Data structure for Union-Find

Find(x)

Read the label of x

Union (C(x), C(y)) when C(x) \neq C(y)

m Findings and n unions

m = |E|  n = |V|

What is the max no. of label changes for some vertex x, say \( n(x) \)?

Total cost of n unions \( \leq \sum_{x \in V} n(x) \)
\[ n(x) \leq \log n \]

So total cost of \( m \) Finds and \( n \) Unions is \( O(m + n \log n) \)

\[ O(1E1 + |V|) \]

The total cost of basic greedy for MST is ordering edges by weight \( O(|E| \log |E|) \)

\[ + O(|E| + |V| \log |V|) \]

\[ \text{If } |E| \geq |V| \log |V|, \text{ i.e. (slightly dense graph)} \]

\[ O(|E|) \]

Goal: Alternate Union-Find data structure with improved performance on Unions

FF UF FU UF FF FF FF FF FF FF

Union-Find SPFA will be more general
How do we represent sets?

We will use trees to represent sets.

Base case: singleton vertices (elements), the root has "rank" $= 0$.

Find

$\text{Find}(x)$ move to the root using parent pointers and report the label

$\text{Cost}: \text{length of the path from } x \text{ to root}$

$\text{Union}(T_1, T_2)$

$\text{Cost} O(1)$
Union by rank heuristic

- Make the root with smaller rank the child of the other root (no change in rank)
- Otherwise, choose arbitrarily

**increment the rank of the final root**

---

**Obs**

1. A root node with rank \( r \) has at least \( 2^r \) descendants

(Rank is related to the maxm distance from any leaf node to the root)

Consequence is that Find takes at most \( O(\log n) \) steps
Cost of \( m \) Finds and \( n \) Unions is bounded by \( O(m \log n + n) \)

2. The no. of nodes with rank \( n \) is bounded by \( \frac{n}{2^n} \). Note that once a node ceases to be a root during the course of Union Find, its rank is fixed and never changes in future. (This node never becomes a root node)

3. The ranks increase monotonically in any path from leaf to root node.

Path compression heuristic

\[ O \left( (m + n) \log^* n \right) \]

\[ \log^* n = \min \left\{ i \mid \log \left( \log \left( \ldots \left( \log n \right) \right) \right) \leq 2 \right\} \]