MATH CSE20 Project 1

DUE October 27, on Ted.

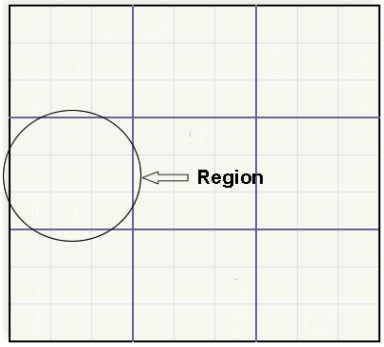
Topic. Sudoku, propositional logic, SAT, and verification

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

FIGURE 1. Sample sudoku puzzle

The *sudoku* problem can be described mathematically. The input is a 9×9 matrix M whose entries are either "blank" or an integer between 1 and 9. A solution fills in the blank spaces with integers between 1 and 9. The following constraints must be met: Each integer from 1 to 9 appears exactly once in

- each row,
- each column,
- and each of the nine 9×9 sub-matrices (regions) pictured below.



The problem is to find any solution meeting the constraints, or return "no solution possible" if there is no such solution.

Sudoku puzzles have become widespread, with many newspapers, cellphone apps, and computer programs providing puzzles for entertainment. Each of those puzzles are created so that there *is* a solution (usually, a unique one). We'll be using propositional logic and openly available SAT solvers to test the difficulty of sudoku puzzles for automated solvers, and to experiment with how different representations of the same puzzle affect the ability of solvers to find solutions quickly.

To use a SAT solver to solve a puzzle for us, we need to *represent* the puzzle as a Boolean formula in CNF format. To do this, we need to have boolean variables that represent parts of

the solution. For example, you might use $x_{3,7,5}$ to represent whether there is a 5 in the solution at the 3'rd row of the 7'th column. Then we need to define clauses that represent the constraints of the puzzle. A clause is a Boolean "Or" of variables and negated variables. So one clause might be $\neg x_{3,7,5} \vee \neg x_{3,7,6}$, expressing that we cannot have both a 5 and a 6 in the 3rd row of the 7'th column. A conjunction or Boolean AND of clauses represents the entire puzzle. There is more than one way to represent puzzles as CNF's. You can pick one, or experiment with a number.

Once you have a way to represent puzzles as CNF's, see how long the MiniSAT solver takes for different puzzles. The MiniSAT solver is available on Ubuntu. You need to learn how to create input files that describe the CNF. David Wheeler has a website on how to use MiniSAT: http://www.dwheeler.com/essays/minisat-user-guide.html

You can use the puzzles from this weeks UT newspaper, or find other puzzles on the web. (Be sure to credit your source of puzzles.) Time the SAT solver for various puzzles. Does the time to solve automatically the puzzles reflect the rated difficulty ratings?

To increase the challenge, sudoku puzzles can come in larger sizes. For example, the Sunday UT newspaper has 16 by 16 sized sudoku puzzles. How much longer does it take than for 9 by 9? If you are ambitious, 25 by 25 puzzles can be found on the internet.

Your project must describe your representation of the puzzles as CNF's, and explain why it is equivalent. You must describe what your source of puzzles was, and give timing info for MiniSAT on each puzzle, or compare times for different representations of the same puzzle. Mention any issues that arose, such as some representations taking very long times.