Sudoku Search Systems

Nash Henry
Eric DiGioacchino

April 24, 2022

Contents

Abstract

Considering the complexity of a number of mathematical problems, none are so popular, and fun, as Sudoku. Of the many ways to solve the a given Sudoku, we encounter only a few of the ways in this project. While some prefer pen and paper methods, a more practical and indirect approach seems to fit best.

1 Introduction

We wanted to create an AI to solve a Sudoku puzzle. Originally, we planned on using simulated annealing but ultimately settled on using Crook's algorithm. Due to the inefficiency and simplicity of Crook's algorithm we also included other means of solving Sudoku.

2 Theory

Sudoku is a common problem you can find in magazines, newspapers, and even game books. It consists of a 9 by 9 grid, divided into 9 sub-blocks, and with some seeded values. The idea of the game is to fill the grid with values ranging from 1 - 9 where a digit is unique for its row, column, and sub-block.

				1				
					6			1
	4							
3				7			6	
	2							
8								
7								

An example Sudoku board with seeded values.

2.1 Brute Force

Given the simple face of Sudoku, a player may assume a brute-force solution is viable. However, the unique search space of a Sudoku board makes it difficult to find solutions.

2.2 Crook's Algorithm

One way of solving a Sudoku is with the pencil-and-paper algorithm described by James Crook. Knowing the seed of the board we look into the possible values of a cell to create a series of candidates. The candidate series allows us to identify patterns in the Sudoku board. These patterns are outside of the scope of this project.
Having the candidate series in-hand we compute for some point on the board, a value. Preferably, the player would prioritize a cell having the least possible candidates. After carefully selecting a value, all candidates of the same digit in the three effected regions (row, column, sub-block) would be canceled out by this value, so they are ignored.
At this point, a novice player would assume the above process is enough to complete every Sudoku. However, this leaves errors on the board. If a player cancels out a value that is shared in a region it may cause an error where all candidates for a cell have been eliminated. Crook solves this by backtracking to a safe state where we know we cannot do any harm and work from there, in some other way.

2.2.1 Algorithm Classification

This method is classified as a Depth-First Search with backtracking which attempts to process every branch of the game tree only working backwards to a safe state after reaching an unstable state. When a stable state is reached, one that meets the rules of the game, the search is discontinued and the path is saved.

2.2.2 Implementation

Because iteration and recursion are synonymous, Crook's algorithm was implemented in both manners, and under several architectures. The Object-Oriented structure has too much overhead to be implemented in recursion, and does not support the iteration format in python. A simple brute-force program using iteration can be made, but as discussed above, cannot account for invalidated cells and thus is an ineffective solution.

2.3 Linear Programming (Linear Optimization)

Linear programming is a method of constrained optimization. On the surface it's like explaining your game to a toddler. It defines a solution under three components: an objective function, decision variable, and some constraints. Python makes this method simple with a module that does the heavy lifting for an LP problem called PuLP. Using some simple entries a problem can described to the LP solver. After being described with operational constraints an objective is formulated.

2.3.1 Algorithm Classification

Linear Programming uses a series of algorithms to solve an optimization, but these sub-processes may not be linear themselves. Because some optimizations are non linear we cannot say for certain that an LP problem will run in linear time, it is also not logarithmic. Given a highly dimensional problem, an LP problem can be solved in polynomial time.
LP problems are not classified as searches despite having a search space. It is classified only as a type of problem in mathematics and computing.

2.3.2 Implementation

In the case of our program we define the features of our game and a game space. Out game space in the context of an LP is a three-dimensional space of the shape

$$(rows \times columns \times values)$$

We shorten the search space by associating a dimension with values and observing the LP-space as a binary vector. This way, when a cell is valid it is 1 and 0 when the cell is invalid. Upon collection the optimized space is read, and where a region is True a number value is associated.

3 Discussions

There are many methods to solving Sudoku puzzles, a few of which we have attempted and covered. Using different searching algorithms and things such as back tracking, attempting to implement these methods just came down to what worked for us and our coding style.The most common way of solving Sudoku puzzles systematically is utilizing backtracking. This is covered withing our project when we utilize crooks algorithm to some extent. Using other methods like ”guess and check” and brute force is also viable but maybe not as fast or as optimized as other solutions.One of the largest limitations of this project for us was familiarity with some of the algorithms we attempted to implement. For things such as simulated annealing for us it was hard for us to implement it into the project due to our knowledge of the algorithm itself. We are familiar with the theory behind it but implementation was just out of reach for us. Even though were unable to implement it within this project we still are playing with it until something gives and we get it to work.

4 Conclusions

Of the ways to solve a Sudoku, iterative with backtracking is the most direct ways to effectively complete the problem. However, our favorite method is Linear Programming for it's novelty. While it's not the easiest method to understand, it is a powerful solution to numerous problems that can be represented in a linear space

References

[1] Crook, J. F. (2009). A Pen-and-Paper Algorithm for Solving Sudoku Puzzles. *American Mathematical Society*, 0–9. https://www.ams.org/notices/200904/rtx090400460p.pdf