Cmsc 191 Exercise 4: Tournament Selection implemented on Genetic Algorithm in Solving Functions

Gimel David F. Velasco
Department of Mathematics and Computer Science
University of the Philippines Baguio
gfvelasco@up.edu.ph

ABSTRACT
This paper demonstrates an implementation of the Genetic Algorithm using a Hybrid Selection Process consisting the Tournament Selection and Roulette-Wheel Selection in solving the root/s for functions.

1. INTRODUCTION
The Genetic Algorithm is an algorithm made by Holland on 1975. The Algorithm is inspired by the concept of Genetics and Natural Selection. It approaches a particular problem by looking for the fittest individual and considers it as the solution to the problem. It approaches a particular problem by looking for the fittest individual and considers it as the solution to the problem. The kind of approach made in the tests are inspired on how the algorithm was implemented in the article made by Hermawanto [1] on solving a 4-tuple function.

2. METHODOLOGY
The program will seek the roots of the following equations:
Problem 1:

$$f(x) = x - cos(x) \tag{1}$$

Problem 2:

$$f(x) = e^{-x}(x - 2) \tag{2}$$

Problem 3:

$$f(x) = x^2 - x - 12 \tag{3}$$

in the interval [-5,5]. The absolute value of the functions above is the set fitness/objective function of the Genetic Algorithm.

The program followed the flowchart of the Genetic Algorithm demonstrated by Hermawanto. With some, changes and modifications made. The different processes and how they are implemented in the code is further explained in the succeeding subsections.

2.1 Data Representation
Each Chromosome represents the possible value of the root for the function. There are 5 alleles found in each chromosome, the summation of which would represent the possible value of the root for the function. Each allele is a real number that is in the interval [-1,1]. So, we are certain that for whatever summation of the alleles in each chromosome, it is still in the interval [-5,5]. This data representation is used so that crossovers would be implemented in order for the algorithm to get the closest root possible and get it quickly.

2.2 Step 1: Initialization
In the initialization step, the initial chromosome population or the first generation of chromosomes was initiated by randomizing the value of each allelle of each chromosome for the whole population. Each chromosome consists of 5 real numbers in [-1,1] of which their summation is the possible root for a given problem/function. This is done repeatedly until 100 chromosomes would fill the population. For the terminating criterion for the program, a maximum of 100000 generations is used in the algorithm and if the fittest chromosome/s would have a fitness of less than 0.00005, the program will end. A crossover rate of 0.25 and a mutation rate of 0.1 is used in the program.

2.3 Step 2: Pre-Evaluation
This step will compute for the fitness of each chromosome per given generation. It will plug each chromosome in the fitness function/objective function which is defined as

$$obj(x) = abs(f(x)) \tag{4}$$

The values of each chromosomes' fitness is then saved in an array and then will be used in the selection process.

2.4 Step 3: Selection
A Hybrid Selection Process is implemented in the algorithm. The Tournament Selection and Roulette-Wheel Selection process is implemented in the algorithm since this selection process performed fastest in the previous genetic algorithm exercise. The Tournament Selection process will take place first. Two random chromosomes, red and blue, will be selected for competition and the chromosome that is more fit will overwrite its competition. The number of competitions is done until there are 25 winners which is 25

2.5 Step 4: Crossover

The selected chromosomes will be the parents of the next generation children chromosomes. The algorithm implements a single point crossover at a random position in each of the chromosome. Each selected parent chromosome is crossover with their neighbor parent chromosome. This process is done until all the parents are crossed over with each other.

2.6 Step 5: Mutation

In the mutation process, instead of implementing a per-allele-mutation as how Hermawanto implemented, a per-chromosome-mutation is used. The number of mutations is based on the mutation rate which is 10

2.7 Step 6: Post-Evaluation

This step will check if the fittest chromosome meets one of the terminating criterion. That is if the fittest chromosome/s found in the population meets a fitness that is less than 0.00005, the algorithm will now jump to Step 8. Or else, Proceed to Step 7.

2.8 Step 7: Repeat Steps 2 to 5

The steps 2 through 7 will repeatedly be performed until the maximum number of generations of 100000 is reached. After the maximum number of generations is reached, the program will then proceed to Step 8.

2.9 Step 8: Solution

This step will take place if one of the terminating criterion is satisfied. This step will now display the Final Generation Chromosomes, the Root of the problem/function and how much time it took to find the solution.

3. RESULTS AND DISCUSSION

For all the problems/functions, the algorithm is able to find the roots of the functions accurate to 4 decimal places. The values that the algorithm has found for each problem is the following
Problem 1:

$$x = 0.7390747925598138 \tag{5}$$

Problem 2:

$$x = 2.0000846801706489 \tag{6}$$

Problem 3:

$$x = -2.9999999456403312 \tag{7}$$

For Problems/functions 1 and 2, the algorithm easily finds the root of the function in 1 second to 10 seconds. But for Problem 3, the algorithm takes from 1 second to more than 200 seconds. All the more, the algorithm is capable to find the root of the functions accurate to 4 decimal places. Though at some tests the roots that the algorithm finds yields 3 to 7 decimal place accuracy. Also, for problem 3, the root that the algorithm finds -3 more frequently than finding 4 as the root.

4. CONCLUSION

The Hybrid Selection Process of Tournament and Roulette-Wheel is capable of finding the roots of the functions. All the more, the roots that the algorithm finds is capable of being accurate up to 7 decimal places. Perhaps if the tolerance is reduced, the algorithm would yield a much more accurate answer but this will cause the algorithm to run longer.

5. REFERENCES

[1] Hermawanto, D.,"Genetic Algorithm for Solving Simple Mathematical Equality Problem", Indonesian Institute of Sciences, n.d.