Nicholas Montaquila

Project Proposal

For my project I am going to use various constraint satisfaction problem solving methods to write a program that solves a Sudoku puzzle. After doing some research, I’ve found that a popular solution is to write a recursive backtracking algorithm that searches through the board for a blank cell, fills it with a legal value, and then repeats until solved. If it gets to a point where it gets to a blank cell and it has no legal values, it backtracks until it gets to a cell it filled that has another legal value other than the one it filled it with previously, and continues. This method operates similarly to depth-first search, trying to reach a terminal state as quickly as possible. This algorithm can potentially take a long time depending on how “lucky” it gets.

I can use some techniques we learned in class to try to improve the performance. Methods like forward checking and arc consistency can help detect failure earlier and initiate the backtracking sooner. Using a minimum remaining values heuristic can help choose shorter paths and reach a terminal state sooner. MRV works by choosing to fill the blank cell with the least remaining legal values to choose from, which means there are less branches of the tree to explore. Using a least constraining value heuristic can help maximize the potential of each choice. LCV chooses to fill each cell with the value that restricts the domain of its neighbors the least. Using these strategies will improve the performance of the backtracking algorithm.

Applying these techniques will improve the performance of the algorithm. At the end of the project, I plan to be able to read in a Sudoku board, solve it, and print out the solution.

Resources:

<https://www.geeksforgeeks.org/backtracking-set-7-suduku/>

<http://www.math.cornell.edu/~mec/Summer2009/meerkamp/Site/Solving_any_Sudoku_II.html>

<http://norvig.com/sudoku.html>