Genetic Algorithm

Computational And Cognitive Neuroscience

Research Question:To what extent can a genetic algorithm simulate the evolution of a set of individuals from a specific population until a targeted individual is reached?

18/12/2020
Rosamelia Carioni Porras ***(Student ID: 6247275)***
Bianca Caissotti di Chiusano **(Student ID: i*6245461*)**

Table of Contents

1.0 Introduction

The modern understanding of what is now known as “Evolution”, dates back to Charles Darwin’s book: “On the Origin of Species” (1859). His theory of evolution by natural selection revolutionised the concept of the origin of species by means of modification (Nielsen, 2012). In his book, Darwin writes “This preservation of favourable variations and the rejection of injurious variations, I call Natural selection” (Darwin, 1864), discussing that individuals born with any kind of advantage would be more likely to survive, “this complex battle of life” and reproduce. It is important to keep in mind that these variations may occur in the course of thousands of generations (Darwin, 1864).

The computational representation of this theory leads to the “Genetic Algorithm” (GA), a simulation of the process of natural selection that results in the creation of new generations, which both inherit and differ from the previous. It is essentially a search technique in which individuals of a population (represented as symbols) simulate natural selection mechanisms, such as crossing-over and mutation (Hosh, 2017).

The origin of genetic algorithms dates back to the late 1960s, originally named as “reproductive plans” by John Holland, a researcher from the University of Michigan (Maad, 2016). These have numerous advantages, firstly they can deal with large numbers of variables and data, (which can also be randomly generated), and through searching techniques, they will work towards finding the optimal solution (Maad, 2016). Moreover GAs often make use of a fitness function. This function gives a value to each individual of a population and “does not require the calculation of derivatives” (Ballard). Even though this is an advantage for the GA, as it helps determine which individuals are the fittest, it can also be seen as a limitation. This is because, overall, a Genetic algorithm is much less complex than biology itself, just considering, for example, the process that goes into decoding DNA. Hence, it is fair to argue that, as Ballard notes in the book “Introduction to Natural Computation”: “Genetic algorithms are huge abstractions that capture just the overall features of the reproductive process” (Ballard).

1.1 Research Question

This report will investigate, analyse and test a version of a genetic algorithm, with the aim of tackling the following research question:

"To what extent can a genetic algorithm simulate the evolution of a set of individuals from a specific population until a targeted individual is reached?".

1.1.1 Sub-questions

To fully tackle the research question of this report, the following sub-questions are useful to keep in mind:

a) What is the influence of a larger and/or smaller population? Is there a notable difference in the number of generations needed to find a solution?
b) What is the influence of a larger and/or smaller mutation rate? Is there a notable difference in the number of generations needed to find a solution?
c) Does the Genetic algorithm still work without mutation?
d) Leaving out the crossover operator. Is there a notable difference in the number of generations needed to find a solution?

2.0 Method

2.1 Natural Selection

As stated previously, the Standard Genetic Algorithm (GA) simulates natural selection mechanisms. Using John Koza's definition of GA, the approach **used** in this report "transforms a set of individuals", by adopting genetic operators that follow the "Darwinian principle of reproduction and survival of the fittest". (John Koza, 1992)(Maad, 2016).

In order to further understand the connection between "Evolution by means of natural selection", and the computational representation, it is critical to grasp the terminology of the biological setting. According to Ballard and Wallace, a population of species consists of individuals with a set of chromosomes. Each pair of chromosomes is a string DNA made up of genes (Ballard). After sexual reproduction, and mutation, the offspring inherits a combination of genetic material from both parents, this is referred to as the individual's genotype. In turn, the genotype is read by proteins that will construct the individual's phenotype, or its appearance as a result of its genes. **Table 2.1** represents the comparison between real-life genetics and the genetic algorithm.

Genetics	Algorithmic Representation
Generation	A population existing at the same time
Population	Array of Individuals
Individual	An object
Chromosome	Array of Characters
Phenotype	String of Characters

Table 2.1: Comparing terminology

Overall, the GA can be branched into four sections: assignation of fitness score, selection of individuals to reproduce, crossover and mutation.

2.1.1 The Fitness Function

One of the properties of Genetic Algorithms is the fitness function. The fitness function implemented within the Genetic Algorithm described in this report assigns a value to each individual based on their similarity to the targeted individual that the algorithm is trying to achieve and reach within a number of generations. The following steps outline the algorithmic thinking behind the described fitness function:

1. The individual chosen as the "target" individual is made up of 11 characters, and is defined as:

   `HELLO WORLD`

2. Each individual of a population, that makes up the current generation, is passed to the fitness function.
3. The first chromosome (or character) of the individual is checked.
4. If the chromosome of the individual is the same, and is in the same position, as the chromosome of the "target" individual, then the fitness score of the individual increases by 1.
5. Steps 3 and 4 are repeated until all chromosomes of the individual are compared with the chromosomes of the "target" individual and afterwards the result is divided by 11.
6. Steps 2 to 5 are repeated until all individuals of the population have been assigned a fitness value.

The fitness value of an individual increases by $\frac{1}{11}$ every time that a chromosome matches with the target's chromosome, such that when the checked individual is the same as the "target" individual, it will reach a fitness value of 1. The implementation of the fitness function can be observed in **APPENDIX A**.

2.1.2 Selection

The technique of selection with a scope of tackling the research question is the "Elitist Approach". It is inspired by the Darwinian concept "the survival of the fittest", the word fittest implying the best-suited individuals for a certain type of environment. By examining species reproduction, it can be fair to argue that if it wasn't for natural selection, there would be a much greater overproduction of offspring. In fact, "populations have the capacity to increase in numbers exponentially" (Gregory T. Ryan, 2009). As it can also be observed in the GA outlined and explained in section 3.0, if every individual is able to reproduce and

sequentially produce two offsprings, each of which will also be able to reproduce and create other two offsprings (and so on), every specie would contribute to an overproduction of offsprings and populations would expand rapidly. Now, this does not occur because of, among other factors, the "struggle for existence". Not every offspring will have the ability to adapt to their environment and either survive or reproduce. Organisms that do succeed in the "struggle for existence", often possess some heritable traits that make them fitter for the environment and will allow them to create more offspring than others.

As stated previously, for the genetic algorithm to simulate this approach, the individuals that will be able to reproduce and generate offspring are chosen using the "Elitist Approach". This type of selection takes the first n fittest individuals from a population (which is sorted in ascending order based on their fitness scores) and makes them the so called "parents" of the new generation of offsprings. Reproduction is represented by the genetic operator "cross-over (discussed in section 2.1.2). This type of selection denies the reproduction between strong and weak individuals (Alabsi, 2012). The main limitation of this type of selection, to keep in mind throughout the report, is that by selecting only the fittest individuals of every generation will lead to a loss of genetic information from one generation to the other.

2.1.3 Reproduction (Crossing over)

After a population has passed through selection mechanisms, the selected individuals will reproduce with each other to create a completely new generation of offspring. This is done using the genetic operator "crossover", that works with a probability of x that determines how likely it is for two individuals to create descendants. There are several different types of crossover methods. The Genetic Algorithm described in this report implements crossover based on a single point chosen by random. This point identifies the position where the genes of a selected individual, acting as a parent, should split and swap with the genes of another selected individual, also acting as a parent. Each crossover will give rise to two new offsprings. The following steps outline the approach used to represent crossover:

1. Two of the individuals selected through the elitist selection (which haven't crossed over with each other yet) are chosen. These individuals are now described as the "parents".
2. A random number is generated. If this number is above the threshold for the chosen crossover rate, then crossover occurs.
3. A crossover point is randomly selected
4. The two parents are cloned to create the new offspring. For example: offspring 1 is the clone of parent 1 and offspring 2 is the clone of parent 2.
5. The chromosomes of the two parents are split at the crossover point.
6. One offspring is made up of one section of one of the parent's chromosomes and the opposite section of the other parent's chromosomes. The other offspring will be made up of the other remaining sections.
7. Each offspring is exposed to the chance of mutation in each of its chromosomes (Section 2.1.2)
8. Two new offsprings have now been created and regarded as new individuals of the next generation.
9. Steps 1 to 8 are repeated until all selected parents have reproduced with each other and the size of the set of offspring of the new generation is the same as the one of the old generation.

Following step 2, if the threshold for crossover is not reached, then the individuals do not reproduce and both of them are sent to the new generation. Overall, these steps lead to the creation of a whole new generation of individuals, created both by the reproduction of the fittest individuals from the previous generation, and/or by carrying over fit individuals, in case these did not reproduce. The following is an example of single point crossover:

Parents with Crossover at point 5 (randomly chosen)

Parent 1: AABCA|ABACCCAAA

Parent 2: BACCA|BBACBBCCA

New Offspring (each clone of one parent)

Offspring 1: AABCAABACCCAAA

Offspring 2: BACCABBACBBCCA

After crossover

Offspring 1: AABCABBACBBCCA

Offspring 2: BACCABBACBBCCA

2.1.4 Mutation

A factor that has to be considered when simulating reproduction, is the impact of variation and inheritance. When DNA is replicated, some "errors" may occur that lead to variations. These errors, or mutations, can be "detrimental, beneficial or neutral" for the new individual. In the Genetic algorithm, the operator of mutation serves to continue introducing diversity in a population. For each gene in the chromosomes of a new individual, there exists a possibility that it will mutate. Mutation can happen at different rates (which will also be tested throughout this report).

3.0 Implementation

The constant variables used in the following code can be observed in APPENDIX B.

3.1 Selection

3.1.1 JAVA

```
Individual[] nextGen = new Individual[population.length];
          int pos = 0;
          Individual[] parents;
          Individual[] kids;

for(int i = 0; i < ELITE_CHOICE; i++){
     for(int j = 0; j < ELITE_CHOICE; j++){
          if(i!=j && pos<population.length-1){
               kids = newIndividuals(population[i],
                         population[j]);
               nextGen[pos] = kids[0];
               pos++;
               nextGen[pos] = kids[1];
               pos++;
          }
     }
}
```

3.2 Reproduction (Crossing over)

3.2.1 JAVA

```
private static Individual[] newIndividuals(Individual parent1, Individual
parent2){

      Individual newIndividual1, newIndividual2;
      double positionCrossOver = (Math.random() *10)+1;
      int startCrossOverAt = (int) positionCrossOver;

      newIndividual1 = parent1.clone();
      newIndividual2 = parent2.clone();

      for(int i=startCrossOverAt;i<newIndividual1.getChromosome().length; i++){
            newIndividual1.getChromosome()[i] = parent2.getChromosome()[i];
            newIndividual2.getChromosome()[i] = parent1.getChromosome()[i];
      }

      return new Individual[] {newIndividual1,newIndividual2};
}
```

3.3 Mutation

3.3.1 JAVA

```
for(int i = 0;i < nextGen.length; i++){
      mutate(nextGen[i]);
}
```

```
private void mutate (Individual individual){

      char[] chromIndiv = individual.getChromosome();
      int counter = 0;

      for(int i = 0; i < chromIndiv.length; i ++){
            if(Math.random() < RATE_OF_MUTATION){
            chromIndiv[i] = alphabet[(int) (Math.random() * 27)];
            }
      }
}
```

4.0 Experiments

To fully tackle the research question, a series of experiments, with the outlined genetic algorithm, were conducted. This process involved changing one parameter (between, elitism rate, crossover rate and mutation rate) while keeping the others constant and recording the results. The stopping criteria considered was finding one individual with the genotype of the target "**`Hello World`**", within a maximum of 100,000 generations.

With this purpose in mind, the following control variables were implemented:

- ***Population size:*** how big the population of study is.
- ***Mutation rate***: how likely it is for each gene to undergo a mutation.
- **Crossover rate**: how likely is for two individuals to reproduce or generate two offspring.
- ***Elitism rate:*** what proportion of the population we considered when choosing the fittests.
- ***Max number of generations***: up to what point we want our program to try and find a converged population (useful to know up to where the program looks for the target).
- ***Runs:*** how many times you want to perform the experiment with certain conditions.

In order to perform these experiments a separate class, "**`TestGA.java`**" was implemented. Figure 4.1 represents an example of what is outputted when this class is run.

```
Number of individuals in population: 100
Rate of mutation: 0.15
Rate of crossover: 1.0
Elitism rate: 0.1
Max number of generations allowed: 10000
Average number of generations needed to find target individual in 20 experiments:
17.55
The solution found after the fewest generations was with: 8 generations.
The solution found after the most generations was with: 33 generations.
```

Figure: 4.1

5.0 Results

5.1 Sub-question A

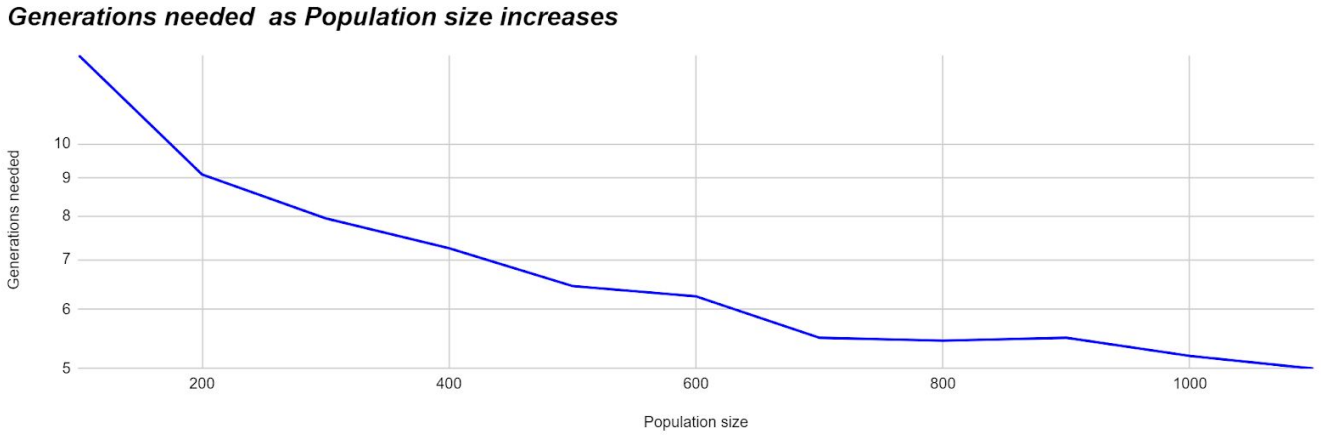
"What is the influence of a larger/or smaller population?"

Initial population size	Average number of generations needed to find **HELLO WORLD**
100	13.15
200	9.1
… … … … … …	
700	5,5
800	5,45

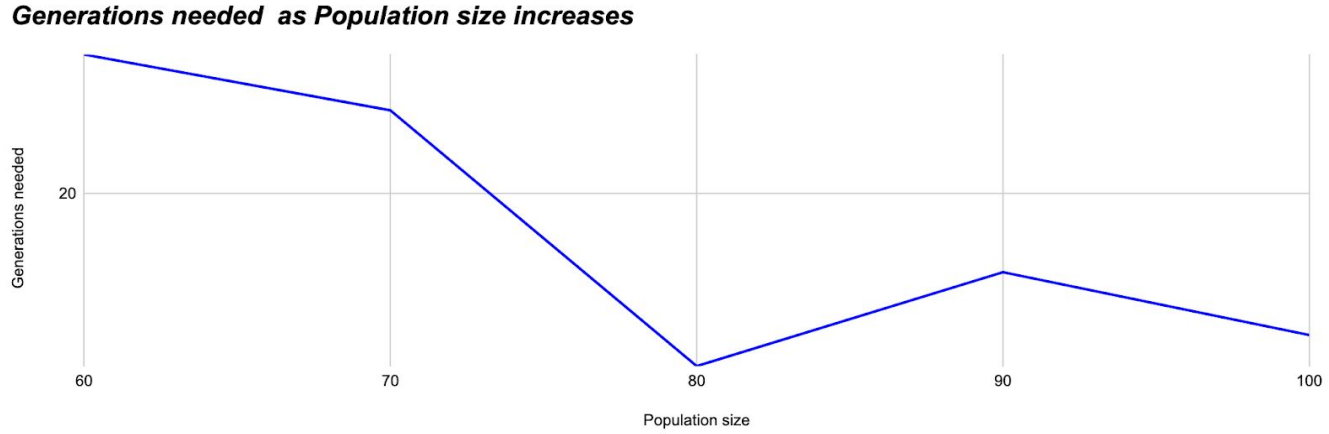
Table 1

Initial population size	Average number of generations needed to find **HELLO WORLD**
100	15.8
90	17.55
… … … … … …	
60	25.25

Table 2



These results are the average of 20 experiments in eah case. Constant parameters: Elitism rate (0,1), Mutation rate (0,15) and and Crossover rate (1).



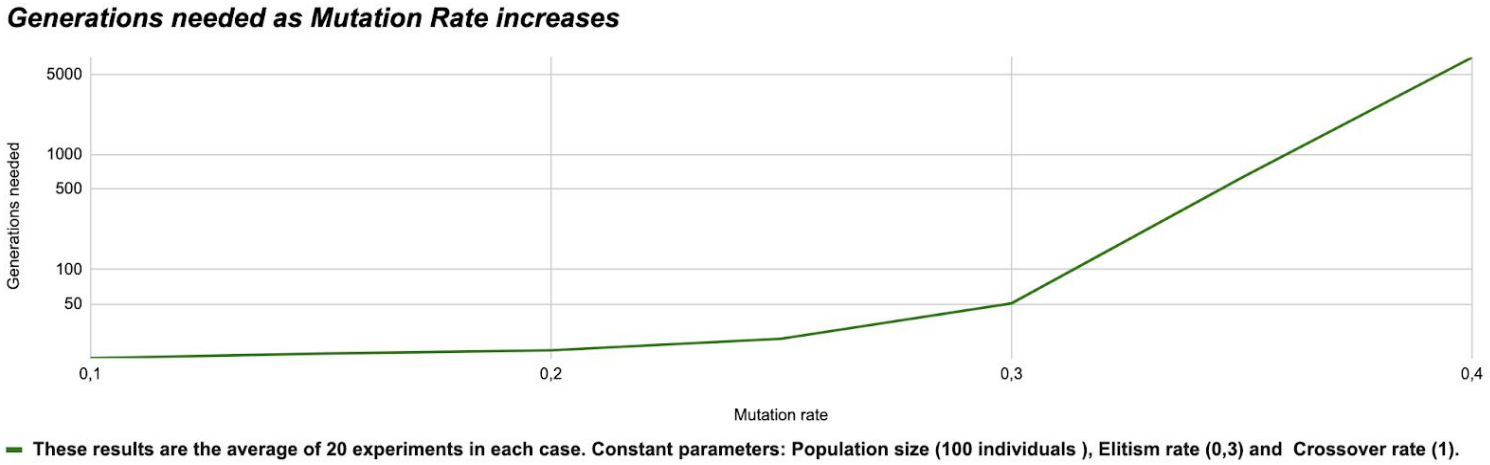
These results are the average of 20 experiments in eah case. Constant parameters: Elitism rate (0,1), Mutation rate (0,15) and Crossover rate (1).

5.2 Sub-question B

What is the influence of a larger/or smaller mutation rate? With an elitism rate of 0.3

Mutation rate	Average number of generations needed to find **HELLO WORLD**
0.1	16,9
0,2	19,8
0,3	50,75

Table 3

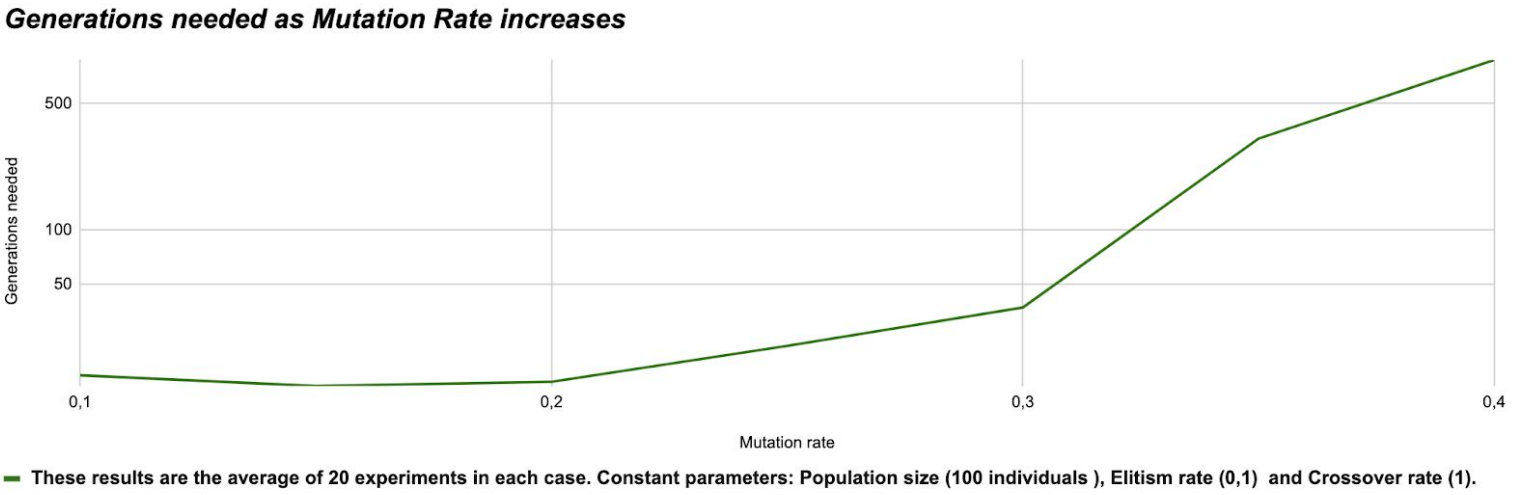


Graph 3

What is the influence of a larger/or smaller mutation rate? With an elitism rate of 0.1

Mutation rate	Average number of generations needed to find **HELLO WORLD**
0.1	15.7
0,2	14.45
0,3	37.25

Table 4



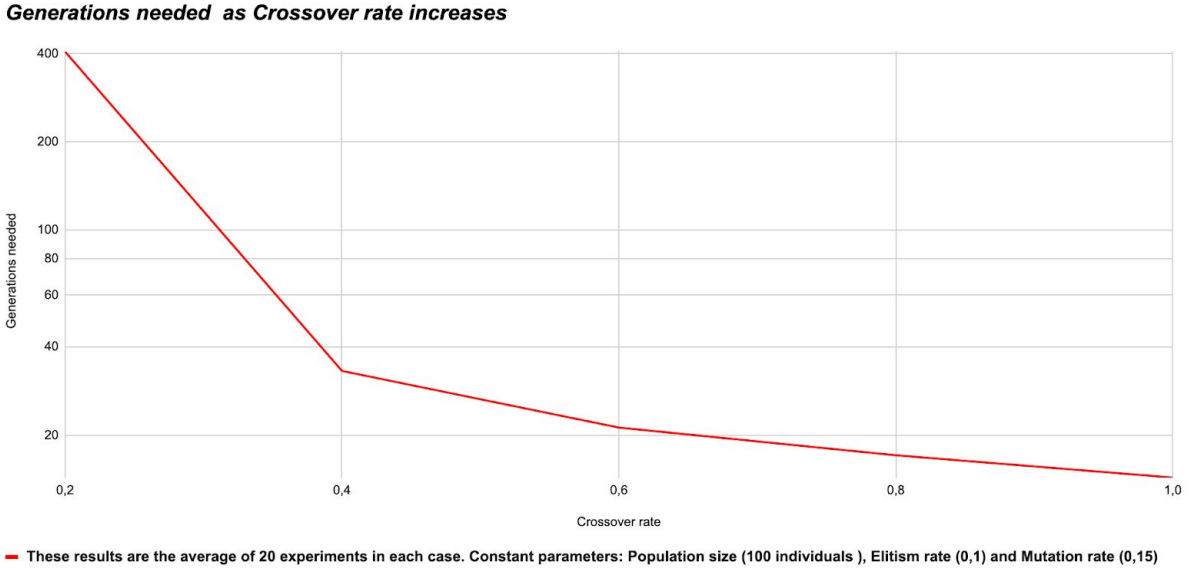
Graph 4

5.3 Sub-question C

What is the influence of a smaller crossover rate? With an elitism rate of 0.1

Crossover rate	Average number of generations needed to find **HELLO WORLD**
1	14.35
0,8	17.1

Table 5

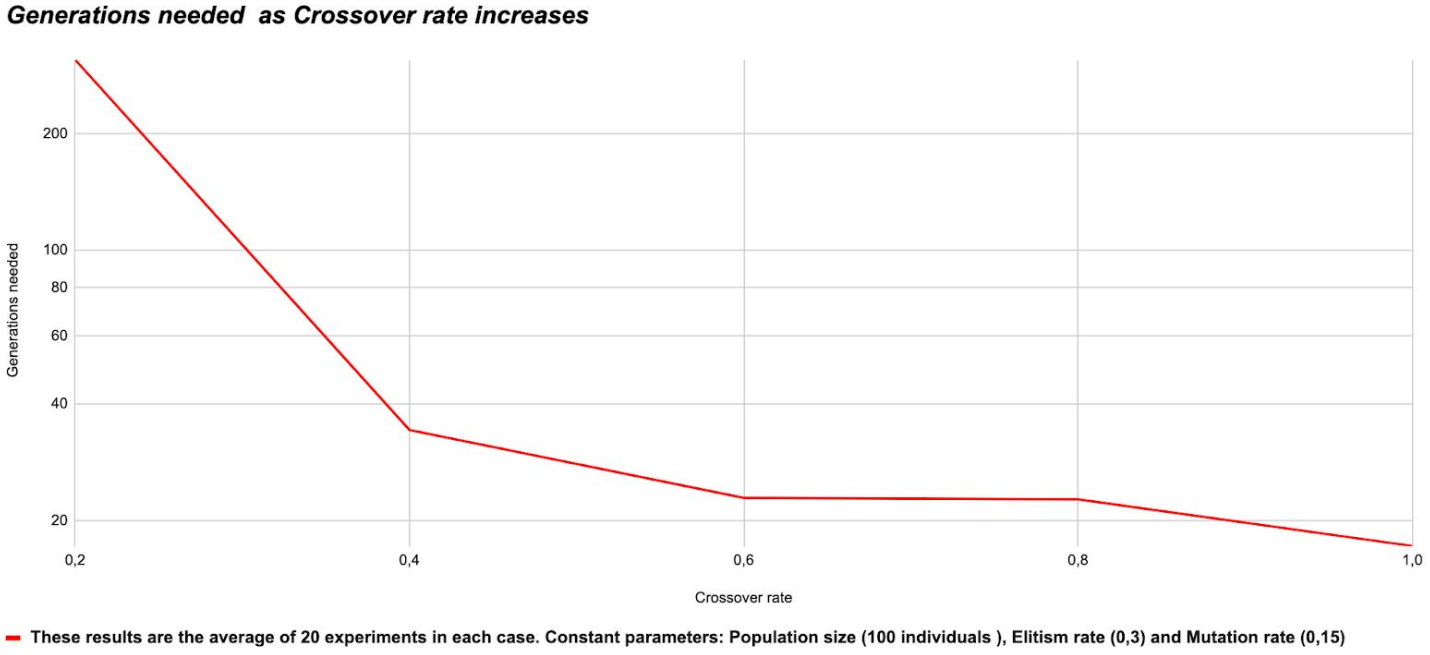


Graph 5

What is the influence of a smaller crossover rate? With an elitism rate of 0.3

Crossover rate	Average number of generations needed to find **HELLO WORLD**
1	17.2
0,8	22.7

Table 6



Graph 6

For each dependent variable, 100 experiments were performed. The average number of generations needed for the algorithm to find "**HELLO WORLD**" (for each tested variable) was recorded after every 20 tests. More raw data can be observed in **APPENDIX C**.

6.0 Conclusion and Observations

"What is the influence of a larger/smaller population? Is there a notable difference in the number of generations needed to find a solution?"

As it can be observed in both Table 1 and Table 2, and their respective graphs, there is a negative correlation between population size and number of generations needed to converge. In other words, as the population size increases, the number of generations needed to converge decreases. For example, with populations that consist of 1000 individuals or higher, the number of generations needed was stable at 5.2-5.0.

With constants for mutation rate (15%), crossover rate (100%), and Elitist selection on the top 10%, we can recognize that with a population of size 80 we obtain the possible optimal number of generations needed to converge, with an average of 15 generations.

One final experiment was performed using a population of size 3000. For this size, the number of generations required is on average 4.45. It is fair to argue therefore, that with larger populations, in terms of size, the chance of finding a target individual increases considerably, assuming that a bigger population holds a larger genetic diversity.

"What is the influence of a larger/smaller mutation rate? Is there a difference in the number of generations needed to find a solution?" and "Does your GA still work without mutation? Why or why not?"

As it can be observed in Table 2 and Table 3, and their respective graphs, with a constant population size of 100 individuals and crossover rate of 100%, the higher the mutation rate, the longer it takes for a population to converge. The results represented by Table 2 were outputted by the GA using an elitist selection of 30%, while the results in Table 3, a selection of 10%.

An important remark is that when the mutation rate was higher than 0.5, in most cases our algorithm could not find the target individual within 100,000 generations. In the same way, when there was no mutation at all, our populations did not converge into the target individual. Moreover, we observed that an "ideal" mutation rate, in both sets of experiments, is 0.1-0.2.

On one hand, we can conclude that mutation is needed in order to provide genetic variation. If a gene is not present in an initial population, the only way of obtaining it is through mutation. While, on the other hand, if the mutation rate is too high, most of the genes

of our offspring change, with the possibility of losing beneficial ones. In fact, with a mutation rate of 100%, all genes of our offspring change, and we lose the benefit of crossover of fit individuals, therefore the search for this becomes a random search.

"Now leave out the crossover operator. Do you see a difference in the number of generations needed to find a solution?"

For the final subquestion, experiments were performed based on changing the crossover rate. While keeping a constant population size (100 individuals), Elitism rate (10% in Table 5 and 30% in Table 6), and mutation rate (15%) we tested the influence of a changing crossover rate (from 0 to 100%) on the amount of generations needed to find a solution.

Three things called our attention. Firstly, populations exposed to a crossover probability of 1 took the least amount of time to find a solution. Secondly, that the populations that took the longest to converge were the ones with 0.2 chance of successful crossover. And finally, populations with a crossover rate of less than 0.1 never converged. All of these findings apply to both experiments (Elitism rate of 0.1 and 0.3).

It is fair to conclude that the higher the crossover rate between two individuals of a population, the faster the population converges. This may be due to the fact that new individuals with genetic material from both parents would be generated more often, and therefore the chances of obtaining more genetic variation and fitter offspring increased. Without crossover, the populations do not converge within our upper limit and we deduce that the reason for this is due to the fact that search becomes a random search based on the mutation of individual genes.

In conclusion, it is fair to argue that the genetic algorithm outlined in this report is able to simulate the general and overall process of evolution by natural selection. Keeping in mind that genetic algorithms are huge abstractions of what actually occurs in real life, this algorithm represents the selection of the fittest individuals (inspired by the concept of the survival of the fittest), simulates reproduction (also considering that not every individual is able to reproduce) and finally includes the possibility of an error in the replication of DNA after reproduction (mutation), thus including variation and diversity.

References

- Alabsi, F. (2012). Comparison of Selection Methods and Crossover Operations using Steady State Genetic Based Intrusion Detection System. Retrieved from https://www.academia.edu/5240870/Comparison_of_Selection_Methods_and_Crossover_Operations_using_Steady_State_Genetic_Based_Intrusion_Detection_System
- Claus Nielsen. (2012). *Animal Evolution : Interrelationships of the Living Phyla: Vol. 3rd ed*. OUP Oxford.
- Darwin, C. (1864). *On the Origin of Species by Means of Natural Selection, or, the Preservation of Favoured Races in the Struggle for Life*. New York, D. Appleton and Co.
- Gregory, T. Ryan (2009, June, 01). Understanding Natural Selection: Essential Concepts and Common Misconceptions. Retrieved from https://www.researchgate.net/publication/225401882_Understanding_Natural_Selection_Essential_Concepts_and_Common_Misconceptions
- Griffiths, A. J. (2019, October 11). Heredity. Retrieved from https://www.britannica.com/science/heredity-genetics
- Hosch, W. (2017, April 19). Genetic algorithm. Retrieved from https://www.britannica.com/technology/genetic-algorithm
- Mijwil, Maad. (2016). Genetic Algorithm Optimization by Natural Selection.

APPENDIX A- The implementation of the Fitness Function

```
/** Method fitness Individual:
It calculates the fitness by giving 1/11 points to each individual for each similar gene to the target.
(Note that the letter needs to be in the same position as the target.)
@param individual: from who I want to calculate the fitness score
*/
private void fitnessIndividual (Individual individual){
    double fitnessScore = 0;
    //check if contains letter
    for (int i=0; i<TARGET.length(); i++){
        // check for letter in right possition, give 1/11 points
        if (individual.getChromosome()[i] == TARGET.charAt(i)){
            fitnessScore += 1;
        }
    }
    fitnessScore = fitnessScore / 11.0;
    //assign fitnessScore to individual
    individual.setFitness(fitnessScore);
}
```

APPENDIX B

The following is an example of the implementation of the constant variables that were used throughout the algorithm and its testing:

```
String target = "HELLO WORLD";
int population = 100;
double mutantRate = 0.15;
double crossOverRate= 1;
double elitism = 0.1;
int maxGenerations = 10000;
```

APPENDIX C

Table and Graph 1: Generations needed as mutation rate increases: Population of size 100, Elitism rate of 0.3 and crossover rate of 1.

Mutation rate (as a proportion of 1)	0,1	0,15	0,2	0,25	0,3	0,35	0,4
Generations needed	16,9	18,55	19,8	24,95	50,75	631,15	7031,75

Table and Graph 2: Generations needed as mutation rate increases: Population of size 100, Elitism rate of 0.1 and crossover rate of 1.

Mutation rate (as a proportion of 1)	0,1	0,15	0,2	0,25	0,3	0,35	0,4
Generations needed	15,7	13,7	14,45	22,85	37,25	320,85	874,5

Table and Graph 3: Generations needed as Population size increases: Elitism rate 0.1, Mutation rate 0.15 and Crossover rate 1.

Population size	100	90	80	70	60
Generations needed	15,8	17,55	15	23	25,25

Table and Graph 4: Generations needed as Population size increases: Elitism rate 0.1, Mutation rate 0.15 and Crossover rate 1.

Population size	100	200	300	400	500	600	700	800	900	1000	1100
Generations needed	13,15	9,1	7,95	7,25	6,45	6,25	5,5	5,45	5,5	5,2	5

Table and Graph 5: Generations needed as Crossover rate increases: Elitism rate 0.1, Mutation rate 0.15 and Population size 100.

Crossover rate (as a proportion of 1)	1	0,8	0,6	0,4	0,2
Generations needed	14,35	17,1	21,25	33,15	406,25

Table and Graph 6: Generations needed as Crossover rate increases: Elitism rate 0.3, Mutation rate 0.15 and Population size 100.

Crossover rate (as a proportion of 1)	1	0,8	0,6	0,4	0,2

Generations needed	17,2	22,7	22,9	34,3	310,3