

ADATSZERKEZETEK ÉS ALGORITMUSOK

AVL fa

AVL fa

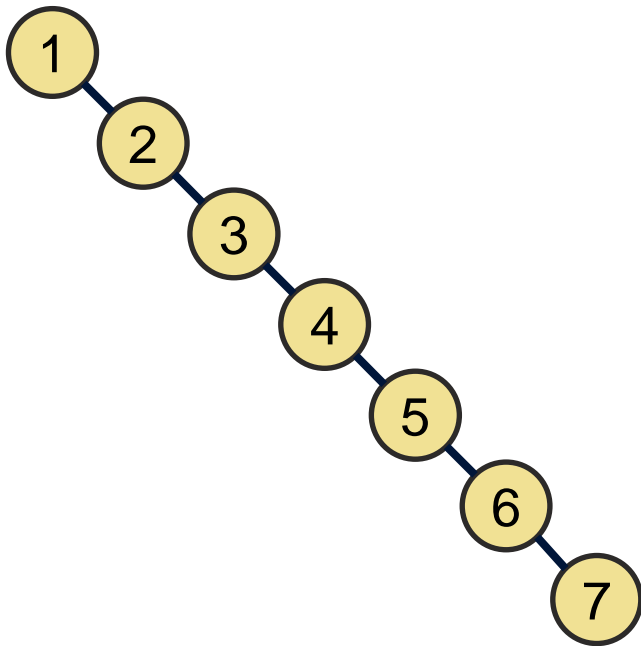
- Bináris keresőfa.
- A bal és jobb részfák magassága legfeljebb 1-gyel tér el egymástól.
- Az AVL fa minden részfája is AVL fa.

Motiváció

Szűrjük be a következő elemeket egy fába az alábbi sorrendben:

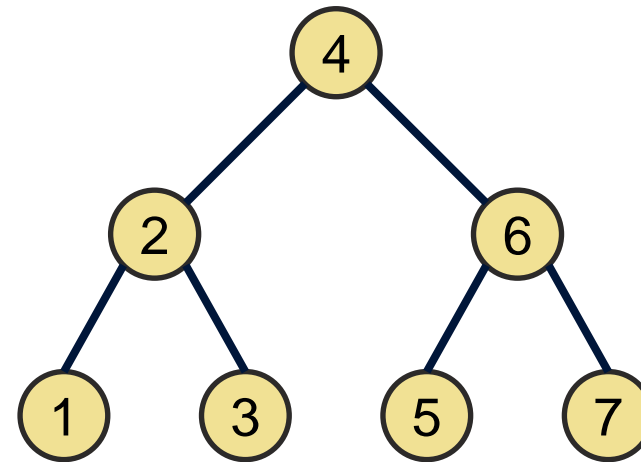
1, 2, 3, 4, 5, 6, 7

Bináris keresőfa:



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

AVL fa:



Garantált **$1,44 \cdot \log_2 n$** 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 konkrét teszt eredménye, csak a nagyságrendek szemléltetésére | | |

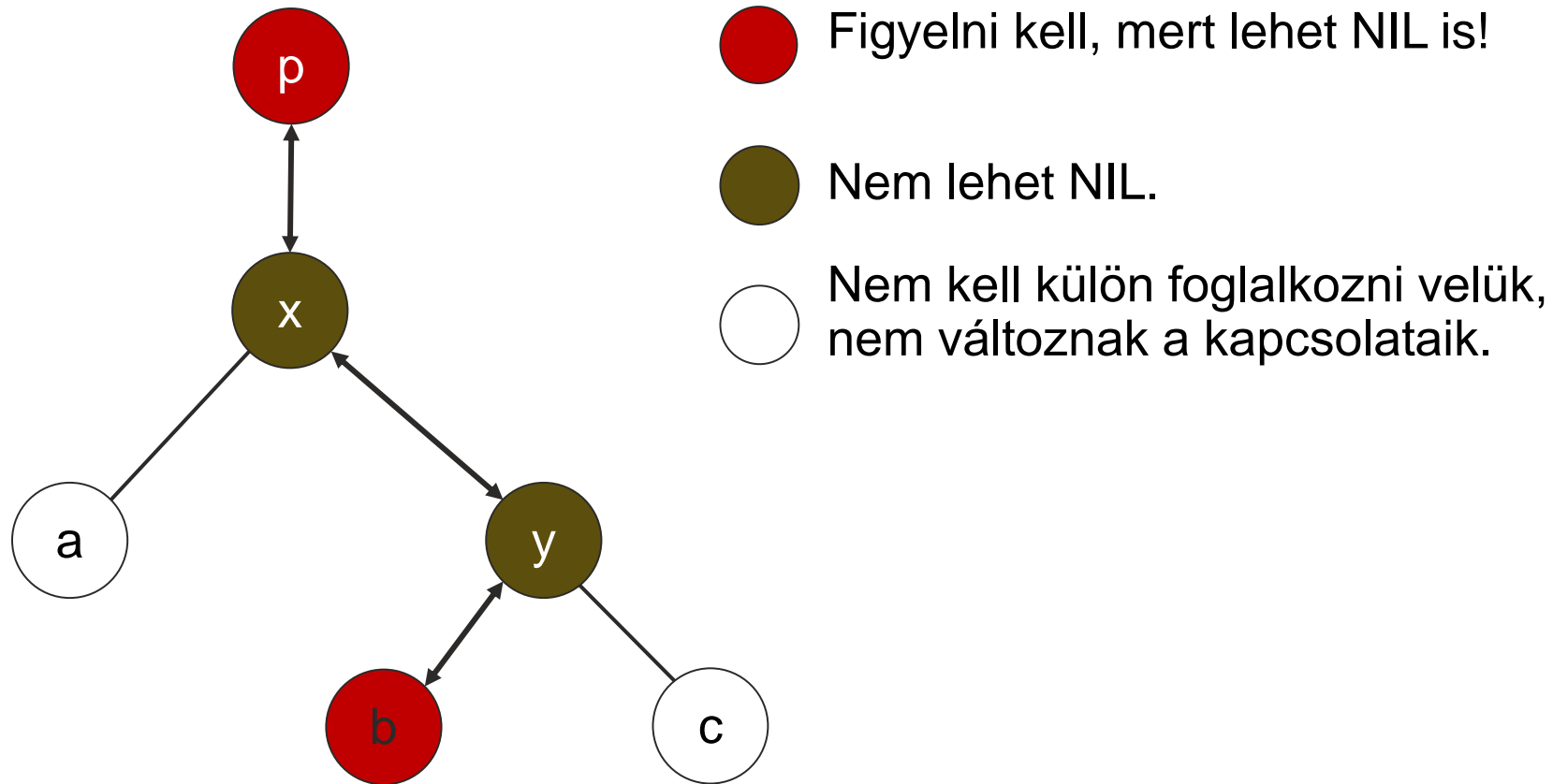
Forgatások

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

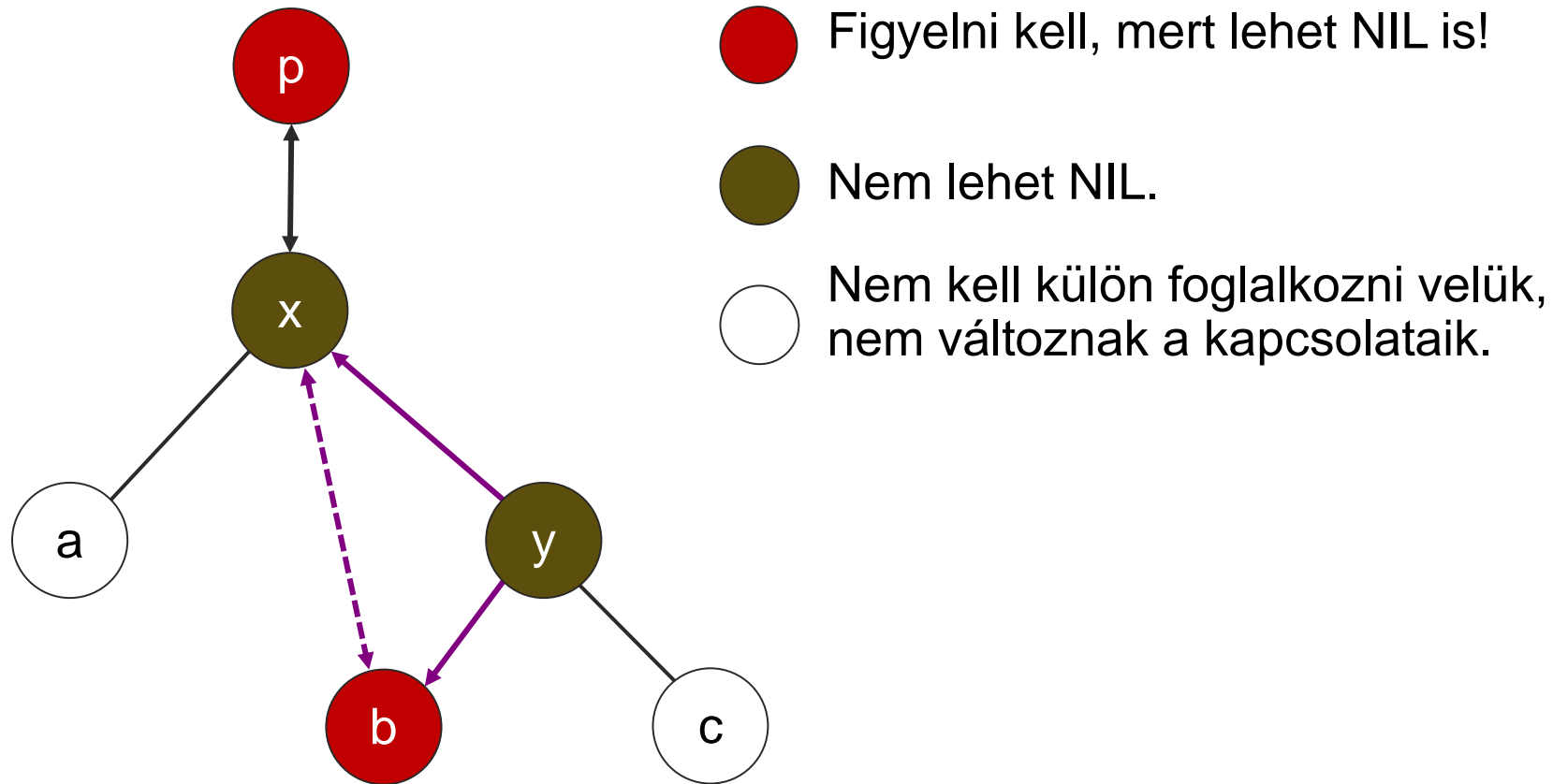
A forgatások során egy csúcspont eggyel lejjebb, és annak egyik gyereke pedig eggyel feljebb kerül a fában.

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

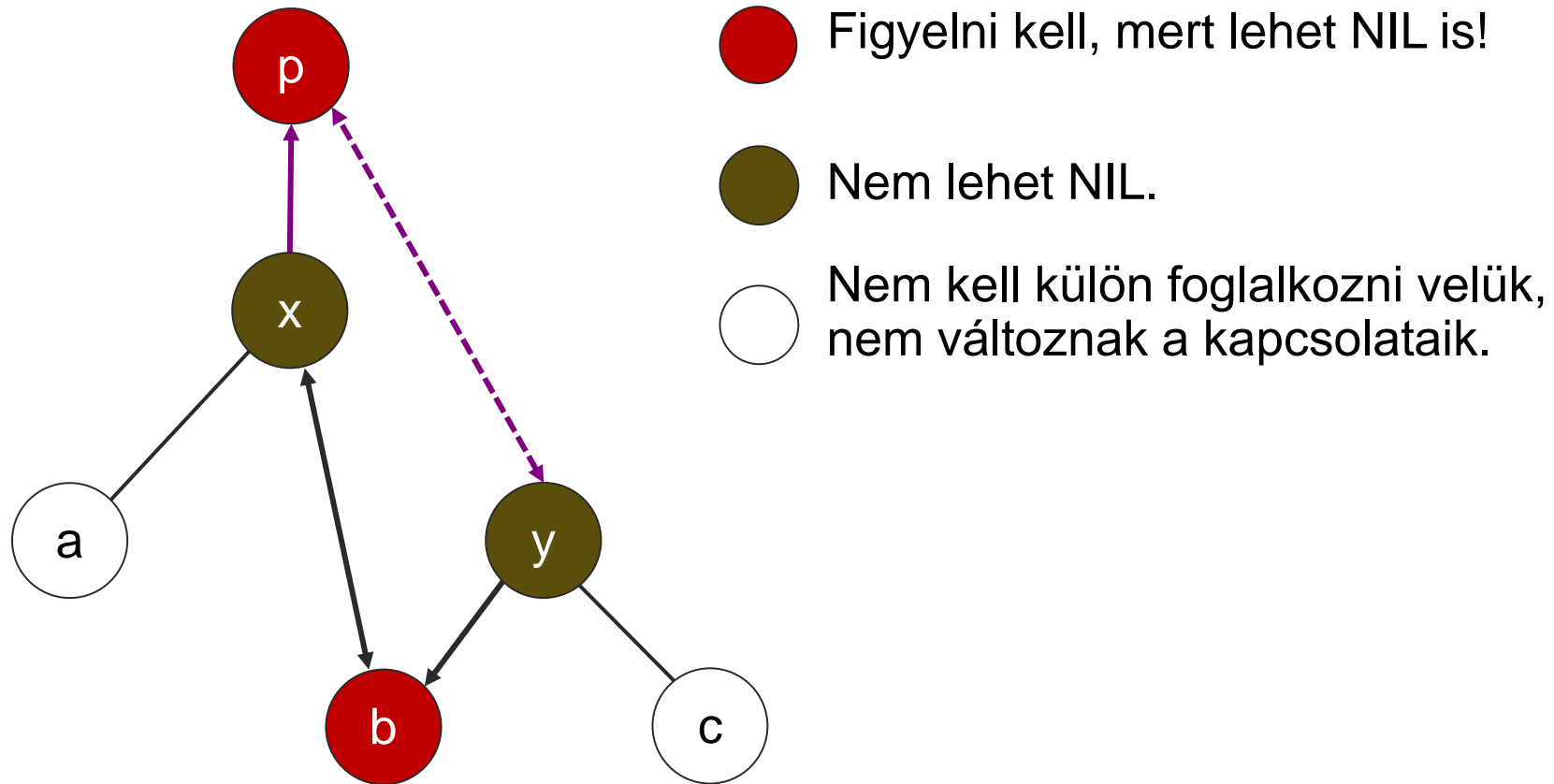
Forgatások: balra forgatás x körül



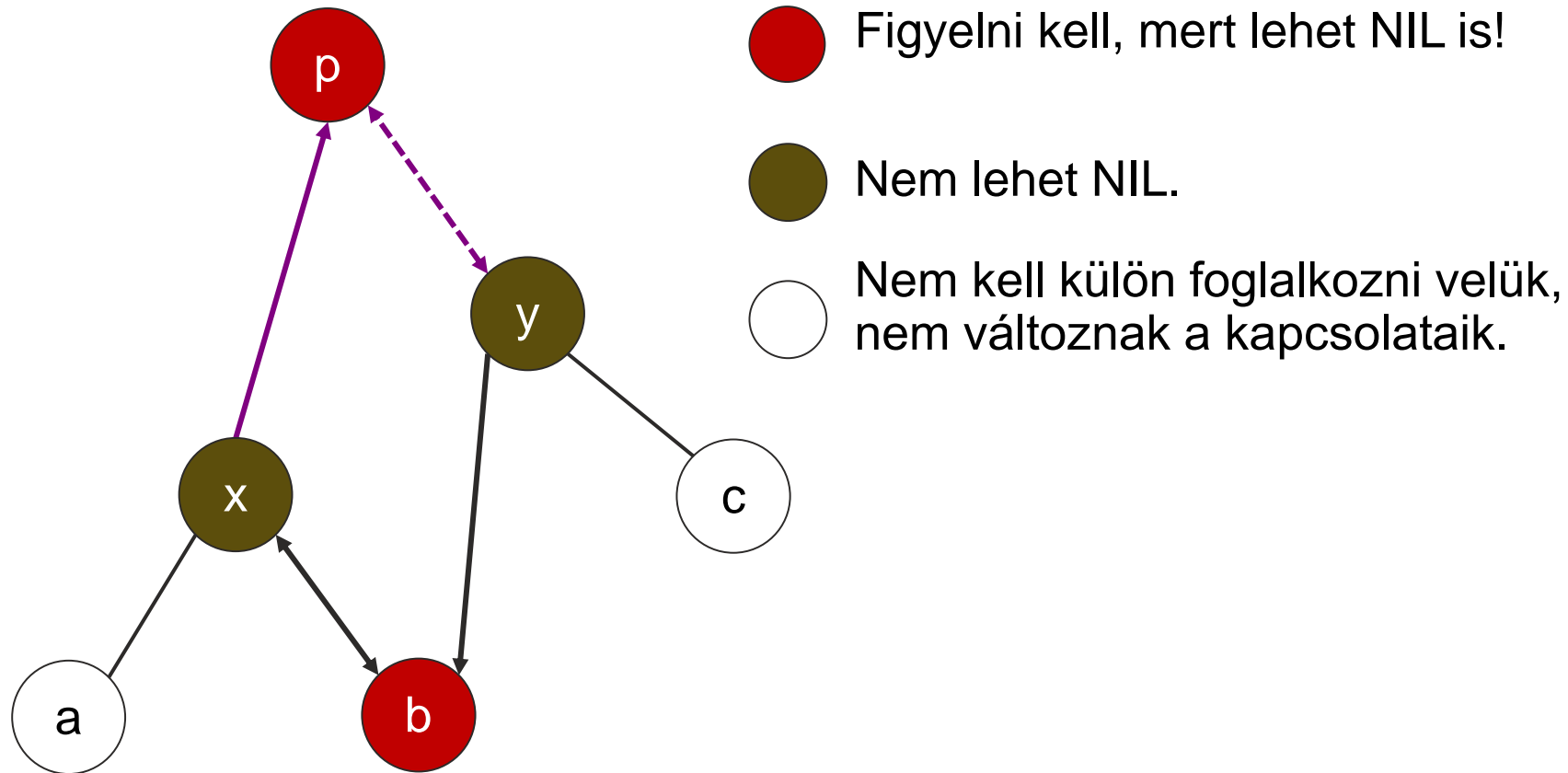
Forgatások: balra forgatás x körül



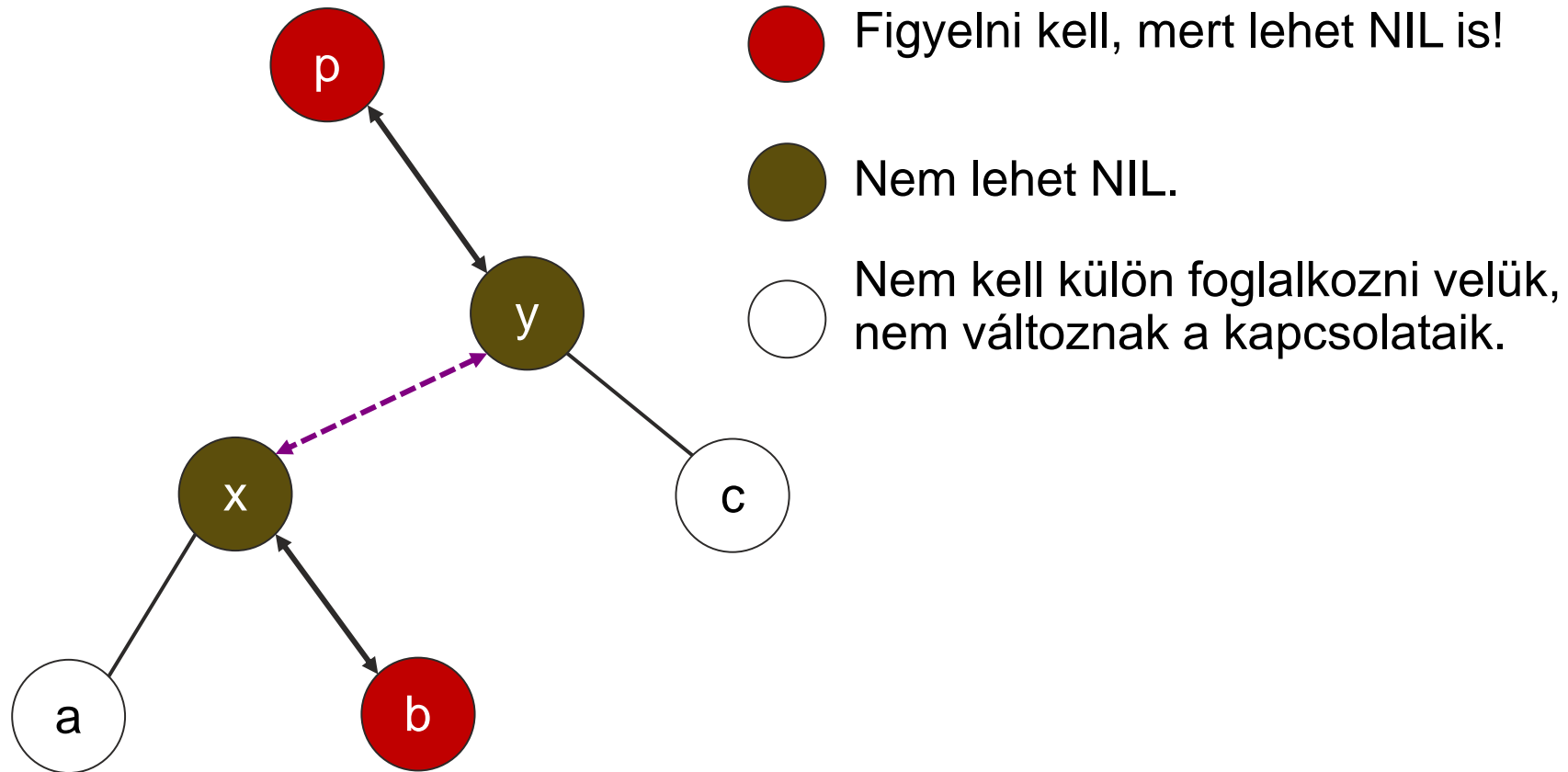
Forgatások: balra forgatás x körül



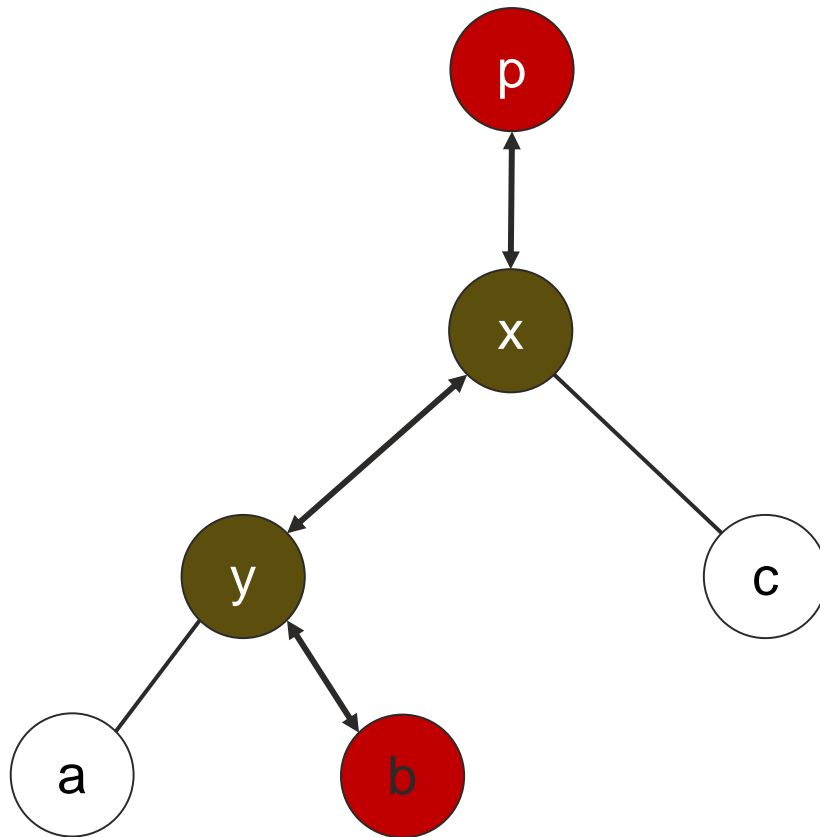
Forgatások: balra forgatás x körül






Forgatások: balra forgatás x körül

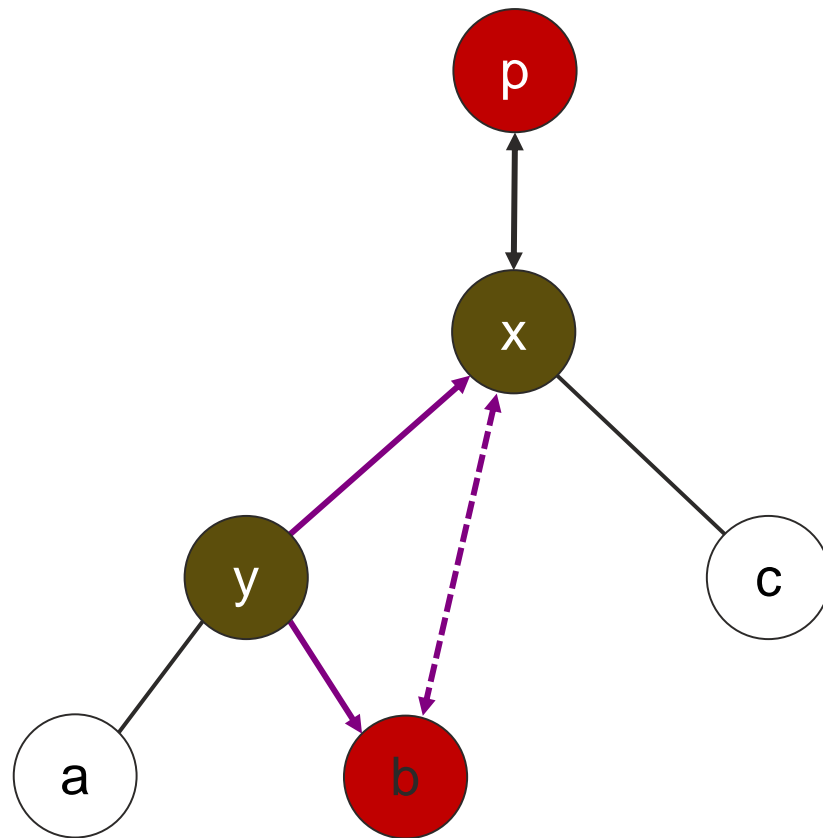





Forgatások: jobbra forgatás x körül



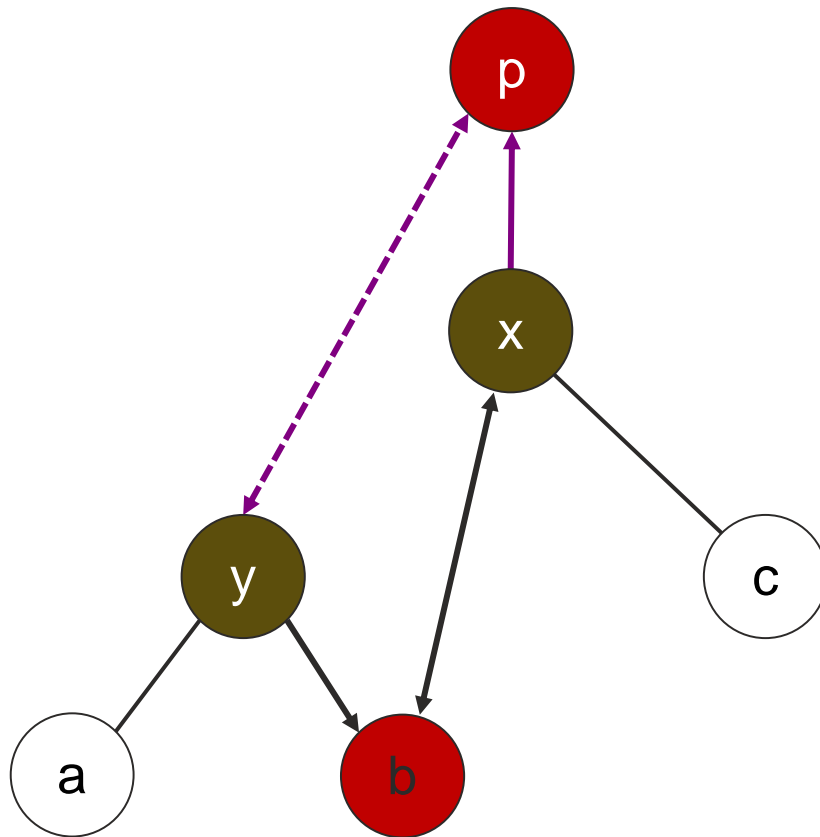
-  Figyelni kell, mert lehet NIL is!
-  Nem lehet NIL.
-  Nem kell külön foglalkozni velük, nem változnak a kapcsolataik.




Forgatások: jobbra forgatás x körül



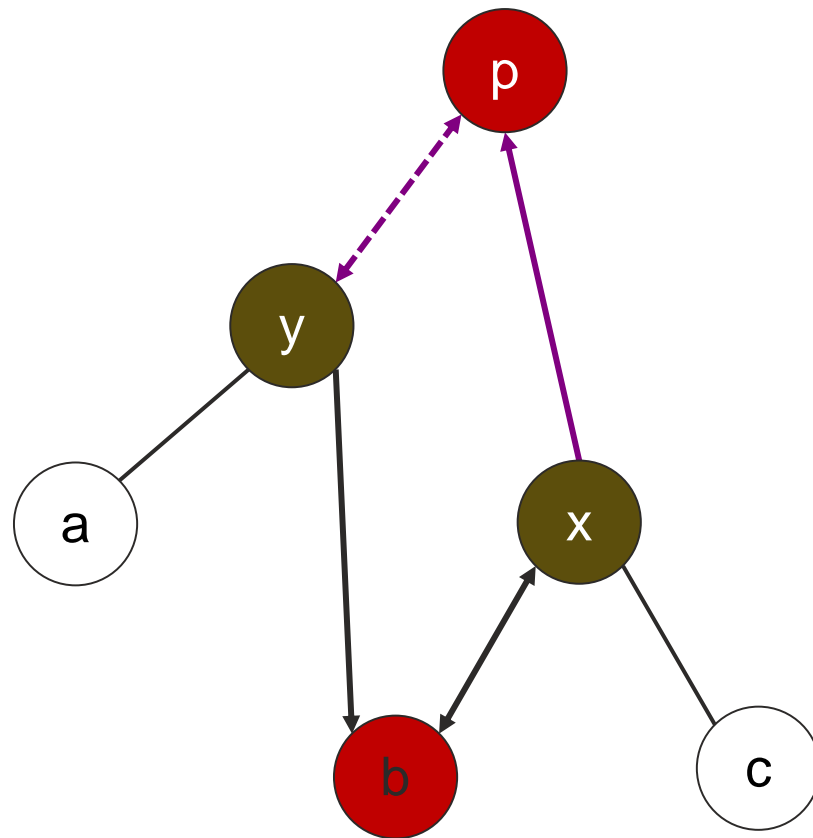
-  Figyelni kell, mert lehet NIL is!
-  Nem lehet NIL.
-  Nem kell külön foglalkozni velük, nem változnak a kapcsolataik.




Forgatások: jobbra forgatás x körül



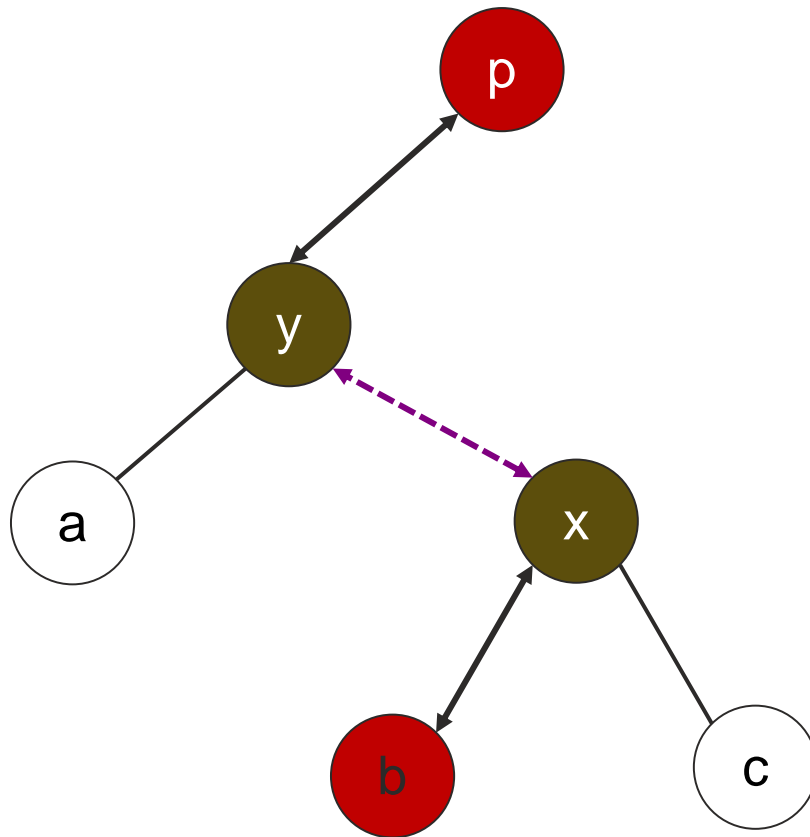
-  Figyelni kell, mert lehet NIL is!
-  Nem lehet NIL.
-  Nem kell külön foglalkozni velük, nem változnak a kapcsolataik.




Forgatások: jobbra forgatás x körül



-  Figyelni kell, mert lehet NIL is!
-  Nem lehet NIL.
-  Nem kell külön foglalkozni velük, nem változnak a kapcsolataik.

Forgatások: jobbra forgatás x körül



-  Figyelni kell, mert lehet NIL is!
-  Nem lehet NIL.
-  Nem kell külön foglalkozni velük, nem változnak a kapcsolataik.

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)

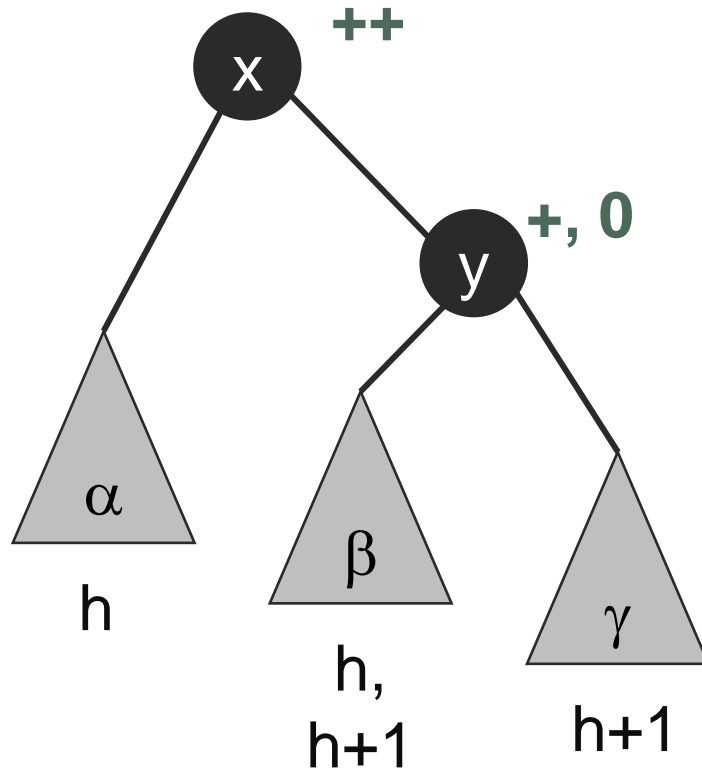
ezért egyértelműen megfeleltethető a diákban használt ++, +, 0, -, -- jelöléssel.

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.

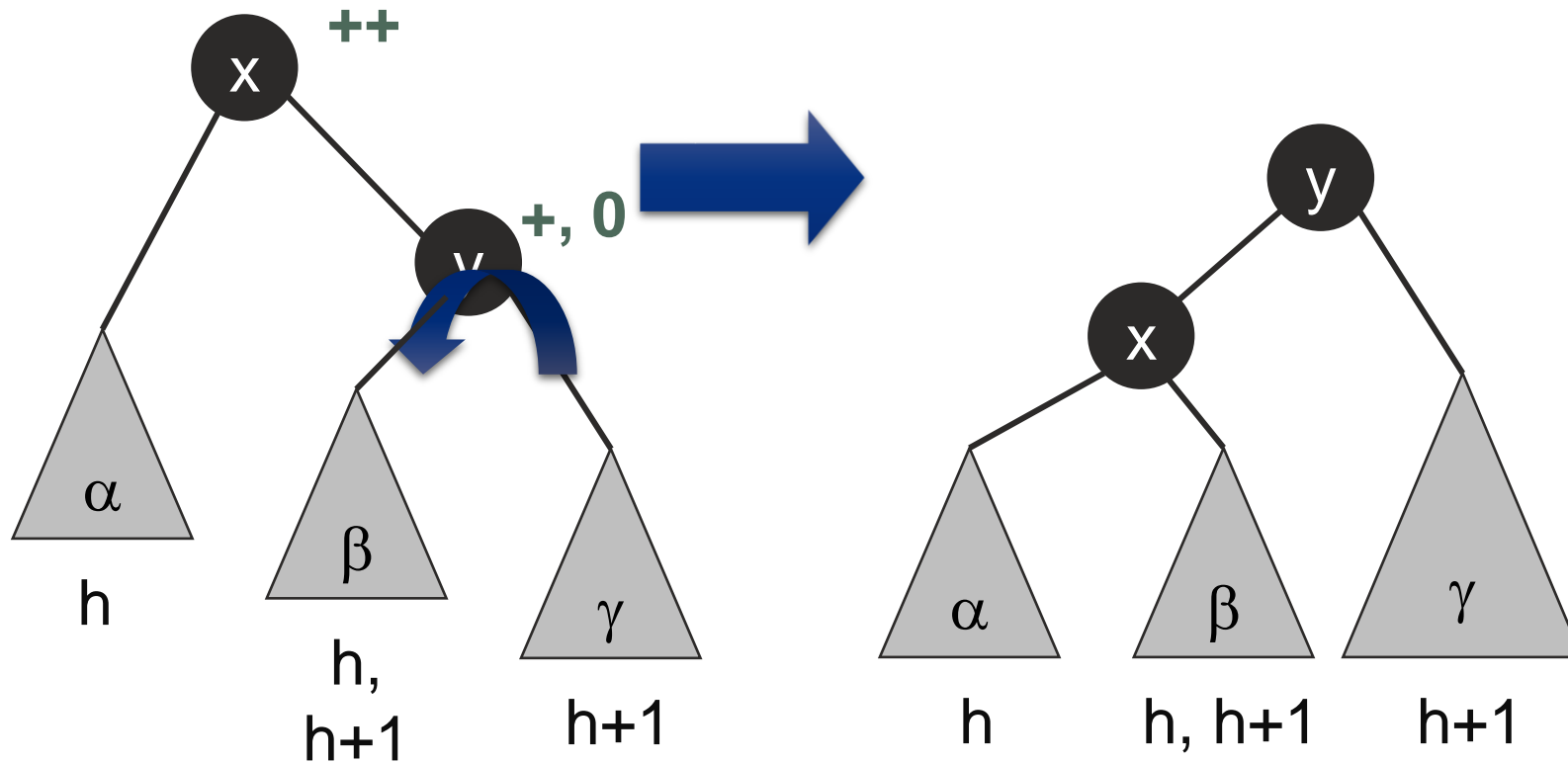


x körüli **balra forgatással** helyreáll az AVL tulajdonság

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

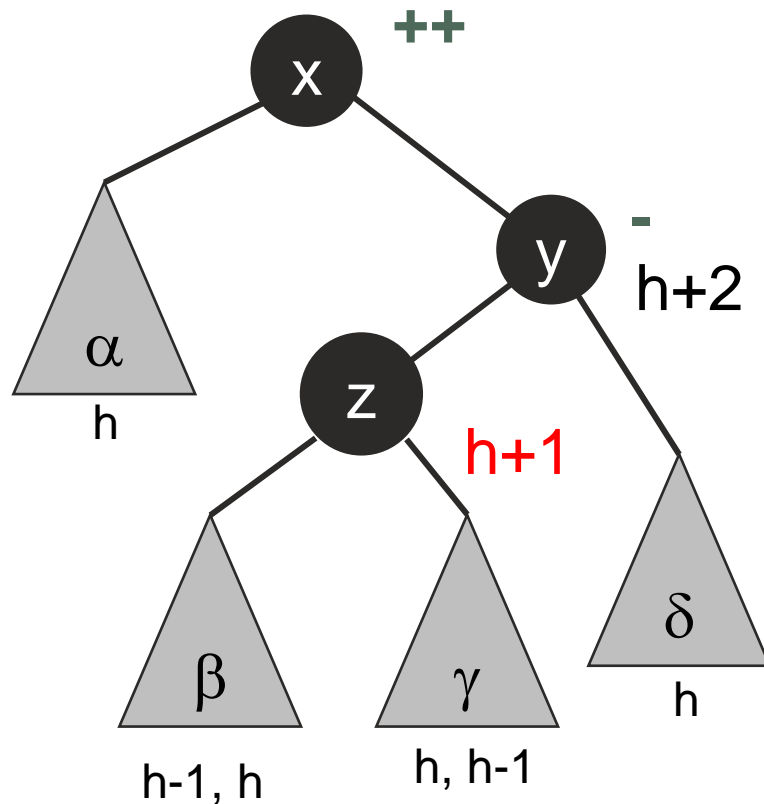
... a forgatás hatása

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



Ne felejtsük el forgatás után a csúcsok magasság mezőit frissíteni!

Kiegyensúlyozás (++, -)



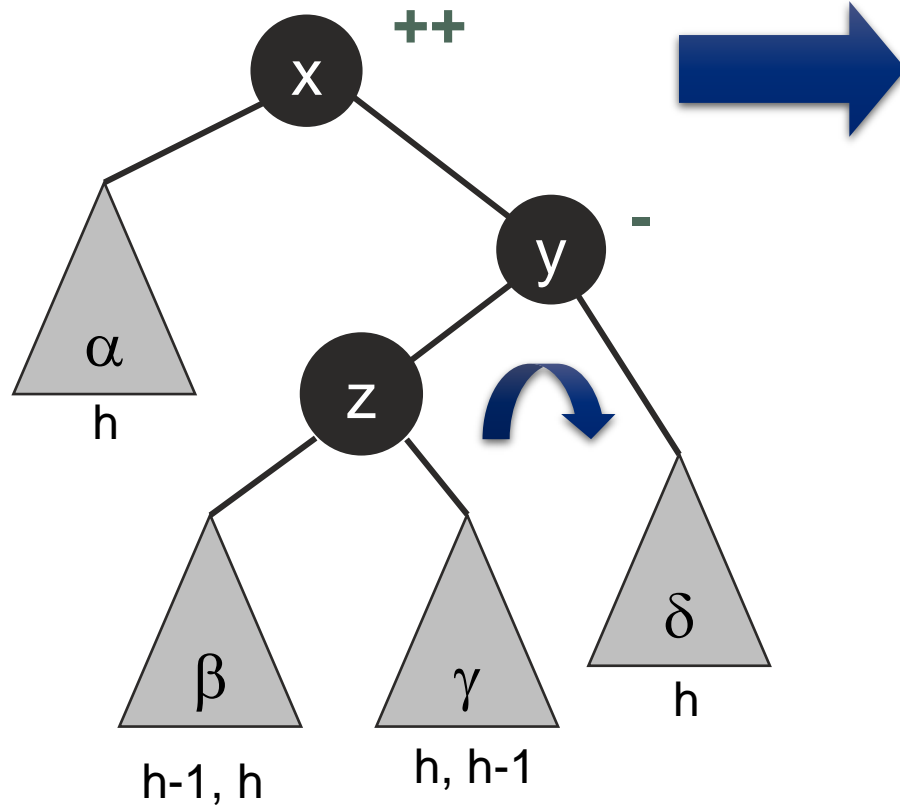
x bal oldali részfája h , jobb oldali pedig $h+2$ magas, és ez sérti az AVL fa tulajdonságot.

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

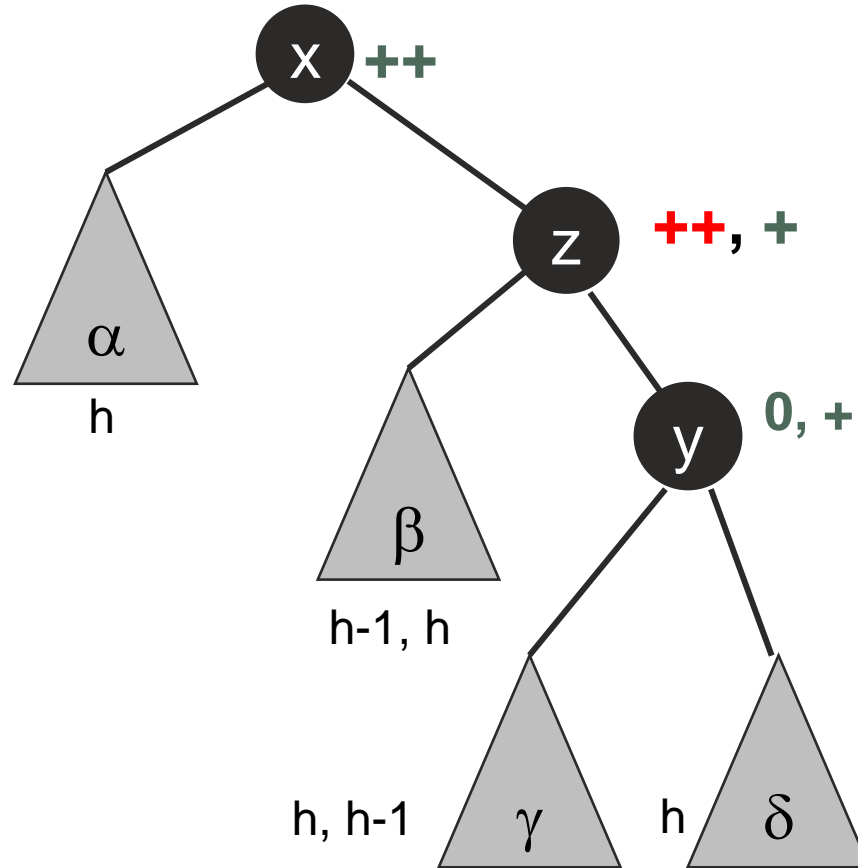
Ezért először egy y körüli **jobbra forgatás**sal szétszedjük z részfáit.

Ennek tükörképe a $(--, +)$ eset.

Az első forgatás...

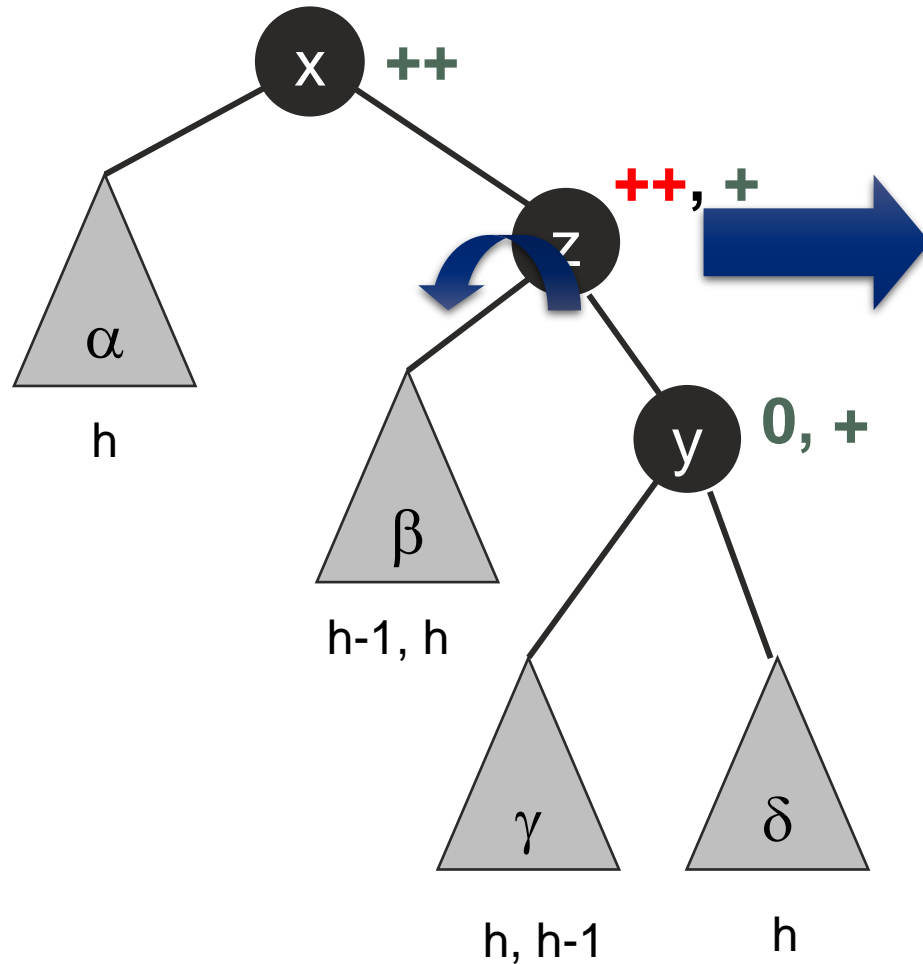


Dupla forgatás: először y körül jobbra...

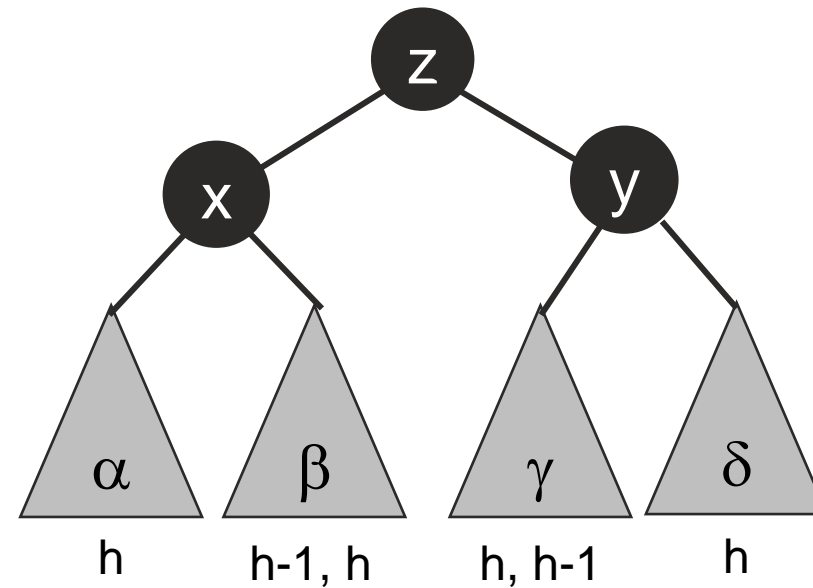


Ne felejtsük el forgatás után a csúcsok magasság mezőjét frissíteni!

... a második forgatás



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



Ezzel helyreállt az AVL fa
tulajdonság ebben a részében.

**Ne felejtsük el forgatás után a
csúcsok magasság mezőit frissíteni!**

AVL fa

Kiegyensúlyozás - röviden

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

AVL megvalósítás

Következő téma