Amit Nijjar

A11489111

CSE 150

Assignment 3 Report

Question 1:

For question 1, I implemented the is_complete method. It returns true if all variables in the csp have been assigned. The algorithm goes through and checks each variable with the is_assigned() function. If any are not assigned, it returns false.

Question 2:

For question 2, I implemented the is_consistant method that returns true when the variable assignment to value is consistent. The algorithm loops through all the variables to find another variable with the constraint. If they don't match the constraint, return false, otherwise return true.

Question 3:

For question 3, I implemented the backtracking algorithm. I used a recursive algorithm that calls itself when it doesn't find a successful solution. I included the is_complete and is_consistant algorithms from parts 1 and 2 in order to make this algorithm work. If the conditions are met, return true, otherwise do the algorithm again on an unassigned variable.

Question 4:

For question 4, I implemented the ac3 algorithm. I used a que to keep track of all the arcs in the problem. If the arc needed to be revised, I called the revise function that returns true if the domain was revised. Revised also checks consistency.

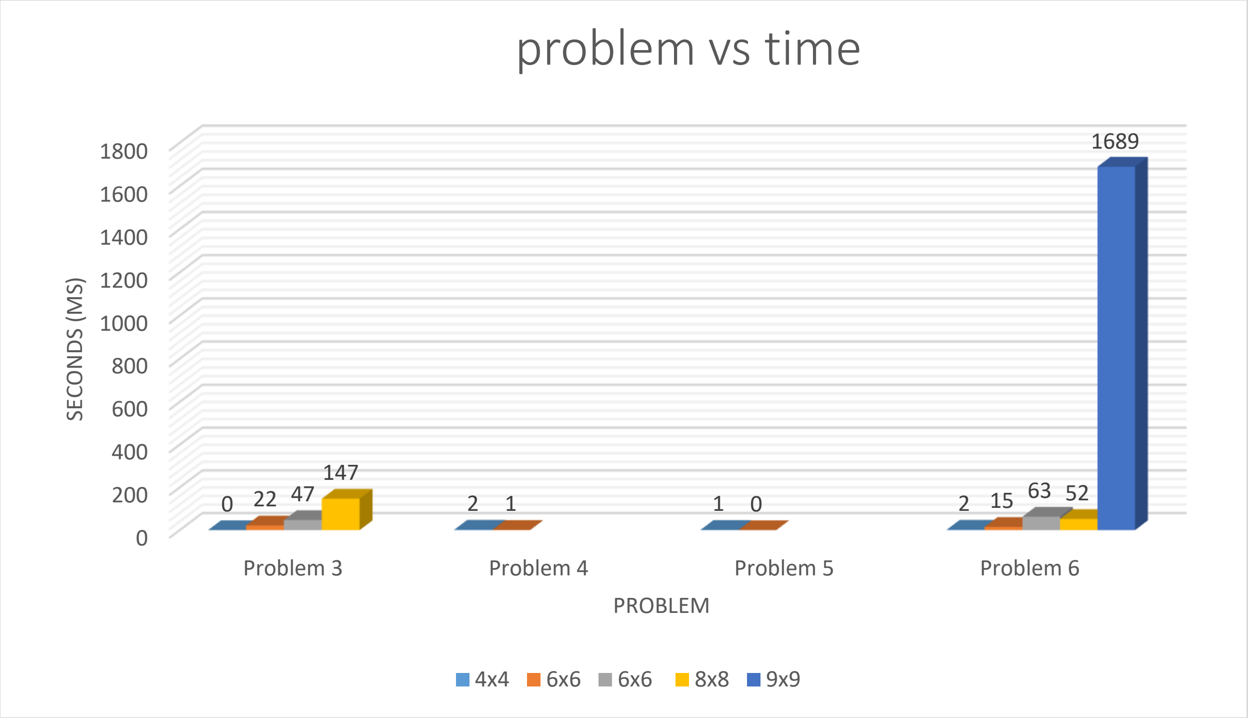
Question 5:

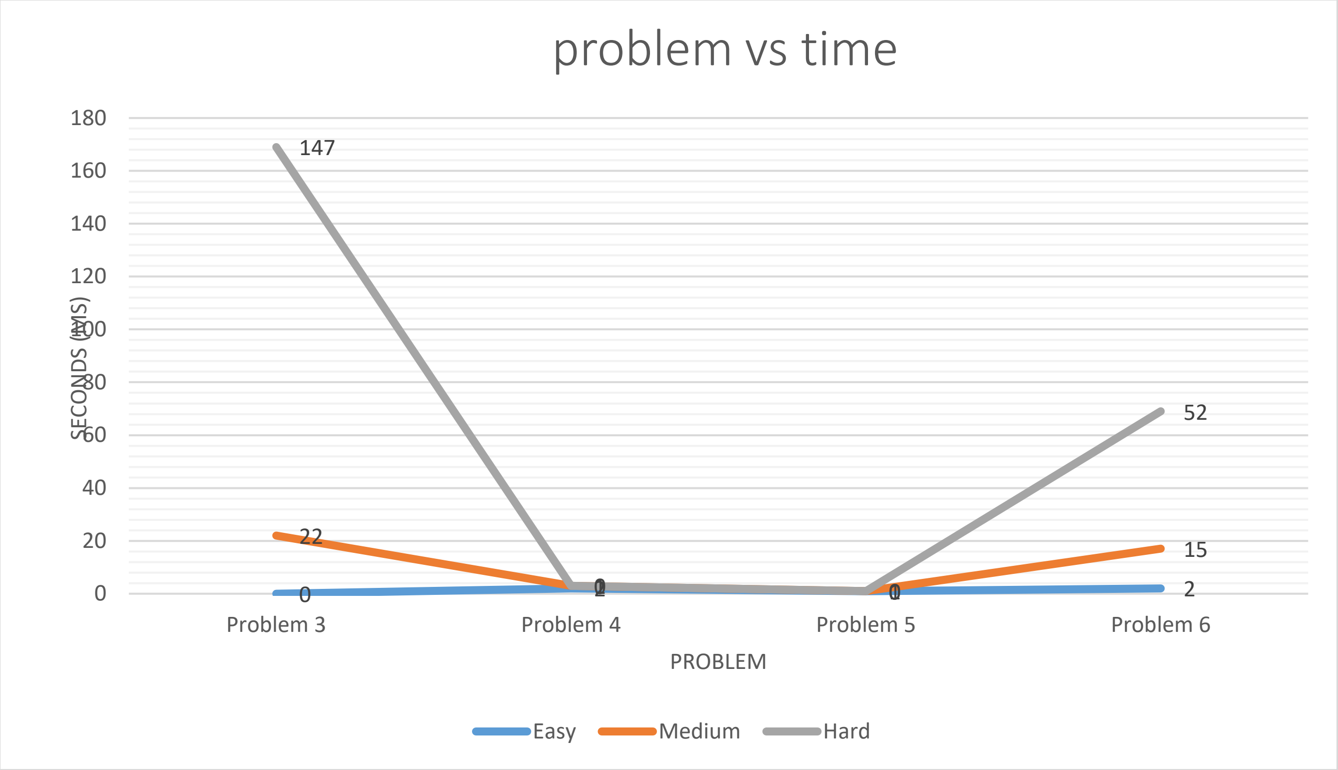
For question 5, I implemented select_unassigned_variable and order_domain_values. The ordering is based on MRV. Pull variables from a queue with minimum remaining priority. The value ordering is LCV. Loop through all variables and check the number of conflicts that would cause.

Question 6:

For question 6, I implemented a faster backtracking algorithm by augmenting the algorithm with MAC inference and variable/value order heuristics. I changed the variable and value heuristic in ac3 to optimize the backtracking algorithm.

Graphs:





As you can see in the graphs, increasing the size of the board increases the runtime of the program. There are more paths for the algorithm to check and more constraints to check for. The ratio of size to runtime is not linear. After a certain size, the increased constraints helped to eliminate unnecessary paths, thus allowing the algorithm to be solved faster.

All of the heuristics sped up the basic solver. The inference heuristic sped up the program by making inferences about criteria allowing for better paths to be chosen. The variable and value ordering heuristic made the program run faster by leading to better variables to be assigned. Using both together helped speed up the algorithm from over 2 minutes to under 5 seconds.

I worked on this assignment alone.