Lab 5 - Genetic Algorithm

Aidan Schneider

CS 2400 031

04/20/23

1. Overall Hill climbing is much less costly than the Genetic algorithm. Just doing an analysis of the code, you can see that Hill climbing only uses a single while loop for each restart in the main hill climbing code. The only other resource intensive thing that hill climbing does is search through its successors for the best state, however that is pretty much where it stops for hill climbing. For the genetic algorithm just in the main code for it there are two for loops that loop through the entire population. On top of that the fitness proportionate selection function is a fairly costly function needing to run through the entire population multiple times for each parent that is chosen.
2. Lowering the population size will cause the accuracy of the algorithm to go down. This is because with fewer individuals the algorithm is not able to explore as much of the state space at once, so getting stuck in local optima would be easier. Also there is a higher chance to get stuck in local optima due to there being less bits overall which means less chance of bit mutation happening which allows the population to get out of local optima. In testing the population of 100 got 99/100 successes on Deceptive Maze 1 and the population of 10 only got 20/100 successes on the same maze.
3. The Genetic Algorithm (GA) gets such higher scores than Simulated Annealing (SA) and Hill Climbing (HC) because of how much of the state space GA is able to explore compared to HC and SA. Due to GA having a population of individuals all exploring different paths in the same restart GA can explore the entire map very quickly whereas SA and HC in one restart only get to go down a single path which limits how much of the state space is observed by them. Since GA sees more of the state space, GA finds the goal more reliably and then simply builds a path to it.