ECE653 Assignment 2: Boolean logic, normal forms, proof systems, solvers, grammars (Lectures 6-16)

Question 1: Short answer questions (30 points)

Please provide short and correct answers to the following questions. You will get points only if you provide appropriate justification. Otherwise you will get 0 points. Each sub-question is worth 10 points.

1. Consider a proof system P for Boolean logic such that given any formula F (valid or otherwise) in conjunctive normal form, P produces a proof of F. That is, P asserts that F is a theorem and hence valid. Is P a sound proof system? Is P a complete proof system? (State the definitions of soundness and completeness of proof systems in order to justify your answers.)

2. Consider the semantic argument method discussed in class for Boolean logic. Alice implemented this method correctly (sound and complete) on her computer using recursion. Her friend Bob asked for the source code and saw some opportunities to optimize it. In the process, he accidentally commented out the recursive case for negation. Is Bob's implementation sound and complete? If yes, argue why Bob's system is both sound and complete. If not, provide an argument as to why Bob's system is unsound or incomplete.

   You can assume that Alice's implementation of the semantic argument method has an input-processing step that performs the following steps in the order given below:

   - Checks if inputs are well-formed. If they are not well-formed, rejects such inputs.
   - For any well-formed input formula F, the input-processing step applies negation to F and converts $\neg F$ to NNF. Let us call the resultant formula G. The semantic argument method is then called on G.

3. Are context-free grammars closed under infinite union (we define this as L = $L_1 \cup L_2 \cup L_3 ...$)? If yes, provide a proof. Else, provide a counter-example.

Question 2: Semantics of Boolean Formulas (20 points)

Write a recursive program Eval that takes as input a Boolean formula F as a graph (defined in class) and a complete assignment A to all variables in F, and correctly evaluates the truth value of F on A and returns it. Please only provide the pseudo-code for Eval in an imperative language like C/Java, and describe its operation in detail. Assume that input formulas are syntactically correct (aka, *well-formed*).

Question 3: Negation Normal Form (20 points)

Write a recursive program NNF() that takes as input a Boolean formula F as a graph (defined in class), and outputs an equivalent formula G such that $F \iff G$ and G is in NNF. Explain why your algorithm is equivalence-preserving.

Question 4: DPLL SAT Solvers (30 points)

Provide the pseudo-code the DPLL SAT solver. Informally argue as to why the DPLL SAT solver is a decision procedure for the Boolean satisfiability problem. (In order to show that a program is a decision procedure for the Boolean satisfiable problem, you have to show two properties: First, you have to show that given a satisfiable input, the procedure will eventually find a satisfying assignment for it and terminate. Second, you have to show that given an unsatisfiable input, the procedure will eventually prove that it is unsatisfiable and terminate.)

Question 5: Tseitin Transformation and Conjunctive Normal Form (+2 extra credit)

Using structural induction prove that the input and output formulas of the Tseitin transformation algorithm are equisatisfiable to each other. Give an argument as to why the Tseitin transformation is a polynomial time algorithm. (Either provide an original proof or if you use a resource, then explain the proof in your own words and cite the resource.)