SHUFFLE AND COLOR A SUDOKU

Sander Perens

Institute of Computer Science, University of Tartu

UNIVERSITY OF TARTU 1632

Background

Sudoku is a logic-based combinatorial number-placement puzzle. There are 81 cells divided into 9 rows and 9 columns. The objective is to fill all empty cells so that in every row, every column and in every 3x3 subgrid a number from 1 to 9 appears only once.

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

Fig. 1: A typical Sudoku puzzle [1

Measuring implementation

We implemented three different algorithms that can successfully solve a Sudoku puzzle:

- Simulated annealing (stochastic)
- Backtracking
- Graph coloring with backtracking

We measured each algorithm on three difficulties. Each algorithm got the same Sudokus to solve at corresponding level. There were 20 different Sudokus used for each level. Algorithms were measured with following difficulties:

- **Easy** - 35 empty cells
- **Medium** - 43 empty cells
- **Hard** - 48 empty cells

Simulated annealing

Simulated annealing is a probabilistic technique that can be used to approximate the global optimum of a function. We select a neighbor by random, if we would be in better situation, we go there. If not, we still go there with some probability, allowing us to get out of local optimum.

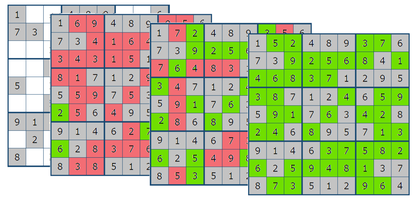

Fig. 2: Approaching global optimum with simulated annealing

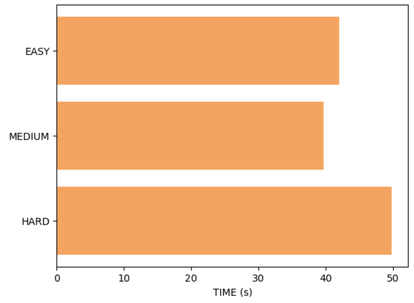

Fig. 3: Solving duration for simulated annealing

This was the most steady algorithm and performed fairly uniform over all difficulties.

Backtracking

Backtracking algorithm searches the solution recursively. It traverses the search tree in depth-first order. At each cell it checks if the solution is still valid. If state is not valid, we try another number, if still not valid, we step back and try another number in previous cell.

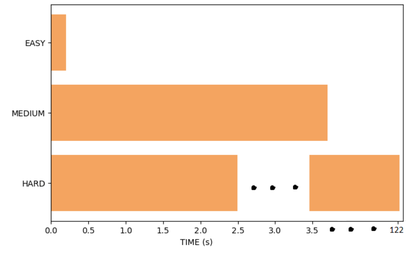

Fig. 4: Solving duration for backtracking

Backtracking was fast when the search space was small.

Graph coloring with backtracking

Graph coloring is a task where no two adjacent vertices have the same color. In Sudoku adjacent vertices are in the same row, in the same column and in the same 3x3 subgrid. Graph coloring with backtracking checks if any adjacent vertix already has that color. If no adjacent vertix has the color, it's assigned to that vertix. If it doesn't succeed with assigning a color, it backtracks.

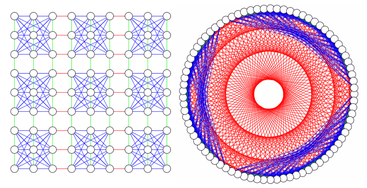

Fig. 5: Sudoku connectivity graph [2

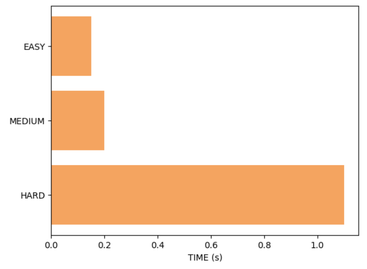

Fig. 6: Solving duration for graph coloring

Graph coloring with backtracking was the best performing approach on all levels of difficulty.

Conclusion

We implemented three algorithms that can successfully solve a Sudoku. One stochastic and two backtracking methods. Graph coloring with backtracking was the fastest to come to the right solution.

References

[1] Visualizing the Sudoku Connectivity Graph `https://en.wikipedia.org/wiki/Sudoku#/media/File:Sudoku_Puzzle_by_L2G-20050714_standardized_layout.svg`

[2] Visualizing the Sudoku Connectivity Graph `https://www.reddit.com/r/dataisbeautiful/comments/6ty4vf/visualizing_the_sudoku_connectivity_graph_more_in/`