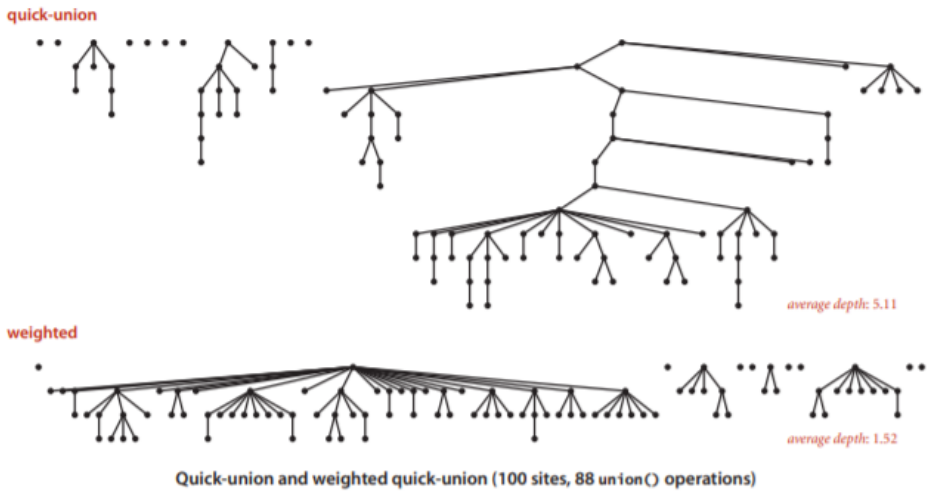
AnthroChassidus

My design was to use a Union Find data structure. Which is like an array that holds values that point to all of the values that it connects to. It's like an array that holds a bunch of different trees this alone would run at O(N) time for union() and find() in the worst case depending on the length of the tree.

I did a weighted quick union with path compression in order to get the fastest time I could. A weighted union-find runs faster at a speed of O(logN) for union() and find(). The weighted version keeps track of the sizes of each tree and connects the smaller tree to the larger one. This way we avoid the worst-case scenario of having a super long tree.

So I had to do a path compression in order to speed it up. You set every other node in the path to point to its "grandparent". This makes the tree almost completely flat and therefore will run in almost constant time for our union and find methods. So I'm left with the constructor running at O(n) union running at almost O(1) and find running at almost O(1).




Quick-union and weighted quick-union (100 sites, 88 `union()` operations)