CSC 320 – Sudoku as SAT

Jonah Rankin – V00808910
Cameron Long V00748439

Summary of Findings

Comparison to Other Solvers

Traditionally special-purpose Sudoku solvers rely on a naïve implementation of backtracking or search algorithms to sift through the puzzle's 6 * 10^{21} possible states. More advanced solvers rely on some sort of search heuristic or constraint propagation to mitigate extensive searching. This is much different than a SAT solver, which translates the problem into a propositional formula that is satisfiable if and only if the SAT solver can find a satisfactory assignment. This implementation requires minimal implementation effort as there is already much existing framework.

The Sudoku Solver was subjected to several test cases, including the difficult puzzles found on norvig.com. Our solver had the following runtimes and results from input puzzles:

Puzzle Difficulty	Runtime(s)	Norvig Runtime	Result
Medium	0.005609	0.01	Satisfiable
Medium	0.006581	0.01	Satisfiable
Hard	0.006912	188.79	Satisfiable
Extreme	0.007541	*untested*	Satisfiable
Impossible	0.20753	1439	Unsatisfiable

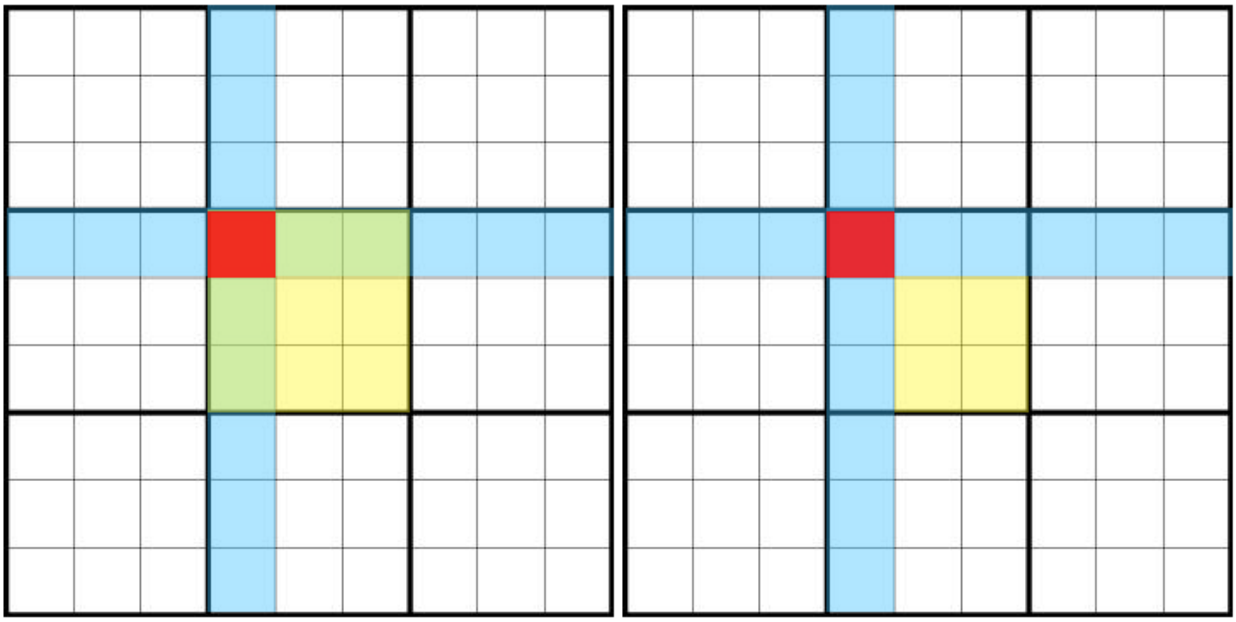
The SAT solver is a massive improvement over a brute-force approach to solving a Sudoku puzzle. The Norvig puzzles struggled with puzzle that had minimal data with one single solution. The SAT solver however, does increase runtime by a factor of 37, but still completes with a mere 0.21 second runtime.

Alternate Encodings

Since the encoding of the puzzle encodes each column, row and 3 x 3 sub grid, there are redundant calculations in the uniqueness of some of the cells in the puzzle. These redundant cells are indicated by the green cells in Figure 1. The alternate encoding of the puzzle removes these redundant cells from the subgrid clauses and improves the performance. As shown in table 1, this improvement is most significant for difficult problems.

Puzzle Difficulty	Original Runtime(s)	Alternate Runtime(s)	Improvement
Medium	0.005609	0.005449	2.9%
Medium	0.006581	0.006212	5.6%
Hard	0.006912	0.006292	8.9%
Extreme	0.007541	0.006593	12.6%
Impossible	0.20753	0.164189	20.9%

Table 1: Encoding performance comparison



Original "Minimal Encoding" *Optimized Subgrid encoding*

Figure 1: New Encoding Procedure

N x N Size puzzles

The translation of encodings can be easily extended to more generalized puzzles. Traditional encodings contains $9^3 = 729$ variables in its encoding, representing the row, column and cell logic needed to solve the puzzle. So, the runtime increases in polynomial time with the size of the grid, and since the Sudoku problem is NP-Complete, no better algorithm complexity is known.