Sudoku Solver

Markos Galikas

February 2022

1 Sudoku

The Sudoku puzzle consists of a grid containing numbers. The aim is to fill all the cells satisfying certain constrains. The standard grid is 9×9 and consists of 9 (3×3) boxes, 9 rows, 9 columns and 81 cells. Initially it is filled with some numbers/clues (Figure 1)[1].

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

Figure 1: A typical Sudoku puzzle

The constrain for this grid, and in a similar way for other grids, are:

- Use only numbers from 0-9.
- Row-Column: Each cell must contain a number.
- Row-Number: Each row can only contain each number from 0-9 once.
- Column-Number: Each column can only contain each number from 0-9 once.
- Box-Number: Each box can only contain each number from 0-9 once.

2 Exact cover

One can start by implementing a backtracking algorithm in order find the solution. This can be substantially slow as the puzzle difficulty increases. The rule is that the fewer the clues the harder it gets, although for uniquely solvable

Sudokus(well-formed) it has to have at least 17. For an efficient solution to a Sudoku puzzle, we model the problem as an exact cover one.
Given a collection S of subsets and a set X, an exact cove is the sub-collection of S that cover all the elements in X. For instance:[2]

Let S = {A, B, C, D, E, F} be a collection of subsets of a set X = {1, 2, 3, 4, 5, 6, 7} such that:

$$\mathbf{S} = \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{matrix} \\ \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 7 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array} \begin{array}{c} = X \\ A \\ B \\ C \\ D \\ E \\ F \end{array}$$

Then the sub-collection {B, D, F} is an exact cover, since each element in X is contained in exactly one of the subsets: B = {1, 4}, D = {3, 5, 6}, F = {2, 7}.

In the same manner we can construct the Sudoku's exact cover matrix. We can viewed the problem as a way to select possibilities (subsets of S) such as that each constraint (X) is 'hit' by exactly one selected possibility. The problem in Sudoku is to assign numbers in a grid so as to satisfy certain constraints. In the standard 9×9 Sudoku variant, the four kinds of constraints are Row-Column(p), Row-Number(rn), Column-number(cn), Box-Number(bn). We number the rows, columns, and boxes from 0 to 8. Then every cell s_{ij} contains 4 constrains(Knuth,2020).

$$p_{ij}\ rn_{ik}\ cn_{jk}\ bn_{xk} \text{ for } 0 \leq i, j < 9,\ 1 \leq k \leq 9 \text{ and } x = 3(i/3) + (j/3)$$

If $s_{ij} = k$ then the number 'k' is placed in the box 'x', in row 'i' and column 'j'. Consequently, for an empty grid, because each of the $9 \cdot 9 = 81$ cells is assigned one of 9 numbers, there are $81 \cdot 9 = 729$ possibilities and since there are 9 rows, 9 columns, 9 boxes and 9 numbers, the total number of constrains are ($9 \cdot 9$ for p + $9 \cdot 9$ for 'rn' + $9 \cdot 9$ for 'cn' + $9 \cdot 9$ for 'bn') = 324.

Row-Column Constraints

	R1 C1	R1 C2	⋯
R1C1#1	1	0	⋯
R1C1#2	1	0	⋯
R1C1#3	1	0	⋯
R1C1#4	1	0	⋯
R1C1#5	1	0	⋯
R1C1#6	1	0	⋯
R1C1#7	1	0	⋯
R1C1#8	1	0	⋯
R1C1#9	1	0	⋯
R1C2#1	0	1	⋯
R1C2#2	0	1	⋯
R1C2#3	0	1	⋯
R1C2#4	0	1	⋯
R1C2#5	0	1	⋯
R1C2#6	0	1	⋯
R1C2#7	0	1	⋯
R1C2#8	0	1	⋯
R1C2#9	0	1	⋯
⋮	⋮	⋮	⋱

Row-Number Constraints

	R1 #1	R1 #2	⋯
R1C1#1	1	0	⋯
R1C1#2	0	1	⋯
⋮	⋮	⋮	⋱
R1C2#1	1	0	⋯
R1C2#2	0	1	⋯
⋮	⋮	⋮	⋱
R1C3#1	1	0	⋯
R1C3#2	0	1	⋯
⋮	⋮	⋮	⋱
R1C4#1	1	0	⋯
R1C4#2	0	1	⋯
⋮	⋮	⋮	⋱
R1C5#1	1	0	⋯
R1C5#2	0	1	⋯
⋮	⋮	⋮	⋱
R1C6#1	1	0	⋯
R1C6#2	0	1	⋯
⋮	⋮	⋮	⋱
R1C7#1	1	0	⋯
R1C7#2	0	1	⋯
⋮	⋮	⋮	⋱
R1C8#1	1	0	⋯
R1C8#2	0	1	⋯
⋮	⋮	⋮	⋱
R1C9#1	1	0	⋯
R1C9#2	0	1	⋯
⋮	⋮	⋮	⋱

Column-Number Constraints

	C1 #1	C1 #2	⋯
R1C1#1	1	0	⋯
R1C1#2	0	1	⋯
⋮	⋮	⋮	⋱
R2C1#1	1	0	⋯
R2C1#2	0	1	⋯
⋮	⋮	⋮	⋱
R3C1#1	1	0	⋯
R3C1#2	0	1	⋯
⋮	⋮	⋮	⋱
R4C1#1	1	0	⋯
R4C1#2	0	1	⋯
⋮	⋮	⋮	⋱
R5C1#1	1	0	⋯
R5C1#2	0	1	⋯
⋮	⋮	⋮	⋱
R6C1#1	1	0	⋯
R6C1#2	0	1	⋯
⋮	⋮	⋮	⋱
R7C1#1	1	0	⋯
R7C1#2	0	1	⋯
⋮	⋮	⋮	⋱
R8C1#1	1	0	⋯
R8C1#2	0	1	⋯
⋮	⋮	⋮	⋱
R9C1#1	1	0	⋯
R9C1#2	0	1	⋯
⋮	⋮	⋮	⋱

Box-Number Constraints

	B1 #1	B1 #2	⋯
R1C1#1	1	0	⋯
R1C1#2	0	1	⋯
⋮	⋮	⋮	⋱
R1C2#1	1	0	⋯
R1C2#2	0	1	⋯
⋮	⋮	⋮	⋱
R1C3#1	1	0	⋯
R1C3#2	0	1	⋯
⋮	⋮	⋮	⋱
R2C1#1	1	0	⋯
R2C1#2	0	1	⋯
⋮	⋮	⋮	⋱
R2C2#1	1	0	⋯
R2C2#2	0	1	⋯
⋮	⋮	⋮	⋱
R2C3#1	1	0	⋯
R2C3#2	0	1	⋯
⋮	⋮	⋮	⋱
R3C1#1	1	0	⋯
R3C1#2	0	1	⋯
⋮	⋮	⋮	⋱
R3C2#1	1	0	⋯
R3C2#2	0	1	⋯
⋮	⋮	⋮	⋱
R3C3#1	1	0	⋯
R3C3#2	0	1	⋯
⋮	⋮	⋮	⋱

Figure 2: Snapshot of Sudoku Cover Matrix[3]

For better understanding we can work though an example. Let us examine the first cell $s_{0,0}$. For the sake of the example we assume that the correct answer is number 1. So the first row of the cover matrix is chosen as a partial solution - subset. Now we examine the columns indicated by a one for that row. These columns are the 4 constrains $p_{0,0}$, $rn_{0,1}$, $cn_{0,1}$, $bn_{0,1}$. Traversing through these columns we remove the particular rows that have also a one. For instance the column $rn_{0,1}$ has ones for the rows {i,j,k} $\Rightarrow$ [{0,0,1}, {0,1,1}, {0,2,1}, ...], meaning that by removing these rows we can no longer place the number 1 anywhere else in that particular row. Then we remove the column of the constrain as it has been fulfilled.

3 Algorithm X

The exact cover problem is represented in Algorithm X by a matrix A consisting of 0s and 1s.The goal is to select a subset of the rows such that the digit 1 appears in each column exactly once, which is exactly what we want for the sudoku solution, ie. every restriction(=column) is fulfilled. The pseudo code is as follows[4]:

If the matrix A has no columns, the current partial solution is a valid solution; terminate successfully

1. Otherwise choose a column c (deterministically).
2. Choose a row r such that Ar, c = 1 (nondeterministically).
3. Include row r in the partial solution.
4. For each column j such that Ar, j = 1,

   for each row i such that Ai, j = 1,

      delete row i from matrix A.

   delete column j from matrix A.
5. Repeat this algorithm recursively on the reduced matrix A.

4 References

1 A typical Sudoku puzzle. [Photograph]. Sudoku, from `https://en.wikipedia.org/wiki/Sudoku`

2 Wikimedia Foundation. Exact cover. Wikipedia. Retrieved February 6, 2022, from `https://en.wikipedia.org/wiki/Exact_cover#Detailed_example`

3 Knuth, D. E. (2020). The Art of Computer Programming (Vol. 4, Ser. PRE-FASCICLE 5C). Addison Wesley, from https://www.inf.ufrgs.br/~mrpritt/lib/exe/fetch.php?media=inf5504:7.2.2.1-dancing_links.pdf

4 Wikimedia Foundation. Knuth's Algorithm X. Wikipedia. Retrieved February 6, 2022, from https://en.wikipedia.org/wiki/Knuth%27s_Algorithm_X