CS 271 (Spring 2013) — Assignment 3
Due: 02/14/2013

This homework is to be done in groups of 3 students, assigned randomly by the program posted on the course web site. Each group can only turn in one solution, with all students' names listed on it. The group composition is posted at `http://www-bcf.usc.edu/~dkempe/CS271/team-assignments-3.html`.

(1) Read Sections 1.1, 1.2, 1.3, and part of Section 1.6 (up to Example 11).

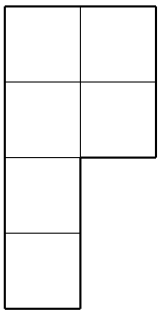
(2) Attend a sherpa session this week. Preferably, that would be the one you're assigned to, but it doesn't have to be. This week, sherpas will be taking attendance. Your group will receive credit based on how many of you were at your sherpa sessions.

(3) Find examples in the real world of the following.

(a) A statement that uses words for logic operators, but not in the sense that computer scientists would. In other words, find a statement that a careful computer scientist would interpret differently from its intent because a logic operator (such as "and", "or", or "if ... then") is used in a different sense from the mathematical one.

(b) A statement that is made ambiguous by ambiguity in how operators (such as "not", "and", "or") bind. Recall the "No food or drink in the lab" example from class.

(4) Latin Squares are a puzzle resembling Sudoku. See `http://latinsquares.com/Site/Welcome_files/LSQexample.pdf` for the rules. Find a set of propositional variables that captures proposed Latin Squares solutions. Then write down a formula as a big $\bigwedge$ (AND) of $\bigvee$ (OR) that is true exactly when the proposed solution is valid. Since Latin Squares can have different shapes (see the example), I recommend modeling the six shapes that are on the 6×6 square as sets $S_1, S_2, \ldots, S_6$, where each set S_i contains 6 pairs (x, y) of the coordinates of the squares in the set. An example would be the set $S_1 = \{(1,1),(1,2),(1,3),(1,4),(2,1),(2,2)\}$ which would capture the shape on the right.



It is strongly recommended that you read pages 32–33 of the textbook to see a model of how to do this.

(5) Solve the following exercises from the textbook

(a) Section 1.1, Exercises 10, 12, 22 (d)–(g), 36,

(b) Section 1.2, Exercises 6, 8, 12, 32

(c) Section 1.3, Exercises 10, 12, 22, 52, 58

(6) [0 points]
Chocolate Problem (4 chocolate bars): In Java, C, or C++, write a resolution proof system, and use it to solve Sudokus. By a "resolution proof system", we mean a program that uses only the resolution rule to derive formulas from known ones. (See Page 74 in the textbook.)

Here are some hints about this:

(a) Resolution proofs work only on formulas of the form $p \vee q \vee r$ or $\neg p \vee s \vee \neg t \vee \neg x \vee \neg y$, i.e., disjunctions (ORs) of variables or negated variables. So your system only needs to deal with such formulas as inputs (axioms). Of course, you may have many such formulas (which corresponds to have a big AND of all your ORs), including ones that only have one variable. (An empty disjunction is "false", which means you have reached a contradiction.)

(b) You may want to think carefully about what data structures to use for representing your formulas. Think about what operations you need to support.

(c) If you have derived the formulas p and $p \vee q$, you don't need the latter and can discard it. That may help you with memory.

(d) You may also want to think carefully about which pair of formulas to apply resolution to if there are multiple candidate pairs. If you want your system to work reasonably fast, you probably want to optimize a bit here.

(e) For the Sudoku part, I recommend writing a front-end that produces formulas automatically after reading the Sudoku from a file. (You almost certainly want to build on Pages 32–33 from the textbook for this.) You can then feed these files to your general resolution proof system.

Notice that this is quite a substantial programming project; I would be surprised if even a good programmer took less than 15–20 hours to solve it. Also, resolution is really not the best way to solve Sukokus (smart exhaustive search works much better); my own Java solution can only solve easy and medium puzzles within reasonable time.

Because this is rather substantial, this chocolate problem can be turned in until 02/21/2013.