Reductions between NP-complete languages

Recapping/formalizing stuff from last worksheet:

(Sipser Definition 7.28) A function $f : \Sigma^* \to \Sigma^*$ is *polynomial time computable* if some polynomial time Turing machine M exists that halts with just $f(w)$ on its tape, when started on any input w.
(Sipser Definition 7.29) Language A is *polynomial time mapping reducible*, or simply *polynomial time reducible*, to language B, written $A \leq_P B$, if a polynomial time computable function $f : \Sigma^* \to \Sigma^*$ exists, where for every w, $w \in A \leftrightarrow f(w) \in B$. The function f is called the polynomial time reduction of A to B.
(Sipser Definition 7.34) A language B is *NP-complete* if it satisfies two conditions: 1) B is in NP, and 2) every A in NP is polynomial time reducible to B. *(You might also have seen the term "NP-hard" before; this refers to any language meeting the second criterion, whether or not it meets the first.)*
(Sipser Theorem 7.36) If B is NP-complete and $B \leq_P C$ for C in NP, then C is NP-complete.

Boolean logic terminology:

(From Sipser p301) A *literal* is a Boolean variable or a negated Boolean variable, as in x or $\overline{x}$. *(You may be more familiar with the notation $\neg x$ for negation.)* A *clause* is several literals connected with $\vee$s, as in $(x_1 \vee \overline{x_2} \vee \overline{x_3} \vee x_4)$. A Boolean formula is in *conjunctive normal form*, called a *cnf-formula*, if it comprises several clauses connected with $\wedge$s, and is a *3cnf-formula* if all the clauses have three literals, as in $(x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (x_3 \vee \overline{x_5} \vee x_6) \wedge (x_3 \vee \overline{x_6} \vee x_4) \wedge (x_4 \vee x_5 \vee x_6)$.
A Boolean formula is *satisfiable* if there is some way to assign T/F values to the variables such that the whole thing evaluates to T.

Our first NP-complete problems:

Let $SAT = \{\langle\phi\rangle \mid \phi \text{ is a satisfiable Boolean formula}\}$. Let $3SAT = \{\langle\phi\rangle \mid \phi \text{ is a satisfiable 3cnf-formula}\}$.
(Sipser Theorem 7.37) SAT is NP-complete. (Sipser Corollary 7.42) $3SAT$ is NP-complete.
We will now show that a new language $VERTEX\text{–}COVER$ is NP-complete, by showing that $3SAT \leq_P VERTEX\text{–}COVER$.
Let $VERTEX\text{–}COVER = \{\langle G, k\rangle \mid G \text{ is an undirected graph that has a } k\text{-node vertex cover}\}$. (A vertex cover of G is a subset of the nodes where every edge of G touches one of those nodes.)
Given a 3cnf-formula ϕ, we construct a graph as follows:

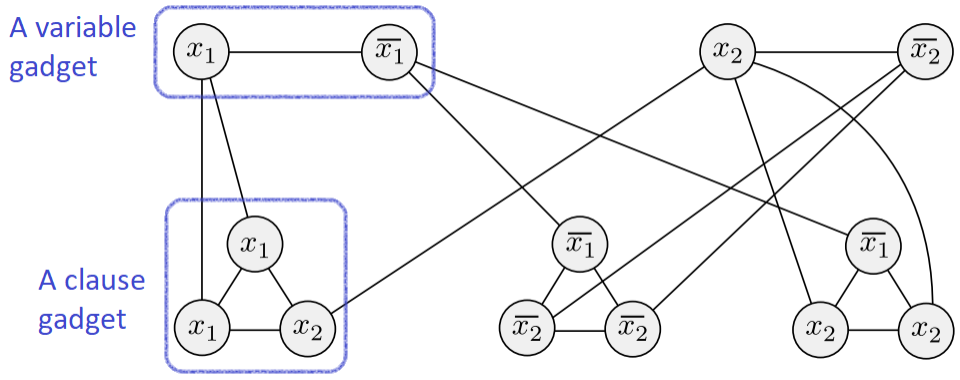

Figure 1: the graph constructed for $\phi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$

For each variable x there is a pair of connected nodes x and $\overline{x}$ (at the top of the example graph), and for each clause $\ell_1 \vee \ell_2 \vee \ell_3$ there is a triangle of connected nodes labeled with those literals (at the bottom of the example). Nodes in variable gadgets are adjacent to all nodes in clause gadgets that share their label. *(Sorry for the wall of text! Please ask if this construction (or anything else on the page) is unclear.)*

a What is the size of a minimal vertex cover in the complete graph K_{100}? (A "complete graph" K_n has n nodes and *every* possible edge.)

b What is the size of a minimal vertex cover in Figure 1? *Hint: first find a good lower bound on the answer using insights from part a.*

c In general for a graph generated in this manner from a *satisfiable* 3cnf-formula ϕ with v variables and c clauses, what is the size of a minimal vertex cover?

d Suppose that a graph generated in this manner from *some* 3cnf-formula ϕ with v variables and c clauses has a vertex cover of the size you determined in the previous problem. Prove that ϕ is satisfiable.

e Prove that $VERTEX\text{–}COVER$ is NP-complete.

f Let $CLIQUE = \{\langle G, k\rangle \mid G$ is an undirected graph containing a K_k subgraph$\}$. Prove that $CLIQUE$ is NP-complete via reduction from $3SAT$. *(K_n is defined in part a; a K_n subgraph is also called a "n-clique". Hint: For ϕ with c clauses, construct your graph such that there is a c-clique iff ϕ is satisfiable.)*

g (Extreme challenge problem)

Let $HAMPATH = \{\langle G, s, t\rangle \mid G$ is a directed graph with a Hamiltonian path from s to $t\}$. Prove that $HAMPATH$ is NP-complete. *(Recall that a path cannot reuse edges, and a path is Hamiltonian if it visits every vertex exactly once. Hint: For each variable, you'll want a gadget which can be traversed in one of two ways, corresponding to assigning T and F to that variable.)*