Andrew Montgomery

3/13/20

Program 4 Analysis

**Can you reliably get good solutions?**

While testing I was able to get good solutions, often solving the puzzle. The success of the program was often directly related to the difficulty of the puzzle. A simple puzzle (just a few non-set values) would generally be solved in less than 10 generations and with minimal population sizes.

**How many generations does it take?**

As mention previously it depends on the difficulty of the puzzle. Unsolved sudoku puzzles I found online would generally take between 100 and 200 generations to solve. I did not test beyond 200 generations due to the time it would take the program to run.

**How does this depend on how hard the puzzle is?**

As the difficulty of the puzzle increased; the population size, number of generations, and time required to solve the puzzle increased greatly. I did find that I would start to see relatively low fitness values after just 10 to 20 generations (around .1, where fitness was calculated between 0 and 1), and then from there the decrease in the fitness value for the best puzzle in each interval of generations would be much lower.

**Does adding additional genetic operators improve performance?**

I did not implement any additional genetic operators to test an improvement in performance due to time restrictions. However, my hypothesis is that additional genetic operators would certainly improve the performance. I think that additional genetic operators would increase the run time of the program from generation to generation but adding additional operators would in general help to find solutions to puzzles much faster, cutting down the overall time it takes to find solutions. I believe that if I had the time to fix my fitness function, I would be able to greatly increase the performance of my program. The number of times the fitness needs to be calculated . If I were to make use of hash tables for implementing my fitness function, it is my understanding that the performance of my program would increase greatly.