Union-Find Algorithm

Sedgewick and Wayne, Section 1.5
Task

• Given a graph, there are 2 operations:
  – Union(v,w)  add an edge to connect 2 vertices
  – Find(v,w): is there a path between v and w?
    • Really should be called isConnected, because it only detects the existence of a path; it doesn’t find it
Applications

• Minimal spanning tree
  – Given a graph, what is the smallest subset of edges that connect the graph?
    • Example: phone lines

• Graph connectivity
  – Given a graph, is there a path between $v$ and $w$?
    • Example: is there a network connection between one computer and another?
Approach #1: Quick-find

- Maintain an array $g$ of size $|V|$ (number of vertices in graph); $g[v] = \text{group number for vertex } v$. $g[v]$ represents $v$’s “group”
- Initially, $g = \{0, 1, 2, \ldots\}$
- Each time we connect $v$ and $w$, change all entries for $v$’s group to $w$’s group
Example

• Initial: [0, 1, 2, 3, 4, 5, 6, 7]
• Union(0,1)
  – [1, 1, 2, 3, 4, 5, 6, 7]
• Union(0,3)
  – [3, 3, 2, 3, 4, 5, 6, 7]
• isConnected(0,3) => true
  – (because g[0] == g[3])
• Union(4,5)
  – [3, 3, 2, 3, 5, 5, 6, 7]
Example, cont.

- [ 3, 3, 2, 3, 5, 5, 6, 7 ]
  - Union(0,4)
    - [ 5, 5, 2, 3, 5, 5, 6, 7 ]
  - Union(2,3)
    - [ 5, 5, 3, 3, 5, 5, 6, 7 ]
  - Union(6, 7)
    - [ 5, 5, 3, 3, 5, 5, 7, 7 ]
- isConnected(0,7) => false
- Union(0,7)
  - [ 7, 7, 3, 3, 7, 7, 7, 7 ]
Implementation

- **Union(v,w)**: change $g[i]$ to $g[v]$ for all $i$ where $g[i] == g[w]$
- **isConnected(v,w)**: $g[v] == g[w]$
- **Complexity:**
  - Union
  - isConnected
Approach 2: “Quick” union

- Maintain an array of indices of “parents”
- If $v$ has no parent ($v$ is root), $\text{parent}[v] == v$
- $\text{union}(v,w)$: $\text{parent}[\text{root}(w)] = v$
- $\text{isConnected}(v,w)$: $\text{root}(v) == \text{root}(w)$
Example

• Initial: [ 0, 1, 2, 3, 4, 5, 6, 7]
• Union(0,1)
  – [ 0, 0, 2, 3, 4, 5, 6, 7 ]
• Union(3,1)
  – [ 3, 0, 2, 3, 4, 5, 6, 7 ]
• isConnected(1,3) => true
  – because root(0) == root(3) == 3
• Union(4,5)
  – [ 3, 0, 2, 3, 4, 4, 6, 7 ]
Example, cont.

- \([ 3, 0, 2, 3, 4, 4, 6, 7 ]\)

- **Union(4,1)**
  - \([3, 0, 2, 4, 4, 4, 6, 7 ]\)
- **isConnected(0,5) => true because**
  - root(0) == root(5) == 4

- **Union(1,2)**
  - \([ 3, 0, 1, 4, 4, 4, 6, 7 ]\)
Example, cont.

- \[ 3, 0, 1, 4, 4, 4, 6, 7 \]

- **Union(6, 7)**
  - \[ 3, 0, 1, 4, 4, 4, 6, 6 \]

- **isConnected(0,7) => false**
  - \( \text{root}(0) \Rightarrow 4 \quad \text{root}(7) \Rightarrow 6 \)

- **Union(0,7)**
  - \[ 3, 0, 1, 4, 4, 4, 0, 6 \]
“Quick” union

• Complexity:
  – Union
  – isConnected
Weighted Quick Union

• Maintain height of each root
• Union\(v, w\): parent[root(w)] = v OR parent[root(v)] = w, depending on heights
• Height of root is incremented when necessary
Example

• Initial: [0, 1, 2, 3, 4, 5, 6, 7]
• Union(0,1)
  – [0, 0, 2, 3, 4, 5, 6, 7]
• Union(3,1)
  – [0, 0, 2, 0, 4, 5, 6, 7] because h(3) == 0, h(0) == 1
• isConnected(1,3) => true
  – because root(0) == root(3) == 0
• Union(4,5)
  – [0, 0, 2, 0, 4, 4, 6, 7]
Example, cont.

- [0, 0, 2, 0, 4, 4, 6, 7]

- **Union(4,1)**
  - [4, 0, 2, 0, 4, 4, 6, 7] because \( h(0) == h(4) \)
    - choose either root. \( h(4) \) is now 2

- **isConnected(0,5) => true** because
  - root(0) == root(5) == 4

- **Union(1,2)**
  - [4, 0, 4, 0, 4, 4, 6, 7] because \( h(2) < h(4) \)
Example, cont.

- \([ 4, 0, 4, 0, 4, 4, 6, 7 ]\)

**Union(6, 7)**

- \([ 4, 0, 4, 0, 4, 4, 6, 6 ]\)

**isConnected(0,7) => false**

- \(\text{root}(0) \to 4 \quad \text{root}(7) \to 6\)

**Union(0,7)**

- \([ 4, 0, 4, 4, 4, 4, 4, 6 ]\) because \(h(6) < h(4)\)
Weighted Quick Union

• Complexity:
  – Union
  – isConnected

• Must consider the minimum number of nodes $n$ in a tree of height $h$?
• $h == 0$: just the root
• $h == 1$: root with 1 child
• $h == 2$: merge 2 trees of height 1 (result?)
• $h == 3$?
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Union</th>
<th>isConnected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quick find</td>
<td>Theta(n)</td>
<td>Theta(1)</td>
</tr>
<tr>
<td>“Quick” union</td>
<td>Theta(n)</td>
<td>Theta(n)</td>
</tr>
<tr>
<td>Weighted quick union</td>
<td>Theta(log n)</td>
<td>Theta(log n)</td>
</tr>
</tbody>
</table>