Homework 7

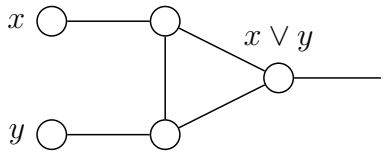
Problem 1. Let D-SAT be the problem of deciding if a CNF formula φ has at least two satisfying assignment. Prove that D-SAT is NP-complete.

Solution. D-SAT $\in$ NP is easy to verify: provide two satisfying witness.

Proving SAT $\leq_P$ D-SAT: for a SAT instance φ, construct $\phi = \varphi \wedge (y \vee \neg y)$, where y is a new variable not in φ. Easy to check that φ is satisfiable iff ϕ is satisfiable.

Problem 2. A coloring of a graph is an assignment of colors to vertices in a graph so that no adjacent vertices have the same color. Let 3-COLORING be the problem of deciding if a given graph G has a coloring using three colors.

(a) We will consider two graph gadgets. First, the triangle graph enforces that the vertices have different colors and you can use two of the colors T and F to encode true and false. Second, the OR gadget given in the following graph implements the logical OR operation. Prove that the OR gadget has the property that (1) if both vertices x and y have color F, then so is the vertex labeled by $x \vee y$; and (2) if one of them has color T, then it is possible to assign T to the vertex $x \vee y$ in the gadget.



(b) Use the above graph gadgets to prove that 3-COLORING is NP-complete.

Solution. (a) Easy to verify by enumeration.

(b) Easy to verify that 3-COLORING $\in$ NP. Now we show that 3-SAT $\leq_P$ 3-COLORING.

Given a 3-SAT instance φ, we first construct a triangle gadget, and name the three vertices as t, f, c.

For each variable x_i, we create two vertices x_i and $\neg x_i$, and create a triangle with x_i, $\neg x_i$ and c.

For each clause $C_i = l_{i1} \vee l_{i2} \vee l_{i3}$, we can create a clause gadget by combining two OR gadget together, and connect the input with corresponding l_i

vertices. We connect the output vertex $l_{i1} \vee l_{i2} \vee l_{i3}$ with c and f, finishing our construction.

We can see that if we have a satisfiable assignment for φ, we color t,f,c with color T F and C, and the variables as in the assignment by T and F. By the construction of OR gadget, we can show that each $C_i = l_{i1} \vee l_{i2} \vee l_{i3}$ output vertex can be colored to T.

On the other hand, if we have a 3-COLORING strategy for G_φ, we can name the three color of t, f, c as T, F, C wlog. By our construction, we can see that each x_i can only be assigned to T/F, and $x_i \neq \neg x_i$. and each output vertex of clause gadget is set to T. By the construction of OR gadget, we can see that there must be at least one input vertex assigned to T by the construction of or gadget. Thus we can extract an satisfying assignment from the coloring strategy.

Problem 3. Write a complete proof for the Claim 1 in the proof of Ladner's theorem discussed in class. That is, show that both $Z(n)$ and $n^{Z(n)}$ are computable in time polynomial in n. The definition of $Z(n)$ is given in the lecture notes.

Solution. We can recursively compute $Z(n)$ as defined in class. We can see that there are at most $O(2^{\log n}) = O(n)$ x such that $|x| \leq \log n$. For each x, we can check whether it is satisfiable by brute force with time cost $O(n)$. Now we run M_i for $i|x|^i = Z(n-1)(\log n)^{Z(n-1)} \leq \log\log n(\log n)^{\log\log n} = O(n)$ time, and check if the $M_i(x)$ decides whether $x \in$ SAT. Thus the total running time to compute $Z(n)$ from $Z(n-1)$ is $O(n^2)$, thus the running time to compute $Z(n)$ is $O(n^3)$. Given $Z(n)$, we can compute $n^{Z(n)}$ using exponentiation by squaring.