Francesca Cambareri & Shyma Alshari

Professor Lili Su

EECE 2560

March 30th, 2024

Project 4 – Sudoku Puzzle Report

This purpose of this project was to use the basic code given and implement a recursion algorithm in order for the program to take in unsolved sudoku puzzles and be able to output the puzzle solved on a board, as well as show the boards conflicts, as well as the number of recursive calls to solve the board. The method to solve the board is within the 'solve' function and uses recursive attempts to fill each cell of the Sudoku board and backtracks when a conflict arises. Using this backtracking algorithm, the program is able to successfully find what number goes into each spot, as it systematically fills the cells of the sudoku board with possible values, and then checks those values for conflicts within the row, column and square every step of the way. When the program does discover a conflict, it backtracks to the previous decision point and tries to find an alternative value that will work better and goes through the same conflict checking process. Our program also includes functions to the board class that allow it to update the conflict vectors, print the board as well as the conflict vectors to the screen, add values to the cell for testing, and then go through and update the conflict vectors once again based off of this addition, as well as clear a cell if a value is found to have a conflict and update the conflict vectors based on that, and finally checks to see if the board has been solved or not, and lets the user know. Below we have included screenshots of some of the results from running the code with the provided 'sudoku.txt' file to show how all of these parts work together to produce a final product.

Figure 1 and 2: Screenshots from part a outputs to show how the conflict vectors are used to help recursively solve the puzzle.

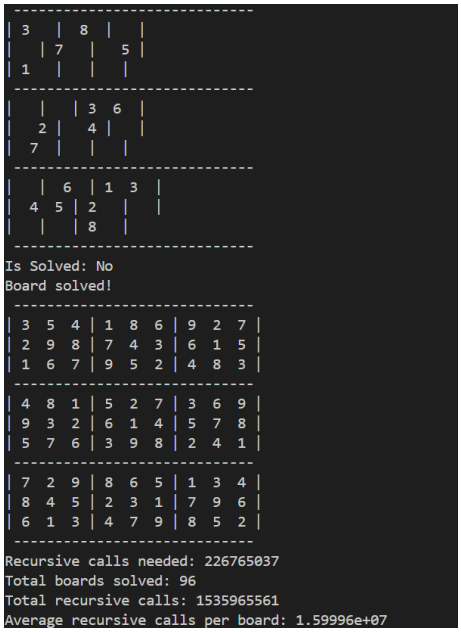

Figure 3: Screenshot from Part B results showing the board before and after it is solved, as well as the final printout at the end of the boards that shows the total number of recursive calls needed to solve the 96 boards, as well as the average number of calls needed to solve the board.