

CMSC 671 Artificial Intelligence - Fall 2013

Homework Assignment 3

Due at the start of class on October 2nd

From the Wikipedia Sudoku page:

“The objective is to fill a 9x9 grid with digits so that each column, each row, and each of the nine 3x3 sub-grids that compose the grid (also called “boxes”, “blocks”, “regions”, or “sub-squares”) contains all of the digits from 1 to 9. The puzzle setter provides a partially completed grid, which typically has a unique solution.”

Below is an example of an easy Sudoku:

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

- Formulate solving a Sudoku puzzle as constraint satisfaction. Submit answers to the questions below. (20 points)
 - What are the variables?
 - What are their domains?
 - What are the constraints formulated as binary constraints?
 - Does the requirement that each digit has to occur in each row, column, and block have to be directly specified? Why or why not?
- Write a Sudoku solver that does not represent the requirement that each digit has to occur in each row, column, and block directly. Your solver must use the AC3 algorithm. (50 points)
 - The input to your algorithm should be a 9x9 grid of characters that are either digits or '*' if the digit for that location is unspecified. It's output should be in the same form. If no solution can be found then simply output "no solution possible".
 - Submit your code and the result of running your program on the Sudoku above.

- Modify your solver to explicitly represent the the requirement that each digit has to occur in each row, column, and block directly. (30 points)
 - Submit your code and the result of running your program on the Sudoku above. Also run your algorithm on the two Sudoku below and report the results (either a solution, the fact that no solution is possible, or the fact that your code never terminated).

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

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