Instituto Politécnico Nacional

Escuela Superior de Cómputo

Evolutionary Computing

Session 3 "Genetic Algorithms"

Student: Naranjo Ferrara Guillermo

Teacher: Rosas Trigueros Jorge Luis

Lab session date: 18 February 2020

Delivery date: 13 March 2020

1 Theoretical Framework

In order to understand what is a genetic algorithm, we first need a notion of genetics so I will talk about a little history of genetics an mention a few important concepts that will help us to see how its studies and principles can be taken to the area of computer science for the solution of problems.

Genetics is a branch of biology that studies how the hereditary characteristics are transmitted from generation to generation.[1]

1.1 A brief history[2]

In the later 1800s, it was discovered that the nucleus of every cell contains large, elongated molecules that were dubbed **chromosomes** ("colored bodies"). It also was discovered that an individual cell reproduces itself by dividing into two identical cells, during which process (dubbed mitosis) the chromosomes make identical copies of themselves.

Meiosis is the process in diploid organisms by which eggs and sperm are created. Diploid organisms, including most mammals, are those in which chromosomes in all cells (except sperm and egg, or germ cells) are found in pairs. During meiosis, one diploid cell becomes four germ cells, each of which has half the number of chromosomes as the original cell. Each chromosome pair in the original cell is cut into parts, which **recombine** to form chromosomes for the four new germ cells. During fertilization, the chromosomes in two germ cells **fuse** together to create the correct number of chromosome pairs. The result is that the genes on a child's chromosome are a mixed-up version of its parents' chromosomes. This is a major source of variation in organisms with sexual reproduction.

The first suggestion that chromosomes are the carriers of heredity was made by Walter Sutton in 1902, two years after Mendel's work came to be widely known. Sutton hypothesized that chromosomes are composed of units dubbed **"genes"** that correspond to Mendelian factors, and showed that meiosis gives a mechanism for Mendelian **inheritance**. Sutton's hypothesis was verified a few years later by Thomas Hunt Morgan via experiments.

DNA and RNA were both discovered by 1920's, but the biggest break came when, in 1953, James Watson and Francis Crick figured out that the structure of DNA is a double helix. In the early 1960s, the combined work of several scientists discovered how the parts of DNA encode the amino acids that make up proteins. A **gene** could now be defined as a sub-string of DNA that codes for a particular protein. Soon after this, it was worked out how the code was translated by the cell into proteins, how DNA makes copies of itself, and how variation arises via copying errors, externally caused mutations, and sexual recombination. This was clearly a "tipping point" in genetics research.

1.2 Bringing Genetics to Computer Science

We can say that Genetic Algorithms (GA) are probabilistic search procedures designed to work on large spaces involving states that can be represented by strings[3] of bits, numbers or symbols.[4]

But how we translate concepts and approaches from Genetics to Computer Science. Well the definition of a GA gives us an idea but lets talk a little bit more about it.

In a GA, a population of candidate solutions to an optimization problem is evolved toward better solutions (inheriting some characteristics to the next generation, as in Genetics). Each candidate solution has a set of properties which can be mutated or altered via different methods. The evolution is an iterative process which usually starts from a random generated population. Every iteration is called a generation.[5]

In each generation, the fitness (the value of the optimization function in the optimization problem being solved) of every individual is evaluated. The more fit individuals are stochastic-ally selected to form a new generation by recombining or mutating its genome.[5]

2 Material and Equipment

In order to this practice we need:

- A working computer.
- Any version (preferentially the last) of python installed in the computer.

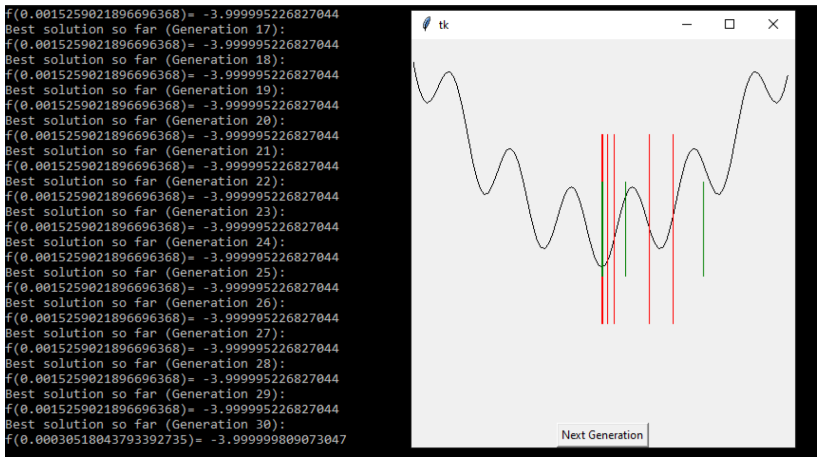
3 Development

3.1 Function of example

The first function to test and observe the behaviour of a GA was the function in the example code. Here's the function:

$$f(x) = 0.05x^2 - 4cos(x) \tag{1}$$

The test was made with a population of 10 chromosomes, each one with a length of 16, and a probability of mutation of 50%. In the image below we can see the result of the test trough 30 generations.



```
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 17):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 18):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 19):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 20):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 21):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 22):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 23):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 24):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 25):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 26):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 27):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 28):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 29):
f(0.0015259021896696368)= -3.999995226827044
Best solution so far (Generation 30):
f(0.00030518043793392735)= -3.999999809073047
```

Figure 1: Test of the GA implementing the example function.

3.2 Rastrigin function 1D

The Rastrigin function has several local minima. It is highly multi-modal, but locations of the minima are regularly distributed. Here is a plot of the function in its two dimensional form.

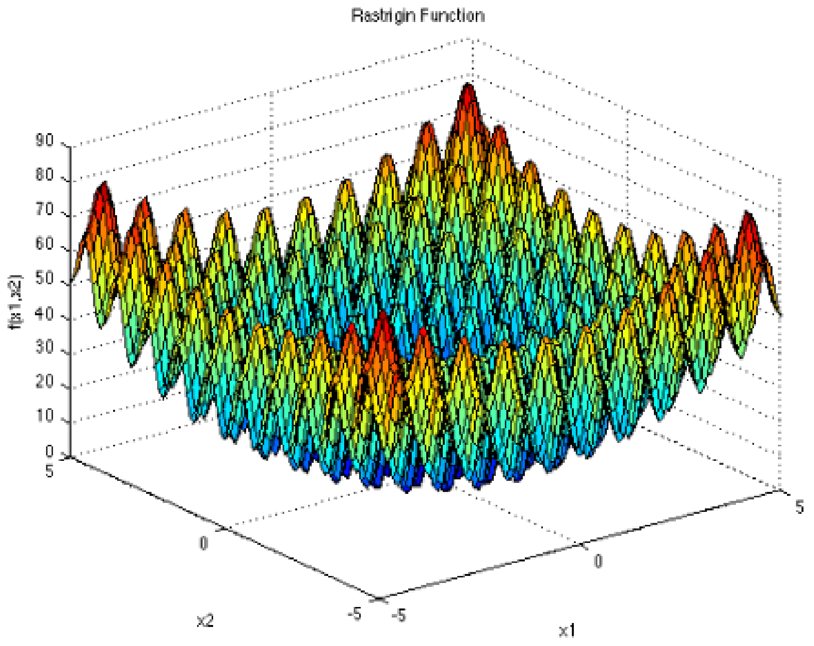


Figure 1: Rastrigin function in its two dimensional form.

Its equation is

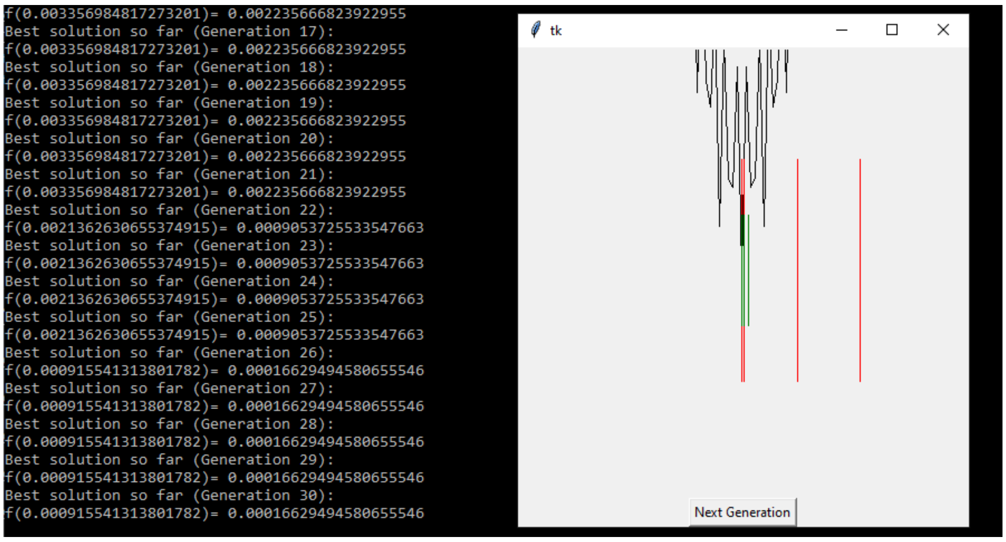
$$f(x) = 10d + \sum_{i=1}^{d} x_i - 10cos(2\pi x_i) \tag{2}$$

where: d - Number of dimensions

For the practice purposes we used the function in its one dimensional form, which is Its equation is

$$f(x) = 10 + x - 10cos(2\pi x) \tag{3}$$

The test was made with a population of 10 chromosomes, each one with a length of 16, and a probability of mutation of 50%. In the image below we can see the result of the test trough 30 generations.



```
f(0.003356984817273201)= 0.002235666823922955
Best solution so far (Generation 17):
f(0.003356984817273201)= 0.002235666823922955
Best solution so far (Generation 18):
f(0.003356984817273201)= 0.002235666823922955
Best solution so far (Generation 19):
f(0.003356984817273201)= 0.002235666823922955
Best solution so far (Generation 20):
f(0.003356984817273201)= 0.002235666823922955
Best solution so far (Generation 21):
f(0.003356984817273201)= 0.002235666823922955
Best solution so far (Generation 22):
f(0.0021362630655374915)= 0.0009053725533547663
Best solution so far (Generation 23):
f(0.0021362630655374915)= 0.0009053725533547663
Best solution so far (Generation 24):
f(0.0021362630655374915)= 0.0009053725533547663
Best solution so far (Generation 25):
f(0.0021362630655374915)= 0.0009053725533547663
Best solution so far (Generation 26):
f(0.000915541313801782)= 0.0001662949458065546
Best solution so far (Generation 27):
f(0.000915541313801782)= 0.0001662949458065546
Best solution so far (Generation 28):
f(0.000915541313801782)= 0.0001662949458065546
Best solution so far (Generation 29):
f(0.000915541313801782)= 0.0001662949458065546
Best solution so far (Generation 30):
f(0.000915541313801782)= 0.0001662949458065546
```

Figure 1: Test of the GA implementing the one dimensional Rastrigin function.

3.3 Ackley function 2D

The Ackley function is widely used for testing optimization algorithms. In its two-dimensional form, as shown in the plot above, it is characterized by a nearly flat outer region, and a large hole at the centre. The function poses a risk for optimization algorithms, particularly hill-climbing algorithms, to be trapped in one of its many local minima. Below it's shown a plot of the function.

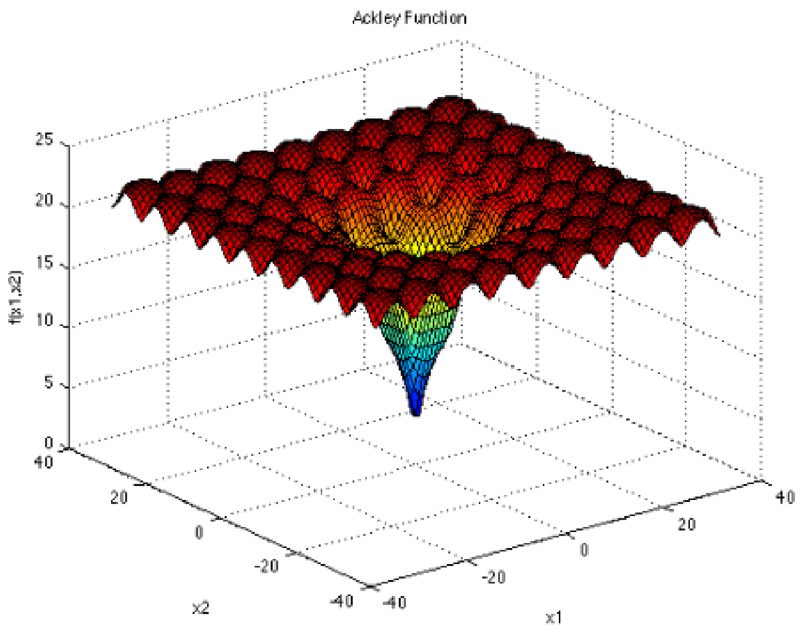


Figure 1: Ackley function.

Its equation is

$$f(x) = -a\,exp(-b\sqrt{1/d\sum_{i=1}^{d} x_i^2}) - exp(1/d\sum_{i=1}^{d} cos(cx_i)) + a + e \tag{4}$$

where: d - Number of dimensions
and the recommended values for the variables are: $a = 20$, $b = 0.2$ and $c = 2\pi$

For the practice purposes we used its two dimensional form. This cause two problems:

1. How to plot the function.
2. How to manage the use of two variables having only on chromosome for the function to evaluate it.

The first problem was solved when the professor told us it wasn't necessary to plot the function and the evolution of the population through the generations.

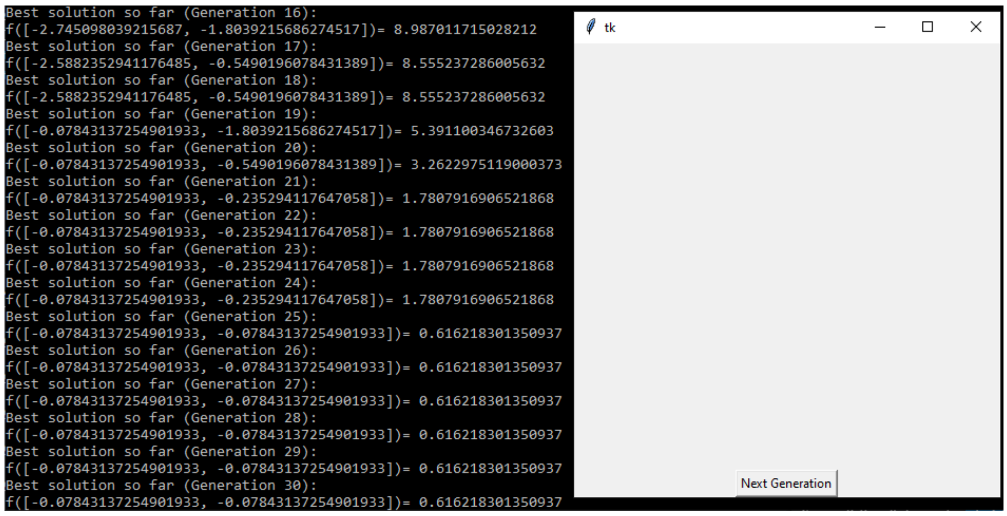
For the second problem the professor also gave us the answer consisting in use a half of the chromosome as the first variable, and the other half as the second, and so the only thing left to figure out was the decoding of the chromosome, but when this idea that was simple.

I only had to consider that the decoding had to return a list containing the respective values for x_1 and x_2, each one having a maximum value of

$$value(x_i) = 2^{lenChromosome/2} - 1 \tag{5}$$

because of the value of each x corresponding to the half of the chromosome.

After fix this details the test was made with the same values as in the other two functions. Here are the results.



```
Best solution so far (Generation 16):
f([-2.745098039215687, -1.8039215686274517])= 8.987011715028212
Best solution so far (Generation 17):
f([-2.5882352941176485, -0.5490196078431389])= 8.555237286005632
Best solution so far (Generation 18):
f([-2.5882352941176485, -0.5490196078431389])= 8.555237286005632
Best solution so far (Generation 19):
f([-0.07843137254901933, -1.8039215686274517])= 5.391100346732603
Best solution so far (Generation 20):
f([-0.07843137254901933, -0.5490196078431389])= 3.2622975119000373
Best solution so far (Generation 21):
f([-0.07843137254901933, -0.235294117647058])= 1.7807916906521868
Best solution so far (Generation 22):
f([-0.07843137254901933, -0.235294117647058])= 1.7807916906521868
Best solution so far (Generation 23):
f([-0.07843137254901933, -0.235294117647058])= 1.7807916906521868
Best solution so far (Generation 24):
f([-0.07843137254901933, -0.235294117647058])= 1.7807916906521868
Best solution so far (Generation 25):
f([-0.07843137254901933, -0.07843137254901933])= 0.616218301350937
Best solution so far (Generation 26):
f([-0.07843137254901933, -0.07843137254901933])= 0.616218301350937
Best solution so far (Generation 27):
f([-0.07843137254901933, -0.07843137254901933])= 0.616218301350937
Best solution so far (Generation 28):
f([-0.07843137254901933, -0.07843137254901933])= 0.616218301350937
Best solution so far (Generation 29):
f([-0.07843137254901933, -0.07843137254901933])= 0.616218301350937
Best solution so far (Generation 30):
f([-0.07843137254901933, -0.07843137254901933])= 0.616218301350937
```

Figure 1: Test of the GA implementing the two dimensional Ackley function.

4 Conclusion

For this practice I did some more tests (which were not included here) for each function, varying the values for the parameters like the probability of mutation and the length of the chromosomes. This tests made me realize how important is to choose the correct parameters for the algorithm to work correctly and to be more efficient.

For example, although the default length of the chromosome works well for the example function, for the rastrigin and ackley functions the algorithm converged to a value that, for our target, it wasn't still the optimal. As well as the length of the chromosome, the probability of mutation influenced in the performance of the algorithm but more in the speed of it to converge to the solution; with smaller probability of evolution the best individual was the same for a considerable amount of generations.

One thing the example code helped me so much was to identify the main functions that an GA has to implement in order to function correctly, as well as how to modulates them in order to have a clean implementation with great cohesion between them.

5 Bibliography

[1]Instituto Bernabeu. ¿Qué es la genética?. [Online]. Available: https://www.ibbiotech.com /es/info/que-es-la-genetica/

[2]M. Mitchel. *Complexity. A guided tour.* New York: Oxford University Press, 2009.

[3]D. E. Goldberg, J. H. Holland, "Genetic Algorithms and Machine Learning", *Machine Learning*, no. 3, pp. 95-99, 1998.

[4]C. Reeves, "Genetic Algorithms", *Handbook of metaheuristics*, Chapter 3, pp. 109-139, September 2010.

[5]"Evolutionary Computing", class notes, Academy of Computer Science, ESCOM-IPN, Primavera 2020.