

Implementing a Sudoku Solver and Generator

Sudoku (Su Doku) is a popular puzzle in Japan with a simple rule: fill a given grid with the numbers 1 to 9, so that every column, row, and 3x3 box indicated by the slightly heavier lines has the numbers 1 to 9. The name is frequently translated as 'number place' in English, although other names such as 'sole number' may be better translations. Of course, in English, the name has multiple spellings: Su Doku is a common alternative. Plenty of samples can be found in newspapers and even booklets. Here is one:

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

Unsolved

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

Solved

The aims of this project are:

1-Implementing a SudoKu solver with the capability of reasoning. The most trivial solution of the puzzle is trying all possibilities for each unfilled square. But, this is really inefficient and needs to check nearly 1953125000000000 possible values. There are more efficient algorithms most of which try to prune the search space as much as possible. In addition to the solution itself, your solver must explain the user how it has come up with this solution. In this way, we can give some clues to the beginners for solving difficult puzzles.

2- Implementing a SudoKu generator which can generate SudoKu samples with a given rate. The rate of a puzzle shows how difficult is it to solve. The number of all possible SudoKu puzzles is 6,670,903,752,021,072,936,960 which is enormous. But, after removing symmetries, the number comes down to 5,472,730,538. You do not need to generate all of these. Your program needs to generate one random puzzle with a give rate in each execution.

The simplest puzzles are those can be solved deterministic. For such puzzles, we can always find an unfilled square with only one possible value. For example, in the above sample, the third square in the last row can get only 2 (why?). If there is no such square, we need to pick up one of the unfilled squares and choose one of its possible values and go forward. If we cannot solve the puzzle, we have to come back and choose another possible value for this square. The rate of a puzzle depends on the number of times we have to choose one value among all possible values for an unfilled square. Puzzles need more choices are more difficult.