Minimum Spanning Tree

- Minimum Spanning Tree
- Prim's Algorithm for MST
- Cuts in Graphs
- Correctness and Optimality of Prim's Algorithm
- Runtime
  - Basic Implementation
  - Vertex-Centric Implementation
  - Heap Based Implementation

Imdad ullah Khan

Prim's Algorithm

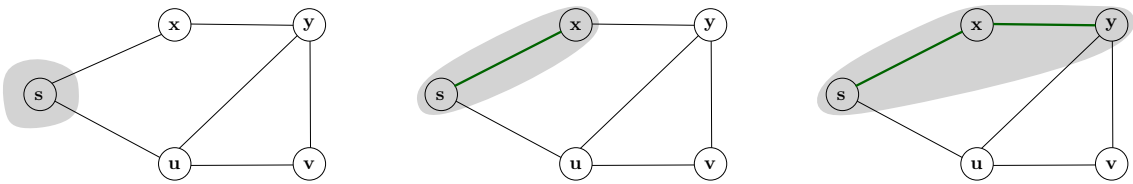
Input: A weighted graph $G = (V, E, w)$, $w : E \to \mathbb{R}$
Output: A spanning tree of G with minimum total weight

The Prim's algorithm

- Maintains a set $R \subset V$ and a tree T spanning vertices in R
- Initially $R = \{s\}$, an arbitrary vertex and $T = \emptyset$
- Grow R by adding one vertex v in every iteration
- Grow T by adding an edge connecting v to some vertex in current
- $V(T) = R$ (vertices spanned by T)
- Select a minimum crossing edge from R to $\overline{R}$

$$\arg\min_{e=(u,v), u \in R, v \notin R} w(e)$$

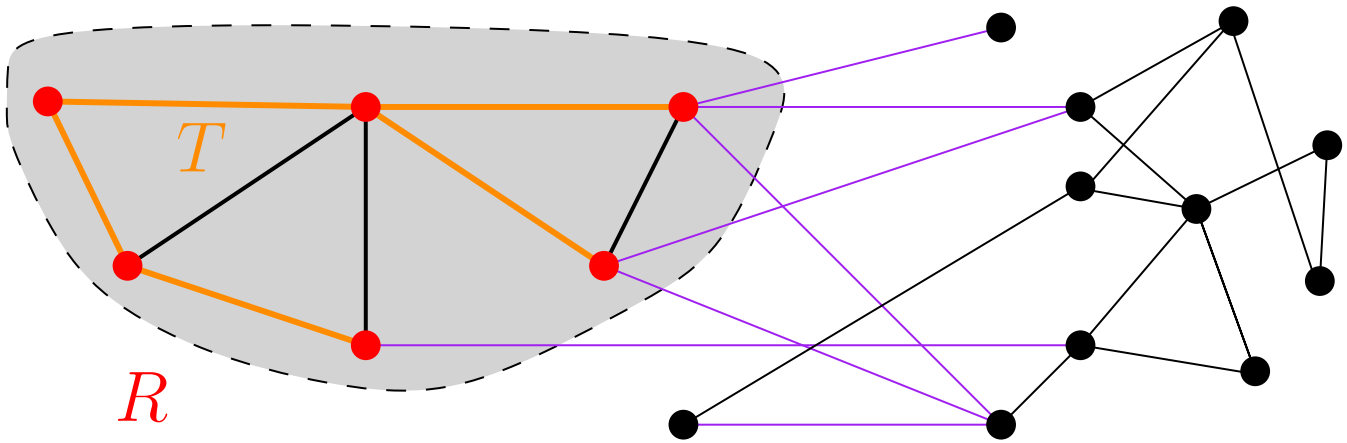
- Add v to R and e to T



Prim's Algorithm

Algorithm Prim's Algorithm for MST in $G = (V, E, w)$

$R \leftarrow \{s\}$ ▷ $s \in V$ an arbitrary vertex
$T \leftarrow \emptyset$ ▷ Begin with an empty tree
while $R \neq V$ **do**
  Get $e = (u, v)$, $u \in R, v \notin R$ with minimum $w(uv)$
  $T \leftarrow T \cup \{e\}$
  $R \leftarrow R \cup \{v\}$



Cuts in Graphs

A cut in G is a subset $S \subset V$

- Denoted as $[S, \overline{S}]$, $S = \emptyset$ and $S = V$ are trivial cuts
- An edge (u, v) is crossing the cut $[S, \overline{S}]$, if $u \in S$ and $v \in \overline{S}$
- Empty Cut Lemma:
  - A graph G is disconnected iff it has a cut with no crossing edge
- Double Crossing Lemma
  - If a cycle crosses a cut, then it has to cross at least twice
- Lonely Crossing Lemma
  - If e is the only edge crossing a cut $[S, \overline{S}]$, then it is not in any cycle
- The Blue Rule
  - If an edge $e \in E$ is the lightest edge crossing some cut $[S, \overline{S}]$, then e belongs to the MST of G

Prim's Algorithm: Correctness and Optimality

Algorithm Prim's Algorithm for MST in $G = (V, E, w)$

$R \leftarrow \{s\}$ ▷ $s \in V$ an arbitrary vertex
$T \leftarrow \emptyset$ ▷ Begin with an empty tree
while $R \neq V$ **do**
  Get $e = (u, v)$, $u \in R, v \notin R$ with minimum $w(uv)$
  $T \leftarrow T \cup \{e\}$
  $R \leftarrow R \cup \{v\}$

Correctness: T is a spanning tree of G

- T is a subgraph of G
- T is connected
- T has no cycle
- T is spanning

Optimality: T is the minimum spanning tree of G

Prim's Algorithm: Correctness and Optimality

Correctness: T is a spanning tree of G

After every iteration i, T is a spanning tree of $G|_R$

Proof: by induction on $|R|$ (iteration i)

- $|R| = |\{s\}| = 1$, and $T = \emptyset$ is a spanning tree of $\{s\}$
- If T spans R, $|R| = i$, then in the next iteration, we add a vertex to R with an edge connecting it to some vertex in R, hence T is a spanning tree of $G|_R$
- T has at most $n-1$ edges (max number of iterations)
- T has at least $n-1$ edges
- If we can't add an edge from $R \neq V$, then $[R, \overline{R}]$ is an empty cut
- Every time we add one edge from $[R, \overline{R}]$ so T has no cycle

Prim's Algorithm: Correctness and Optimality

Optimality: T is the minimum spanning tree of G

Proof follows from the cut property

If an edge $e \in E$ is the lightest edge crossing some cut $[S, \overline{S}]$, then e belongs to the MST of G

- In every iteration we added the lightest edge crossing the cut $[R, \overline{R}]$
- The blue rule guarantees this to be part of the MST
- Hence, by the blue rule, the output T is an optimal spanning tree