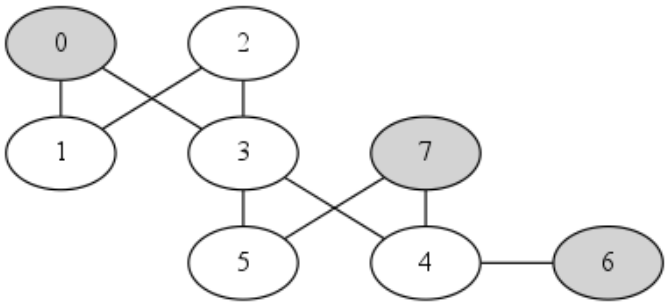
CPSC 320: NP-Completeness, or The Futility of Laying Pipe*

1 Steiner. . . Something-or-Others

UBC recently replaced its aging steam heating system with a new hot water system. A set of locations needs water delivered and there's another set of intermediate points through which we can deliver water. Some of these points can be connected—at varying costs—by laying new pipe, others cannot. You'd like to figure out the cheapest way to connect every delivery location to water.

We can model the problem as a graph, by representing locations and intermediate points as nodes. Possible connections are represented as weighted edges, where the weight of an edge is the cost of laying pipe between its two endpoints. For simplicity, moving forward, we'll assume that all costs are 1.

1. **Build intuition through examples.** Here's an instance, where shaded nodes are in S. Indicate a solution to this problem, that is, a way to connect up all of the shaded nodes. Draw your own small examples. What are trivial instances?



2. Formalize the problem as an *optimization* problem, where the goal is to minimize or maximize something.

   Also, formalize the problem as a decision problem, where the answer is Yes or No. We'll call the decision problem the Steiner Tree (ST) problem. What is an instance of ST? A potential solution? A good solution?

3. Think about similar problems by considering what an optimal solution to ST looks like. Is it a path? A cycle? A tree? A graph? Can we reduce ST to a similar problem we've seen before?

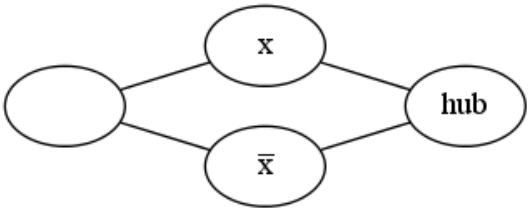
4. Prove that the ST (Steiner Tree) problem is in NP. Remember: it's in NP if it's "efficiently certifiable". The "certificate" is usually what we'd think of as a solution.

2 A Reduction from 3-SAT to ST

We've already shown that ST $\in$ NP. To show that it's NP-complete, we need to also show that it is NP-hard; that is, that it's at least as hard as every other problem in NP. We'll use 3-SAT to help us, since we already know that 3-SAT is at least as hard as every other problem in NP.

1. Which of these would show that ST is at least as hard as 3-SAT: reducing from ST to 3-SAT in polynomial time or reducing from 3-SAT to ST in polynomial time? (Hint: We **know** that any problem in NP can be reduced to 3-SAT in polynomial time. We want to **prove** that any problem in NP can be reduced to ST in polynomial time.)

2. Here is a sketch of a "variable gadget" to help with our reduction. How can we "shade in" (put in S) some of these vertices and choose an appropriate k (maximum number of edges in the solution) to enforce 3-SAT's choice that x or $\overline{x}$ is true but not both?



3. Draw a graph with four variable gadgets (w, x, y, and z), all sharing a single "hub" node. Make sure your layout still enforces choosing either true or false but not both for each of the four variables, yet allows all 16 possible combinations of their truth values.

4. Now, find a way to add one or more nodes and edges to your graph and choose a k in order to represent the clause $(\overline{w} \vee \overline{x} \vee y)$ and enforce that: at least one of $\overline{w}$, $\overline{x}$, and y is true and also (still) each variable is either true or false but not both. (z isn't in this clause, which is fine. In most 3-SAT problems, not all variables are in all clauses.)

5. Give a complete reduction from 3-SAT to ST such that the answer to the ST instance you produce is `YES` if and only if the answer to the original 3-SAT instance is `YES`.

6. Analyze the runtime of your reduction (to show that it takes polynomial time).

3 Reduction Correctness

Now, we'll prove that our reduction is correct. To do this, we need to show that instance I is a Yes-instance of 3-SAT if and only if instance $I' = (G, S, k)$ is a Yes-instance of ST. So, our proof should proceed in two directions, one for the "if" and one for the "only if".

3.1 If I is a Yes-instance of 3-SAT then I' is a Yes-instance of ST

1. We usually want to go further than the assumption that I is a Yes-instance, and describe what is a good solution (i.e., working certificate) for I.

   Consider what a solution for 3-SAT looks like and use it to finish the statement: "Since I is a Yes-instance of 3-SAT, there must be..."

2. To prove that I' is a Yes-instance of ST, we similarly try to show the existence of a good solution (working certificate) to that instance, and conclude that therefore the answer to the instance is `YES`. What does a good solution for ST look like?

3. Given a truth assignment that satisfies I, how would you select the edges of the good solution of I'?

4. Finish the proof to show that the ST certificate actually works, i.e., is a solution to the instance.

3.2 If I' is a Yes-instance of ST then I is a Yes-instance of 3-SAT

1. First, an apparently unrelated side-track: **How many edges must there be in a tree with n nodes?** So... **if you use at most k edges to form a connected graph, how many nodes can you connect?**

2. Now complete the proof. *Hints:* Many pieces of this proof will be very similar to the previous one, but you'll also need to show that any solution (working certificate) for the ST instance has its edges where you intended them to be when you designed your reduction. It may help to think about the number of nodes you can possibly connect, given a maximum of k edges.