Lesson 10, Combinatorial Models: Minimum Spanning Tree

1   Combinatorial Models

Many optimization problems are naturally modeled by a combinatorial structure, such as a **graph**. For the next few weeks, we will be talking about several famous combinatorial optimziation problems: *minimum spanning tree*, *traveling salesperson*, and *vehicle routing problem.*

Combinatorial optimization problems are usually more general than the network problems we've seen before as they operate on a **graph** instead of a **network**. Recall that a network is a special type of graph.
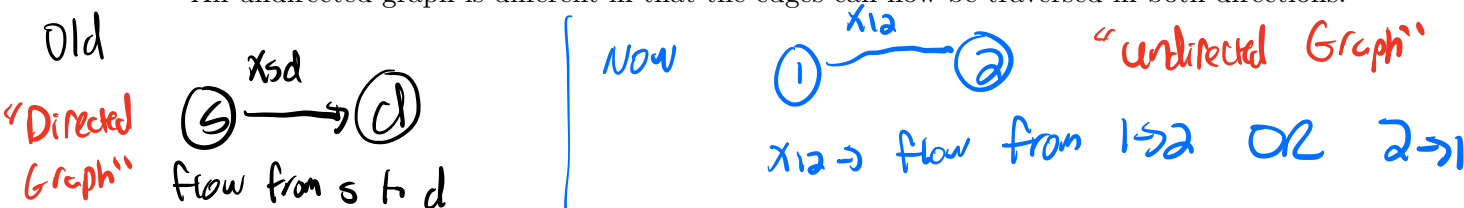
For now, we will develop integer programs to model these problems. Integer programming is appropriate for traveling salesperson and vehicle routing problems (they're NP-Complete which means **really** hard to solve); but, like the other networks we have done so far, there are more efficient ways to handle MST.
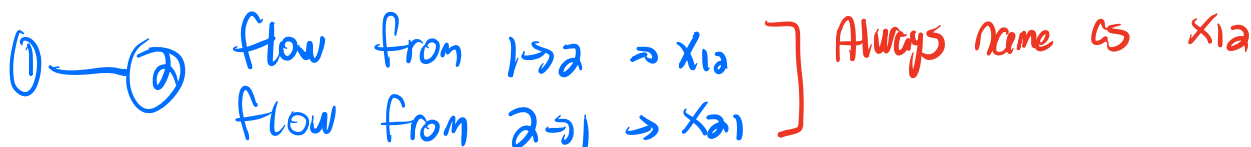
2   Graph Terminology

Suppose $G = (V, E)$ is an *undirected* graph. (So far we have worked with directed graphs.) Recall that:

- V is the set of vertices or nodes $\leftrightarrow$ or N

- E is the set of edges

An undirected graph is different in that the edges can now be traversed in both directions.

Old
"Directed Graph"

x_{sd}
$\overset{x_{sd}}{s \rightarrow d}$
flow from s to d

Now
x_{12}
$1 \longrightarrow 2$ "undirected Graph"
$x_{12} \Rightarrow$ flow from $1 \rightarrow 2$ OR $2 \rightarrow 1$

As a result, it's good to be consistent in naming the edges. For example, using our old style of naming, an edge connecting nodes 1 and 2 could be both $x_{1,2}$ and $x_{2,1}$. As a result, we will always name edges from lower index to higher index.
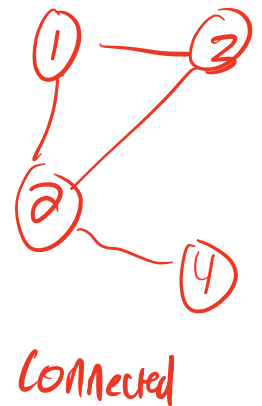
$1 \longrightarrow 2$
$\left.\begin{array}{l} \text{flow from } 1 \rightarrow 2 \Rightarrow x_{12} \\ \text{flow from } 2 \rightarrow 1 \Rightarrow x_{21} \end{array}\right]$ Always name as x_{12}

$\left.\begin{array}{c} 1 — 2 \\ 3 — 4 \end{array}\right]$ 5 undirected edges $x_{12}, x_{13}, x_{14}, x_{24}, x_{34}$

Path → Sequence of edges to take from a to b

Graph G is **connected** if for *every* pair of vertices a, $b \in V$, there exists a **path** of edges in G connecting a and b.
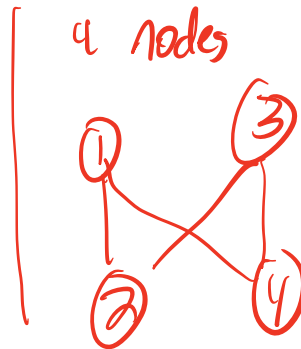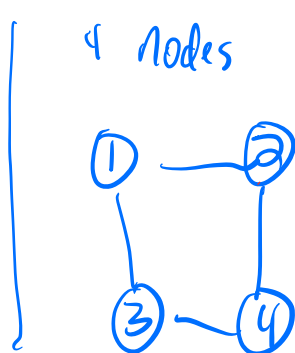
Example:

Connected
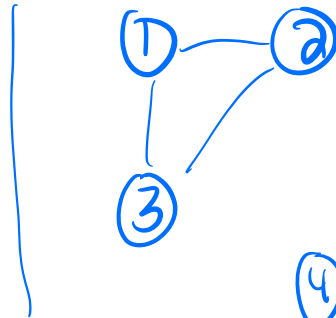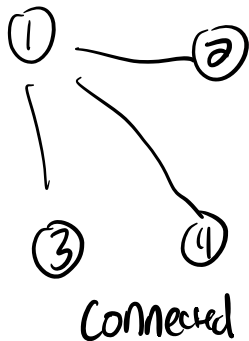
Disconnected
no path
between
$\{1,2,3\}$
and
$\{4,5,6\}$

Connected

A **cycle** is a closed path of nodes meaning that the first and last node in the path is the same vertex. **Its a loop in a graph**
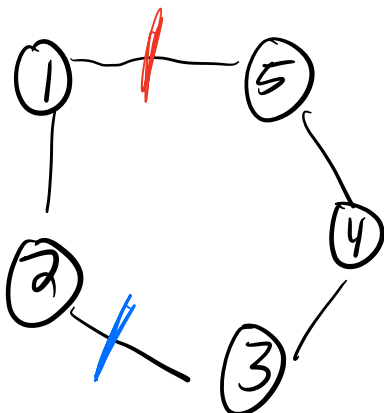
Example:

3 nodes

4 nodes

4 nodes

If a cycle
has n. nodes
a cycle has
exactly n edges

If G is a connected graph that contains a **cycle**, then the removal of a single edge from the cycle does not destroy the connectivity of G.

1. Illustrate a cycle and prove the fact that if G has a cycle, removing any edge from the cycle does not make the graph disconnected.

Connected ⇒ path between every two nodes

Removing Red edge doesn't disconnect any two nodes

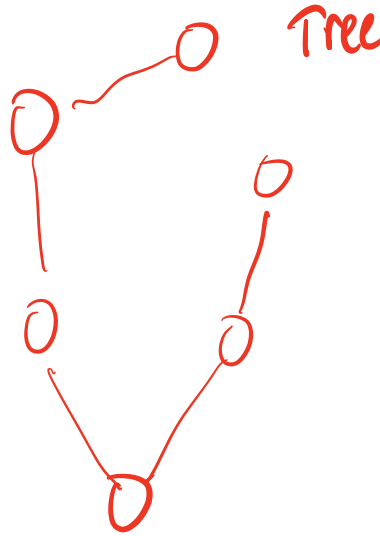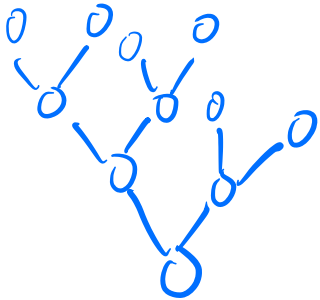Removing Blue edge does not disconnect graph

2

Removing both edges disconnects the graph

In general → Removing any 1 edge does not disconnect a cycle
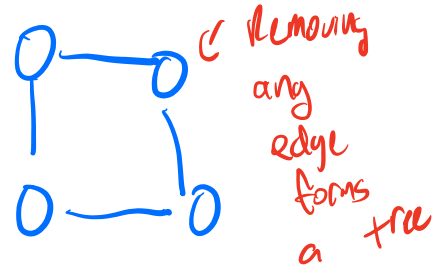Removing any 2 edges does disconnect a cycle

A connected graph that contains no cycles is called a **tree**.
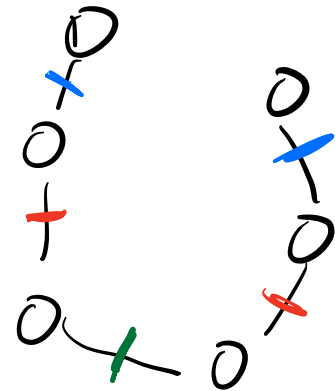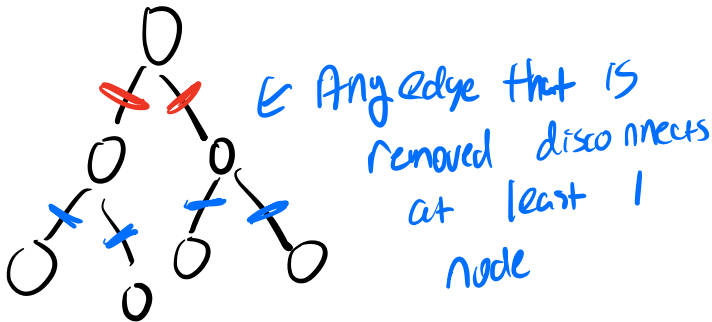
Tree

Tree

Removing 1 edge
from a cycle
forms a tree

← Removing
any
edge
forms
a tree

Trees are **minimally connected**: if we remove any edge from a tree, the resulting graph is disconnected.

2. Convince yourself of the previous fact by drawing a tree on 6 vertices. Are there any edges you can remove from the graph without losing connectivity?

← Any edge that is
removed disconnects
at least 1
node

If a tree has n nodes it should have n-1 edges

Side note: Trees are really important in tons of real world problems. For example, every time you make a Google search or use GoogleMaps an algorithm is called where one of the key parts is analyzing a huge tree.

A Spanning tree is a tree that touches every node in the graph

> A **spanning tree** of a graph G is a subset of the edges of G that form a *tree that connectes every vertex in V* (i.e., it spans the nodes of G).

3. Consider the graph G on 5 vertices below.

(a) Draw two different spanning trees of G.

① (blue drawing)
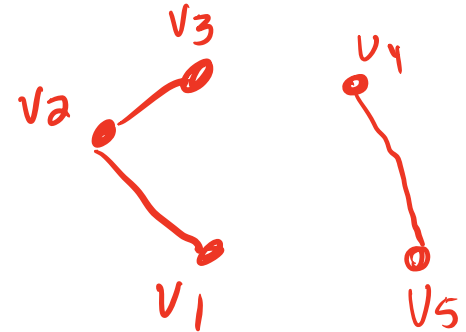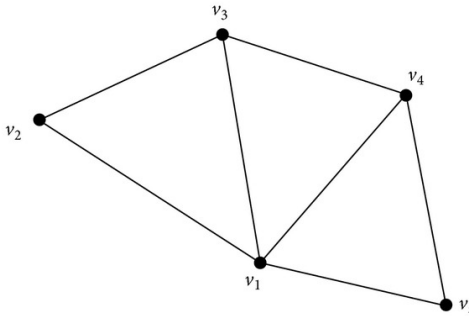
Not a spanning tree, this is a "forrest"
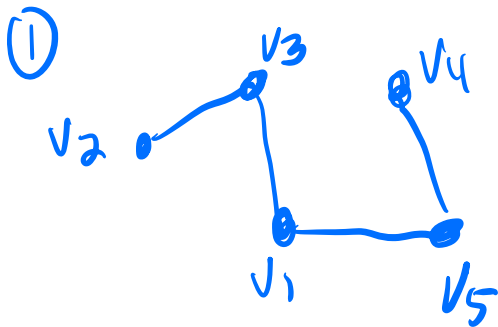
Tree ✓
Every node ✓
Spanning Tree
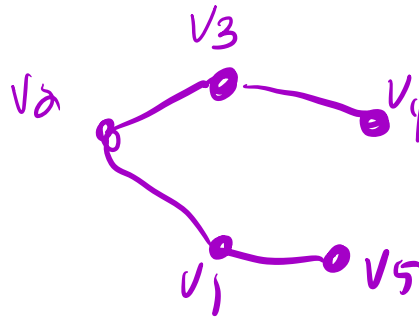
Tree ✓
every node ✓
Spanning Tree

(b) How many edges does each spanning tree have? $\boxed{4}$

(c) List the elements (edges) of the spanning tree.

Blue: $(2,3)$, $(1,3)$
$(1,5)$, $(4,5)$

Purple: $(1,5)$, $(1,2)$, $(2,3)$, $(3,4)$

(d) How many edges does a spanning tree of a graph G with vertex set V have in general?

$|V|-1$ edges in a spanning tree

4

3   Minimum Spanning Tree (MST) Problem

> Associate a cost c_{ij} with every edge $(i, j) \in E$. The problem of finding the *spanning tree* of graph G with *minimum edge cost* is known as the **minimum spanning tree problem**.

- Real world applications of the minimum spanning tree problem include planning road, electrical, data, phone, and water networks.

4. Solve the minimum cost spanning tree problem for the graph shown below. (The numbers on the edges are the edge costs.)
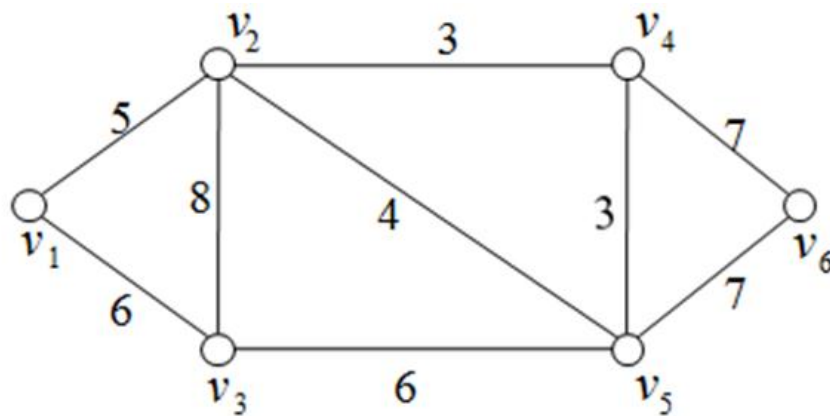
Minimum spanning tree $T =$
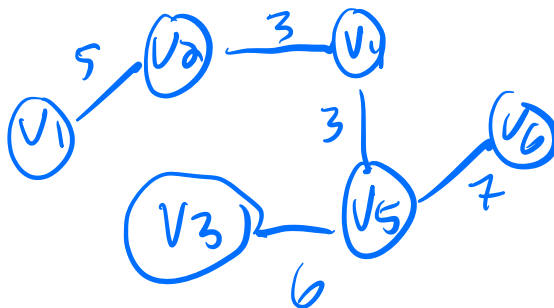
Total cost: 24

If you want to learn about algorithms for this prob take SA403

Next time→ IP formulation of MST

4   MST IP Formulation

It is more efficient to solve a min-cost spanning tree problem via an algorithm (like maybe from SA403), it is still instructive to model it using IP. Specifically because fully understanding the MST problem formulation will be helpful in understanding traveling salesperson and vehicle routing.

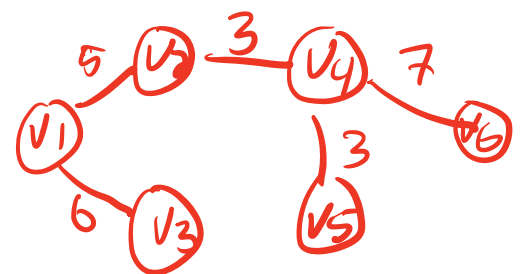A local phone company is interested in laying cable from the main road (where the Main switch is located) to a new housing subdivision, and wants to do so in the least expensive way. It has the option of laying cable from the road to any house, or it can lay cable between the houses. Each house must be connected through some path to the road. The following matrix gives the total cost of laying cable between any two locations, where the first location is the main road.

$$C = \begin{array}{c c} & \begin{array}{c c c c c c} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \left[\begin{array}{c c c c c c} 0 & 25 & 25 & 15 & 10 & 30 \\ 25 & 0 & 10 & 25 & 20 & 15 \\ 25 & 10 & 0 & 20 & 30 & 15 \\ 15 & 25 & 20 & 0 & 15 & 20 \\ 10 & 20 & 30 & 15 & 0 & 20 \\ 30 & 15 & 15 & 20 & 20 & 0 \end{array} \right] \end{array}$$

30 ⇒ distance from 1→6 or 6→1

How should the phone company connect the houses to the road in order to minimize its total cost?

5. Draw a graph to represent this problem below.
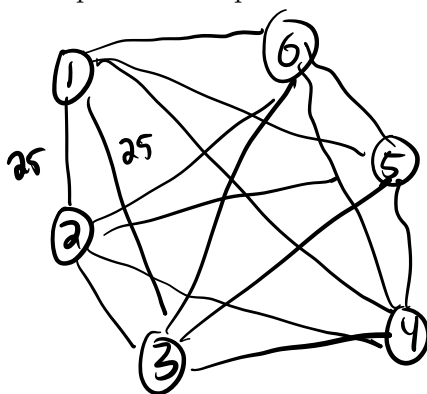
Each edge represents laying wire between 2 houses

6. Define the **sets**, **variables**, and **parameters** for the problem.

Goal: Find the edges to make a MST

Sets: N ⇒ nodes ∈ V for vertices in book
E ⇒ Edges

Variables: which edges are part of tree?
$x_{ij} = 1$ *if edge* (i,j) *is part of tree* 0 *o.w.*
for all $(i,j) \in E$

Parameters: c_{ij} ⇒ *cost of each edge* (i,j) *for all* $(i,j) \in E$

7. Write the objective function in both concrete and parameterized form.

$$\min: 25\, x_{12} + 25\, x_{13} + \cdots + 20\, x_{56}$$

$$\min \sum_{(i,j) \in E} c_{ij}\, x_{ij}$$

8. Minimum spanning trees must touch every vertex. Write a concrete constraint that ensures that vertex 1 is covered by the spanning tree returned by the solver. Do the same for vertices 2 and 3. Finally, write an parameterized class of constraints that ensures that every vertex is covered by an edge. (This should remind you of set covering hopefully!)

node 1: $\underbrace{x_{12} + x_{13} + x_{14} + x_{15} + x_{16}}_{\text{each edge touching node 1}} \geq 1$

$x_{in} \, \bigcirc \, x_{nj}$

node 2: $x_{12} + x_{23} + x_{24} + x_{25} + x_{26} \geq 1$

Any node: $\underset{(i,n) \in E}{\sum} x_{in} + \underset{(n,j) \in E}{\sum} x_{nj} \geq 1$ for $n \in N$

9. Spanning trees contain $\boxed{|N|-1}$ edges. Write a concrete constraint that ensures that the solution returned by the solver has exactly $\boxed{5}$ edges. Convert this constraint to parameterized form.

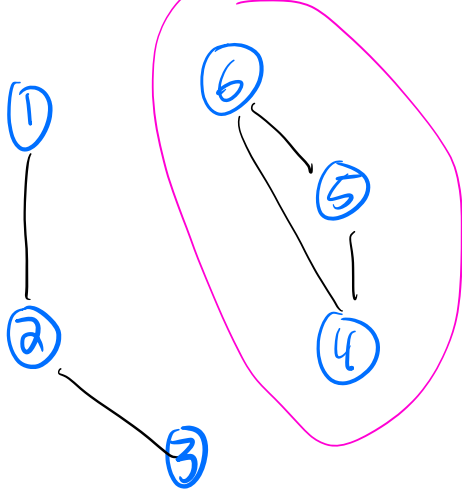$$x_{12} + x_{13} + x_{14} + \cdots + x_{56} = 5$$

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

$N = \{1, 2, 3, 4, 5, 6\}$ $|N| = $ Cardinality $= \#$ of elements

10. At this point, we have constraints that enforce:

at least ↗

- Every node in the graph is connected to one edge] **set covering**
- The graph has exactly $n-1$ edges $n =$ # of nodes

Is there a solution that satisfies these two constraints while not producing a spanning tree?

Goal: Draw a picture that satisfies these constraints but is not a spanning tree

Not a tree b/c contains a cycle

Need a constraint to prevent this cycle

11. Write a concrete constraint that prevents the graph that you sketched above. Then write a set of parameterized constraints that prevents ANY graphs of this kind from being returned by the solver. (There is one constraint for every subset of vertices of G.)

$$X_{45} + X_{46} + X_{56} \leq 3 - 1 = 2$$

↗
Prevents a cycle on nodes 4,5,6

To eliminate all cycles of size 3, I need $\binom{6}{3}$ constraints → 20

To eliminate all cycles of size 4, I need $\binom{6}{4}$ constraints → 15

For all subgraphs on 3 nodes, no cycles on this graph → $$\sum_{\substack{(i,j) \in E \\ i \in S, j \in S}} X_{ij} \leq |S| - 1 \quad \text{for all } S \subseteq N$$
$$|S| \geq 3$$

> These constraints above are called **cycle-elimination** or often **subtour-elimination** constraints. In real-life applications, there are usually way too many possible "cycle-elimination" constraints to include them all in a model. In practice, these constraints may be added iteratively to eliminate cycles in a solution returned by the solver. We will do something like this in the context of vehicle routing.

8

For all subsets S of nodes N, no cycle is allowed on the nodes of S.

For graphs of size 3 you need $\binom{n}{3}$ constraints of this form.

12. Combining all of the steps above, write the full parameterized model for MST.

Sets
let N be the set of nodes
let E be the set of edges

Var (cont.)

let $x_{ij} = 1$ if edge (i,j) is included in tree $\forall (i,j) \in E$

Parameters

c_{ij} cost of edge (i,j) $\forall (i,j) \in E$

Obj

min cost: $\sum_{(i,j) \in E} c_{ij} x_{ij}$

Constraints

$$\sum_{(i,n) \in E} x_{in} + \sum_{(n,j) \in E} x_{nj} \geq 1 \quad \text{for } n \in N$$

$|N|$ constraints
1 for each node

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

1 constraint

$$\sum_{\substack{(i,j) \in E: \\ i \in S, j \in S}} x_{ij} \leq |S| - 1 \quad \forall \quad S \subseteq N \quad |S| \geq 3$$

$\binom{N}{3} + \binom{N}{4} + \cdots$

Total # is about $2^{|N|}$

9

↗ can't implement all of these