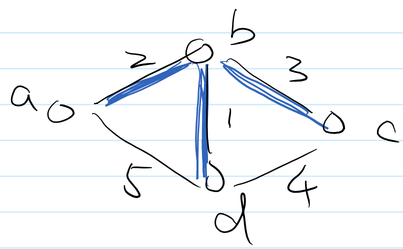
COMPSCI 330 Lecture 11 Minimum Spanning Trees

Monday, October 3, 2016 2:13 PM

- Property of Minimum Spanning Trees
- Prim's Algorithm
- Kruskal's Algorithm

- motivation: connect vertices (building, computers) with minimum cost.
- graph: set of "possible" edges

  edge weight = cost of connecting two vertices



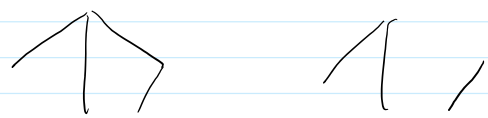
graph is undirected

edge weight $w(e) \geq 0$

$\Longrightarrow$ no point to select a cycle

$\Longrightarrow$ should always find a tree

- spanning tree: for a graph G, tree T is a spanning tree if T connects same set of vertices, and edges of T are all edges of G (trees are always connected)

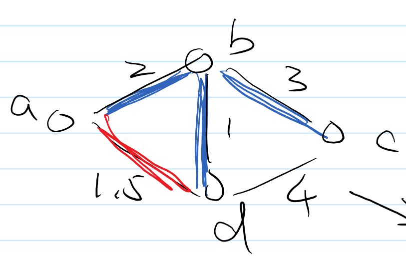


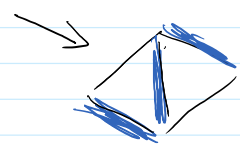
cost of the tree $w(T) = \sum_{e \in T} w(e)$

- Minimum Spanning Tree (MST) tree T with minimum cost $w(T)$ among all spanning trees.
- DFS/BFS trees are all spanning trees.
- tree edges / off-tree edges

  for off-tree edge (a, d)

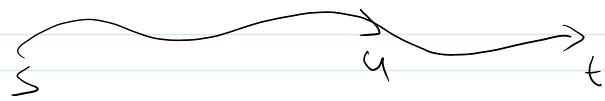
  $\longrightarrow$ cycle (a, b, d)



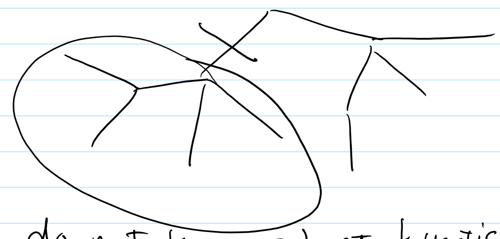


- swap operation: add edge e to spanning tree T then remove edge e' in the cycle created by e.
- swap can improve cost of spanning trees.

- Designing algorithm: Recall: for shortest path

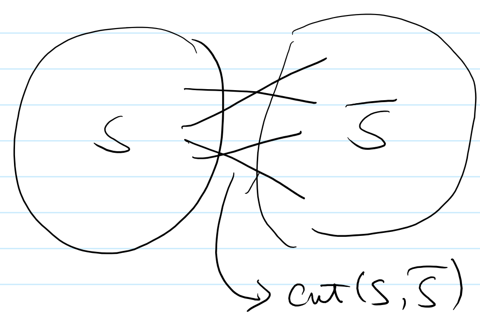


for MST

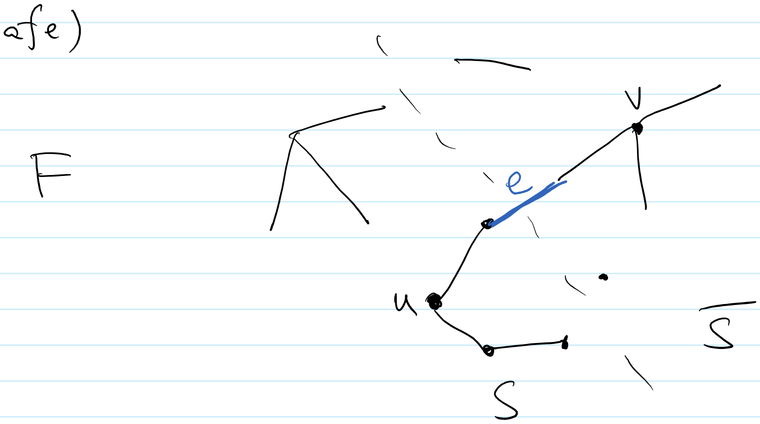


do not know subset of vertices (Dynamic Programming Fails)

- "safe" edges
  - Cut: partition vertices into two parts, cut is the set of edges that crosses the two parts



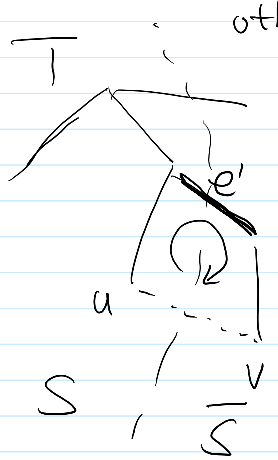
- Theorem (Key): Suppose $F \subset T$ for some MST T, if $(S, \bar{S})$ is cut that respects the components of F, and e is (one of) the min cost edge in cut$(S, \bar{S})$, then $F \cup \{e\} \subseteq T'$ for some MST T'. (and e is safe)



Proof: Let $e = (u, v)$, $u \in S$, $v \in \bar{S}$

if $(u, v) \in T$ then the theorem is true ✓

otherwise (u, v) is an off-tree edge



Consider the cycle created by adding (u,v) to T.

there must be an edge $e' \in T$ in the cycle that also crosses cut $(S, \bar{S})$

$w(e) \le w(e')$ (by definition of e)

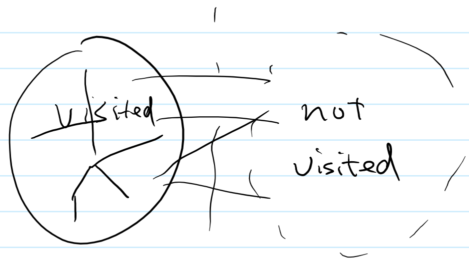
$T + e - e'$ is also a spanning tree (swap)

$w(T + e - e') \le w(T) + w(e) - w(e') \le w(T)$

$T + e - e'$ is also a MST

$F \cup \{e\} \subseteq T + e - e'$ $\square$

- Prim's algorithm
  - Similar to Dijkstra
  - maintain visited vertices S, a tree on S
  - Theorem $\Longrightarrow$ need to find min cost edge between visited/unvisited



Prim's algorithm

initialize $d(s) = 0$, mark s as visited

for any edge (s, u), $h(u) = w(s, u)$ ($prev(u) = s$)

for any other vertex $h(u) = +\infty$

for $i = 2$ to n

  let u to be the vertex with minimum $h(u)$ among unvisited vertices

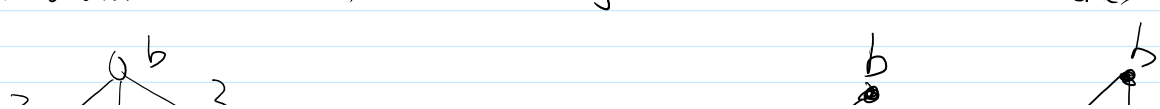
  add $(prev(u), u)$ to the tree, mark u visited

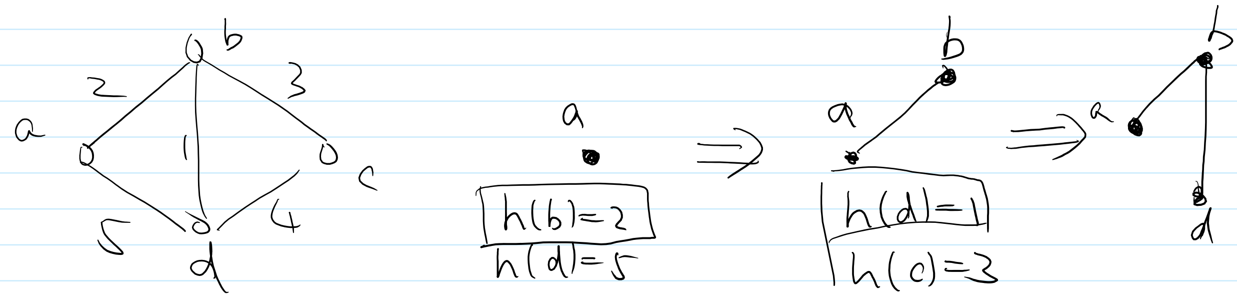
  for all edges (u, v)

    if $w(u, v) < h(v)$

      $h(v) = w(u, v)$ ($prev(v) = u$)

$h(u)$: for unvisited vertex u, min cost of its connection to visited vertex

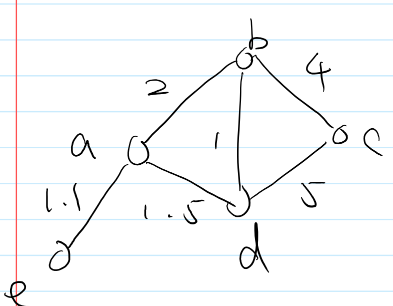




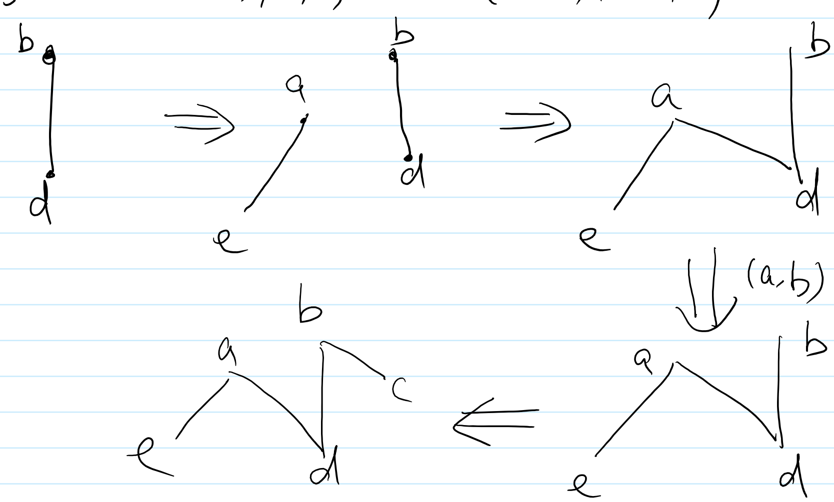
- running time $O(m + n\log n)$
- correctness: induction using the Key theorem.

- Kruskal's algorithm
  - try to select edges in MST
  - what edge? min cost edge if it does not introduce a cycle.

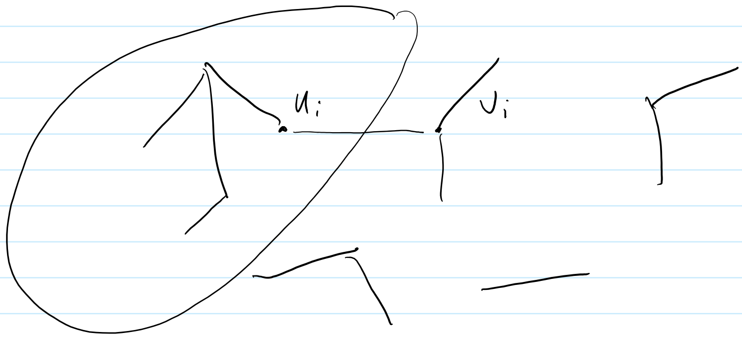
Sort edges in ascending order $e_1 = (u_1, v_1)$, $e_2 = (u_2, v_2) \dots e_m = (u_m, v_m)$
initialize each vertex in their own connected component, $T = \phi$
for i = 1 to m
    if u_i, v_i are in different components
        add (u_i, v_i) to T
        merge the components of u_i, v_i



Sort edges (b, d) (a, e), (a, d) (a, b) (b, c), (c, d)



- Correctness: if we connected u_i, v_i



let S be set of vertices connected to u_i, (u_i, v_i) is safe because of cut $(S, \bar{S})$

- running time: $O(m \log m)$