IIT DHARWAD

Department of Computer Science and Engineering
2022 Batch

|| सा विद्या या विमुक्तये ||

भारतीय प्रौद्योगिकी संस्थान धारवाड
Indian Institute of Technology Dharwad

Sudoku Solver

Contributors

Varun Kumar(220010021)
Pardeev Reddy(220010040)
Sai Tharun(220010057)
Sucharitha(220110005)
Sayan Mal(220120021)
Shiva Prasad(220010054)

Monu Soyal
Advisor

April 09, 2024

INTRODUCTION

Sudoku, a popular puzzle game, has intrigued enthusiasts for decades with its simple rules yet challenging gameplay. As technology advances, the application of artificial intelligence (AI) techniques in solving Sudoku puzzles has gained significant attention. The integration of AI algorithms offers promising solutions to efficiently solve Sudoku puzzles, ranging from traditional backtracking approaches to more sophisticated techniques like constraint satisfaction and genetic algorithms. In this project, we explore the implementation of an AI-powered Sudoku solver, aiming to develop a robust and efficient system capable of solving Sudoku puzzles of varying complexities. By leveraging AI, we seek to provide an intelligent solution that not only solves Sudoku puzzles but also demonstrates the potential of AI in solving real-world problems efficiently.

Chapter 1

RESEARCH STATEMENT

1.1 Problem Identification

We will be trying to evaluate various techniques for solving Sudoku puzzles by a machine. We would be learning sudoku puzzles of various dimensions which would be solved using artificial intelligence techniques like backtracking, genetic algorithms, simulated annealing etc.

Within the problem statement, the following questions should be addressed:

- What is a Sudoku?
- Why Sudoku?
- What algorithms will we use in this project?
- What will we be learning from completing this project?

1.2 Problem Formulation

For our Sudoku solver project, we plan to employ a combination of backtracking, heuristic approach, and simulated annealing techniques to efficiently tackle the Sudoku puzzle-solving problem. Backtracking will serve as the foundational algorithm, systematically exploring different possible solutions and backtracking when encountering dead ends. Additionally, we'll integrate heuristic approaches to guide the search process, prioritizing the most promising paths based on heuristic evaluations of the puzzle state. Furthermore, simulated annealing, inspired by the annealing process in metallurgy, will allow us to explore a diverse solution space while gradually converging towards optimal solutions by probabilistically accepting worse solutions initially. By leveraging these complementary techniques, we aim to develop a robust and versatile Sudoku solver capable of efficiently handling puzzles of varying complexities.

Chapter 2

INDIVIDUAL CONTRIBUTION

Week-1: Sudoku Solver using Backtracking

- **Varun and Sucharitha:** Implementing the core Sudoku solving logic based on the backtracking algorithm.
- **Tharun and Pardeev:** Developing helper functions to support the Sudoku solving logic, such as checking for valid moves and updating the puzzle state.
- **Sayan and Shiva:** Writing example usage scenarios to demonstrate how to use the implemented Sudoku solver.

Week-2: Backtracking with Heuristic Approach

- **Tharun and Sayan:** Enhancing the backtracking algorithm with a heuristic approach to improve efficiency.
- **Varun and Pardeev:** Continuing to develop helper functions required for the backtracking solver with heuristic approach.
- **Shiva and Sucharitha:** Implementing input functions for Sudoku puzzles and functions for printing the solved puzzle.

Week-3: Sudoku Solver using Simulated Annealing (1st part)

- **Sucharitha and Pardeev:** Defining class structures and data representations for the Sudoku solver using simulated annealing.
- **Sayan and Varun:** Creating evaluation functions to assess the quality of solutions generated by simulated annealing.
- **Tharun and Shiva:** Providing example usage scenarios for the simulated annealing-based Sudoku solver.

Week-4: Sudoku Solver using Simulated Annealing (2nd part)

- **Tharun and Sucharitha:** Implementing the simulated annealing algorithm for solving Sudoku puzzles.
- **Varun and Pardeev:** Developing perturbation functions to introduce randomness and variation in the solution search process.
- **Sayan and Shiva:** Writing input processing functions to prepare Sudoku puzzles for input into the simulated annealing solver.

Chapter 3

THEORETICAL FRAMEWORK

Backtracking Algorithm

- **Conclusion:** The backtracking algorithm efficiently searches through the solution space by recursively trying different combinations of numbers and backtracking when a dead-end is reached.
- **Advantages:**
  - Simple and easy to implement.
  - Guarantees finding a solution if one exists.
  - Can handle various constraints and conditions.

Backtracking with Heuristic Approach

- **Conclusion:** Adding heuristics to the backtracking algorithm improves its efficiency by guiding the search towards more promising solutions, potentially reducing the search space.
- **Advantages:**
  - Speeds up the search process by prioritizing more promising paths.
  - Reduces the number of unnecessary iterations.
  - Can lead to faster solutions for large and complex puzzles.

Simulated Annealing Algorithm

- **Conclusion:** Simulated annealing is a stochastic optimization algorithm inspired by the annealing process in metallurgy. It gradually reduces the probability of accepting worse solutions as the search progresses, allowing exploration of a broader solution space.
- **Advantages:**
  - Can escape local optima and explore a wider range of solutions.

- Doesn't get stuck in local maxima/minima.
- Allows for exploration of non-optimal solutions initially, potentially finding better solutions later.

Algorithm	**Can generate a Puzzle**	**Time Complexity**
Backtracking	NO	$O(N!)$
Backtracking with Heuristic Approach	NO	$O(N^{2N})$
Simulated Annealing	YES	$O(N^2 + I \cdot (E + N^2))$

Table 3.1: Time Complexities of Sudoku Solving Algorithms

Where N being the size of Sudoku, I is the no. of iterations and E the complexity of Evaluation Function which depends on several factors like no. of iterations to confirm it's stuck and start reheating it.

3.1 Future Improvments

- **Optimized Neighbor Selection**: Explore different strategies for selecting neighboring states more efficiently. This could involve implementing smarter algorithms for generating neighboring states or refining the swapping mechanism to improve exploration of the solution space.

- **Fine-Tuning Parameters**: Experiment with different values for parameters such as initial temperature, cooling rate, and maximum iterations to find optimal settings for the simulated annealing algorithm. Fine-tuning these parameters can improve convergence speed and solution quality.

- **Enhanced Evaluation Function**: Develop a more sophisticated evaluation function that considers additional factors to better assess the quality of Sudoku solutions. This could include incorporating domain-specific heuristics or leveraging machine learning techniques to learn from example Sudoku puzzles.

- **User Interface Enhancements**: Enhance the user interface to provide more intuitive interaction options and visualizations. This could include features such as real-time puzzle generation, step-by-step solution visualization, and customization options for users.

- **Hybrid Approaches**: Explore hybrid approaches that combine simulated annealing with other optimization techniques, such as genetic algorithms or constraint programming. Integrating multiple algorithms can exploit their complementary strengths and lead to better performance.