Faculty of Engineering and Technology
Department of Electrical Engineering

Artificial Intelligence

ENCS3340

PROJECT "1"

GRADUATION PROJECTS DISTRIBUTION -USING GENETICS ALGORITHM-

Prepared By:
Yasmin Abusharbak – 1182523
Munther Anati – 1182028

Instructor:
Dr. Aziz Qaroush

Section "2"

December 12th, 2021

Abstract

This project aims to perform a distribution of a set of graduation projects over a number of student groups, using Genetics Algorithm. Given a data of 38 different projects, and 36 different groups of students with their preferred three choices, the goal is to assign each group to a project -which can be assigned to one group only-, and the assignment must be as close as possible to the first preferred choices of the groups.

Contents

1 Theory

1.1 Searching Techniques

Searching Algorithms are designed to check for an element or retrieve an element from any data structure where it is stored [1]. A population-based algorithm is an **algorithm** that maintains an entire set of candidate solutions, each solution corresponding to a unique point in the search space of the problem [2]. Local search is a **heuristic method for solving computationally hard optimization problems**. Local search can be used on problems that can be formulated as finding a solution maximizing a criterion among a number of candidate solutions [3].

1.2 Genetics Algorithm

Genetic Algorithms (GAs) are adaptive heuristic search algorithms that belong to the larger part of evolutionary algorithms. Genetic algorithms are based on the ideas of natural selection and genetics. These are intelligent exploitation of random search provided with historical data to direct the search into the region of better performance in solution space. They are commonly used to generate high-quality solutions for optimization problems and search problems. Genetic algorithms simulate the process of natural selection which means those species who can adapt to changes in their environment are able to survive and reproduce and go to next generation. In simple words, they simulate “survival of the fittest” among individual of consecutive generation for solving a problem. Each generation consist of a population of individuals and each individual represents a point in search space and possible solution. Each individual is represented as a string of character/integer/float/bits. This string is analogous to the Chromosome [4].

Main concepts for Genetic Algorithm:

1.2.1 Population

The population of individuals are maintained within search space. Each individual represents a solution in search space for given problem. Each individual is coded as a finite length vector (analogous to chromosome) of components. These variable components are analogous to Genes. Thus, a chromosome (individual) is composed of several genes (variable components).

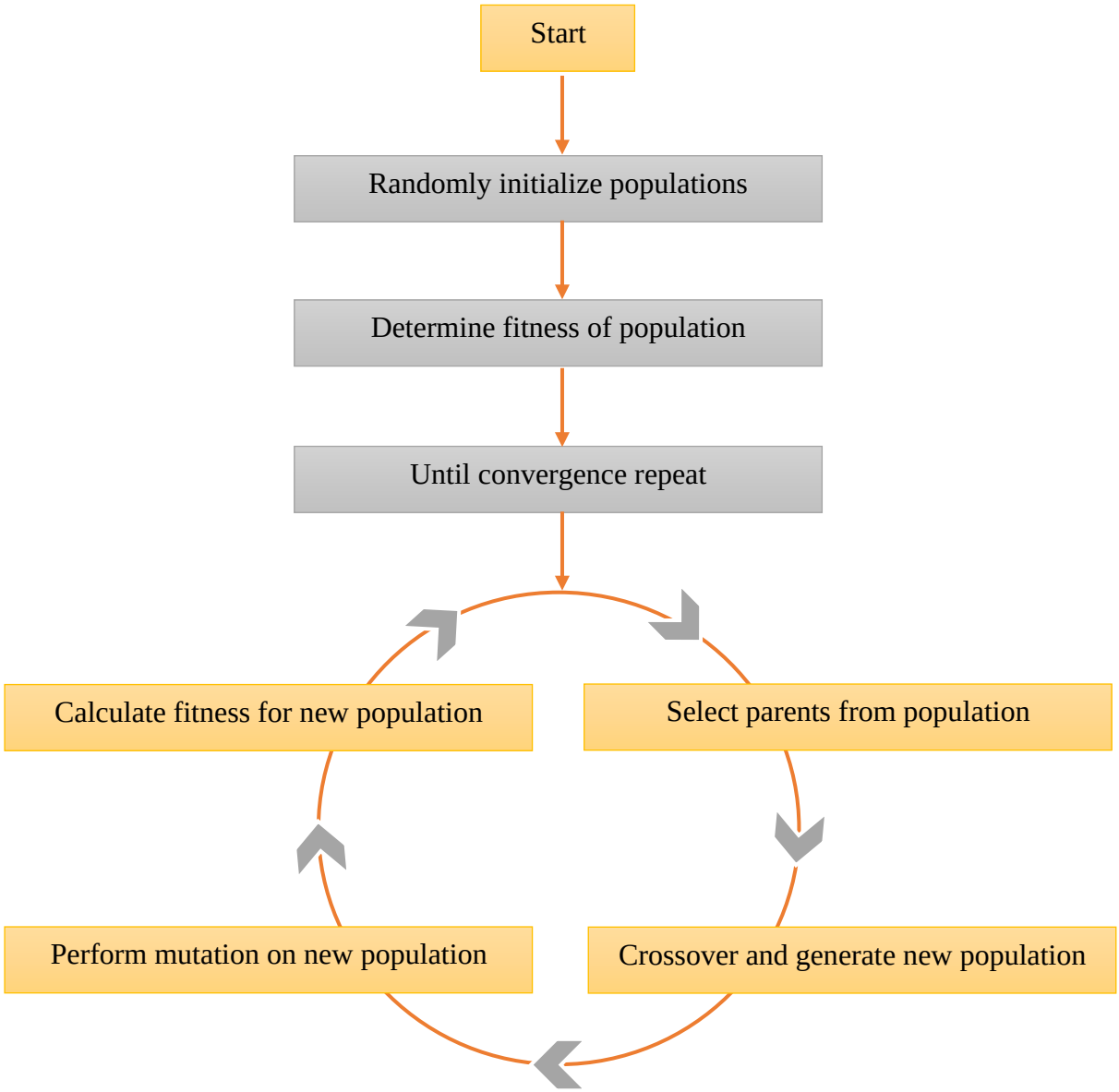
1.2.2 Fitness

A Fitness Score is given to each individual which **shows the ability of an individual to “compete”**. The individual having optimal fitness score (or near optimal) are sought.

1.2.3 GA Operators

Once the initial generation is created, the algorithm evolves the generation using following operators:

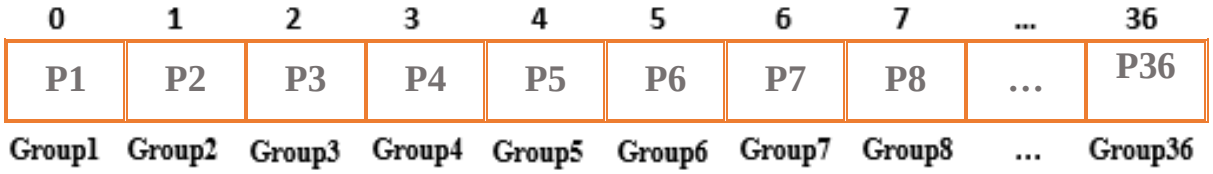
- **Selection Operator:** The idea is to give preference to the individuals with good fitness scores and allow them to pass their genes to successive generations.
- **Crossover Operator:** This represents mating between individuals. Two individuals are selected using selection operator and crossover sites are chosen randomly. Then the genes at these crossover sites are exchanged thus creating a completely new individual (offspring).
- **Mutation Operator:** The key idea is to insert random genes in offspring to maintain the diversity in the population to avoid premature convergence.



2 Procedure

2.1 Problem Formulation

- Problem: Graduation Projects Distribution
- Objectives: Assigning each students group a project, where a project can be assigned by one group only. The assignment should be as close as possible to students most preferred choice.
- Given Data:
  1- Graduation Projects: 38 different topics to be distributed
  2- Student Groups: 36 different groups of students to distribute projects over
  3- Preferred Choices: each group has 3 choices of projects numbers, first, second and third choice, where the first choice is the most preferred, then the second, and finally the third.
- Searching Method: Genetics Algorithm
- Solution: a solution to look for is a one that satisfies the objectives mentioned above, or a solution that maximizes the number if matches with the students' preferred choices.
  The solution is a chromosome of length that is equal to the number of student groups (in this case 36). The chromosome was represented as an array-list of size 36. The values of the genes carried by that chromosome represent the number of the projects assigned for the groups. The number of the group is indicated by the index or the position that the gene value is assigned to.



  * Such that:

  Pi ∈ {1, 38}; representing projects numbers that MUST be unique
  Groupi+1 is related to the index i, and represents the group number

- Thus, the search space or the population generated initially, is a set of chromosomes, each of length 36.

2.2 Genetics Algorithm Operators

As described previously, the genetics searching algorithm goes through various stages to optimize the solution. Starting from the randomly generated population, the following operators were specified and used:

- **Fitness Calculation**: the fitness score of a chromosome increases as the number of matches between the gene values of that chromosome/ solution and the preferred choices of the groups, increases.
  Assuming that the number of matches with the first-choice column among the chromosome is f1, the number of matches with the second-choice column is f2, and the number of matches with the third-choice column is f3. The total fitness of a chromosome is defined as:
  $$Fitness = f1 + f2 + f3$$
  As known, the chromosome with a larger fitness value, has a higher chance to compete with other solutions.
- **Selection Method**: the very first parents to reproduce are selected out of the randomly generated population depending on their fitness values. The best chromosome (the solution with the maximum fitness value), and the second best one, are chosen from the population and used to reproduce.
- **Crossover Method**: the followed crossover technique is a subsequence crossover method. At each iteration, crossover is applied on the parents selected to reproduce. It is performed by randomly generating an index, say index i, the gene values starting from the index i till the end of the chromosome are swapped between the two parents, and the rest of gene values (at indices less than the generated index i) are kept the same. However, in this problem, no duplicates are allowed, in other words, same project can not be assigned by more than one group, hence, simple mapping is performed to avoid repetition of project numbers among the individual chromosome.
- **Mutation Technique**: to get better generations and increase diversity, some gene values are randomly changed. A variable was defined as the mutation chance, mutation is performed whenever a random value generated at each iteration is less than the mutation chance. Mutation is performed over an individual chromosome by simply swapping the values of two gene at two randomly generated different indices.

3 Sensitivity Analysis

Since the main goal of using Genetics Algorithm is to find a solution that maximizes the number of matches, the previously mentioned operators were specified after several testing stages. Other than tracing each part of the implemented program step by step, several variables were declared and tested for various values.

Mutation chance, number of iterations, population size, total fitness, and number of matches with the first-choice column, were defined as variables and initialized by different values to finally determine the values that enhance the performance of the algorithm. Final values were used as recommendations (editable) in the program interface for the user.

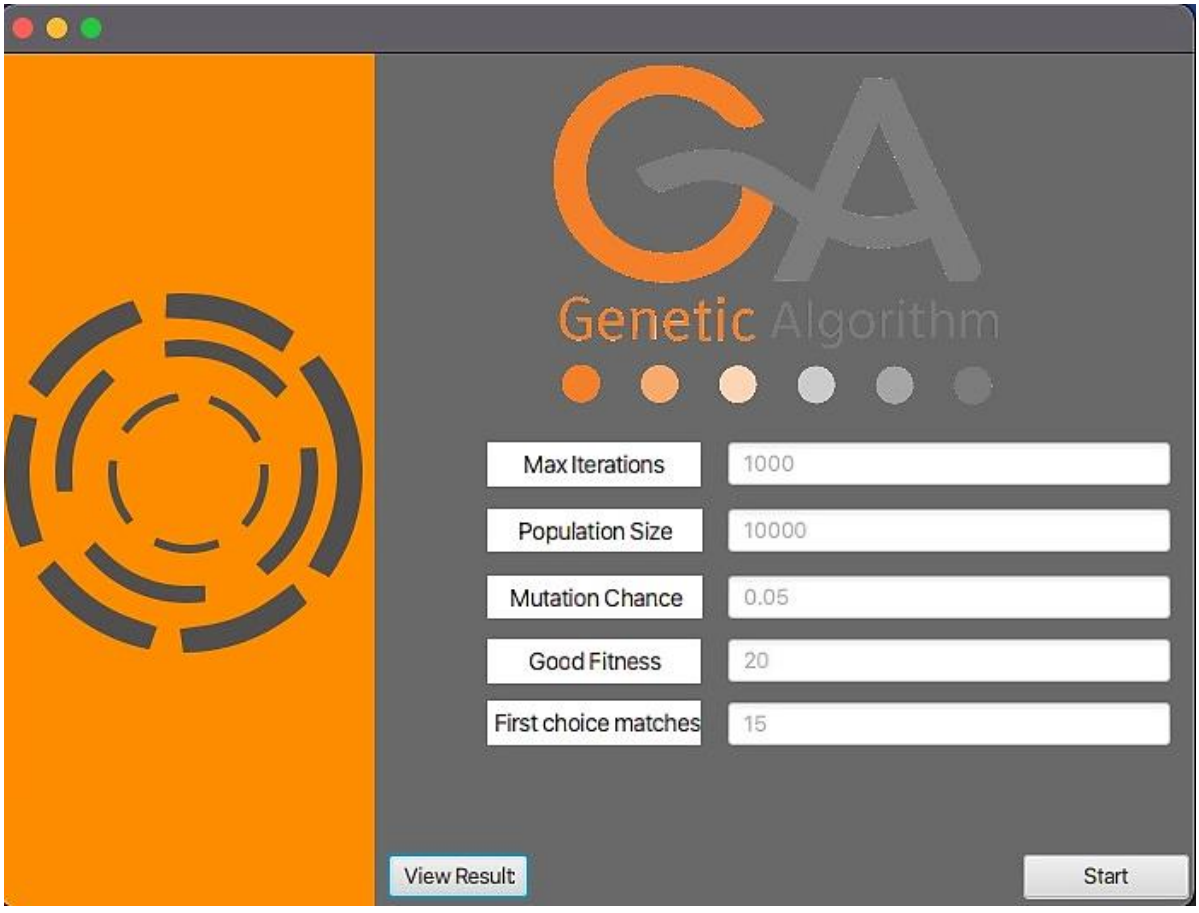


Figure 1 : First User Interface

Other than these variables, several crossover methods were used and tested, starting from the quite complex partially mapped crossover (PMX), to the quite simple two values swap, the chosen method is the subsequence crossover method, with a simple mapping to avoid duplicates. Moreover, for a better performance, in the crossover operation, a simple detection was added, for the two parents, any gene value that is already assigned to the right group, it was skipped and not swapped or mapped.

PMX and the simple two values swap methods were not used due to the low fitness values of the produced generations.

For the mutation technique, various mutation rates were tested for their efficiency, the programmed was forced for a higher mutation rate at the beginning, however, this affected the performance exactly the opposite, a high rate of diversity inserted into the generations has lowered fitness values. So, it was kept random, and the mutation occurs only when a randomly generated value is less than the mutation chance variable, that is recommended to be 0.05.

Fitness score was the most challenging part of the problem. It was tested to be the number of the matches with the first-choice column only, but due to the high similarity rate and repetition in the data given, this was not quite efficient. It was finally set to be the total number of matches with the three columns of choices.

Selection method was also challenging, it was tested to be depending on the number of matches with the first-choice column only, which forces the algorithm to be quite greedy and as expected, low overall fitness values were produced.
Also, it was tested to have a two stages selection, where a set of solutions with the maximum number of matches with the first-choice column were selected, and then the two chromosomes with the highest total fitness values were selected to reproduce were used, stages were also switched, both cases were tested but fitness values of the produced generations were not convincing. Finally, the chosen method depends totally on the overall fitness value (total number of matches with the three columns), as it produces the generations with the best fitness values.

4 Conclusion

Genetics Algorithm is good to use whenever the search space is big and there are states that are not necessary to keep, as in this algorithm, only good states are kept and used to produce new generations, and the process is repeated.

However, Genetics Algorithm is affected by several operators, and this has made the implementation more challenging. Throughout the implementation of this program, different problems were faced and the reason of the low performance was sometimes hard to detect. To get the program to this stage of performance, all the previously mentioned operators were tested and changed many times, by keeping all factors constant and changing one factor at a time.

In conclusion, the algorithm has a quite high rate of randomness, and this has varied the produced result solution even at each runtime.

5 References

[1]: https://www.geeksforgeeks.org/searching-algorithms/

[2]: https://www.igi-global.com/dictionary/nature-inspired-methods-multi-objective/22995#:~:text=What%20is%20Population%20Based%20Algorithm%3F%20Definition%20of%20Population,point%20in%20the%20search%20space%20of%20the%20problem.

[3]: https://en.wikipedia.org/wiki/Local_search_(optimization)#:~:text=In%20computer%20science%2C%20local%20search,a%20number%20of%20candidate%20solutions.

[4]: https://www.geeksforgeeks.org/genetic-algorithms/