CSL 356: Analysis and Design of Algorithms

Ragesh Jaiswal

CSE, IIT Delhi

Greedy Algorithms: Example

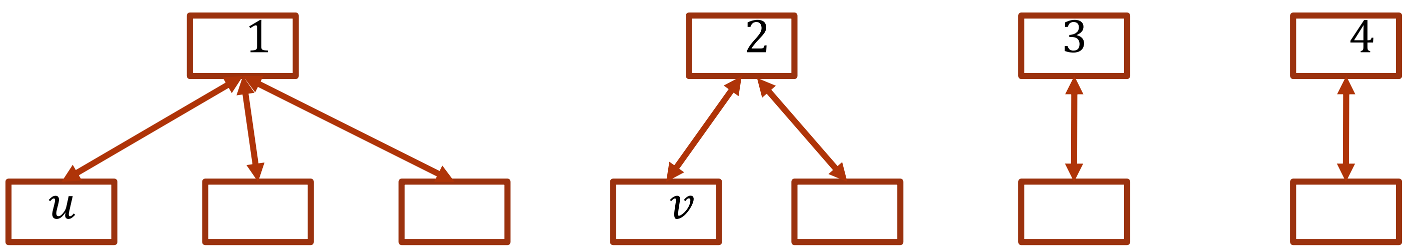
Kruskal's Algorithm(G)
- $S = E; T = \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T = T \cup \{e\}$
  - $S = S - \{e\}$

Kruskal's Algorithm(G)
- $S = E; T = \{\}$
- While the edge set T does not connect all the vertices
  - *//Note that* $G' = (V, T)$ *contains disconnected components*
  - Let $e = (u, v)$ be the minimum weight edge in the set S
  - If u and v are in different components
    - $T = T \cup \{e\}$
  - $S = S - \{e\}$

Greedy Algorithms: Example

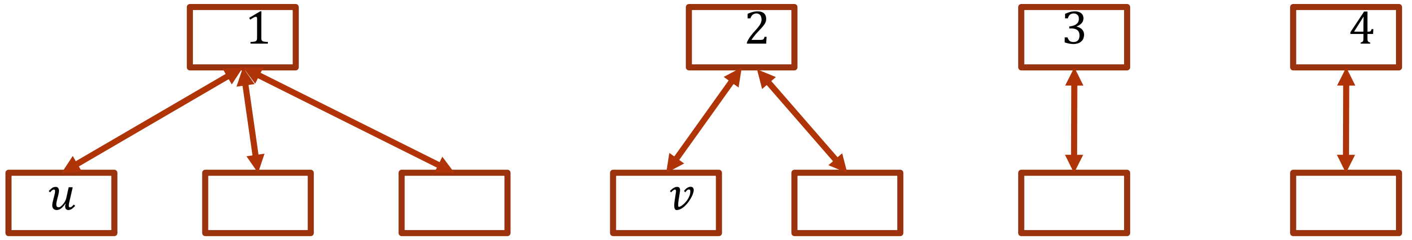
- Union-Find: Used for storing partition of a set of elements. The following two operations are supported.
  - **Find(v)**: Find the partition to which the element v belongs
  - **Union(u, v)**: Merge the partition to which u belongs with the partition to which v belongs.

- Consider the following data structure.



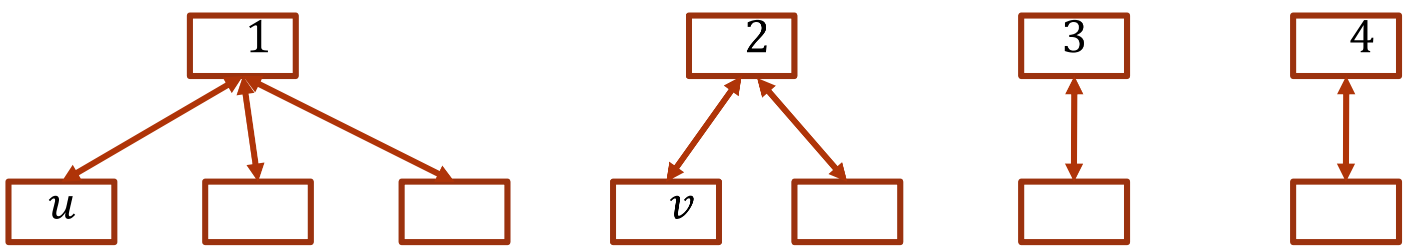
Greedy Algorithms: Example

- Suppose we start from a full partition (each partition contain one element). How much time do the following operations take:
  - **Find($\boldsymbol{v}$)**:
  - **Union($\boldsymbol{u}, \boldsymbol{v}$)**:



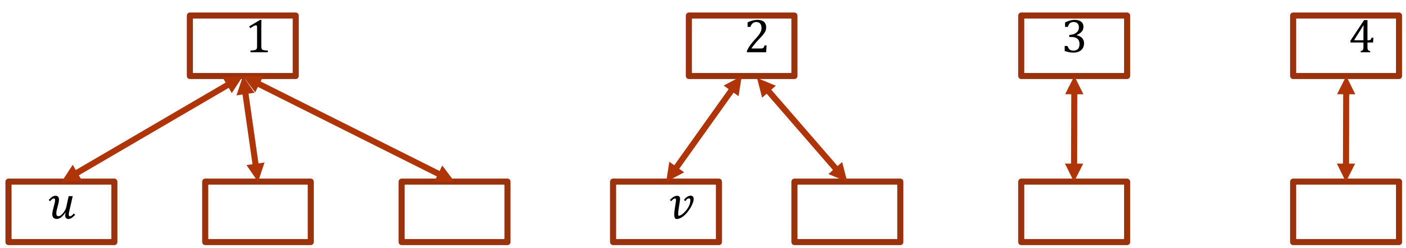
Greedy Algorithms: Example

- Suppose we start from a full partition (each partition contain one element). How much time do the following operations take:
  - **Find($\boldsymbol{v}$)**: $O(1)$
  - **Union($\boldsymbol{u}, \boldsymbol{v}$)**:



Greedy Algorithms: Example

- Suppose we start from a full partition (each partition contain one element). How much time do the following operations take:
  - **Find($\boldsymbol{v}$)**: $O(1)$
  - **Union($\boldsymbol{u, v}$)**:
    - <u>Claim</u>: Performing k union operations takes $O(k \log k)$ time in the worst case.



Greedy Algorithms: Example

- Kruskal using union-find:

Kruskal's Algorithm(G)
- $S = E; T = \{\}$
- While the edge set T does not connect all the vertices
  - *//Note that* $G' = (V, T)$ *contains disconnected components*
  - Let $e = (u, v)$ be the minimum weight edge in the set S
  - If ($\text{Find}(u) \neq \text{Find}(v)$)
    - $T = T \cup \{e\}$
    - Union(u, v)
  - $S = S - \{e\}$

- What is the running time?

Greedy Algorithms: Example

- Kruskal using union-find:

Kruskal's Algorithm(G)
- $S = E; T = \{\}$
- While the edge set T does not connect all the vertices
  - *//Note that* $G' = (V, T)$ *contains disconnected components*
  - Let $e = (u, v)$ be the minimum weight edge in the set S
  - If ($\text{Find}(u) \neq \text{Find}(v)$)
    - $T = T \cup \{e\}$
    - Union(u, v)
  - $S = S - \{e\}$

- What is the running time?
  - $O(|E| \cdot \log |V|)$

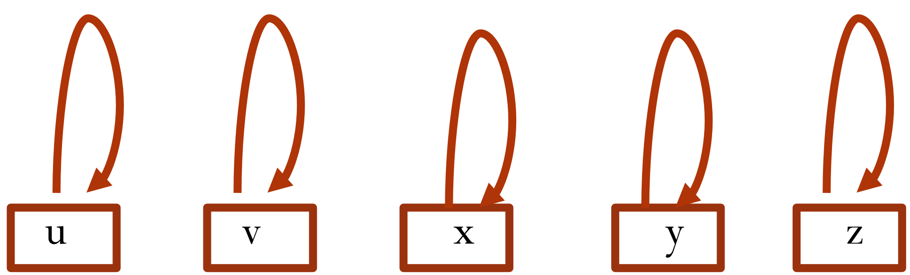
Digression: Union-Find

Digression: Union-Find

- Suppose some application requires Union operation to be quicker perhaps at the cost of increasing the running time of Find. Can you think of a data structure?

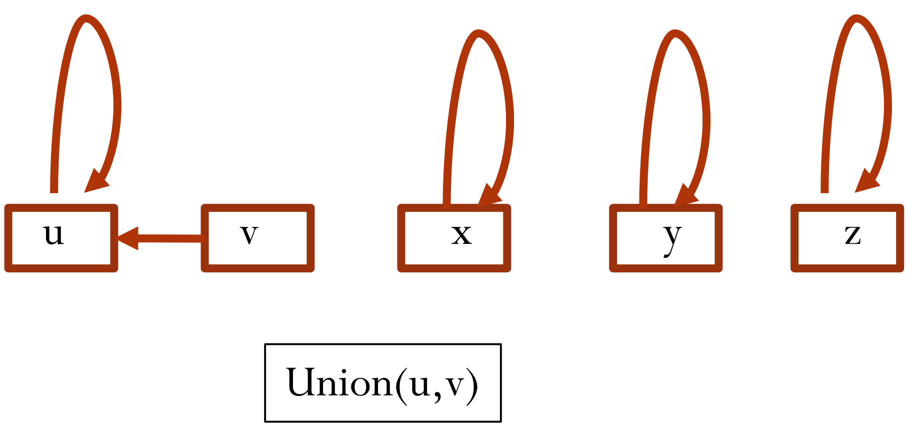
Digression: Union-Find

- Suppose some application requires Union operation to be quicker perhaps at the cost of increasing the running time of Find. Can you think of a data structure?
- Suppose we associate a partition with one of its elements. Think of this element as the representative of this partition.
- If we start from full partition. (each partition contains one element.)



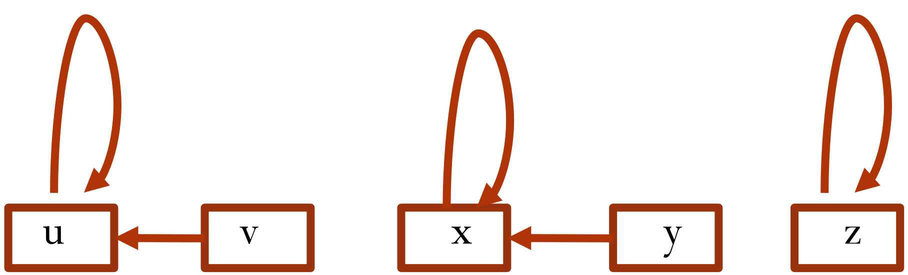
Digression: Union-Find

- Suppose some application requires Union operation to be quicker perhaps at the cost of increasing the running time of Find. Can you think of a data structure?
- Suppose we associate a partition with one of its elements. Think of this element as the representative of this partition.
- If we start from full partition. (each partition contains one element.)



Digression: Union-Find

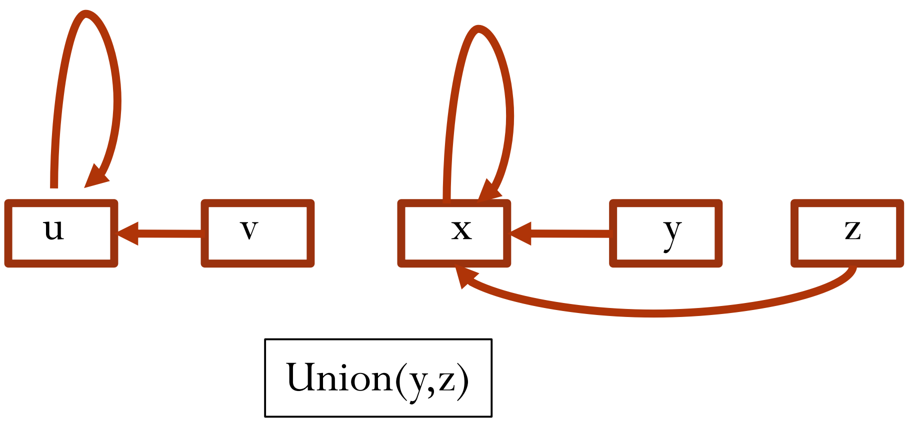
- Suppose some application requires Union operation to be quicker perhaps at the cost of increasing the running time of Find. Can you think of a data structure?
- Suppose we associate a partition with one of its elements. Think of this element as the representative of this partition.
- If we start from full partition. (each partition contains one element.)



Union(x,y)

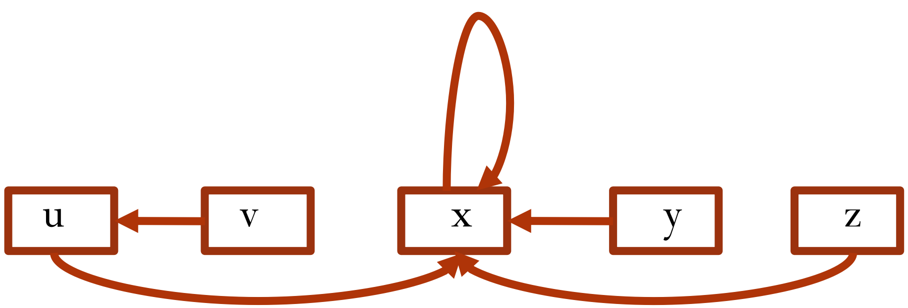
Digression: Union-Find

- Suppose some application requires Union operation to be quicker perhaps at the cost of increasing the running time of Find. Can you think of a data structure?
- Suppose we associate a partition with one of its elements. Think of this element as the representative of this partition.
- If we start from full partition. (each partition contains one element.)



Digression: Union-Find

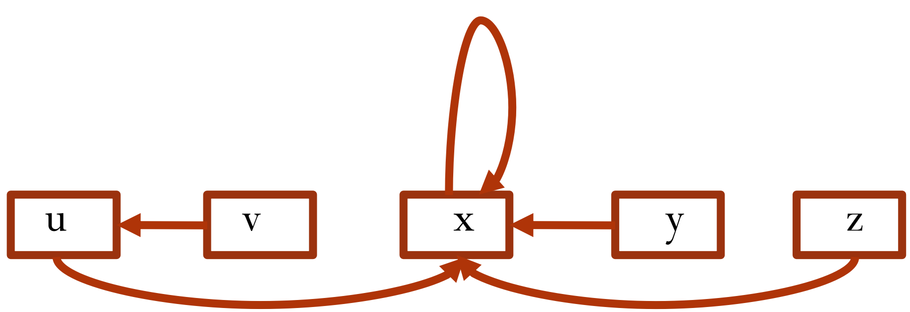
- Suppose some application requires Union operation to be quicker perhaps at the cost of increasing the running time of Find. Can you think of a data structure?
- Suppose we associate a partition with one of its elements. Think of this element as the representative of this partition.
- If we start from full partition. (each partition contains one element.)



Union(v,y)

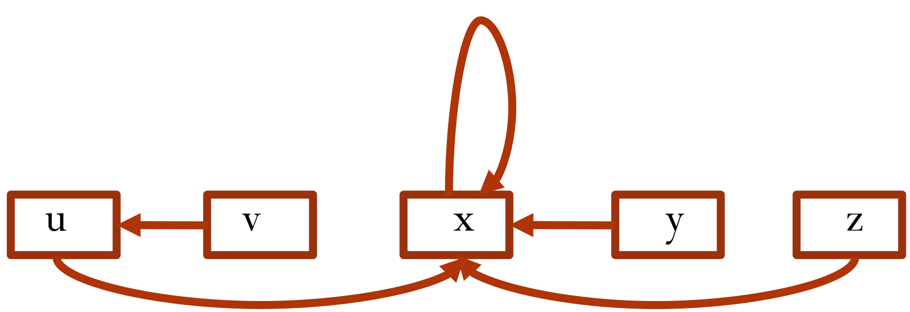
Digression: Union-Find

- Suppose we start from a full partition (each partition contain one element). How much time does the following take:
  - <u>Union</u>:
  - <u>Find</u>:



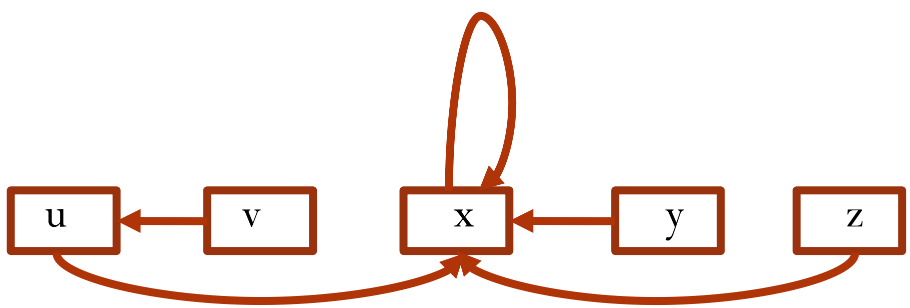
Digression: Union-Find

- Suppose we start from a full partition (each partition contain one element). How much time does the following take:
  - Union: whatever find takes + $O(1)$
  - Find:



Digression: Union-Find

- Suppose we start from a full partition (each partition contain one element). How much time does the following take:
  - Union: whatever find takes + $O(1)$
  - Find: $O(\log n)$



End

Problems to think about:

1. Give an algorithm to find a minimum spanning tree of a weighted graph with minimum multiplicative cost. Multiplicative cost of a spanning tree is the product of edge weights.