CSE 3302 Lab Assignment 5

Due December 5, 2012

Goals:

1. Experience with a functional language.
2. Experience with symbol manipulation.

Requirements:

1. Design, code, and test a Scheme or SML program that uses a top-down semantic/analytic tableau method to test whether a propositional formula is a tautology (e.g. valid). This method negates the conjecture and shows that all truth assignments are contradictory.

   a. Your input should be in prefix/functional/list form and use single letters for propositions and a clear notation for "not", "and", and "or". "not" will be unary, but "and" and "or" may connect an arbitary number of subexpressions. Parentheses or other bracketing will be useful.

   b. The recursive processing is described later. Besides indicating whether or not an expression is a tautology, your program should also provide:

      1. Summary counts for the number of leaf and non-terminal nodes occurring at each level in the search tree.

      2. When a leaf situation is reached during the search.

         a. If contradictory, just include it in a level count.
         b. If not contradictory, stop and print the (partial) truth assignment.

2. Email your program and test cases for the provided expressions to `mehra.nourozborazjany@mavs.uta.edu` by 12:45 p.m. on December 5. Be sure to indicate the platform you used.

Getting Started:

1. The negated input expression should be initially preprocessed (recursively) using DeMorgan's laws. This will move the nots deep into the subexpressions.

2. Each call in the recursion will take the following four arguments and returns an indication of terminate/continue:

   a. Optional conjunctive subexpression.
   b. Set of disjunctive subexpressions (viewed conjunctively).
   c. Set of true atoms.
   d. Set of false atoms.

3. The root call provides the preprocessed input expression as either 2.a. or 2.b. while the last two sets are empty.

4. A call proceeds in the following order:

   a. Conjunctive expressions are decomposed until none remain. If a non-negated atom is encountered, it is inserted in the third set. If a negated atom is encountered, it is inserted in the fourth set. Before inserting, however, you should check to see if it is in the other atom set (indicating contradiction).
   b. The first disjunctive subexpression should be removed from 2.b. and decomposed into subexpressions. Each subexpression will be processed recursively as set 2.a. (or in set 2.b.), but with the "tail" of 2.b. passed along with sets 2.c. and 2.d.