Sudoku, Search, and Constraint Satisfaction

Taron Moe-Stull
taronmoe@icloud.com

September 2025

Abstract

This paper presents an experimental evaluation of five algorithmic approaches for solving standard 9×9 Sudoku puzzles across multiple difficulty levels. Three deterministic constraint-based methods: Backtracking (BT), Forward Checking (FC), and Arc Consistency (AC-3), are evaluated alongside two stochastic approaches, Simulated Annealing (SA) and a Genetic Algorithm (GA). Performance is measured using decision-based metrics rather than runtime, including assignments, backtracks, prunes, arc revisions, and fitness evaluations.

The constraint-based methods reliably solve easy and medium puzzles, with Forward Checking and AC-3 significantly reducing search compared to plain backtracking. AC-3 demonstrates the strongest overall performance, maintaining low search cost even as puzzle difficulty increases. While the stochastic methods do not consistently produce complete solutions, both exhibit measurable improvement over time through their learning curves. These results highlight the effectiveness of constraint propagation for structured combinatorial problems and the exploratory potential of stochastic optimization techniques.

Problem Statement and Hypothesis

The objective of this work is to design a system capable of reading Sudoku puzzles and solving them using one of five algorithmic strategies, while evaluating performance using decision-based metrics rather than time-based measurements. Each algorithm operates on identical puzzle inputs and produces validated outputs, allowing direct comparison across methods.

It is hypothesized that plain Backtracking will solve easy and medium puzzles but will exhibit rapidly increasing backtracking on harder puzzles. Forward Checking is expected to reduce unnecessary search by pruning invalid assignments earlier in the search process, leading to improved performance. Arc Consistency (AC-3) is expected to further reduce search by enforcing constraint consistency across related cells, making it the strongest deterministic approach overall.

The stochastic approaches, Simulated Annealing and the Genetic Algorithm, are not expected to consistently solve puzzles to completion. However, it is hypothesized that both methods will demonstrate improving fitness trends over iterations or generations, even when a complete solution is not reached.

Algorithms Implemented

Backtracking (BT) performs a depth-first search over empty cells using a fixed left-to-right variable ordering. Values are attempted sequentially, and the algorithm backtracks upon encountering an

invalid assignment. Minimum remaining value heuristics are intentionally not used in order to provide a baseline comparison against more advanced constraint-based methods. Performance is measured using assignment attempts and backtracks.

Forward Checking (FC) extends backtracking by maintaining a domain of allowable values for each unassigned cell. After each assignment, inconsistent values are pruned from neighboring domains. If any domain becomes empty, the algorithm fails immediately and backtracks. Variable selection remains first-empty, and performance metrics include assignments, backtracks, and total prunes.

Arc Consistency (AC-3) enforces consistency across all related cell pairs by iteratively removing values that lack supporting alternatives in neighboring domains. The implementation tracks assignments, prunes, backtracks, and arc revisions. This additional constraint propagation significantly reduces search space on harder puzzles.

Simulated Annealing (SA) begins with a complete board that satisfies all 3×3 subgrid constraints. Fitness is defined as the number of conflicts in rows and columns, where lower values indicate better solutions and zero represents a valid solution. Neighbor states are generated by swapping two non-given values within a randomly selected subgrid. Moves that improve fitness are accepted, while worse moves may be accepted probabilistically based on a temperature schedule.

The Genetic Algorithm (GA) represents each individual as a board that respects all given values. Fitness is determined by counting duplicate values in columns and subgrids. Tournament selection is used to choose parents, crossover occurs on a row-wise basis, and mutation is implemented through swaps of non-given values. The best individual from each generation is preserved, and progress is tracked through fitness evaluations and learning curves.

Experimental Approach

All experiments were conducted within a consistent execution framework. Each run begins by loading and validating a puzzle before applying the selected solver. Deterministic algorithms were executed once per puzzle and difficulty, while stochastic methods were executed multiple times to account for inherent variability.

Decision-based metrics were collected during execution, and all produced boards were validated against standard Sudoku constraints prior to being recorded as solutions. Hyperparameters for SA and GA were selected through preliminary experimentation to ensure meaningful progress without excessive computational cost.

Results

Easy Puzzles

Puzzle	Solver	Solved	Assignments	Backtracks	Prunes	Arc Revisions
Easy-P1	BT	Yes	150	102	–	–
Easy-P2	BT	Yes	678	633	–	–
Easy-P1	FC	Yes	58	10	102	–
Easy-P2	FC	Yes	687	642	1833	–
Easy-P1	AC-3	Yes	48	–	93	93
Easy-P2	AC-3	Yes	45	–	94	94

Discussion and Summary

The experimental results align closely with the initial hypotheses. For deterministic methods, Backtracking demonstrates rapidly increasing search cost as puzzle difficulty increases. Forward Checking consistently reduces search by identifying dead ends earlier, while Arc Consistency (AC-3) demonstrates the strongest and most scalable performance across all tested difficulties.

The stochastic methods proved less reliable in producing complete solutions. However, their learning curves clearly demonstrate progressive improvement in fitness over time. Simulated Annealing successfully reached valid solutions in select cases, while the Genetic Algorithm consistently reduced fitness across generations without full convergence.

Overall, this work highlights the effectiveness of constraint propagation for structured combinatorial problems and demonstrates the potential of stochastic optimization methods when combined with sufficient computational resources or hybrid strategies.