HW2
Theory of Computation

Brian DeFlaminio

1.) $(X \vee Y) \wedge (X \vee \bar{y}) \wedge (\bar{X} \vee Y) \wedge (\bar{X} \vee \bar{Y})$

No. The above formula is not satisfiable as both variables are present in their normal and complement forms as a part of the formula, meaning that with no combination of X and Y (TT, TF, FT, FF) can we satisfy the formula.

346 2.) CONNECTED = {⟨G⟩ | G is a connected undirected graph}

We can show that CONNECTED is in P by finding the "Big O" for it. Turing machine M described on pg 186 that decides CONNECTED has the following steps:

1) Select the first node of G (the input) and mark it. $O(n)$
2) Repeat the following stage until no new nodes are marked: $O(n+1)$
3) For each node in G, mark it if it is attached by an edge to a node that is already marked. $O(n)$
4) Scan all nodes of G to determine whether they are all marked. If they are, accept, if not, reject. $O(n)$

Combining the time complexity of the respective steps nets us an $O(n^4)$ complexity. P is the class of problems solved in polynomial time, and n^4 is polynomial, so CONNECTED is in the class P.

3) To show that ALL_{DFA} is in P we construct TM M such that M = {⟨M, w⟩ | Where M accepts iff no path exists from the start state to each non-accepting state}

1) Mark the starting node $O(n)$
2) if the start node isn't an accept state, reject. Otherwise mark every node that the start node has a path to. $O(n+1)$
3) Reject if any marked states are rejecting states. Otherwise repeat step 2 with each new marked state. $O(n)$
4) Accept if every state has been marked and is an accepting state. If an unaccepting state has been marked, reject. $O(n)$

Similar to Q2 the big O is n^4 which is polynomial and thus, in P.

4) $ISO = \{\langle G, H\rangle \mid G$ and H are isomorphic graphs $\}$
To show that $ISO \in NP$ we construct TM M where
$M =$ "On input $\langle G, H\rangle$ where G and H are undirected graphs:
1) Let N_1 represent the number of nodes in G and N_2 the number of nodes in H. If $N_1 \neq N_2$, reject.
2) Randomly (non-deterministically) select a permutation of all nodes, we will call this combo Z
3) For pairs of nodes in Z, check that both nodes from G and H respectively are both edges. If the corresponding nodes are not the same, reject. If they're both edges, repeat until all have been checked, then accept.
Due to the non-determinism in step 2, ISO is in NP.

5) Given an undirected graph G and a designated subset C of G's nodes, is it possible to convert G to a directed graph by assigning directions to each of it's edges so that every node in C has indegree 0 or outdegree 0, and every other node in G has indegree at least 1? To prove that this problem is NP-complete we need the following: 1) to show that the problem is in NP and 2) $A \leq_p B$ for all A in NP
Next we discuss the non-determinism present in the algorithm as it will be a lot easier to verify the requirements for a converted G when we are given an answer. Finding the solution ourselves takes much more work assuming there's a generalizable algorithm. This covers it's inclusion in NP and in order to prove the second NP-completeness requirement we reduce 3SAT to our problem.
For each pair of literals of variable x, namely X and $\bar{X}$ connect them by an edge. This makes it so when X is True (outgoing) then $\bar{X}$ is False (incoming) and vice versa. For each clause create a vertex that connects to the previously created corresponding literal vertices in C.
Each vertex in C will be either an all-incoming or all-outgoing one to make the satisfying assignment.

HW2
Theory of Computation

Brian DeFlaminio

6. Give a P-time algorithm that produces a parse tree for a String and a CFG that generates the String. Algorithm B is defined as:

B = "On input S and G where S is a String generated by G, a CFL
1) Break S into $S_1 S_2 S_3 \ldots S_n$ where n is the length of S and each S_n is a Character in the L(G)
2) For x=1 to n:
3) Generate S_x from G
4) Record rule used in G in parse tree format
5) If S_x cannot be generated by L(G) reject
6) Accept once the entire $S = S_1 S_2 \ldots S_n$ String has been replicated and it's parse tree should be complete.

7) A) For M and M' to be equivalent then all of their States, and thus their languages, must be the same. In MINIMIZE on instruction 6, (q, r) is only added to Graph G if $\delta(q,a), \delta(r,a)$ is an edge of G. Later in instruction 8 we form M' based off of subsets of what is included in G. Through this we've only populated M' with States considered indistinguishable from States in M.
B) If M' wasn't minimal then M wouldn't be minimal. We know that M is minimal though and that M' is considered indistinguishable from M. So it follows that M' is minimal.
C) In order to consider all nodes in G like the algorithm given in the question we take $O(n^2)$ time which is polynomial.

8) Show that all problems in NP are P-time reducible to D. To show this we reduce 3SAT, a known NP, to D. Given a 3CNF formula ϕ we construct polynomial P with all of the same variables, doing so will prove $NP \leq_P D$

3CNF $\downarrow$ Φ	D $\downarrow$ P
X	X
$\bar{X}$	$1-X$
$\wedge$	$*$
$\vee$	$(1-(1-x_1)(1-x_2)\cdots(1-x_n))$

Using the defined assignments we are able to turn a 3SAT problem to a Polynomial one, also we can now tell if a polynomial P has an integral root depending on if the ϕ form is satisfiable.

9) Assuming that SAT is in P, we attempt to brute-force a satisfying assignment by substituting X_1 for 0 and 1
7.35 ↳ in two respective formulas we will call ϕ_0 and ϕ_1. Choose one assignment that works and then repeat the process for X_2. Rinse and repeat for X_n.