World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:4, No:2, 2010

Fractal Dimension: An Index to Quantify Parameters in Genetic Algorithms

Mahmoud R. Shaghaghian

***Abstract*—**Genetic Algorithms (GAs) are direct searching methods which require little information from design space. This characteristic beside robustness of these algorithms makes them to be very popular in recent decades. On the other hand, while this method is employed, there is no guarantee to achieve optimum results. This obliged designer to run such algorithms more than one time to achieve more reliable results. There are many attempts to modify the algorithms to make them more efficient. In this paper, by application of fractal dimension (particularly, Box Counting Method), the complexity of design space are established for determination of mutation and crossover probabilities (P_m and P_c). This methodology is followed by a numerical example for more clarification. It is concluded that this modification will improve efficiency of GAs and make them to bring about more reliable results especially for design space with higher fractal dimensions.

***Keywords*—**Genetic Algorithm, Fractal Dimension, Box Counting Method, Weierstrass-Mandelbrot function.

I. INTRODUCTION

ASSOCIATED with a typical design problem, nowadays, many different techniques are used for optimizing the design space. Most of these techniques can be classified under either calculus-based techniques or direct search methods. However, the calculus-based methods are deficient in robustness over the broad spectrum of optimization function that arises in engineering optimization. In recent years, the direct-search techniques, which are problem-independent, have been proposed as an elixir for the difficulties associated with the traditional techniques. One of these techniques is genetic algorithms (GAs) which are global search and optimization methods that mimic natural biological evolution.

Genetic Algorithms were, firstly, invented by John Holland in the early 1970 [1]. Subsequent to him, many researches tried to improve the technique to obtain more reliable results [2]-[5]. These modifications are usually due to choice of effective parameters for genetic operations for a particular problem. For example a certain crossover structure may be proposed for specific situations in a problem to make a more efficient genetic algorithm. The efficiency consists of reducing CPU time and increasing accuracy of results.

Fractals are objects which have similar appearances when viewed at different scales. Such objects have details at arbitrarily small scales, making them too complex to be represented by Euclidian space; hence, they are assigned a non-integer dimension. Generally, continuous and non-differentiable functions have been represented as a curve with fractal dimensions between one and two. There are different ways to define fractal dimension, most being equivalent in the continuous domain. However, when applied in practice to discrete data sets, different ways lead to different results.

In this study it is tried to modify a Genetic Algorithm by implication of concept of fractal dimension. This dimension depicts complexity of design space and adapts probability of mutation and crossover for each generation. The methodology is described in the subsequent section and followed by a numerical example for more clarification. Some concluding remarks are presented at end of the paper.

M. R. Shaghaghian is with the Islamic Azad University – Fars Science and Research Branch, Iran (phone: +(98) 9171129009; fax: +(98) 7284692110; e-mail: shaghaghian@gmail.com).

II. METHODOLOGY

To attain the purpose of this paper the following methodology was employed. In the Genetic Algorithm a population of chromosomes initialized and fitness values associated with each chromosome are evaluated. In this stage, due to Euclidian locations of points which are represented by chromosomes and their fitness values, fractal dimension is calculated. Moreover, Creation of new chromosomes and removing some members for further generations will be performed iteratively by applying several operations using parameters obtained by employing the fractal dimension. In other words, parameters, such as mutation and cross over probability, are determined dynamically for each generation by aid of fractal dimension. This procedure may continue until termination criterion is encountered.

Abrupt variations of fitness value for neighbor points in space indicate that more random search to obtain optimum value is required. On other hand, while a gradual variation of fitness value is recognized, a more coherent search is required for this purpose (Fig. 1). Since increasing mutation probability intends to require more random search to optimize function, the curve with more abrupt variation requires higher mutation probability. Fractal dimension can represent abruptness of a curve. Therefore, mutation probability may be determined as follow for next generation:

$$P_m^{i+1} = D^i - I \tag{1}$$

where P_m^{i+1} is probability of mutation to construct (i+1)th generation, I is dimension of design space (I=1, 2, …, n), and D_i is fractal dimension of design space, including fitness values, using all previous generations.

Appropriate searching algorithms look homogeneously for optimum solution. Fractal dimension can be utilized to control homogeneousness of searching. While populations in a generation scattered inhomogeneously, in a GA, crossover operation may manage production of new offspring and make them spread more homogeneously in space. Hence, increase of fractal dimension may cause to reduce crossover probability. Consequently, crossover probability may be determined as follow for next generation:

$$P_c^{i+1} = I - D^i - 1 \tag{2}$$

where P_c^{i+1} is probability of crossover to construct (i+1)th generation, I is dimension of design space, and D_i is fractal dimension of design space, without fitness values, using all previous generations.

There are many methods available for estimating the fractal dimension of data sets. Among these methods, Box Counting Method (BCM) is intuitive, easy to employ, and applicable to sets in any dimension [6]. In fact, it has been applied on images of almost everything, from river systems to the cluster of galaxies. The methodology may be explained on a fractal curve; a curve of infinite detail by virtue of its self-similarity. The length of the curve is indefinite, increasing as the resolution of the measuring instrument increases. By covering the curve with boxes of length lb, as illustrated in Fig. 2, BC method gives the fractal dimension as follows:

$$D = -\lim \frac{\log N_b(l_b)}{\log l_b} \tag{3}$$

where $N_b(l_b)$ is the number of boxes needed to completely cover the curve and D_b corresponds to the slope of log $N_b(l_b)$ versus log l_b plot. Equation (3) holds over a finite range o box-sizes; the smallest boxes having a span of x, where x is the resolution of independent variable, and a height of a, where a is the resolution of the measurement ($X(x)$ in Fig. 2).

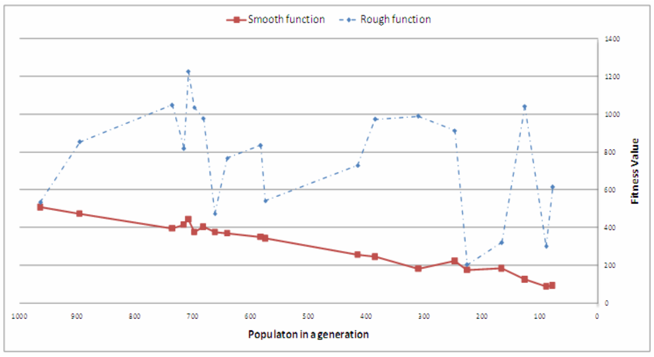

Fig. 1 A typical generation consists 20 populations

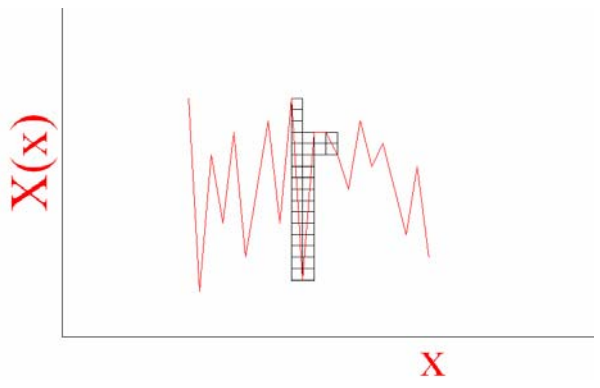

Fig. 2 Box-counting methodology for a typical time series record

Applying the proposed methodology helps not to select the values for probabilities of mutation and crossover blindly. It improves efficiency of both operations to achieve quicker and more accurate results. A numerical example in the subsequent section may clarify the method for application and indicates advantageous of modifications in comparison with an ordinary genetic algorithm in which the parameters selected randomly and improved after a complete run.

III. Numerical Example

In order to clarify the method proposed in the previous section, a function with known fractal dimension is used for optimization. This function is Weierstrass-Madelbrot fractal function which is introduced as follow:

$$W(x) = \sum_{n=-\infty}^{n=\infty} \frac{\left(1 - \cos\left(b^n x\right)\right)}{b^{(2-D)n}} \tag{4}$$

While fractal dimension is known, probabilities of mutation would be preset and constant during a complete run. But, usually this value is unknown and has to be determined and improved during a run. Therefore, P_m will not constant in a run. For this example, since there is one independent variable, I is equal to 1. Implementing Equation 1, P_m becomes D - 1. However, P_c is not constant and has to be determined during the run.

Figs. 3 to 5 show evolution of generations with regard to different fractal dimensions and two types of GAs. In ordinary GA, P_m considered to be 0.05 and P_c is 0.9. Looking upon these figures, while the overall fractal dimension of the function is low (near unity), both ordinary and modified GA may achieve optimal value in reasonable period (Fig. 3). In contrary, when fractal dimension increases, the modified GA leads to not only accurate results but also improves CPU time of the run.

IV. Conclusion

Ordinary GAs usually captured in local optima. Mutation is a good way to escape these local optima in all GAs. However, increasing the probability of mutation will destroy logics behind searching procedure. Furthermore, crossover is a powerful tool to create new offspring and search through a design space. Increasing the probability of crossover may lead to bypass optima and increase period of searching. Conversely, low probability of crossover restricts and even terminates searching without achieving proper results. A suitable value for these parameters will accelerate reaching to more reliable results.

Fractal dimension is a good way to give an overview to design space for selecting proper parameters in a GA. In this research, it is concluded that design spaces with lower fractal dimension are more robust with their crossover and mutation probabilities. In addition, for high fractal dimensions it is recommended to consider higher mutation probability and low crossover probability. A simple equation proposed to indicate relation between these values.

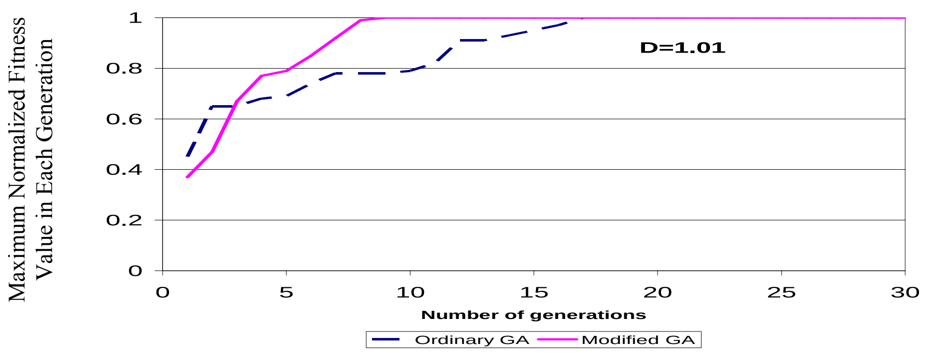

Fig. 3 Comparison Ordinary and Modified GA for Weierstrass-Mandelbrot function with D=1.01

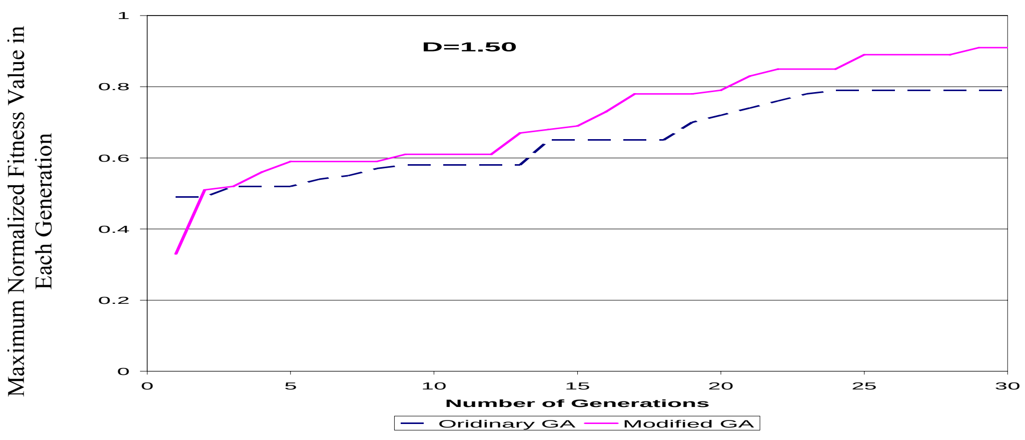

Fig. 4 Comparison Ordinary and Modified GA for Weierstrass-Mandelbrot function with D=1.50

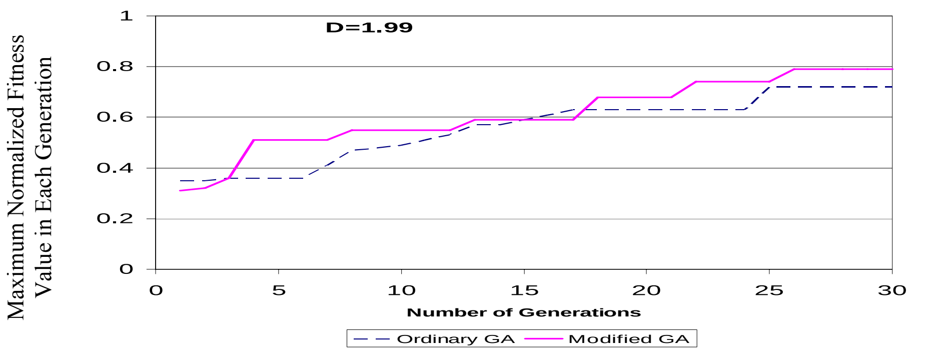

Fig. 5 Comparison Ordinary and Modified GA for Weierstrass-Mandelbrot function with D=1.99

REFERENCES

[1] J. H. Holland, (1975), "Adaptation in natural and artificial systems", Ann Arbor: The University of Michigan Press, 1975.
[2] J. A. Vasconcelos, J. A. Ramírez, R. H. C. Takahashi., and R. R. Saldanha, (2001), "Improvements in Genetic Algorithms", IEEE Transactions on Magnetics, 37(5), 2001, 3414-3417.
[3] J. Andre, P.Siarri, and T.Dognon, "An improvement of the standard genetic algorithm fighting premature convergence in continuous optimization", Advances in Engineering Software, 32(1), 2001, 49-60.
[4] O.Hrstka, and A.Kučerová, "Improvements of real coded genetic algorithms based on differential operators preventing premature convergence", Advances in Engineering Software, 35(3-4), 2004, 237-246.
[5] Z.Ye, Z.Li, and M.Xie, "Some improvements on adaptive genetic algorithms for reliability-related applications", Reliability Engineering & System Safety, 2009, Article in Press.
[6] G. R. Rakhshandehroo, M. R.Shaghaghian, A. R.Keshavarzi, and N. Talebbeydokhti, "Temporal variation of velocity components in a turbulent open channel flow: Identification of fractal dimensions", Applied Mathematical Modeling, 33, 2009, 3815-3824.