IJCCCE Vol.14, No.1, 2014

Investigating the Guidance Feature of Searching in the Genetic Algorithm

Dr. Laith Jasim Saud[1] and Dr. Mohamed Jasim Mohamed[2]

[1]Control and Systems Eng. Dept, University of Technology, Baghdad

[2]Control and Systems Eng. Dept, University of Technology, Baghdad

e-mail: laith_js@yahoo.com, moh62moh@yahoo.com

Received: 18/6/2012

Accepted: 1/4/2013

Abstract: There is an argument about the optimization capability of the Genetic Algorithm (GA) and whether its approach of search is guided or random. Although it has its own criticism, Schema theorem was the main effort for explaining the way a GA works, and justifying that the GA search is guided. After that, Schema theorem efforts continued for analyzing and justifying GA optimization approach with no complete analytic proof so far.

In this paper a different approach, namely a statistical approach, is used to test and justify guidance feature in the Genetic Algorithm when it is used for optimization purposes. Different standard functions have been tried and different tests have been done, and the results obtained proved that the GA is a guided search method and not random at all. Moreover, the tests carried out and results obtained proved the importance and necessity of each of the operators or techniques used by the GA.

Keywords – Genetic Algorithms, Genetic Algorithms criticisms, Optimization algorithms.

1. Introduction

Genetic Algorithms (GAs) are a particular class of evolutionary computation algorithms which all share common techniques like inheritance, mutation, selection, and crossover (also called recombination) [1,2]. The Genetic Algorithm (GA) was initially developed, based on the mechanics of biological evolution, by John Holland at the University of Michigan (1970's), motivated by the desire to understand processes in natural systems and to design artificial systems retaining the robustness and adaptation properties of natural systems [3]. Holland's original GA is known as the simple genetic algorithm (SGA). Later on, many developments have been made to the algorithm but with the same core principle, providing efficient techniques for optimization and machine learning applications widely used in business, science and engineering. Gas, in general, is widely used for different applications in business, science and engineering. Among these applications are [4]:

- Optimization– numerical and combinatorial optimization problems.
- Robotics – trajectory planning.
- Machine learning– designing neural networks, classification and prediction.
- Signal processing– filter design.
- Design– semiconductor layout, aircraft design, and communication networks.
- Automatic programming– evolve computer programs for specific tasks, design cellular automata and sorting networks.
- Economic– development of bidding strategies, and emergence of economics markets.
- Immune systems–model somatic mutations.
- Ecology– model symbiosis and resource flow.
- Population genetics.

2. GAs and optimization problems

Physics, biology, economy, sociology or engineering, often have to deal with the classical problem of optimization. Generally, optimization methods can be divided into derivative and non-derivative, or non-gradient, methods as shown in Figure (1) [5]. GAs belong to the non-derivative class of optimization methods.

Purely analytical derivative methods, which depend on functions differentiability, widely proved their efficiency in dealing with optimization problems. They nevertheless suffer from an insurmountable weakness, as reality rarely obeys to those wonderful differentiable functions which analytic methods need. Non-derivative, or non-gradient, methods are more suitable for general engineering design problems.

One reason is that engineering design problems often consist of a mixture of numerical simulations, analytical calculations and other non-mathematically analytic formal kind of information, for which there is no easy way of calculating derivatives of the objectives function. These methods are also known as black box methods as they do not require any derivatives of the objective function in order to calculate the optimum. Another advantage of these methods is that they are more likely to find a global optimal, and not be stuck on local optima as gradient methods might do [5], [6].

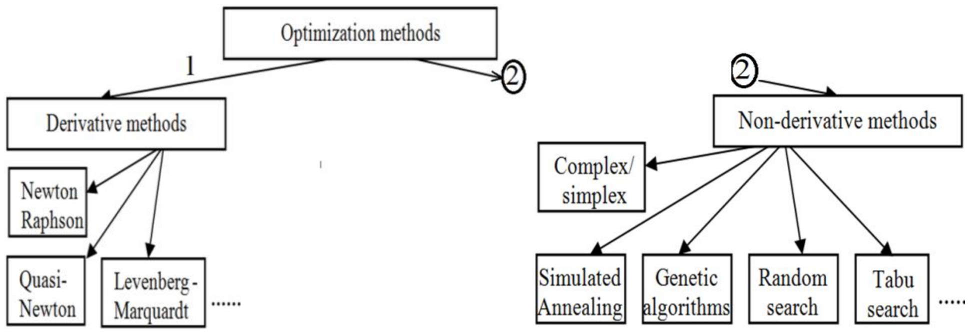


Figure (1): Some optimization methods

Sometimes it is possible to start the optimization procedure with the non-derivative methods to explore the objective space and identify the promising regions, and then switch to other optimization techniques in order to refine the search if needed. There are different ways of developing hybrids among the two optimization approaches [5], [7], [8].

3. GA Operators and Algorithm

The operators represent the main part of the basic genetic algorithm. The GA has five main operators which are: encoding, selection, crossover, mutation, and reinsertion. There are no exact rules for dealing with the operators but there are many guidance rules. The operators depend to a large extent on the problem context. Figure (2) shows a typical basic GA algorithm flowchart [9].

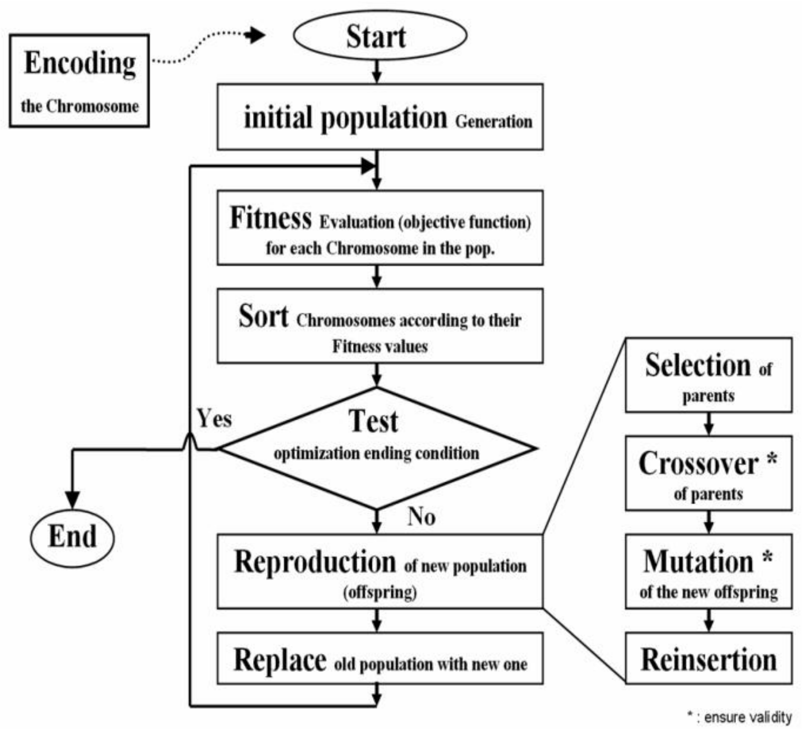

Encoding the Chromosome
Start
initial population Generation
Fitness Evaluation (objective function) for each Chromosome in the pop.
Sort Chromosomes according to their Fitness values
Yes
Test optimization ending condition
No
End
Reproduction of new population (offspring)
Replace old population with new one
Selection of parents
Crossover * of parents
Mutation * of the new offspring
Reinsertion
* : ensure validity

Figure (2) A typical basic GA algorithm flowchart

4. GA features as an optimization method

There is no opt method with all advantages and no disadvantages. The best method to use depends to a large extent on the problem context. As much as it regards the GA as an optimization method, GA is useful for problems with the following features [4], [9], [10]:

- The problem under consideration is combinatorial.
- The objective function is discontinuous, highly nonlinear, stochastic, or has unreliable or undefined derivatives. Of course it is capable of supporting multi-objective optimization problems.
- The presence of vast number of possible discrete solutions, i.e., large search spaces.
- The presence of local optimal solutions.
- The presence of noisy environment.
- The system under consideration is complex, time-varying, and highly nonlinear.

One other point counted to the GA is that it can use a different fitness function from iteration to iteration, which allows incorporating new data in the model if it becomes available.

5. Criticisms to GA searching capability

The question that most people who are new to the field of genetic algorithms ask at this point is why such a process should do anything useful. Why should one believe that this is going to result in an effective form of search or optimization? The first answer comes from Holland through the "Schemata Theory (ST)" which he formalized its framework and which was later popularized by Goldberg [3], [11], [12]. The ST represents an attempt to explain how the GA works and defend its search capability. The theory is based on modeling the search space as hyper-cubes in which each vertex represents a possible solution. The ST has many weak points [11], [13]. The "Exact Schemata Theory" which came after the ST, represents an attempt to overcome some of the weakness points in the ST and it managed to do so [13]. But, still there is no complete answer or proof for each point of concern in the GA as an optimization approach. Just as an example, no theory so far can tell us how long we can expect to wait for a GA to converge [13]. Generally, GA convergence has been a main issue to many researchers [14] – [20]. And despite the fact that GA has been applied to many difficult optimization problems, many criticism points have been mentioned about it and most of them are about its convergence, like:

- It is non-trivial to analyze its convergence property [16,18].
- Its convergence to the optimal solution could never be achieved without maintaining the best solution in the population, either before or after selection process [15,20].

Other than the argument above, and due to the nature of the GA architecture, and due to the great deal of randomness and assumptions in choosing, defining or deciding matters related to the many GA operators, and due to the lack of a complete analytic proof which settles the argument, many argue that the GA is a random search approach and not a guided one.

In this paper we are concerned of

settling a certain specific point, namely: is the GA a random or a guided approach of search?

6. Test functions

The approach adopted in this paper to judge if the GA is a guided search or just a random search is a statistical approach. In this approach, the GA and a random search method will be used to find the optimal value for five well known standard test functions [21].Then, a comparison and a judgment will be made according to the obtained results. The test functions used are given in the following sections.

6.1. Rotated hyper-ellipsoid function

A sample for this function is shown in figure (3).

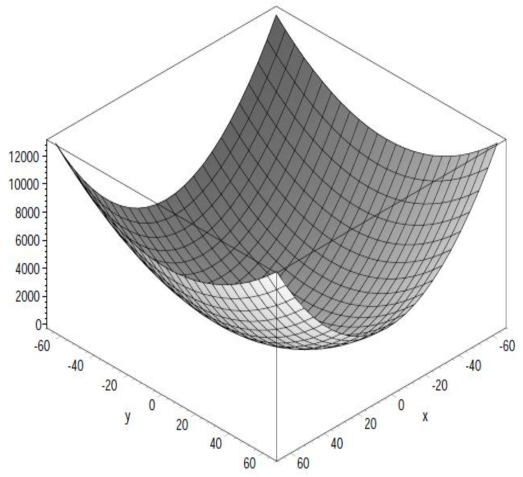


Figure (3): A sample Rotated hyper-ellipsoid function.

With respect to the coordinate axes, this function produces rotated hyper-ellipsoids. It is continuous, convex and unimodal. The function has the following general definition:

$$f(x)=\sum_{i=1}^{n}\sum_{j=1}^{i}x_j^2 \qquad (1)$$

Test area is usually restricted to hypercube -65.536 $\leq$ x_i $\leq$ 65.536, i=1,2,...…,n.

Its global minimum equal $f(x)$ = 0 is obtainable for, x_i= 0, i = 1,2,…..,n.

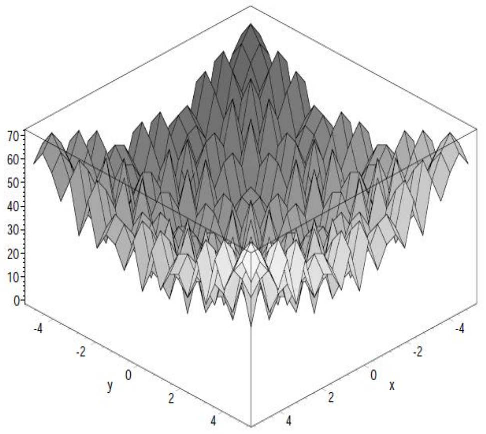
6.2. Rastrigin's function

Rastrigin's function, for which a sample is shown in figure (4), is based on the function of De Jong with the addition of cosine modulation in order to produce frequent local minima. Thus, the test function is highly multimodal. However, the location of the minima is regularly distributed. The function has the following definition:

$$f(x)= 10n+\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)] \qquad (2)$$

Test area is usually restricted to hypercube -5.12$\leq$ x_i $\leq$5.12, x_i= 0, i=1,2,...…,n.

Its global minimum equal $f(x)$ = 0 is obtainable for, i=1,2,…..,n.



Figure(4) : A sample Rastrigin's function.

6.3.Schwefel's function

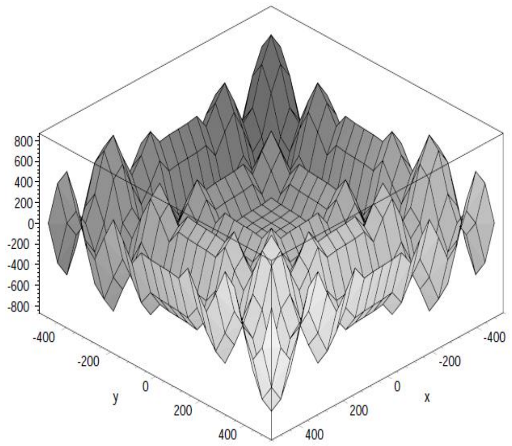
Schwefel's function, for which a sample is shown in figure (5), is deceptive in that the global minimum is geometrically distant, over the parameter space, from the next best local minima.

Therefore, the search algorithms are potentially prone to convergence in the wrong direction. This function has the following definition:

$$f(x)=\sum_{i=1}^{n}[-x_i \sin(\sqrt{|x_i|}] \qquad (3)$$

Test area is usually restricted to hyphercube -500 ≤ x_i≤ 500, *i*=1,2,......,*n*..

Its global minimum *f*(*x*) = (418:9829**n*) is obtainable for *xi* = 420.9687, *i*=1,2,......,*n*.



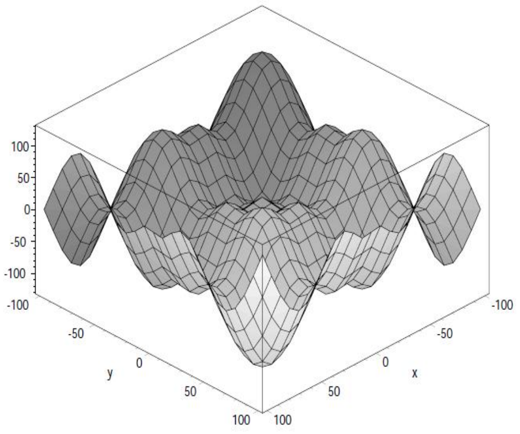
Figure(5) : A sample Schwefel's function.

6.4. Michalewicz's function

The Michalewicz function is a multimodal test function (owns *n*! local optima). Figure (6) shows a sample of this function. The parameter *m* defines the "steepness" of the valleys or edges. Larger *m* leads to more difficult search. For very large *m* the function behaves like a needle in the haystack (the function values for points in the space outside the narrow peaksgive very little information on the location of the global optimum). This function has the following definition.

$$f(x)=-\sum_{i=1}^{n}\sin(x_i).[\sin(\frac{ix_i^2}{\pi}]^{2m} \qquad (4)$$

It is usually set *m* = 10. Test area is usually restricted to hyphercube 0 ≤ xi ≤ π.*i*=1,2,…,*n*.The global minimum value has been approximated by *f*(*x*) =4.687 for *n* = 5 and by *f*(*x*) =9.66 for *n* = 10. Respective optimal solutions are not given.



Figure(6) : A sample Michalewicz's function.

6.5. Deceptive functions

A deceptive problem is a class of problems in which the total size of the basins for local optimum solutions is much larger than the basin size of the global optimum solution. Clearly, this is a multimodal function. The general form of deceptive function is given by the following formulae:

$$f(x) = -[\frac{1}{n}\sum_{i=1}^{n}g_i(x_i)]^{\beta} \qquad (5)$$

where β is a fixed non-linearity factor.

The deceptive function f(x) is classified into the following three types, depending on the location of a global optimum in n-dimensional space [22].

Type I is a simple deceptive problem:

$$g_i(x) = \begin{cases} \alpha - x & if\ 0 \le x \le 0.8 \\ \frac{x-\alpha}{1-\alpha} & if\ 0.8 < x \le 1 \end{cases} \qquad (6)$$

where α = 0.8.

Type II is a medium-complex deceptive problem:

$$g_i(x) = \begin{cases} \alpha - x & if\ 0 \le x \le 0.8 \\ \frac{x-\alpha}{1-\alpha} & if\ 0.8 < x \le 1 \end{cases} \qquad (7)$$

Or;

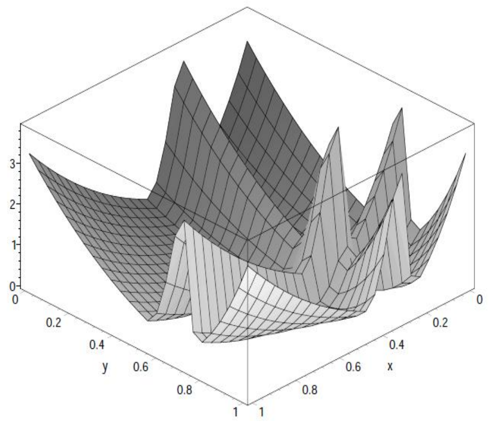
$$g_i(x) = \begin{cases} \frac{1-\alpha-x}{1-\alpha} & if\ 0 \le x \le 0.8 \\ x - 1 + \alpha & if\ 0.8 < x \le 1 \end{cases} \qquad (8)$$

Randomly chosen according to each dimension i (i =0, 1, ··· ,n), where α = 0.8.

Type III is a complex deceptive problem:

$$g_i(x) = \begin{cases} -\frac{x}{\alpha_i} + \frac{4}{5} & if\ 0 \le x < \frac{4}{5}\alpha_i \\ \frac{5x}{\alpha_i} - 4 & if\ \frac{4}{5}\alpha_i \le x \le \alpha_i \\ \frac{5(x-\alpha_i)}{\alpha_i - 1} + 1 & if\ \alpha_i \le x \le \frac{1+4\alpha_i}{5} \\ \frac{x-1}{1-\alpha_i} + \frac{4}{5} & if\ \frac{1+4\alpha_i}{5} < x \le 1 \end{cases} \qquad (9)$$

where α_i is a different random number between 0 and 1 depending on each dimension i (i =0, 1, ··· ,n).



Figure(7) : A sample Deceptive function

Types I and II are special cases of the complex deceptive problem type III. Clearly formulae (9) should be suitably adjusted to get type I and II.

For all three types of $g_i(x_i)$, the region with local optima is 5^n-1 times larger than the region with a global optimum in the *n*-dimensional space. The number of local optima is 2^n-1 for Type I and Type II deceptive problems and 3^n-1 for Type III.

For our example β=1 and;

x[30] = [0.2, 0.4, 0.7, 0.9, 0.1, 0.3, 0.5, 0.7, 0.9, 0.8, 0.6, 0.4, 0.2, 0.5, 0.8, 0.1, 0.6, 0.9, 0.8, 0.4, 0.7, 0.5, 0.3, 0.4, 0.1, 0.9, 0.5, 0.8, 0.4, 0.3], Optimal f(x) = -1.

7. The proposed GA procedure

The procedure steps for the GA which is proposed here to solve the optimization problem considering the chosen test functions are listed below. The flowchart for the proposed GA is given in figure (8) .

Step-1: Given a test function $f(X_i)$, determine the number of parameters to be optimized in the function. Where, X_i is a vector of parameters (i=1,2,..n- n represents the number of parameters).

Step-2: Specify the lower and upper bounds (range of real values) of the parameters.

Step-3: Since, an integer code is used in the proposed GA for every parameter in the chromosome; we need to determine the upper value of each integer to get the real value with a certain resolution. For example, if the range of the real value of a certain parameter is -65.536<=xi<=65.536 and the required resolution is 0.001, then the total integer instances needed to represent this parameter in the chromosome are equal to (65.536-(-65.536))/0.001=131072. To determine the real value of each parameter from the integer code, we use the following rule.For example, assume that the integer code of a certain parameter is 7349, then, to find the real value for this parameter, we apply;

Real value= Integer of parameter*(Max real value-Min real value)/131072+Min real value

Real value = 7349*(65.537-(-65.537)/131072-65.537=7349/1000-65.537=7.349-65.537=-58.188

Step-4: Construct the chromosome of the underlying optimized function, where each parameter is represented by one gene within the integer code.

Step-5: Determine the type of the selection method and the type of crossover and mutation operators.

Step-6: Set up GA parameters: Crossover probability Pc, Mutation probability Pm, Population size, and Maximum number of generation.

Step-7: - Randomly create the initial population of the GA (this population is later called old population). This population is composed of a group of individuals or chromosomes each of which is denoted by X_{ij} ,(i=1,2...n) (j=1,2...m),where: n is the number of parameters, m is number of population size. - Set the population generation of GA to g=1.

Step-8: For each individual of the current GA population (old population), compute the value of the function $f(X_i)$, then calculate the Darwinian fitness value for this individual as shown below.

$$\text{Fitness}(X_i) = \frac{1}{f(X_i)}$$

Step-9: Make the individuals of the current population in descending order according to the fitness value, and then get the best one as a solution to the underlying function.

Step-10: Copy a certain number of best individuals of current population to a new place which is called new population, this operation is called elitism operation.

Step-11: Select individuals from the current population using the selection method predefined in step-5, and apply genetic operations on them (crossover and mutation) to produce new children (offspring).

Step-12: The resultant children from the above step are put in the new population till new population is filled by children.

Step-13: Clear all of the individuals in the old population, which represent the parents, and transfer the individuals of the new population which represent the children (offspring) to take the place of the old population.

Step-14: Empty the space of the new population from the children to make it ready for the next generation.

Step-15: If the maximum number of generations g of GA is achieved, go to step-16, otherwise set g=g+1 and go to step-8, where g denotes the number of generations for GA.

Step-16: Print the last best individual which represents the solution of the function then stop.

The parts composing the proposed GA are explained below:

P1: This part is used to point out to the selected gene to be mutated based on mutation probability.

P2: This part is used to count the number of mutation operation executions in each run for the purpose of statistic.

P3: This part is used to determine the probability of execution for the first type and the second type of the mutation operation. The value 0.5 used here indicates that each type have a probability of 50%.

P4: This part represents the first type of the proposed mutation operation which is useful for searching all the space of the parameters.

P5: This part represents the second type of the proposed mutation operation which is useful for the fine tuning of the parameters.

P6: This part prevents the parameters from exceeding the required range of the parameters.

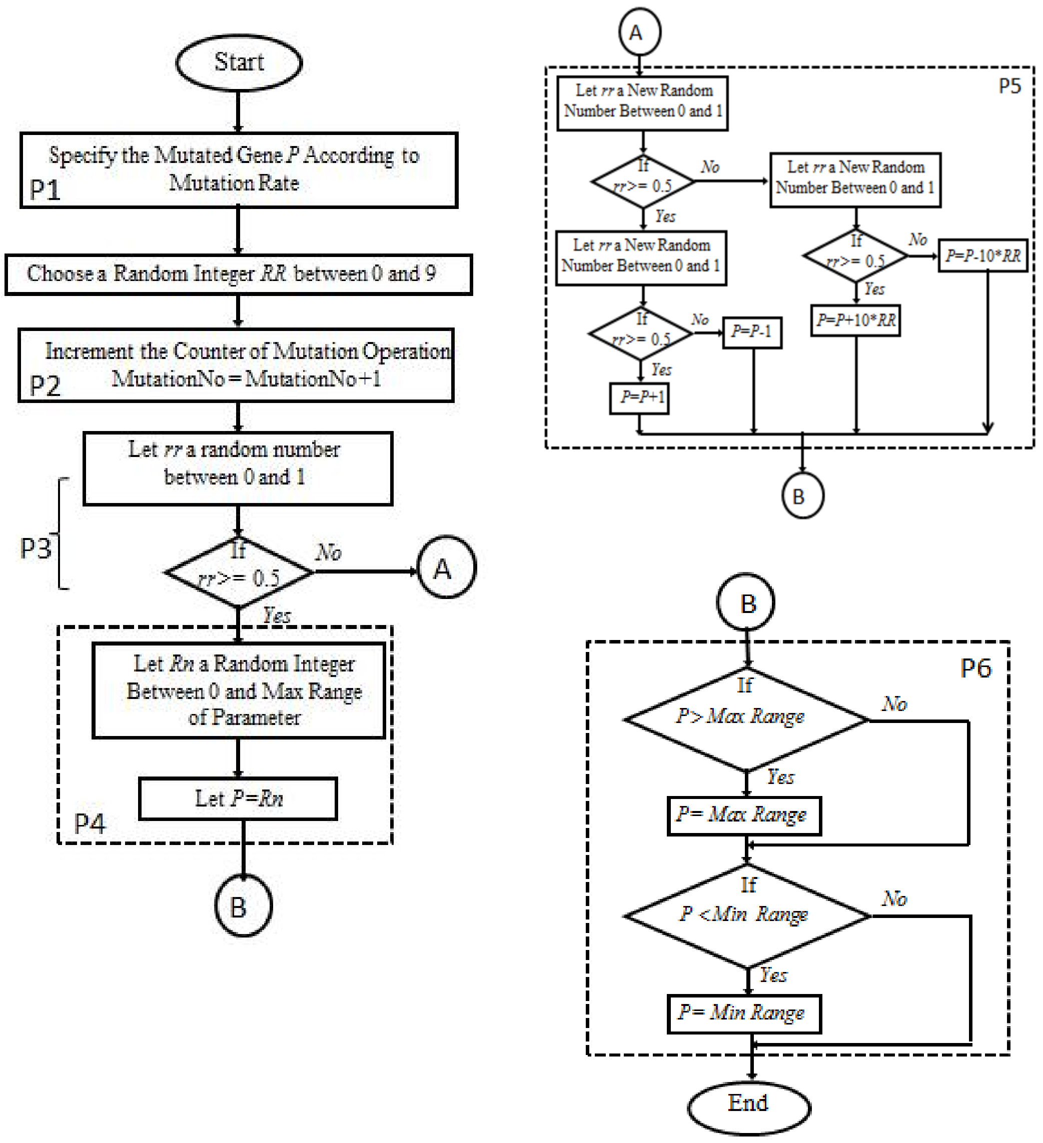

Start
Specify the Mutated Gene P According to Mutation Rate
P1
Choose a Random Integer RR between 0 and 9
Increment the Counter of Mutation Operation MutationNo=MutationNo+1
P2
Let rr a random number between 0 and 1
P3
If rr>= 0.5
No
Yes
A
Let Rn a Random Integer Between 0 and Max Range of Parameter
Let P=Rn
P4
B
A
P5
Let rr a New Random Number Between 0 and 1
If rr>= 0.5
No
Yes
Let rr a New Random Number Between 0 and 1
Let rr a New Random Number Between 0 and 1
If rr>= 0.5
No
Yes
P=P-10*RR
P=P+10*RR
If rr>= 0.5
No
Yes
P=P-1
P=P+1
B
B
P6
If P> Max Range
No
Yes
P= Max Range
If P <Min Range
No
Yes
P= Min Range
End

Figure (8): The flowchart of theproposed GA.

8. Results

Tables (1) to (5) show the results obtained for each of the five test functions. Four each test function, the random search has been tried as well as four cases for the GA search. Two main purposes were aimed at from these tests.

The first is to compare the random search with the GA search, and the second is to check and justify the importance of the main GA operators. The results clearly show that the GA is not a random search and that it really has guidance for two reasons, the first one is that it never fails to find the optimal, and the second is that it always requires a number of search iterations (generations) far less than the random search. The results also reflected the importance of the GA operators, as ignoring one of them reflects very badly on the search process and hence finding the optimal solution.

The GA settings for all of the cases considered were as follows:

- Creation operation is used with 10 individuals that are created for each next generation.
- Elitism strategy is used. 4 best individuals are retained.
- K-Tournament selection is used with k=0.75
- Two point crossover with Pc =0.85
- Mutation with two modes and with probability Pm =0.05
- Population size =100.

Two types of operators are applied to the chromosome that is selected to undergo the mutation operation. For the first operator, the integer of the mutated gene is replaced by new integer choosing randomly from the full integer range of that gene. The second operator adds a random displacement chosen randomly from the following cases ±1, ±10xr (where, r is a random integer number in the range 1-9) to the current integer of the gene that is undergoing the mutation operation. The first operator lets the GA explore all integer space while the second operator does a fine tuning to the suboptimal solutions.

9. Conclusions

A statistical approach was adopted in this paper to compare the performance of the GA algorithm as an optimization method and the random search method. The aim of the comparison was to show definitely that the GA is a guided search and not some sort of random search. The approach was based on trying the GA and the random search on five standard test functions and then comparing the GA and the random search performances. The results obtained stated with no doubt that the GA, unlike the random search, is really a guided search as for all the cases it reached the final optimal solution with an accepted number of iterations (generations).

Moreover, to check the effectiveness of the GA operators in guiding the search, tests have been carried out with and without the mutation, crossover, and biased selection. The results obtained showed and verified the role of these operators to guide the search to the optimal solution with failure possibility of zero.

Table 1. Results for rotated hyperellipsoid function.

Search Type	Dimension of Function	Range of Parameters in Search Space	Integer Code for Every Parameters	Number of Implemented Runs & max Generations	Number of Failed Runs	Number of Succeeded Runs	Maximum Generations for Succeeded Runs	Minimum Generations for Succeeded Runs	The Average Generations for All Succeeded Runs
Random	30	65.536 to -65.536	131072	100 100000	All	0	-	-	-
GA with No Bias Selection	30	65.536 to -65.536	131072	100 100000	All	0	-	-	-
Crossover + Selection only	30	65.536 to -65.536	131072	100 100000	Most near optimum	0	-	-	-
Mutation + Selection only	30	65.536 to -65.536	131072	100 100000	All	0	-	-	-
Full GA	30	65.536 to -65.536	131072	1000 30000	0	All	22929	11469	15495

Table 2.Results for Rastrigin'sfunction

Search Type	Dimension of Function	Range of Parameters in Search Space	Integer Code For Every Parameters	Number of Implemented Runs & max Generations	Number of Failed Runs	Number of Succeeded Runs	Maximum Generations for Succeeded Runs	Minimum Generations for Succeeded Runs	The Average Generations for All Succeeded Runs
Random	30	-5.12 to 5.12	10240	100 100000	All	0	-	-	-
GA with No Bias Selection	30	-5.12 to 5.12	10240	100 100000	All	0	-	-	-
Crossover + Selection only	30	-5.12 to 5.12	10240	100 100000	6	94	97361	48304	706081
Mutation + Selection only	30	-5.12 to 5.12	10240	100 100000	All	0	-	-	-
Full GA	30	-5.12 to 5.12	10240	1000 30000	0	All	20298	5872	9628

Table 3.Results for Schwefel's function.

Search Type	Dimension of Function	Range of Parameters in Search Space	Integer Code For Every Parameters	Number of Implemented Runs& max Generations	Number of Failed Runs	Number of Succeeded Runs	Maximum Generations for Succeeded Runs	Minimum Generations for Succeeded Runs	The Average Generations for All Succeeded Runs
Random	30	-500 to 500	1000000	100 100000	All	0	-	-	-
GA with No Bias Selection	30	-500 to 500	1000000	100 100000	All	0	-	-	-
Crossover + Selection only	30	-500 to 500	1000000	100 100000	Most near optimum	0	-	-	-
Mutation + Selection only	30	-500 to 500	1000000	100 100000	All	0	-	-	-
Full GA	30	-500 to 500	1000000	1000 50000	0	All	37746	18511	24923

Table 4.Results for Michalewicz's function.

Search Type	Dimension of Function	Range of Parameters in Search Space	Integer Code For Every Parameters	Number of Implemented Runs& max Generations	Number of Failed Runs	Number of Succeeded Runs	Maximum Generations for Succeeded Runs	Minimum Generations for Succeeded Runs	The Average Generations for All Succeeded Runs
Random	10	π0 to	3141	100 100000	All	0	-	-	-
GA with No Bias Selection	10	π0 to	3141	100 100000	All	0	-	-	-
Crossover + Selection only	10	π0 to	3141	100 100000	0	All	60784	7088	22202
Mutation + Selection only	10	π0 to	3141	100 100000	All	0	-	-	-
Full GA	10	π0 to	3141	1000 30000	0	All	14830	432	1894

Table 5. Results for Deceptive function

Search Type	Dimension of Function	Range of Parameters in Search Space	Integer Code For Every Parameters	Number of Implemented Runs & max Generations	Number of Failed Runs	Number of Succeeded Runs	Maximum Generations for Succeeded Runs	Minimum Generations for Succeeded Runs	The Average Generations for All Succeeded Runs
Random	30	0 to 1	1000	100 100000	All	0	-	-	-
GA with No Bias Selection	30	0 to 1	1000	100 100000	All	0	-	-	-
Crossover + Selection only	30	0 to 1	1000	100 100000	2	98	97741	23255	54600
Mutation + Selection only	30	0 to 1	1000	100 100000	All	0	-	-	-
Full GA	30	0 to 1	1000	1000 20000	0	All	14037	4019	6996

References

[1] R. Horst and P.M. Pardalos (eds.), “Handbook of Global Optimization”, Kluwer Dordrecht, 1995.

[2] M. Mitchell, “An Introduction to Genetic Algorithms”, Cambridge, MA: MIT Press, 1996.

[3] Holland J.H., “Adaptation in natural and artificial system”, Ann Arbor, The University of Michigan Press, 1975.

[4] Olesya Peshko, “Global Optimization genetic Algorithms”, http://www.cas.mcmaster.ca/~cs777/ presentations/3_GO_Olesya_Genetic_Algorithms.pdf

[5] J. Andersson, “A survey of multi objective optimization in engineering design”, Technical Report: LiTH-IKP-R-1097, Department of Mechanical Engineering, Linkping University, 2000, (www.lania.mx/~ccoello/ andersson00.pdf.gz).

[6] O. T. Sehitoglu and G. Üçoluk, “A Building Block Favoring Reordering Method for Gene Positions in Genetic Algorithms”, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001),San Francisco, USA, 2001.

[7] D. A. Savic, G. A. Walters, and M. Schwab, “Multi objective Genetic Algorithms for Pump Scheduling in Water Supply”, AISB International Workshop on Evolutionary Computing, Berlin, April 1997.

[8] D. Whitley, “An Overview of Evolutionary Algorithms”, Journal of Information and Software Technology, N43, 2001.

[9] L. J. Saud, “Topology Design Optimization Using GA.”, PhD Thesis, Control and Systems Engineering Department, University of Technology, 2006.

[10] M. Angelova, and T. Pencheva, “Tuning Genetic Algorithm Parameters to Improve Convergence Time”, International Journal of Chemical Engineering, 2011.

[11] Darrell Whitley, “A Genetic Algorithm Tutorial”, Colorado State University, USA, c*iteseerx.ist.psu.edu.(pdf file), 1998.*

[12] D.E., Goldberg, “Genetic Algorithms in Search, Optimization and Machine Learning”, Addison Wesley, (1989).

[13] William C. Liles, R. Paul Wiegand, “Introduction to Schema Theory, A survey lecture of pessimistic & exact schema theory”, ECLab, George Mason University.

((cs.gmu.edu/~eclab/papers/lecturepres/schem. .pdf)).

[14] W. Kosiński, S. Kotowski , and Z. Michalewicz, " On convergence and optimality of genetic algorithms", IEEE Congress on Evolutionary Computation, 2010.

[15] G. Rudolph, " Convergence analysis of canonical genetic algorithms", IEEE Transactions on Neural Networks , vol. 5, no. 1, 1994.

[16] F. Lin, C. Zhou , K.C. Chang, " Convergence rate analysis of allied genetic algorithm", 49th IEEE Conference on Decision and Control, 2010.

[17] H. Lv, J. Zheng, J. Wu, C. Zhou, "The Convergence Analysis of Genetic Algorithm Based on Space Mating", Fifth International Conference on Natural Computation, China, 2009.

[18] L. Ming, Y. Wang , Y. Cheung , "On Convergence Rate of a Class of Genetic Algorithms", Automation Congress WAC '06. World, 2006.

[19] B. H´eder and J. Bit ´o, "Convergence Analysis of Genetic Algorithm Applied for Dynamic Optimization of erminal to Base Station Assignment in Satellite Fed BFWA Systems", International Workshop on Satellite and Space Communications, Italy, 2008.

[20] M. Khosraviani, S. Pour-Mozafari, M. M.iEbadzadeh, "Convergence Analysis of Quantum-inspired Genetic Algorithms with the Population of a Single Individual", GECCO'08, USA, 2008.

[21] MarcinMolga, CzesławSmutnicki, 2005, "Test functions for optimization needs", paper, ((www.zsd.ict.pwr.wroc.pl/files/docs/functions .pdf))

[22] H. Suzuki, and H. Sawai, "Chemical Genetic Algorithms - Coevolution between Codes and Code Translation", Proceedings of the 8th International Conference on Artificial Life, USA, 2002.