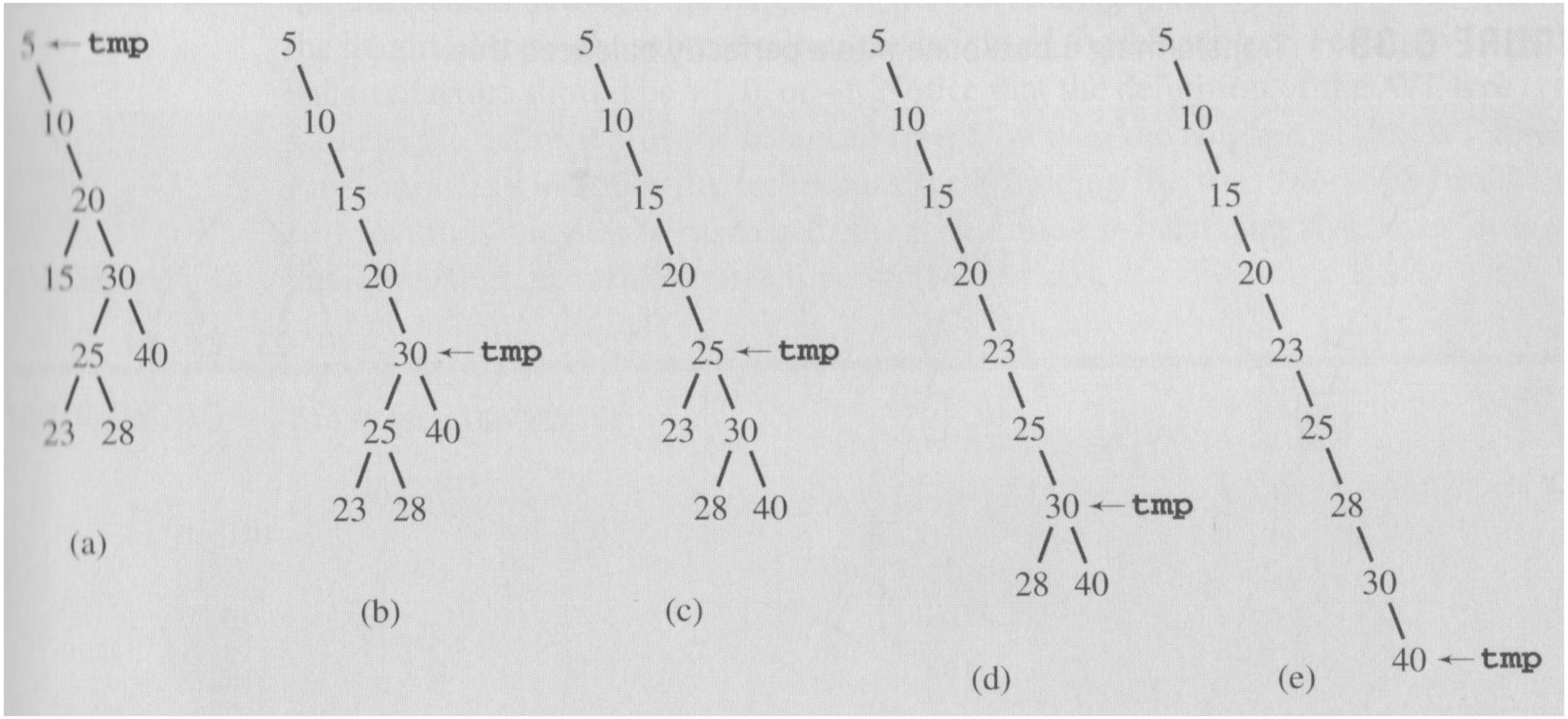
DSW algoritam ima dvije faze:

1) pretvaranje binarnog stabla u “listoliku” strukturu *kralježnica* (*backbone*, *vine*)
2) pretvaranje *kralježnice* u savršeno uravnoteženo stablo

Prva faza: pretvorba binarnog stabla u kralježnicu

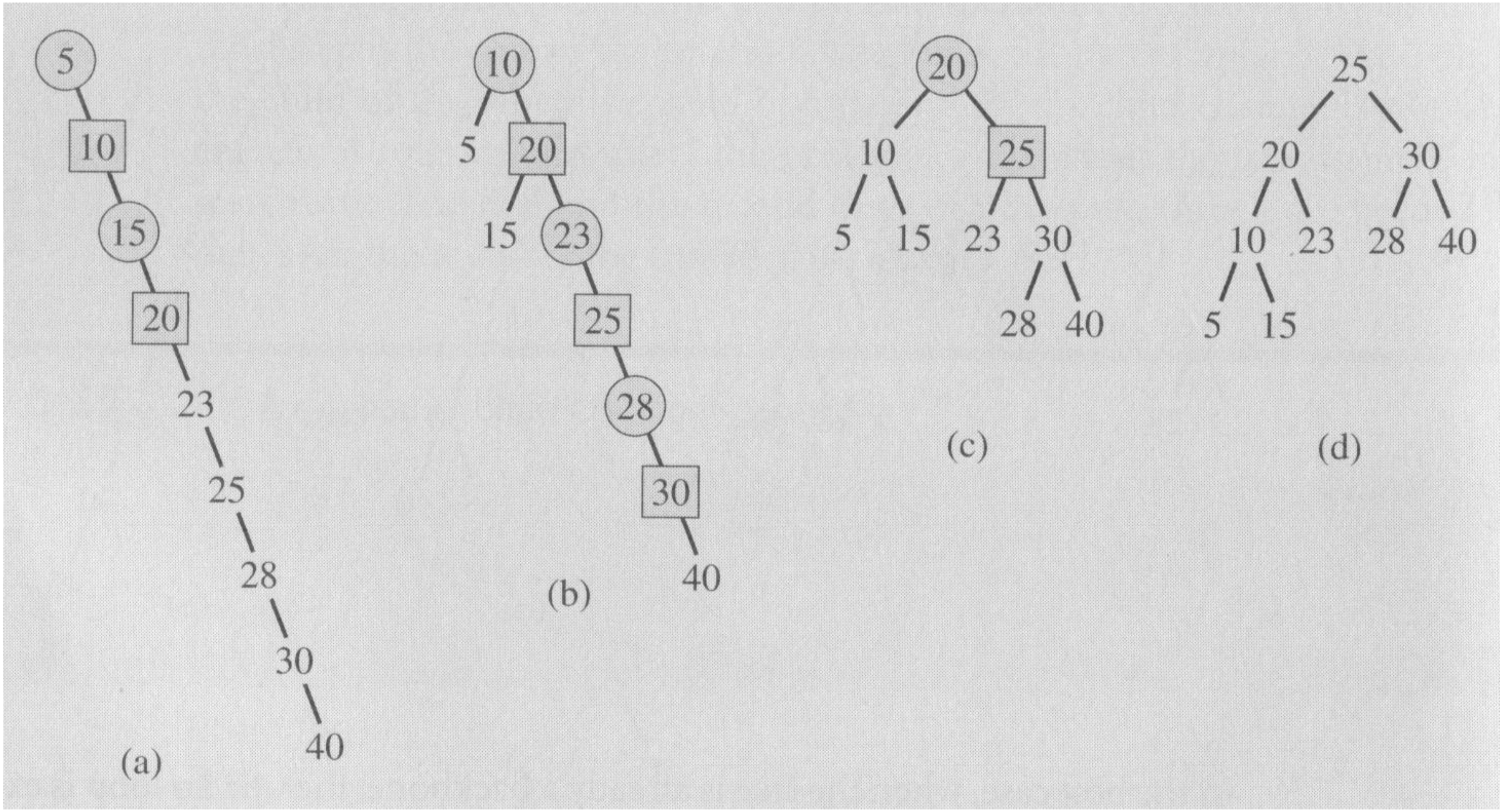


```
CreateBackbone (root)
tmp = root
while tmp != NULL
      if tmp  has left child ch
            right rotate ch about tmp
            redirect tmp to the ch
      else
            redirect tmp to its right child
```

Rotacija iziskuje poznavanje roditelja od `ch` pa realizacija zahtijeva još jedan pokazivač.
Sve se može i simetrično obrnuti, dakle stvarati kralježnicu lijevim rotacijama.

U najgorem slučaju, `while` petlja se obavlja (2n – 1) puta, s (n – 1) rotacijom.
Dakle, složenost je O(n).

Druga faza: pretvorba kralježnice u savršeno stablo



Rotacije na slici: okrugli - roditelji (osi rotacije),
kvadratni - djeca koja moraju postati roditelji.

```
CreatePerfectTree (n)
```

`h =` *largest integer less than or equal to* $\log_2(n+1)$

`m = 2^h - 1` `//broj čvorova u punim razinama budućeg stabla`

make totally `n−m` *left rotations, taking every second node and rotating it about its parent, starting from the top*

```
while m > 1
      m = m/2
```

make totally `m` *left rotations, taking every second node and rotating it about its parent,* (*starting from the top*)

Komentar: n elemenata u potpunom binarnom stablu popunit će h razina, gdje je h najveći cijeli broj manji od $\log_2(n+1)$, i još će njih $n - (2^h - 1)$ biti u najnižoj, nepopunjenog razini. Taj ostatak se rješava rotacijama prije `while` petlje, a potpuno popunjene razine rotacijama unutar petlje.

Složenost ove faze također je O(n) pa je ukupna složenost uravnotežavanja stabla DSW algoritmom O(n).

AVL stabla (izvorno *admissible tree*)

- Georgii Maksimovič Adelson-Velskii, Yevgenii Mikhailovič Landis

Nedostatak DSW algoritma je uravnotežavanje cijelog stabla, iako je ravnoteža najčešće narušena samo lokalno.

AVL algoritam uravnotežuje stablo lokalno, ali ne jamči savršenu uravnoteženost.

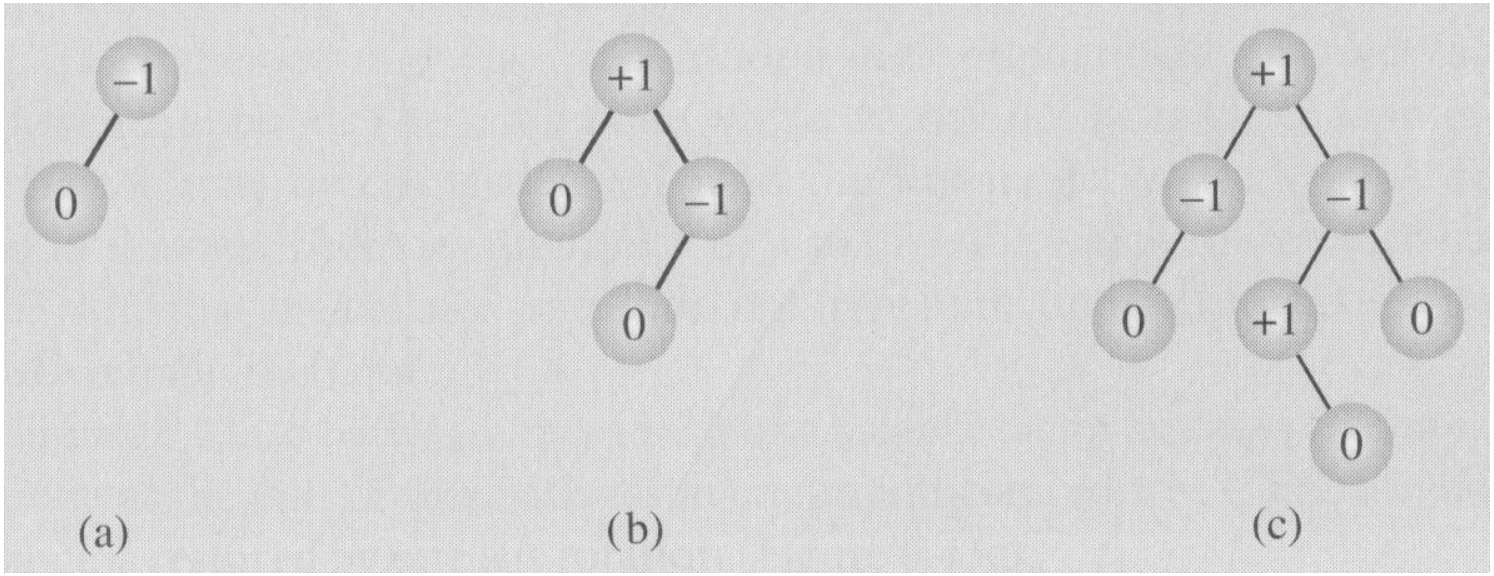
AVL stablo - stablo koje zadovoljava AVL definicijsko pravilo

AVL definicijsko pravilo propisuje faktore ravnoteže (*balance factors*).

faktor ravnoteže FR = razlika visina desnog i lijevog podstabla

(može i obratno, ali ovo je uobičajeno)

AVL definicijsko pravilo: faktori ravnoteže svih čvorova moraju biti –1, 0 ili 1.



Iz definicije AVL stabla slijedi da je najmanji broj čvorova u stablu određen rekurzivnom relacijom $AVL_h = AVL_{h-1} + AVL_{h-2} + 1$, gdje su $AVL_0 = 0$ i $AVL_1 = 1$ polazne vrijednosti.

Teorijski se dokazuju sljedeće granice visine AVL stabla u ovisnosti o broju (čvorova) n (Drozdek: Appendix A.5):

$$\log_2(n+1) \le h \le 1{,}44 \cdot \log_2(n+2) - 0{,}328$$

.

Prema tome, složenost pretraživanja AVL stabla je $O(\log_2 n)$.

U savršeno uravnoteženom stablu iste visine vrijedi $h = \log_2(n+1)$

$\Rightarrow$ pretraživanje AVL stabla u najgorem slučaju je 44 % sporije od pretraživanja savršenog stabla.

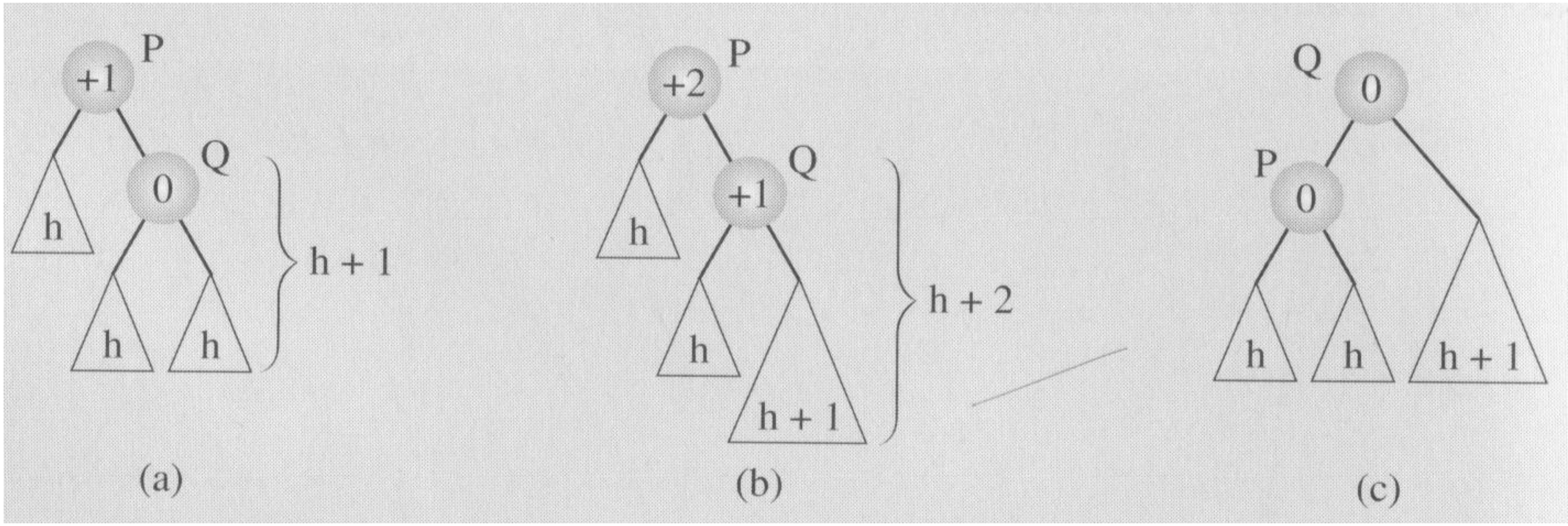
Dodavanje čvora u AVL stablo:

1. naći mjesto i ubaciti čvor
   - pretraživanje AVL stabla isto je kao i pretraživanje običnog binarnog search-stabla
2. uravnotežiti stablo
   - od roditelja dodanog čvora krenuti prema korijenu i osvježavati faktore ravnoteže
   - svakom čvoru na tom putu FR se može povećati ili smanjiti za =1, ovisno o tome kojem se njegovom podstablu promijenila visina
   - prvi i jedini koji će zahtijevati intervenciju bit će onaj kojemu osvježeni FR bude jednak –2 ili 2
   - moguća su četiri slučaja, po dva simetrična (čvor može ispasti iz ravnoteže samo ako je prethodno imao FR = ±1)

Situacije 1 i 2 (simetrična)

- dodavanje u desno (lijevo) podstablo desnog (lijevog) djeteta (P = roditelj, Q = dijete)
- prepoznavanje: 1 $\leftrightarrow$ FR(P) = 2 i FR(Q) = 1

  2 $\leftrightarrow$ FR(P)= –2 i FR(Q) = –1

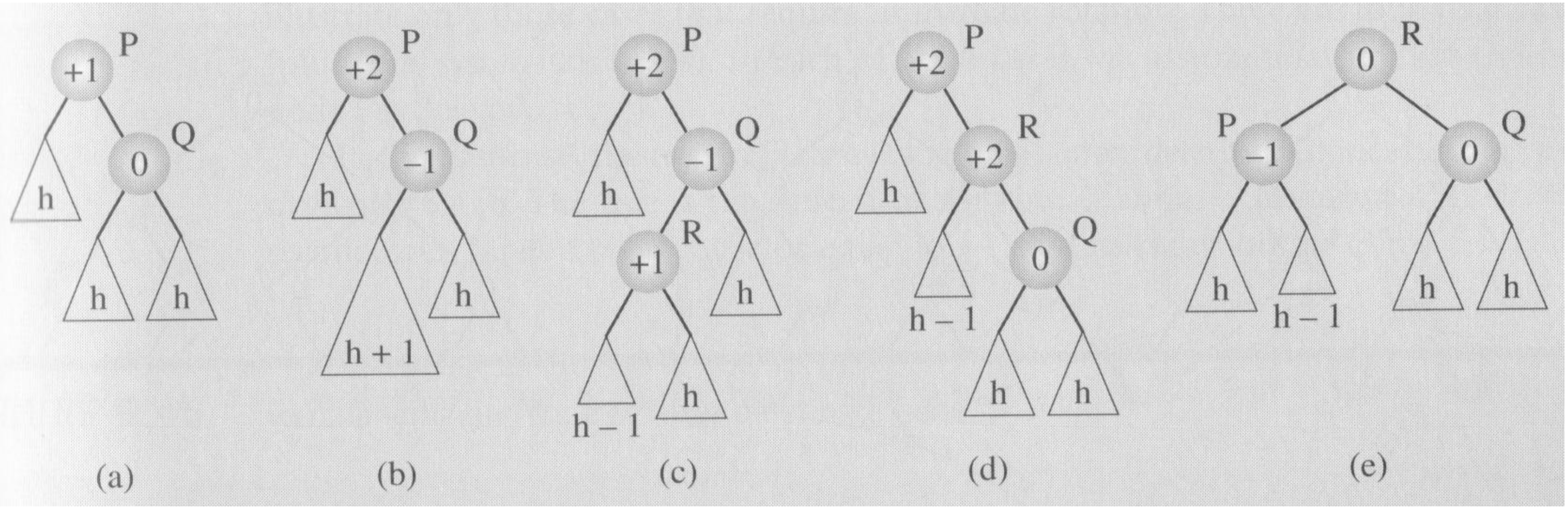


Rješenje: rotacija djeteta Q oko roditelja P.

Situacije 3 i 4 (simetrična)

- dodavanje u lijevo (desno) podstablo desnog (lijevog) djeteta (P = roditelj, Q = dijete)
- prepoznavanje: 3 $\leftrightarrow$ FR(P) = 2 i FR(Q) = –1

  4 $\leftrightarrow$ FR(P)= –2 i FR(Q) = 1



Rješenje: 1) rotacija R oko Q (slika d)

2) rotacija R oko P (slika e)

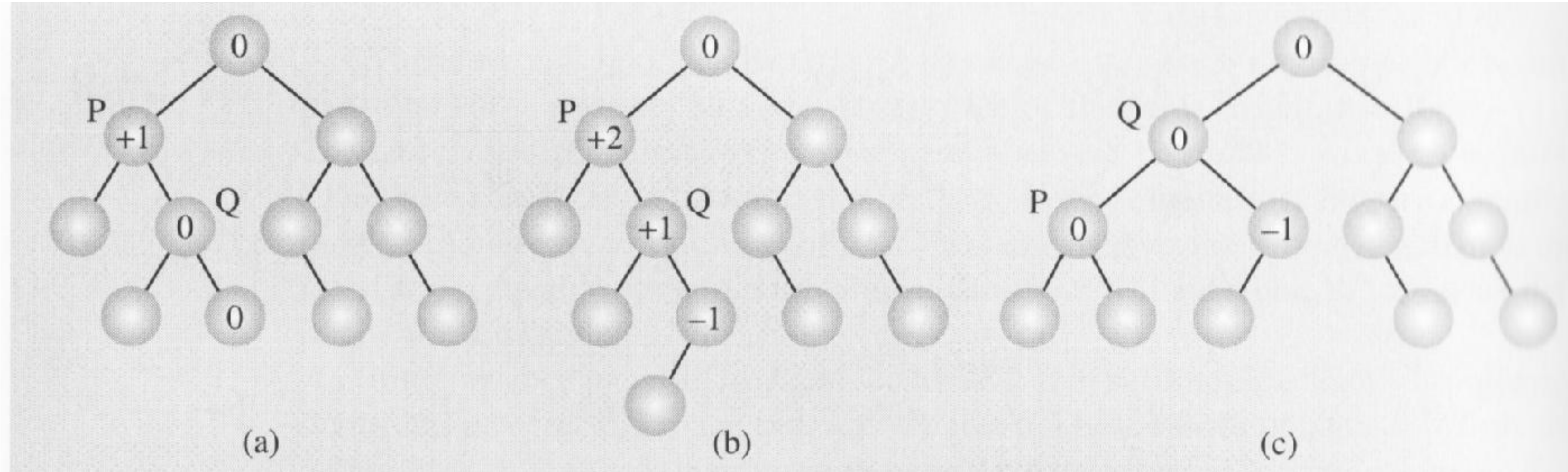
Treba li provjeravati stanje iznad P?

Srećom NE - ukupna visina stabla u sva četiri slučaja ostaje ista (=h+2)!

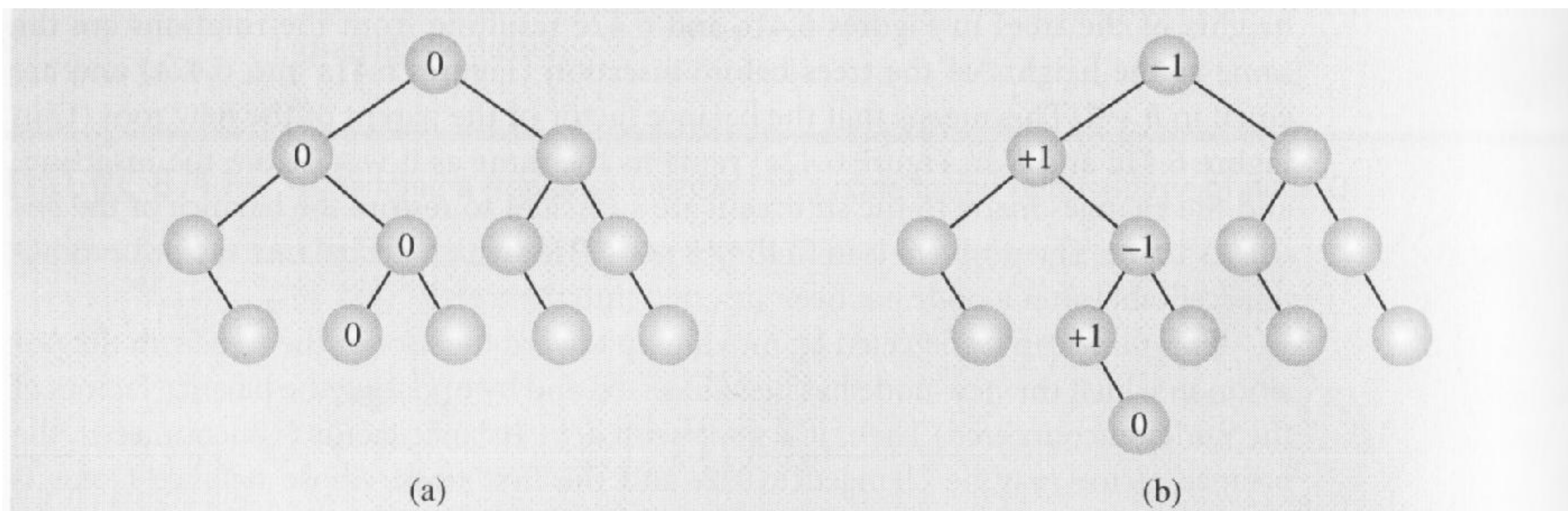
Kako pronaći korijen P podstabla narušene ravnoteže (onaj kojemu je novi FR = ±2)?

- uspinjati se od ubačenog čvora prema korijenu i osvježavati visine podstabala, odnosno faktore ravnoteže, svakog čvora na tom putu
- ako je novi FR(x) = ±1, nastaviti bez intervencije
- ako je novi FR(x) = 0, sve je u redu $\rightarrow$ gotovo
- korijen P mogu (ali i ne moraju) postati samo čvorovi koji su prethodno imali faktor ravnoteže ±1, a novi je ±2, pa prvi takav postaje P

Primjer traženja P:



Ako su svi faktori ravnoteže na putu =0, osvježavanje će biti potrebno sve do korijena, ali bez drugih intervencija.



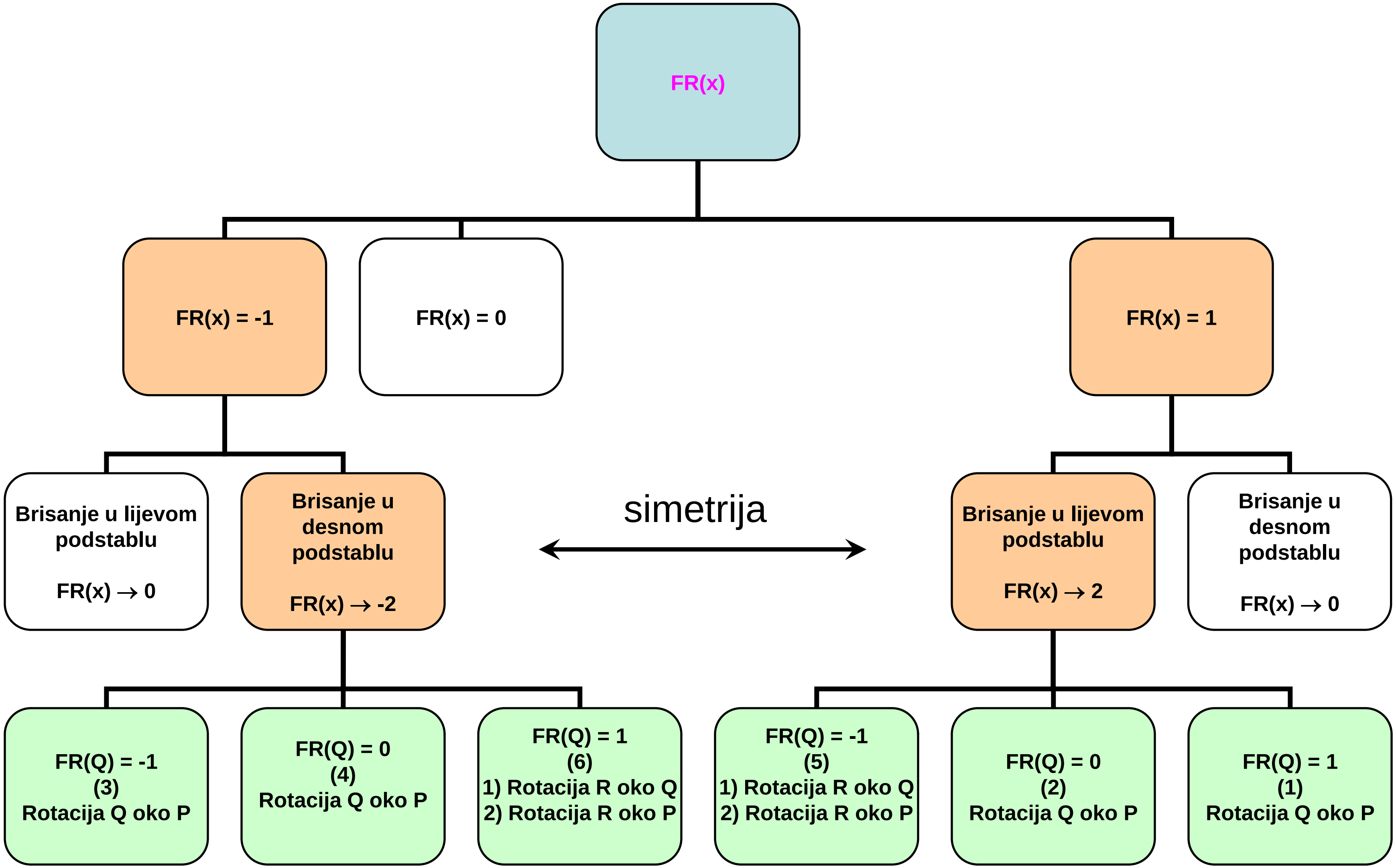
Brisanje čvora u AVL stablu:

1. ukloniti čvor (Deletion by Copying!)
2. uravnotežiti stablo (ne samo do prvog neuravnoteženog!)
   - krenuti od roditelja čvora iz kojeg su prepisani podatci (i koji je stvarno uklonjen iz stabla), ili iz roditelja lista ako je izbrisan list, prema korijenu i osvježavati faktore ravnoteže FR
   - ako je FR(x) postao ±1, znači da brisanje nije promijenilo visinu podstabla kojemu je čvor x korijen (podstabla koja su započinjala djecom od x su bila jednaka, a jedno je sada niže) pa je osvježavanje završeno
   - ako je FR(x) postao =0, brisanje je izazvalo promjenu visine podstabla kojemu je x korijen (podstabla kojima su djeca od x korijeni bila su različite visine, a sada su jednaka) pa osvježavanje treba nastaviti, ali bez ikakve intervencije
   - ako je FR(x) postao ±2, potrebna je intervencija (tri slučaja + tri simetrična)
   - nakon intervencije, opet treba provjeriti FR(x) i ako je sada jednak ±1, osvježavanje je završeno;
     za novi FR =0, nastaviti s osvježavanjem

Situacije: promatramo čvor x

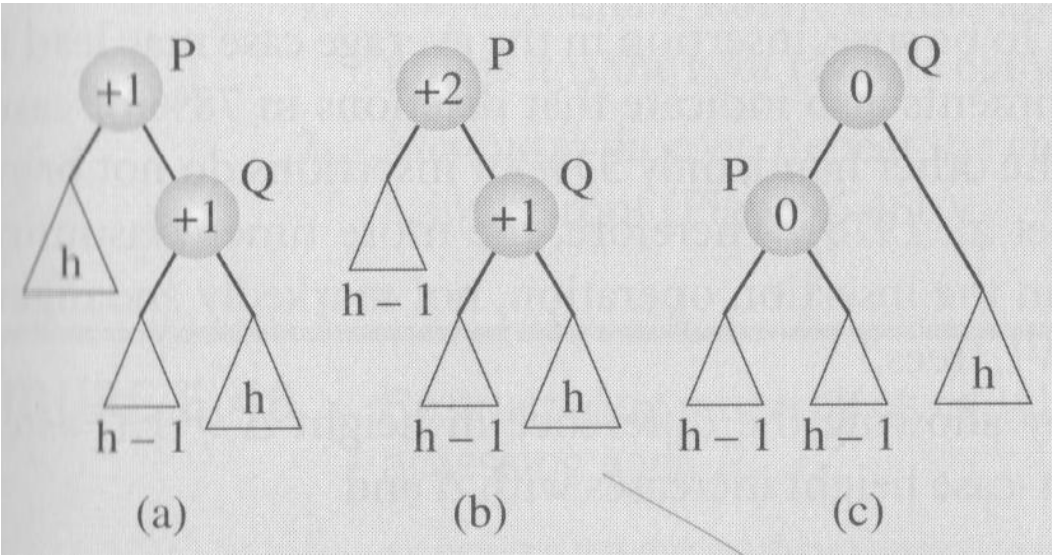
- ako je novi FR(x) = ±2, prije je morao biti FR(x) = ±1; da je bio =0, brisanje ga ne bi izbacilo iz ravnoteže
- brisanje se moglo dogoditi ili u lijevom ili u desnom podstablu od x
- brisanje u lijevom podstablu kada je FR(x) = –1, kao i brisanje u desnom podstablu kada je FR(x) = 1, prouzročit će FR(x) = 0 i ta dva slučaja neće zahtijevati intervenciju
- brisanje u desnom podstablu kada je FR(x) = –1 i brisanje u lijevom podstablu kada je FR(x) = 1 prouzročit će FR(x) = ±2 i zahtijevaju intervenciju; u oba slučaja postupak ovisi o djetetu u podstablu koje se nije mijenjalo, a budući da to dijete može imati FR –1, 0 ili 1, ukupno dobivamo šest različitih situacija

Stablo odlučivanja za brisanje čvora u AVL stablu



Slučajevi 1 i 2 (3 i 4 simetrični):

- brisanje u lijevom podstablu kada je FR(x) = 1 ili u desnom podstablu kada je FR(x) = –1



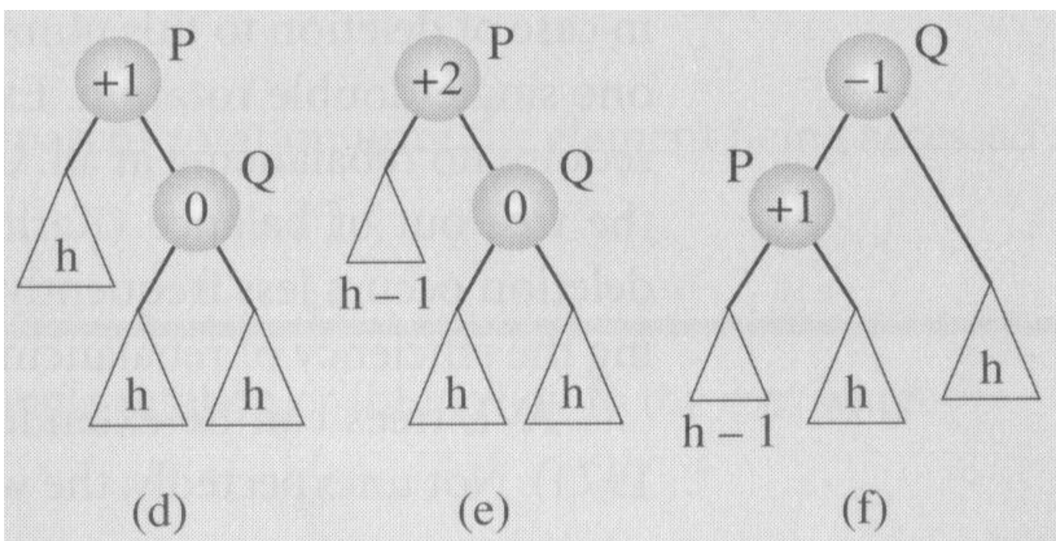
Prepoznavanje:

1 ↔ FR(P)=2 i FR(Q)=1

2 ↔ FR(P)=2 i FR(Q)=0

3 ↔ FR(P)=–2 i FR(Q)=–1

4 ↔ FR(P)=–2 i FR(Q)=0



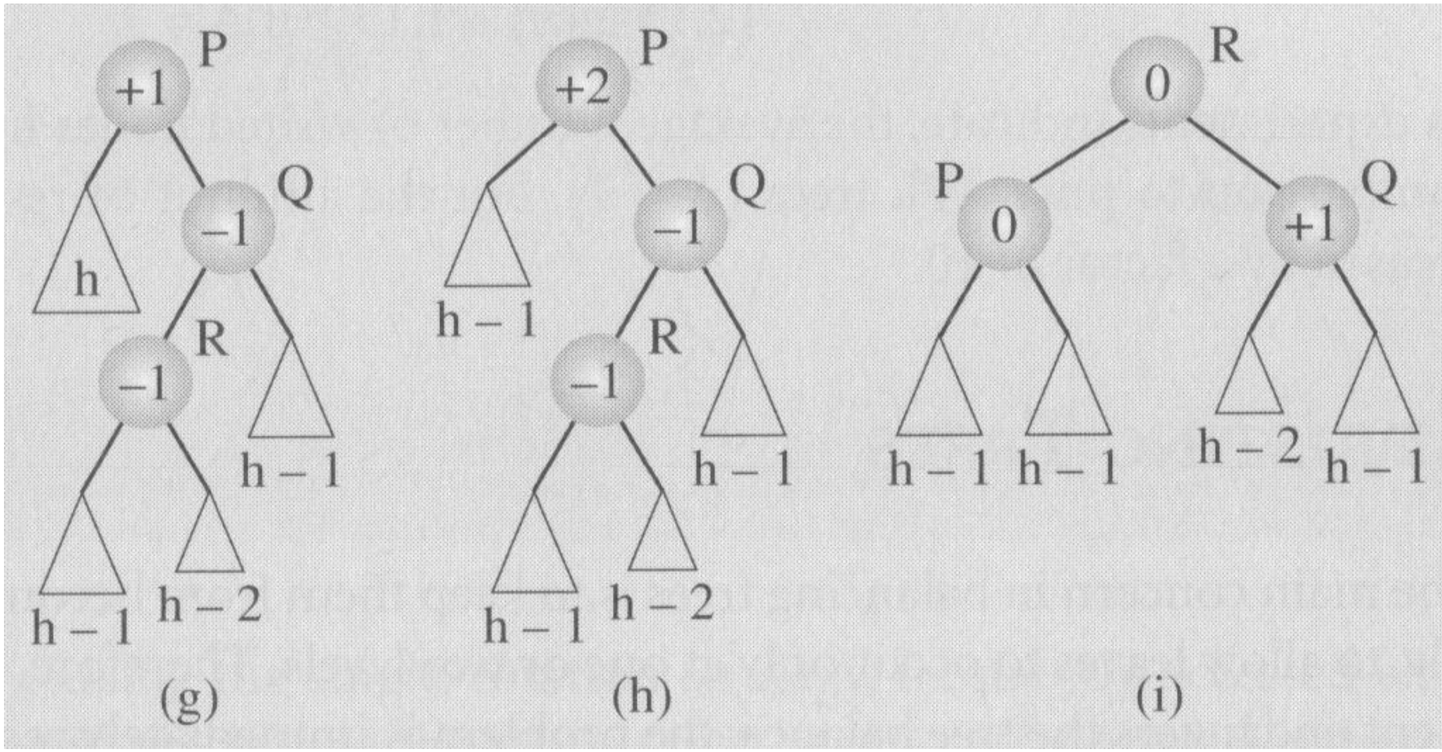
Rješenje:

rotacija Q oko P

- isto kao slučajevi 1 i 2 za dodavanje

Slučaj 5 (6 simetričan):

- brisanje u lijevom podstablu kada je FR(x) = 1 ili u desnom podstablu kada je FR(x) = –1



Prepoznavanje:

5 $\leftrightarrow$ FR(P)=2 i FR(Q)=–1

6 $\leftrightarrow$ FR(P)=–2 i FR(Q)=1

Rješenje je neovisno o FR(R):

1. rotacija R oko Q
2. rotacija R oko P

- isto kao slučajevi 3 i 4 za dodavanje

Još jedna vrsta uravnoteženih stabala su tzv. crveno-crna (red-black) stabla koja su zapravo varijanta B-stabala, ali niskog (4.) reda.

Samopodešavajuća stabla (*Self-Adjusting Trees*)

Primarna najmena binarnih search stabala je brz pristup podatcima. DSW algoritam i AVL stabla to omogućavaju, ali cijena je relativno komplicirano održavanje (sporije dodavanje i brisanje). Samopodešavajuća stabla su alternativa strogim metodama, a slijede ideju samopodešavajućih lista - podatke kojima se češće pristupa podići na više razine. Zato svaki čvor mora "znati" koliko mu se puta pristupilo.

Dvije osnovne strategije:

1. Single rotation – čvor kojem se pristupilo rotirati oko roditelja

1. Moving to the Root - čvor kojem se pristupilo postaje korijen stabla
   - varijanta ove strategije je i tzv. *Splaying*; dobra kad se nekim elementima pristupa puno češće nego ostalima (vidi Drozdek...)