A study on Genetic Algorithms for finding the global minimum of a set of functions compared to Hill Climbing and Simulated Annealing aproaches.

Tamaș Luca-Ștefan - Group E3, Year 2

1. Abstract:

This study presents the behaviour of a genetic algorithm in various scenarios of finding the global minimum of 4 different functions, with multiple dimension input vectors. This paper will analyze the differences of the multiple scenarios and will conclude that genetic algorithms are very powerful in comparison with Hill Climbing and Simulated Annealing, having the time complexity far more reduced than the two alternatives, with great and accurate results.

2. Introduction:

This paper will analyze the behaviour of genetic algorithms on 4 functions: *De Jong, Schwefel, Rastrigin, Michalewicz.* The algorithms will try to find the minimum of the 4 functions using 3 types of inputs: 5, 10 and 30 dimensional vectors. The increase of the dimension in input will affect the algorithms performance, but not as much as it did with the Simmulated Annealing and Hill Climbing, where the time needed for running significantly grew.

3. Method:

The algorithms used in this study are based on the structured following the pseudocode found on this page. The program represents all the individuals as an array of bits for increasing the efficiency of finding better solutions.

The formula used to calculate the number of bits for each number is:

$$bistringLength = \lceil \log_2(precision^{10} \cdot (b - a)) \rceil$$

Where precision is the maximum precision of the representation, b is the upper bound and a is the lower bound for the real values in the vectors.

Transforming the string of bits into the real numbers will be done using the fomrula:

$$x = \frac{binaryNumber(x_{bitnary}) \cdot (b - a) + a}{2^{bitstringLength} - 1}$$

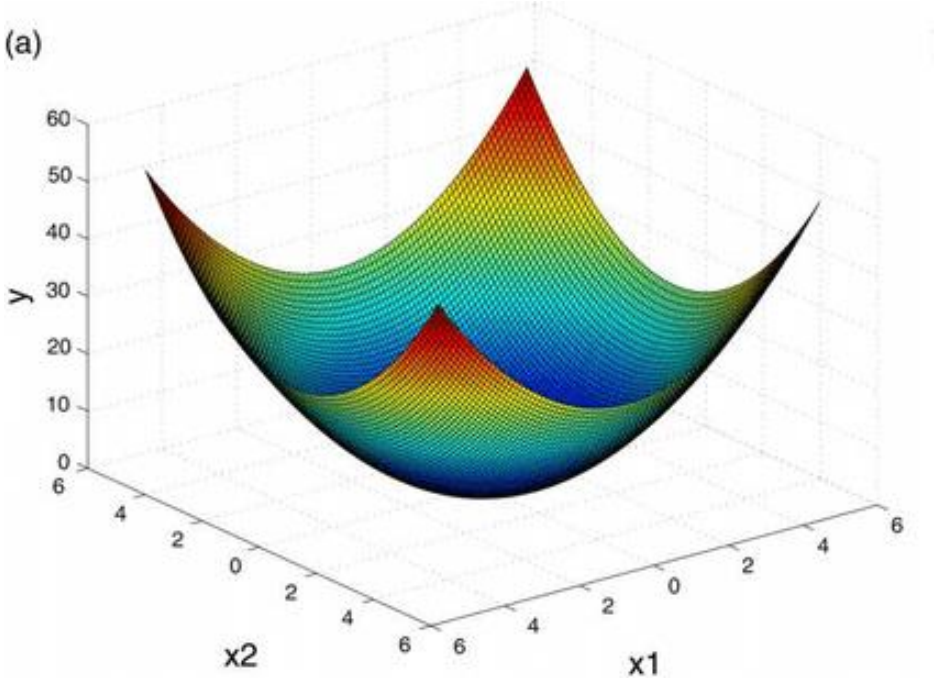
Where a and b are the previous numbers, and binaryNumber() is a functions that converts a binary string into a decimal value.

Genetic Algorithm

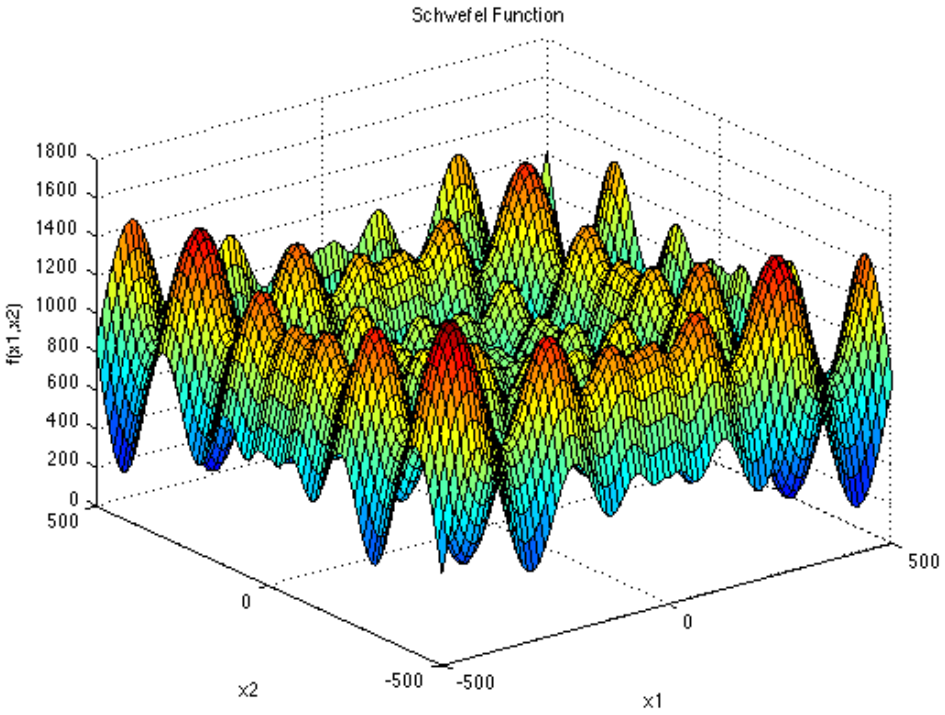
In computer science and operations research, a **genetic algorithm** (**GA**) is a metaheuristic inspired by the process of natural selection that belongs to the larger class of evolutionary algorithms (EA). Genetic algorithms are commonly used to generate high-quality solutions to optimization and search problems by relying on biologically inspired operators such as mutation, crossover and selection. Some examples of GA applications include optimizing decision trees for better performance, solving sudoku puzzles, hyperparameter optimization, etc.

Used Functions

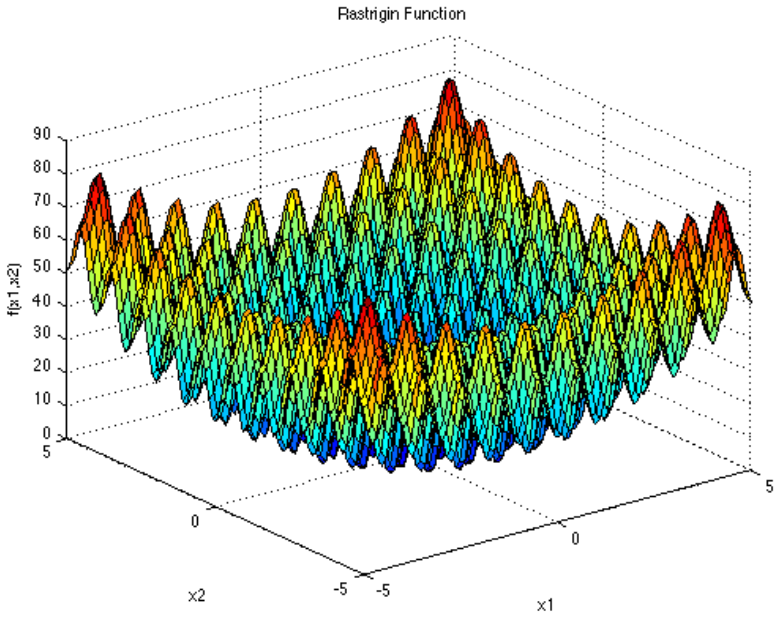
a) De Jong's Function: $f(x) = \sum_{i=1}^{n}(x_i^2)$, $x_i \in [-5.12, 5.12]$;



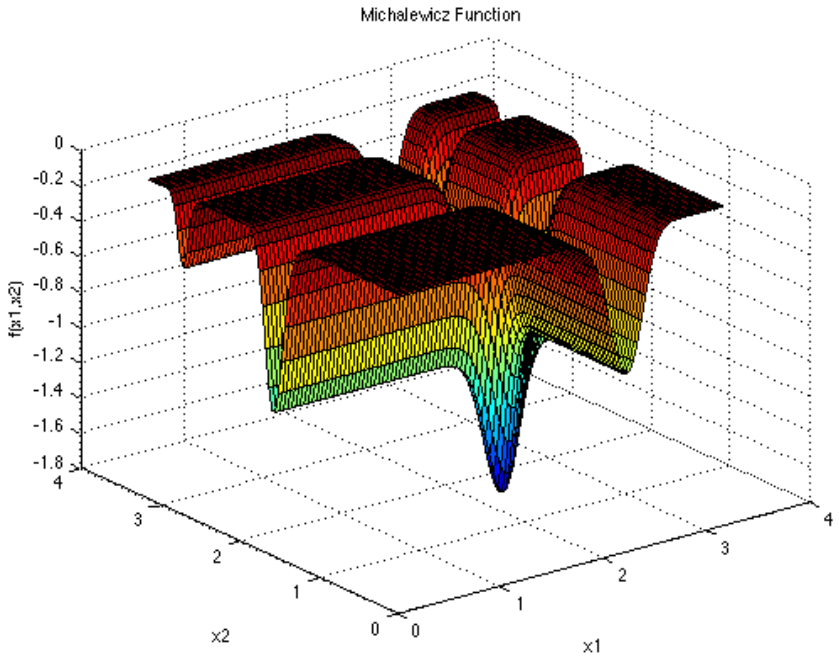
b) Schwefel's Function: $f(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{|x_i|})$, $x_i \in [-500, 500]$;



c) Rastrigin's Function: $f(x) = 10n + \sum_{i=1}^{n}\left(x_i^2 - 10\cos(2\pi x_i)\right)$, $x_i \in [-5.12, 5.12]$;



d) Michalewicz's Function: $f(x) = -\sum_{i=1}^{n}\sin(x_i)\sin^{2m}\left(\frac{ix_i^2}{\pi}\right)$, $x_i \in [0, \pi]$, $m = 10$;



4. Experimental setup:

Each function was tested with 5, 10 and 30 dimensions for reach type of algorythm.

The precision used was 5 decimals for the integers, so that the results get close enough to the global minimum of the functions. For a good precision i choose the population to have 100 individuals, but for good diversity in evolution i choose the mutation rate to be 0.05. The algorithm checks for each population the best score and if the population minimum value found does not overcome the best minimum overall for 5000 generations it will stop.

The results presented bellow consist of average values for both time and function result minimum. Each cell contains the average function minimum found in the time written in seconds:

Dimension\Function	De Jong	Schwefel	Rastrigin	Michalewicz
5	0.00000 / 16.5421s	-2092.53 / 18.8238s	1.93849 / 15.2012s	-4.52241 / 10.295s
10	0.05783 / 43.005s	-3887.8 / 50.066s	9.48716 / 40.151	-9.13243 / 39.504s
30	0.07135 / 174.67s	-9939.38 / 161.308s	45.0798 / 133.061s	-20.6089 / 97.7012s

5. Conclusion:

From all the data acquired, we can conclude that all the algorithms have a different behaviour for each function and different dimensions. We can observe that the dimension of input affects the time of running and the results overall, but not as much as it affetcted Simulated Annealing and Hill Climbing Algorithms. While HCBI seems to get the best results, it takes a lot of time to compute each result comparing to Genetic Algorithms that perform well with far less time. BCWI also takes a lot more time to compute the data, but does not come with a better solution than HCBI, so this approach is worse than Genetic Algorithms. HCFI seems to be much faster than the other HC algorithms, but not as fast as Genetic Algorithms with far worse results. SA is in the middle, running for a small amount of time and getting close enough results to the global minimum, but overall Genetic Algorithms are still better at both time and accuracy. Considering all the observation taken, we can deduce that Genetic Algorithms outperform all other types of algorithms studied previously.

6. Bibliography:

A) De Jong's function: http://www.geatbx.com/docu/fcnindex-01.html#P89_3085

B) Schwefel's function: http://www.geatbx.com/docu/fcnindex-01.html#P150_6749

C) Rastrigin's function: http://www.geatbx.com/docu/fcnindex-01.html#P140_6155

D) Michalewicz's function: http://www.geatbx.com/docu/fcnindex-01.html#P204_10395

E) De Jong graph: https://www.researchgate.net/figure/llustrates-the-2D-Dejong-Dejong-2D-and-2D-Ackley-Ackley-2D-functions-The-selected_fig4_285729352

F) Schwefel graph: https://www.sfu.ca/~ssurjano/schwef.png

G) Rastrigin graph: https://www.sfu.ca/~ssurjano/rastr.png

H) Michalewicz graph: https://www.sfu.ca/~ssurjano/michal.png

I) Genetic Algorithm definition: https://en.wikipedia.org/wiki/Genetic_algorithm