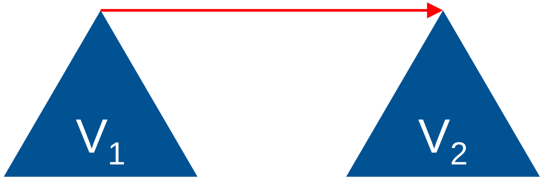
Union by Rank and Path Compression Heuristics

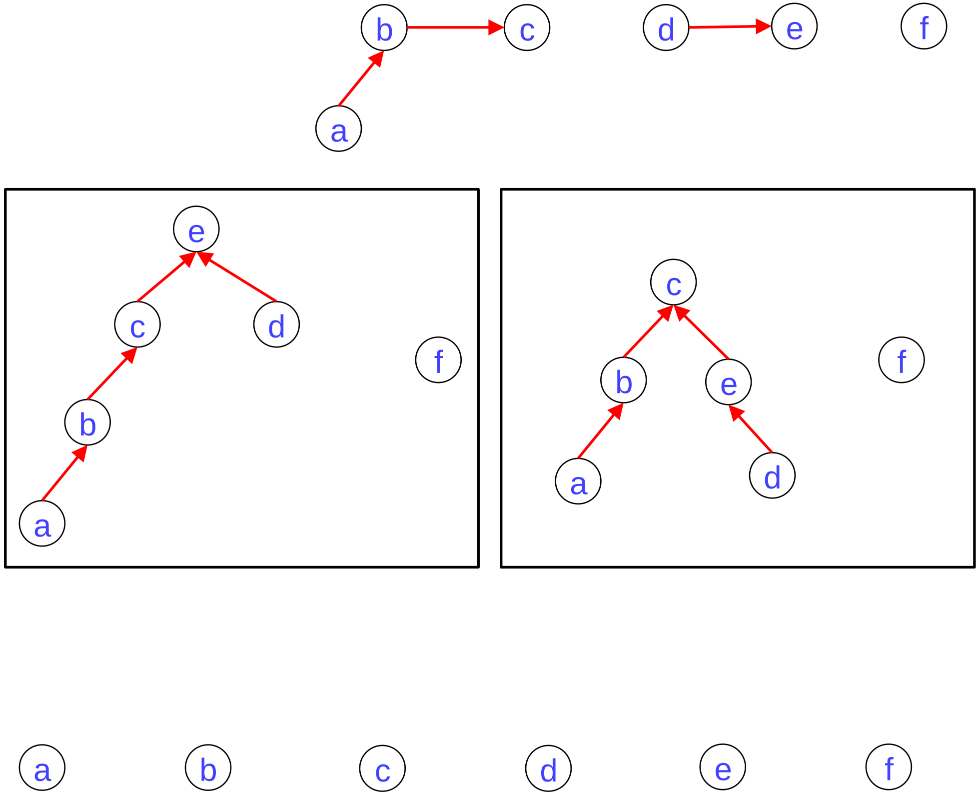
Dr. Shrutilipi Bhattacharjee, Assistant Professor, Dept. of IT, NIT Karnataka, India

Union by Rank

- What are the options to point one tree to the other?
- If we have two trees with number of vertices V_1 and V_2, then we will make the lighter tree point to the heavier one
- Lets say, $V_1 < V_2$

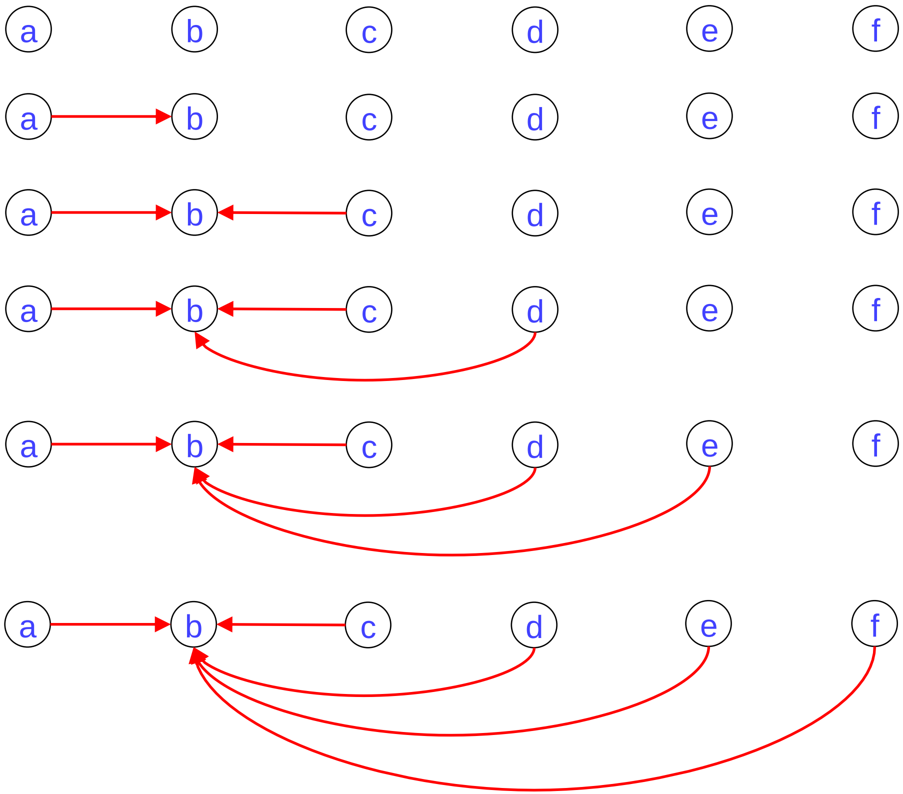


- Then what will happen to our previous example?



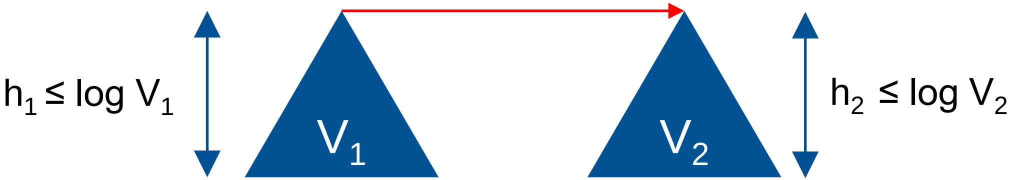
Union by Rank

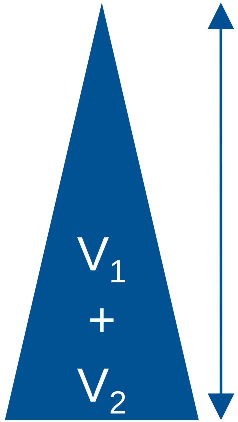
- Then what will happen to our previous example?
- What is the height of this tree?
  - 1 only
  - **find**() will take very little time
- So now we have to see that if we use this rule, what can be the height of the tree in the worst case?
- How high the tree become?
- **Claim:** A tree with V_1 vertices has height less than or equal to *log* V_1



Union by Rank

- **Union by Number of Vertices**
- **Claim:** A tree with V_1 vertices has height less than or equal to $\log V_1$
- Lets say, $V_1 < V_2$
- Let us assume that the induction hypothesis is true till this stage of our procedure



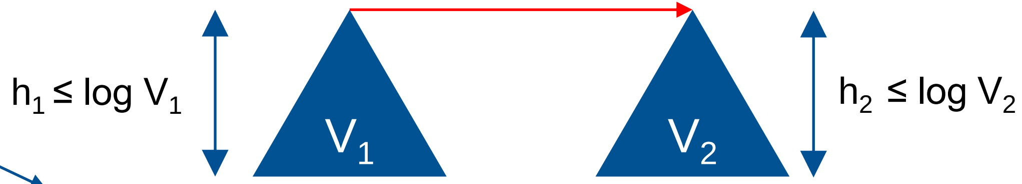


- As a consequence, we have to show that the resultant tree has height **$H \leq \log (V_1 + V_2)$**

$H \leq \log (V_1 + V_2)$

Union by Rank

- Height of the resultant tree: $\mathbf{H = max(h_1 + 1, h_2)}$



$$h_2 \le \log V_2 \le \log (V_1 + V_2)$$

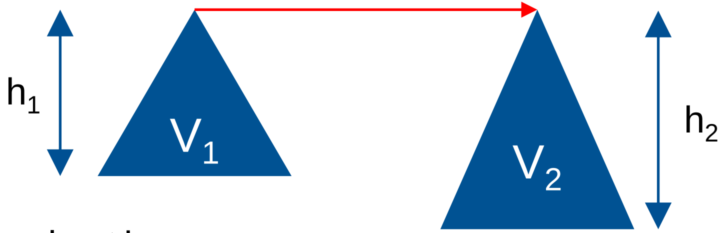
$$\begin{aligned} h_1 + 1 &\le (\log V_1) + 1 \\ &= \log (2V_1) \\ &\le \log (V_1 + V_2) \text{ as } V_1 < V_2 \end{aligned}$$

- Is the base case true?

- The whole process is called ***Union by Rank***
- Rank is the number of vertices in the tree

Union by Rank

- **Union by Height**



- Lets say, $h_1 \le h_2$
- If we have two trees with heights h_1 and h_2 ($h_1 \le h_2$), then we will make the root of the shallow tree point to the root of the taller one
- **Claim:** A tree with height h has atleast 2^h vertices
  - Height of the resultant tree: $\mathbf{h = max(h_1 + 1, h_2)}$
  - So, the number of vertices in the new tree: $\mathbf{V_{12} = V_1 + V_2}$
  - Now, $V_1 \ge 2^{h1}$ *and* $V_2 \ge 2^{h2}$
  - Then, $V_{12} \ge 2^{h1} + 2^{h2} \ge 2^{h2}$

    $\ge 2^{h1} + 2^{h1}$ (since $h_1 \le h_2$) $= 2^{h1+1}$

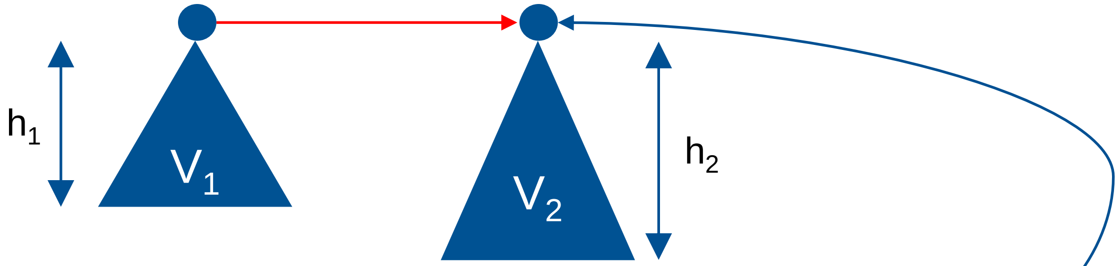
Number of vertices in the new tree $\ge \max(2^{h1+1}, 2^{h2})$

$= 2^{\max(h1+1,\, h2)}$

$= 2^h$

Union() & Find()

- How much time does the **union**() take?
  - The root vertex will either keep track of the height of the tree or the number of vertices in it



  - When **union**() happens, in the root of the heavier/taller tree:
    - Either you update the height as: $\mathbf{max(h_1 + 1, h_2)}$
    - Or, update the number of vertices as: $\mathbf{V_1 + V_2}$
  - The **union**() takes O(1) time
- How much time does the **find**() take?
  - The **find**() takes O(log V) time

<u>Time Complexity of Kruskal's Algorithm</u>

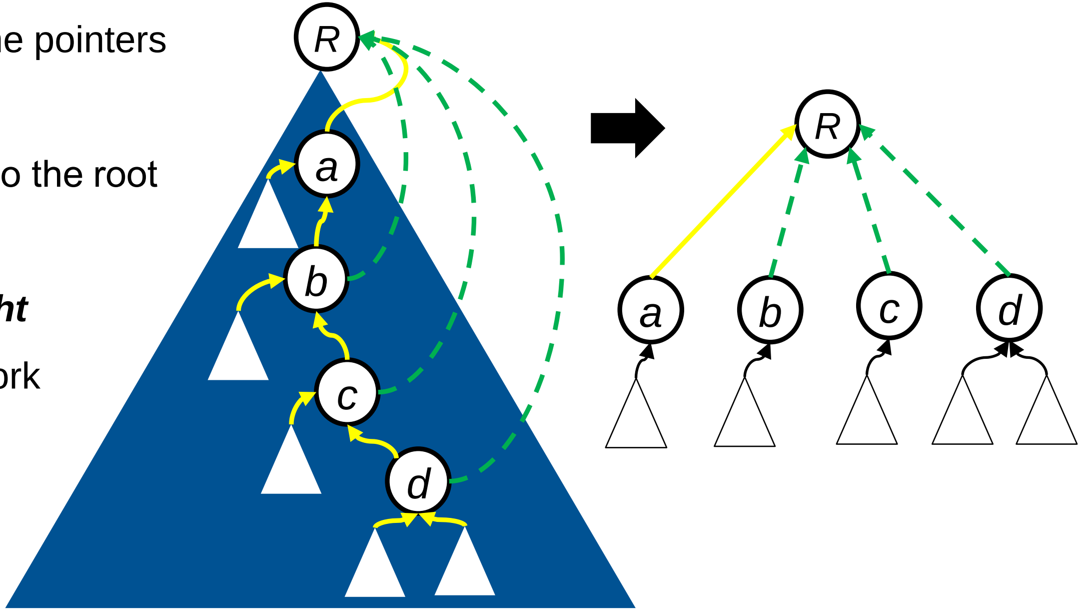
- $O(E \log E + U.\ V + F\ .\ E)$

  $= (E \log E + V + E \log V)$

  $= (E \log V)$ given, $\log V \leq \log E \leq 2\log V$

Path Compression

- Improve the time required for **find**()
- Can we do something at this point, so as to improve the performance of future **find**()s?
  - Because we might have to traverse the same path again in the future
  - We will do some modifications with the pointers
  - We will directly point them to the root
  - That's how vertices are being closer to the root
- For the ***path compression*** technique
  - We can not work with ***union by height***
  - ***Union by number of vertices*** will work



Note:

The ***union-find*** data structure is NOT a standalone entity**.** We need to keep a cross-reference from the vertex in the adjutancy list data structure

Next Lecture

Prim's Algorithm

Thank you for your attention...

Any question?

Contact:
Department of Information Technology, NITK Surathkal, India
6th Floor, Room: 13
Phone: +91-9477678768
E-mail: shrutilipi@nitk.edu.in, shrutilipi.bhattacharjee@tum.de