Noah Fabrick
2/22/24

Homework 2 - Modified Brute Force and Genetic Algorithms

The below solution uses my submitted *GA.py , genetic_alg.py , and MBFA_GA_Fabrick.py* files to answer the homework's questions.

Problem 1

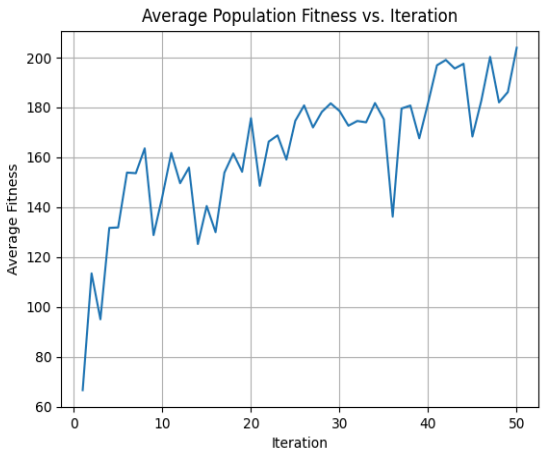
Example terminal readout from my modified brute force algorithm file:

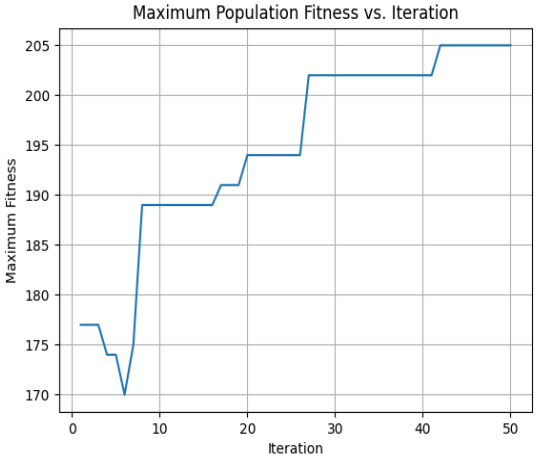
Modified Brute Force Algorithm Results after 50 attempts (weight [kg],value [$]):
1st Best Weight: 45 1st Best Value: 205
1st Best Inventory: [(3, 6), (3, 17), (4, 2), (7, 24), (7, 28), (1, 3), (5, 27), (7, 23), (4, 15), (1, 20), (2, 21), (1, 19)]
2nd Best Weight: 44 2nd Best Value: 197
2nd Best Inventory: [(3, 6), (6, 8), (5, 27), (1, 19), (3, 17), (4, 10), (1, 3), (7, 23), (7, 28), (1, 20), (4, 15), (2, 21)]
3rd Best Weight: 45 3rd Best Value: 185
3rd Best Inventory: [(5, 27), (3, 6), (4, 10), (6, 8), (9, 22), (7, 24), (7, 28), (1, 20), (1, 19), (2, 21)]

This spread exemplifies how well my MBFA algorithm works on different iterations, averaging a best solution value in the 190's and sometimes obtaining a best solution in the 200's and 180's.

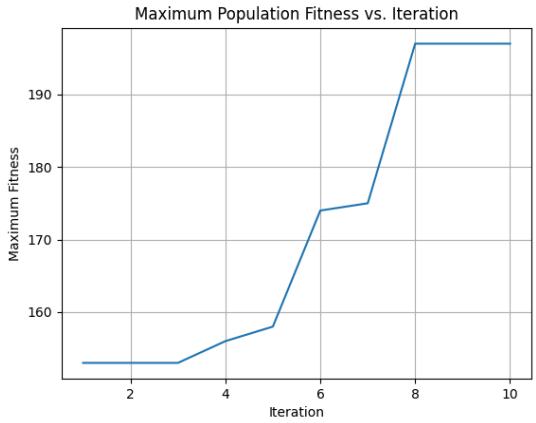
Problem 2

One of my best genetic algorithm runs with 50 iterations produced a best solution value of 205 and the following plots.

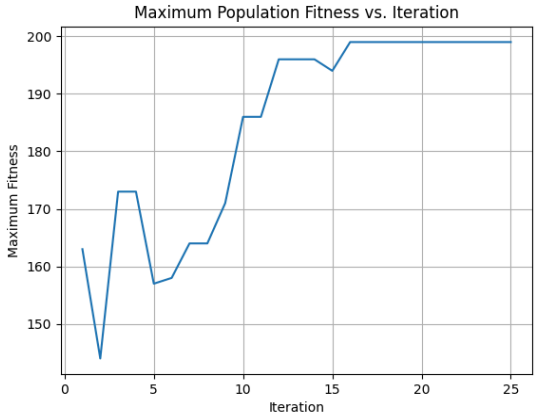




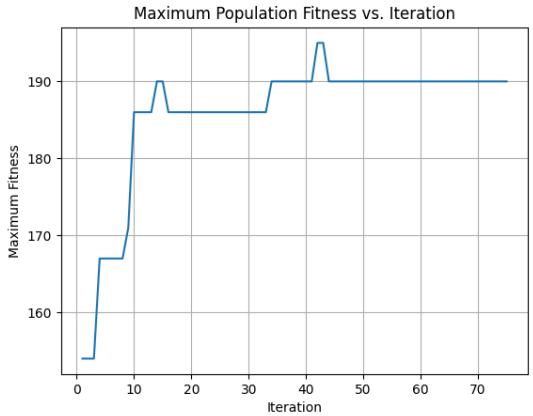
A notable trend observed in the maximum fitness vs. iteration graph is how the best fitness solution is lost around iteration 7, meaning my selection function or some other error is occurring in my genetic algorithm. Depending on the run and number of iterations, this error can cause the genetic algorithm to perform quite worse as seen in the following implementations.



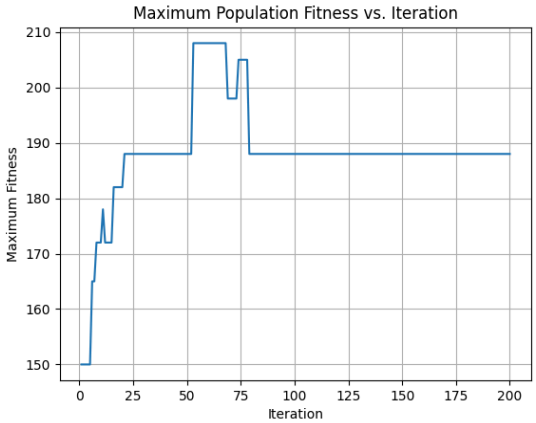
10 iterations, 197 best solution



25 iterations, 199 best solution



75 iterations, 190 best solution



200 iterations, 188 best solution

As shown, the number of iterations does not necessarily mean a better solution for my algorithm. One would expect that a proper genetic algorithm will converge to a better solution with more iterations, so something is wrong with my algorithm. The selection process should keep the current best solutions and then add crossovered and mutated options to create a new population to be reiterated upon, which my algorithm is only sometimes doing properly. The above graph of 200 iterations shows this error where an excellent solution of 207 is discarded for some reason and the algorithm converges onto 188 as the best solution.

Problem 3

In the ideal cases presented earlier, both my modified brute force and genetic algorithms can produce an excellent solution with a value greater than 200. Due to a selection error in my genetic algorithm, my modified brute force algorithm will more consistently produce better solutions given the same number of iterations. Given more iterations, my MBFA will perform even better and converge to multiple better solutions due to the nature of brute force, and will consistently outperform my genetic algorithm. Although this is likely due to coding errors in my

genetic algorithm, the KNAPSACK problem of 20 options might just be a better problem for a MBFA to solve as the number of combinations is relatively low with constraints and brute force will eventually find ideal solutions given enough iterations. For larger options pools with more complex fitness functions and stricter constraints, a proper genetic algorithm would absolutely outperform a brute force algorithm.