Splay trees

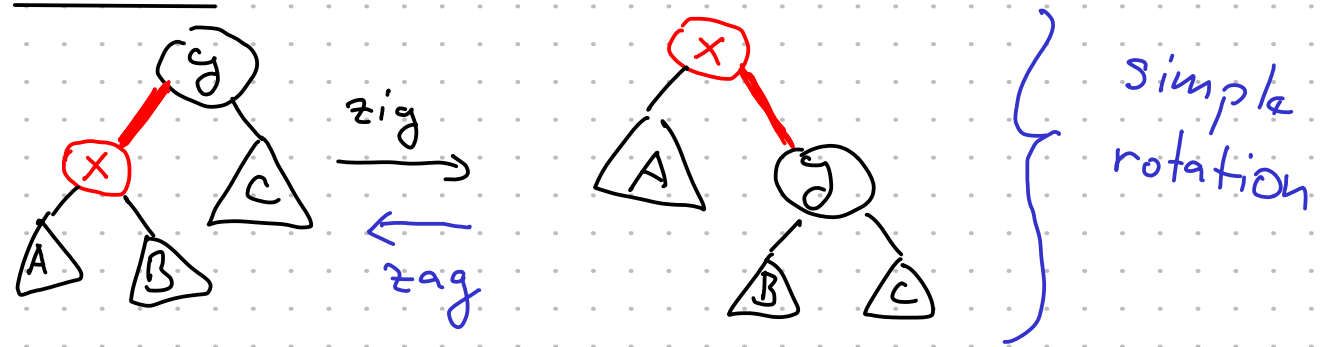
- self-adjusting BST
- balancing via splay operation

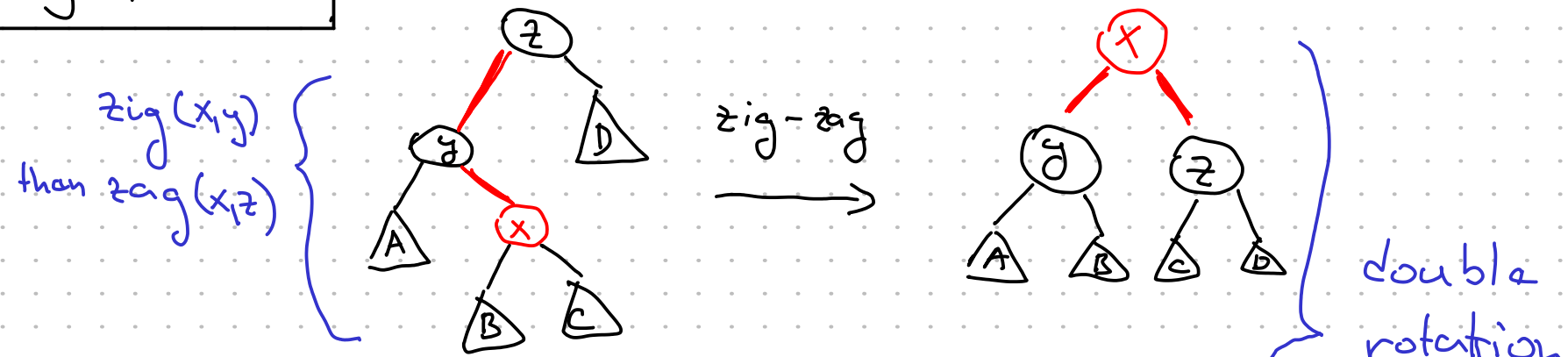
Splay(x):

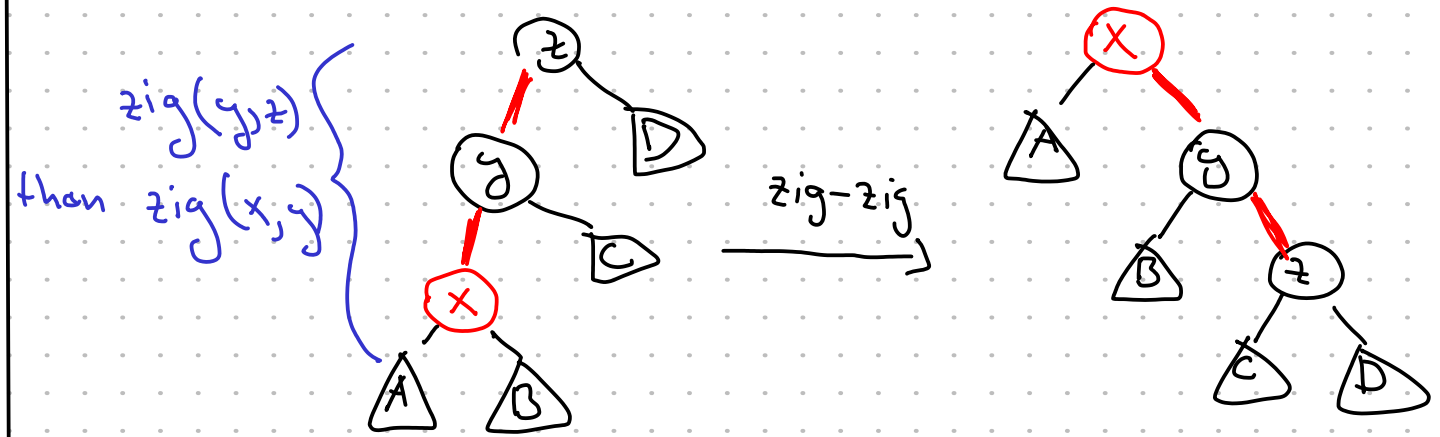
- move to root using rotations
- use double rotations if possible

⇒ ≤ 1 simple rot. (at the root)

Rotations







Operations:

- always splay the deepest node operation touches
- up to $\Theta(n)$ worst case complexity (tree can be path)
- splaying ensures amortized $O(\log n)$ (analysis at the lecture)
- ops work like "ordinary" BST only with the splay at the end
  → another way is to first splay and then use split and join (easier analysis)

Find(x)

- search for x (like ordinary BST)
- then splay x (if found) or last visited node (if not found)

Insert (x)

- search for x
- if not found insert under last visited node
- splay(x)
  ! even if found in search !

Delete (x)

- search for x
- x has ≤ 1 children ⇒ remove x (like in ordinary BST)
- otherwise find successor, swap it with x and remove x (it has ≤ 1 children now)
- splay parent of removed node or last visited node if x not found