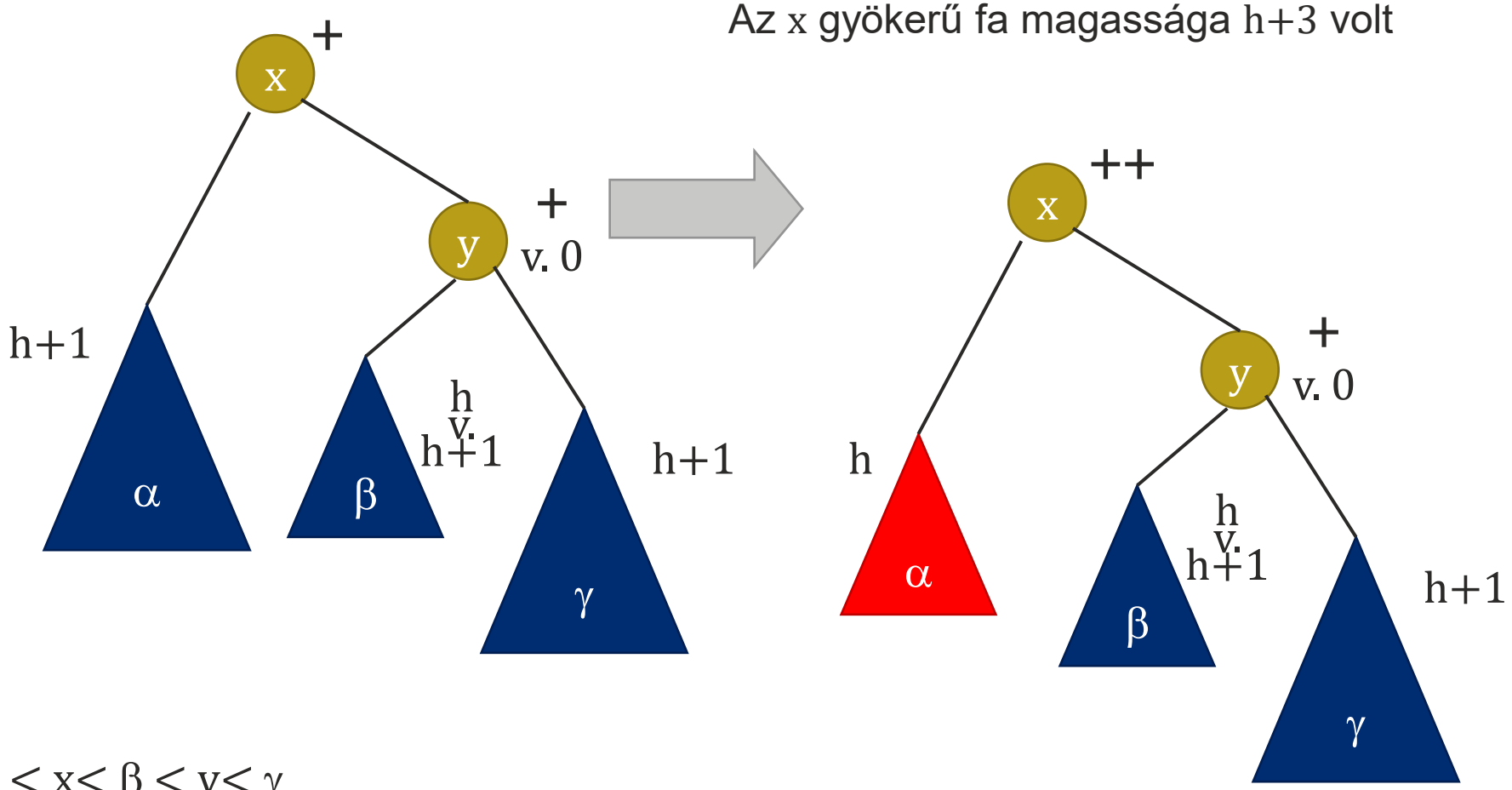
ADATSZERKEZETEK ÉS ALGORITMUSOK

AVL fa törlés
„Hierarchikus adatszerkezetek, keresési fák”

AVL fák – újrakiegyensúlyozás törlésnél

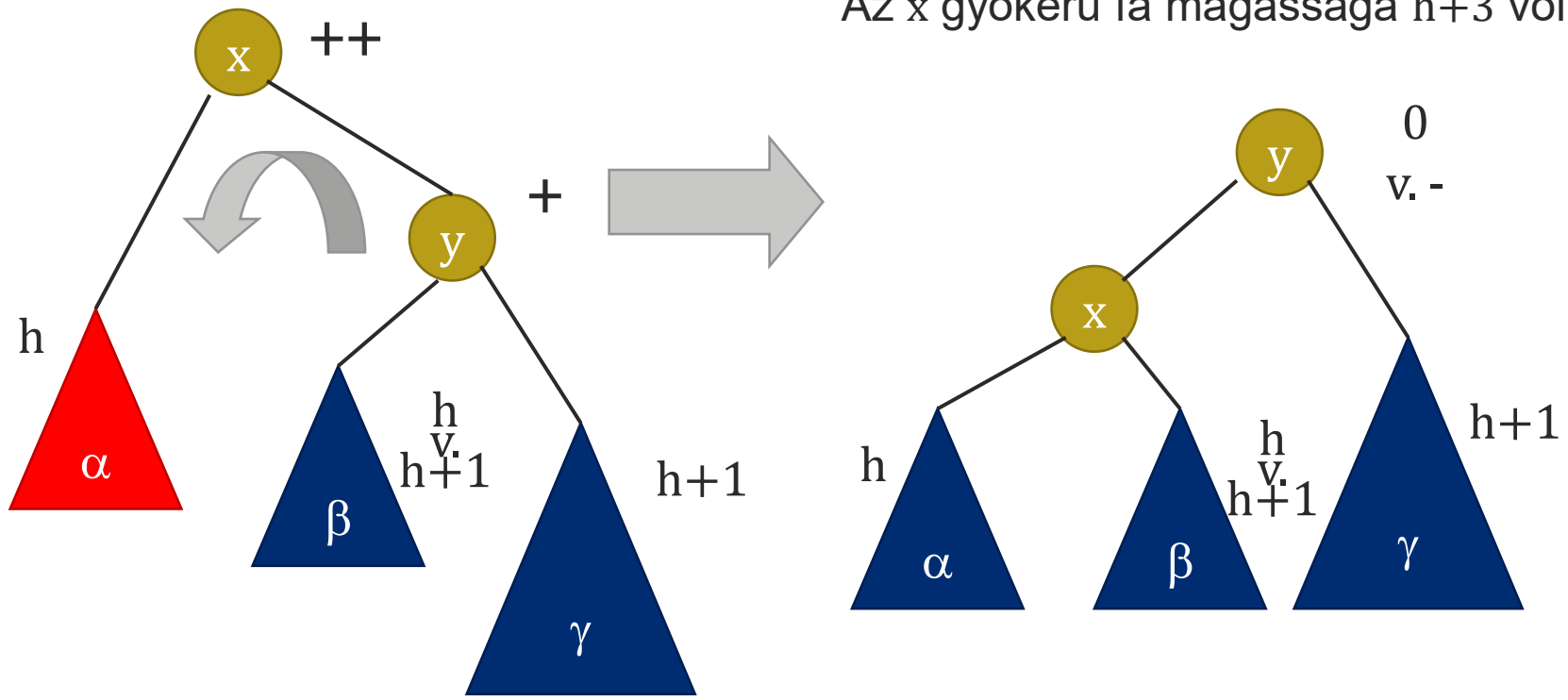
A törlés az α részfában történt.
Ennek a magassága $h+1$ volt és h lett.
Az x gyökerű fa magassága $h+3$ volt



$\alpha < x < \beta < y < \gamma$

A (++,+) (++,0) szabályok

A törlés az α részfában történt.
Ennek a magassága $h+1$ volt és h lett.
Az x gyökerű fa magassága $h+3$ volt. Forgatás:

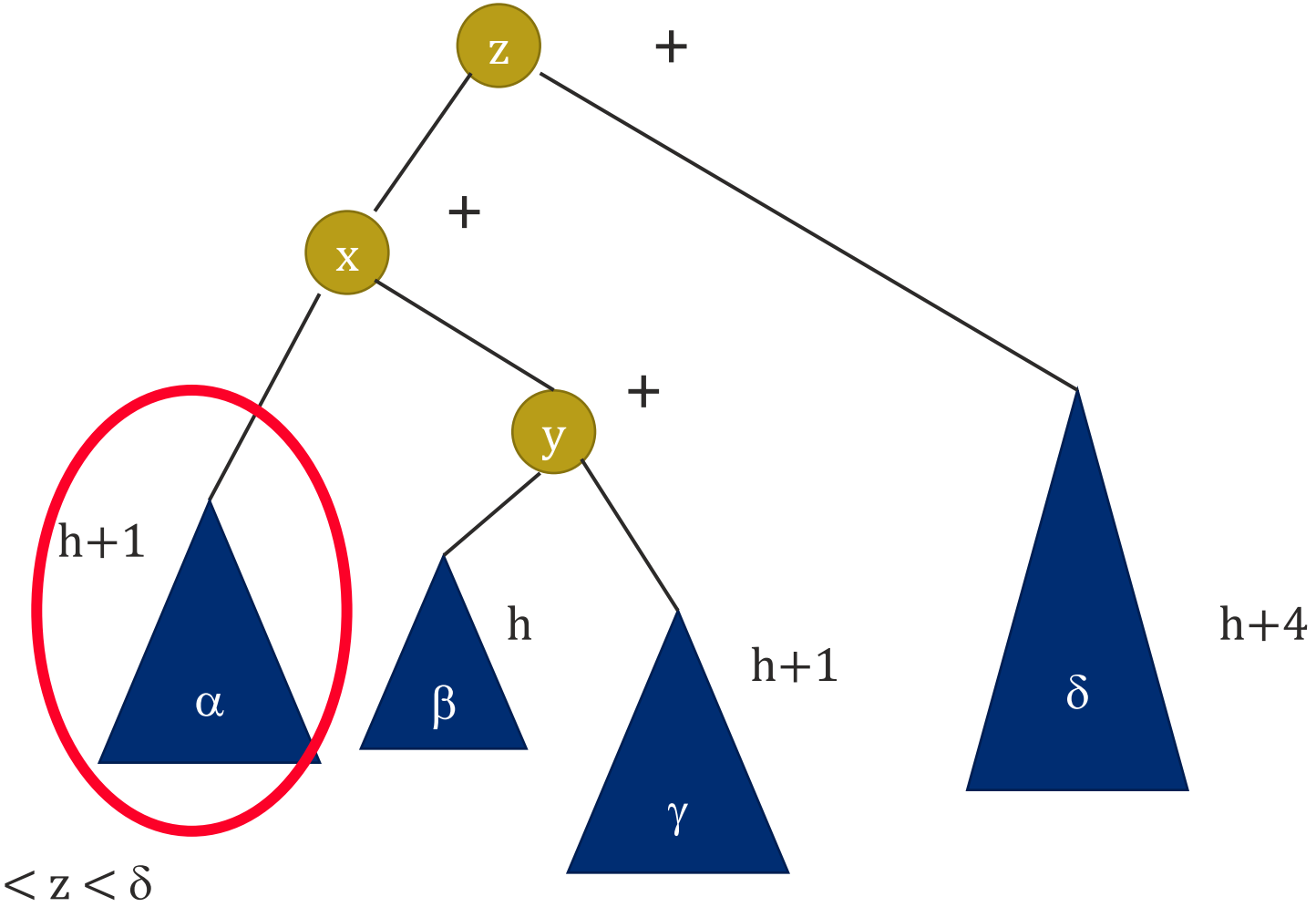


$\alpha < x < \beta < y < \gamma$

A forgatás után h+2 a magasság. Ezért feljebb, a befoglaló fában (ha van), nem biztos, hogy változatlanul érvényes az AVL tulajdonság, **feljebb kell menni** ellenőrizni, amíg a gyökérig nem jutunk.

A (++,+) szabály

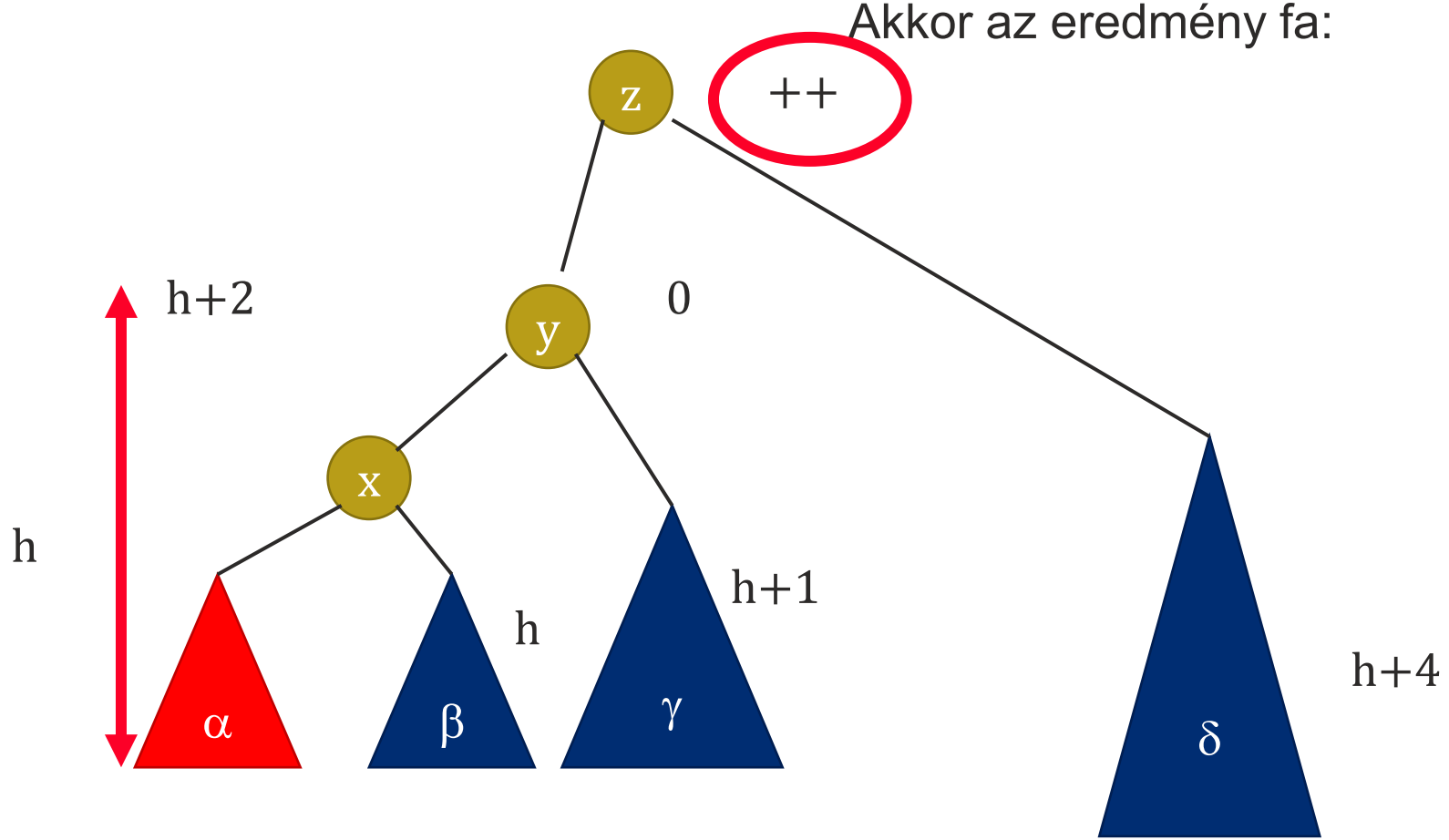
Ha az eredeti fában például:



$\alpha < x < \beta < y < \gamma < z < \delta$

A (++,+) szabály

Akkor az eredmény fa:

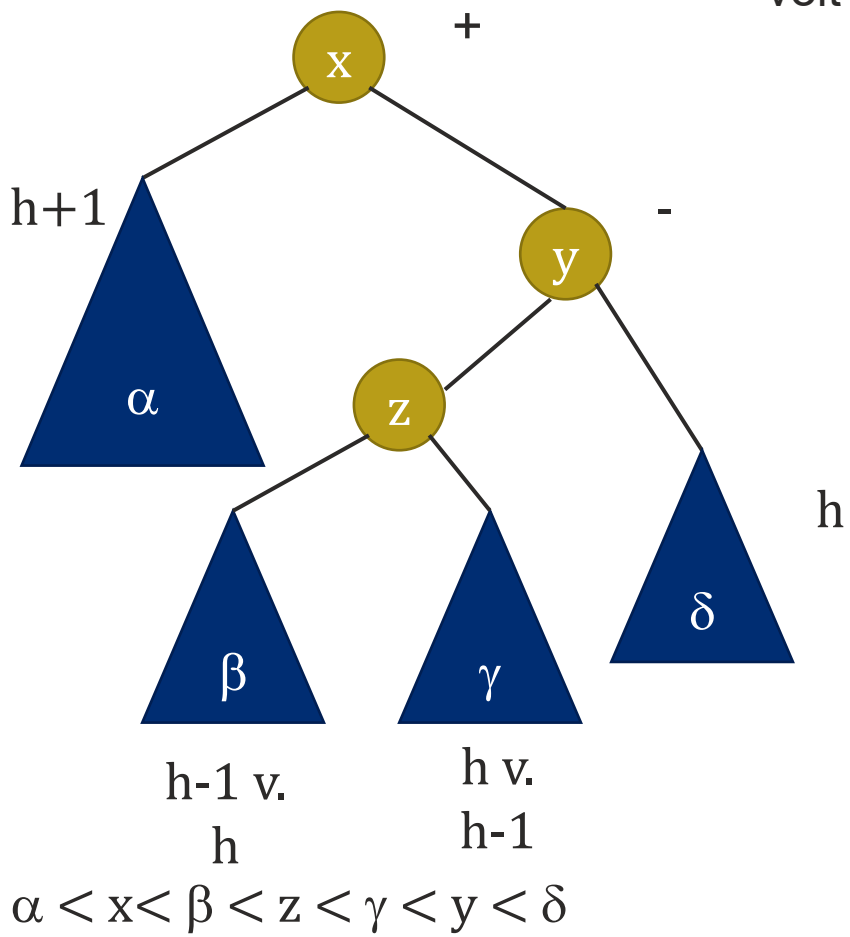


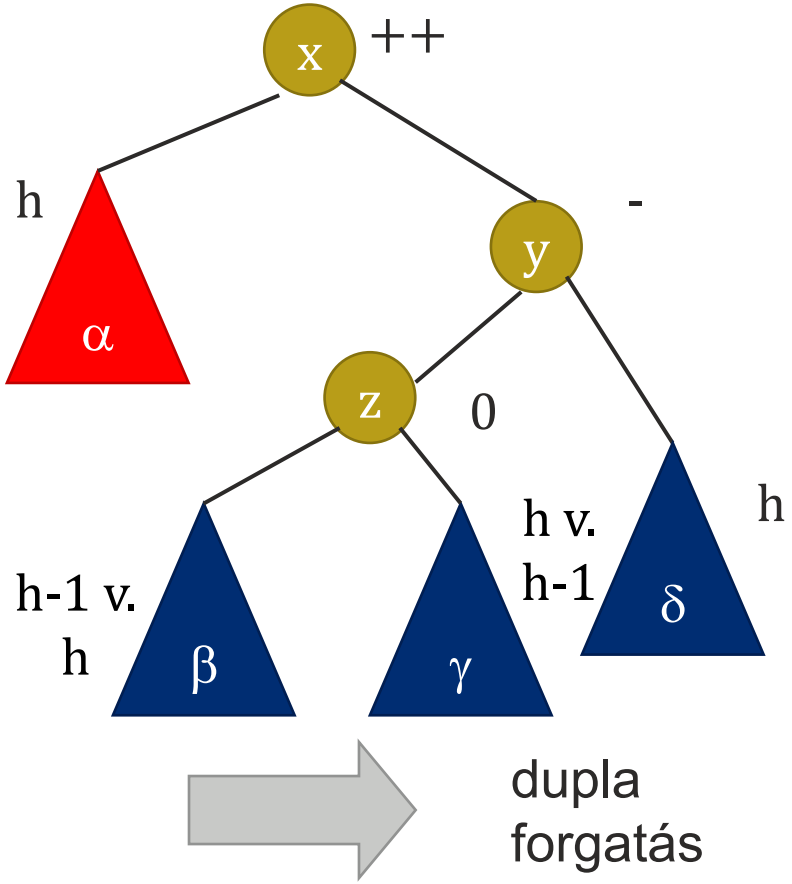
$\alpha < x < \beta < y < \gamma < z < \delta$

feljebb kell menni ellenőrizni, és szükség szerint helyreállítani, amíg a gyökérig nem jutunk.

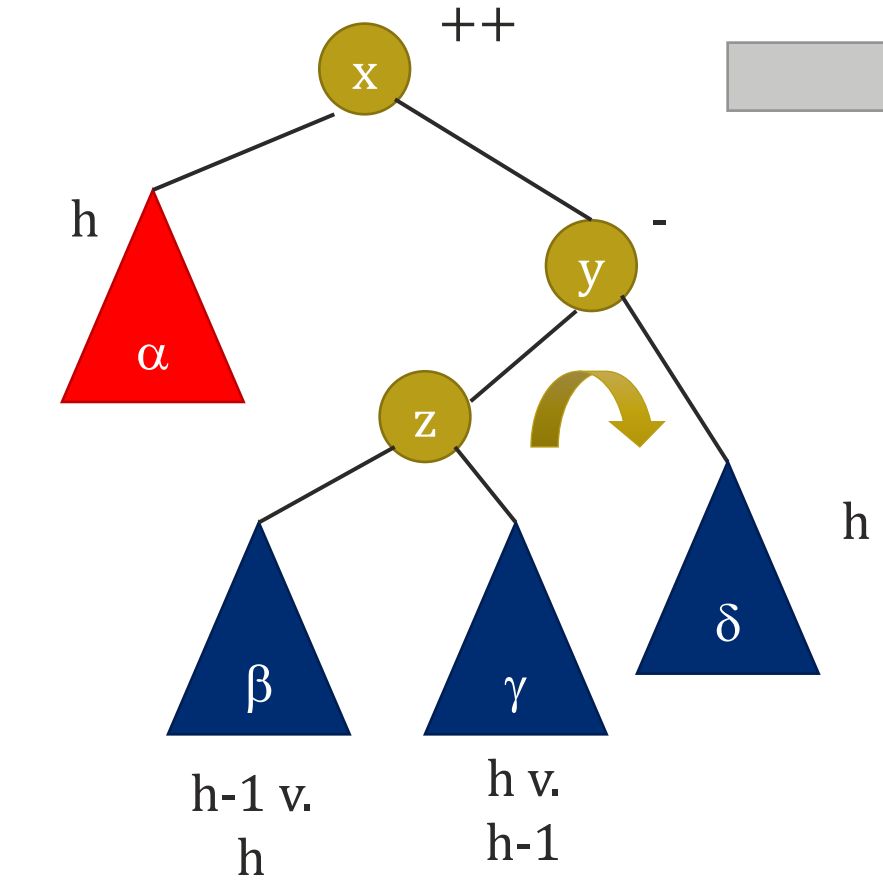
A (++,-) szabály

A törlés az α részfában történik. Ennek a magassága $h+1$ volt és h lett. Az x gyökerű fa magassága $h+3$.



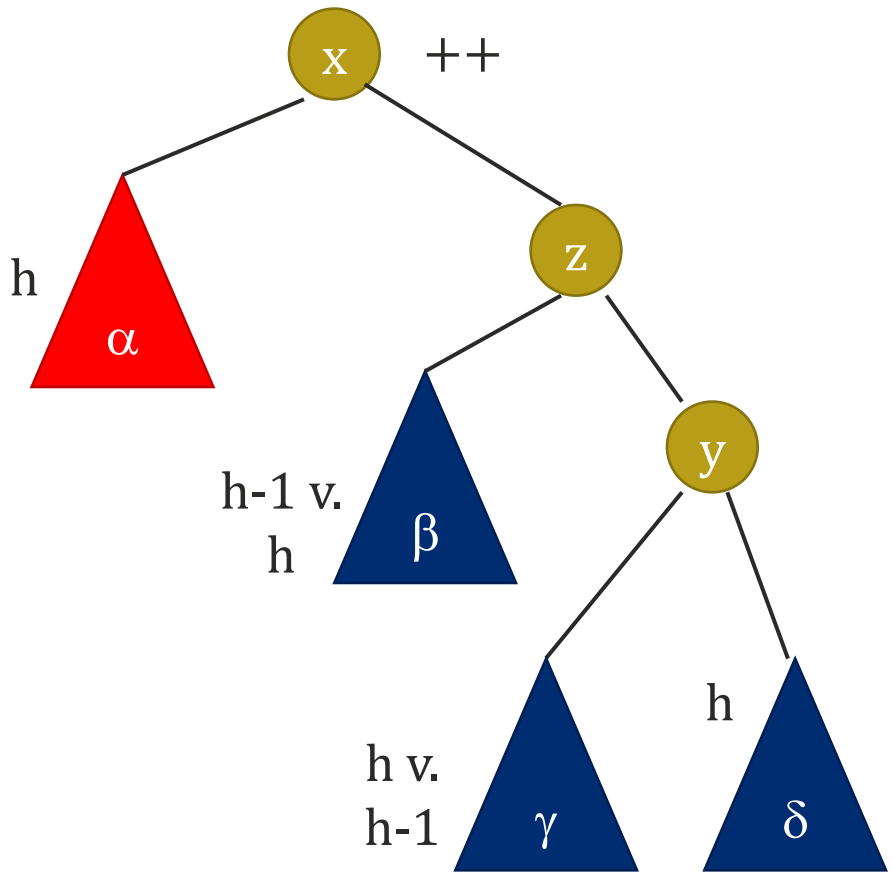


A (++,-) szabály



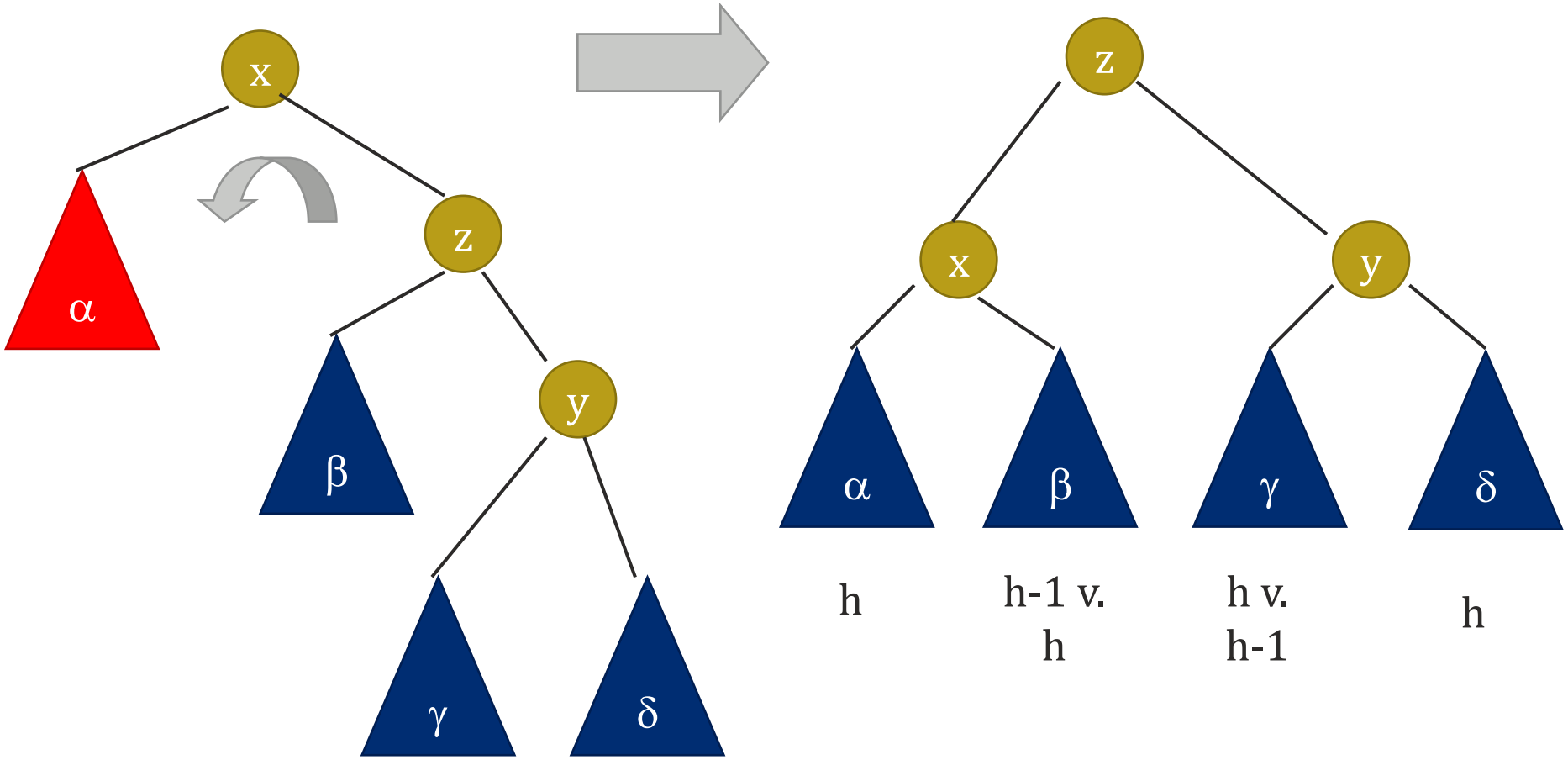
$\alpha < x < \beta < z < \gamma < y < \delta$

Dupla forgatás kell: először jobbra:



A (++,-) szabály

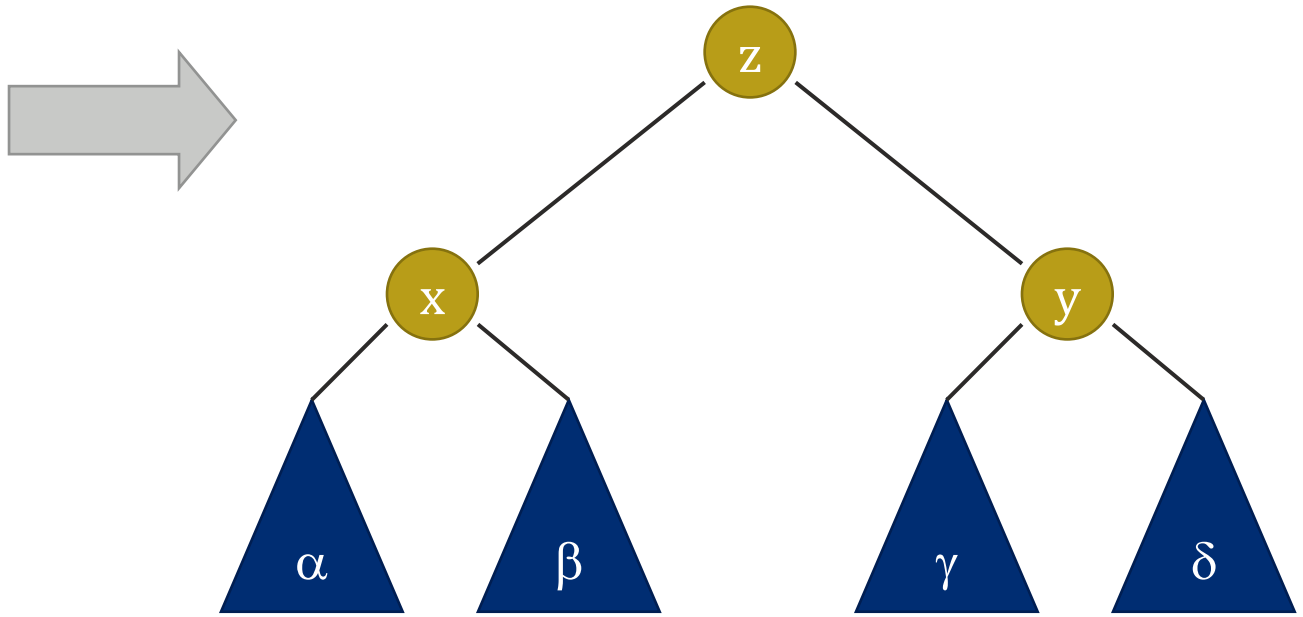
Dupla forgatás kell: azután balra:



A (++,-) szabály

Végeredmény

A forgatás után $h+2$ a magasság.
Ezért feljebb, a befoglaló fában (ha van), nem biztos, hogy változatlanul érvényes az AVL tulajdonság, **feljebb kell menni** ellenőrizni, amíg a gyökérig nem jutunk.



Továbbra is igaz:

$\alpha < x < \beta < z < \gamma < y < \delta$

Újrakiegyensúlyozás törlésnél

- Összefoglalva:
  - Mivel az x gyökerű fa magassága csökkent a forgatással, ezért feljebb is, ha van befoglaló fa, elromolhatott az AVL tulajdonság
  - A törlés után a törölt elem szülőjétől kezdve felfelé haladva a gyökér felé újra számoljuk a csúcsok címkéit ezen az útvonalon
  - Ha egy x csúcs címkéje ++ vagy – – lesz, akkor az x gyökerű (rész)fa (esetleg dupla) forgatásával helyreállítjuk annak AVL tulajdonságát
  - Ha x nem a gyökér, akkor feljebb kell lépni és folytatni kell az ellenőrzést
  - Szélsőséges esetben az adott útvonal minden pontjában forgatni kell

Újrakiegyensúlyozás törlésnél

- Tétel
  - Az n pontú AVL-fából való törlés után legfeljebb $1{,}44\log_2 n$ (sima vagy dupla) forgatás helyreállítja az AVL-tulajdonságot.
- Bizonyítás
  - az előzőekből következik.

Törlés vs. beszúrás

- Törlési esetek eltérnek a beszúrástól a következőkben:
  - Lehetséges a $(--,0)$ illetve $(++,0)$ **kiinduló** állapot is
  - A fa gyökeréig fel **kell** menni az ellenőrzés során

Összefoglalás

- AVL fák
  - Az első dinamikusan kiegyensúlyozott fák
  - A magasság az optimális 44%-án belül
  - Újrakiegyensúlyozás forgatásokkal
  - $\mathcal{O}(\log n)$

Megvalósítás

És összefoglalás

AVL fa

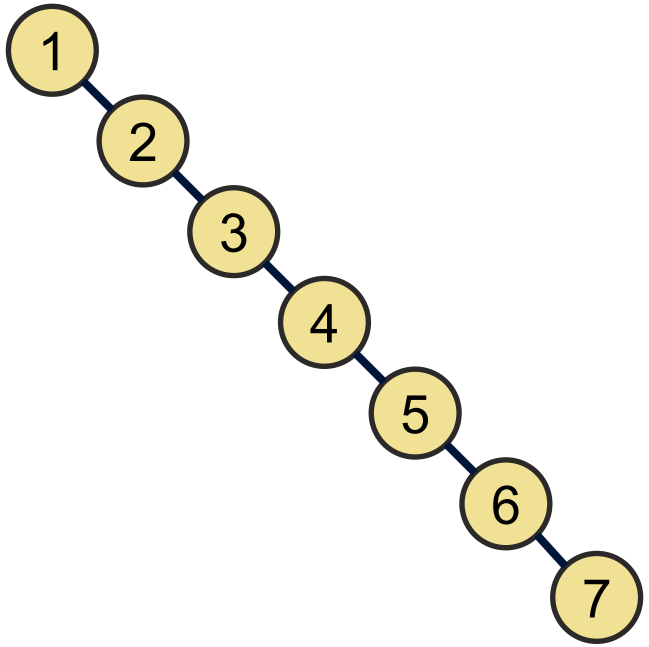
- Bináris keresőfa.
- A bal és jobb részfák magassága legfeljebb 1-gyel tér el egymástól.
- Az AVL fa minden részfája is AVL fa.

Motiváció

Szúrjuk be a következő elemeket egy fába az alábbi sorrendben:

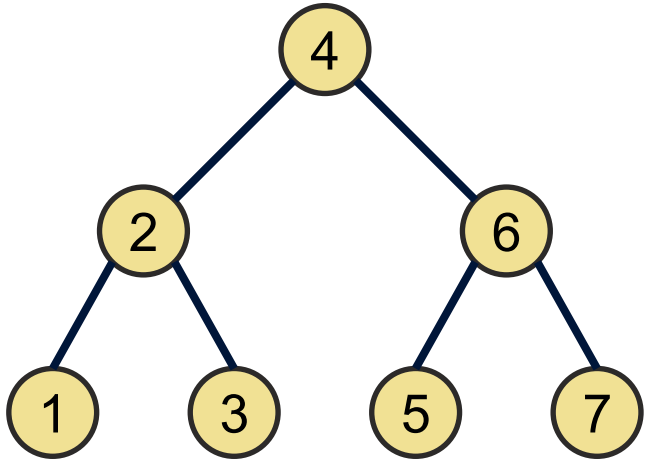
1, 2, 3, 4, 5, 6, 7

Bináris keresőfa:



A fa magassága **n**,
mintha csak egy lista lenne

AVL fa:



Garantált **1,44*log$_2$n** magasság

Bináris keresőfák teljesítménye

Beszúrt random elemek száma	Kiegyensúlyozatlan fa magassága *	AVL fa magassága *
10	6	4
100	12	8
1000	23	12
10000	37	16
100000	159	19
* egy konrét teszt eredménye, csak a nagyságrendek szemléltetésére		

Forgatások

Bináris keresőfákban a forgatások megváltoztatják a fa alakját, a sorrendiség megőrzése mellett.

A forgatások során egy csúcspont eggyel lejjebb, és annak egyik gyereke pedig eggyel feljebb kerül a fában.

A forgatás irányát mi úgy definiáljuk, hogy merre mozdulnak el a csomópontok.

AVL fa – részfák magassága

- Ahhoz, hogy a bináris keresőfa kiegyensúlyozottságát „olcsón” fenn tudjuk tartani, minden csúcsot kibővítünk egy magasság mezővel, melyben az adott részfa magasságát tároljuk.
- Definíció szerint az üres fa magassága nulla.

```
class Node {
public:
        //...
   int height;
   void update_height();
};
```

- A magasságot a forgatások után nekünk kell frissíteni az update_height() segédfüggvénnyel.

AVL fa – egyensúly információ

- A fa kiegyensúlyozásához viszont nem kell a részfák magassága, elég, ha tudjuk a magasságok különbségét.
- Ezért a csúcs struktúránkat kiegészítjük egy egyensúly tagfüggvénnyel, amivel ezt a különbséget lehet lekérdezni.

```
class Node {
public:
        //...
  int balance_factor() const;
};
```

- A `balance_factor()`-t úgy definiáljuk, hogy

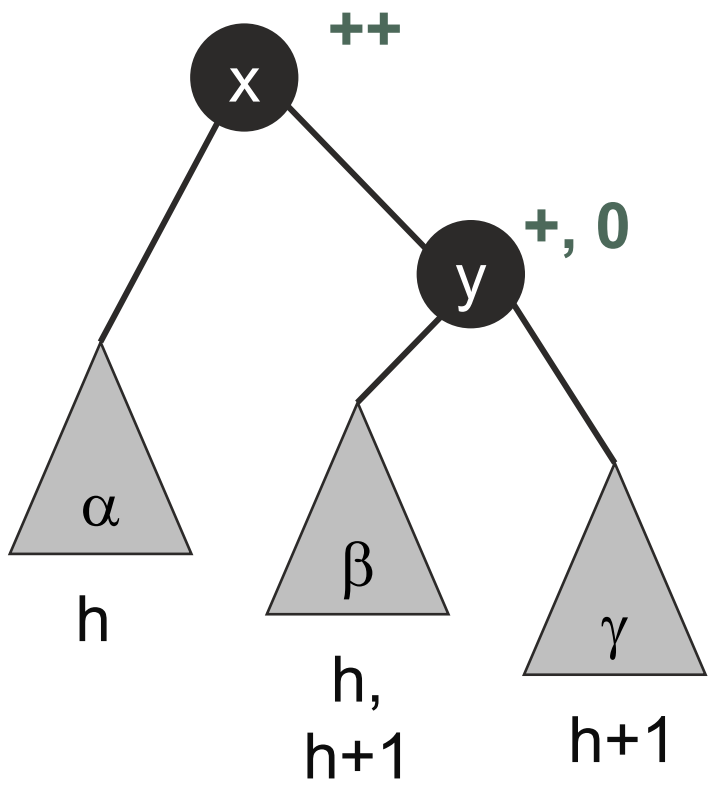
  (jobb magasság – bal magasság)

  ezért egyértelműen megfeleltethető a diákban használt ++, +, 0, -, -- jelöléssel.

AVL fa – kiegyensúlyozás

- Minden beszúrás és törlés művelet után meghívunk egy kiegyensúlyoz (`_rebalance()`) függvényt arra a pontra, ahol a módosítás történt.
- Feladata:
  - Ha sérül az AVL fa tulajdonság, forgatásokkal helyreállítani. (Ezek esetei a későbbi diákon.)
  - Frissíteni a csúcsokban a magasságinformációt.
- Szükséges forgatások száma a legrosszabb esetben:
  - Beszúrásnál konstans
  - Törlésnél a fa magasságával arányos

Kiegyensúlyozás (++, +) és (++, 0)



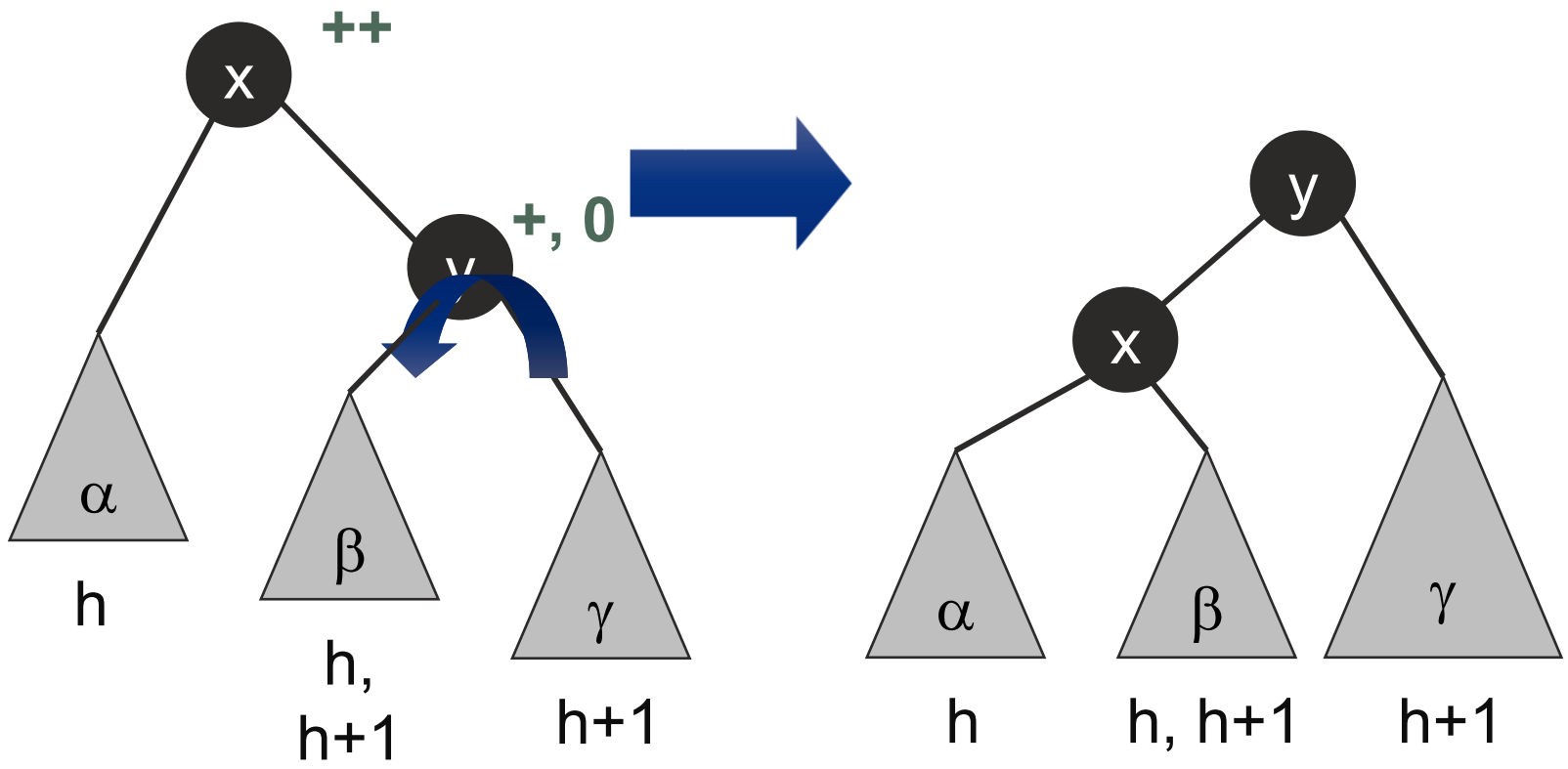
x bal oldali részfája ***h***, jobb oldali pedig ***h+2*** magas, és ez sérti az AVL fa tulajdonságot.

x körüli balra forgatással helyreáll az AVL tulajdonság

Ennek tükörképe a (--,-) és (--,0) esetek.

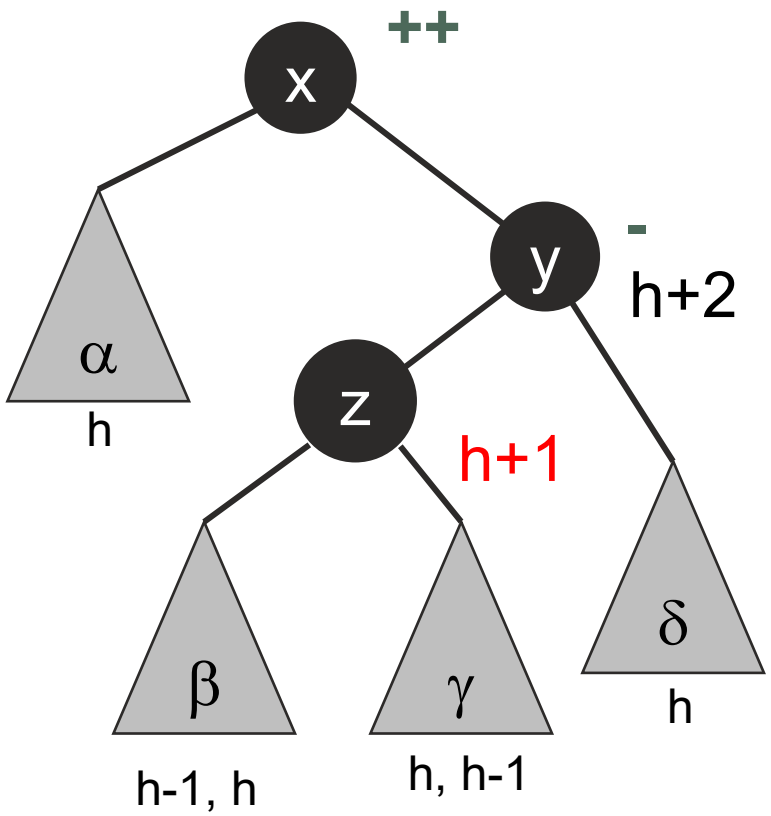
... a forgatás hatása

Ha a részfa magassága változott, úgy **y** szülőjénél folytatjuk a kiegyensúlyozást.



Ne felejtsük el forgatás után a csúcsok magasság mezőit frissíteni!

Kiegyensúlyozás (++, -)



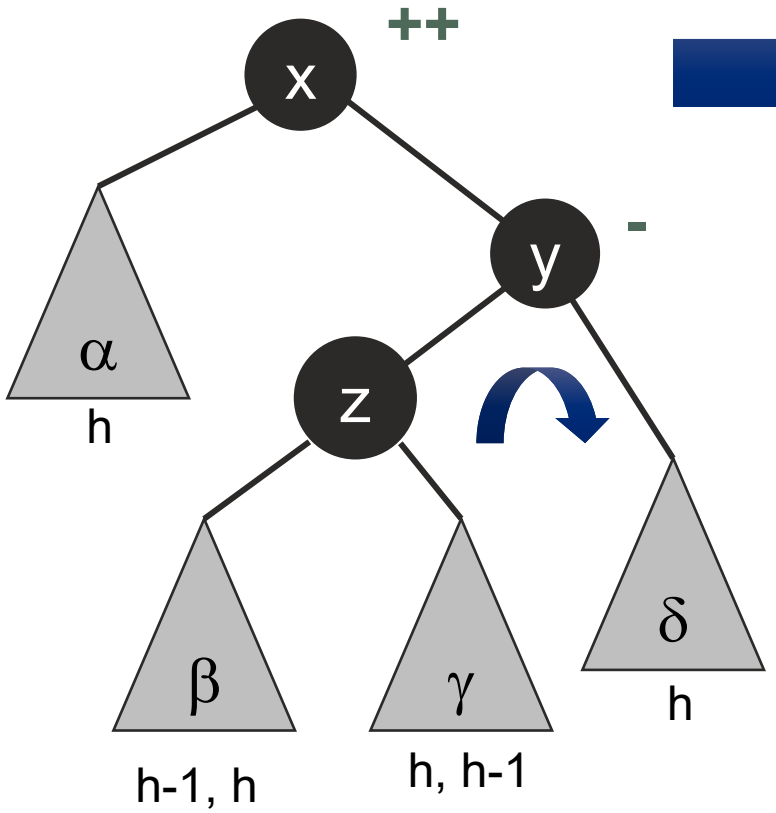
x bal oldali részfája ***h***, jobb oldali pedig ***h+2*** magas, és ez sérti az AVL fa tulajdonságot.

Most az **x** körüli balra forgatás nem oldja meg a problémát, mert a forgatás során az **y**-ról **x**-re átszálló **z** gyökerű fa a „túlsúlyos”.

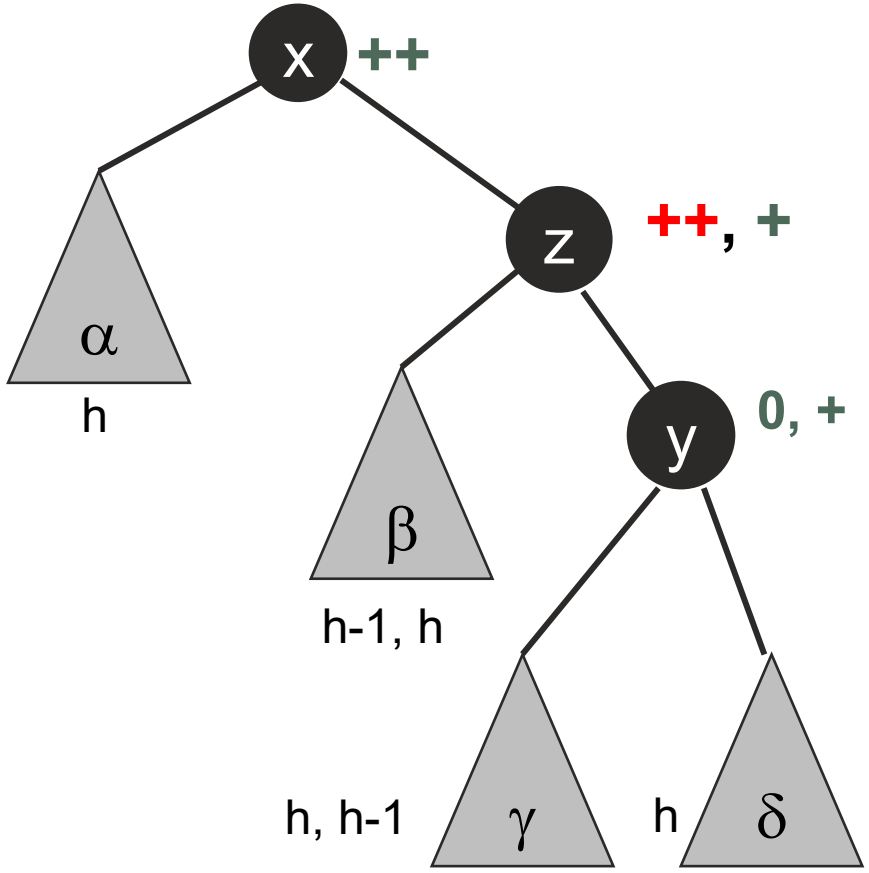
Ezért először egy **y körüli jobbra forgatás**sal szétszedjük **z** részfáit.

Ennek tükörképe a (--, +) eset.

Az első forgatás…

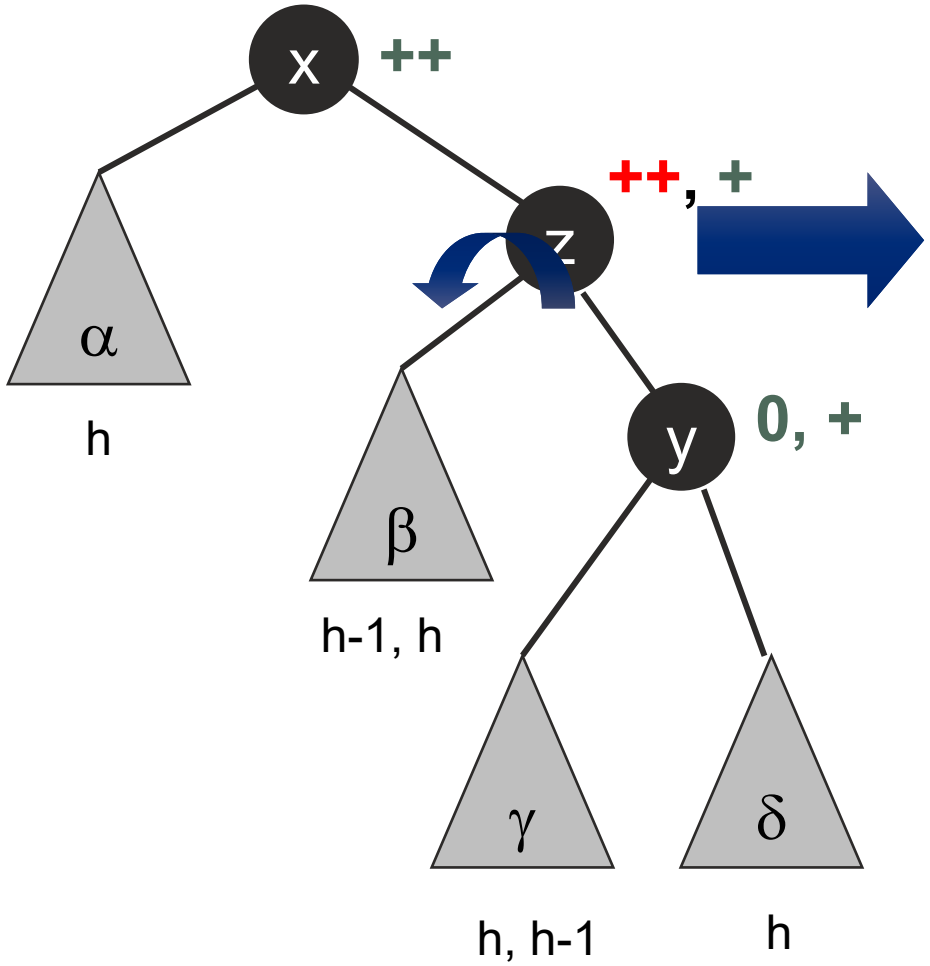


Dupla forgatás: először **y körül jobbra**…

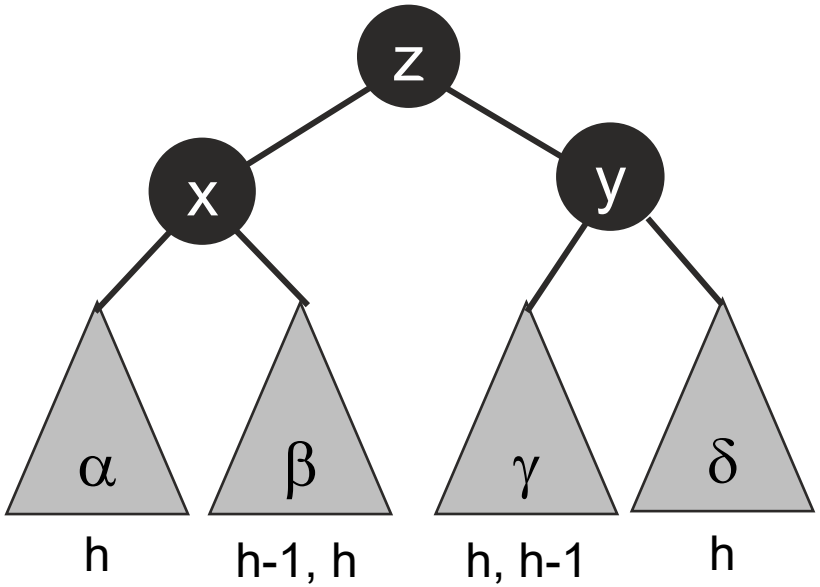


Ne felejtsük el forgatás után a csúcsok magasság mezőjét frissíteni!

… a második forgatás



… majd **x körül balra.**
Ez a második forgatás ugyanaz, mint az első esetnél.



Ezzel helyreállt az AVL fa tulajdonság ebben a részfában.

Ne felejtsük el forgatás után a csúcsok magasság mezőit frissíteni!

Kiegyensúlyozás - röviden

(++, -)	gyereket jobbra, szülőt balra forgatni
(++, 0) és (++, +)	szülőt balra forgatni
(--, +)	gyereket balra, szülőt jobbra forgatni
(--, 0) és (--, -)	szülőt jobbra forgatni