CIS 511 Homework 5

Stephen Phillips, Dagaen Golomb

April 8, 2015

Problem 1

Problem 2

Part a

Part b

Problem 3

Problem 4

If $P = NP$, then there exists a polynomial time decider D for SAT runs in polynomial time. From this we can create a new machine M that finds the assignment of the variables. The idea is that we assign true or false to each of the variables in order, and use D to determine if it is a valid assignment. If there are multiple assignments we just pick the one that works in order of the variables.

function $M(\phi)$
    Create ψ, a modifiable copy of ϕ
    Create bit vector b for the final assignment
    for all $i \in \{1, \ldots, n\}$ **do**
        Set $\psi' \leftarrow \psi(x_i = 1)$
        if $D(\psi')$ accepts **then**
            $\psi \leftarrow \psi'$
            $b_i \leftarrow 1$
        else
            $\psi \leftarrow \psi(x_i = 0)$
            $b_i \leftarrow 0$
        end if
    end for
    Output b
end function

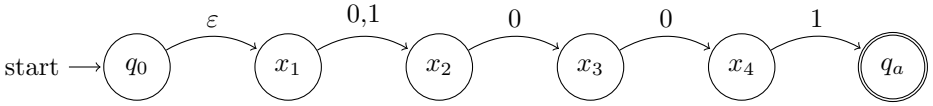
In words, we greedily go through the variables assigning them. We initially assign each variable x_i to 1, and use D to check if that assignment is feasbile. If it is, then there must be a valid assignment where x_i is 1, and if not x_i must be 0. So we assign x_i appropriately. Then we update the formula to reflect this choice, and continue appropriately.

Now we show this runs in polynomial time. Let n be the number of variables, and m be the number of clauses in the original formula. On each iteration of the outer loop, we update the formula, which takes $O(m)$ time, and we run the decider D, which runs polynomial time for some polynomial $p(n + m)$. There are n iterations of the loop, so in total this takes $O(nmp(n + m))$ time, which is still a polynomial.

Problem 5

Problem 6

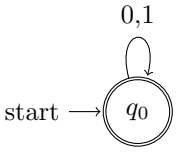
We want to, given a CNF formula ϕ, construct an NFA that accepts all nonaccepting configurations of ϕ. The idea is we create a sub-NFA for each clause that accepts if and only if the clause is false. To illustrate, we use a simple formula over 4 variables $\phi(x_1, x_2, x_3, x_4) = (x_2 + x_3 + \bar{x_4})$



This takes in as input the bit string representing the assignment of all the variables. Each of the successive nodes represents the variables, and they transition to the next node on values that would allow the clause to reject. Notice that since x_1 does not appear in the clause, it can take whatever value it wants. So this accepts the non-satisfying strings $0, 1001$.

We can generalize this to more clauses, by making the initial state have ε-transitions to a chain for each clause, constructed in the exact same way as the above example. That would give, with m variables and c clauses, cm nodes. The transition function would take then $O(cm)$ space to create. This means it can be created in $O(cm)$, or polynomial, time. This new NFA would accept on any rejecting input since it only takes one failing clause to make the formula reject.

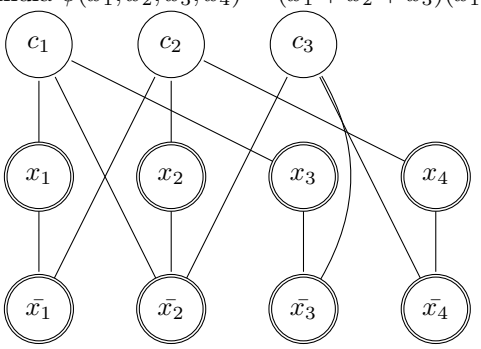
If this formula has no accepting input, this NFA would reduce to one that accepts $\{0, 1\}^*$, or:



Therefore if we could reduce this NFA in polynomial, we would be able to solve SAT in polynomial time. In other words we just reduced SAT to minimizing an NFA. Therefore if $P \neq NP$ then there is no polynomial time solution for this.

Problem 7

We want to show that the problem of designating directions to edges of an undirected graph G so that a given subset C will have all nodes in it have either indegree 0 or outdegree 0, and all other nodes have indegree at least one is NP-complete. We reduce SAT to this. And example of the mapping is show for the formula $\phi(x_1, x_2, x_3, x_4) = (x_1 + \bar{x_2} + x_3)(\bar{x_1} + x_2 + x_4)(\bar{x_2} + \bar{x_3} + \bar{x_4})$



The highlighted nodes are the ones in C.

We want to create such a graph for any SAT formula ϕ. So in words the procedure is for every variable x_i, create nodes x_i and $\bar{x_i}$, and for every clause c_j create a node c_j. Connect the node x_i (or $\bar{x_i}$) to node c_j if the variable x_i (or $\bar{x_i}$) is in the clause c_j. Also create an edge between nodes x_i and $\bar{x_i}$. Place all the nodes x_i and $\bar{x_i}$ into C.

This graph has a solution the the direction problem if and only if the formula ϕ has a solution. If the formula ϕ has a solution, select the true literals $x_i/\bar{x_i}$ as the 'all outdegree' or 0 indegree nodes in C, and the complement pair as the 'all indegree' or 0 outdegree ones. Since only the pairs of literal nodes $x_i/\bar{x_i}$ are

connected in C, this does not lead to an inconsistency. If this is the satisfying assignment then we should have at least one edge into each clause node c_j. Similarly, if we have such an assignment of edge directions, we can reconstruct the satisfying assignment, since only one in each of the pairs of literal nodes $x_i/\bar{x}_i$ has 'all outdegree' we can set those literals to true, and we know by the direction assignment that every clause will have one corresponding true literal inside it. Therefore, this graph has a solution to the problem if and only if the the formula has a satisfying assignment.