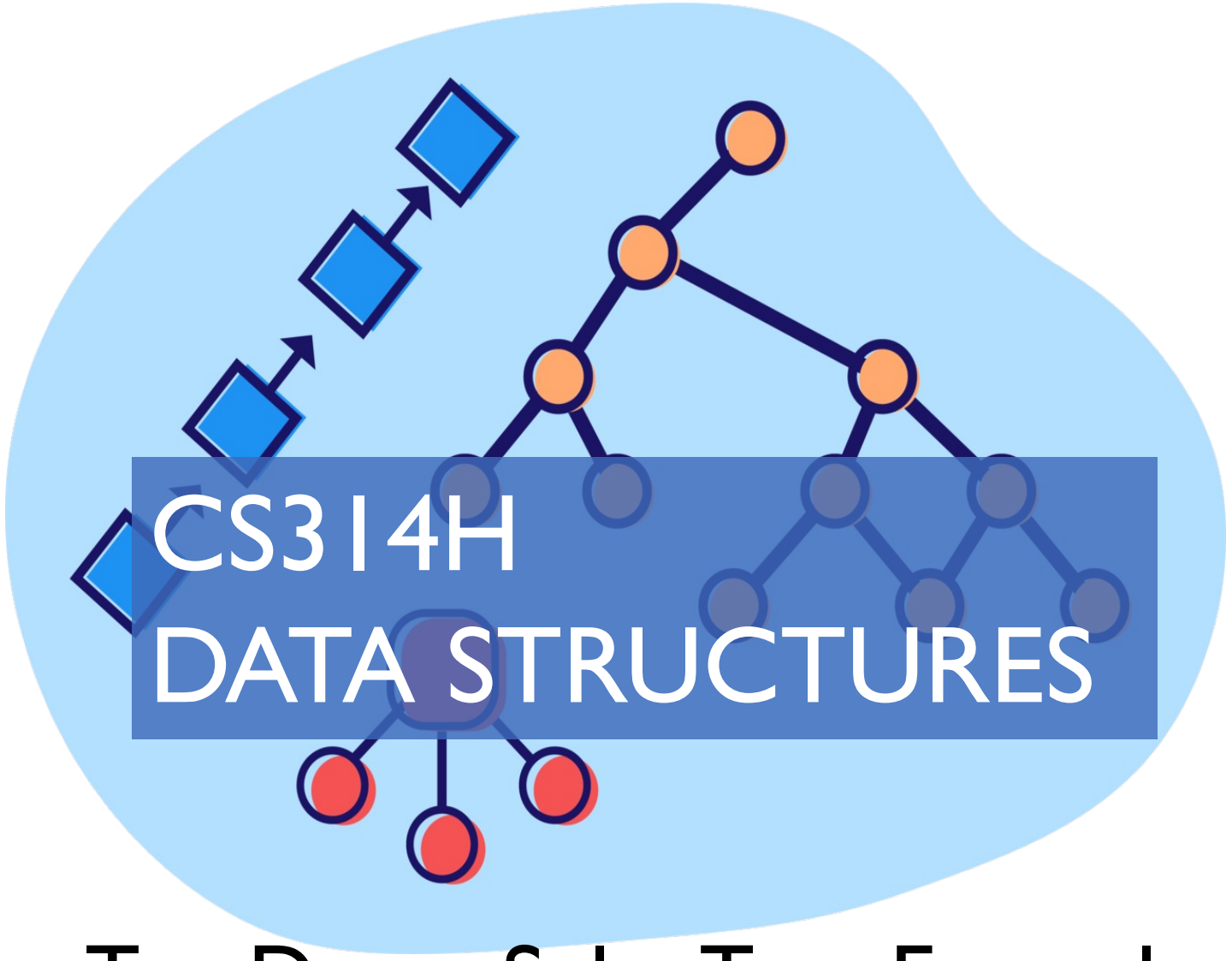


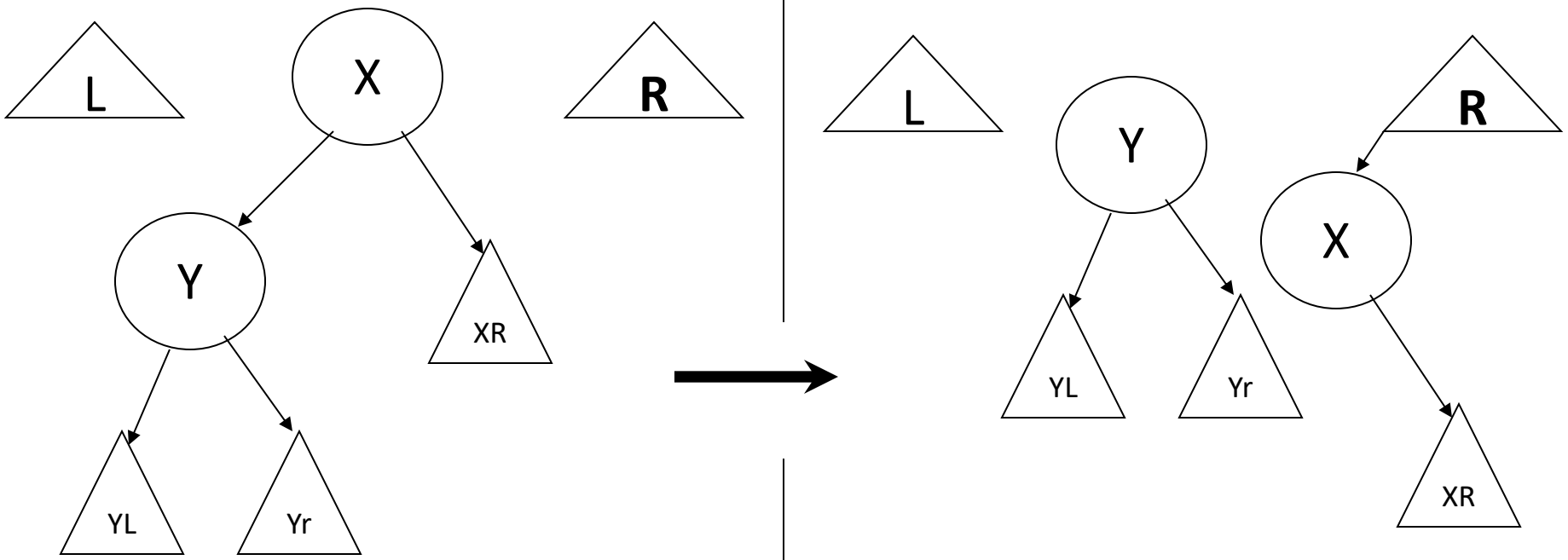
Top Down Splay Tree Example

Mikyung Han

Top down: Why not traverse just once?

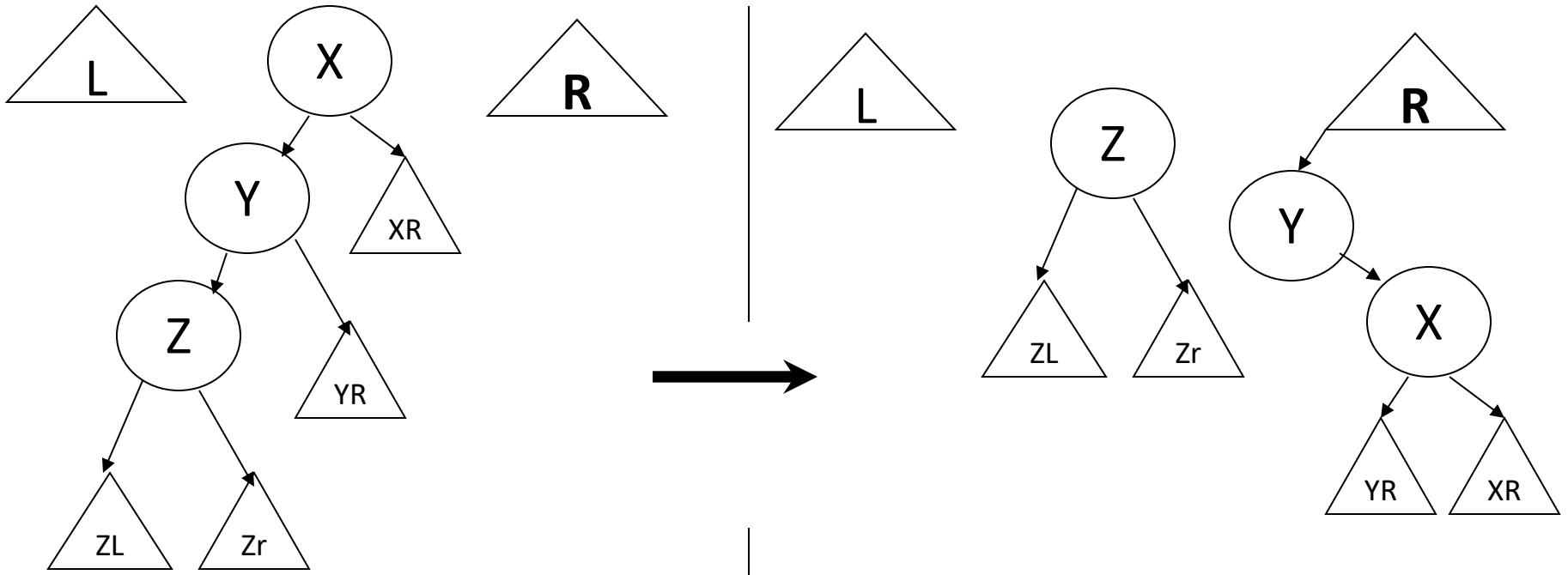
- Bottom-up splaying requires traversal from root to the node that is to be splayed, and then rotating back to the root
- Top down tries to eliminate one of the traversals

Case 1: Zig



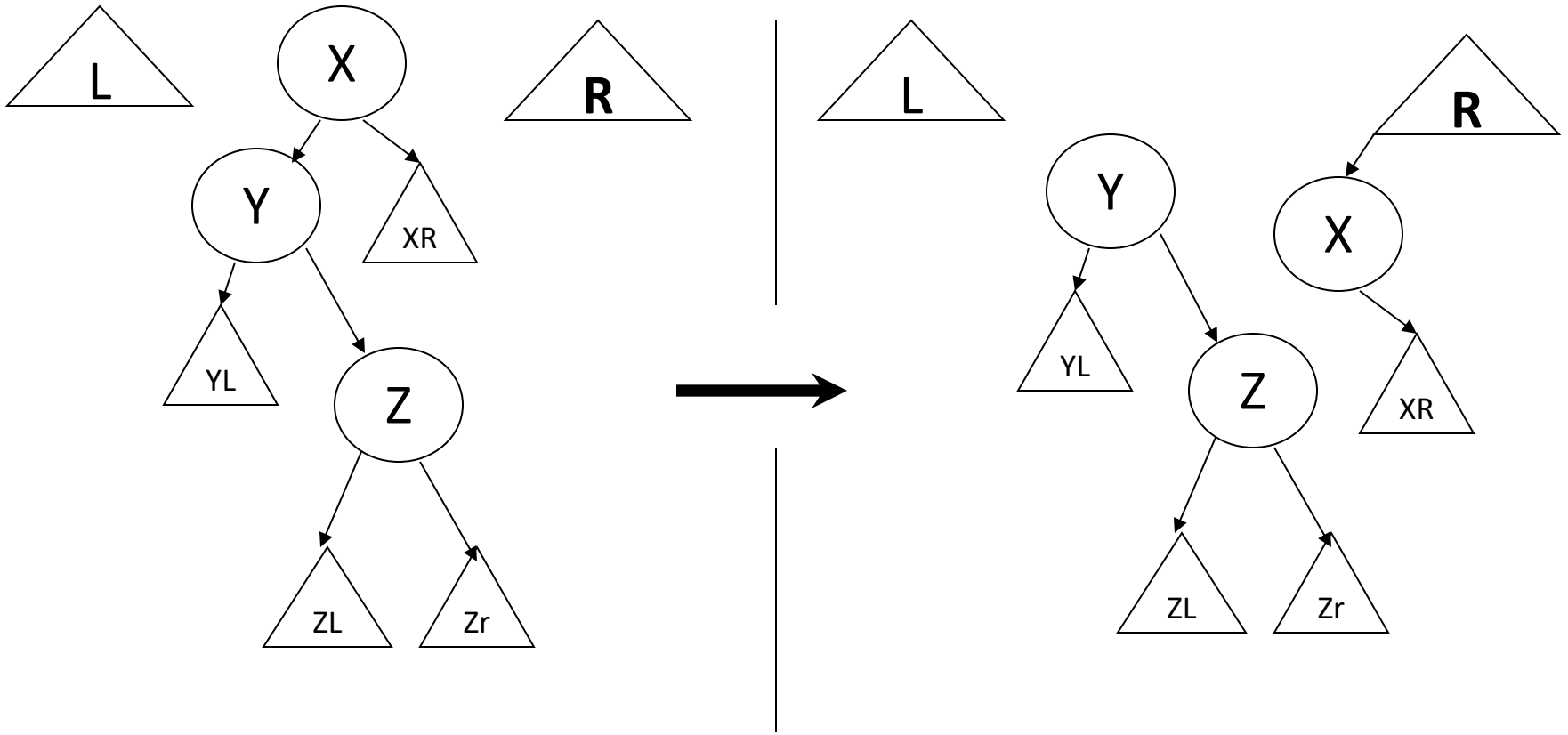
If Y should become root, then X and its *right* subtree are made left children of the smallest value in R, and Y is made root of “center” tree

Case 2: Zig-Zig



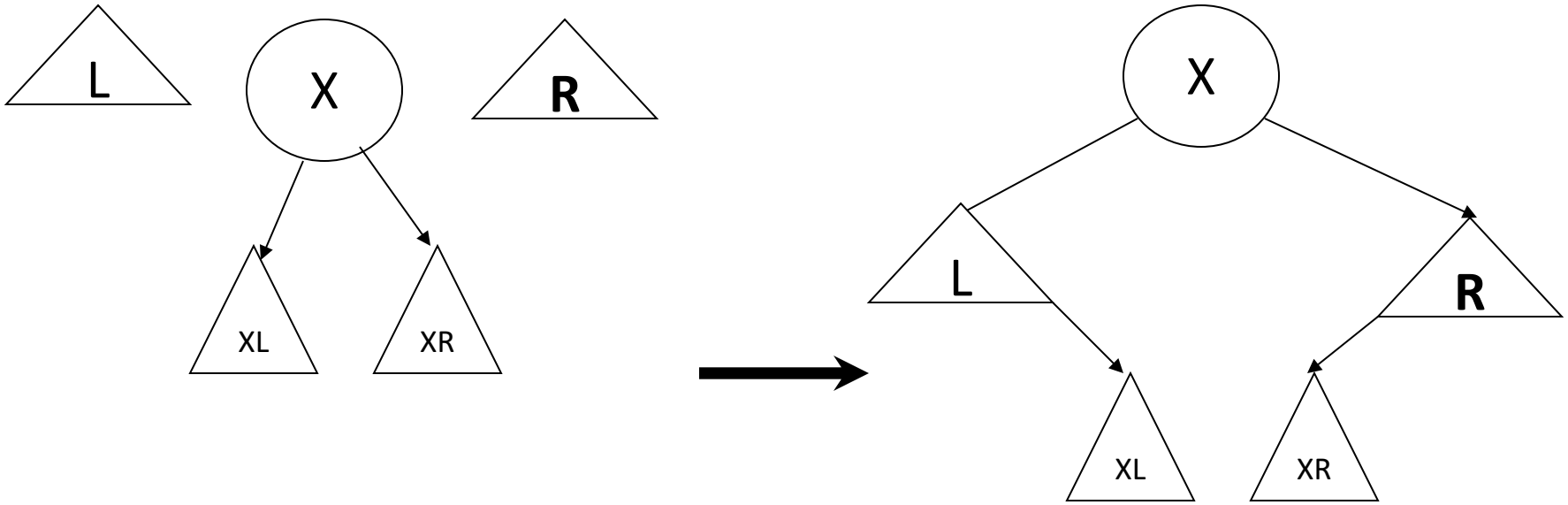
The value to be splayed is in the tree rooted at Z. *Rotate Y about X* and attach as left child of smallest value in R

Case 3: Zig-Zag(Simplified)



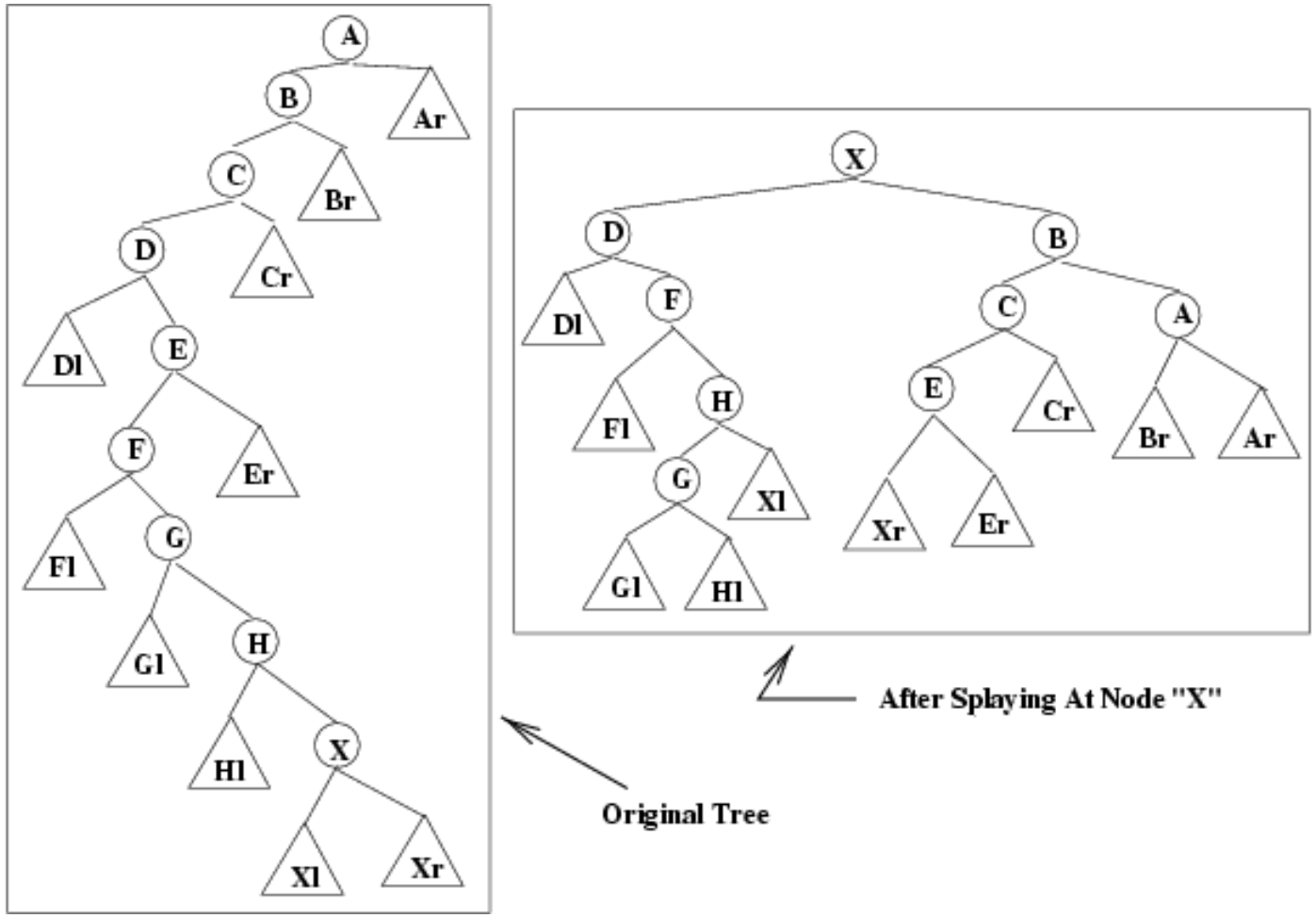
The value to be splayed is in the tree rooted at Z. To make code simpler, the Zig-Zag rotation is reduced to a single Zig. This results in more iterations in the splay process.

Reassembling the Splay Tree

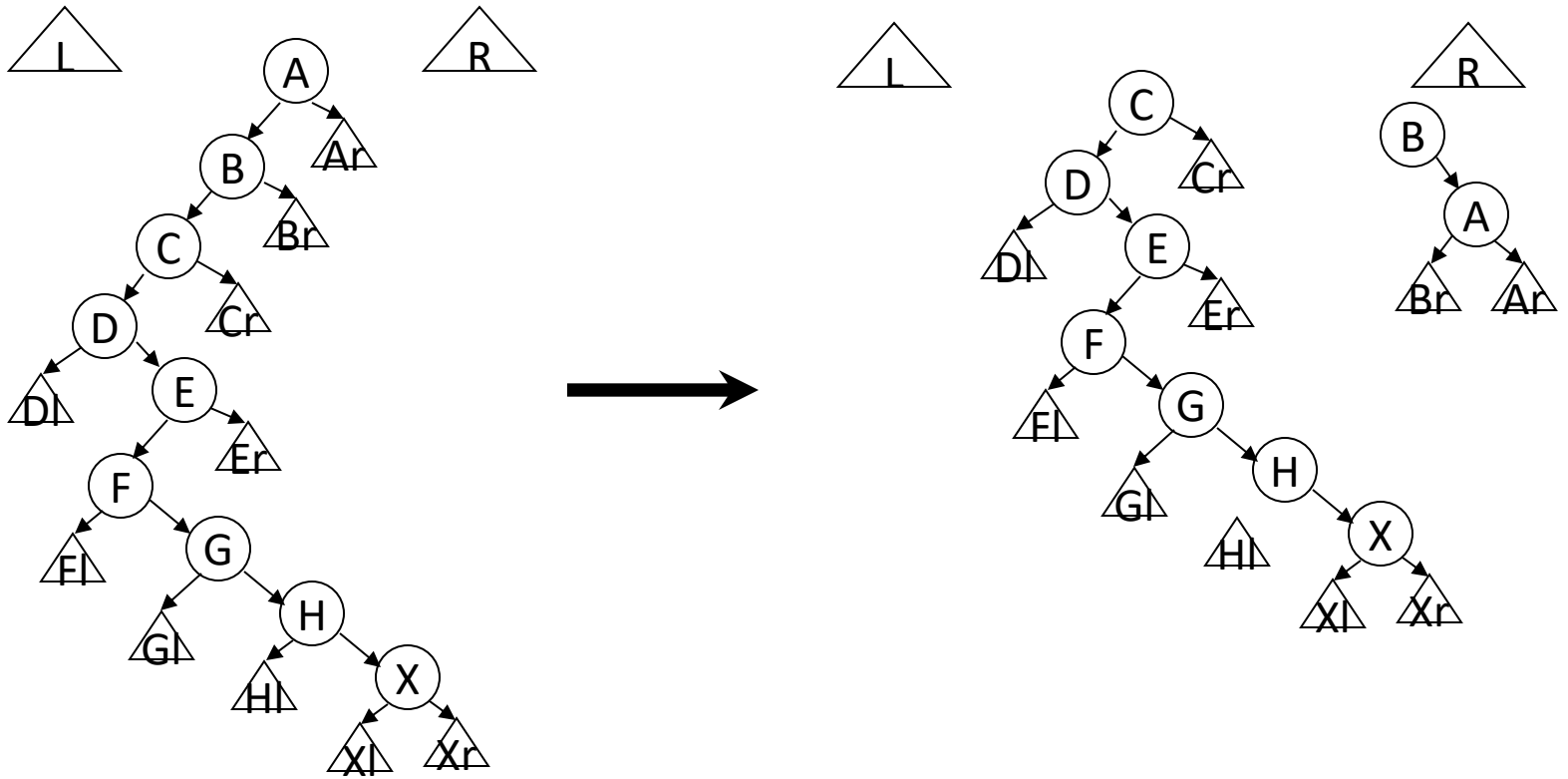


When the value to be splayed to the root is at the root of the “center” tree, we have reached the point where we are ready to reassemble the tree. This is accomplished by a) making XL the right child of the maximum element in L, b) making XR the left child of the minimum element in R, and then making L and R the left and right children of X

Example: (from bottom-up)



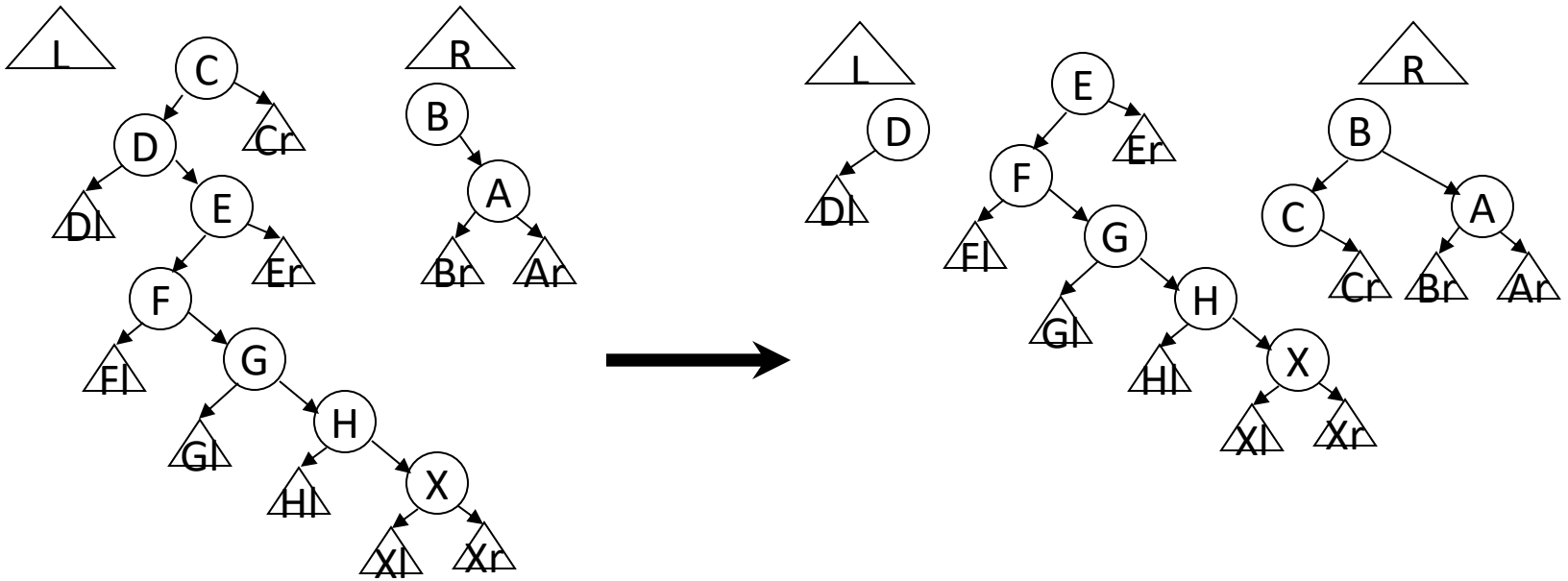
Operation 1: Zig-Zig



Rotate B around A and make L child of minimum element in R (which is now empty)

L is still empty, and R is now the tree rooted at B. Note that R contains nodes > X but not in the right subtree of X.

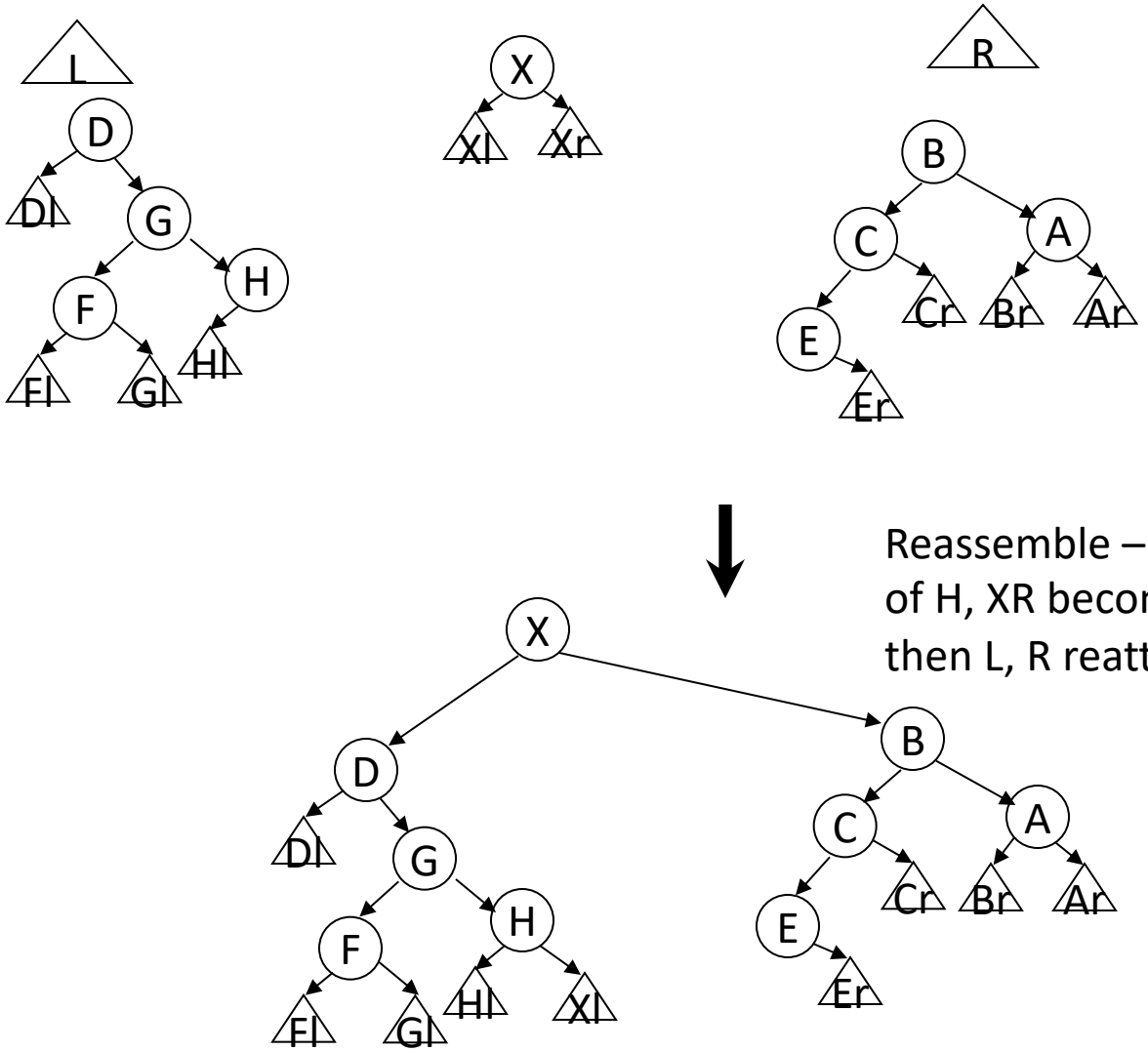
Operation 2: Zig-Zag



Just perform
Zig
(simplified
Zig-Zag)

L was previously empty and it now consists of node D and D's left subtree

After X reaches root:



This configuration was achieved by doing Zig Zig (of F, G) followed by a Zig (node H)

Reassemble – XL becomes right subtree of H, XR becomes left subtree of E, and then L, R reattached to X

Note that this is not the same tree as was obtained by doing BU splaying.