Defining a Problem

Evolutionary Sudoku Solver Optimization Genetic Algorithm

Team
Joshua Soteras
Sanskar Thapa
Brian Mcnaughton
Brian Pham
Neel khunt

CS 5660 Advanced Topics In Artificial Intelligence

November 11th, 2025

Step 1: What is the Problem

Informal Description

Our team needs to create an AI system that can effectively solve Sudoku puzzles of varying difficulty levels through the use of evolutionary algorithms.These will help us narrow down valid and efficient solutions faster than slower methods such as backtracking.

Formal Definition

Evolutionary Algorithms are used for Optimization Problems, hence our Sudoku Solving Problem. The reason for solving a Sudoku grid is considered an optimization problem: the goal is **feasibility** (finding any valid grid) with **efficiency** (finding this valid grid or solution as *quickly* as possible).

- Objective Function: f(x)
    - Minimize the number of constraint violations in a 9x9 Sudoku grid
    - f(x) = 0 represents a valid solution
- Decision Variables
    - 81 integer variables representing each cell in the grid (values 1-9)
- Constraints
    - Each row must contain digits 1-9 without repetition
    - Each column must contain digits 1-9 without repetition
    - Each 3x3 sub-grid must contain digits 1-9 without repetition
    - Pre-filled cells must maintain their given values

Step 2: Why does the Problem Need to be Solved?

Motivation: Educational Value

This problem serves education value in the form of understanding how evolutionary algorithms handle constraint satisfaction problems. It demonstrates evolutionary algorithm concepts like fitness functions, genetic operators, and population dynamics in a familiar, visualizable domain.

Solution Benefits: Algorithm Research

This problem gives a well defined benchmark for testing and comparing different evolutionary algorithms, selection methods, and genetic operators. Benchmarks like success rates and convergence speed are easily measured and compared to each other.

Solution Use: Practical Applications

Sudoku in itself is primarily a recreational activity designed to help someone combat boredom or stimulate their brain, but solving Sudoku with evolutionary algorithms benefits real-world constraint satisfaction and scheduling problems in planning systems, resource allocation, and operations research.

Step 3: How Would I Solve the Problem?

Manual Steps to Approach

On an ordinary sudoku 9x9 grid, it is based on edge cases and strategies that could be identified. Depending on the complexity of the grid and the randomization of numbers positioned, we would track the amount of time it takes to solve a 9x9 sudoku grid, even bigger sudoku grids if needed to test constraints.

Manual Solution Designs

Algorithms like deterministic backtracking for each row and column that is completed from the grid, in addition to priority queues to choose the next number that is needed on the grid would be the initial manual solution we would start off with.

Prototypes

On our initial approach, we could probably use multiple prototypes, especially to implement rules of sudoku. One prototype could consist of the number of puzzles that are solved without searching, and the accuracy of completion. Another prototype could consist of implementing rules that would affect the completion rate of the sudoku grid, as well as finding scenarios that would have a grid that has multiple solutions or no solution at all.

- Prototypes/Experiment Ideas
  - Test different population sizes (50 - 500 individuals)
  - Initial Population: Have different board configurations (random valid digit placements with the board)
  - Compare Mutation Rates
  - Benchmark Against Backtracking Algorithms to determine efficiency and solution quality .

Reflection

Attempting to find a valid solution to a Sudoku board in an efficient time highlights the importance of this being categorized as an Optimization Problem rather than a learning problem. Determining and brainstorming the steps exhibits how evolutionary algorithms transform a discrete puzzle into a continuous search through a solution space. The main challenge is going to be designing genetic operators that respect Sudoku's strict constraints while maintaining population diversity. Defining this problem showcased that success depends heavily on problem representation and fitness function design, demonstrating that puzzles like Sudoku can benefit from AI approaches when traditional methods still struggle.