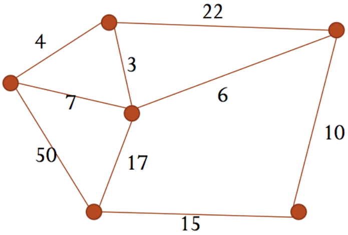
COL351: Analysis and Design of Algorithms

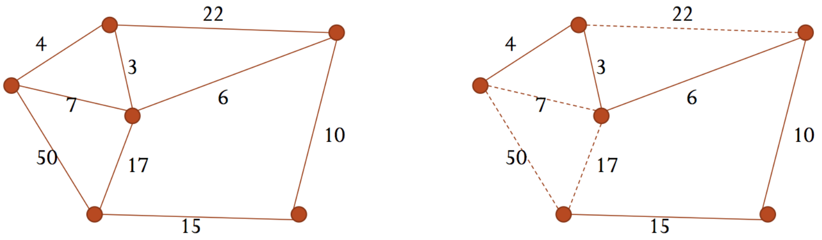
Ragesh Jaiswal, CSE, IITD

Greedy Algorithms

- Spanning Tree: Given a strongly connected graph $G = (V, E)$, a *spanning tree* of G is a subgraph $G' = (V, E')$ such that G' is a tree.
- Minimum Spanning Tree (MST): Given a strongly connected weighted graph $G = (V, E)$, a *Minimum Spanning Tree* of G is a spanning tree of G of minimum total weight (i.e., sum of weight of edges in the tree).

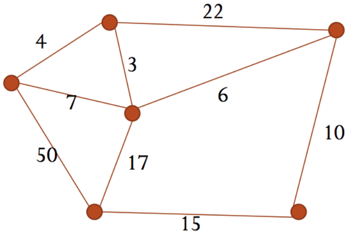


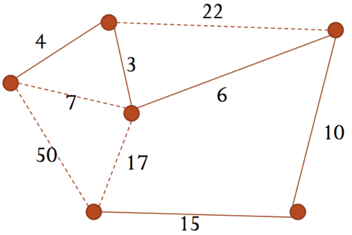
- Spanning Tree: Given a strongly connected graph $G = (V, E)$, a *spanning tree* of G is a subgraph $G' = (V, E')$ such that G' is a tree.
- Minimum Spanning Tree (MST): Given a strongly connected weighted graph $G = (V, E)$, a *Minimum Spanning Tree* of G is a spanning tree of G of minimum total weight (i.e., sum of weight of edges in the tree).



Problem

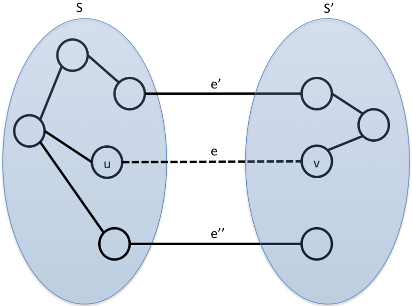
Given a weighted graph G where all the edge weights are distinct, give an algorithm for finding the MST of G.





Theorem

Cut property: Given a weighted graph $G = (V, E)$ where all the edge weights are distinct. Consider a non-empty proper subset S of V and $S' = V \setminus S$. Let e be the least weighted edge between any pair of vertices (u, v), where u is in S and v is in S'. Then e is necessarily present in *all* MSTs of G.



Algorithm

`Prim's Algorithm(G)`

- $S \leftarrow \{u\}$ //u *is an arbitrary vertex in the graph*
- $T \leftarrow \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight edge between S and $V \setminus S$
  - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \cup \{w\}$

Algorithm

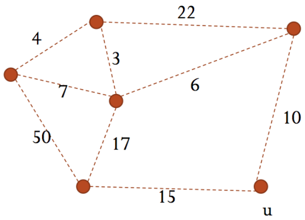
`Kruskal's Algorithm(G)`

- $S \leftarrow E$; $T \leftarrow \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \setminus \{e\}$

Algorithm

`Prim's Algorithm(G)`

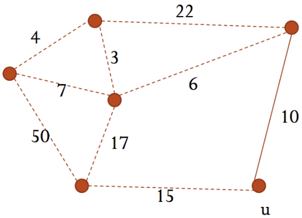
- $S \leftarrow \{u\}$ *//u is an arbitrary vertex in the graph*
- $T \leftarrow \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight edge between S and $V \setminus S$
  - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \cup \{w\}$



Algorithm

`Prim's Algorithm(G)`

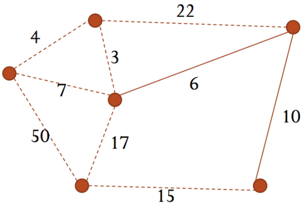
- $S \leftarrow \{u\}$ *//u is an arbitrary vertex in the graph*
- $T \leftarrow \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight edge between S and $V \setminus S$
  - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \cup \{w\}$



Algorithm

`Prim's Algorithm(G)`

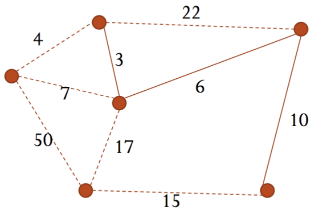
- $S \leftarrow \{u\}$ *//u is an arbitrary vertex in the graph*
- $T \leftarrow \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight edge between S and $V \setminus S$
  - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \cup \{w\}$



Algorithm

`Prim's Algorithm(G)`

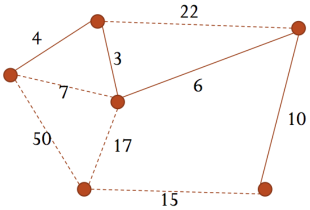
- $S \leftarrow \{u\}$ *//u is an arbitrary vertex in the graph*
- $T \leftarrow \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight edge between S and $V \setminus S$
  - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \cup \{w\}$



Algorithm

`Prim's Algorithm(G)`

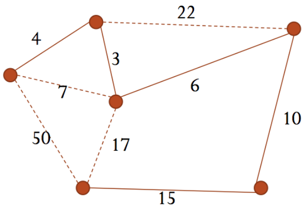
- $S \leftarrow \{u\}$ *//u is an arbitrary vertex in the graph*
- $T \leftarrow \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight edge between S and $V \setminus S$
  - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \cup \{w\}$



Algorithm

`Prim's Algorithm(G)`

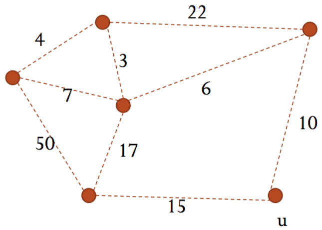
- $S \leftarrow \{u\}$ //*u is an arbitrary vertex in the graph*
- $T \leftarrow \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight edge between S and $V \setminus S$
  - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \cup \{w\}$



Algorithm

`Kruskal's Algorithm(G)`

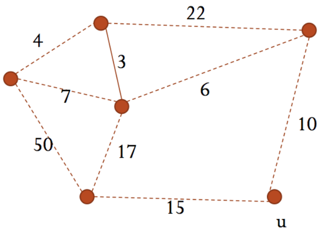
- $S \leftarrow E$; $T \leftarrow \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \setminus \{e\}$



Algorithm

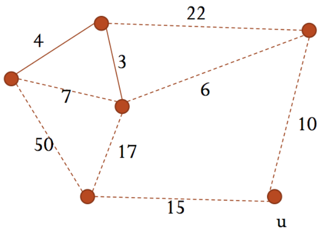
`Kruskal's Algorithm(G)`

- $S \leftarrow E$; $T \leftarrow \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \setminus \{e\}$



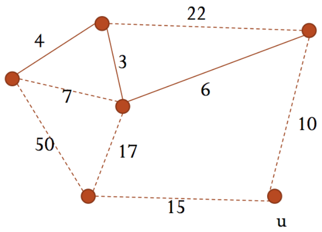
Algorithm

`Kruskal's Algorithm(G)`
- $S \leftarrow E$; $T \leftarrow \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \setminus \{e\}$



Algorithm

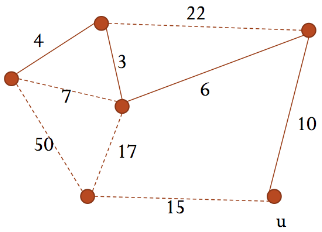
`Kruskal's Algorithm(G)`
- $S \leftarrow E$; $T \leftarrow \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \setminus \{e\}$



Algorithm

`Kruskal's Algorithm(G)`

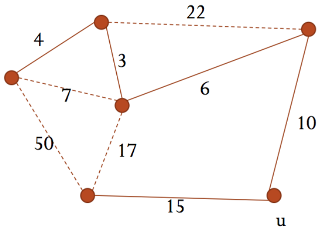
- $S \leftarrow E$; $T \leftarrow \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \setminus \{e\}$



Algorithm

`Kruskal's Algorithm(G)`

- $S \leftarrow E$; $T \leftarrow \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \setminus \{e\}$



Algorithm

`Prim's Algorithm(G)`

- $S \leftarrow \{u\}$ *//u is an arbitrary vertex in the graph*
- $T \leftarrow \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight edge between S and $V \setminus S$
  - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \cup \{w\}$

- What is the running time of Prim's algorithm?

Algorithm

```
Prim's Algorithm(G)
```
- $S \leftarrow \{u\}$ *//u is an arbitrary vertex in the graph*
- $T \leftarrow \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight edge between S and $V \setminus S$
  - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \cup \{w\}$

- What is the running time of Prim's algorithm?
  $O(|E| \cdot \log |V|)$
  - Using a priority queue.

Algorithm

`Kruskal's Algorithm(G)`
- $S \leftarrow E$; $T \leftarrow \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \setminus \{e\}$

Algorithm

`Kruskal's Algorithm(G)`
- $S \leftarrow E$; $T \leftarrow \{\}$
- While the edge set T does not connect all the vertices
  - *//Note that $G' = (V, T)$ contains dicsonnected components*
  - Let e be the minimum weight edge in the set S
  - ~~If e does not create a cycle in T~~
  - If u and v are in different components of G'
    - $T \leftarrow T \cup \{e\}$
  - $S \leftarrow S \setminus \{e\}$

End