CSCI338 HW5

Brock Ellefson
with collaboration with Elizabeth (Lizzie) Andrews

April 20, 2017

1 Show that $\mathbf{ALL}_{DFA}$ is in P

ALL_{DFA} = { <M> | M is a DFA and L(M) = $\sum^*$ }

We need to show that ALL_{DFA} is within P. Given any DFA M, we can determine in polynomial time if M can accept all strings from $\sum^*$. This can be accomplished by g either a breadth-first or a depth-first search, both of which are proven to be able to run in polynomial time. If a non-accepting state is reached in M, then L(M)$\neq \sum^*$ and <M> is not in ALL_{DFA}.

Therefore ALL_{DFA} is in P.

2 Show that ISO $\in$ NP

G and H can be verified in polynomial time. Let G' be a reordering of the nodes in G so that they are identical to H. Prove that G' is identical to H by the following:

-Let L be a list that will contain all visited nodes
-If the number of nodes in G' is not eqivilent to H, reject.
-For each g in G' and each h in H, if each g is equivilent to h, add g to L
-If the number of nodes in L is the same as the number of nodes in H and G', accept, otherwise reject.

Therefore a constructor was created that verifies G adn H in polynomial time.
Therefore ISO $\in$ NP

3 SPATH and LPATH

3.1 Show that SPATH $\in$ P

Construct a polynomial time algorithm that decides SPATH.

-Place a mark on node a to be the beginning node.
-Let i = 0. While i ¡ k, repeat the following step:
—For all edges (s, t) in G, if s is marked and t is unmarked, mark t with i + 1.

-If node b $\leq$ k, accept. Otherwise, reject.

Therefore, SPATH $\in$ P

3.2 Show that LPATH is NP-Complete

Prove LPATH $\in$ NP

-G = <a,b,k>
-Nondeterministically create path within G thats length is a minimum of k -If path starts a and ends with b and path only visits a node once, accept. Otherwise, reject
Therefore LPATH $\in$ NP. Onward, prove that every NP problem is reducible to LPATH. Acomplish this by reduceing the problem UHAMPATH to LPATH.

-Construct a formula f' which is a UHAMPATH formula given by f' = f where f = LPATH <G, a, b, k - 1>, k being the nodes in G.
-f' is in UHAMPATH iff f is in LPATH. If f' $\in$ UHAMPATH then there exists in G a simple path from a to b that touches all n nodes in G, by definition of a Hamiltonian Path. Therefore this path n - 1 length.
f $\in$ LPATH.

If f $\in$ LPATH, then G contains a path of length n - 1 from a to b, by our definition of LPATH. Since G has n nodes, this means that the path touches every node in G, and it only touches each node one time. Therefore, the path must be a Hamiltonian Path
f' $\in$ UHAMPATH.

Therefore LPATH is NP-Complete

4 Show that DOUBLE-SAT is NP-Complete

DOUBLE-SAT is a variant of 3SAT, it is in NP.
Show that 3SAT is reducible to DOUBLE-SAT.

-Construct a DOUBLE-SAT formula f' = f $\wedge$ (x $\vee$ $\neg$x), where f is a 3SAT formula.
-f is a satisfying assignment iff f' has two satisfying assignments.
-If f has a satisfying assignment u, then f' has two satisfying assignments, either ((u, x) = false) or ((u, x) = true).
-If f' has two satisfying assignments, then the clause (x $\vee$ $\neq$x) has two possible satisfying assignments, either (x = true) or (x = false).

Therefore DOUBLE-SAT is NP-Complete

5 DOMINATING-SET NP-Complete

DOMINATING-SET is in NP.
Reduce VERTEX-COVER to DOMINATING-SET.

-Construct a graph G' such that G' has a dominating set that is size k iff G has a vertex cover equivilent to k. -If G' has a dominating set S that is size k, replace any node n in S that has the edge (u, v) with either of its endpoints, u or v. This is continued until we get a set S' that only contains vertices contained in G.
-S' is a vertex cover for G, because for every edge e in the set of edges belonging to G, e is adjacent to a node in S'.
-If a vertex cover S in G that is size k, then S must also be a dominating set for G'.
Therefore for any node n in G', either n $\in$ S, or n has some adjacent node that is in S. Assuming that G is a connected graph and does not contain any isolated vertices, there exists some edge (n, w) in the set of all edges.

Therefore n is either in S or is connected to some other node in S. Therefore DOMINATING-SET is NP-Complete