Instituto Politécnico Nacional

Escuela Superior de Cómputo

Evolutionary Computing

3CM1

# 2 Genetic Algorithm

Santiago Nieves Edgar Augusto

Teacher: Jorge Luis Rosas Trigueros

Date: 12/08/2016

Deadline: 13/08/2016

**Theoretical framework:**

Genetic algorithms search the solution space of a function through the use of simulated evolution ie the survival of the test strategy In general the ttest individuals of any population tend to reproduce and survive to the next generation thus improving successive generations, genetic algorithm is a metaheuristic inspired by the process of natural selection that belongs to the larger class of evolutionary algorithms. Genetic algorithms are commonly used to generate high-quality solutions to optimization and search problems by relying on bio-inspired operators such as mutation, crossover and selection.

In a genetic algorithm a population of candidate solutions to an optimization problem is evolved toward better solutions. Each candidate solution has a set of properties which can be mutated and altered; traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible.

The algorithm requires:

- A genetic representation of the solution domain.
- A fitness function to evaluate the solution domain.

Genetic algorithm has some parts:

- Heredity: There must be a process in place by which children receive the properties of their parents. If creatures live long enough to reproduce, then their traits are passed down to their children in the next generation of creatures.
- Variation: There must be a variety of traits present in the population or a means with which to introduce variation. For example, let's say there is a population of beetles in which all the beetles are exactly the same: same color, same size, same wingspan, same everything. Without any variety in the population, the children will always be identical to the parents and to each other. New combinations of traits can never occur and nothing can evolve.
- Selection: There must be a mechanism by which some members of a population have the opportunity to be parents and pass down their genetic information and some do not. This is typically referred to as "survival of the fittest." For example, let's say a population of gazelles is chased by lions every day. The faster gazelles are more likely to escape the lions and are therefore more likely to live longer and have a chance to reproduce and pass their genes down to their children. The term fittest, however, can be a bit misleading.

**Material and equipment:**

Software:

Sublime Text.

Python 2.7.6.

Ubuntu 16.04 LTS.

Hardware:

Procesador: Intel® Core™ i3 CPU M 380 @ 2.1GHz × 4.

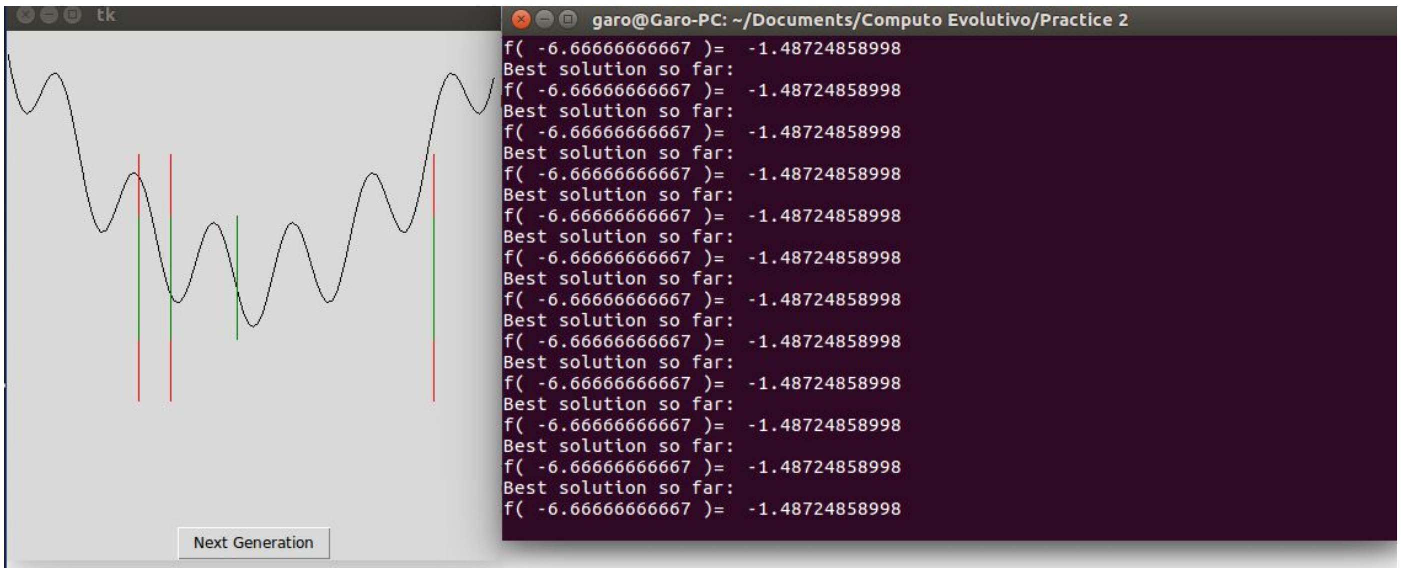
**Body:**

Given the program of genetic algorithm, modify the code to support 2 functions (Rastrigin and Ackley) and check the iterations of the converge with different probabilities and differents sizes.

We need to do a table where in the table has the probability and the number of chromosome of the first program (Table 1).

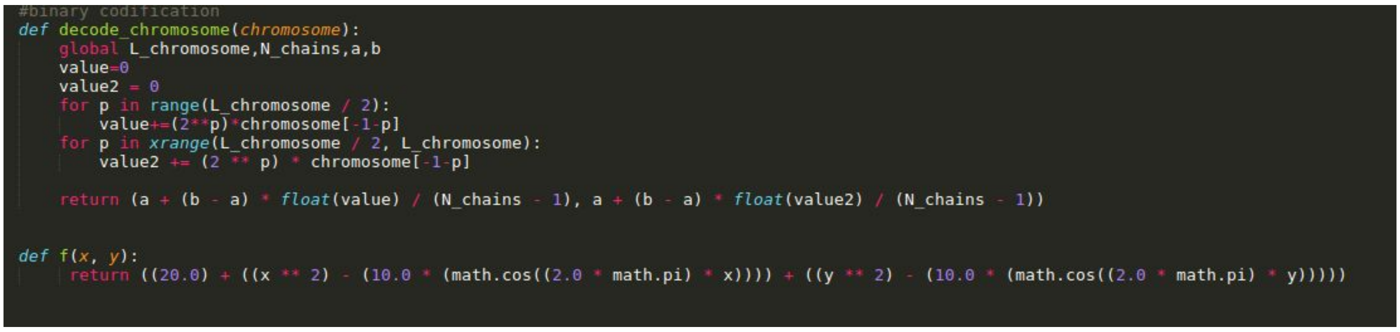
| **Probability / bits** | **4** | **8** | **16** | **32** |
|---|---|---|---|---|
| **0.1** | 1 | 4 | 7 | 34 |
| **0.25** | 4 | 5 | 12 | 40 |
| **0.50** | 1 | 1 | 1 | 33 |
| **0.75** | 1 | 20 | 4 | 8 |

Table1. The number of iterations of the best result with probability and bits.



```
f( -6.66666666667 )=  -1.48724858998
Best solution so far:
f( -6.66666666667 )=  -1.48724858998
Best solution so far:
f( -6.66666666667 )=  -1.48724858998
Best solution so far:
f( -6.66666666667 )=  -1.48724858998
Best solution so far:
f( -6.66666666667 )=  -1.48724858998
Best solution so far:
f( -6.66666666667 )=  -1.48724858998
Best solution so far:
f( -6.66666666667 )=  -1.48724858998
Best solution so far:
f( -6.66666666667 )=  -1.48724858998
Best solution so far:
f( -6.66666666667 )=  -1.48724858998
Best solution so far:
f( -6.66666666667 )=  -1.48724858998
Best solution so far:
f( -6.66666666667 )=  -1.48724858998
Best solution so far:
f( -6.66666666667 )=  -1.48724858998
```

The output of the first program.

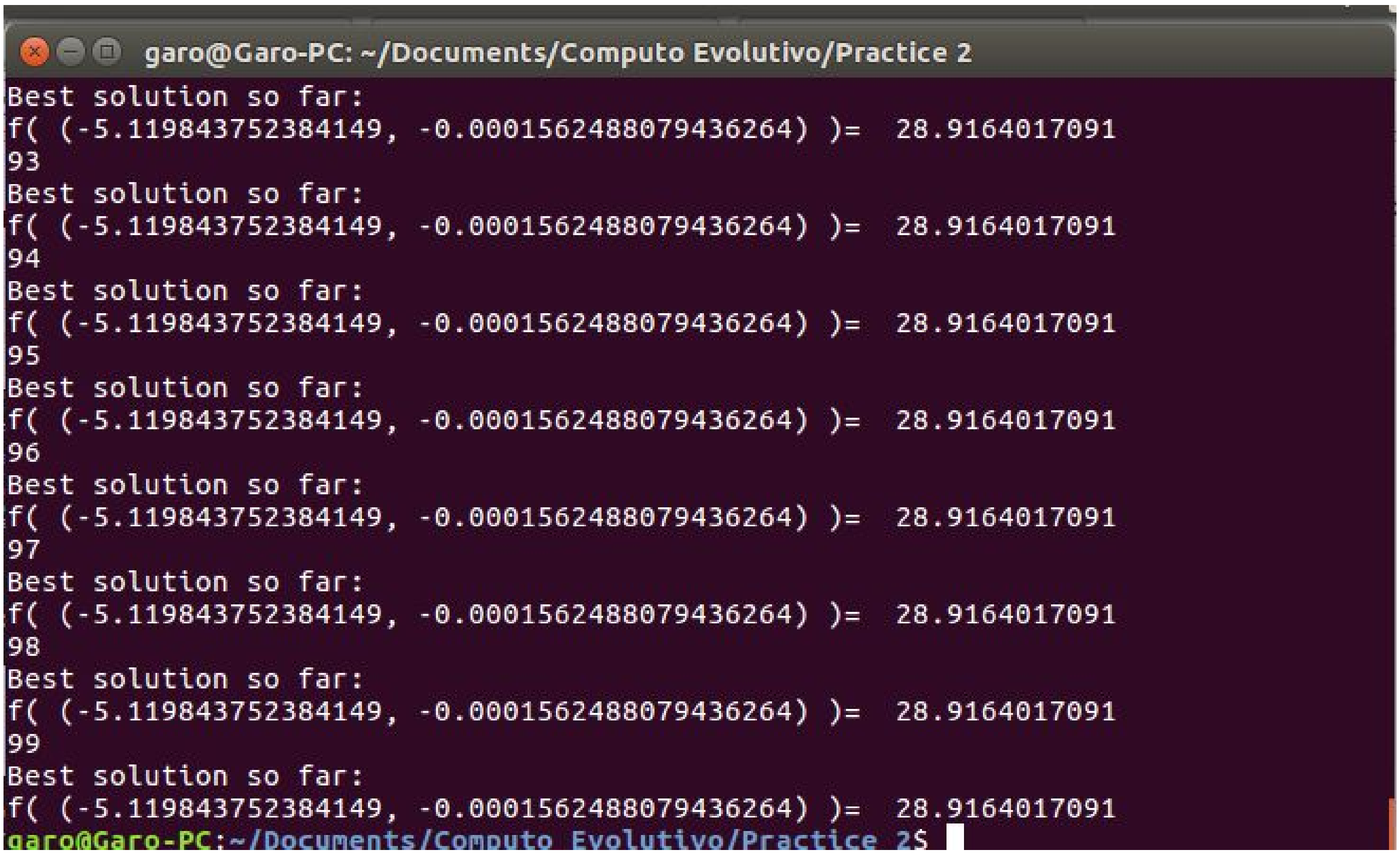


```
#binary codification
def decode_chromosome(chromosome):
    global L_chromosome,N_chains,a,b
    value=0
    value2 = 0
    for p in range(L_chromosome / 2):
        value+=(2**p)*chromosome[-1-p]
    for p in xrange(L_chromosome / 2, L_chromosome):
        value2 += (2 ** p) * chromosome[-1-p]

    return (a + (b - a) * float(value) / (N_chains - 1), a + (b - a) * float(value2) / (N_chains - 1))


def f(x, y):
     return ((20.0) + ((x ** 2) - (10.0 * (math.cos((2.0 * math.pi) * x)))) + ((y ** 2) - (10.0 * (math.cos((2.0 * math.pi) * y)))))
```

The function Rastrigin.



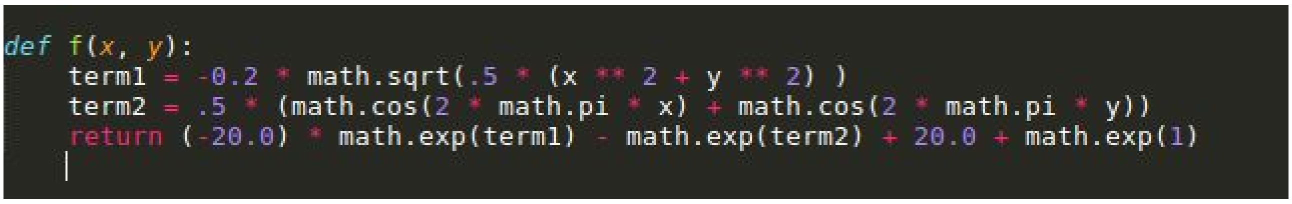
```
garo@Garo-PC: ~/Documents/Computo Evolutivo/Practice 2
Best solution so far:
f( (-5.119843752384149, -0.0001562488079436264) )=  28.9164017091
93
Best solution so far:
f( (-5.119843752384149, -0.0001562488079436264) )=  28.9164017091
94
Best solution so far:
f( (-5.119843752384149, -0.0001562488079436264) )=  28.9164017091
95
Best solution so far:
f( (-5.119843752384149, -0.0001562488079436264) )=  28.9164017091
96
Best solution so far:
f( (-5.119843752384149, -0.0001562488079436264) )=  28.9164017091
97
Best solution so far:
f( (-5.119843752384149, -0.0001562488079436264) )=  28.9164017091
98
Best solution so far:
f( (-5.119843752384149, -0.0001562488079436264) )=  28.9164017091
99
Best solution so far:
f( (-5.119843752384149, -0.0001562488079436264) )=  28.9164017091
garo@Garo-PC:~/Documents/Computo Evolutivo/Practice 2$
```

The output with the function Rastrigin.

We need to do a table of Rastrigin Function where in the table has the probability and the number of chromosome of the first program (Table 2).

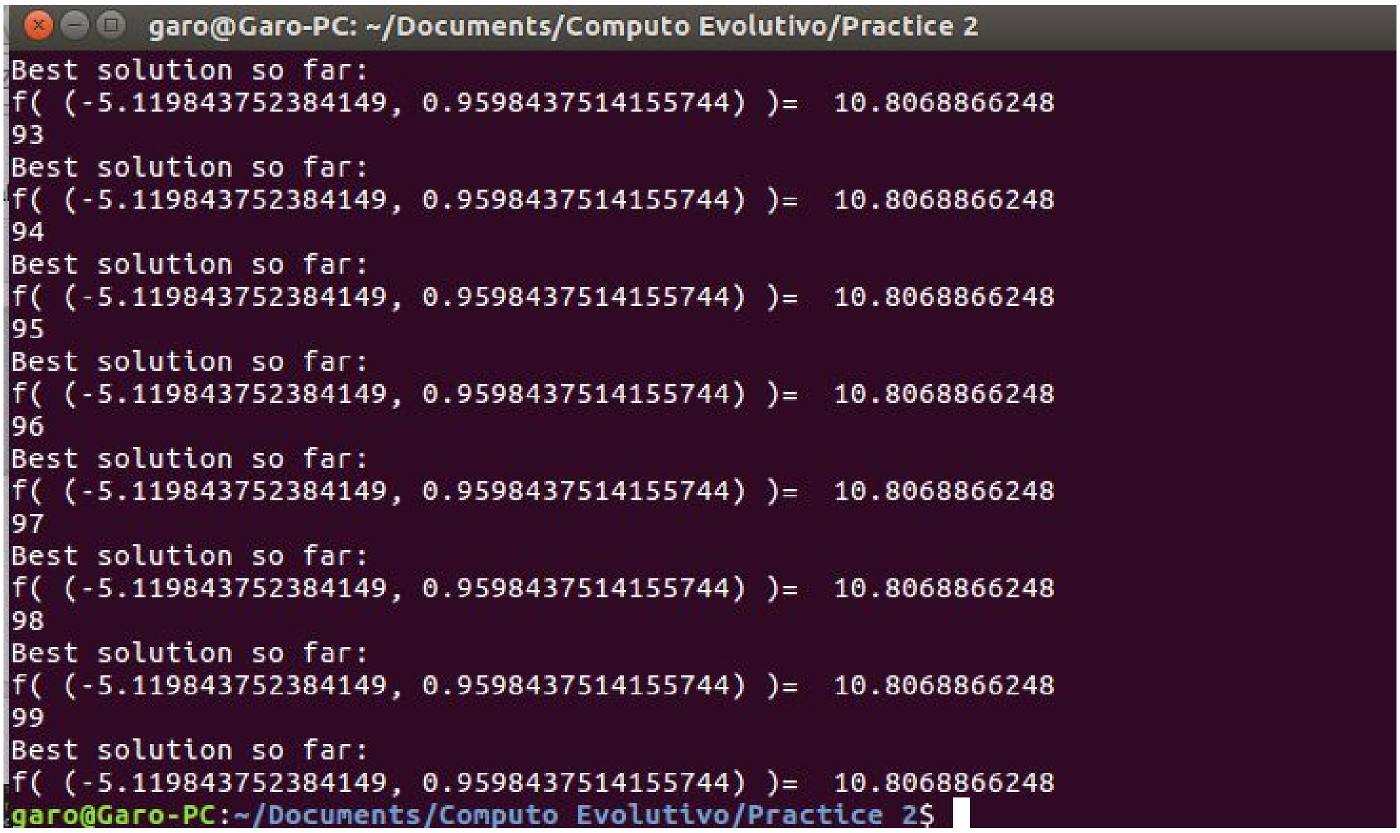
| **Probability / bits** | **8** | **16** | **32** | **64** |
|---|---|---|---|---|
| **0.1** | 12 | 24 | 153 | 14 |
| **0.25** | 12 | 35 | 26 | 77 |
| **0.50** | 9 | 4 | 66 | 56 |
| **0.75** | 103 | 14 | 29 | 28 |

Table 2. The number of iterations of the best result with probability and bits with Rastrigin.



```python
def f(x, y):
    term1 = -0.2 * math.sqrt(.5 * (x ** 2 + y ** 2) )
    term2 = .5 * (math.cos(2 * math.pi * x) + math.cos(2 * math.pi * y))
    return (-20.0) * math.exp(term1) - math.exp(term2) + 20.0 + math.exp(1)
```

The function Ackley.



```
garo@Garo-PC: ~/Documents/Computo Evolutivo/Practice 2
Best solution so far:
f( (-5.119843752384149, 0.9598437514155744) )=  10.8068866248
93
Best solution so far:
f( (-5.119843752384149, 0.9598437514155744) )=  10.8068866248
94
Best solution so far:
f( (-5.119843752384149, 0.9598437514155744) )=  10.8068866248
95
Best solution so far:
f( (-5.119843752384149, 0.9598437514155744) )=  10.8068866248
96
Best solution so far:
f( (-5.119843752384149, 0.9598437514155744) )=  10.8068866248
97
Best solution so far:
f( (-5.119843752384149, 0.9598437514155744) )=  10.8068866248
98
Best solution so far:
f( (-5.119843752384149, 0.9598437514155744) )=  10.8068866248
99
Best solution so far:
f( (-5.119843752384149, 0.9598437514155744) )=  10.8068866248
garo@Garo-PC:~/Documents/Computo Evolutivo/Practice 2$
```

The output with the function Ackley.

We need to do a table with Ackley Function where in the table has the probability and the number of chromosome of the first program (Table 3).

| **Probability / bits** | **4** | **8** | **16** | **32** |
|---|---|---|---|---|
| **0.1** | 1 | 10 | 91 | 432 |
| **0.25** | 4 | 29 | 55 | 159 |
| **0.50** | 1 | 19 | 398 | 41 |
| **0.75** | 2 | 5 | 211 | 43 |

Table 3. The number of iterations of the best result with probability and bits with Ackley.

## Conclusions:

In this practice I can learn about Genetic Algorithm, when the teacher teached about it I didn't catch the idea but when I did this practice I could step the topic up, in the beginning of the practice I didn't understand the idea of this topic and why we need to use the function Ackley and Rastrigin but step by step I could understand the topic, I think it's important to practice this kind of topic because if you only see the theory you never can understand it so it's necessary to put into practice this topic it's the way easier to learn about that.

## Bibliography

*A Genetic Algorithm for Function Optimization A Matlab Implementation Christopher R Houck North Carolina State University and Jeery A Joines North Carolina State University and Michael G Kay North Carolina State University Age*

*https://en.wikipedia.org/wiki/Genetic_algorithm*