SUDOKU SOLVER

November 21, 2024

A Project by

Abhinav Bitragunta

Aman Gupta

Srinidhi Sai Boorgu

Ansh Jain

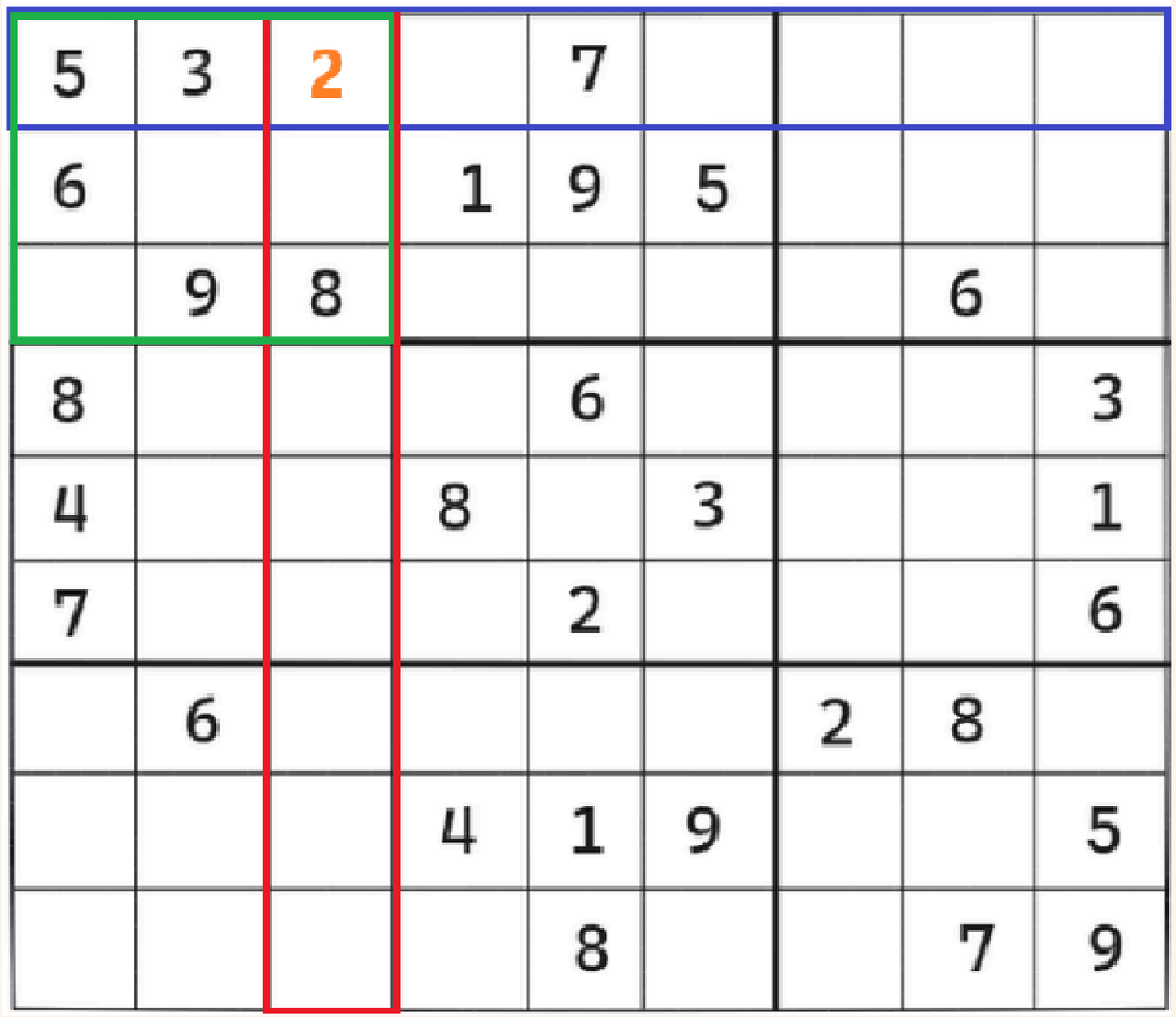
INITIAL STEPS

Regular backtracking, O(n) search time

Regular backtracking using hash arrays, O(1) search

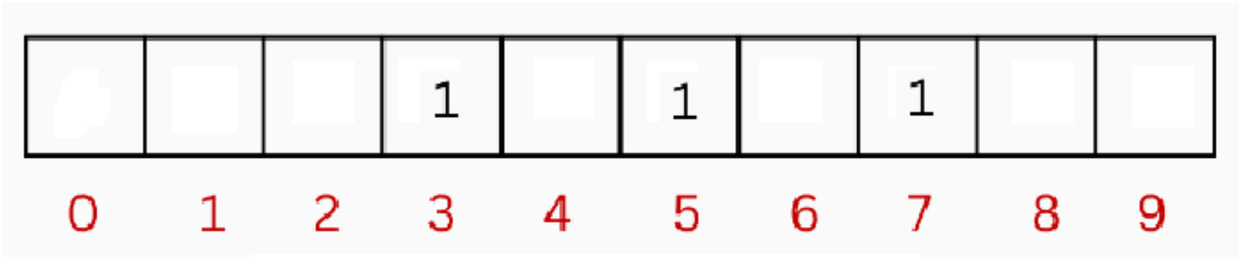
Regular backtracking, hashing with bits.

Looking for faster algorithms...


5 3 2 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

ARRAY HASHING

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



BIT HASHING

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

0 0 1 0 1 0 1 0 0 0

9 8 7 6 5 4 3 2 1 0

FASTER ALGORITHMS

Sudoku, being NP-complete, can be converted into other NP-complete problems. There exist efficient algorithms such problems.

The exact cover problem is one such problem, and Donald Knuth’s ‘Algorithm X’ implemented using ‘Dancing Links’ is a known efficient algorithm used to solve it.

THE EXACT COVER PROBLEM

Given a set S and another set where each element is a subset to S, is it possible to select a set of subsets such that every element in S exist in exactly one of the selected sets?

$$S = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

$$S^* = \{\{1, 2, 5, 7\}, \{3, 4, 6\}, \{1, 5, 9\}, \{1, 3, 4, 5, 6, 8, 9\}, \{2, 7, 8\}\}$$

$$A = \{1, 2, 5, 7\}$$

$$B = \{3, 4, 6\}$$

$$C = \{1, 5, 9\}$$

$$D = \{1, 3, 4, 5, 6, 8, 9\}$$

$$E = \{2, 7, 8\}$$

BINARY MATRIX REPRESENTATION

$$A = \{1, 2, 5, 7\}$$
$$B = \{3, 4, 6\}$$
$$C = \{1, 5, 9\}$$
$$D = \{1, 3, 4, 5, 6, 8, 9\}$$
$$E = \{2, 7, 8\}$$

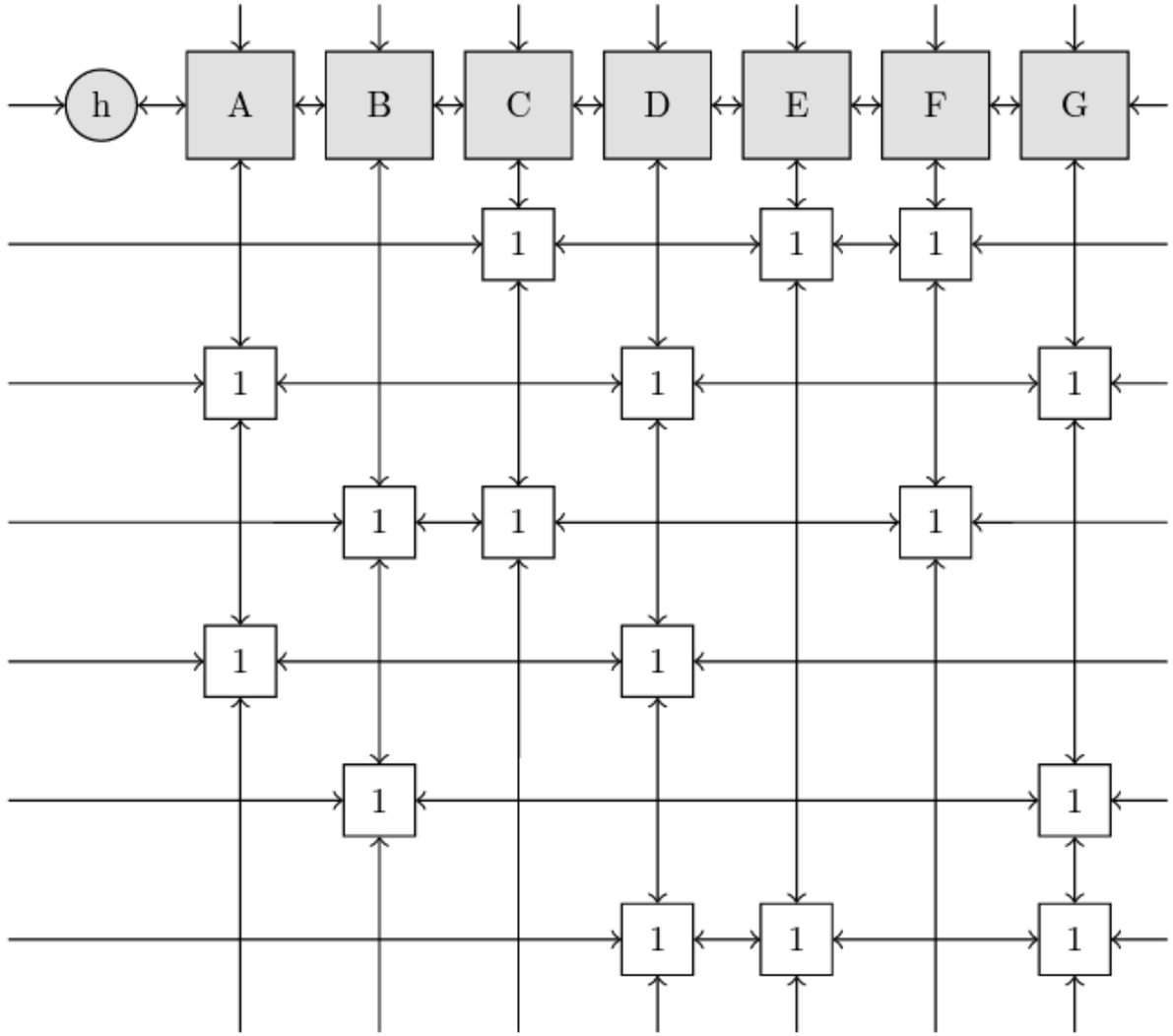
$$\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array} \begin{array}{c} \begin{array}{ccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix} \end{array}$$

ALGORITHM X

$$
\begin{array}{c} \\ A \\ B \\ C \\ D \\ E \end{array}
\begin{array}{c}
\begin{array}{ccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{array} \\
\left(\begin{array}{ccccccccc}
1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0
\end{array}\right)
\end{array}
$$

TOROIDAL LINKED LIST

$$\begin{matrix} A & B & C & D & E & F & G \end{matrix}$$
$$\begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix}$$



EXACT COVER

Convert

Convert Sudoku into the exact cover problem

Solve

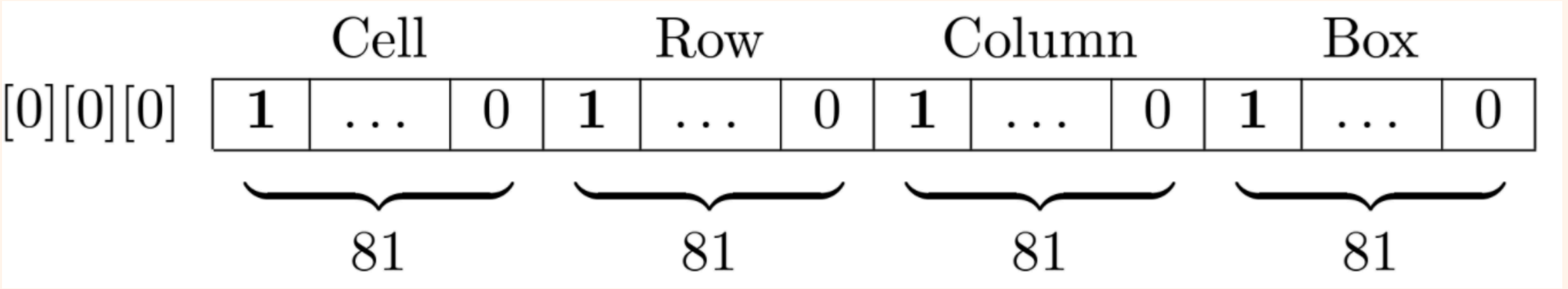
Identify constraints and solve using Knuth’s DLX

Convert back

Convert solution into solved Sudoku format

CONVERSION TO EXACT COVER

Represent a choice as [Row index][Column index][Value - 1]




```
RCidx   = row      * n   + col                , //constr 0: cell
RVidx   = row      * n   + val + n * n        , //constr 1: row-val
CVidx   = col      * n   + val + 2 * n * n    , //constr 2: col-val
GVidx   = gridnum  * n   + val + 3 * n * n    ; //constr 3: grid-val
```

CONVERTING BACK TO SUDOKU

```
constraintType  = (solutionRowNode->col->index)/(n*n);

if(constraintType == 0)      cell_index = solutionRowNode->col->index;
else                         val = (solutionRowNode->col->index)%(this->n) + 1;
```

THANK YOU