Disjoint Sets 1/25/22

Initial Setting: {0}, {1}, {2}, {3}, {4}, {5}

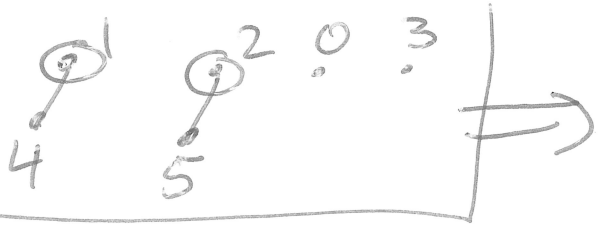
- union (takes 2 sets + puts them into) merges 2 sets into 1
- find (return the "leader" of a set containing a value

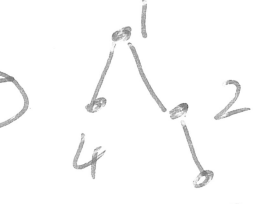
union(1,4)  {1,4} {0} {2} {3} {5}

union(2,5)  {2,5} {1,4} {0} {3}

union(2,4) {1,2,4,5} {0} {3}

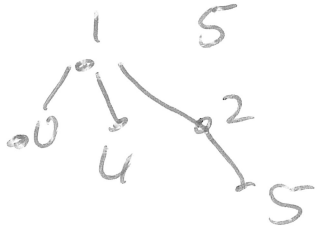
union(1,5) ⇒ NO ACTION





In tree view we attach one root to another.

union(0,1)



runtime find height of tree.

Path Compression will keep height small.

Storage

par | 0 | 1 | 2 | 3 | 4 | 5 |

par[i] stores the parent node of i

Union 1,4 par | 0 | 1 | 2 | 3 | 1 | 5 |
0 1 2 3 4 5

Union 2,5 par | 0 | 1 | 2 | 3 | 1 | 2 |

Union code (a,b)

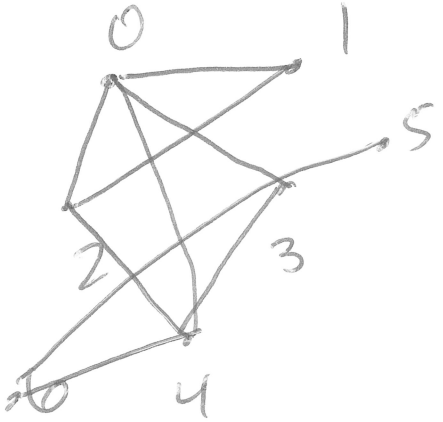
```
a = find(a)
b = find(b)
par[b] = a;
```

Regular find

```
find(int a) {
    if (par[a] == a) return a;
    return find(par[a]);
}
```

par[a] = // Path Compression!

Standard Problem to Solve w/ Disjoint Set



Cities 0 .. n-1
Build roads btw pairs of cities, and at each time slot we want to know the # of component.

Size of each component could be stored in a separate array.

Artwork Key Pts

1. White squares unioned in D.J. set.
2. Run process in reverse, "unpainting" squares.
3. When a square becomes white, we'll union it w/all of its white neighbors.