ISE 426
Optimization models and applications

Lecture 19 — November 6, 2014

- Minimum Spanning Tree
- Network Design

Announcements:

- Case studies will be assigned after the quiz. Form into groups of 4 or 5.

The Minimum Spanning Tree Problem

- Given a graph $G = (V, E)$ and a function $c : E \to \mathbb{R}_+$, find a subgraph of G (that is, a subset of E) such that all (pairs of) nodes of V are connected by at least one path.
- This is a problem very well known in Optimization as the Minimum Spanning Tree (MST) problem
- It can be solved very easily by first sorting the edges according to non-decreasing c. Two algorithms:

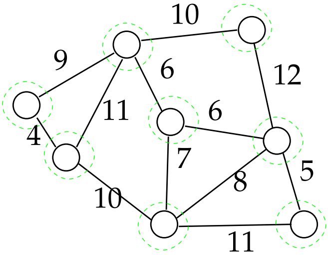
Prim: Create a tree starting from a single node (of your choice), including one edge at a time.

Kruskal: Include one edge at a time in that order, ensuring no cycles are created in the graph.

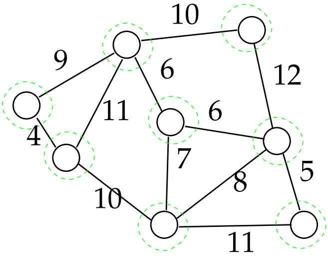
How Kruskal works

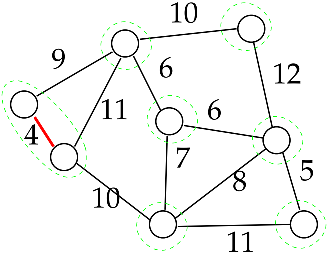
Sort the edges by weight from smallest to largest. Add one edge at a time, as long as they do not create a cycle. If an edge creates a cycle - skip it.

How Kruskal works

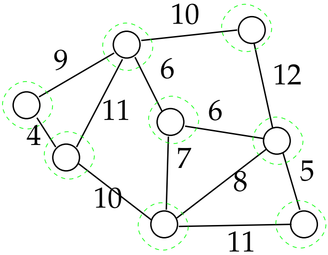


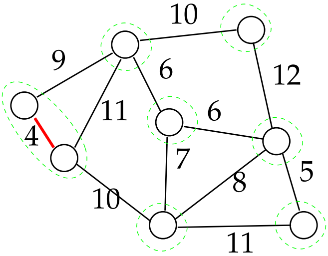
How Kruskal works

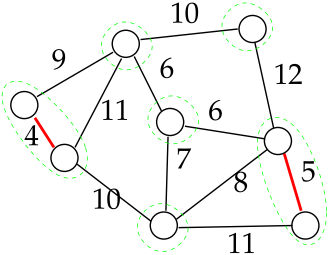




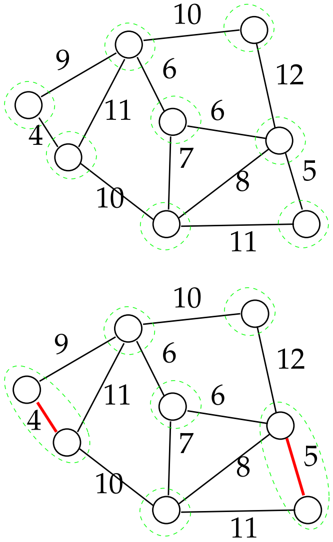
How Kruskal works



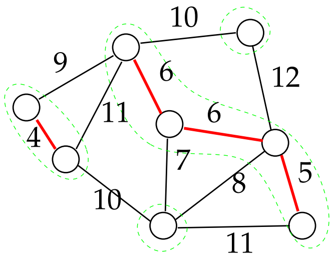




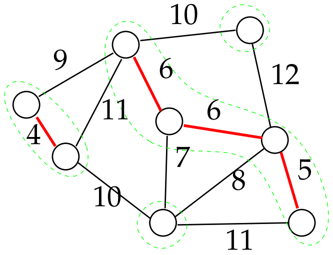
How Kruskal works

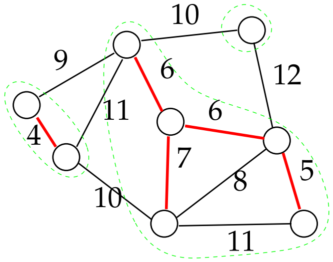


How Kruskal works (cont.)

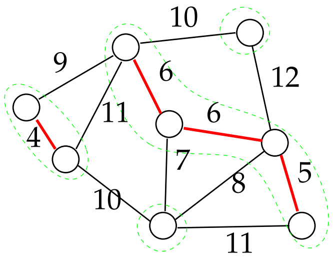


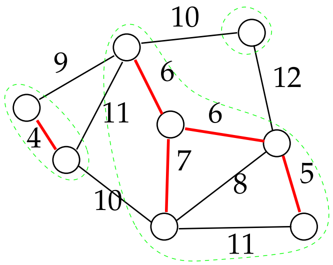
How Kruskal works (cont.)

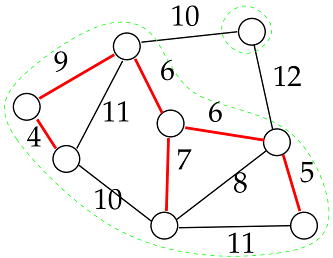




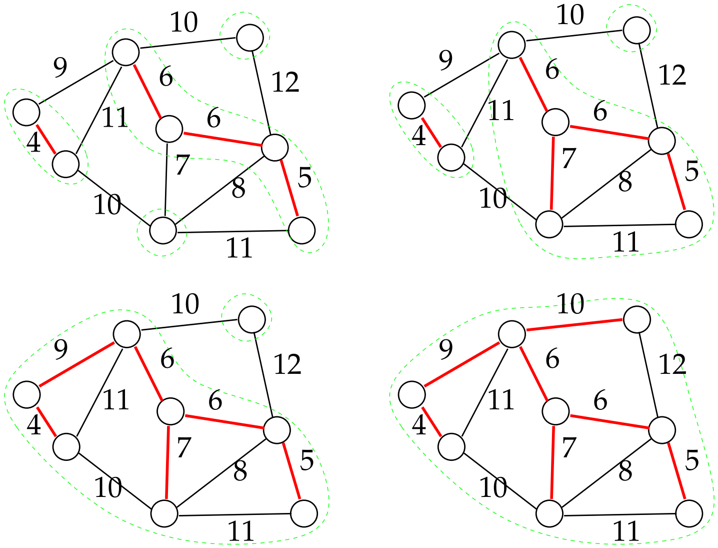
How Kruskal works (cont.)







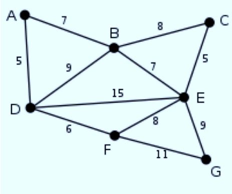
How Kruskal works (cont.)



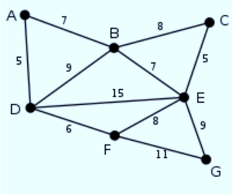
How Prim works

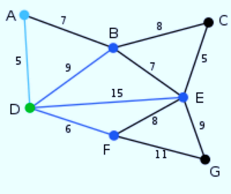
Pick any node a. Let the current set $S = \{a\}$. Consider all nodes connectedto S by one edge. Pick a node whose is closest to any node in S. Add this node to S, procceed in the same manner until all nodes are added.

How Prim works

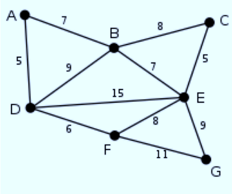


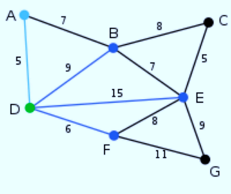
How Prim works

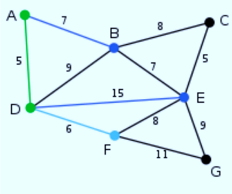




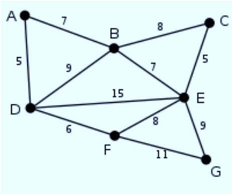
How Prim works

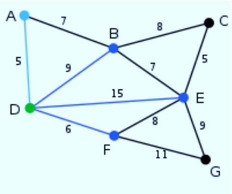


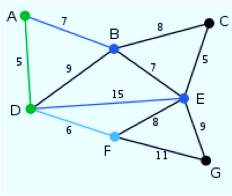


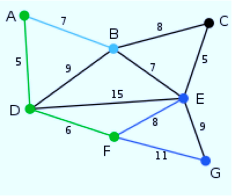


How Prim works

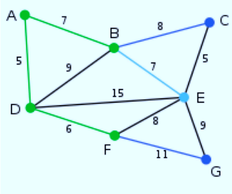




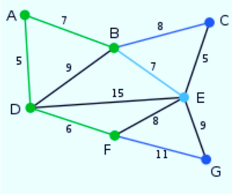


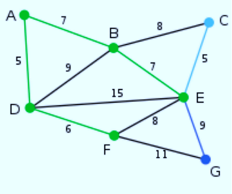


How Prim works (cont.)

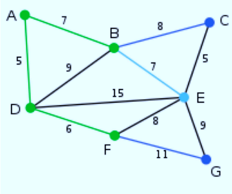


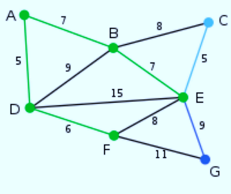
How Prim works (cont.)

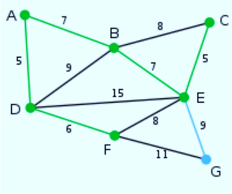




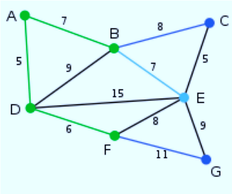
How Prim works (cont.)

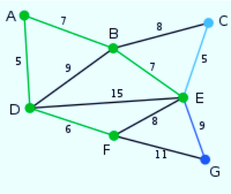


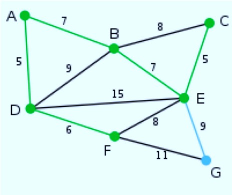


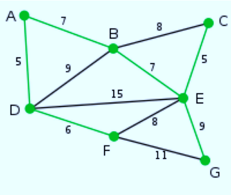


How Prim works (cont.)









How do we model this as an Optimization model?

Given a graph $G = (V, E)$, determine a subset $S \subseteq E$ such that

- S is a tree (how do you write this constraint?)
- The edges in S *span* the whole graph

i.e. The edges in S connect any two nodes of V

We need a subset of $E \Rightarrow$ a binary variable x_{ij} for each $\{i, j\} \in E$.

Graph Theory tells us that any spanning tree for a graph of n nodes has exactly $n - 1$ edges.

$\Rightarrow$ Does that mean that

$$\sum_{\{i,j\} \in E} x_{ij} = |V| - 1$$

is enough?

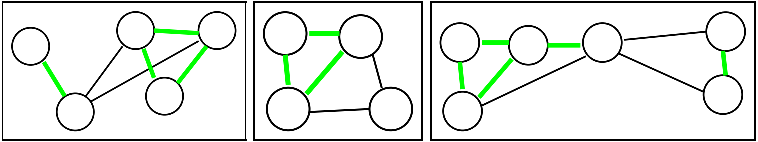
Too good to be true

$$
\begin{array}{ll}
\min & \sum_{\{i,j\}\in E} c_{ij} x_{ij} \\
 & \sum_{\{i,j\}\in E} x_{ij} = |V| - 1 \\
 & x_{ij} \in \{0,1\} \quad \forall \{i,j\} \in E
\end{array}
$$

Too good to be true

$$\begin{array}{ll} \min & \sum_{\{i,j\}\in E} c_{ij}x_{ij} \\ & \sum_{\{i,j\}\in E} x_{ij} = |V| - 1 \\ & x_{ij} \in \{0,1\} \quad \forall \{i,j\} \in E \end{array}$$

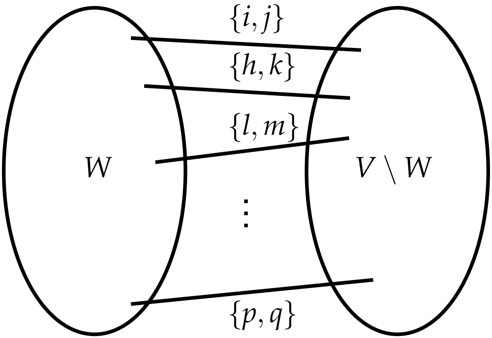
Optimal solutions of this model can be (green edges $\in S$):



None of them is a tree. None of them spans the graph.

Formulation #1: cuts

- "There is a path between any two nodes of G". That is,
- "There is no node pair that is not connected". Or,
- "If we split the set of nodes V in two parts W and $V \setminus W$, they must be connected."



Formulation #1: cuts (cont'd)

- Partition V into two subsets: W and $V \setminus W$.
- Would you accept a solution that contains no edge between W and $V \setminus W$? No!

$$
\begin{array}{lll}
\min & \sum_{\{i,j\} \in E} c_{ij} x_{ij} & \\
 & \sum_{\{i,j\} \in E : i \in S, j \in V \setminus S} x_{ij} \geq 1 & \forall S \subset V : S \neq \emptyset \\
 & x_{ij} \in \{0, 1\} & \forall \{i, j\} \in E
\end{array}
$$

This is a "weak" formulation and gives bad lower bounds.

Formulation #2: flow

Spanning $\equiv$ "There is a path between any two nodes s, t of G".

- Ensuring this would avoid those wrong solutions, as they would violate at least one of the *connection* constraints.
- We could use two sets of variables:

  flow: define a path between any two nodes in V
  edge: one if the edge is used by at least one flow, zero otherwise

- We need a set of flow variables for each pair (s, t) of nodes
- Binary variables x_{ij}: one if there is (even just one unit of) flow on $\{i, j\}$, doesn't matter if $i \to j$ or $j \to i$

Formulation #2: flow

- All pairs of nodes: $n(n-1)$
- Enough to connect a node r of our choice to all other nodes

$\Rightarrow$ Ensure that there be $n-1$ paths: from r to all other nodes

- $s \to t \quad \equiv \quad (s \to r) + (r \to t)$
- Use flow variables for each node $k \in V \setminus \{r\}$ for path $r \to k$
- Include in S the edges of E that host at least one path

i.e. If an edge $\{i, j\}$ is used by a path, then $x_{ij} = 1$

Formulation #2: flow

Variables:

- f_{ij}^k flow variables (binary): flow in both directions $i \to j$ and $j \to i$ on edge $\{i, j\}$ for a path that is going from r to k
- x_{ij} binary variables for any edge $\{i, j\} \in E$: 1 if $\{i, j\}$ is included in the solution, 0 otherwise

Constraints:

1. Conservation of flow
2. No flow on edge $\{i, j\}$ for any destination k if it is not included in the solution

Formulation #2: flow

$$
\begin{array}{lll}
\min & \sum_{\{i,j\}\in E} c_{ij}x_{ij} & \\
 & \sum_{j:\{i,j\}\in E}(f_{ij}^k - f_{ji}^k) = 0 & \forall i \in V \setminus \{k\}, \forall k \in V \setminus \{r\} \\
 & \sum_{j:\{r,j\}\in E} f_{rj}^k = 1 & \forall k \in V \setminus \{r\} \\
 & f_{ij}^k \leq x_{ij} & \forall \{i,j\} \in E, \forall k \in V \setminus \{r\} \\
 & f_{ji}^k \leq x_{ij} & \forall \{i,j\} \in E, \forall k \in V \setminus \{r\} \\
 & x_{ij} \in \{0,1\} & \forall \{i,j\} \in E, \forall k \in V \setminus \{r\} \\
 & f_{ij}^k \in [0,1] & \forall \{i,j\} \in E
\end{array}
$$

Q.: Can we replace $f_{ij}^k \leq x_{ij}$ and $f_{ji}^k \leq x_{ij}$ with

$$f_{ij}^k + f_{ji}^k \leq x_{ij}$$

Network design

We want to design a telecommunication network that connects a set of cities using their railway network (built long ago and with some pipes along the rails that are basically unused), so as to spend nothing on digging costs.

- The railway network can be defined as a graph $G = (V, E)$.
- Installing a unit of network capacity on edge $\{i, j\} \in E$ (for instance, a fiber optics cable) costs c_{ij}

  (it depends on the distance between cities i and j)
- Once installed, it carries up to U Mb/s of data traffic in both directions (U Mb/s $i \to j$ and U Mb/s $j \to i$)
- For any ordered pair (k, l) of cities in a set P, there is a planned amount of data traffic from k to l denoted as d_{kl}.

$\Rightarrow$ Determine how much capacity to install on all edges of G in order to satisfy all traffic demands $(k, l) \in P$ while minimizing total installation cost.

Network design

$$
\begin{array}{lll}
\min & \sum_{\{i,j\} \in E} c_{ij} y_{ij} & \\
& \sum_{j \in V : \{i,j\} \in E} (x_{ij}^{kl} - x_{ji}^{kl}) = 0 & \forall i \in V : k \neq i \neq l, \forall (k,l) \in V^2 \\
& \sum_{j \in V : \{k,j\} \in E} (x_{kj}^{kl} - x_{jk}^{kl}) = 1 & \forall (k,l) \in P \\
& \sum_{(k,l) \in P} d_{kl} x_{ij}^{kl} \leq U y_{ij} & \forall \{i,j\} \in E \\
& \sum_{(k,l) \in P} d_{kl} x_{ji}^{kl} \leq U y_{ij} & \forall \{i,j\} \in E \\
& x_{ij}^{kl}, x_{ji}^{kl} \in \{0,1\} & \forall \{i,j\} \in E, \forall (k,l) \in P \\
& y_{ij} \in \mathbb{Z} & \forall \{i,j\} \in E
\end{array}
$$