Rupesh Gudipudi, Varun Srivastava, Ayush Jha

27 September 2019

COMP 560 Assignment 1 – KenKen Puzzle Solver

Simple Backtracking Algorithm

The simple backtracking algorithm begins by starting at the first cell of the grid with coordinates (0,0). It then traverses through the grid by going row by row, one cell at a time. As it traverses, it puts a value into the cell to see if it works. This value must be a unique value that is not already repeated in the row/column and must satisfy the operation block. If the value works, it will go to the next cell. If it doesn't it will backtrack to the previous cell and put in other values that can work and the process is repeated.

Best Backtracking Algorithm

The best backtracking algorithm is similar to the simple backtracking algorithm but it traverses through the grid by going through each operation block one at a time instead of going row by row. It puts in values within the block one cell at a time and checks to see if the row, column, and operation constraints aren't violated. If these violations occur, there is a backtrack within the block to try other values and if there are no more values that can work, there is a backtrack to the previous block itself. This is considered our best backtracking algorithm because it is looking at a group of nodes at once and this can speed up the process and limit the number of iterations it takes to solve the puzzle.

Local Search Algorithm

The local search algorithm initially forms a grid full of values that are unique to each row and column. Then there is a swap function called that swaps a certain row/column to make sure the operation constraint is fulfilled. Swapping these rows and columns allows values to be put into the operation blocks while not violating the row and column constraint. The utility function for the local search was a violation based function. The algorithm would calculate how many block constraint violations would be made for the current state and the swap state, and if the swap state had less than the current, the swap would occur. The nodes would change randomly in each iteration.

Contributions

Ayush: Implemented the best backtracking search algorithm and local search algorithm

Rupesh: Worked on simple backtracking search

Varun: Worked on simple backtracking search

Together: Discussed the logic and implementation for 3 types of searches