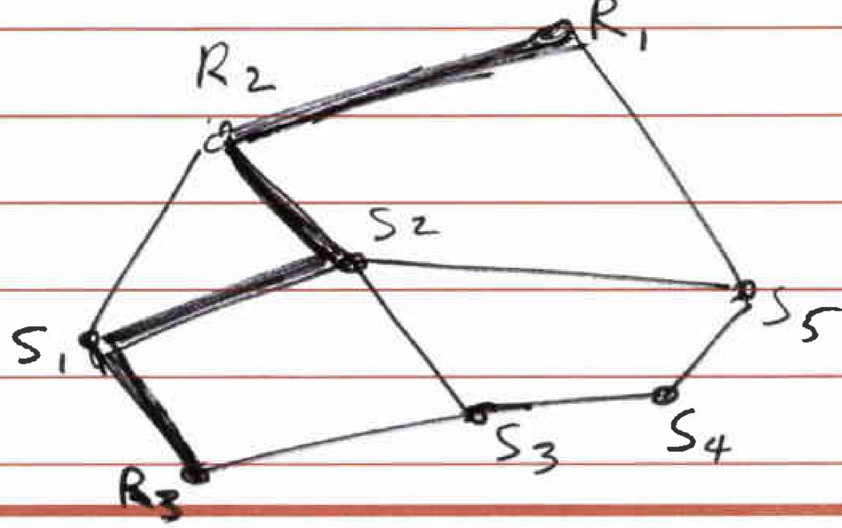
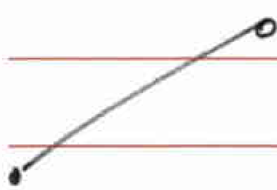
1- Show the problem is in NP

2- Choose a problem Y we already know is NP-Complete

3- $Y \leq_p$ problem at hand

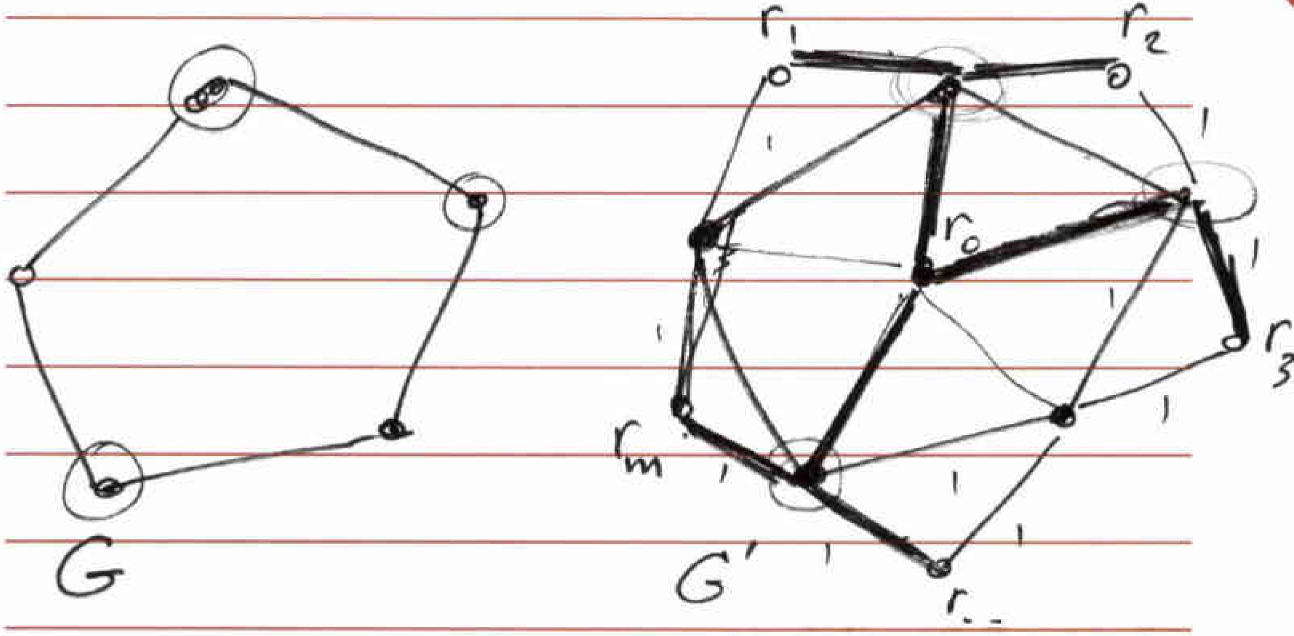


1- Certificate : tree T that ~~have~~ has cost at most C.



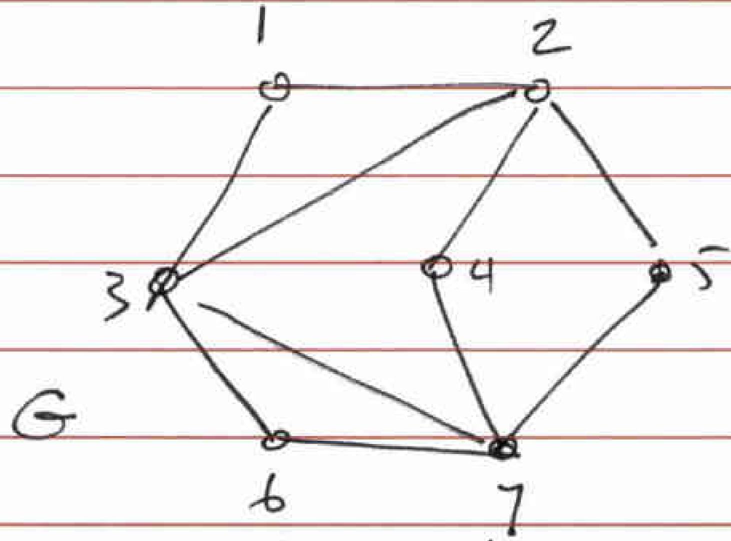
certifier : confirm this is a tree. (BFS) ✓

check cost ~~C~~ of T is ~~C~~ $\leq C$ ✓



A) If there is a vertex cover set of size at most k in G, I can find a steiner tree of cost at most m+k in G'

B) If there is a steiner tree of ~~size~~ cost at most m+k in G', I can find a vertex cover set of size at most k.



Is there an indep set of size at least $\underline{k}$ in G?

$S_1 = \{(1,2),(1,3)\}$

$S_2 = \{ \quad - \quad - \quad \}$

$\vdots$

$S_7 = \{ \quad - \quad - \;. \}$ $\quad \Sigma^6$

3-SAT problem

$$\underbrace{(\quad)\wedge(\quad)\wedge(\quad)\wedge\ldots(\quad)}_{k\text{-clauses}}$$

$$\underbrace{(X \& Y \& Z)\ldots\ldots(!X \& !Y \& !Z)}_{8\text{ clauses}}$$

$\underline{\quad}X$

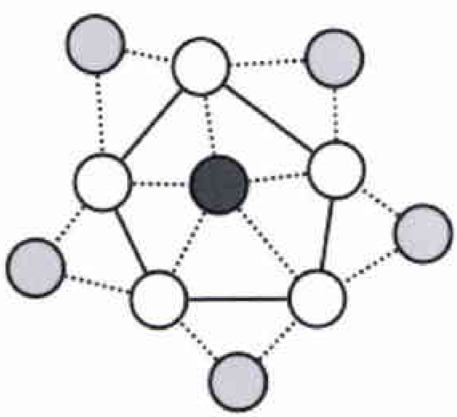
repeat this $k/8$ times

Note that discussion 12 is the same that it was last semester.

Solution Outlines:

1A: To prove it is in **NP**: the certificate is the tree itself; for the verifier, we can confirm it has every vertex from *R* is in the tree, and that the total cost is at most C.

To prove it is **NP**-Complete: Suppose we wanted to solve vertex cover and had a solution to Steiner Tree available to us. We can create a new graph G' that starts with a copy of G and adds, for each edge, a new vertex connected to the two endpoints. We also add a new vertex connected to all of the original vertices. Every edge gets a cost of one; set the required vertex set to be only the new ones. This new graph has a Steiner tree of cost C = m+k if and only if the original graph had a vertex cover of size k.
(m edges will be necessary to connect each new one on the "between" edges to existing vertices, and can connect to k original vertices; those k original vertices then each connect to the last added new vertex with one edge each to form a tree connecting all new vertices; those k incident original vertices constitute the vertex cover in the original graph.)
For example: suppose we wanted to find a vertex cover of size k=3 in the subgraph below consisting of the white vertices and the solid edges. We add the dark grey vertex in the middle; we will need k edges to connect it to the vertex cover of size k, should one exist. We can then connect each of the *m* light grey new vertices to the relevant vertex cover, thus making all grey nodes connected.



2A: To prove it is in **NP**: the certificate is a set C of sets; we can confirm in polynomial time that for each pair i,j in C, S_i and S_j have no common elements.

To prove it is **NP**-Complete: Suppose we wanted to solve independent set and had a solution to Set Packing available to us. We can create a set for each vertex in the graph; we give each edge a distinct label and give each set elements equal to the labels of its incident vertices. A set packing of size k corresponds exactly to an independent set of size k.

3A: To prove it's in NP: given a truth value assignment, we can count how many clauses are satisfied and compare it to 15k / 16.

To prove it's NP-Complete: Suppose I want to solve the original 3-SAT problem and I have a solution to 3-SAT(15/16) available to me. If I add 3 new variables, I can add 8 clauses so that only 7 of those clauses can be satisfied (all 8 distinct possible clauses). If I do this k/8 times, I will have 2k clauses, of which 15k/16 can be satisfied *if and only if* 100% of the original *k* could be satisfied.