CS302 - Problem Set 10

Due: Monday, Dec 4

Here are some problem definitions:

- k-INDSET: Given an undirected, unweighted graph $G = (V, E)$, is there a set $V' \subseteq V$ such that $|V'| \geq k$, and for all $v, u \in V'$, there is no edge $\{u, v\} \in E$? (If yes, output the set.)
- k-CLIQUE: Given an undirected, unweighted graph $G = (V, E)$, is there a set $V' \subseteq V$ such that $|V'| \geq k$, and for all $v, u \in V'$, there is an edge $\{u, v\} \in E$? (If yes, output the set.)
- DOUBLE-SAT: Given a CNF formula with at most l clauses, where l is a polynomial, involving the variables $x_1, x_2, \ldots, x_n$ and their negations, are there at least two different satisfying solutions? For example, $(x_1 \vee \neg x_1 \vee \neg x_2) \wedge (x_2 \vee x_3) \wedge (\neg x_3)$ has two valid assignments, $x_1 = 1, x_2 = 1, x_3 = 0$ and $x_1 = 0, x_2 = 1, x_3 = 0$. (If yes, output two solutions.)

1. (*) (Extra practice with basic graph properties, please at least think about!)
   (a) [**0 points**] Prove that if you have an undirected connected graph (i.e. there is a path from every vertex to every other vertex) with n vertices and m edges, that $n = O(m)$. (Assume there is at most one edge between every pair of vertices and no self-loops.)
   (b) [**0 points**] Explain why a for graph with n vertices and m edges, that $m = O(n^2)$. (Assume there is only one edge between every pair of vertices and no self-loops.)
2. (**) *Detecting Negative Cycles*
   (a) [**6 points**] Describe how you would change the Bellman Ford algorithm to *detect* (do not need to output description) negative cycles, and explain why it is correct.
   (b) [**3 points**] What is the runtime of your algorithm to *detect* negative cycles? Explain.
3. (*(*)) [**6 points**] Explain why any problem in NP (using our definition from class) can be solved in exponential (i.e. $O(2^{n^k})$) time where n is the size of the input, and k is a positive constant.
4. (**) Prove DOUBLE-SAT is NP-Complete:
   (a) [**11 points**] Prove DOUBLE-SAT is in NP.
   (b) Prove DOUBLE-SAT is NP-Hard:
      i. [**6 points**] Describe a reduction from an NP-hard problem to DOUBLE-SAT, and show the reduction takes polynomial time.
      ii. [**11 points**] Prove that there is a solution to the NP-hard problem if and only if there is a solution to the DOUBLE-SAT problem.
5. (**(*)) Show that k-CLIQUE reduces to k-INDSET.

(a) **[6 points]** Describe the reduction from k-CLIQUE to k-INDSET, and explain why it takes polynomial time.

(b) **[11 points]** Prove that there is a solution to the k-CLIQUE problem if and only if there is a solution to the k-INDSET problem.

6. (***) Show that 3-SAT reduces to k-INDSET.

(a) **[6 points]** Describe the reduction from 3-SAT to k-INDSET, and explain why it takes polynomial time.

(b) **[11 points]** Prove that there is a solution to the 3-SAT problem if and only if there is a solution to the k-INDSET problem.