CSL 356: Analysis and Design of Algorithms

Ragesh Jaiswal

CSE, IIT Delhi

Greedy Algorithms: Example

- Job Scheduling: You are given n jobs and you are supposed to schedule these jobs on a machine. Each job i consists of a duration $T(i)$ and a deadline $D(i)$. The lateness of a job wrt. a schedule is defined as $\max(0, F(i) - D(i))$, where $F(i)$ is the finishing time of job i as per the schedule. The goal is to minimize the maximum lateness.
- Greedy Strategies:
  - Smallest jobs first.

Greedy Algorithms: Example

- Job Scheduling: You are given n jobs and you are supposed to schedule these jobs on a machine. Each job i consists of a duration $T(i)$ and a deadline $D(i)$. The lateness of a job wrt. a schedule is defined as $\max(0, F(i) - D(i))$, where $F(i)$ is the finishing time of job i as per the schedule. The goal is to minimize the maximum lateness.
- Greedy Strategies:
  - ~~Smallest jobs first~~.
  - Earliest deadline first.

GreedyJobSchedule
- Sort the Jobs in increasing order of deadlines and schedule the jobs on the machine in this order.

Greedy Algorithms: Example

- Claim: There is an optimal schedule with no idle time (time when the machine is idle).
- Definition: A schedule is said to have inversion if there are a pair of jobs (i, j) such that
  1. $D(i) < D(j)$, and
  2. Job j is performed before job i as per the schedule.
- Claim: There is an optimal schedule with no idle time and no inversion.
- Proof: Consider an optimal schedule O. First if there is any idle time we obtain another optimal schedule O_1 without idle time. Suppose O_1 has inversions. Consider one such inversion (i, j).

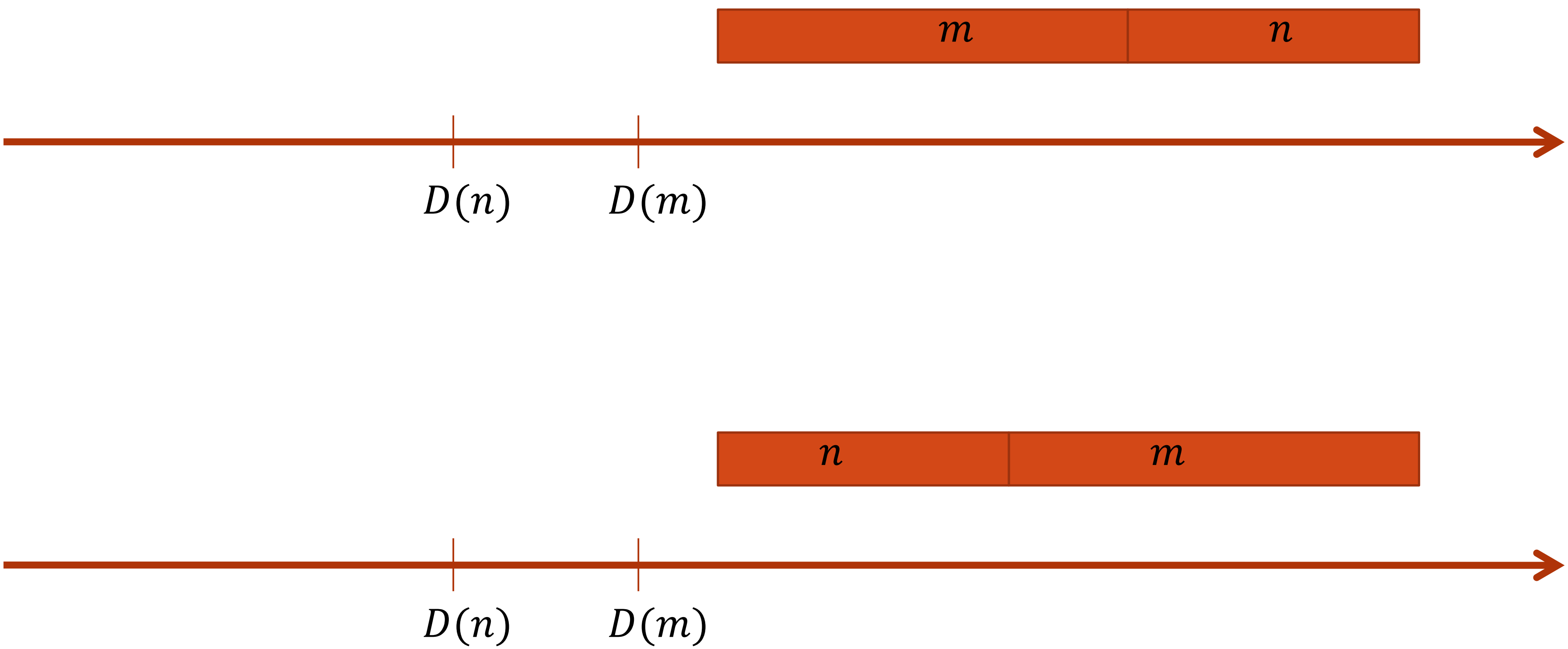
Greedy Algorithms: Example

- Claim: There is an optimal schedule with no idle time and no inversion.
- Proof: Consider an optimal schedule O. First if there is any idle time we obtain another optimal schedule O_1 without idle time. Suppose O_1 has inversions. Consider one such inversion (i, j).

- There exists a pair of adjacently scheduled jobs (m, n) such that the schedule has an inversion wrt. (m, n).

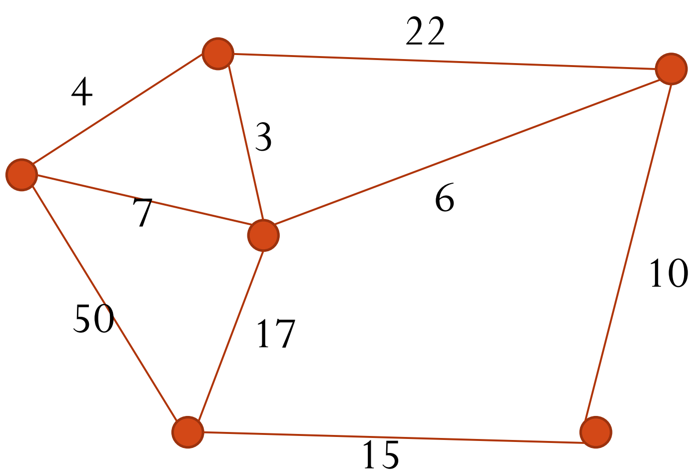
Greedy Algorithms: Example

- <u>Claim</u>: *Exchanging* m and n does not increase the maximum lateness.



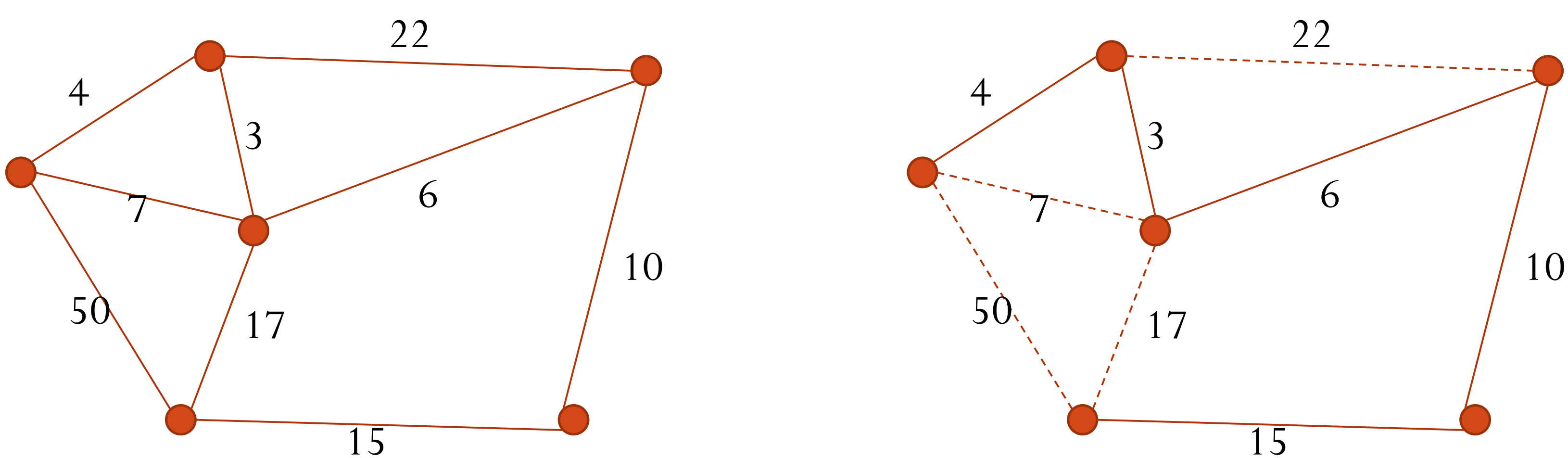
Greedy Algorithms: Example

- 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: 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 weights of edges in the tree).



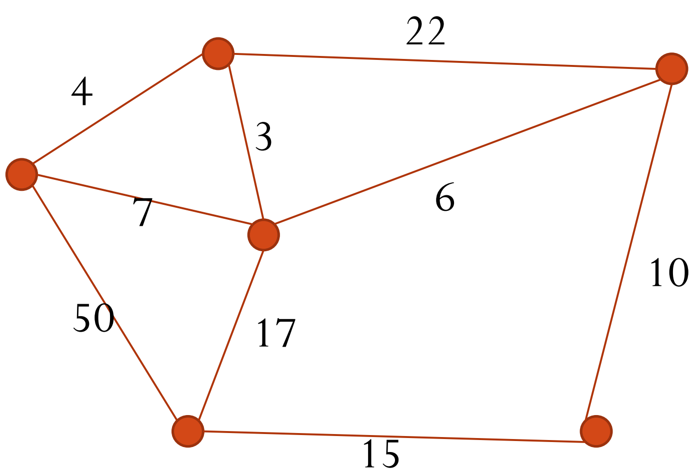
Greedy Algorithms: Example

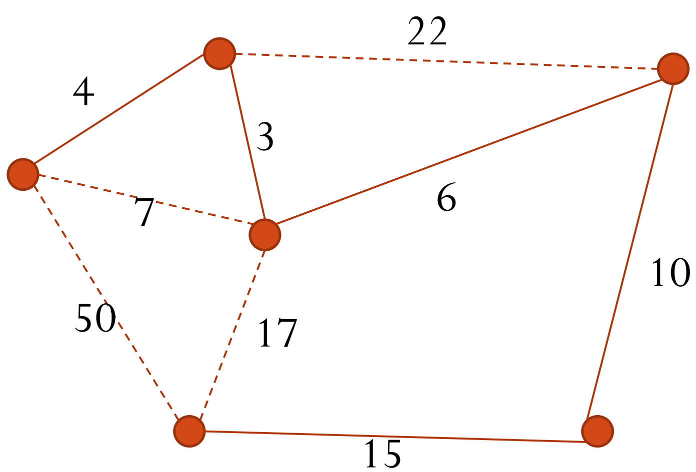
- <u>Spanning Tree</u>: Given a strongly connected graph $G = (V, E)$, a spanning tree of G is a subgraph $G' = (V, E')$ such that G' is a tree.
- <u>Minimum Spanning Tree</u>: Given a strongly connected weighted graph $G = (V, E)$. A minimum spanning tree of G is a spanning tree of G of minimum total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



Greedy Algorithms: Example

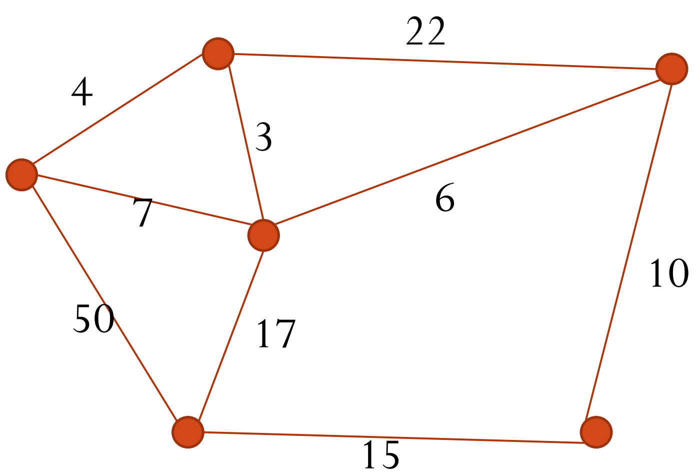
- Problem: Given a weighted graph G where all the edge weights are distinct. Give an algorithm for finding the MST of G.

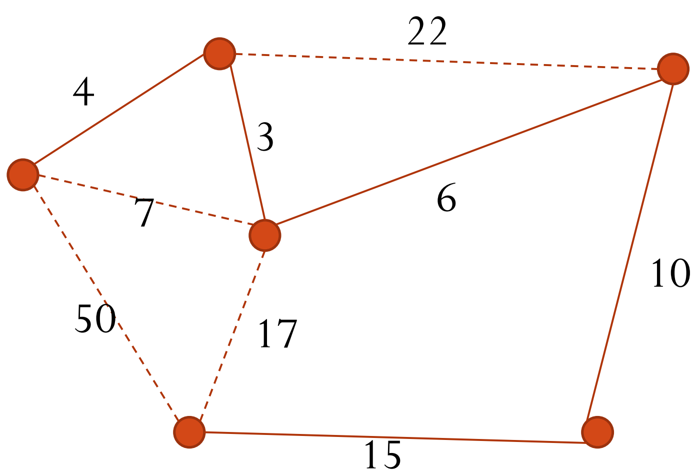




Greedy Algorithms: Example

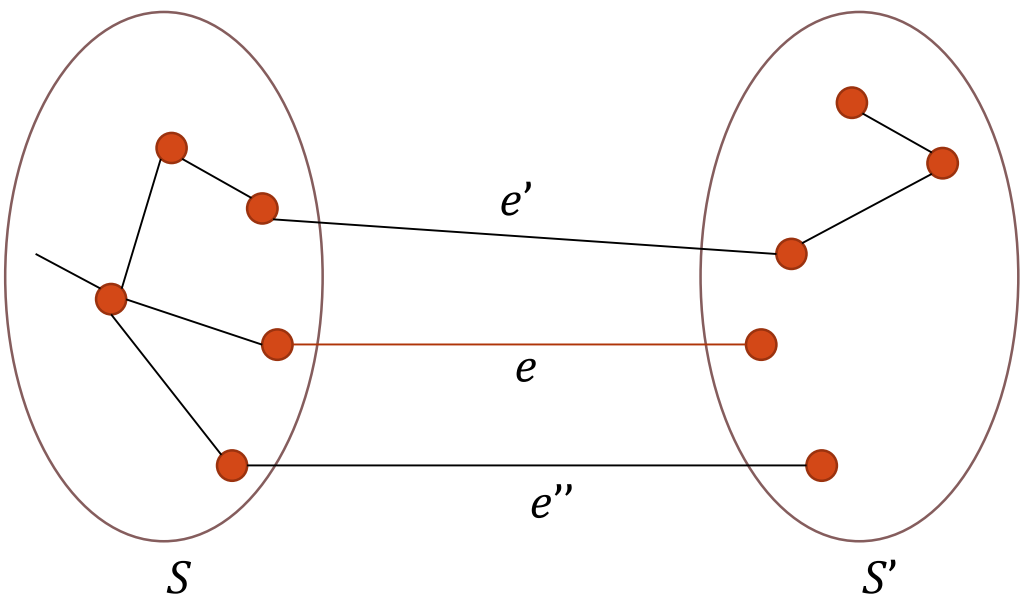
- <u>Theorem (Cut Property)</u>: Given a weighted graph $G = (V, E)$ where all the edge weights are distinct. Consider a non-empty proper subset of S of V and $S' = V \setminus S$. Let e be the least weighted edge between any pair of vertices (u, v) where u is in S and v is in S'. Then e is necessarily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





Greedy Algorithms: Example

- <u>Theorem (Cut Property)</u>: Given a weighted graph $G = (V, E)$ where all the edge weights are distinct. Consider a non-empty proper subset of 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.
- <u>Proof</u>:



Greedy Algorithms: Example

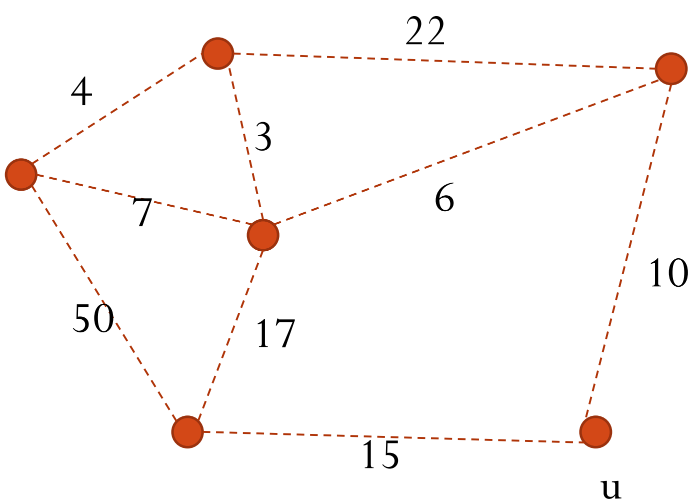
Prim's Algorithm(G)
- $S = \{u\}$ //u *is any arbitrary vertex in the graph*
- $T = \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight cut edge between S and $V \backslash S$
  - $T = T \cup \{e\}$
  - $S = S \cup \{w\}$

Kruskal's Algorithm(G)
- $S = E$; $T = \{\}$
- 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
    - $T = T \cup \{e\}$
  - $S = S - \{e\}$

Greedy Algorithms: Example

Prim's Algorithm(G)

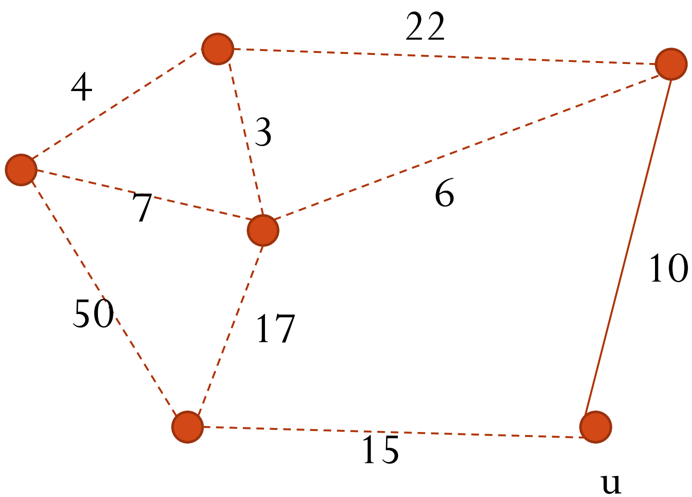
- $S = \{u\}$ //u *is any arbitrary vertex in the graph*
- $T = \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight cut edge between S and $V \setminus S$
  - $T = T \cup \{e\}$
  - $S = S \cup \{w\}$



Greedy Algorithms: Example

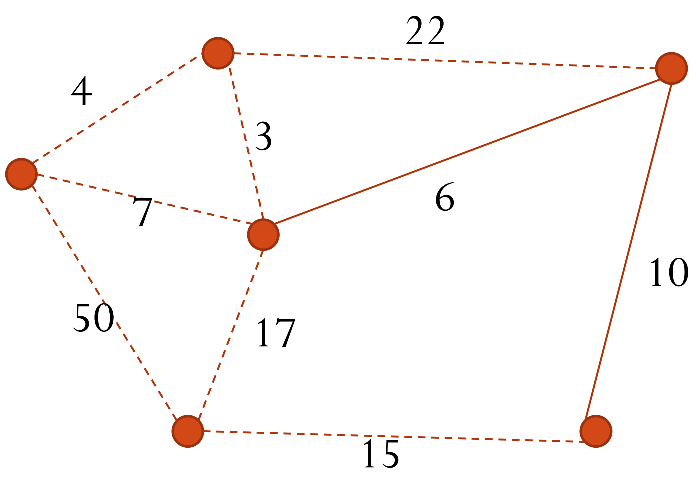
Prim's Algorithm(G)

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



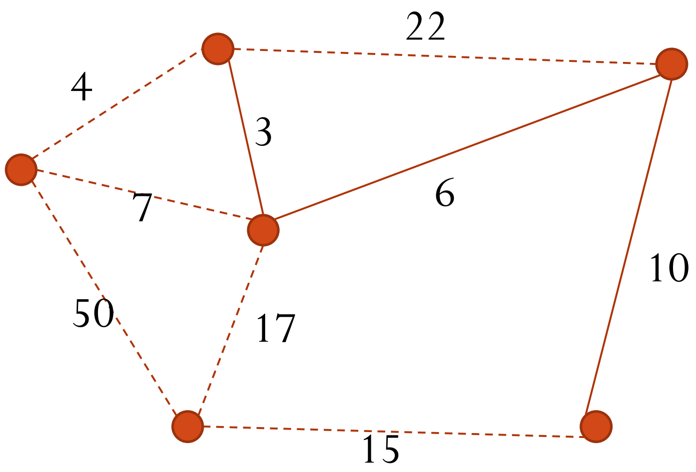
Greedy Algorithms: Example

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



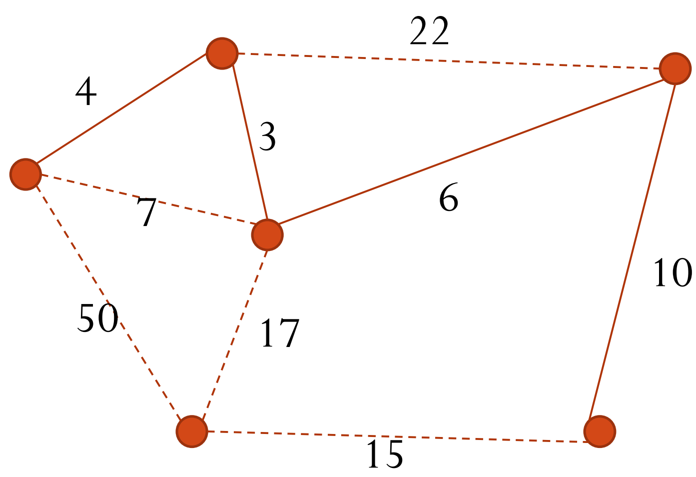
Greedy Algorithms: Example

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



Greedy Algorithms: Example

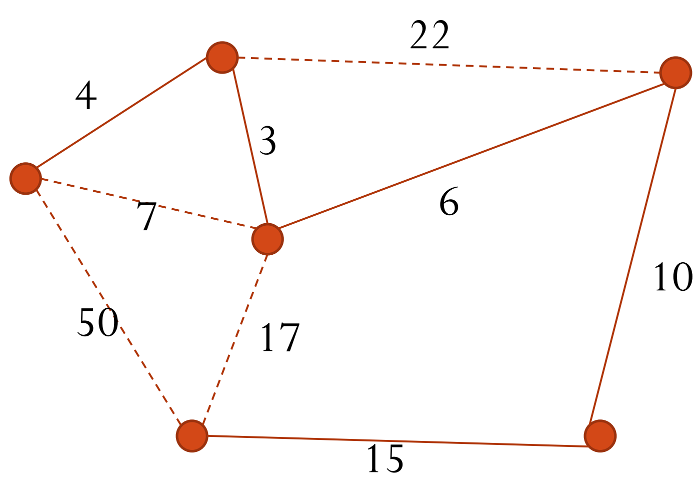
Prim's Algorithm(G)
- $S = \{u\}$ //u *is any arbitrary vertex in the graph*
- $T = \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight cut edge between S and $V \setminus S$
  - $T = T \cup \{e\}$
  - $S = S \cup \{w\}$



Greedy Algorithms: Example

Prim's Algorithm(G)

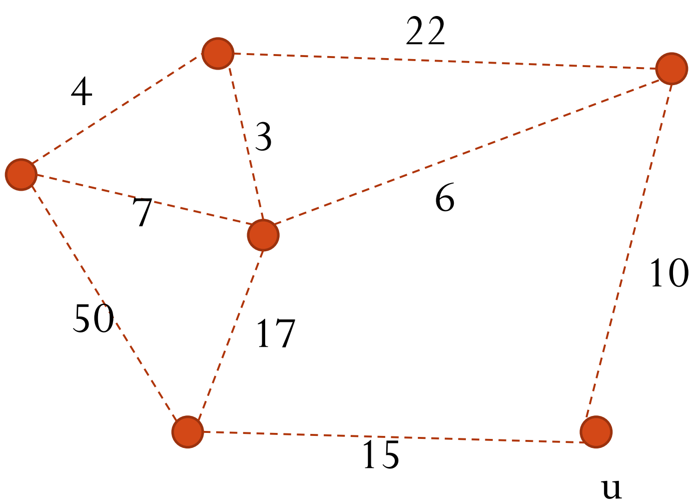
- $S = \{u\}$ $//u$ *is any arbitrary vertex in the graph*
- $T = \{\}$
- While S does not contain all vertices
  - Let $e = (v, w)$ be the minimum weight cut edge between S and $V \backslash S$
  - $T = T \cup \{e\}$
  - $S = S \cup \{w\}$



Greedy Algorithms: Example

Kruskal's Algorithm(G)

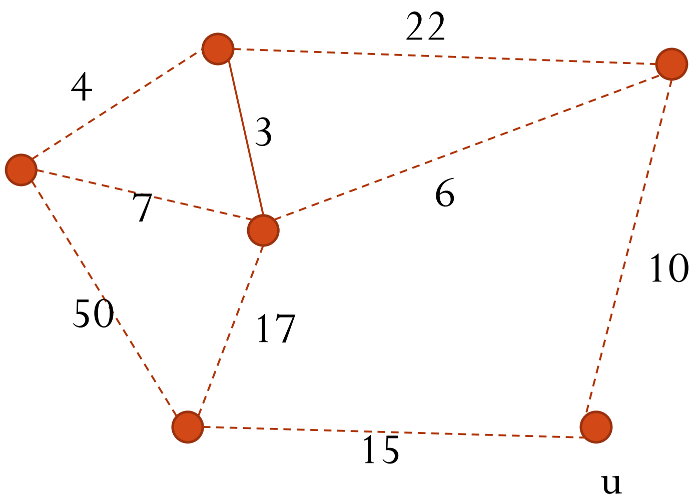
- $S = E$; $T = \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T = T \cup \{e\}$
  - $S = S - \{e\}$



Greedy Algorithms: Example

Kruskal's Algorithm(G)

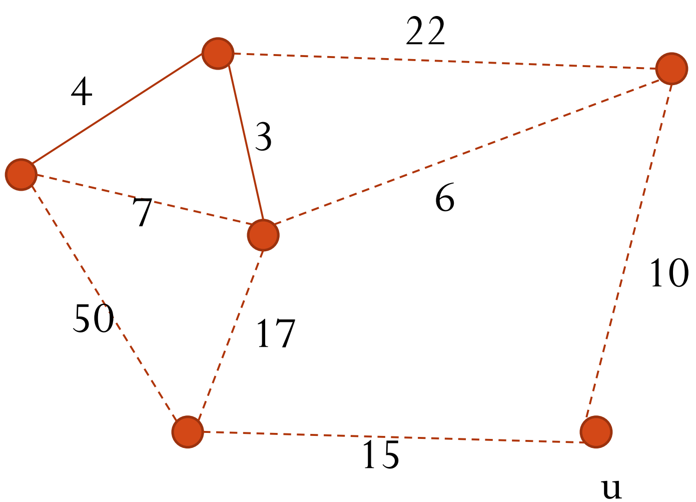
- $S = E; T = \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T = T \cup \{e\}$
  - $S = S - \{e\}$



Greedy Algorithms: Example

Kruskal's Algorithm(G)

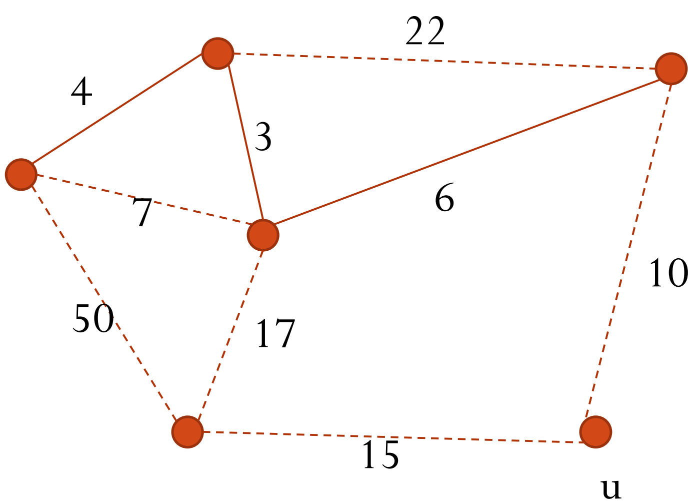
- $S = E$; $T = \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T = T \cup \{e\}$
  - $S = S - \{e\}$



Greedy Algorithms: Example

Kruskal's Algorithm(G)

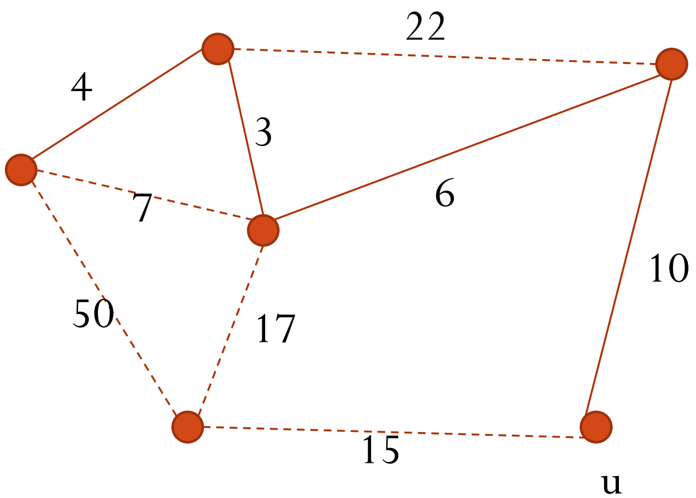
- $S = E; T = \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T = T \cup \{e\}$
  - $S = S - \{e\}$



Greedy Algorithms: Example

Kruskal's Algorithm(G)

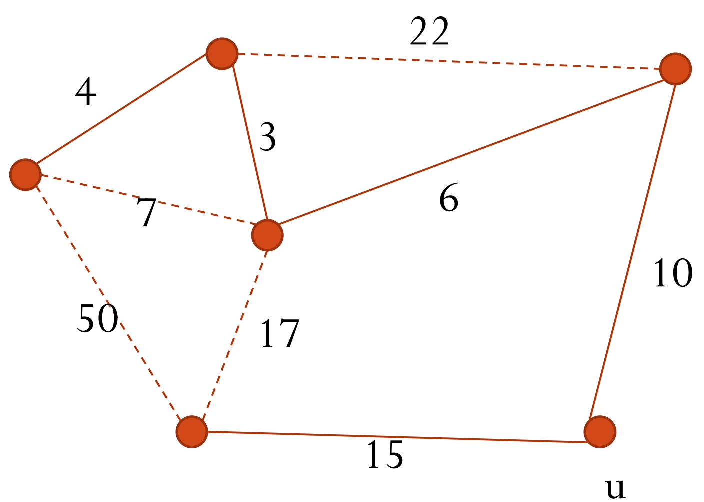
- $S = E$; $T = \{\}$
- While the edge set T does not connect all the vertices
  - Let e be the minimum weight edge in the set S
  - If e does not create a cycle in T
    - $T = T \cup \{e\}$
  - $S = S - \{e\}$



Greedy Algorithms: Example

Kruskal's Algorithm(G)

- $S = E; T = \{\}$
- 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
    - $T = T \cup \{e\}$
  - $S = S - \{e\}$



Greedy Algorithms: Example

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

- Running time?

Greedy Algorithms: Example

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

- Running time?
  - $O(|E| \cdot \log |V|)$

Greedy Algorithms: Example

Kruskal's Algorithm(G)
- $S = E$; $T = \{\}$
- 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
    - $T = T \cup \{e\}$
  - $S = S - \{e\}$

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

End