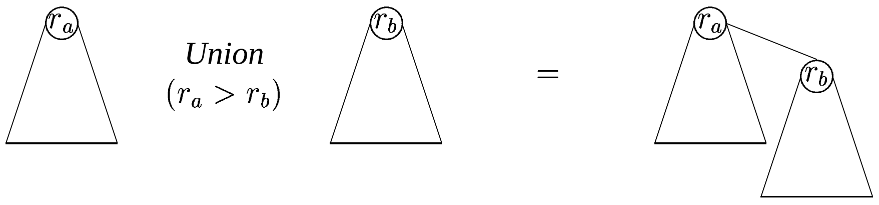
Disjoint Sets using Ranks

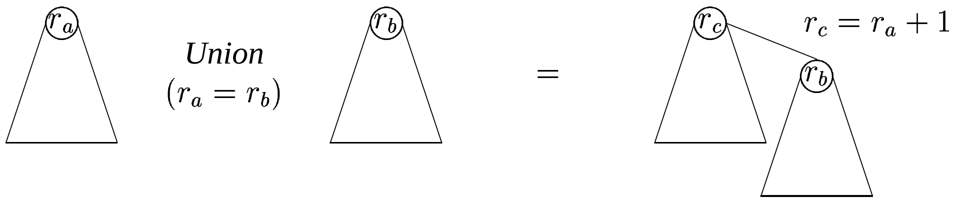
In connection with Disjoint Sets, we ensured logarithmic height by joining trees according to their relative sizes (union by size). Now, we consider an alternative where we store a *rank* instead of the size.

When a new singleton set is created (the operation *MakeSet*), the node is given rank 0. When two trees are joined by a union, the root with the smaller rank is made a child of the other (details below):



Thus, no nodes have their ranks changed.

If the roots have the same rank, an arbitrary choice is made and the root after the operation has its rank incremented:



The operation *Find* does not change the structure.

Question a: Show by induction in the number of operations which are carried out that there are at least 2^r nodes in a tree of rank r. □

Question b: Argue that if a node of rank r has a child of rank r', then $r > r'$. □

Question c: Show that *Find* is $O(\log n)$, where n is the number of elements in the structure. □

The following gives a connection to the rank defined in [K90].

Question d: Show that the rank of a node is always one less than the height of the node. □