

Sudoku solving methods

Article by King Seh Horng (Last update: 24 June 2009)

A. Introduction

Sudoku is a logic based, combinatorial number or letter placement puzzle. The objective of the puzzle is to fill a NxN grid so that the row, column and box will not have multiple occurrences of the same number. The classic Sudoku is played with 9x9 (NxN) cells and within these cells there are 3x3 sub boxes. This article will be focusing on the algorithms for varying NxN Sudoku. The only rules for the classic Sudoku are:

- Each row must have cells with unique number
- Each column must have cells with unique number
- Each sub box must have cells with unique number.
- The numbers used in the cells can only be 1 to N.

B. Logical approaches

For non-guessing Sudoku, it is possible to solely use logical approaches to complete the puzzle. Logical approaches can usually perform better than brute force and guessing algorithms. In some difficult Sudoku particularly when $N > 9$, guess algorithms and logical approaches must be used together for efficiency and speed. These algorithms must be used repetitively until the loop returns the same unsolved result as the previous loop. This means all the algorithms is exhausted and therefore, the Sudoku is unsolvable.

1. Row, column and box evaluations

Since Sudoku works on a simple logic that in each row, column and box, all the cells within them must contain a unique number. As such, it is possible to make an accurate approach. This algorithm must work by scanning each cell from first to last for solutions.

The program needs to collect the numbers used in the row, column and box settings in respect to each cell. After obtaining a list of used numbers, it should compute the possible solutions by comparing (1 to N) to list of the used numbers. If there is only one possible solution, the cell must contain that number and therefore, the overall Sudoku can be updated. The concept is illustrated below:

X		5		8		9		
2	7							
3								
4								
6								

By this approach, the algorithm can derive a solution with X equals to 1 because it is the only possible number for the Sudoku to be valid. This concept can also be understood from the visual interpretation of the puzzle.

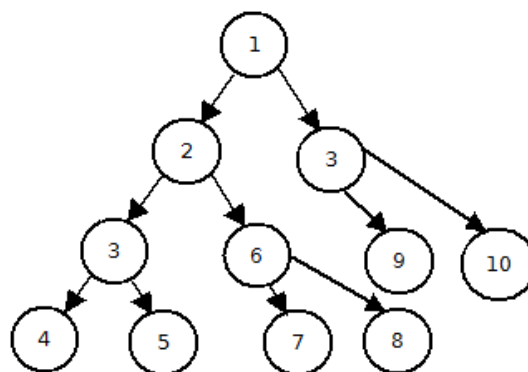
2. Possible solutions comparisons

In respect of each cell, this algorithm will get the list of possible solutions for only the other concerning unknown and known cells in the row, column or box settings (individual setting). From this list, the program can work out if only a total of (N-1) numbers are possible in the other cells. In other words, if this condition is true, the current cell must have that number not found in the list of possible solutions for the other concerning cells.

C. Illogical approach

The illogical approach which will be discussed here is the depth first search. This search algorithm is used for the guessing aspects. This is necessary when the logical approaches are exhausted.

To minimize the elapsed time for solving the puzzle, the logical approaches should be attempted again after each illogical approach is used. Nevertheless, there is a high chance that the previous guess is wrong which will end up in an unsolvable Sudoku. Hence, the program must revert one step back and attempt the alternative paths. In the event of reversion, the memory kept for the current paths can be cleared. A diagram of the depth first search concept is illustrated below:



The circle 1 is the initial Sudoku before any guessing and must either be derived through logical approaches or the given Sudoku. Furthermore, the circles under circle 1 denote the logically derived Sudoku after added a guessed number in one of the cells. This progress must be saved to allow backtracking if the initial guesses (e.g. 1 -> 2 -> 3 -> 4) resulted in an unsolvable Sudoku. In this instance, the algorithm should try the next guesses at 1 -> 2 -> 3 -> 5 and so on until a solution is found.

If backtracking is initiated in circle 1, this means all the possible guesses made below circle 1 leads to no solution and therefore, must be made to escape the loop. In this case, the Sudoku can be concluded as unsolvable.

D. Limitations

9x9 standard Sudoku has a significantly lesser guess branches in any arrangements of the cells when compared to $N > 9$ Sudoku puzzles. Therefore, it can enjoy the logical and illogical approaches described. However, depending on the nature of the Sudoku, some $N > 9$ Sudoku can have many branches in the guessing tree. This would cause the program to take minutes or even hours to compute a solution.