

Union-Find data structure

Each node has a “parent” attribute and a “rank” attribute.

Makeset(x):

```
parent(x) <- x
rank(x) <- 0
```

Union(x,y) //prerequisite: x and y are tree roots

```
Case 1: (rank(x)<rank(y)): parent(x)<-- y
Case 2: (rank(y)<rank(x)): parent(y)<-- x
Case 3: (rank(x)=rank(y)): parent(x)<-- y; rank(y)<-- rank(y)+1
```

FindSet(x):

```
if (x < > parent(x))
  then parent(x)<-- FindSet(parent(x))
return parent(x)
```

After doing $O(m)$ find operations and $O(n)$ makeset and union operations, it is easy to see that the overall complexity is $O(m + n + \sum_x \#\{\text{updates of } \text{parent}(x)\})$.

The following invariant is easy:

Lemma 1 *At any time, along any leaf-root path, the node ranks are strictly increasing.*

Now, color a node x in red iff $\text{rank}(p(x)) > 2\text{rank}(x)$. The lemma implies that every time $\text{parent}(x)$ is updated, the value $\text{rank}(\text{parent}(x))$ increases, and so x becomes red after at most $\text{rank}(x)$ updates of its pointer to its parent. Thus, if D denotes the maximum number of red nodes along any leaf-root path at any time, then the overall complexity is at most

$$O(m + n + \sum_i i \#\{x : \text{rank}(x) = i\} + mD).$$

The following invariant is also easy:

Lemma 2 *For any tree T , we have $|T| \geq 2^{\text{rank}(\text{root}(T))}$.*

The first two lemmas imply that the maximum rank is at most $\log_2 n$ and that $D \leq \log_2 \log_2 n$.

Lemma 3 *For any i , $\#\{x : \text{rank}(x) = i\} \leq n/2^i$.*

Proof: Consider a node u (which may or may not be a tree root) of rank equal to i , and let T_u denote the set of nodes in the tree of which it was root, right after its rank was incremented from $i - 1$ to i . By the second lemma, T_u has size at least 2^i . From that point on, any node of T_u is in a tree whose root is either u (at first) or some node of higher rank (possibly after some further unions.) Thus, if v is any other node of rank equal to i , T_v and T_u are disjoint, hence the lemma.

So the overall complexity is bounded by

$$O(m + n + \sum_i in/2^i + m \log_2 \log_2 n) = O(m \log \log n).$$