Graph Algorithms

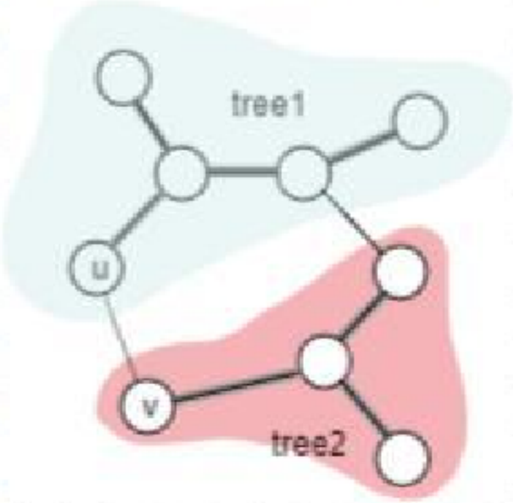
CS3104

Dr. Samit Biswas, *Assistant Professor,*
Department of Computer Sc. and Technology,
Indian Institute of Engineering Science and Technology, Shibpur

Email: samit@cs.iiests.ac.in

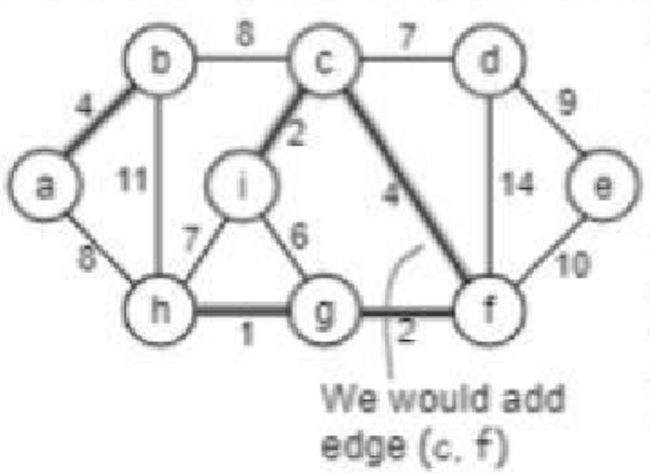
Kruskal's Algorithm

- How is it different from Prim's algorithm?
  - Prim's algorithm grows one tree all the time
  - Kruskal's algorithm grows multiple trees (i.e., a forest) at the same time.
  - Trees are merged together using **safe** edges
  - Since an MST has exactly |V| - 1 edges, after |V| - 1 merges, we would have only one component

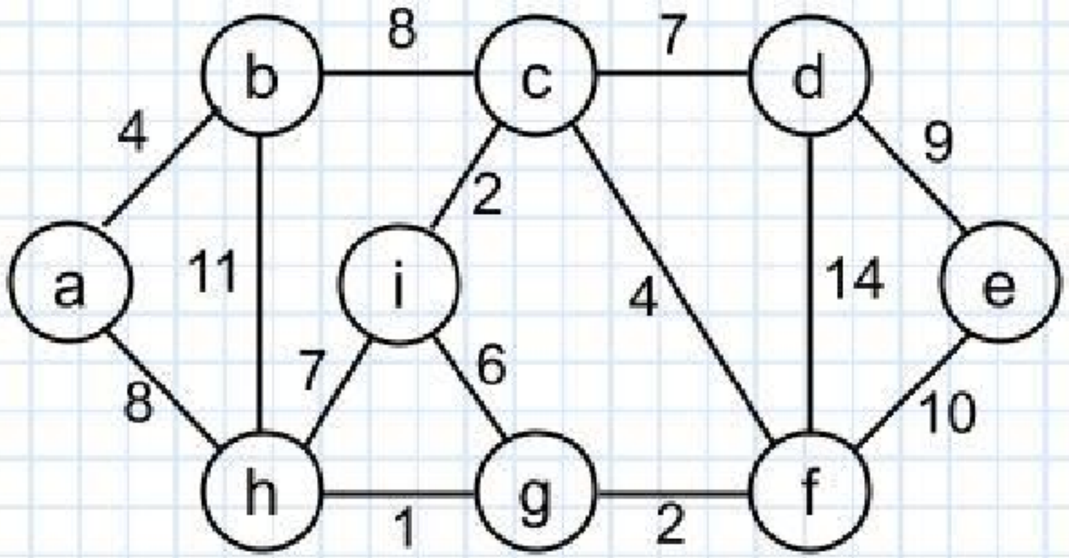


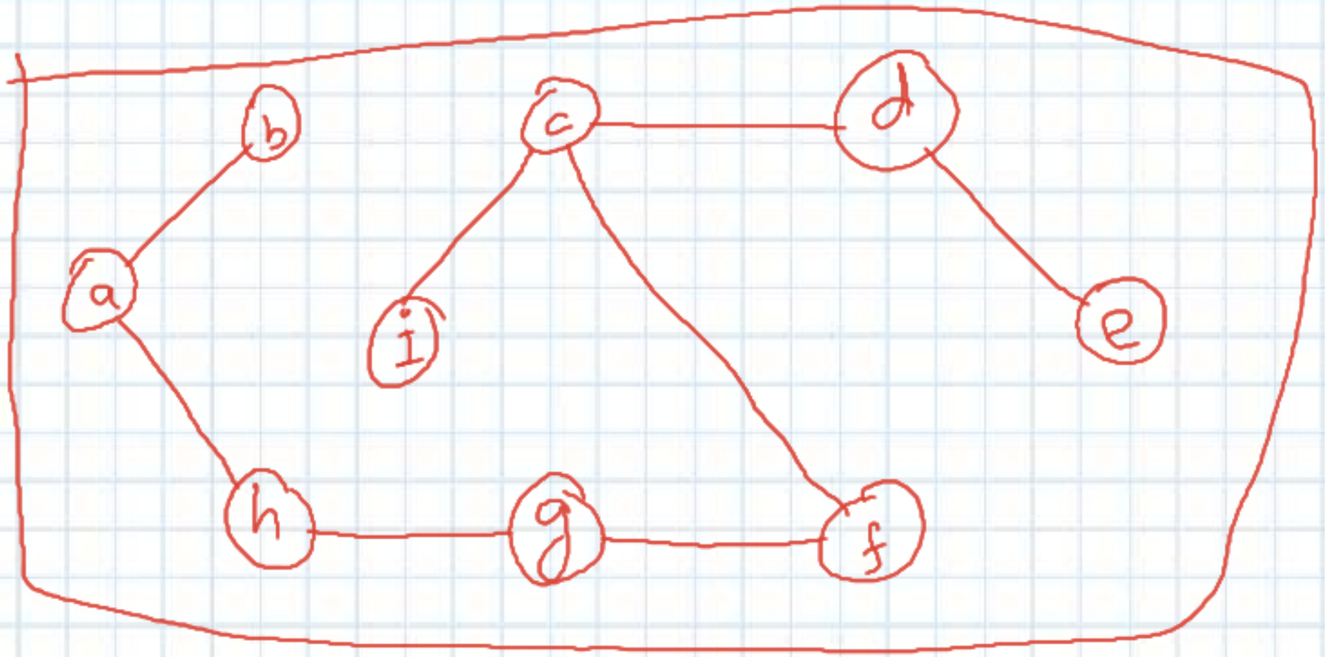
Kruskal's Algorithm

- Start with each vertex being its own component
- Repeatedly merge two components into one by choosing the **light** edge that connects them
- Which components to consider at each iteration?
  - Scan the set of edges in monotonically increasing order by weight



Example: Kruskal's Algorithm

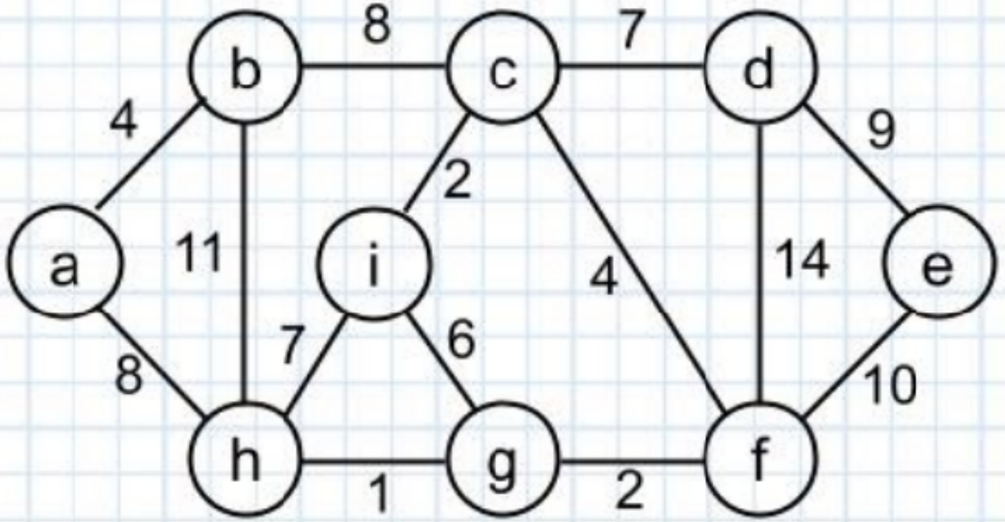




1. (h, g)
2. (c, i), (g, f)
4. (a, b), (c, f)
6. (i, g)
7. (c, d), ~~(i, h)~~
8. (a, h), ~~(b, c)~~
9. (d, e)
10. (e, f)
11. ~~(b, h)~~
14. ~~(d, f)~~

{a}{b}, {c}, {d}{e}{f}

{g}, {h}, {i}



1. (h, g)
2. (c, i), (g, f)
4. (a, b), (c, f)
6. (i, g)
7. (c, d), (i, h)
8. (a, h), (b, c)
9. (d, e)
10. (e, f)
11. (b, h)
14. (d, f)

{a} {b}, {c}, {d} {e} {f}
{g}, {h}, {i}

initial:	{a} {b} {c} {d} {e} {f} {g} {h} {i}
Add {h, g}	{g, h} {a}, {b}, {c} {d}, {e}, {f} {i}
Add {c, i}	{g, h}, {c, i}, {a}, {b}, {d}, {e}, {f}
Add {g, f}	{g, h, f}, {c, i}, {a}, {b}, {d}, {e}
Add {a, b}	{g, h, f}, {c, i}, {a, b}, {d}, {e}
Add {f, i}	{g, h, f, c, i}, {a, b}, {d}, {e}
Ignore {i, g}	{g, h, f, c, i}, {a, b}, {d} {e}

Implementation of Kruskal's Algorithm

- Uses a disjoint-set data structure to determine whether an edge connects vertices in different components
- **Disjoint Set Operations**
  - **MAKE-SET(x)**: creates a new set whose only member (and thus representative) is x.
  - **UNION(x, y)**: unites the dynamic sets that contain x and y, say S_x and S_y, into a new set that is the union of these two sets. We assume that the two sets are disjoint prior to the operation.
  - **FIND-SET(x)**: returns a pointer to the representative of the (unique) set containing x.

Kruskal's Algorithm

Algorithm KRUSKAL(V, E, w)

1. $A \leftarrow \varnothing$
2. **for** each vertex $v \in V$
3. **do** MAKE-SET(v)
4. sort E into non-decreasing order by w
5. **for** each (u, v) taken from the sorted list
6. **do if** FIND-SET(u) ≠ FIND-SET(v)
7. **then** $A \leftarrow A \cup \{(u, v)\}$
8. UNION(u, v)
9. **return** A

$O(E \log E)$

$O(\log V)$

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