Lecture 21

CSE 331

Minimum Spanning Tree Problem

Input: Undirected, connected $G = (V,E)$, edge costs c_e

Output: Subset $E' \subseteq E$), s.t. $T = (V,E')$ is connected
$C(T)$ is minimized

If all $c_e > 0$, then T is indeed a tree

Rest of today’s agenda

Greedy algorithm(s) for MST problem

Kruskal’s Algorithm

Joseph B. Kruskal

Input: $G=(V,E)$, $c_e > 0$ for every e in E

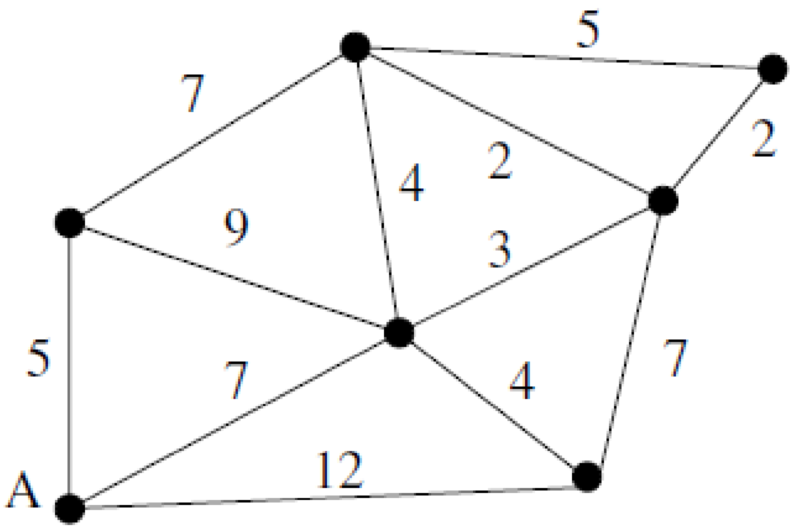
$T = \emptyset$

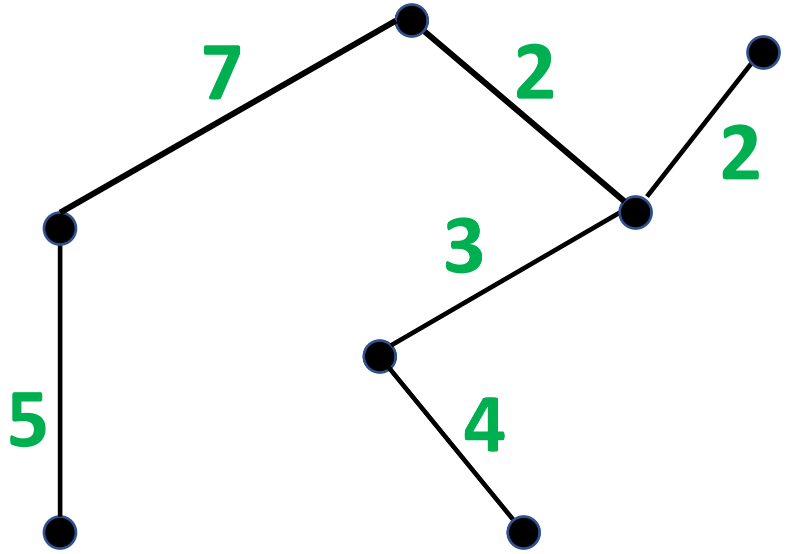
Sort edges in increasing order of their cost

Consider edges in sorted order

If an edge can be added to T without adding a cycle then add it to T

Kruskal's Algorithm





Input: G=(V,E), $c_e > 0$ for every e in E

T = Ø

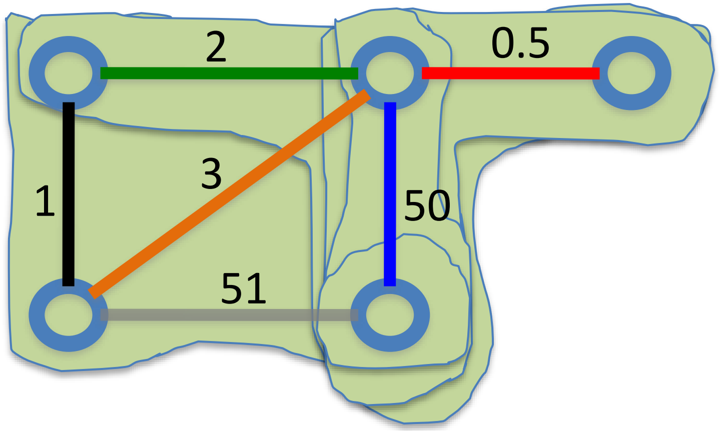
Sort edges in increasing order of their cost 2, 2, 3, 4, 4, 5, 5, 7, 7, 7, 9, 12

Consider edges in sorted order

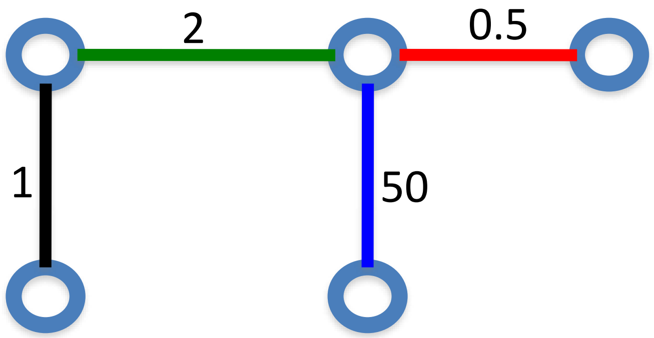
If an edge can be added to T without adding a cycle then add it to T

Prim's algorithm

Similar to Dijkstra's algorithm



Robert Prim



Input: G=(V,E), c_e> 0 for every e in E

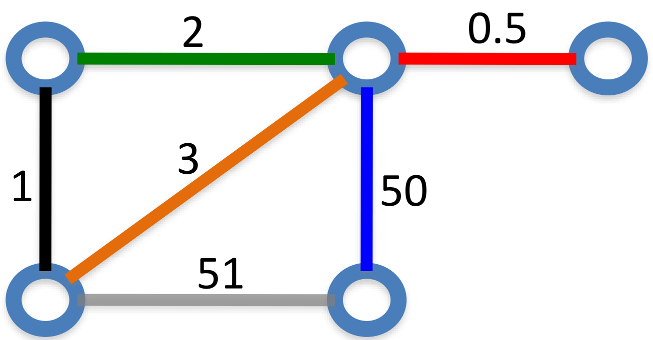
S = {s}, T = Ø

While S is not the same as V

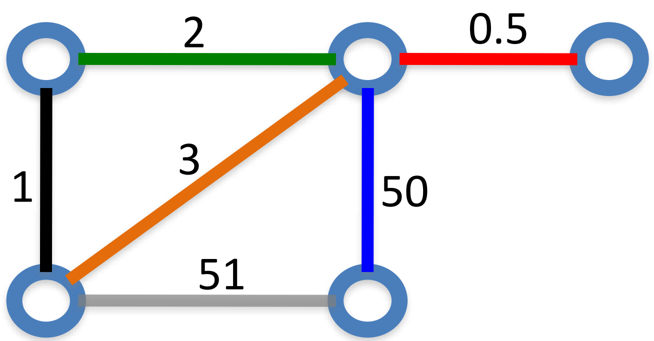
Among edges e= (u,w) with u in S and w not in S, pick one with minimum cost

Add w to S, e to T

Reverse-Delete Algorithm



Input: G=(V,E), $c_e > 0$ for every e in E



T = E

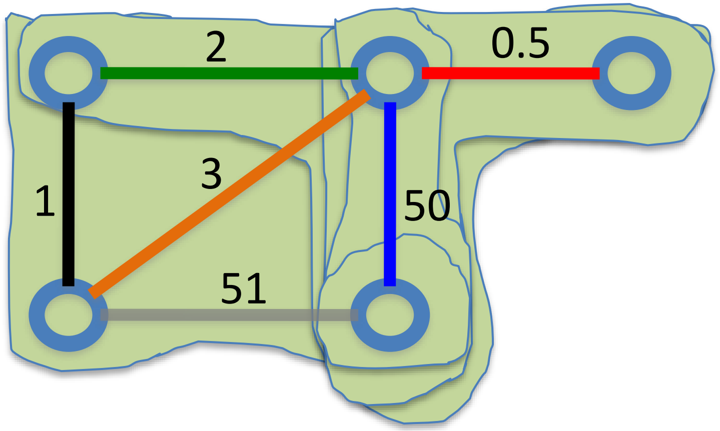
Sort edges in decreasing order of their cost

Consider edges in sorted order

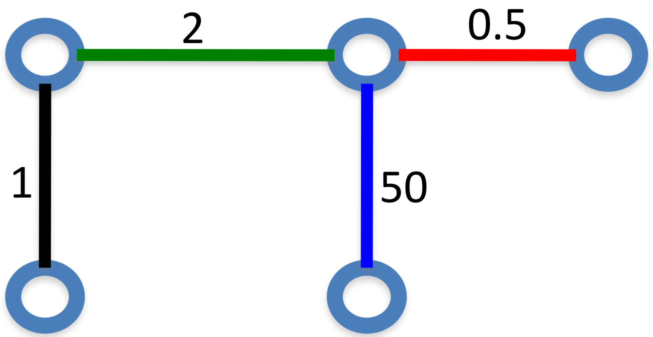
If an edge can be removed T without disconnecting T then remove it

Prim's algorithm

Similar to Dijkstra's algorithm



Robert Prim



Input: G=(V,E), c_e> 0 for every e in E

S = {s}, T = Ø

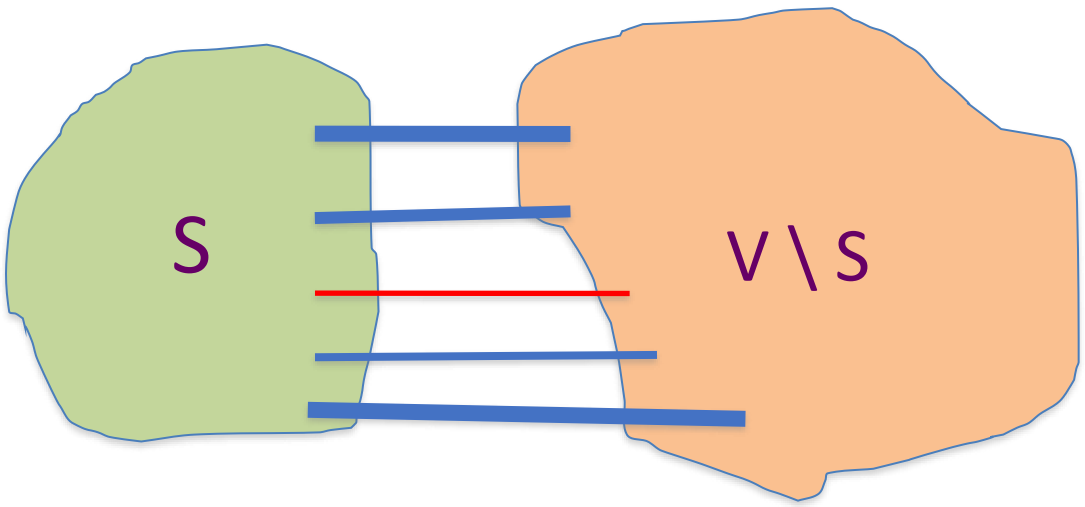
While S is not the same as V

Among edges e= (u,w) with u in S and w not in S, pick one with minimum cost

Add w to S, e to T

Cut Property Lemma for MSTs

Condition: S and V\S are non-empty



Cheapest crossing edge is in **all** MSTs

Assumption: All edge costs are distinct

Agenda

Optimality of Prim’s algorithm

Prove Cut Property Lemma

Optimality of Kruskal’s algorithm

Remove distinct edge weights assumption

On to the board…