# Minimum Cost Spanning Trees: Prim's Algorithm

Madhavan Mukund

https://www.cmi.ac.in/~madhavan

Mathematics for Data Science 1
Week 12

# Minimum cost spanning tree (MCST)

- Weighted undirected graph,
  $G = (V, E), W : E \to \mathbb{R}$
  - $G$ assumed to be connected

# Minimum cost spanning tree (MCST)

- Weighted undirected graph,
  $G = (V, E), W : E \to \mathbb{R}$
  - $G$ assumed to be connected
- Find a minimum cost spanning tree
  - Tree connecting all vertices in $V$

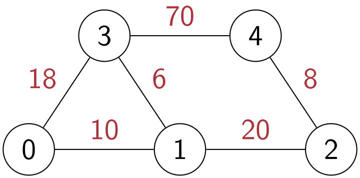
# Minimum cost spanning tree (MCST)

- Weighted undirected graph, $G = (V, E), W : E \to \mathbb{R}$
  - $G$ assumed to be connected
- Find a minimum cost spanning tree
  - Tree connecting all vertices in $V$
- Strategy
  - Incrementally grow the minimum cost spanning tree
  - Start with a smallest weight edge overall
  - Extend the current tree by adding the smallest edge from the tree to a vertex not yet in the tree

# Minimum cost spanning tree (MCST)

- Weighted undirected graph, $G = (V, E), W : E \to \mathbb{R}$
  - $G$ assumed to be connected
- Find a minimum cost spanning tree
  - Tree connecting all vertices in $V$
- Strategy
  - Incrementally grow the minimum cost spanning tree
  - Start with a smallest weight edge overall
  - Extend the current tree by adding the smallest edge from the tree to a vertex not yet in the tree

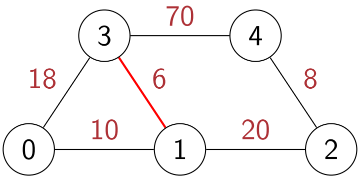
Example



# Minimum cost spanning tree (MCST)

- Weighted undirected graph, $G = (V, E), W : E \to \mathbb{R}$
  - $G$ assumed to be connected
- Find a minimum cost spanning tree
  - Tree connecting all vertices in $V$
- Strategy
  - Incrementally grow the minimum cost spanning tree
  - Start with a smallest weight edge overall
  - Extend the current tree by adding the smallest edge from the tree to a vertex not yet in the tree

Example

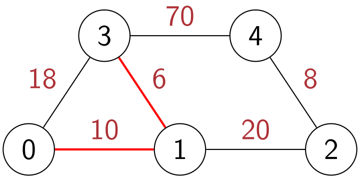



- Start with smallest edge, $(1, 3)$

# Minimum cost spanning tree (MCST)

- Weighted undirected graph, $G = (V, E), W : E \to \mathbb{R}$
  - $G$ assumed to be connected
- Find a minimum cost spanning tree
  - Tree connecting all vertices in $V$
- Strategy
  - Incrementally grow the minimum cost spanning tree
  - Start with a smallest weight edge overall
  - Extend the current tree by adding the smallest edge from the tree to a vertex not yet in the tree

Example

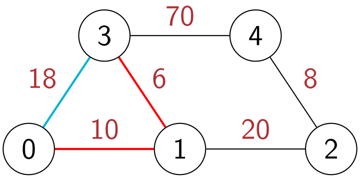



- Start with smallest edge, $(1, 3)$
- Extend the tree with $(1, 0)$

# Minimum cost spanning tree (MCST)

- Weighted undirected graph, $G = (V, E), W : E \to \mathbb{R}$
  - $G$ assumed to be connected
- Find a minimum cost spanning tree
  - Tree connecting all vertices in $V$
- Strategy
  - Incrementally grow the minimum cost spanning tree
  - Start with a smallest weight edge overall
  - Extend the current tree by adding the smallest edge from the tree to a vertex not yet in the tree

Example

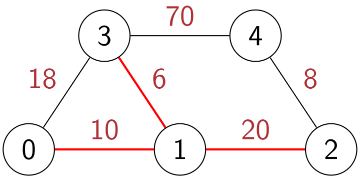



- Start with smallest edge, $(1, 3)$
- Extend the tree with $(1, 0)$
- Can't add $(0, 3)$, forms a cycle

# Minimum cost spanning tree (MCST)

- Weighted undirected graph, $G = (V, E), W : E \to \mathbb{R}$
  - $G$ assumed to be connected
- Find a minimum cost spanning tree
  - Tree connecting all vertices in $V$
- Strategy
  - Incrementally grow the minimum cost spanning tree
  - Start with a smallest weight edge overall
  - Extend the current tree by adding the smallest edge from the tree to a vertex not yet in the tree

Example

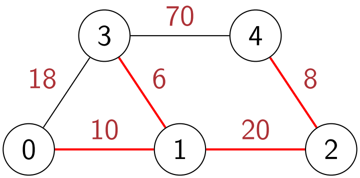



- Start with smallest edge, $(1, 3)$
- Extend the tree with $(1, 0)$
- Can't add $(0, 3)$, forms a cycle
- Instead, extend the tree with $(1, 2)$

# Minimum cost spanning tree (MCST)

- Weighted undirected graph, $G = (V, E), W : E \to \mathbb{R}$
  - $G$ assumed to be connected
- Find a minimum cost spanning tree
  - Tree connecting all vertices in $V$
- Strategy
  - Incrementally grow the minimum cost spanning tree
  - Start with a smallest weight edge overall
  - Extend the current tree by adding the smallest edge from the tree to a vertex not yet in the tree

Example




- Start with smallest edge, $(1, 3)$
- Extend the tree with $(1, 0)$
- Can't add $(0, 3)$, forms a cycle
- Instead, extend the tree with $(1, 2)$
- Extend the tree with $(2, 4)$

# Prim's algorithm

- $G = (V, E)$, $W : E \to \mathbb{R}$

# Prim's algorithm

- $G = (V, E)$, $W : E \to \mathbb{R}$
- Incrementally build an MCST
  - $TV \subseteq V$ : tree vertices, already added to MCST
  - $TE \subseteq E$ : tree edges, already added to MCST

# Prim's algorithm

- $G = (V, E)$, $W : E \to \mathbb{R}$
- Incrementally build an MCST
  - $TV \subseteq V$ : tree vertices, already added to MCST
  - $TE \subseteq E$ : tree edges, already added to MCST
- Initially, $TV = TE = \emptyset$

# Prim's algorithm

- $G = (V, E)$, $W : E \to \mathbb{R}$
- Incrementally build an MCST
  - $TV \subseteq V$ : tree vertices, already added to MCST
  - $TE \subseteq E$ : tree edges, already added to MCST
- Initially, $TV = TE = \emptyset$
- Choose minimum weight edge $e = (i, j)$
  - Set $TV = \{i, j\}$, $TE = \{e\}$ MCST

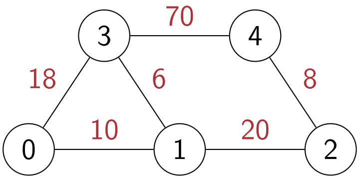
# Prim's algorithm

- $G = (V, E)$, $W : E \to \mathbb{R}$
- Incrementally build an MCST
  - $TV \subseteq V$ : tree vertices, already added to MCST
  - $TE \subseteq E$ : tree edges, already added to MCST
- Initially, $TV = TE = \emptyset$
- Choose minimum weight edge $e = (i, j)$
  - Set $TV = \{i, j\}$, $TE = \{e\}$ MCST
- Repeat $n - 2$ times
  - Choose minimum weight edge $f = (u, v)$ such that $u \in TV$, $v \notin TV$
  - Add $v$ to $TV$, $f$ to $TE$

# Prim's algorithm

- $G = (V, E)$, $W : E \to \mathbb{R}$
- Incrementally build an MCST
  - $TV \subseteq V$ : tree vertices, already added to MCST
  - $TE \subseteq E$ : tree edges, already added to MCST
- Initially, $TV = TE = \emptyset$
- Choose minimum weight edge $e = (i, j)$
  - Set $TV = \{i, j\}$, $TE = \{e\}$ MCST
- Repeat $n - 2$ times
  - Choose minimum weight edge $f = (u, v)$ such that $u \in TV$, $v \notin TV$
  - Add $v$ to $TV$, $f$ to $TE$

Example



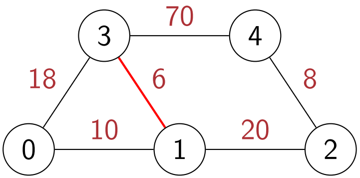

$TV = \emptyset$

$TE = \emptyset$

# Prim's algorithm

- $G = (V, E)$, $W : E \to \mathbb{R}$
- Incrementally build an MCST
  - $TV \subseteq V$ : tree vertices, already added to MCST
  - $TE \subseteq E$ : tree edges, already added to MCST
- Initially, $TV = TE = \emptyset$
- Choose minimum weight edge $e = (i, j)$
  - Set $TV = \{i, j\}$, $TE = \{e\}$ MCST
- Repeat $n - 2$ times
  - Choose minimum weight edge $f = (u, v)$ such that $u \in TV$, $v \notin TV$
  - Add $v$ to $TV$, $f$ to $TE$

Example



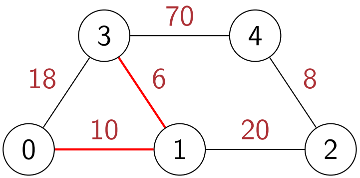

$TV = \{1, 3\}$

$TE = \{(1, 3)\}$

# Prim's algorithm

- $G = (V, E)$, $W : E \to \mathbb{R}$
- Incrementally build an MCST
  - $TV \subseteq V$ : tree vertices, already added to MCST
  - $TE \subseteq E$ : tree edges, already added to MCST
- Initially, $TV = TE = \emptyset$
- Choose minimum weight edge $e = (i, j)$
  - Set $TV = \{i, j\}$, $TE = \{e\}$ MCST
- Repeat $n - 2$ times
  - Choose minimum weight edge $f = (u, v)$ such that $u \in TV$, $v \notin TV$
  - Add $v$ to $TV$, $f$ to $TE$

Example



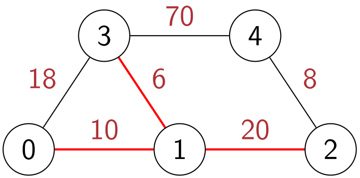

$TV = \{1, 3, 0\}$

$TE = \{(1, 3), (1, 0)\}$

# Prim's algorithm

- $G = (V, E)$, $W : E \to \mathbb{R}$
- Incrementally build an MCST
  - $TV \subseteq V$ : tree vertices, already added to MCST
  - $TE \subseteq E$ : tree edges, already added to MCST
- Initially, $TV = TE = \emptyset$
- Choose minimum weight edge $e = (i, j)$
  - Set $TV = \{i, j\}$, $TE = \{e\}$ MCST
- Repeat $n - 2$ times
  - Choose minimum weight edge $f = (u, v)$ such that $u \in TV$, $v \notin TV$
  - Add $v$ to $TV$, $f$ to $TE$

Example



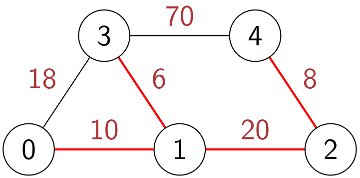

$TV = \{1, 3, 0, 2\}$

$TE = \{(1, 3), (1, 0), (1, 2)\}$

# Prim's algorithm

- $G = (V, E)$, $W : E \to \mathbb{R}$
- Incrementally build an MCST
  - $TV \subseteq V$ : tree vertices, already added to MCST
  - $TE \subseteq E$ : tree edges, already added to MCST
- Initially, $TV = TE = \emptyset$
- Choose minimum weight edge $e = (i, j)$
  - Set $TV = \{i, j\}$, $TE = \{e\}$ MCST
- Repeat $n - 2$ times
  - Choose minimum weight edge $f = (u, v)$ such that $u \in TV$, $v \notin TV$
  - Add $v$ to $TV$, $f$ to $TE$

Example




$$TV = \{1, 3, 0, 2, 4\}$$

$$TE = \{(1,3), (1,0), (1,2), (2,4)\}$$

# Correctness of Prim's algorithm

## Minimum Separator Lemma

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

# Correctness of Prim's algorithm

## Minimum Separator Lemma

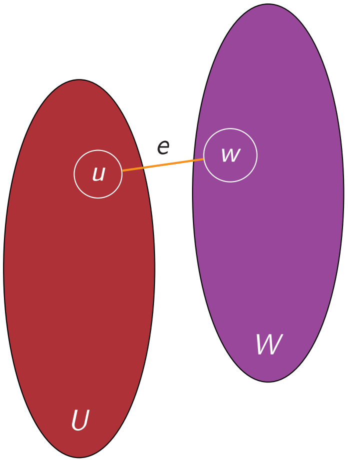
- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

- Assume for now, all edge weights distinct

# Correctness of Prim's algorithm

## Minimum Separator Lemma

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$



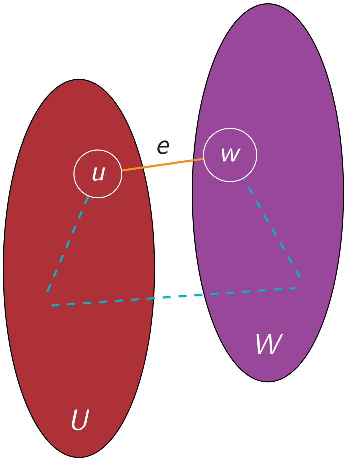

- Assume for now, all edge weights distinct
- Let $T$ be an MCST, $e \notin T$

## Correctness of Prim's algorithm

### Minimum Separator Lemma

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

- Assume for now, all edge weights distinct
- Let $T$ be an MCST, $e \notin T$
- $T$ contains a path $p$ from $u$ to

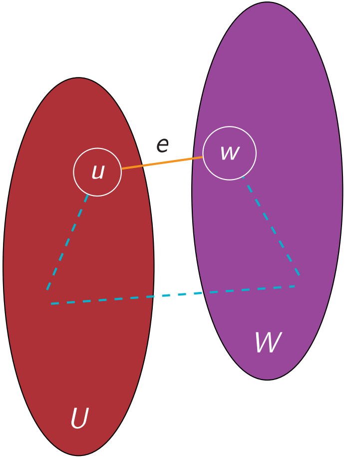


# Correctness of Prim's algorithm

## Minimum Separator Lemma

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

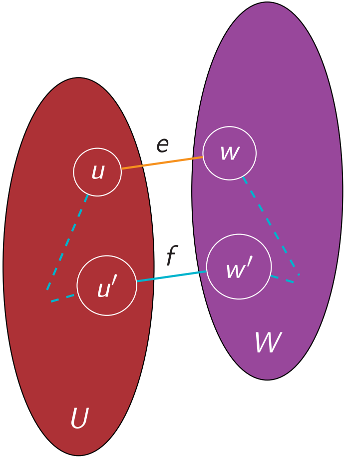
- Assume for now, all edge weights distinct
- Let $T$ be an MCST, $e \notin T$
- $T$ contains a path $p$ from $u$ to
  - $p$ starts in $U$, ends in $W$



# Correctness of Prim's algorithm

## Minimum Separator Lemma

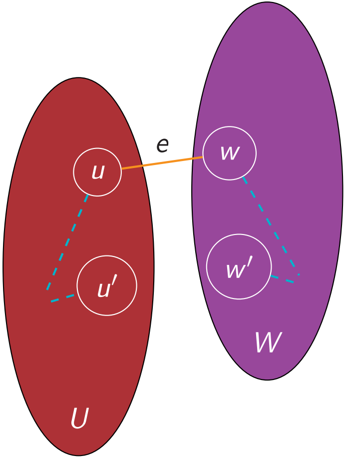
- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$




- Assume for now, all edge weights distinct
- Let $T$ be an MCST, $e \notin T$
- $T$ contains a path $p$ from $u$ to $w$
  - $p$ starts in $U$, ends in $W$
  - Let $f = (u', w')$ be the first edge on $p$ crossing from $U$ to $W$

## Minimum Separator Lemma

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

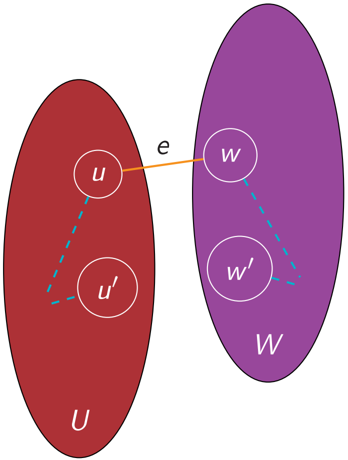



- Assume for now, all edge weights distinct
- Let $T$ be an MCST, $e \notin T$
- $T$ contains a path $p$ from $u$ to
  - $p$ starts in $U$, ends in $W$
  - Let $f = (u', w')$ be the first edge on $p$ crossing from $U$ to $W$
  - Drop $f$, add $e$ to get a cheaper spanning tree

# Correctness of Prim's algorithm

## Minimum Separator Lemma

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

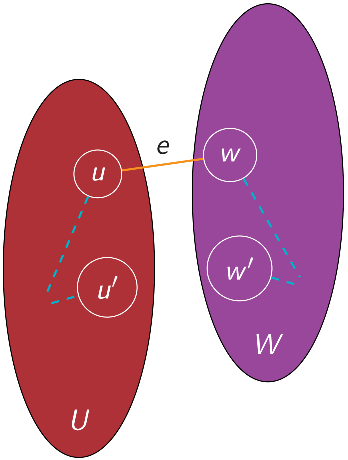



- Assume for now, all edge weights distinct
- What if two edges have the same weight?

## Minimum Separator Lemma

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

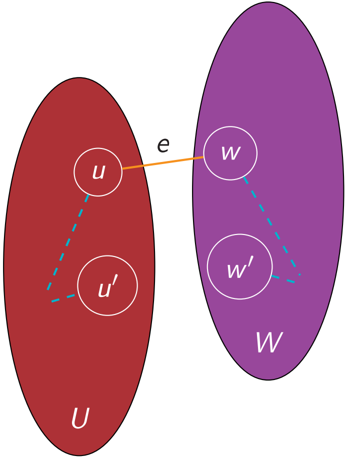
- Assume for now, all edge weights distinct
- What if two edges have the same weight?
- Assign each edge a unique index from $0$ to $m - 1$



# Correctness of Prim's algorithm

## Minimum Separator Lemma

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$




- Assume for now, all edge weights distinct
- What if two edges have the same weight?
- Assign each edge a unique index from $0$ to $m - 1$
- Define $(e, i) < (f, j)$ if $W(e) < W(j)$ or $W(e) = W(j)$ and $i < j$

# Correctness of Prim's algorithm

## Minimum Separator Lemma

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

# Correctness of Prim's algorithm

**Minimum Separator Lemma**

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

- In Prim's algorithm, $TV$ and $W = V \setminus TV$ partition $V$

# Correctness of Prim's algorithm

## Minimum Separator Lemma

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

- In Prim's algorithm, $TV$ and $W = V \setminus TV$ partition $V$
- Algorithm picks smallest edge connecting $TV$ and $W$, which must belong to every MCST

# Correctness of Prim's algorithm

**Minimum Separator Lemma**

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

- In Prim's algorithm, $TV$ and $W = V \setminus TV$ partition $V$
- Algorithm picks smallest edge connecting $TV$ and $W$, which must belong to every MCST

- In fact, for any $v \in V$, $\{v\}$ and $V \setminus \{v\}$ form a partition

# Correctness of Prim's algorithm

**Minimum Separator Lemma**

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

- In Prim's algorithm, $TV$ and $W = V \setminus TV$ partition $V$
- Algorithm picks smallest edge connecting $TV$ and $W$, which must belong to every MCST

- In fact, for any $v \in V$, $\{v\}$ and $V \setminus \{v\}$ form a partition
- The smallest weight edge leaving any vertex must belong to every MCST

# Correctness of Prim's algorithm

## Minimum Separator Lemma

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

- In Prim's algorithm, $TV$ and $W = V \setminus TV$ partition $V$
- Algorithm picks smallest edge connecting $TV$ and $W$, which must belong to every MCST
- In fact, for any $v \in V$, $\{v\}$ and $V \setminus \{v\}$ form a partition
- The smallest weight edge leaving any vertex must belong to every MCST
- We started with overall minimum cost edge

# Correctness of Prim's algorithm

## Minimum Separator Lemma

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

- In Prim's algorithm, $TV$ and $W = V \setminus TV$ partition $V$
- Algorithm picks smallest edge connecting $TV$ and $W$, which must belong to every MCST

- In fact, for any $v \in V$, $\{v\}$ and $V \setminus \{v\}$ form a partition
- The smallest weight edge leaving any vertex must belong to every MCST
- We started with overall minimum cost edge
- Instead, can start at any vertex $v$, with $TV = \{v\}$ and $TE = \emptyset$

# Correctness of Prim's algorithm

**Minimum Separator Lemma**

- Let $V$ be partitioned into two non-empty sets $U$ and $W = V \setminus U$
- Let $e = (u, w)$ be the minimum cost edge with $u \in U$, $w \in W$
- Every MCST must include $e$

- In Prim's algorithm, $TV$ and $W = V \setminus TV$ partition $V$
- Algorithm picks smallest edge connecting $TV$ and $W$, which must belong to every MCST

- In fact, for any $v \in V$, $\{v\}$ and $V \setminus \{v\}$ form a partition
- The smallest weight edge leaving any vertex must belong to every MCST
- We started with overall minimum cost edge
- Instead, can start at any vertex $v$, with $TV = \{v\}$ and $TE = \emptyset$
- First iteration will pick minimum cost edge from $v$

# Summary

- Prim's algorithm grows an MCST starting with any vertex
- At each step, connect one more vertex to the tree using minimum cost edge from inside the tree to outside the tree
- Correctness follows from Minimum Separator Lemma