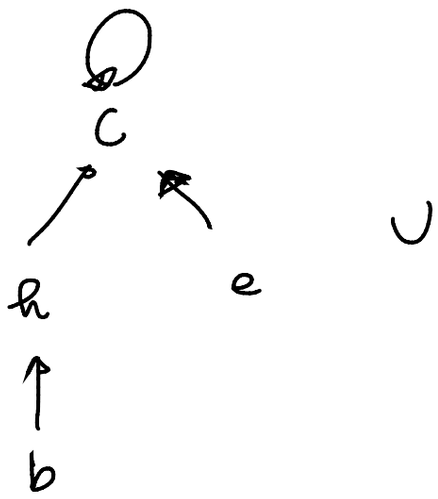
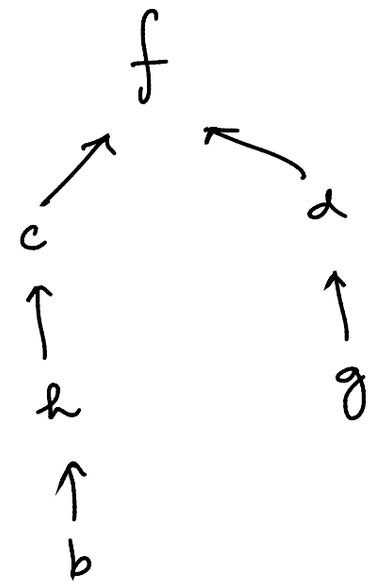
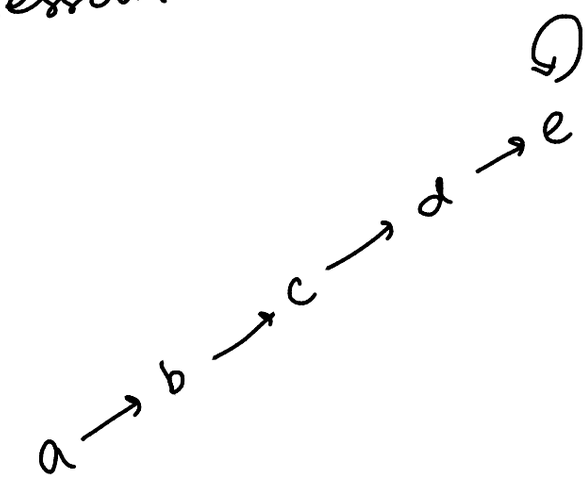
Can also use a tree

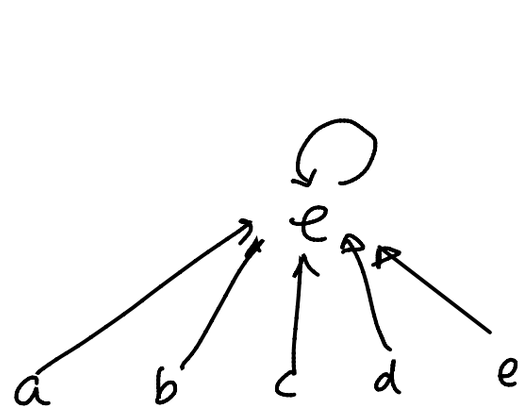


∪



Path compression:



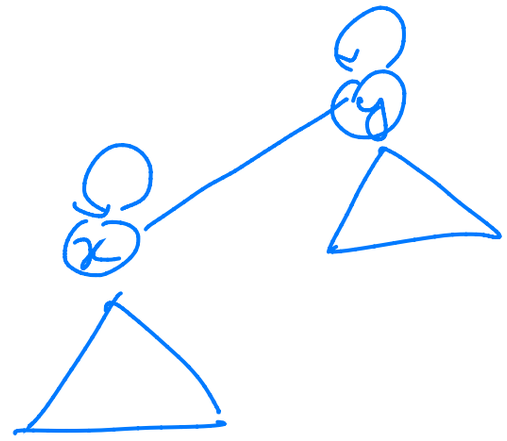


Rank: Maintains a heuristic upper bound on the height

MAKE-SET(x)

$p[x] \leftarrow x$

$rank[x] \leftarrow 0$



UNION-SET(x, y)

LINK (FIND-SET(x), FIND-SET(y))

pointer to representative

LINK (x, y)

If $rank[x] > rank[y]$

then $p[y] \leftarrow x$

(strict greater than case)

(we make the "shorter" tree to be a child of the "taller" tree)

else $p[x] \leftarrow y$

($rank[x] \leq rank[y]$)

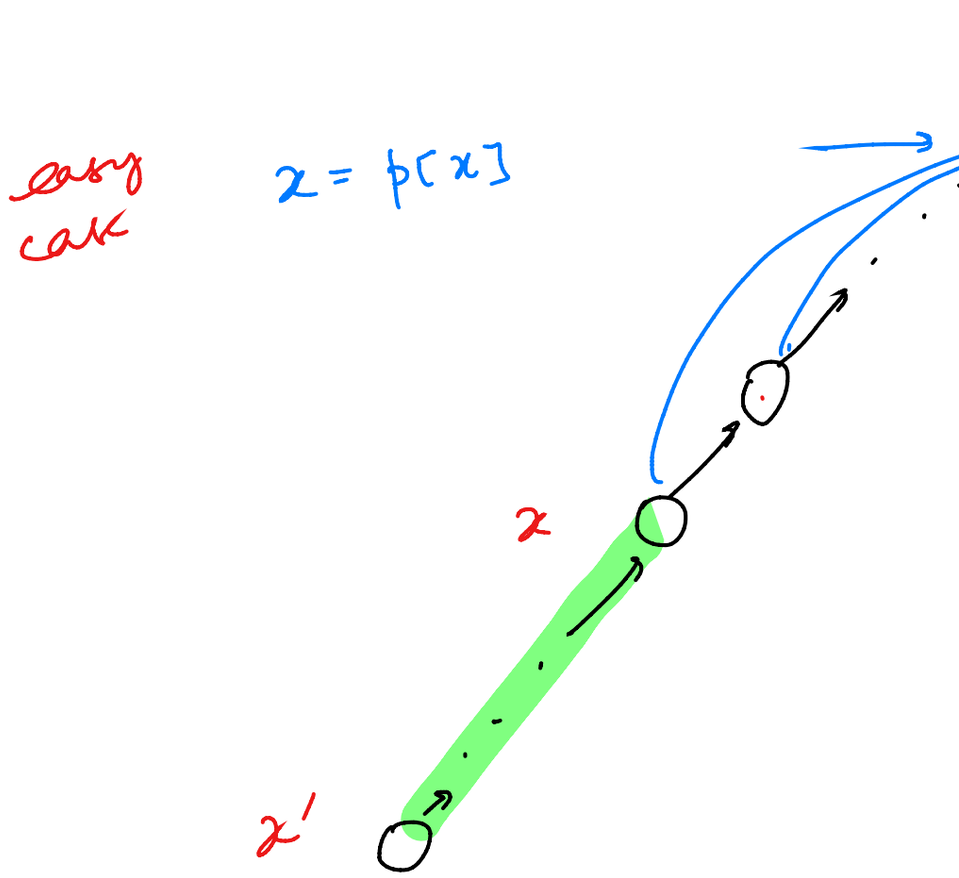
if $rank[x] = rank[y]$

then $rank[y] \leftarrow rank[y] + 1$

+2 ?

```
FIND-SET(x)
  if   x ≠ p[x]
  then p[x] ← FIND-SET(p[x])
  return p[x]
```

- going up the tree all the way back to root
- as the recursion unfurl any node above x points to the representative
- next access of x and above takes only 2 steps



easy call

- x' below x, you still hop 1 node at a time, once you hit x, we immediately jump to parent / root.

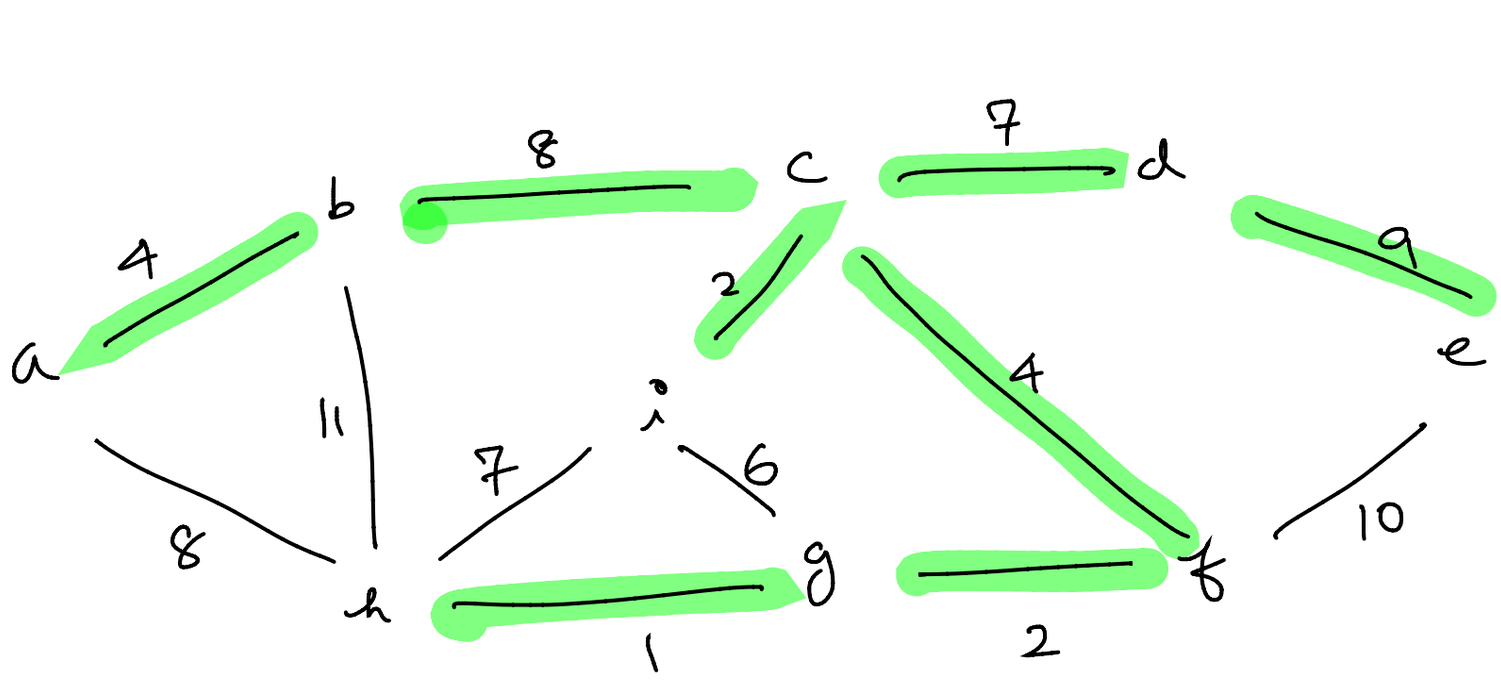
Minimum Spanning Tree

Undirected Graph $G = (V, E)$

Goal: Find $T \subseteq E$ such that

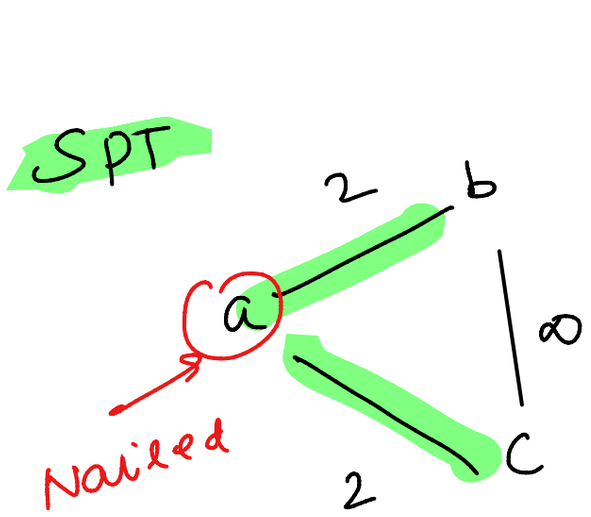
- T connects all the vertices
- $\sum_{(u,v) \in T} w(u,v)$ is a minimum.

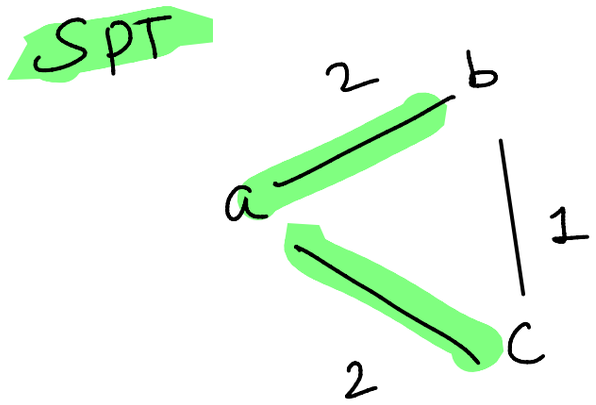
Example:



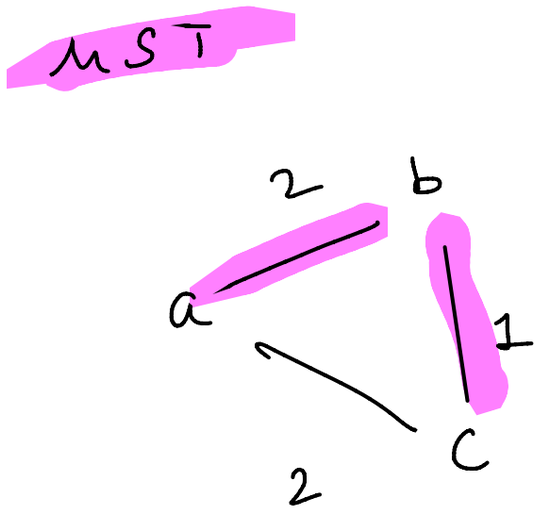
can remove (b, c) and add (a, d)

Shortest Path tree vs Minimum Spanning Tree





vs



Say a is source

Shortest Path: From source what is the best to each node

Minimum Span: From some connected component what is the best to another component not connected

(already an idea for an algo)

↓

Building roads that connect cities

- $A \subseteq MST$
- Safe Edge $(u,v) \cup \{A\} \subseteq MST$

GENERIC - MST (G, w)

$A \leftarrow \phi$

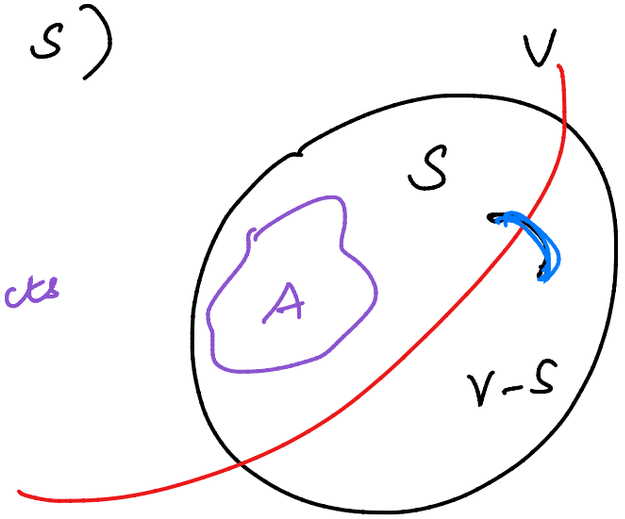
while A is not an MST already

do find (u,v) safe for A

$A \leftarrow A \cup \{(u,v)\}$

return A

cut : $(S, V-S)$



cut respects A

light edge (weight is minimum of all edges that cross the cut)