Homework 1

COSC 560
Temporal Verification and Real-Time Systems

Due at the start of class on: 9/24/2019

Note: You can submit your homework through e-mail to Stanley.Bak@georgetown.edu. For written portions, you can also optionally hand them to me in class. Homework is due at the start of class. Late homework incurs a 50% penalty.

Problem 1 [Propositional Logic]

1. Prove that the following semantic entailments are true by using truth tables. **[10 points]**
   - $(p \rightarrow q) \rightarrow r, s \rightarrow \neg p, t, \neg s, t \rightarrow q \models r$
   - $\emptyset \models q \rightarrow (p \rightarrow (p \rightarrow (q \rightarrow p)))$
2. Prove the following inferences in propositional logic using natural deduction proof rules. **[15 points]**
   - $(p \rightarrow q) \rightarrow r, s \rightarrow \neg p, t, \neg s, t \rightarrow q \vdash r$
   - $\vdash q \rightarrow (p \rightarrow (p \rightarrow (q \rightarrow p)))$
   - $\vdash p \vee \neg p$

Problem 2 [Frege Proof System] Frege's Propositional Calculus [1]uses six axioms and one inference rule (Modus Ponens). Prove that the six axioms are valid using proof rules of natural deduction. **[15 points]**

1. $\vdash (\phi \rightarrow (\psi \rightarrow \phi))$.
2. $\vdash ((\phi \rightarrow (\psi \rightarrow \eta)) \rightarrow ((\phi \rightarrow \psi) \rightarrow (\phi \rightarrow \eta)))$.
3. $\vdash ((\phi \rightarrow (\psi \rightarrow \eta)) \rightarrow ((\psi \rightarrow (\phi \rightarrow \eta))))$.
4. $\vdash (\phi \rightarrow \psi) \rightarrow (\neg\psi \rightarrow \neg\phi)$.
5. $\vdash \neg\neg\phi \rightarrow \phi$.
6. $\vdash \phi \rightarrow \neg\neg\phi$.

Problem 3 (Bonus) [Complexity of Some Problems in Propositional Logic] Two of the commonly used normal forms for formulas in propositional logic is Conjunctive Normal Form (CNF) and Disjuctive Normal Form (DNF). A formula ϕ is said to be in k-CNF form if it is a conjuction of subformulas, i.e., $\phi = \psi_1 \wedge \psi_2 \wedge \psi_l$ where, each ψ_i is a disjuction of k

[1] https://en.wikipedia.org/wiki/Frege%27s_propositional_calculus

literals, i.e., $\psi_i = p_1 \vee p_2 \vee \ldots \vee p_k$. A formula ξ is said to be in k-DNF if it is a disjunction of subformulas, i.e., $\xi = \eta_1 \vee \eta_2 \vee \ldots \vee \eta_l$ where, each η_i is a conjunction of k literals, i.e., $\eta_i = p_1 \wedge p_2 \wedge \ldots \wedge p_k$. Here, p_i can be a propositional variable or its negation. Now prove the following complexity theoretical problems over propositional logic.

1. Given a k-DNF formula, prove that its satisfiability can be decided in polynomial time. **[5 points]**

2. Given a k-CNF formula, prove that to decide whether the given formula is a tautology can be decided in polynomial time. **[5 points]**

3. Given a 2-CNF formula, prove that its satisfiability can be decided in polynomial time. **[5 points]**

Problem 4 [Predicate Logic Proof]

Using proof rules of natural deduction in first order logic (predicate logic), we proved in class that $\neg\forall x P(x) \vdash \exists x \neg P(x)$. Prove the other direction: **[10 points]**

$$\exists x \neg P(x) \vdash \neg\forall x P(x)$$

Problem 5 [Solving Sudoku Using SAT Solvers] Sudoku is a popular number-placement puzzle that originated in France in the end of the 19th century. Modern Sudoku was likely invented by Howard Garns from Connersville, Indiana and was first published in 1979 under the name Number Place. The objective of the puzzle is to place numbers 19 on a 9×9 grid, such that each number occurs only once in every row, every column, and every of the nine 3×3 sub-grids that compose the main grid. Sudoku puzzles are grids that have been partially occupied with numbers. The task is then to occupy the remaining fields in such a way that the constraints on rows, columns, and sub-grids are satisfied. A sample Sudoku problem and its solution are given in Figure 1. For more information about Sudoku refer to its Wikipedia page at `http://en.wikipedia.org/wiki/Sudoku`.

This problem has two parts. In the first part, you will write the boolean constraints in mathematical notation for solving a Sudoku puzzle. In the second part, you will write code and invoke a SAT solver to solve the Sudoku instance.

Part 1:

1. Write the boolean formula for the constraints that the number 5 can occur at most once in the first row. **[3 points]**

2. Write the boolean formula for the constraints that the number 6 can occur at most once in the first column. **[3 points]**

3. Write the boolean formula for the constraints that the number 9 can occur at most once in the top left 3×3 sub-grid. **[3 points]**

Part 2: Encode the above constraints in a SAT solver and solve the Sudoku instance given in Figure 1, **using only boolean variables**. One of the widely used techniques is to encode these constraints using the API of the SAT solver. Two popular options are as follows; 1) Using Z3 (downloadable at `http://z3.codeplex.com/`) and use Python API (z3py) and 2) Use MiniSAT (downloadable at `http://minisat.se/`) and use C++ API. Alternatively, you can write code to generate constraints in SMT-2 format (described at `http://smtlib.cs.uiowa.edu`) and give the SMT-2 file as an input to the binary of SAT solver. **[15 points]**

	1		4		2		5	
5								6
			3		1			
7		5				4		8
2		8				5		9
			9		6			
6								2
	7		1		3		4	

Figure 1: Sudoku puzzle for Problem 5.

Is your solution unique? Prove it with a SAT solver or provide a second solution. **[5 points]**

Bonus: If you delete the 1 in the top left box of the Sudoku problem in Figure 1, how many solutions are there? **[5 points]**