CSE 6321 - Solutions to Problem Set 4

1. (7.38) Show that if $P = NP$, a polynomial time algorithm exists that produces a satisfying assifnment when given a satisfiable Boolean formula. (Note: The algorithm you are asked to provide computes a function, but NP contains languages, not functions. The $P = NP$ assumption implies that SAT is in P, so testing satisfiability is solvable in polynomial time. But the assumption doesn't say how this test is done, and the test may not reveal satisfying assignments. You must show that you can find them anyway. Hint: Use the satisfiability tester repeatedly to find the assignment bit-by-bit.)

Solution:

Assuming $P = NP$, we have a polynomial time algorithm deciding SAT. To produce a satisfying assignment for a satisfiable formula ϕ, substitute $x_1 = 0$ and $x_1 = 1$ in ϕ and and test the satisfiability of the two resulting formulas ϕ_0 and ϕ_1. At least one of these must be satisfiable because ϕ is satisfiable. If ϕ_0 is satisfiable, pick $x_1 = 0$; otherwise ϕ_1 is satisfiable, pick $x_1 = 1$. That gives the value of x_1 in a satisfying assignment. Make that substitution permanent and determine a value for x_2 in a satisfying assignment in the same way. Continue until all variables been substituted.

2. Let

$$DOUBLE - SAT = \{\langle\phi\rangle \mid \phi \text{ is a boolean function that has at least two satisfying assignments}\}.$$

Show that $DOUBLE - SAT$ is NP-complete.

Solution:

On input ϕ, a nondeterministic polynomial time machine can guess two assignments and accept if both assignments satisfy ϕ. Thus $DOUBLE - SAT$ is in NP. We show $SAT \leq_P DOUBLE - SAT$. The following TM, F, computes the polynomial time reduction f.

$F =$ "On input $\langle\phi\rangle$, a Boolean formula with variables $x_1, x_2, \ldots, x_m$:

1. Let ϕ' be $\phi \vee (x \wedge \overline{x})$, where x is a new variable
2. Output $\langle\phi'\rangle$."

If $\phi \in SAT$, then ϕ' has at least two satisfying assignments that can be obtained from the original satisfying assignment of ϕ by changing the value of x. If $\phi' \in DOUBLE - SAT$, then ϕ is also satisfiable, because x does not appear in ϕ. Therefore $\phi \in SAT$ iff $f(\phi) \in DOUBLE - SAT$.

3. (7.26) Let ϕ be a 3cnf-formula. An **$\neq$-assignment** to the variables of ϕ is one where each clause contains two literals with unequal truth values. In other words, an $\neq$-assignment satisfies ϕ without assigning three true literals in any clause.

1. Show that the negation of any $\neq$-assignment to ϕ is also an $\neq$-assignment.
2. Let $\neq$SAT be the collection of 3cnf-formulas that have an $\neq$-assignment. Show that we obtain a polynomial time reduction from 3SAT to $\neq$SAT by replacing each clause c_i

$$(y_1 \vee y_2 \vee y_3)$$

with the two clauses

$$(y_1 \vee y_2 \vee z_i) \text{ and } (\bar{z}_i \vee y_3 \vee b)$$

where z_i is a new variable for each clause c_i and b is a single additional new variable.

3. Conclude that $\neq$SAT is NP-complete.

Solution:

1. In an $\neq$-assignment each clause has at least one literal assigned 1 and at least one literal assigned 0. The negation of an $\neq$-assignment preserves this property, so it too is an $\neq$-assignment.
2. To prove that the given reduction works, we need to show that if the formula ϕ is mapped to ϕ', then ϕ is satisfiable (in the ordinary sense) iff ϕ' has an $\neq$-assignment. First, if ϕ is satisfiable, we can obtain an $\neq$-assignment for ϕ' by extending the assignment to ϕ so that we assign z_i to 1 if both literals y_1 and y_2 in clause c_i of ϕ are assigned 0. Otherwise we assign z_i to 0. Finally, we assign b to 0. Second, if ϕ' has an $\neq$-assignment we can find a satisfying assignment to ϕ as follows. By part (a) we may assume the $\neq$-assignment assigns b to 0 (otherwise, negate the assignment). This assignment cannot assign all of y_1,y_2 and y_3 to 0, because doing so would force one of the clauses in ϕ' to have all 0s. Hence, restricting this assignment to the variables of ϕ gives a satisfying assignment.
3. Clearly, $\neq$SAT is in NP. Hence, it is NP-complete because $3SAT$ reduces to it.

4. (7.29) A **coloring** of a graph is an assignment of colors to its nodes so that no two adjacent nodes are assigned the same color. Let
$3COLOR = \{< G >$: the nodes of G can be colored with three colors such that no two nodes joined by an edge have the same color$\}$.
Show that $3COLOR$ is NP-complete.

Solution:
The proof is a simple reduction from $\neq$SAT. We are given a set of clauses $C_1, ..., C_m$ each with three literals, involving the variables $x_1, ..., x_n$, and we are asked whether there is a truth assignment on the variables such that no clause has all literals true, or all literals false.
We shall construct a graph G, and argue that it can be colored with colors $\{0, 1, 2\}$ if and only if all clauses can take diverse values. Triangles play an important role again: a triangle forces us to use up all three colors on its nodes. Thus, our graph has for each variable x_i a triangle $[a, x_i, \neg x_i]$; all these triangles share a node a(it is the node at the top colored "2" in Figure 1).
Each clause C_i is also represented by a triangle, $[C_{i1}, C_{i2}, C_{i3}]$(bottom of Figure 1). Finally, there is an edge connecting C_{ij} with the node that represents the jth literal of C_i. This completes the construction of the graph G (see Figure 1 for example).

We claim that G can be colored with colors $\{0, 1, 2\}$ if and only if the given instance of $\neq$SAT is satisfiable. In proof, suppose that the graph is indeed 3-colorable. We can assume, by changing color names if necessary, that node a takes the color 2, and so for each i one of the nodes x_i and $\neg x_i$ is colored 1 and the other 0. If x_i takes the color 1 we think that the variable is true. Otherwise it is false. How can the clause triangles be colored? If all literals in a clause are true, then the corresponding triangle cannot be colored, since color 1 cannot be used; so the overall graph is not 3-colorable. Similarly if all literal are false. This completes the proof of one direction.
For the other direction, suppose that a satisfying (in the $\neq$SAT sense) truth assignments exists. We color

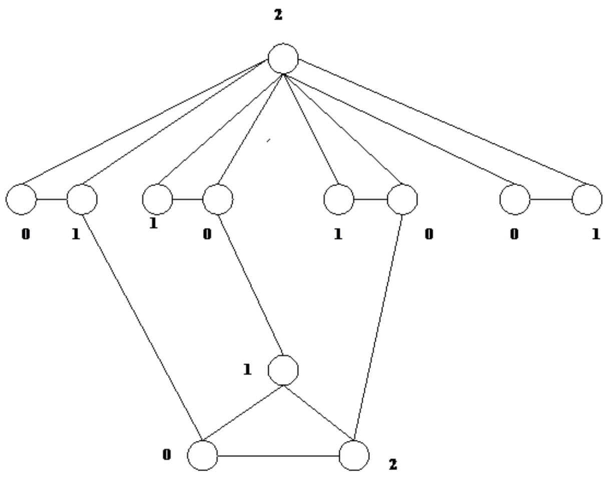

Figure 1: The reduction to 3-COLORING

node a by color 2, and the variable triangles in the way that reflects the truth assignment. And for any clause, we can color the clause triangle as follows: We pick two literals in it with opposite truth values (they exist, since the clause is satisfied) and color the vertices corresponding to them with the available color among $\{0, 1\}$ (0 if the literal is true, 1 if it is false); we then color the third node 2.