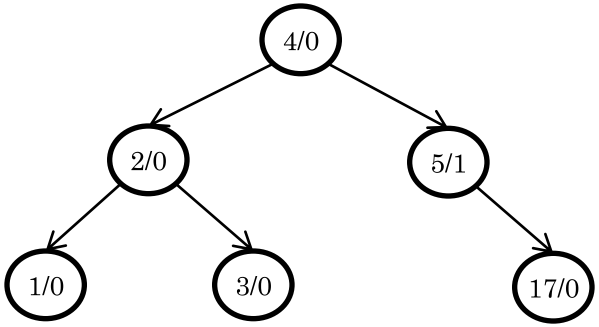


AVL-DREVO (angl. *AVL tree*)

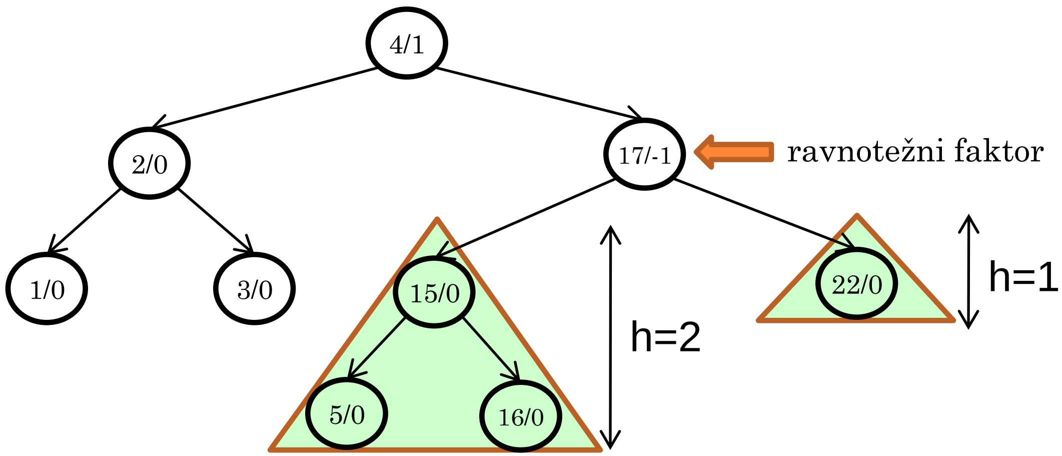
AVL-DREVO

- poimenovano po avtorjih Adelson-Velskii in Landis
- delno poravnano binarno iskalno drevo
- za vsako vozlišče velja, da se višini obeh poddreves razlikujeta največ za 1
- višina maksimalno izrojenega AVL-drevesa z *n* elementi je:

$$h \le 1.44 \log_2(n+1)$$

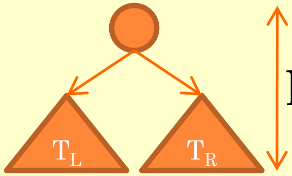
- zahtevnost osnovnih operacij je reda $O(\log n)$

Primer AVL-drevesa



- ravnotežni faktor vozlišča je razlika višin desnega in levega poddrevesa

- višina praznega drevesa je 0
- višina drevesa je dolžina najdaljše poti od korena do listov



$$h = \max(h(T_L), h(T_R)) + 1$$

Implementacija AVL-drevesa

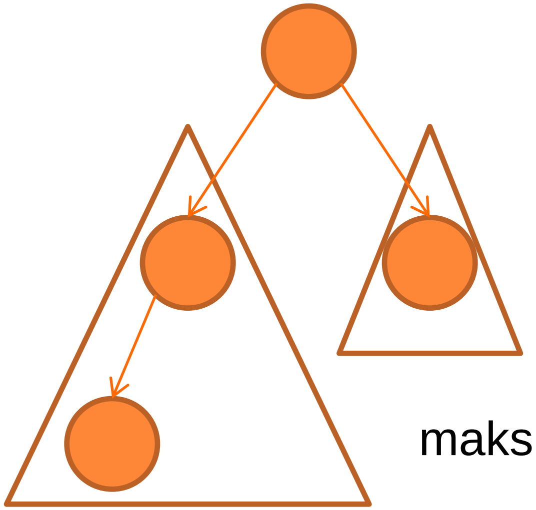
K običajnemu vozlišču BST dodamo še ravnotežni faktor:

```
public class AVLTreeNode extends BSTreeNode {
  int balance ;
} // class AVLTreeNode
```

```
public class BSTreeNode {
  Comparable key ;
  BSTreeNode left, right ;
} // class BSTreeNode
```

Višina AVL-dreves

Zgradimo maksimalno izrojeno AVL-drevo:

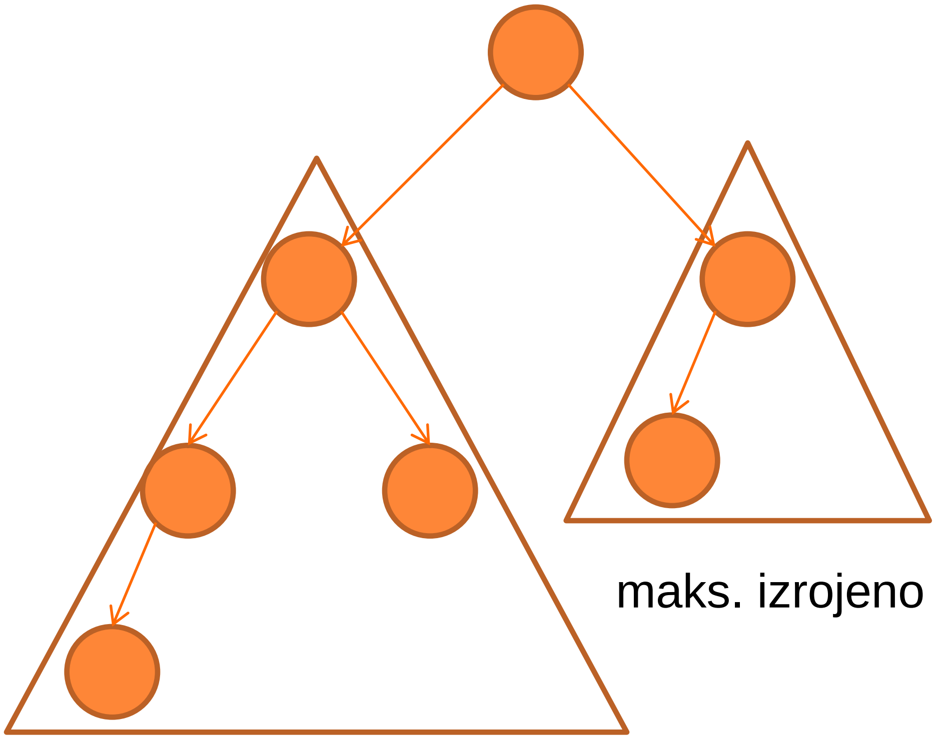


maks.izr. v=1

maks. izrojeno v=2

Višina AVL-dreves

Zgradimo maksimalno izrojeno AVL-drevo:

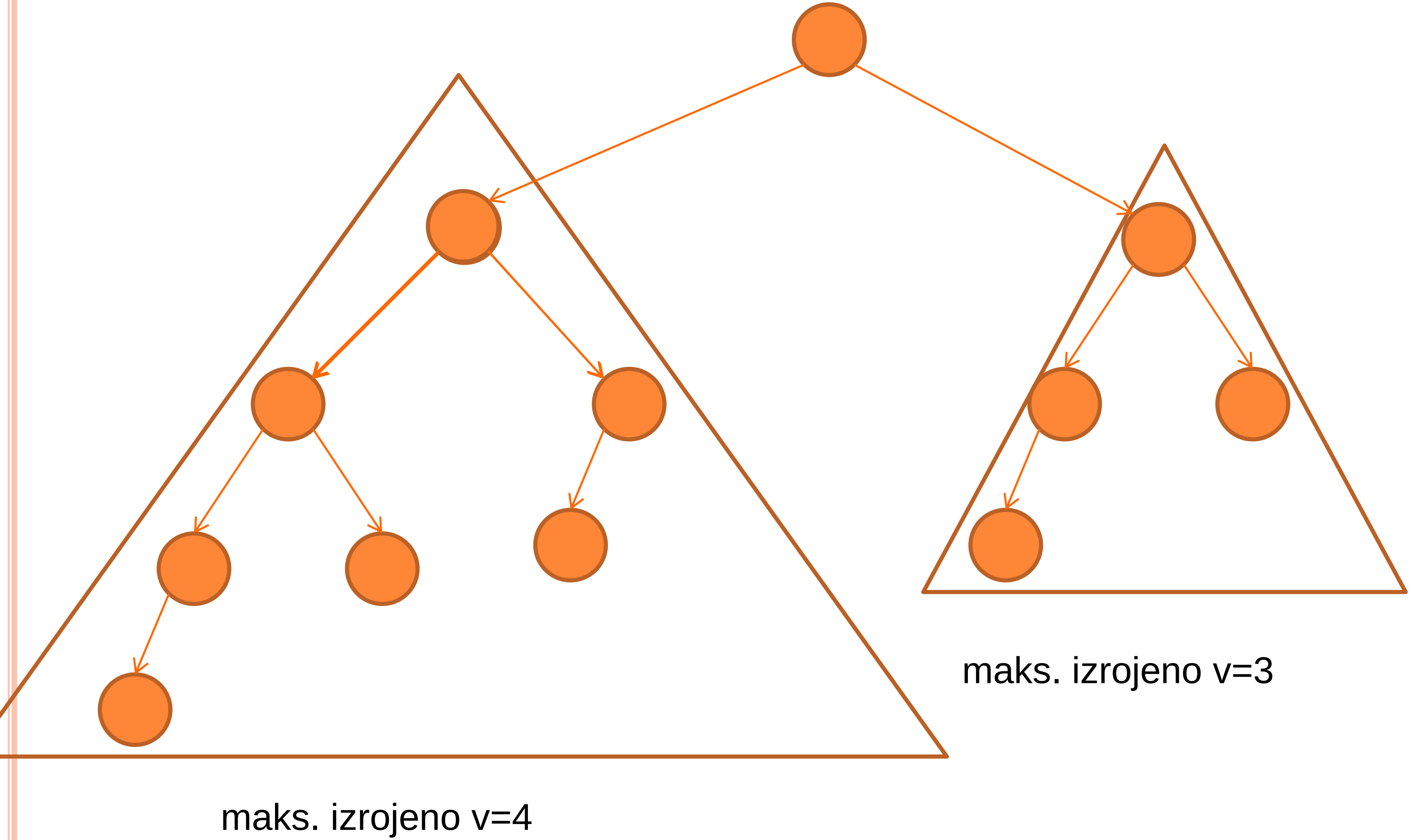


maks. izrojeno v=2

maks. izrojeno v=3

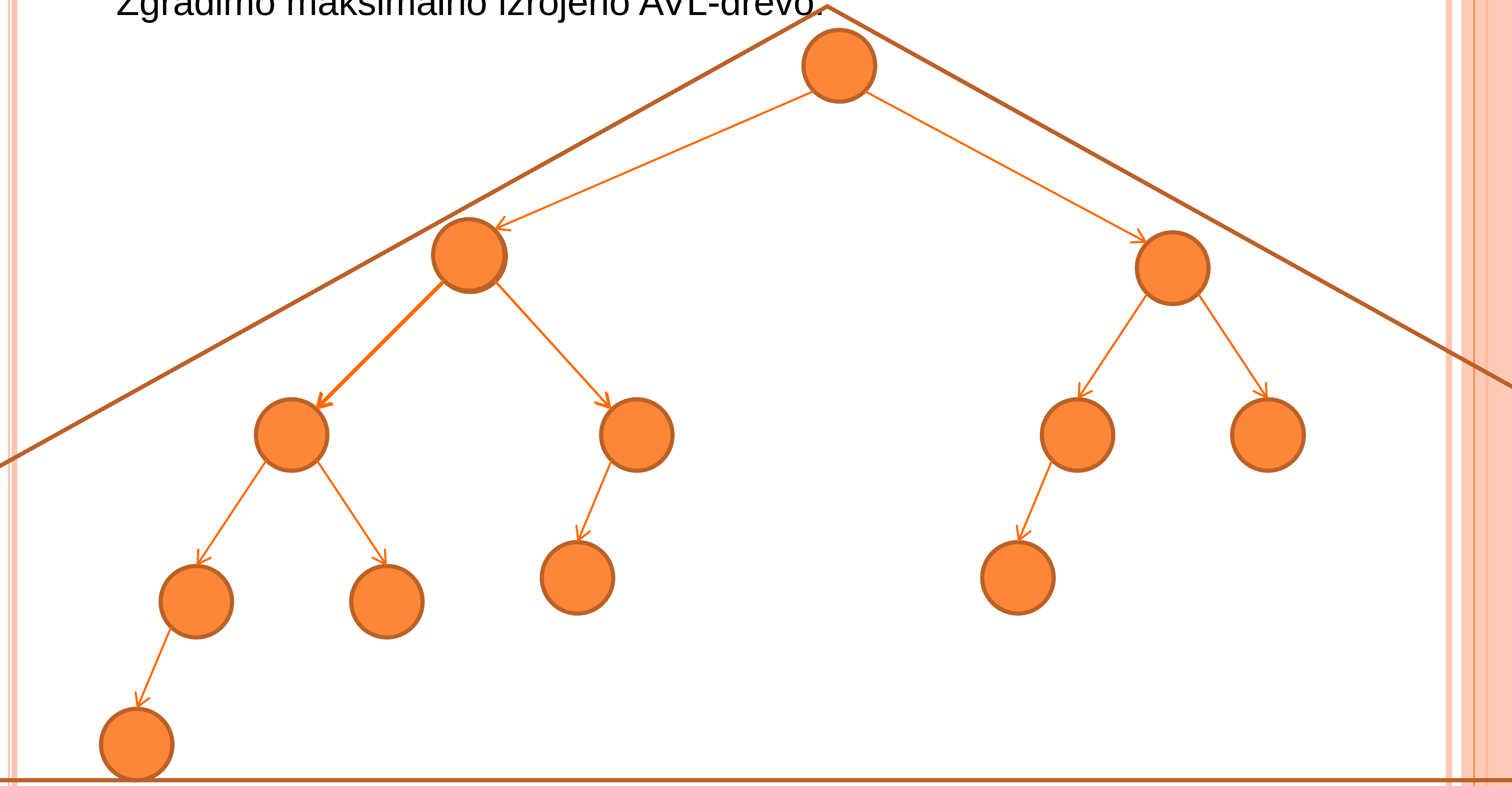
Višina AVL-dreves

Zgradimo maksimalno izrojeno AVL-drevo:



Višina AVL-dreves

Zgradimo maksimalno izrojeno AVL-drevo:



maks. izrojeno v=5

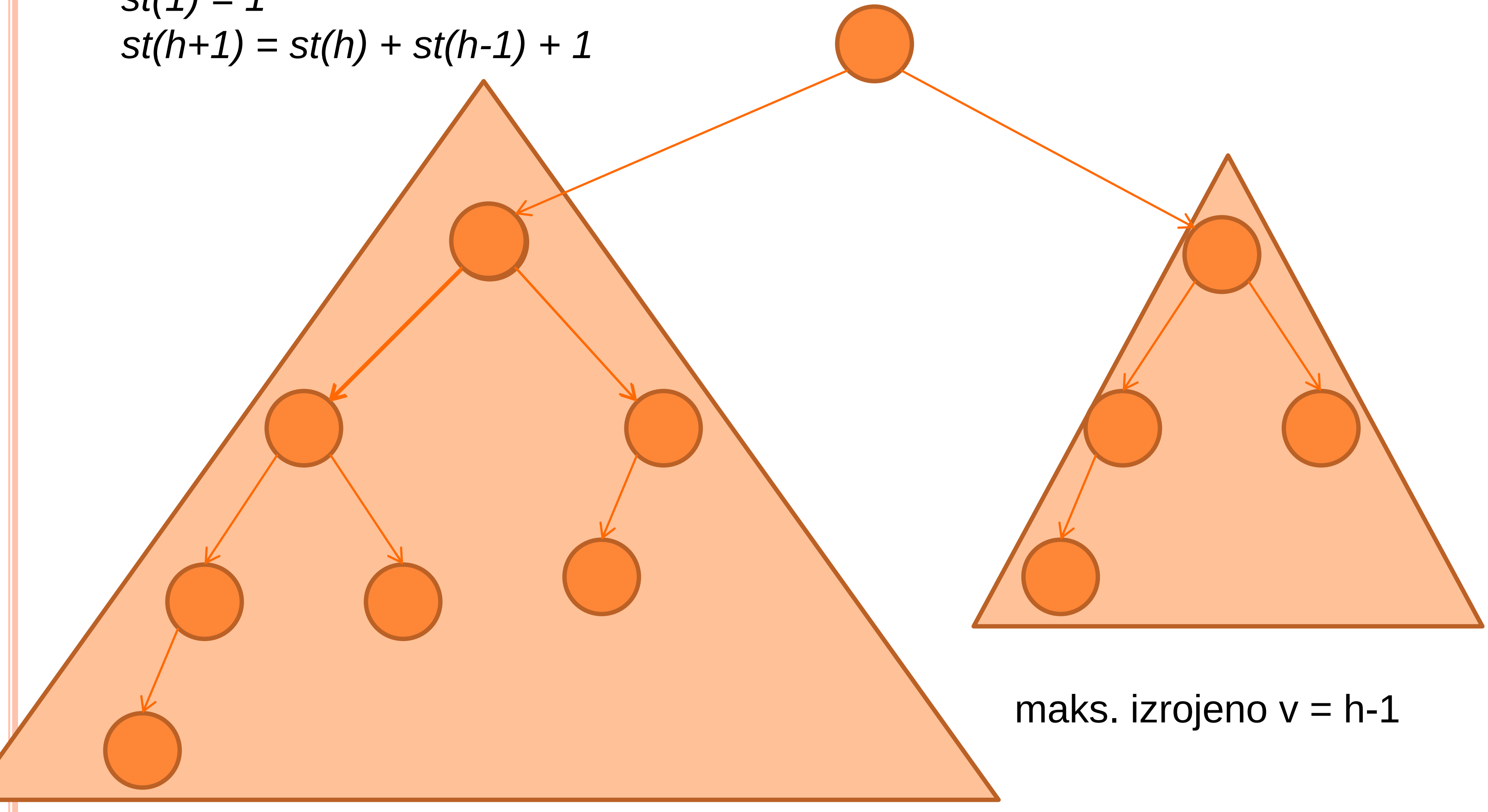
Višina AVL-dreves

Enačba za število vozlišč maksimalno izrojenega drevesa:

$st(0) = 0$

$st(1) = 1$

$st(h+1) = st(h) + st(h-1) + 1$



maks. izrojeno v = h-1

maks. izrojeno v = h

Višina AVL-dreves

Enačba za število vozlišč maksimalno izrojenega drevesa:

$st(0) = 0$

$st(1) = 1$

$st(h+1) = st(h) + st(h-1) + 1$

Fibonaccijeva števila:

$Fib(0) = 0$

$Fib(1) = 1$

$Fib(h) = Fib(h-1) + Fib(h-2)$

$$\boxed{Fib(h+2) = st(h) + 1}$$

Za vajo dokažite to enačbo z matematično indukcijo po *h*.

Višina AVL-dreves

Za vajo dokažite še tole enačbo z indukcijo po *h*:

$$\phi^h \leq Fib(h+2) \leq \phi^{h+1}, h \geq 0$$

$$\phi = \frac{1+\sqrt{5}}{2} \approx 1.618$$

Višina AVL-dreves

Sedaj lahko izpeljemo število vozlišč *n* iz:

st(h) +1 = Fib(h+2) = n+1

in

$$\phi^h \leq Fib(h+2) \leq \phi^{h+1}, h \geq 0$$

$$\phi^h \leq n+1 \leq \phi^{h+1}$$

$$\boxed{h \leq \log_\phi(n+1) \approx 1.44 \log_2(n+1)}$$

AVL-DREVO

Zagotavlja časovno zahtevnost osnovnih operacij reda *O(log n)*:

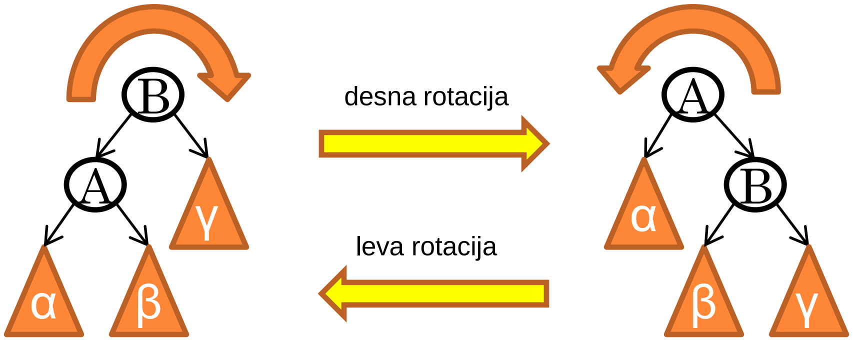
- **iskanje**: enako kot pri običajnem BST → *O(log n)*
- **dodajanje**:
  - dodamo list;
  - preračunavamo ravnotežne faktorje navzgor;
  - eventuelno potrebna rotacija, ki postopek zaključi, sicer se do korena preračunavajo faktorji→ $O(2\log n) = O(\log n)$
- **brisanje**:
  - nadomestimo element z minimalnim iz desnega poddrevesa (ali z maksimalnim iz levega poddrevesa);
  - preračunavamo ravnotežne faktorje navzgor;
  - eventuelno potrebne rotacije;
  - v najslabšem primeru se do korena preračunavajo faktorji→ $O(2\log n) = O(\log n)$

Rotacije drevesa

- ob spremembah strukture AVL-drevesa (z operacijami dodajanja in brisanja elementov) je potrebno drevo transformirati in ohraniti delno poravnanost.

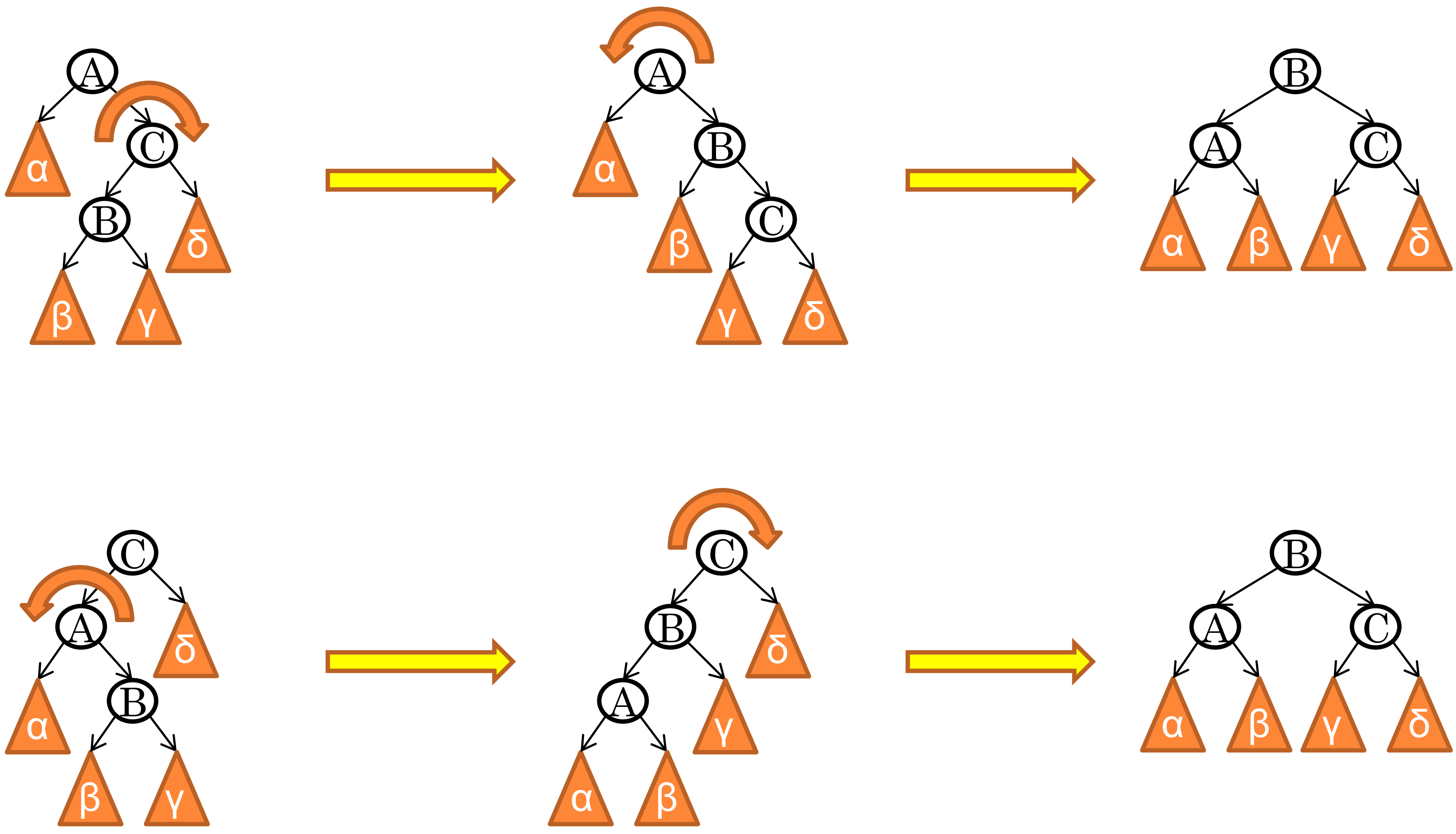
- drevo transformiramo z operacijami enojne in dvojne rotacije

Enojna rotacija:



Rotacije drevesa

Dvojna rotacija:



Dodajanje elementa v AVL-drevo (1/3)

1. Element dodamo v list drevesa kot pri navadnem BST.

2. Preverimo ravnotežni faktor vseh vozlišč na poti navzgor od vstavljenega lista do korena drevesa.

   - če je absolutna vrednost ravnotežnega faktorja večja kot 1, je potrebno drevo popravljati

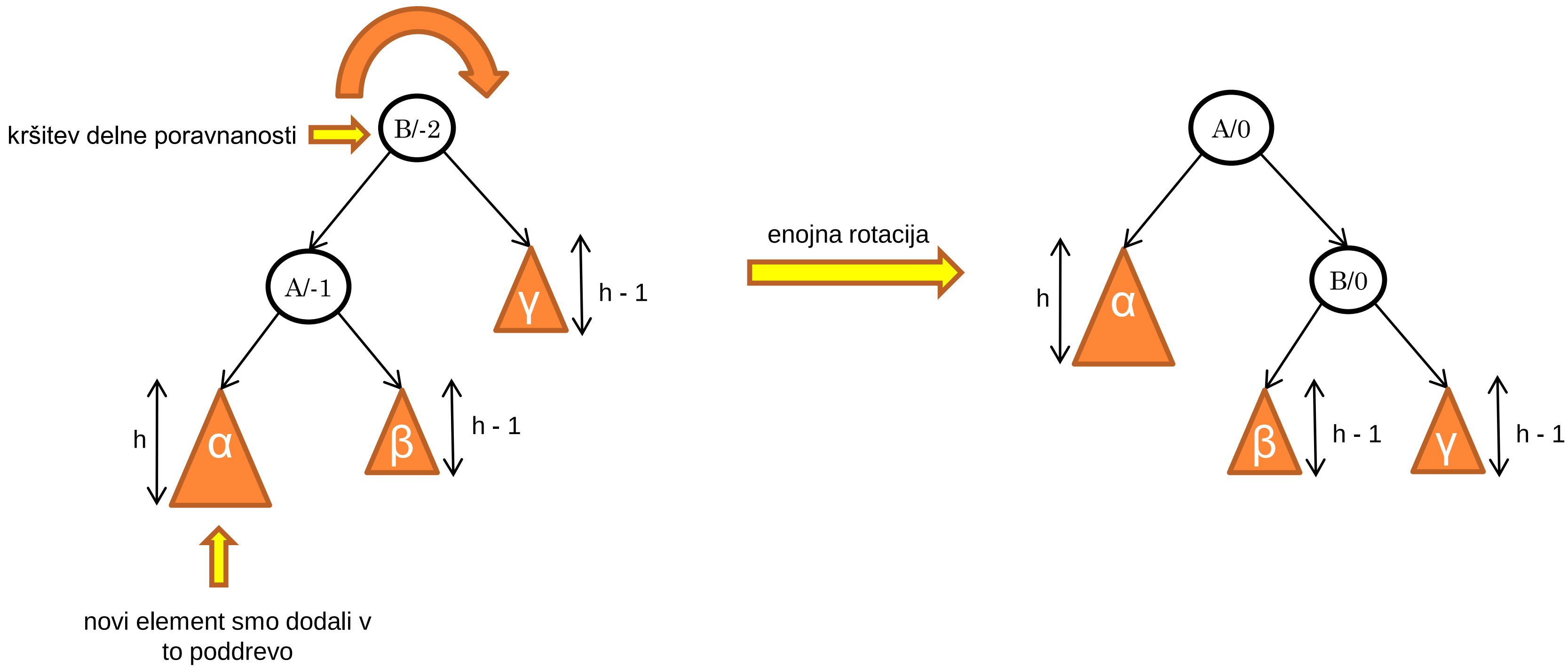
   - v najslabšem primeru je potrebno popravljati ravnotežni faktor vse do korena - ko pa pride do rotacije (enojne ali dvojne), je postopek zaključen

   - časovna kompleksnost je reda $O(\log n)$

Dodajanje elementa v AVL-drevo (2/3)

Možna sta dva primera:

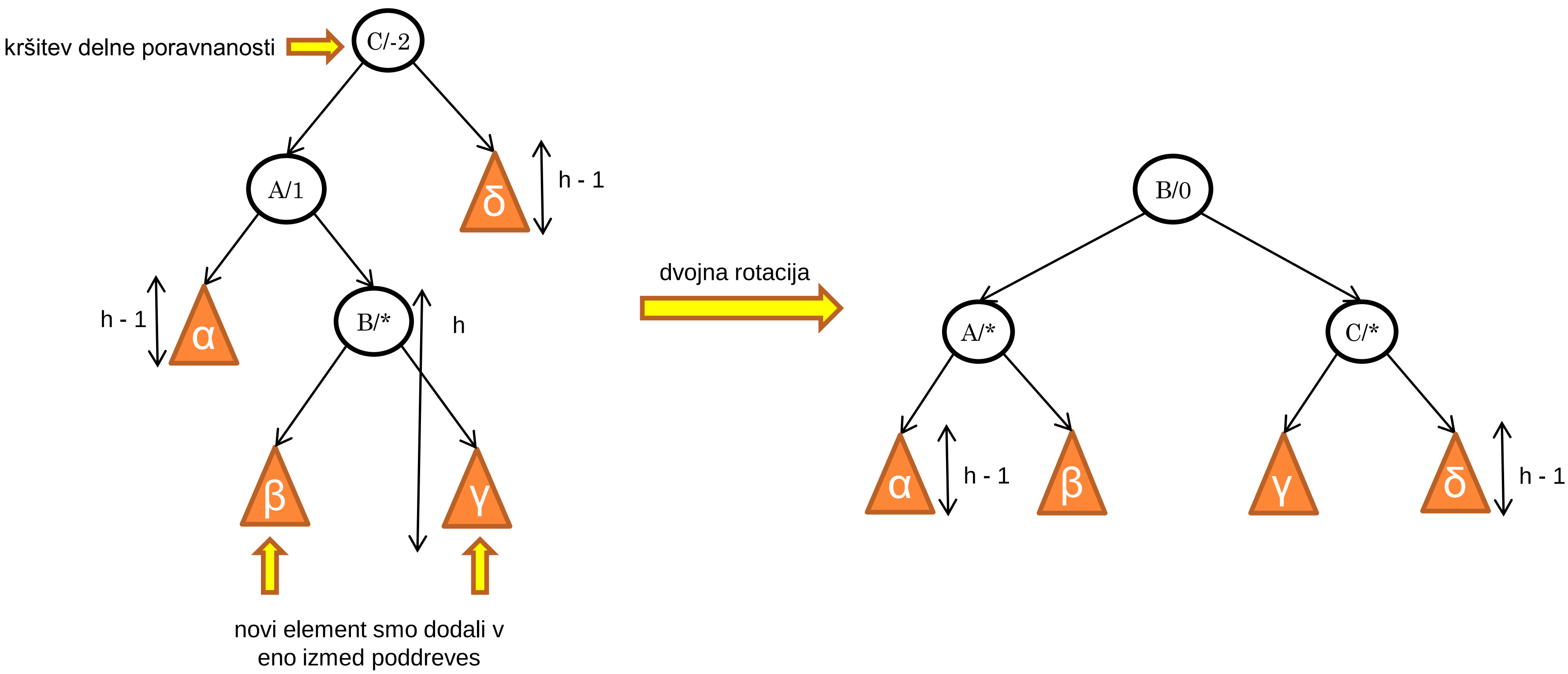
1. Koren ima absolutno vrednost ravnotežnega faktorja 2, sin pa 1 in imata oba faktorja **isti** predznak:



DODAJANJE ELEMENTA V AVL-DREVO (3/3)

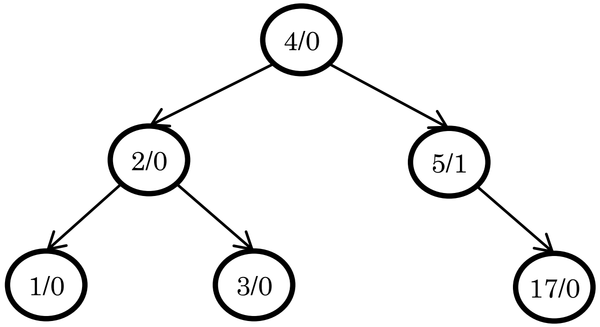
Možna sta dva primera:

2. Koren ima absolutno vrednost ravnotežnega faktorja 2, sin pa 1 in faktorja imata **različna** predznaka:



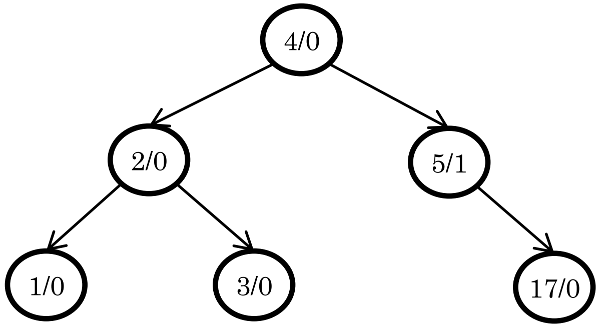
Primer (1/8)

Podano je AVL drevo na sliki. V drevo dodajte naslednje elemente: 22, 16, 15 in 14.



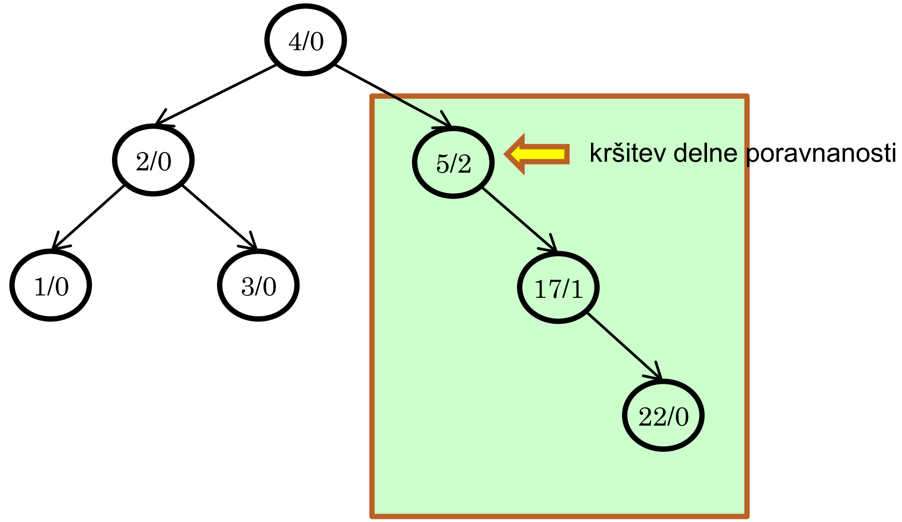
Primer (2/8)

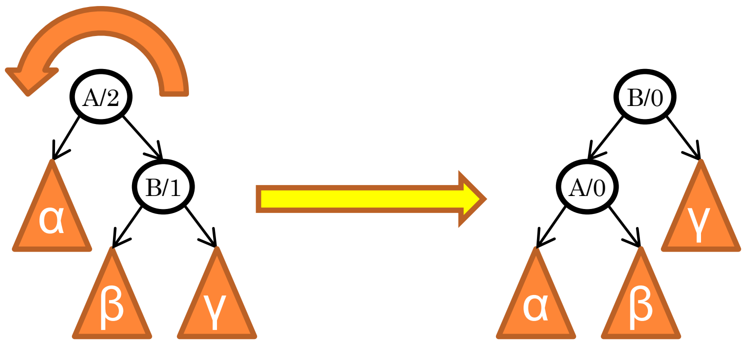
Dodamo 22…



PRIMER (2/8)

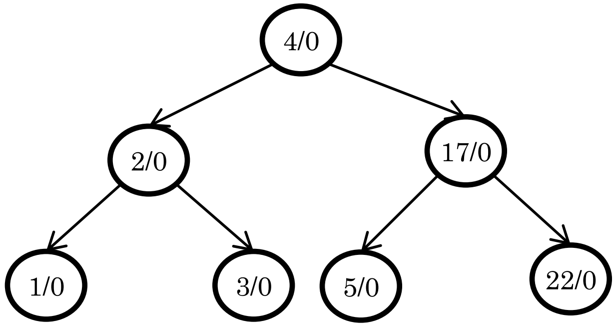
Dodamo 22...





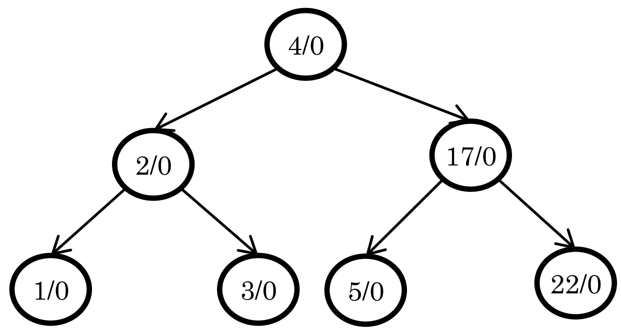
Primer (3/8)

Dodamo 22.



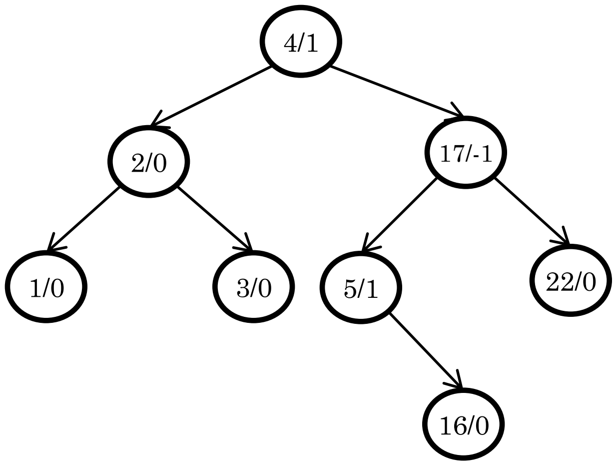
Primer (4/8)

Dodamo 16.



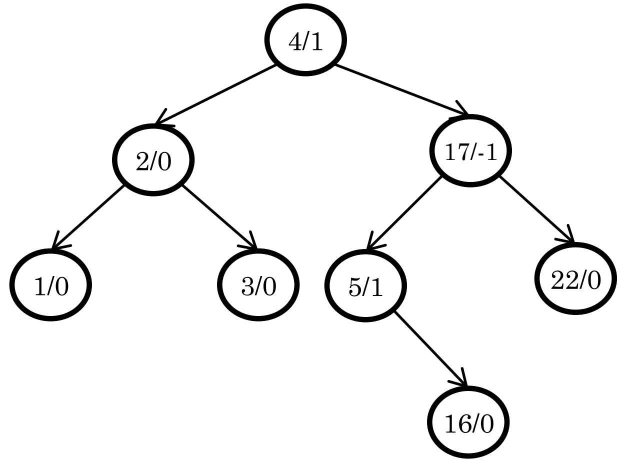
Primer (4/8)

Dodamo 16.



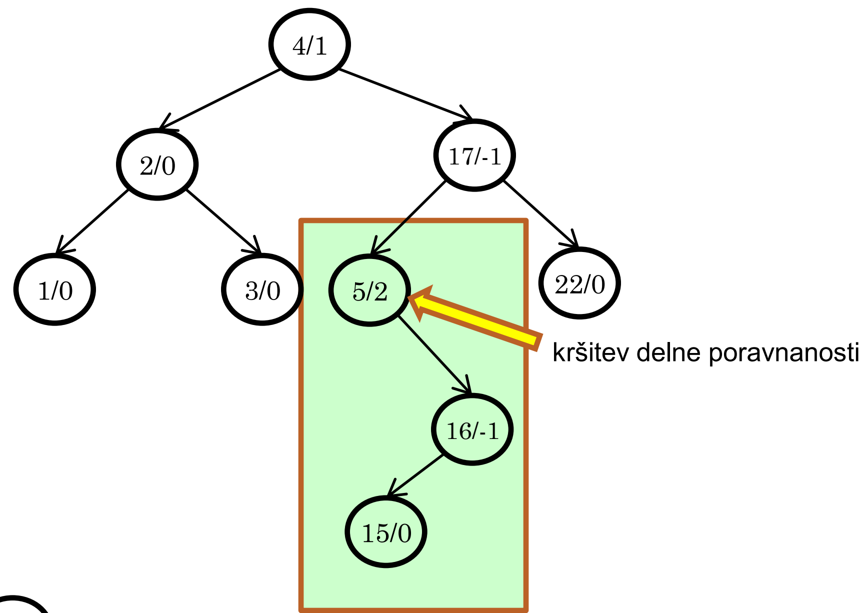
Primer (5/8)

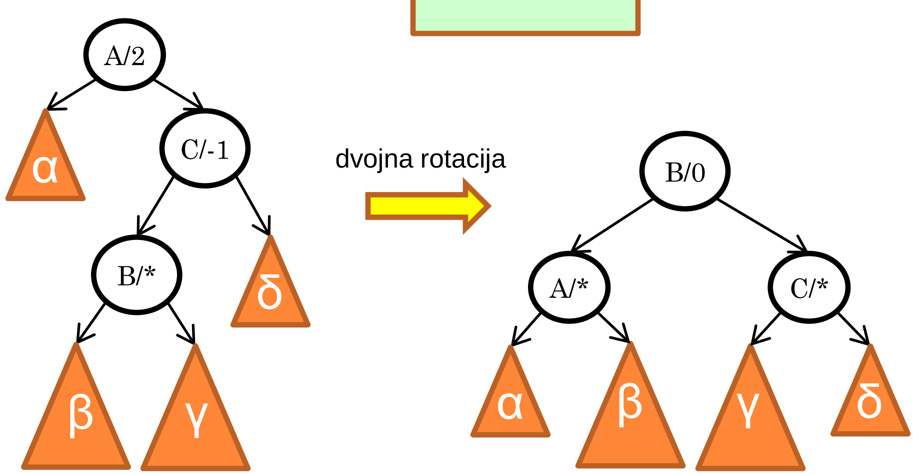
Dodamo 15.



PRIMER (5/8)

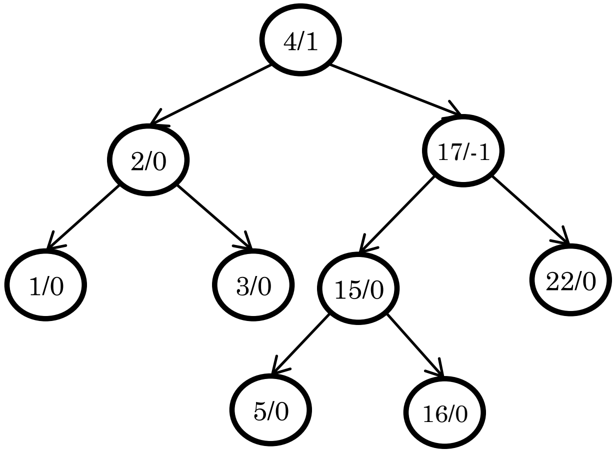
Dodamo 15…





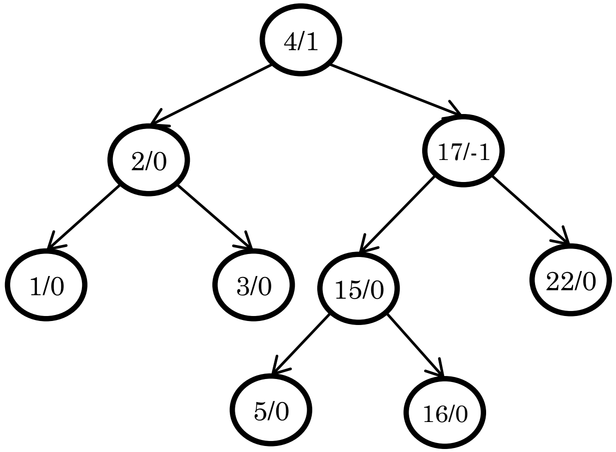
PRIMER (6/8)

Dodamo 15.



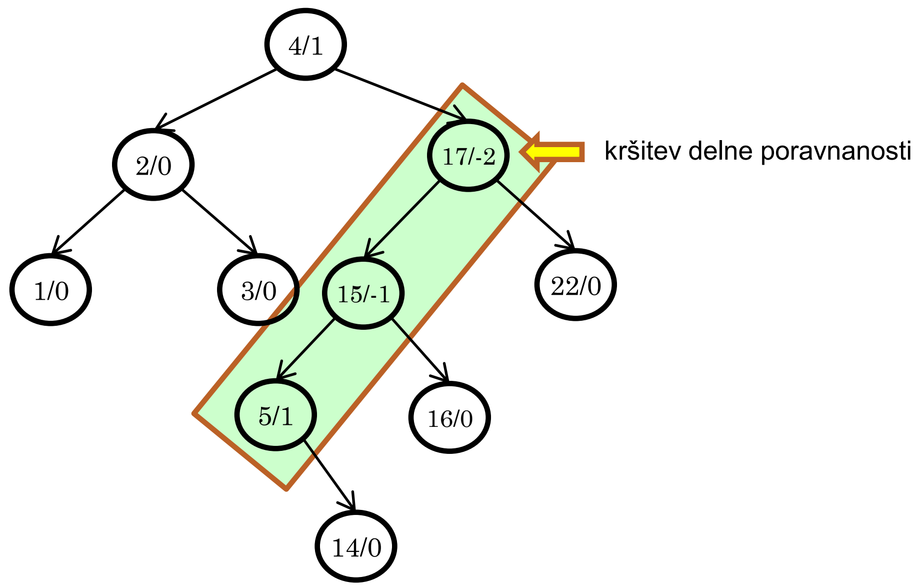
Primer (7/8)

Dodamo 14.



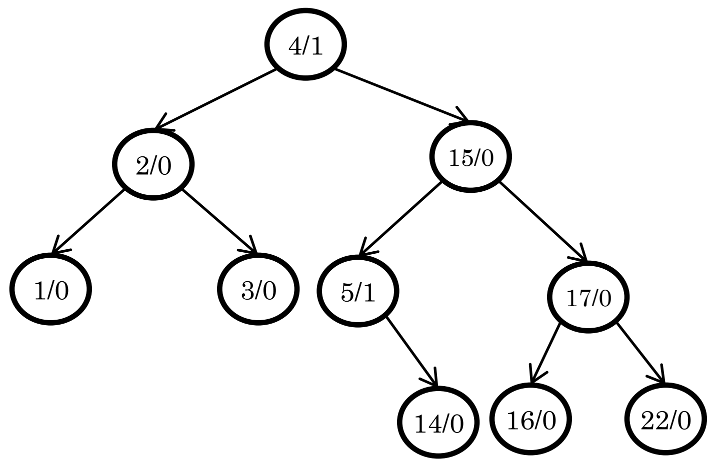
Primer (7/8)

Dodamo 14…



Primer (8/8)

Dodamo 14.



Brisanje elementa iz AVL-drevesa

1. Element brišemo iz drevesa kot pri navadnem BST:
   - če je element list drevesa, ga enostavno zbrišemo
   - če ima element samo enega sina, ga zbrišemo ter na njegovo mesto postavimo njegovega sina
   - če ima element dva sina, zbrišemo največji element iz levega poddrevesa ali najmanjši element iz desnega poddrevesa, ki nadomesti dejansko zbrisano vozlišče

2. Preverimo ravnotežni faktor vseh vozlišč na poti navzgor od očeta dejansko zbrisanega vozlišča do korena drevesa.

   - če je absolutna vrednost ravnotežnega faktorja večja kot 1, je potrebno drevo popravljati

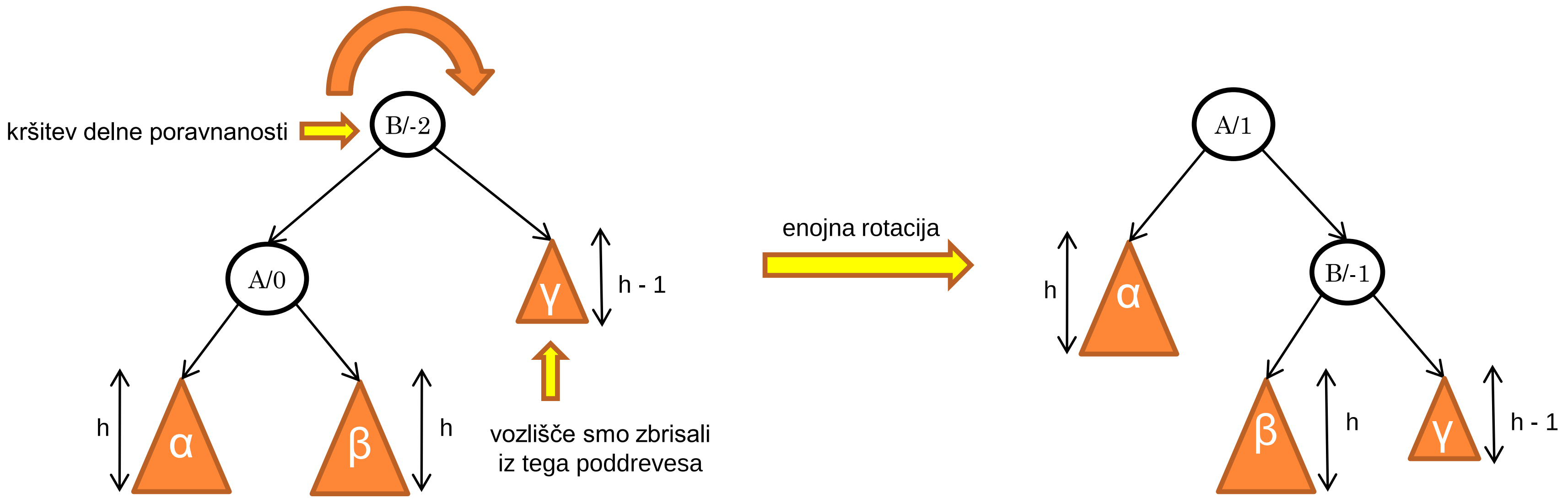
   - v najslabšem primeru je potrebno po celi poti od zbrisanega vozlišča do korena poravnavati drevo

   - časovna kompleksnost je reda $O(\log n)$

Brisanje elementa iz AVL-drevesa (1/3)

Možna sta oba primera analogna tistim ob dodajanju elementa v drevo.

Ob brisanju elementa iz drevesa je možen tudi primer, ko ima sin ravnotežni faktor enak 0.

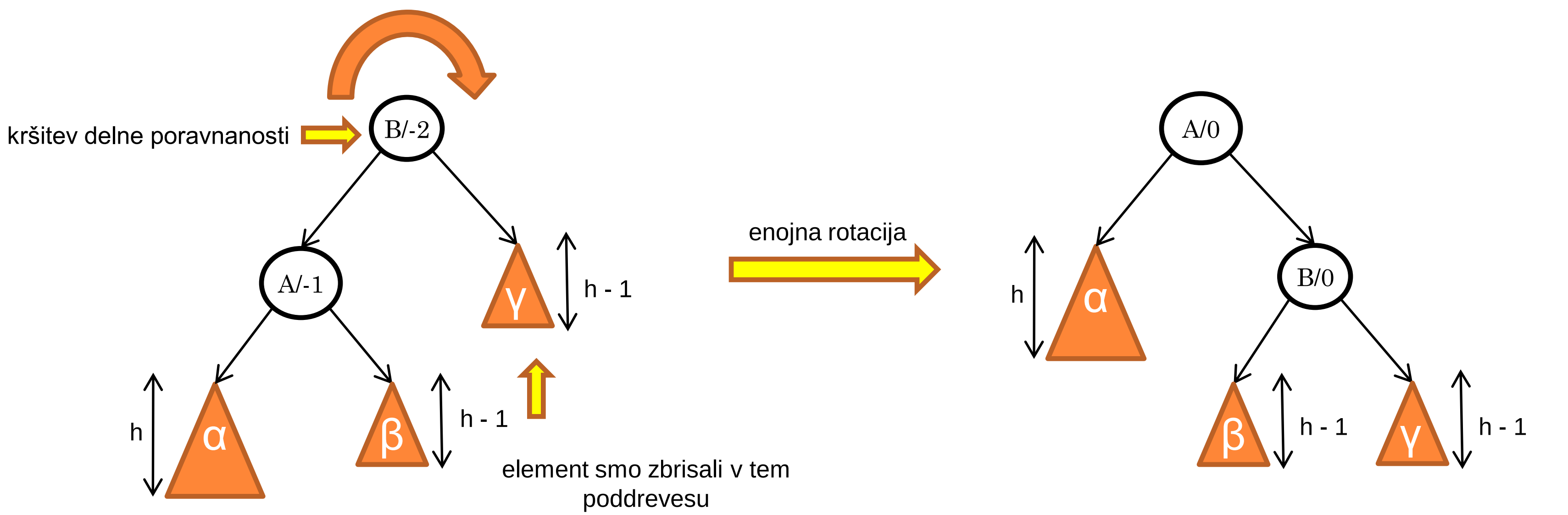


Poddrevo je enako visoko kot pred brisanjem → končamo

Brisanje elementa iz AVL-drevesa (2/3)

Možna sta oba primera analogna tistim ob dodajanju elementa v drevo:

1. Koren ima absolutno vrednost ravnotežnega faktorja 2, sin pa 1 in imata oba faktorja **isti** predznak:

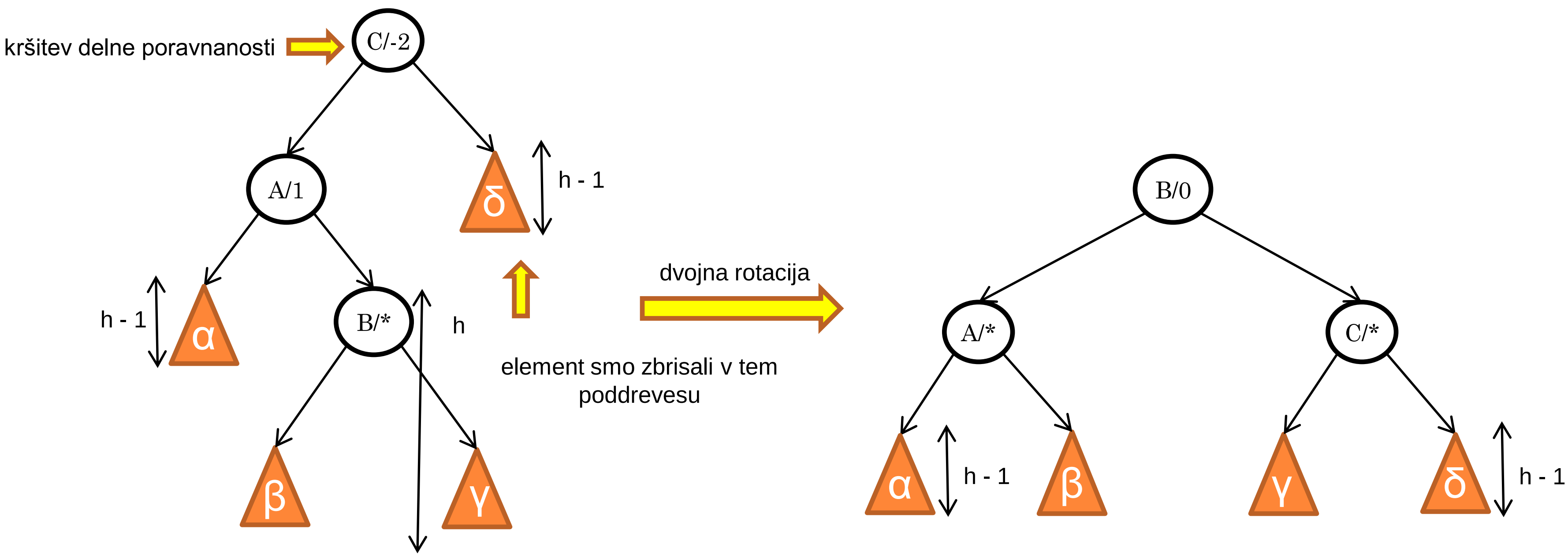


Poddrevo je za ena nižje kot pred brisanjem → nadaljujemo s popravljnjem faktorjev navzgor

Brisanje elementa iz AVL-drevesa (3/3)

Možna sta oba primera analogna tistim ob dodajanju elementa v drevo:

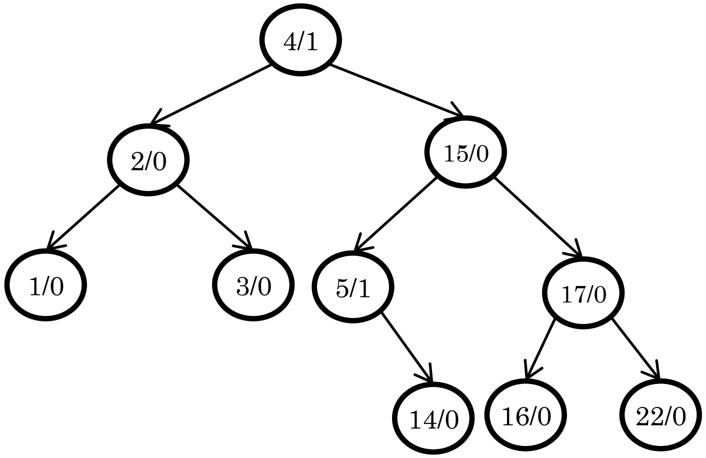
2. Koren ima absolutno vrednost ravnotežnega faktorja 2, sin pa 1 in faktorja imata **različna** predznaka:



Poddrevo je za ena nižje kot pred brisanjem → nadaljujemo s popravljnjem faktorjev navzgor

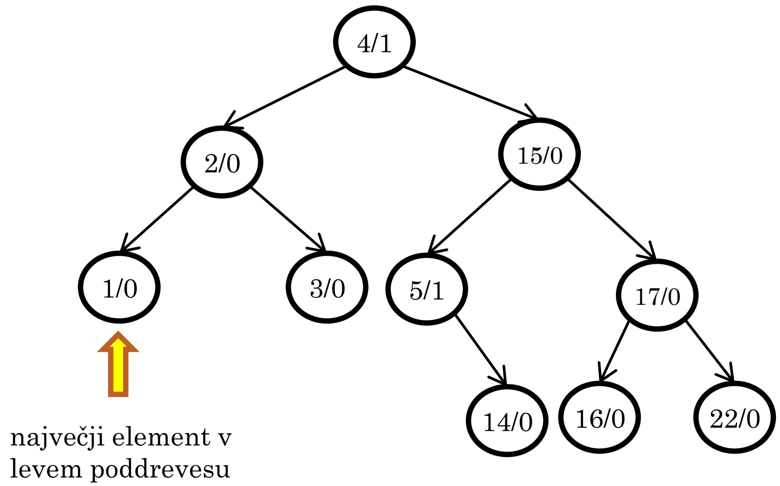
Primer (1/9)

Podano je AVL-drevo. Izbriši elemente 2, 1, 22 in 17 v tem vrstnem redu.



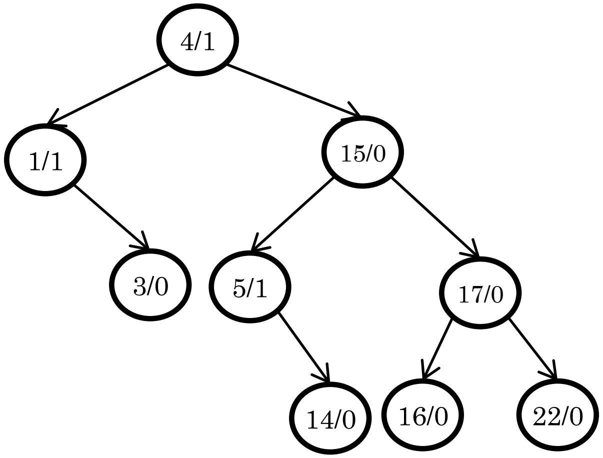
PRIMER (2/9)

Brišemo 2…



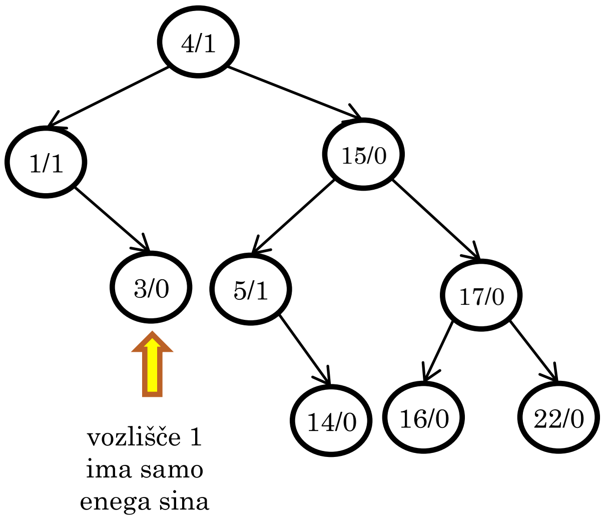
PRIMER (3/9)

Brišemo 2.



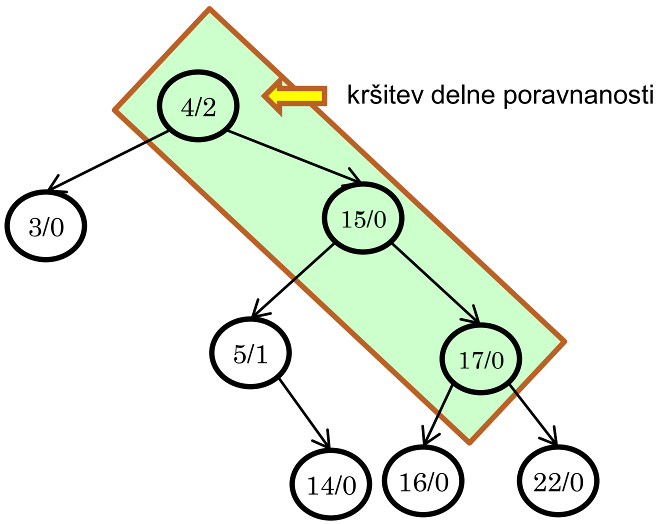
Primer (4/9)

Brišemo 1…



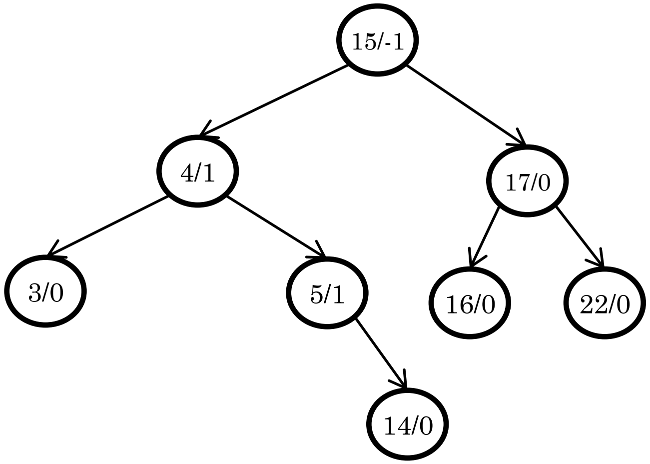
Primer (5/9)

Brišemo 1…



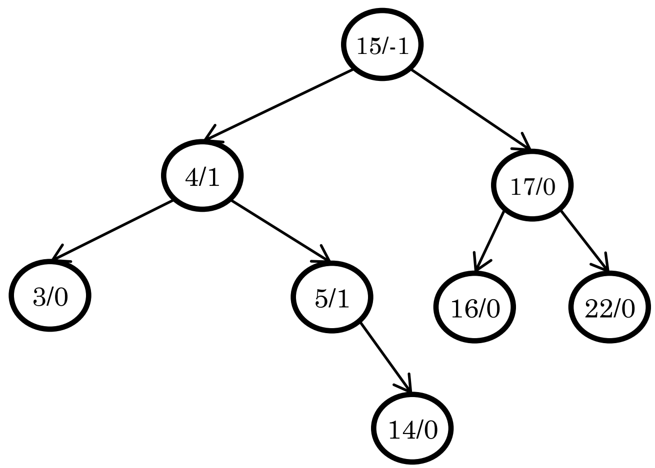
Primer (6/9)

Brišemo 1.



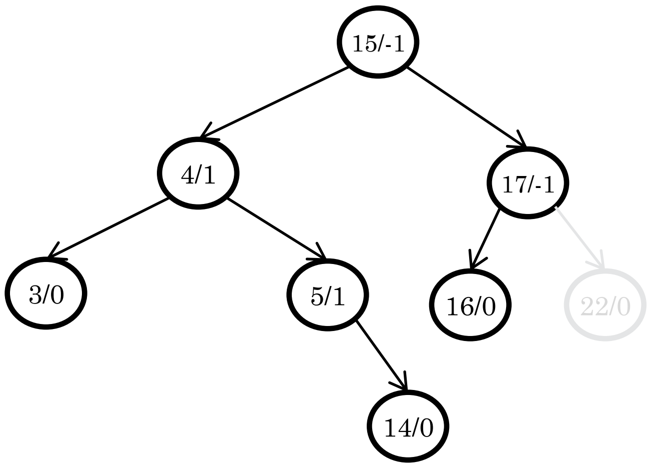
Primer (7/9)

Brišemo 22.



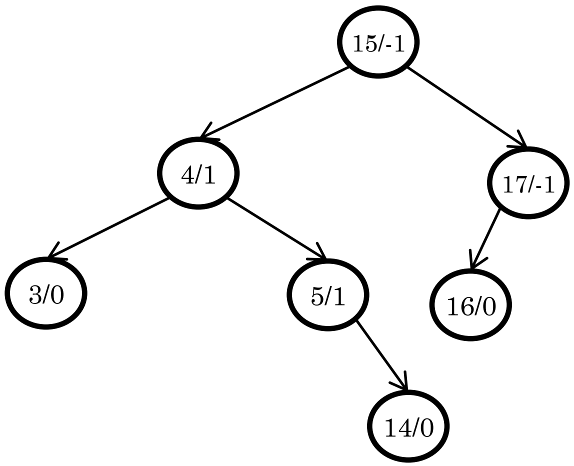
PRIMER (7/9)

Brišemo 22.



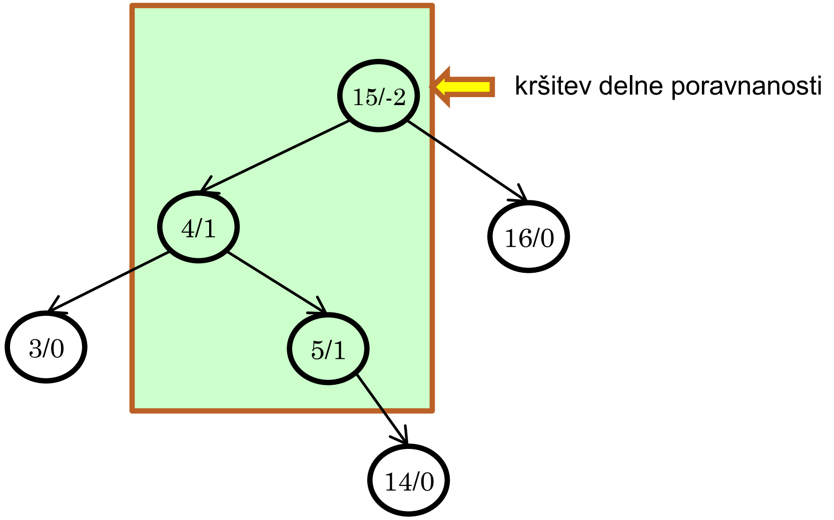
Primer (8/9)

Brišemo 17.



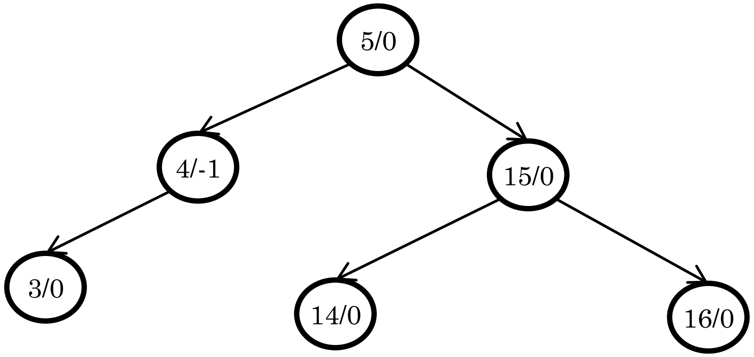
Primer (8/9)

Brišemo 17…



Primer (9/9)

Brišemo 17.



Povzetek

- delno poravnano binarno iskalno drevo

- za vsako vozlišče velja, da se višini obeh poddreves razlikujeta največ za 1

- višina maksimalno izrojenega AVL-drevesa z *n* elementi je:

$$h \leq 1.44 \log_2(n+1)$$

- zahtevnost vseh operacij je reda $O(\log n)$