Lecture 22: Disjoint Sets

Agenda:

- 1^{st} implementation — an array of representatives (review)
- 2^{nd} implementation — forest of rooted trees

Reading:

- Textbook pages 501 – 509

An array of representatives (recall)

- $R(x)$ — the representative of the set containing x
  - $\Theta(n)$ space
  - how to describe a set now:
    elements with the same representative are in a same set
  - report a set in $\Theta(n)$ time

- `procedure MakeSet(`x`) **initialize representative for `x

  $R(x) \leftarrow x$

- `procedure Find(`x`)    **representative of `x

  `return` $R(x)$

- `procedure Union(`x, y`)`

```
rx ← R(x)
ry ← R(y)
for j ← 1 to n do
    if R(j) = ry then
        R(j) ← rx
```

- Running time per operation $\Theta(n)$

2^{nd} implementation — forest of rooted trees

- Forest of rooted trees:
  - elements of a set $\longleftrightarrow$ nodes in the rooted trees
  - representative of a set $\longleftrightarrow$ root of the tree
  - each node needs only 'parent' $\longrightarrow$ implement via an array
  - $P(x)$ — parent of x, for $x = 1, 2, \ldots, n$

- procedure MakeSet(x) **initialize parent for x

  $P(x) \leftarrow x$

- procedure Find(x) **return root of the tree containing x

  while $P(x) \neq x$ do
  $\quad x \leftarrow P(x)$
  return x

- procedure Union(x, y) **make root of x's tree
  **a child of root of y's tree

  $rx \leftarrow$ Find(x)
  $ry \leftarrow$ Find(y)
  $P(rx) \leftarrow ry$ $\qquad$ $P(ry) \leftarrow rx$

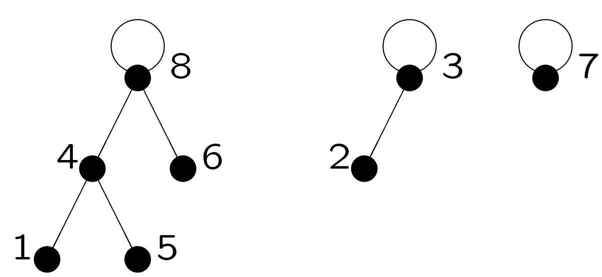
- Running time per operation ???

An example:

sets at start	{1}	{2}	{3}	{4}	{5}	{6}	{7}	{8}
index	1	2	3	4	5	6	7	8
parent	1	2	3	4	5	6	7	8
`Union(1,4)`	**4**	2	3	4	5	6	7	8
`Find(1)`	4							
`Find(4)`				4				
`Union(2,3)`	4	**3**	3	4	5	6	7	8
`Union(5,1)`	4	3	3	4	**4**	6	7	8
`Union(1,8)`	4	3	3	**8**	4	6	7	8
`Union(6,5)`	4	3	3	8	4	**8**	7	8
`Find(6)`						8		
`Find(3)`			3					

sets at finish {1, 4, 5, 6, 8}, {2, 3}, {7}

forest at finish:



Finding connected components of a graph:

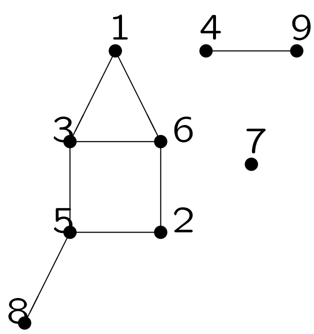
- `procedure ConnectedComponents(`G`)`

```
for each vertex v ∈ V(G) do
    MakeSet(v)
for each edge (x,y) ∈ E(G) do
    if not SameComponent(x,y,G) then
        Union(x,y)
```

- `procedure SameComponent(`x, y, G`)`

```
return Find(x) = Find(y)
```

- An example:



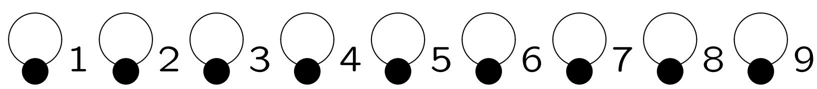
$V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$

$E = \{\{1, 3\}, \{1, 6\}, \{2, 5\}, \{2, 6\}, \{3, 5\}, \{3, 6\}, \{4, 9\}, \{5, 8\}\}$

Finding connected components of a graph:

- Graph $G = (V, E)$:
  $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
  $E = \{\{1, 3\}, \{1, 6\}, \{2, 5\}, \{2, 6\}, \{3, 5\}, \{3, 6\}, \{4, 9\}, \{5, 8\}\}$

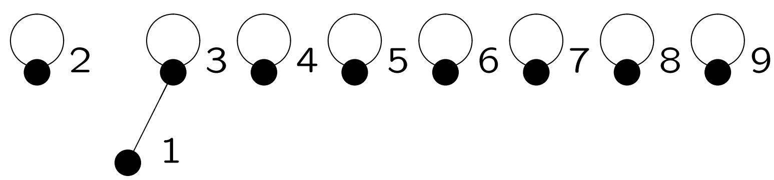
- After `MakeSets`:



Finding connected components of a graph:

- Graph $G = (V, E)$:
  $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
  $E = \{\{1, 3\}, \{1, 6\}, \{2, 5\}, \{2, 6\}, \{3, 5\}, \{3, 6\}, \{4, 9\}, \{5, 8\}\}$

- After considering edge $(1, 3)$:



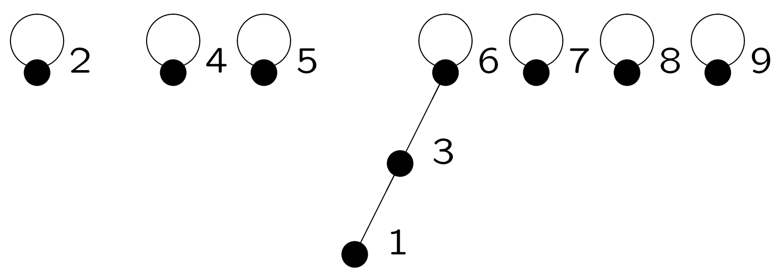
Finding connected components of a graph:

- Graph $G = (V, E)$:

  $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$

  $E = \{\{1, 3\}, \{1, 6\}, \{2, 5\}, \{2, 6\}, \{3, 5\}, \{3, 6\}, \{4, 9\}, \{5, 8\}\}$

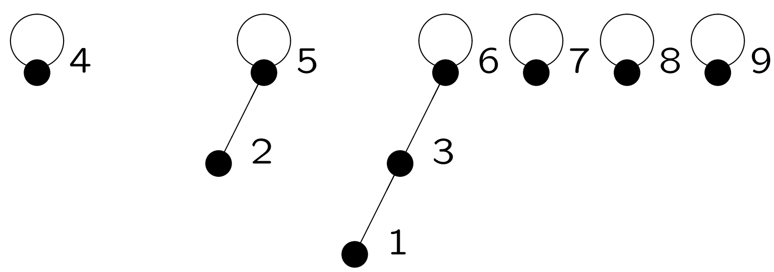
- After considering edge $(1, 6)$:



Finding connected components of a graph:

- Graph $G = (V, E)$:
  $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
  $E = \{\{1, 3\}, \{1, 6\}, \{2, 5\}, \{2, 6\}, \{3, 5\}, \{3, 6\}, \{4, 9\}, \{5, 8\}\}$

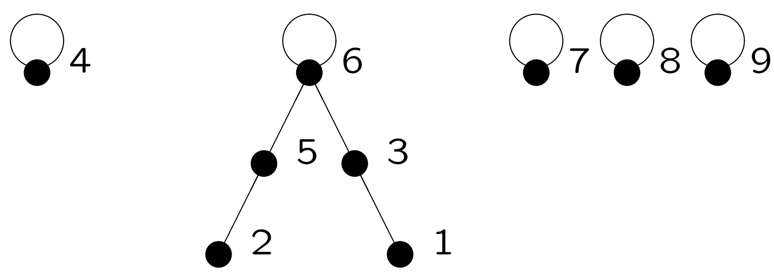
- After considering edge $(2, 5)$:



Finding connected components of a graph:

- Graph $G = (V, E)$:
  $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
  $E = \{\{1, 3\}, \{1, 6\}, \{2, 5\}, \{2, 6\}, \{3, 5\}, \{3, 6\}, \{4, 9\}, \{5, 8\}\}$

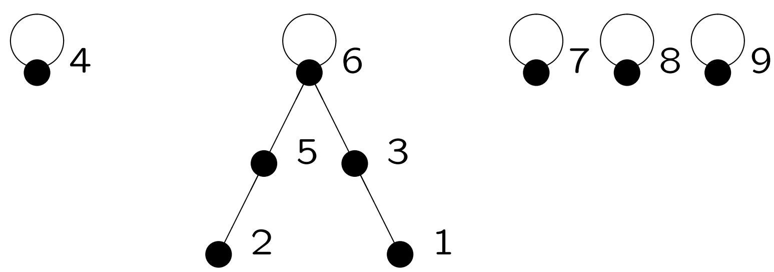
- After considering edge $(2, 6)$:



Finding connected components of a graph:

- Graph $G = (V, E)$:
  $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
  $E = \{\{1, 3\}, \{1, 6\}, \{2, 5\}, \{2, 6\}, \{3, 5\}, \{3, 6\}, \{4, 9\}, \{5, 8\}\}$

- After considering edges $(3, 5)$ and $(3, 6)$



(no change since $3, 5, 6$ are already in a same component):

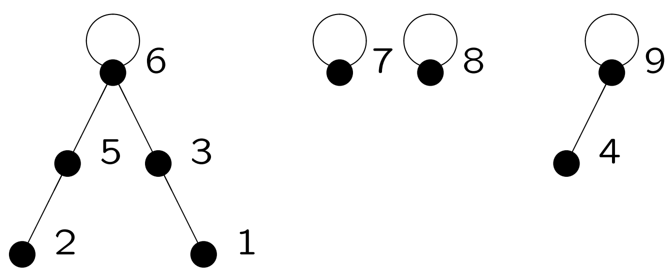
Finding connected components of a graph:

- Graph $G = (V, E)$:

  $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$

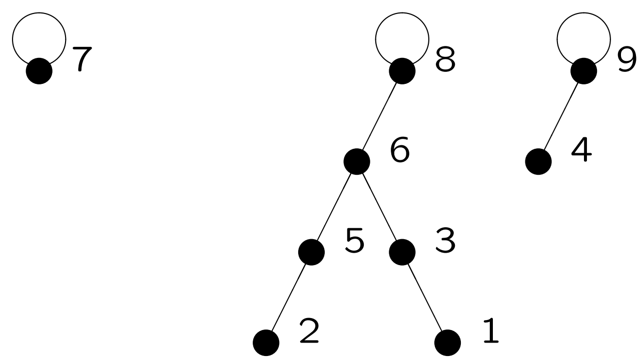
  $E = \{\{1, 3\}, \{1, 6\}, \{2, 5\}, \{2, 6\}, \{3, 5\}, \{3, 6\}, \{4, 9\}, \{5, 8\}\}$

- After considering edge $(4, 9)$:

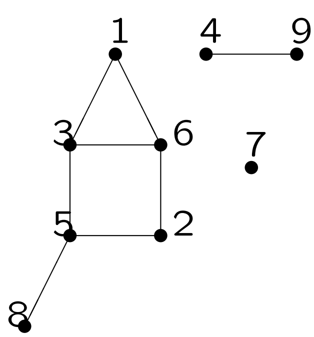


Finding connected components of a graph:

- Graph $G = (V, E)$:
  $V = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$
  $E = \{\{1, 3\}, \{1, 6\}, \{2, 5\}, \{2, 6\}, \{3, 5\}, \{3, 6\}, \{4, 9\}, \{5, 8\}\}$
- After considering edge $(5, 8)$:



- Sets at finish:
  {1, 2, 3, 5, 6, 8} with representative 8
  {4, 9} with representative 9
  {7} with representative 7
- Therefore, there are 3 connected components



Analysis of the implementation:

- n `MakeSet` and $(m - n)$ `Union/Find`
- Each `MakeSet` — $\Theta(1)$ time
- Each `Find` — $\Theta(\text{depth}(x))$ time

  since we need to get to the root of the tree containing element x
- Each `Union` — $\Theta(\text{depth}(x) + \text{depth}(y))$ time

  since we need to find the representatives for x and y and then use constant time to update
- Worst case:

  $\Theta(n + (m - n)n) = \Theta(mn)$ time (assuming $m >> n$)
- On average, $\Theta(n)$ per operation

  amortized running time analysis

Conclusion: Forest of rooted trees is NOT better than Array of representatives

Yet it allows speedup, just adding some tricks ...

(next lecture)

Have you understood the lecture contents?

well	ok	not-at-all	topic
☐	☐	☐	disjoint sets?
☐	☐	☐	3 operations
☐	☐	☐	forest of rooted trees
☐	☐	☐	finding connected components
☐	☐	☐	running time analysis