SCS-3547 Term Project Report

Primary Objective[1]

- Create an intelligent agent that uses graph/tree search to solve program generated sudoku puzzles

Optional Objectives[2]

- Make the program so that each solution generated is different and keep track of the steps it takes to get to the solution. Use the step tracker as a performance evaluator of how efficient the intelligent agent is.

Techniques and Incorporation

The intelligent agent uses a backtracking search algorithm, also known as a tree search, to explore the possible solutions. The algorithm starts with an empty board and recursively tries to fill in the cells with numbers from 1 to 9. When a cell is filled, the algorithm checks if the resulting board is valid (i.e., no duplicates in rows, columns, or boxes). If the board is invalid, the algorithm backtracks to the previous cell and tries a different number.

To attempt to produce a different solution each time, the solver uses a randomized search approach. When trying to fill in a cell, the algorithm shuffles the numbers from 1 to 9 and tries them in a random order. This would help better analyze the performance of the intelligent agent.

The constraints used for the agent is that there should be only one of each number in every row and column. While this can definitely be improved in the program, one way to do this is by also incorporating constraint satisfaction solving.

Engineering Trade-Offs

I did this project by myself and used research to help find the resources and info I needed to complete the project.

Performance of Intelligent Agent

The time taken to find a solution depends on the difficulty of the puzzle and the number of empty cells. For easier puzzles, the agent may find a solution quickly, while for harder puzzles, it may take longer. However, as of now, the agent may be unable to find the most optimal solution. When the improvement is made in the program so that the intelligent agent is able to generate different solutions for the same problem, this issue can be resolved.

[1] This is the main objective of the project.
[2] These objectives are more like extensions to the program if able to be incorporated.

Project Difficulty
The ideal sudoku only has one solution which must have at least 17 filled squares. With less, multiple solutions for the same sudoku possible. The current version of the project works mostly as is and it is quite difficult to be able to find the different solutions for the puzzle.

Here are ways to possibly improve the performance of the agent:

Heuristics: Improving heuristics, such as most constrained variable or most constraining variable, can help the agent to focus on the most promising areas of the search space.

Pruning: Improving pruning techniques, such as forward checking or constraint propagation, can help to reduce the search space and improve solution times.

Parallelization: Parallelizing the search process can help to speed up the solution time by exploring multiple branches of the search space simultaneously.

Project Uniqueness
This project is definitely unique and interesting as it provides a new perspective on a traditional game. Using an intelligent agent to this application brings a new level of excitement and challenge to the classic puzzle game. By leveraging the capabilities of an intelligent agent, this project offers a fresh and innovative approach to solving Sudoku, allowing users to experience the game in a more dynamic and interactive way.

Resources
- **https://www.technologyreview.com/2012/01/06/188520/mathematicians-solve-minimum-sudoku-problem/**
- **https://medium.com/analytics-vidhya/sudoku-backtracking-algorithm-and-visualization-75adec8e860c**