CS 511
Formal Methods for High-Assurance Software Engineering
Homework Assignment 02 (adjusted)

Out: 11 September 2020
Due: Friday, 18 September 2020, by 11:59 pm

A few administrative issues Homework Assignment 01:

- If you did not open a Gradescope account for Homework Assignment 01, you need to open it now, after which you need to add yourself to the CS511 roster for this semester. The entry code fo CS511 Fall 2020 is: **9N6E68**.
- There are six problems in this assignment. The first 4 problems have to be solved by hand. The last 2 problems are implementation problems.
- ~~If you submit any or all of the first 4 problems, you should typeset them with Latex and submit them in a single `.pdf` file.~~
- ~~If you do one or both of the last 2 problems, you should submit each as a separate file, one with extension `.smt2` and one with extension `.py`.~~
- **Ignore the preceding two bullet points. You should submit a single .pdf file to Gradescope, where you include a link to a `.smt2` file (for Problem 5) and a link to a `.py` file (for Problem 6). The two latter files should be stored in the repository of your Github account.**
- For full credit in the homework, you need to complete 4 out of 6 problems in this assignment. Each is worth 4 points. Of course, you may want to try all 6 problems. You will get credit for all extra exercises you do (correctly!).

Problem 1 [LCS, page 88]: Exercise 1.5.7. Do all three parts: (a), (b), and (c).

Problem 2 There are two parts in this problem:

1. Consider the exercise on page 13 of Lecture Slides 07 entitled, "Do You Believe de Morgan's Laws?". Do part 1 only. Start by parenthesizing the wff as $\neg(p \wedge (q \wedge r)) \rightarrow (\neg p \vee (\neg q \vee \neg r))$.
2. Consider the exercise on page 18 of Lecture Slides 08 entitled, "Analytic Tableaux". Do part 1 only. Again here, start by parenthesizing the wff as $\neg(p \wedge (q \wedge r)) \rightarrow (\neg p \vee (\neg q \vee \neg r))$.

Problem 3 Go to the set of Lecture Notes with file name `2020-09-07.fCtC.pdf`, posted on Piazza under the "Resources" webpage. Do part 1 of Exercise 13 on page 12.

Hint: Before you try this problem, make sure you understand how Example 11 on page 10 is handled.

Problem 4 This continues the preceding problem. Again, go to the set of Lecture Notes with file name `2020-09-07.fCtC.pdf`. Do part 2 of Exercise 13 on page 12.

Problem 5 The parity function on $n \geqslant 2$ truth-value (or Boolean) arguments returns *true* if an even number of its arguments are *true*, and it returns *false* if an odd number of its arguments are *true*. Your task is to write a Z3 script (with file extension `.smt2`) three different but equivalent propositional wff's for the parity function over 4 propositional variables $\{p1, p2, p3, p4\}$. The script should start with the variable declarations:

```
(declare-const p1 Bool)
(declare-const p2 Bool)
(declare-const p3 Bool)
(declare-const p4 Bool)
```

followed by propositional wff φ for the parity function in CNF (Conjunctive Normal Form), propositional wff ψ for the parity function in DNF (Disjunctive Normal Form), and propositional wff θ for the parity that uses only the biconditional "$\leftrightarrow$" as logical connective.

Your script should also check if your formulas are indeed equivalent.

Hint. Two propositional wff's φ_1 and φ_2 are equivalent iff:

- the wff $(\varphi_1 \leftrightarrow \varphi_2)$ is a tautology, or equivalently iff:
- the wff $(\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_2 \rightarrow \varphi_1)$ is a tautology.

A propositional wff ψ is a tautology iff $\neg\psi$ is unsatisfiable.

Problem 6 Repeat Problem 5 above, now implemented as a Z3Py script (with file extension `.py`).