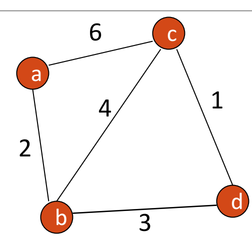
Day 25

MINIMUM SPANNING TREE PROBLEM
PRIM'S ALGORITHM
KRUSKAL'S ALGORITHM

Spanning Tree

Consider the following graph:



A *spanning tree* of a connected graph *G* is a a connected acyclic subgraph of *G* that includes all of *G*'s vertices.

It takes on the form of a tree.

The above graph has several spanning trees, including:

{ab, ac, cd}

{ac, bc, cd}

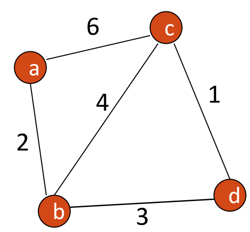
Minimum Spanning Tree

A *minimum spanning tree* of a weighted, connected graph *G* is a spanning tree of minimum total weight.

If there is a spanning tree, then there is a minimum spanning tree.

For the graph from the prior slide:

{ab, bd, cd}   [value 2 + 3 + 1 = 6]



Prim’s Algorithm

Grow a tree.

Pick a node.

Pick the lowest cost edge out of that node, and connect those two nodes.

From now on, pick the lowest cost edge of all the nodes already forming the tree, ensuring that adding that node does not create a cycle.

Repeat until all nodes are included.

Prim's Algorithm in detail

ALGORITHM *Prim(G)*

//Prim's algorithm for constructing a minimum spanning tree
//Input: A weighted connected graph $G = \langle V, E \rangle$
//Output: E_T, the set of edges composing a minimum spanning tree of G
$V_T \leftarrow \{v_0\}$ //the set of tree vertices can be initialized with any vertex
$E_T \leftarrow \varnothing$
for $i \leftarrow 1$ **to** $|V| - 1$ **do**
find a minimum-weight edge $e^* = (v^*, u^*)$ among all the edges (v, u)
such that v is in V_T and u is in $V - V_T$
$V_T \leftarrow V_T \cup \{u^*\}$
$E_T \leftarrow E_T \cup \{e^*\}$
return E_T

Kruskal's Algorithm

Grow forests.

Sort the edges from lowest to highest edge cost.

Create a forest in which each node forms a tree.

Pick the lowest cost edge and join the two trees, if that does not cause a cycle.

Continue until all trees are merged into a single tree.

Kruskal's Details and Analysis

ALGORITHM *Kruskal(G)*

//Kruskal's algorithm for constructing a minimum spanning tree
//Input: A weighted connected graph $G = \langle V, E \rangle$
//Output: E_T, the set of edges composing a minimum spanning tree of G
sort E in nondecreasing order of the edge weights $w(e_{i_1}) \leq \cdots \leq w(e_{i_{|E|}})$
$E_T \leftarrow \varnothing$; $ecounter \leftarrow 0$ //initialize the set of tree edges and its size
$k \leftarrow 0$ //initialize the number of processed edges
while $ecounter < |V| - 1$ **do**
  $k \leftarrow k + 1$
  if $E_T \cup \{e_{i_k}\}$ is acyclic
    $E_T \leftarrow E_T \cup \{e_{i_k}\}$; $ecounter \leftarrow ecounter + 1$
return E_T