# Sudoku Coursework Report

The approach that I took for the Sudoku solver combines Constraint Propagation (CP) with optimised backtracking and involves advanced heuristics such as Minimum Remaining Value (MRV) and Least Constraining Value (LCV). This strategy is more ambitious and complex than simple brute-force backtracking methods, such as basic backtracking that tries every possible number in every empty cell. Constraint propagation reduces the search space by eliminating impossible values (Russell and Norvig, 2016, p. 203). It reduces the size of the problem before proceeding to backtracking is even necessary. By optimising the backtracking phase with MRV and LCV, the solver not only selects the most constrained cell to fill next (MRV) (Russell and Norvig, 2016, p. 216) but also prioritises the filling order based on which choices are least likely to lead to a conflict (LCV) (Russell and Norvig, 2016, p. 216). The chosen method is well-suited for the task of Solving Sudoku. It mimics human-like reasoning in solving puzzles — starting with obvious choices (constraint propagation: naked and hidden singles) and then exploring less certain options (backtracking with heuristics).Previous research demonstrated that combining Backtracking Search with CP and the MRV heuristic leads to a marked reduction in the search space (Schermerhorn, 2015). My approach extends this further by also incorporating the LCV.

Initially, the algorithm uses CP methods, in particular, naked and hidden singles, to simplify the puzzle grid. Naked singles (Naked Single, 2024) are cells with one possible candidate (Candidate, 2024), while hidden singles are the sole candidates within a row, column, or block, despite potentially being obscured among other candidates (Hidden Singles, 2024). Mathematically, this can be described as reducing the set of candidates $C_{i,j}$ for each cell $(i, j)$ to a singleton based on the constraints of the game. After exhausting the CP methods, it invokes backtracking, navigated by the MRV heuristic, which preferentially selects the cell with the fewest legal candidates, thereby minimising the branching factor. The MRV heuristic doesn't give an exact answer in situations where it identifies at least two variables with the same minimal domain size (Bakker, 2015). This is where the LCV decides the order in which to try the candidate numbers for that cell by prioritising those that leave the maximum degree of freedom for subsequent cell assignments. If MRV narrows down the choices for "where to play next," LCV helps with "what to play." Formally, if $S$ represents the state of the Sudoku grid and $V_{i,j}$ the set of potential values for cell $(i, j)$, the backtracking function seeks a solution where the next cell selection $(i', j')$ satisfies $|V_{i',j'}| = \min_{(i,j)\in S} |V_{i,j}|$ (MRV), and the values are attempted in ascending order of the count of their elimination from the remaining cells' candidate sets (LCV).

Prior to backtracking optimisation, the solver's performance on Hard Sudoku number 10 was measured at 27.70 seconds, and number 11 at 45.43 seconds. With the integration of MRV and LCV, the resolution time jumped to an impressive 0.29 and 0.79 seconds respectively, indicating a significant efficiency improvement. From a computational complexity perspective, the CP methods reduce the upper bound of the search space from $O(9^n)$ — where $n$ is the number of empty cells — to a potentially much smaller space by eliminating impossibilities. The MRV heuristic decreases the branching factor by focusing on cells with fewer legal moves and reduces the average case time complexity. LCV minimises the likelihood of further backtracks and allows a more linear progression through the solution space.

I evaluated the efficacy of the solver against challenging puzzles, and quantitative metric demonstrates the practical benefits of integrating MRV and LCV heuristics with constraint propagation. Qualitatively, the solver closely mimics human deductive techniques, systematically narrowing down possibilities for a user-friendly and logical solving process. However, the inherent exponential complexity of backtracking suggests room for advancement. Future work could explore the use of machine learning algorithms to predict the most promising search paths, or the development of parallel processing techniques to concurrently explore multiple branches of the solution space.

## References


Bakker, J., 2015. *A generic solver for constraint satisfaction problems*. Ph.D. thesis. Faculty of Science and Engineering.

Candidate, 2024. [Online]. Accessed: March 24, 2024. Available from: http://sudopedia.enjoysudoku.com/Candidate.html.

Hidden singles, 2024. [Online]. Accessed: March 24, 2024. Available from: https://sudoku.com/sudoku-rules/hidden-singles/.

Naked single, 2024. [Online]. Accessed: March 24, 2024. Available from: http://sudopedia.enjoysudoku.com/Naked_Single.html.

Russell, S.J. and Norvig, P., 2016. *Artificial intelligence: a modern approach*. Pearson.

Schermerhorn, M., 2015. A sudoku solver.