Lesson 11. Traveling Sales(person) Problem (TSP)

1 Today...

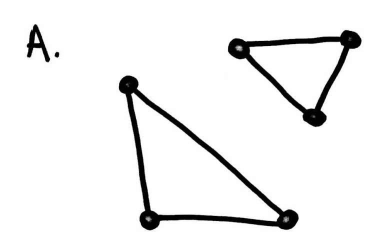
- Tours and TSP
- Visiting Graduate Schools: IP (Integer-Programming) Formulation of TSP

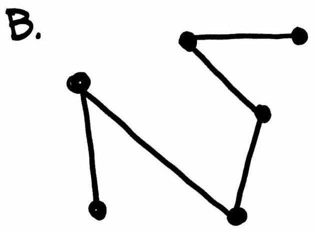
2 Tours and TSP

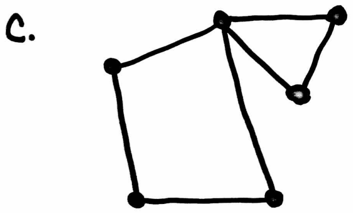
A **tour** is a route that visits every location exactly once (and closes the "loop" by returning back where it started).

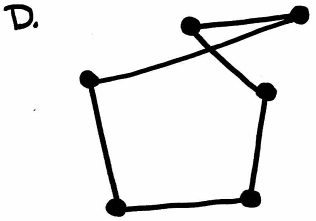
In graph terminology, a **tour** is a single *cycle* that touches every node of the graph.

Problem 1. For each graph below, does the set of edges represent a tour through the 6 nodes? If not, explain why not.









Given a graph $G = (V, E)$ with edge weights (representing costs or distances), the **Traveling Salesman Problem (TSP)** seeks a *minimum cost tour of G*.

Problem 2. TSP has a long and interesting history. Look up TSP here `http://www.math.uwaterloo.ca/tsp`. Write down a cool fact about TSP here.

3 Visiting Graduate Schools: IP Formulation of TSP

Problem 3. A college student is interested in visiting as many graduate schools as possible. She reasons that a single visit to each school is appropriate, and she wants to return to her own campus only after visiting all the schools. It is conceivable that she visits the schools in any order, but she would like to minimize the amount of driving she has to do. If the distance between schools i and j is $d_{i,j}$ $(i < j)$, where the matrix D of distances is given below, in which order should she visit the schools? Note that school 1 is her current school.

$$D = \left[\begin{array}{c|ccccc}
 & 2 & 3 & 4 & 5 & 6 \\
\hline
1 & 16 & 23 & 14 & 8 & 15 \\
2 & - & 12 & 19 & 9 & 13 \\
3 & - & - & 7 & 25 & 16 \\
4 & - & - & - & 18 & 15 \\
5 & - & - & - & - & 20
\end{array}\right]$$

(a) Draw the graph $G = (V, E)$ of the network below. Include node labels and edge costs. Highlight a collection of edges that form a tour of the graph (doesn't have to be the minimum distance tour).

(b) Write a concrete model to minimize the cost of the edges used in a "tour". Include constraints that ensure that each node touches exactly □ edges. (Why?)

(c) Do we have all of the constraints that we need? Can you think of a collection of edges that satisfies the constraints we have so far, but that is not a tour? (Sketch it here.)

(d) Write a concrete constraint that prevents the graph that you sketched above from being selected by the solver.

(e) Using abstract notation write a set of constraints that prevents ANY graphs of this kind from being returned by the solver. There is such one constraint for every subset C of vertices of G such that □ (restrict the number of vertices in C). These are called **subtour elimination constraints**.

> For even a moderately sized problem, there are way too many **subtour elimination** constraints to add them ALL to the model. In practice we could add these iteratively. Unlike the minimum spanning tree problem, we aren't guaranteed to have "good" success with this problem.