Genetic Algorithm – an Approach to Solve Global Optimization Problems

PRATIBHA BAJPAI

Amity Institute of Information Technology, Amity University,
Lucknow, Uttar Pradesh, India, pratibha_bajpai@rediffmail.com

DR. MANOJ KUMAR

LBSIMDS,
Lucknow, Uttar Pradesh, India,
mauryamanoj2008@rediffmail.com

Abstract

The genetic algorithm (GA) is a search heuristic that is routinely used to generate useful solutions to optimization and search problems. It generates solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover. Genetic algorithms are one of the best ways to solve a problem for which little is known. They are a very general algorithm and so work well in any search space. All you need to know is what you need the solution to be able to do well, and a genetic algorithm will be able to create a high quality solution.

Keywords: ***Global Optimization; Genetic Algorithms; Rastrigin's function***

1.0 Introduction

Darwin's theory of Evolution states that all life is related and has descended from a common ancestor. The theory presumes that complex creatures have evolved from more simplistic ancestors naturally over time. In a nutshell, as random genetic mutations occur within an organism's genetic code, the beneficial mutations are preserved because they aid survival -- a process known as "natural selection." These beneficial mutations are passed on to the next generation. Over time, beneficial mutations accumulate and the result is an entirely different organism.

Genetic algorithms are adaptive heuristic search algorithm premised on the Darwin's evolutionary ideas of natural selection and genetic. The basic concept of genetic algorithms is designed to simulate processes in natural system necessary for evolution. As such they represent an intelligent exploitation of a random search within a defined search space to solve a problem. First pioneered by John Holland in the 60's, GAs has been widely studied, experimented and applied in many fields in engineering world. Not only does genetic algorithm provide an alternative method to solving problem, it consistently outperforms other traditional methods in most of the problems link. Many of the real world problems which involve finding optimal parameters might prove difficult for traditional methods but are ideal for genetic algorithms. This paper starts with the description of various GA operators in section 2. Section 3 gives the outline of the genetic algorithm. In section 4, we introduce global optimization and discuss how genetic algorithm can be used to achieve global optimization and illustrate the concept with the help of Rastrigin's function. In section 5, we explore the reasons why GA is a good optimization tool. The discussion ends with a conclusion and future trend.

2.0 Genetic Algorithm

A population of individuals is maintained within search space for a GA, each representing a possible solution to a given problem. Each individual is coded as a finite length vector of components, or variables, in terms of some alphabet, usually the binary alphabet {0,1}. To continue the genetic analogy these individuals are likened to chromosomes and the variables are analogous to genes. Thus a chromosome (solution) is composed of several genes (variables).

The chromosome then looks like this:

Chromosome 1	1101100100110110
Chromosome 2	1101111000011110

Each chromosome has one binary string. Each bit in this string can represent some characteristic of the solution. Of course, there are many other ways of encoding. This depends mainly on the solved problem. For example, one can encode directly integer or real numbers; sometimes it is useful to encode some permutations and so on.

2.1 Initialization

Initially many individual solutions are randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions. Traditionally, the population is generated randomly, covering the entire range of possible solutions (the search space). Occasionally, the solutions may be "seeded" in areas where optimal solutions are likely to be found.

2.3 Selection

During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select the best solutions. Other methods rate only a random sample of the population, as this process may be very time-consuming.

Most functions are stochastic and designed so that a small proportion of less fit solutions are selected. This helps keep the diversity of the population large, preventing premature convergence on poor solutions. Popular and well-studied selection methods include roulette wheel selection and tournament selection.

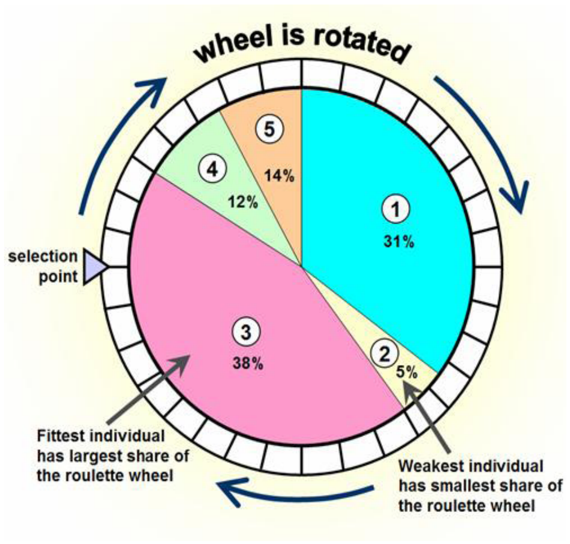
2.3.1 Roulette Wheel Selection:

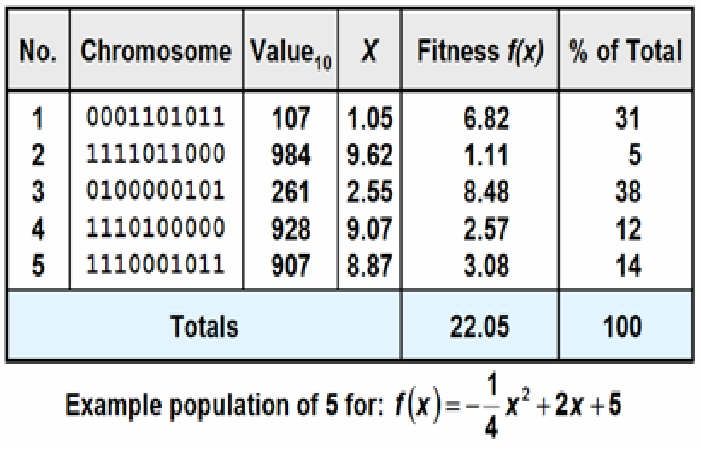
• Fitness level is used to associate a probability of selection with each individual solution.

• We first calculate the fitness for each input and then represent it on the wheel in terms of percentages.

• In a search space of ‘N’ chromosomes, we spin the roulette wheel.

• Chromosome with bigger fitness will be selected more times.


wheel is rotated
1
31%
2
5%
3
38%
4
12%
5
14%
selection point
Fittest individual has largest share of the roulette wheel
Weakest individual has smallest share of the roulette wheel



No.	Chromosome	$Value_{10}$	X	Fitness $f(x)$	% of Total
1	0001101011	107	1.05	6.82	31
2	1111011000	984	9.62	1.11	5
3	0100000101	261	2.55	8.48	38
4	1110100000	928	9.07	2.57	12
5	1110001011	907	8.87	3.08	14
Totals				22.05	100

Example population of 5 for: $f(x) = -\frac{1}{4}x^2 + 2x + 5$

Value10: Value of chromosome to the base 10

X = Value10normalized between 0 to 10 range

Fig 1. Roulette Wheel selection

2.3.2 Tournament Selection:

•Two solutions are picked out of the pool of possible solutions, their fitness is compared, and the better is permitted to reproduce.

•Deterministic tournament selection selects the best individual in each tournament.

•Can take advantage of parallel architecture

2.4 Crossover

Next step is to perform crossover. This operator selects genes from parent chromosomes and creates a new offspring. The simplest way how to do this is to choose randomly some crossover point and everything before this point copy from a first parent and then everything after a crossover point copy from the second parent.

Crossover can then look like this (| is the crossover point):

Table 1: Single Point Crossover

Chromosome 1	11011 | 00100110110
Chromosome 2	11011 | 11000011110
Offspring 1	11011 | 11000011110
Offspring 2	11011 | 00100110110

There are other ways how to make crossover, for example we can choose more crossover points. Crossover can be rather complicated and very depends on encoding of the chromosome. Specific crossover made for a specific problem can improve performance of the genetic algorithm.

2.5 Mutation

After a crossover is performed, mutation takes place. This is to prevent falling all solutions in population into a local optimum of solved problem. Mutation changes randomly the new offspring. For binary encoding we can switch a few randomly chosen bits from 1 to 0 or from 0 to 1. Mutation can then be following:

Table 2: Mutation

Original offspring 1	1101111000011110
Original offspring 2	1101100100110110
Mutated offspring 1	1100111000011110
Mutated offspring 2	1101101100110110

The mutation depends on the encoding as well as the crossover. For example when we are encoding permutations, mutation could be exchanging two genes.

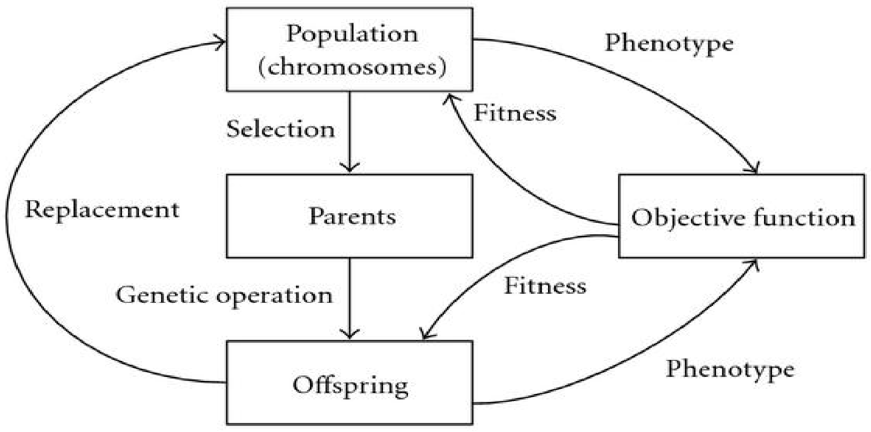


Fig 2: Genetic Algorithm Cycle

*Phenotype is the coding scheme used to represent the chromosomes.

3.0 Outline of the Basic Genetic Algorithm

1. **[Start]** Generate random population of *n* chromosomes (suitable solutions for the problem)
2. **[Fitness]** Evaluate the fitness *f(x)* of each chromosome *x* in the population
3. **[New population]** Create a new population by repeating following steps until the new population is complete

3.1 **[Selection]** Select two parent chromosomes from a population according to their fitness (the better fitness, the bigger chance to be selected)

3.2 **[Crossover]** With a crossover probability cross over the parents to form a new offspring (children). If no crossover was performed, offspring is an exact copy of parents.

3.3 **[Mutation]** With a mutation probability mutate new offspring at each locus (position in chromosome).
3.4 **[Accepting]** Place new offspring in a new population
4. **[Replace]** Use new generated population for a further run of algorithm
5. **[Test]** If the end condition (for example number of populations or improvement of the best solution) is satisfied, **stop**, and return the best solution in current population
6. **[Loop]** Go to step **2**

4.0 Global Optimization by Genetic Algorithm

Optimization is the science of finding decisions that satisfy given constraints, and meet a specific goal at its optimal value. In engineering, constraints may arise from physical limitations and technical specifications; in business, constraints are often related to resources, including manpower, equipment, costs, and time.

The objective of global optimization is to find the "best possible" solution in nonlinear decision models that frequently have a number of sub-optimal (local) solutions. In the absence of global optimization tools, engineers and researchers are often forced to settle for feasible solutions, often neglecting the optimum values. In practical terms, this implies inferior designs and operations, and related expenses in terms of reliability, time, money, and other resources.
The classical optimization techniques have difficulties in dealing with global optimization problems. One of the main reasons of their failure is that they can easily be entrapped in local minima. Moreover, these techniques cannot generate or even use the global information needed to find the global minimum for a function with multiple local minima.

The genetic algorithm solves optimization problems by mimicking the principles of biological evolution, repeatedly modifying a population of individual points using rules modeled on gene combinations in biological reproduction. Due to its random nature, the genetic algorithm improves the chances of finding a global solution. Thus they prove to be very efficient and stable in searching for global optimum solutions. It helps to solve unconstrained, bound-constrained, and general optimization problems, and it does not require the functions to be differentiable or continuous.

We next discuss an example that shows how to find the global minimum of Rastrigin's function using genetic algorithm. For two independent variables, Rastrigin’s function is defined as

$$Ras(x) = 20 + x_1^2 + x_2^2 - 10(\cos 2\pi x_1 + \cos 2\pi x_2).$$

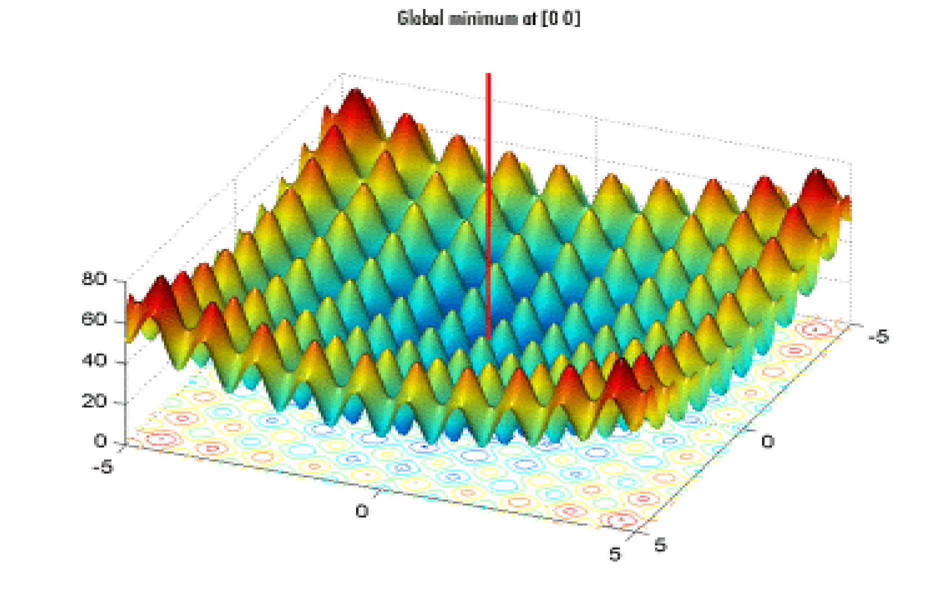


Fig 3: Global minimum of Rastrigin's Function

As the plot shows, Rastrigin's function has many local minima—the "valleys" in the plot. However, the function has just one global minimum, which occurs at the point [0 0] in the *x*-*y* plane, as indicated by the vertical line in the plot, where the value of the function is 0. At any local minimum other than [0 0], the value of Rastrigin's function is greater than 0. The farther the local minimum is from the origin, the larger the value of the function is at that point.

The following contour plot of Rastrigin's function shows the alternating maxima and minima.

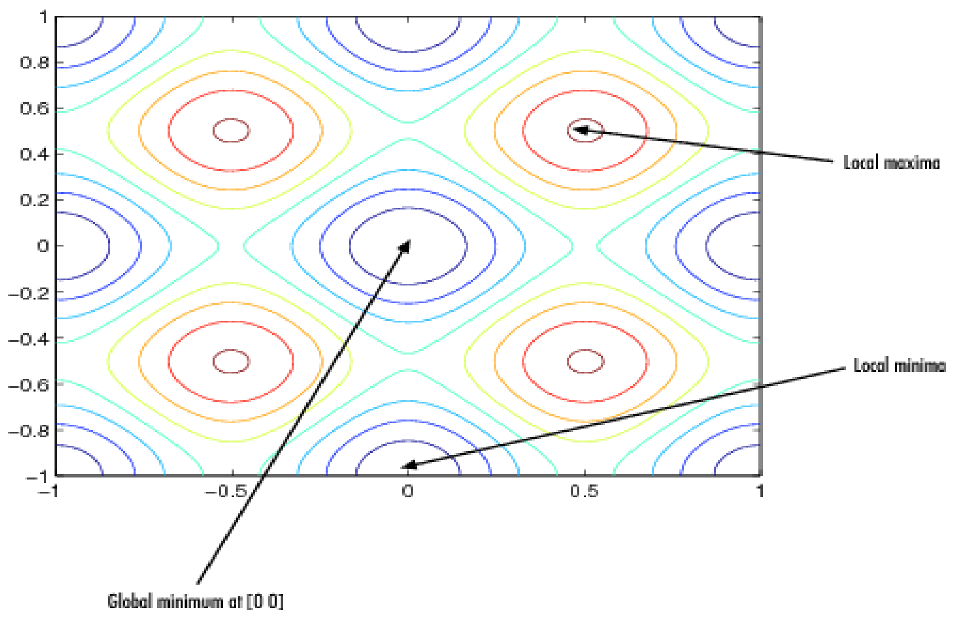


Fig 4: Contour Plot of Rastrigin's function

The final value of the fitness function (Rastrigin's function) when the algorithm terminated:

Objective function value: 0.05531602101322264

The value obtained is very close to the actual minimum value of Rastrigin's function, which is 0.

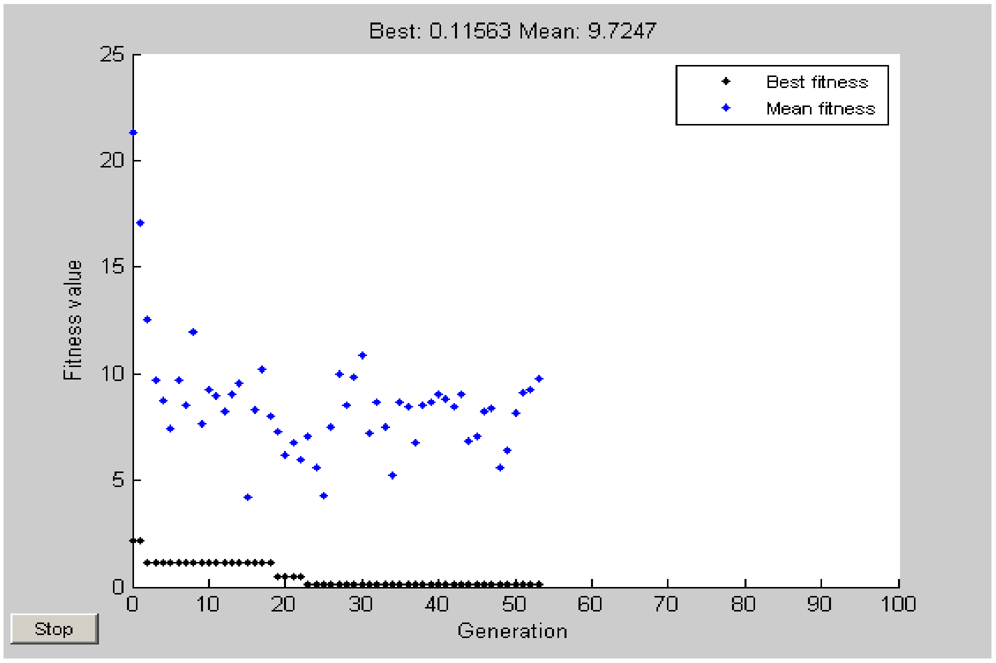


Fig 5: Plot of the best and mean values of the fitness function at each generation.

Source: *www.mathworks.com/products/global-optimization*

The points at the bottom of the plot denote the best fitness values, while the points above them denote the averages of the fitness values in each generation. The plot also displays the best and mean values in the current generation numerically at the top.

5.0 What makes GA a Good Global Optimization Tool?

- The first and most important point is that genetic algorithms are intrinsically parallel. Most other algorithms are serial and can only explore the solution space to a problem in one direction at a time, and if the solution they discover turns out to be suboptimal, there is nothing to do but abandon all work previously completed and start over. However, since GA has multiple offspring, they can explore the solution space in multiple directions at once. If one path turns out to be a dead end, they can easily eliminate it and continue work on more promising avenues, giving them a greater chance each run of finding the optimal solution.
- Due to the parallelism that allows them to implicitly evaluate many schemas at once, genetic algorithms are particularly well-suited to solving problems where the space of all potential solutions is truly huge - too vast to search exhaustively in any reasonable amount of time. It directly samples only small regions of the vast fitness landscape and successfully finds optimal or very good results in a short period of time after directly.
- Another notable strength of genetic algorithms is that they perform well in problems for which the fitness landscape is complex - ones where the fitness function is discontinuous, noisy, changes over time, or has

many local optima. GA has proven to be effective at escaping local optima and discovering the global optimum in even a very rugged and complex fitness landscape. However, even if a GA does not always deliver a provably perfect solution to a problem, it can almost always deliver at least a very good solution.

- Another area in which genetic algorithms excel is their ability to manipulate many parameters simultaneously. GAs are very good at solving such problems, in particular, their use of parallelism enables them to produce multiple equally good solutions to the same problem, possibly with one candidate solution optimizing one parameter and another candidate optimizing a different one and a human overseer can then select one of these candidates to use.
- Finally, one of the qualities of genetic algorithms which might at first appear to be a liability turns out to be one of their strengths, namely, GAs know nothing about the problems they are deployed to solve. Instead of using previously known domain-specific information to guide each step, they make random changes to their candidate solutions and then use the fitness function to determine whether those changes produce an improvement.

6.0 Conclusion

The problem of finding the global optimum in a space with many local optima is a classic problem for all systems that can adapt and learn. GA provides a comprehensive search methodology for optimization. GA is applicable to both continuous and discrete optimization problems. In global optimization scenarios, GAs often manifests their strengths: efficient, parallelizable search; the ability to evolve solutions with multiple objective criteria; and a characterizable and controllable process of innovation.

7.0 Future Trends

Some limitations of GAs are that in certain situations, they are overkill compared to more straightforward optimization methods such as hill-climbing, feed forward artificial neural networks using backpropagation, and even simulated annealing and deterministic global search. To make genetic algorithm more effective and efficient it can be incorporated with other techniques within its framework to produce a hybrid genetic algorithm that can reap best from its combination. Hybrid genetic algorithms have received significant interest in recent years and are being increasingly used to solve real-world problems quickly, reliably and accurately without the need for any forms of human intervention. More research needs to be concentrated on the development of hybrid design alternatives for its efficiency enhancement.

References:

[1] Charles C. Peck, Atam P. Dhawan, “Genetic algorithms as global random search methods: An alternative perspective”, Evolutionary Computation, Volume 3 Issue 1, MIT Press, March 1995.
[2] D. E. Goldberg, “Genetic Algorithms in Search, Optimization and Machine Learning”, Addison Wesley Publishing Company, Ind. USA, 1989.
[3] http://mathworld.wolfram.com/GlobalOptimization.html.
[4] http://www.maplesoft.com/products/toolboxes/globaloptimization
[5] L.Painton, J.Campbell, “Genetic algorithms in optimization of system reliability” Reliability, IEEE Transactions on Volume: 44 , Issue: 2, Digital Object Identifier: 10.1109/24.387368, Publication Year: 1995 , Page(s): 172 – 178
[6] L. T. Leng, Guided genetic algorithm, Doctoral Dissertation. University of Essex, 1999.
[7] M. Syrjakow and H. Szczerbicka, “Combination of direct global and local optimization methods”, in IEEE Conference on Evolutionary Computation. Perth, Western Australia: IEEE, 1995, pp. 326-333.
[8] Melanie Mitche, “An introduction to genetic algorithm”, MIT press, 1998.
[9] P. Larrañaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, S. Dizdarevic, “Genetic Algorithms for the Travelling Salesman Problem: A Review of Representations and Operators”, Artificial Intelligence Review , Volume 13 Issue 2, Kluwer Academic Publishers, April 1999.
[10] R. Fletcher, “Practical Methods of Optimization”, John Willey & Sons, 1986.
[11] T. Weise, “Global Optimization Algorithms - Theory and Application”. Available online at www.it-weise.de/projects/book.pdf
[12] Törn and A. Zilinskas, Global optimization, in Lecture Notes in Computer Science, vol. 350: Springer-Verlag, 1989.