International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

Performance Analysis of Different Selection Techniques in Genetic Algorithm

Priyanka Sharma[1], Dr. Rajesh Gargi[2]

[1]CSE Department, Geeta College of Engineering, Naultha, Panipat, India

[2]Director, Geeta Engineering College, Naultha, Panipat, India

Abstract: ***This Paper compares the performance of different selection techniques in GA using De Jong's function1 as function to be used fitness function. Genetic algorithm is one of the optimization techniques that can be used to solve the problems of function maximization. It can be said as a search procedure inspired by principles from natural selection and genetics [LOB00]. It is often used as an optimization method to solve problems where very little is known about the objective function. The operation of the genetic algorithm is very simple. It starts with a population of random individuals, each corresponding to a particular candidate solution to the problem to be solved. Then, the best individuals survive, mate, and create offspring, originating a new population of individuals. This process is repeated a number of times, and usually leads to better and better individuals.***

Keywords: Genetic algorithm, De Jong's function, Function Maximization

1. Introduction

Selection is the process of choosing two parents from the population for crossing. After deciding on an encoding, the next step is to decide how to perform selection i.e., how to choose individuals in the population that will create offspring for the next generation and how many offspring each will create. The purpose of selection is to emphasize fitter individuals in the population in hopes that their off springs have higher fitness. Chromosomes are selected from the initial population to be parents for reproduction. The problem is how to select these chromosomes. According to Darwin's theory of evolution the best ones survive to create new offspring.

Selection is a method that randomly picks chromosomes out of the population according to their evaluation function. The higher the fitness function, the more chance an individual has to be selected. The selection pressure is defined as the degree to which the better individuals are favored. The higher the selection pressured, the more the better individuals are favored. This selection pressure drives the GA to improve the population fitness over the successive generations. The convergence rate of GA is largely determined by the magnitude of the selection pressure, with higher selection pressures resulting in higher convergence rates.

The convergence rate of GA is largely determined by the magnitude of the selection pressure, with higher selection pressures resulting in higher convergence rates. We can distinguish two types of selection scheme, proportionate selection and ordinal-based selection.

Proportionate-based selection picks out individuals based upon their fitness values relative to the fitness of the other individuals in the population. An ordinal-based selection scheme selects individuals not upon their raw fitness, but upon their rank within the population.

Selection has to be balanced with variation form crossover and mutation. Too strong selection means sub optimal highly fit individuals will take over the population, and introduces the diversity needed for change and progress, too weak selection will result in too slow evolution.

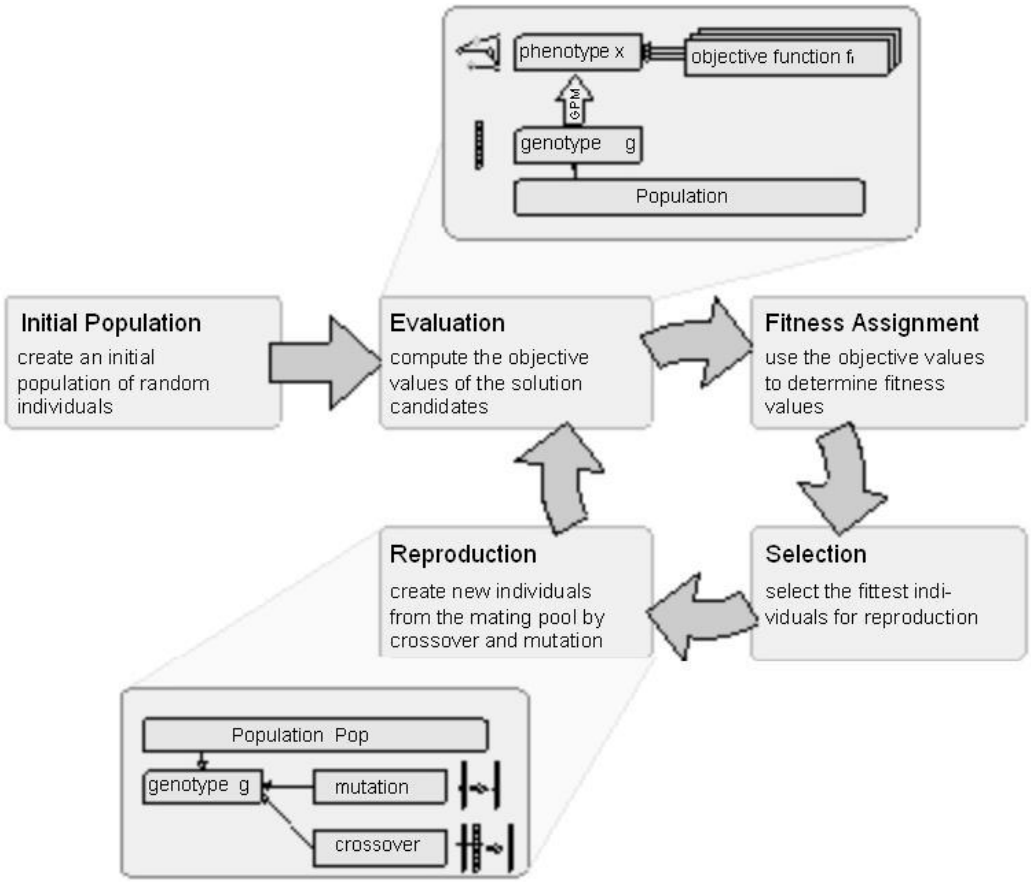
2. Genetic Algorithm

A genetic algorithm (GA) is a search technique used in computing to find exact or approximate solutions to optimization and search problems [3]. Genetic algorithms are categorized as global search heuristics. Genetic algorithms are a particular class of evolutionary algorithms (EA) that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover.

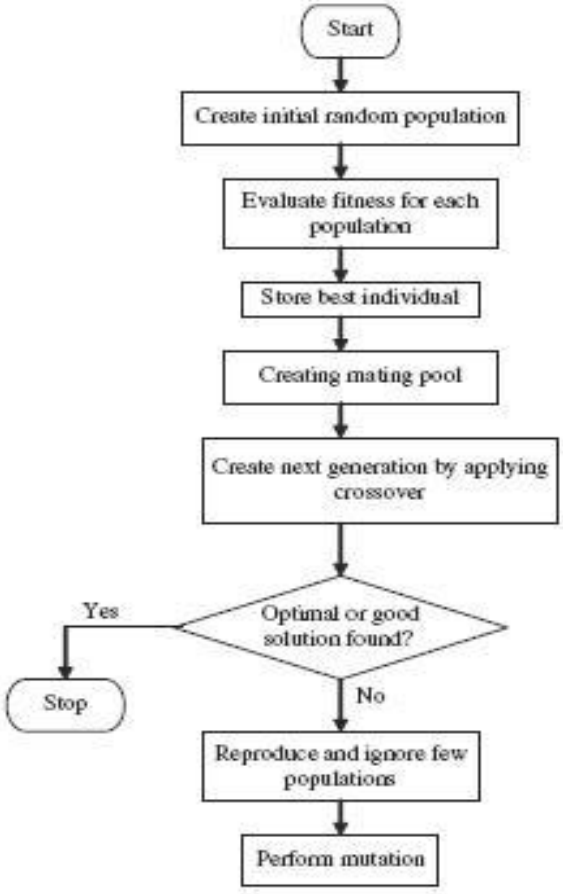
2.1 Simple Genetic Algorithm

The basic genetic algorithm is as follows:

- [start]: Genetic random population of n chromosomes (suitable solutions for the problem). •
- [Fitness]: Evaluate the fitness f(x) of each chromosome x in the population. •
- [New population]: Create a new population by repeating following steps until the new population is complete,
  [selection]: select two parent chromosomes from a population according to their fitness (the better fitness, bigger are the chances to get selected).
  [crossover]: With a crossover probability, cross over the parents to form new offspring (children). If no crossover asperformed, offspring is the exact copy of parent.
- [Mutation]: With a mutation probability, mutate new offspring at each locus (position in chromosome).
- [Accepting]: Place new offspring in the new population.
- [Replace]: Use newly generated population for a further sum of the algorithm.
- [Test]: If the end conditions are satisfied, stop and return the best solution in current population.
- [Loop]: Go to step2 for fitness evaluation.



Basic Cycle of Genetic Algorithm



3. Problem Statement

Here the problem is comparing the performance of different selection techniques in GA using De Jong's function1 as function to be used fitness function.

1. It starts with a population of random individuals, each corresponding to a particular candidate solution to the problem to be solved.
2. Then, the best individuals survive, mate, and create offspring, originating a new population of individuals.
3. This process is repeated a number of times, and usually leads to better and better individuals.

3.1 Genetic Operation

The operation starts with a population of random individuals, each corresponding to a particular candidate solution to the problem to be solved. Then, the best individuals survive, mate, and create offspring, originating a new population of individuals. This process is repeated a number of times, and usually leads to better and better individuals.

The steps of applying GA relating our problem at hand are:

1. Choosing an Encoding scheme.
2. Choosing Fitness function.
3. Choosing Operators.
4. Choosing Parameters.
5. Choosing an Initialization method and Stopping criteria

3.2 Encoding scheme

The application of a genetic algorithm to a problem starts with the encoding. The basis of genetics in nature is a chromosome. A particular solution to the problem can then be represented by a specific assignment of values to the decision variables. The set of all possible solutions is called the search space and a particular solution represent a point in that search space. Various types of encoding schemes are:

1. Binary encoding
2. Permutation encoding
3. Value/Real encoding
4. Tree encoding
5. Octal encoding &
6. Hexadecimal encoding.

3.3 Real/Value encoding

Direct value encoding can be used in problems where some complicated value such as real numbers are used. Use of binary encoding for this type of problems would be very difficult. In value encoding, every chromosome is a string of some values. Values can be anything connected to problem, form numbers, real numbers or chars to some complicated objects

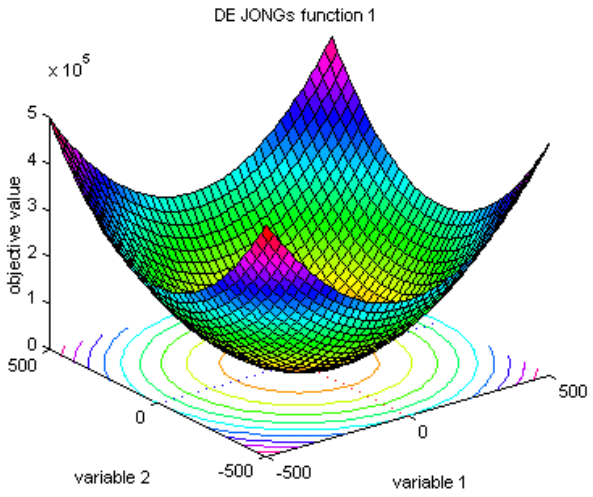
Chromosome A 1 5 3 2 6 4 7 9 8
Chromosome B 8 5 6 7 2 3 1 4 9

The next step after encoding is to select the fitness function.

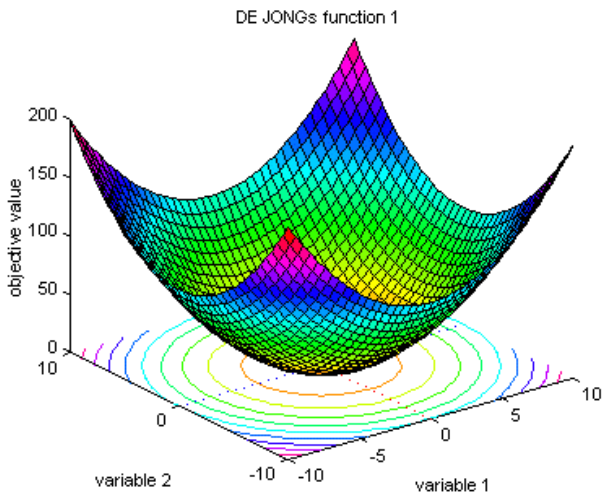
a) ***Fitness function*** *a function that can assign a score to any possible solution. The score is a numerical value that indicates how well a particular solution solves the problem. The score is the fitness of the individual solution. It represents how well the individual suits to the environment. In this case, the environment is the search space. The task of the GA is to discover solutions that have higher fitness values among the set of all possible solutions.*

For our problem the fitness function chosen is from literature of benchmark functions commonly used in order to test optimization procedures dedicated for multidimensional, continuous optimization task. The quality of optimization procedures are frequently evaluated by using common standard literature benchmarks.
There are several classes of such test functions, all of them are continuous:
(a) Unimodal, convex, multidimensional,
(b) Multimodal, two-dimensional with a small number of local extremes,
(c) Multimodal, two-dimensional with huge number of local extremes
(d) Multimodal, multidimensional, with huge number of local extremes.
The one which is used in this work is De Jong's function 1, also called as sphere model. The simplest test function is De Jong's function 1. It is also known as sphere model. It is continuous, convex and unimodal. It has the following general definition:

$F_1(x) = \sum_{i=1}^{n} x_i^2$, $-5.12 <= x_i <= 5.12$
$F_1(x)$=sum(x(i)^2), where, i =1:n and -5.12<=x(i)<=5.12.
Global minimum is at: f (x)=0, x(i)=0, where, i =1:n



(a)



(b)

Visualization of De Jong's function 1 using different domains of the variables, however, both graphics look similar, just the scaling changed.(a) Surface plot of the function in a very large area from -500 to 500 for each of the variables, (b) The function at a smaller area from -10 to 10.

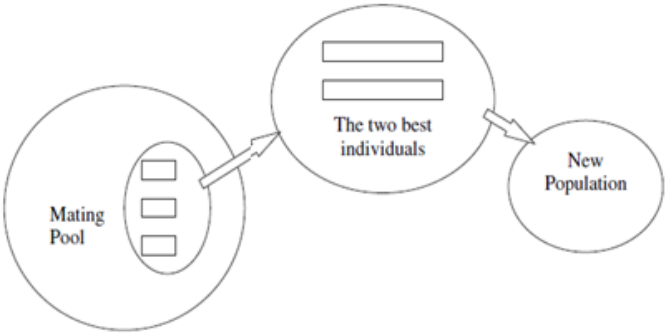
b)Operators

Once the encoding and the fitness function are specified, the implementer has to choose selection and genetic operators to evolve new solutions to the problem being solved. The selection operator simulates the "survival-of-the-fittest". There are various mechanisms to implement this operator, and the idea is to give preference to better individuals. Selection replicates individuals with high fitness values and removes individuals with low fitness values.

c) Selection

Selection is the process of choosing two parents from the population for crossing. After deciding on an encoding, the

next step is to decide how to perform selection i.e., how to choose individuals in the population that will create offspring for the next The higher the fitness function, the more chance an individual has to be selected. The selection pressure is defined as the degree to which the better individuals are favored. Selection has to be balanced with variation form crossover and mutation. Too strong selection means sub optimal highly fit individuals will take over the population, and it reduces the diversity needed for change and progress, too weak selection will result in too slow evolution.



Selection Processes

Roulette wheel selection-- We use Roulette selection which is one of the traditional GA selection techniques.. The principle of roulette wheel selection is a linear search through a roulette wheel with the slots in the wheel weighted in proportion to the individual's fitness values [2][4]. A target value is set, which is a random proportion of the sum of the finesses in the population. The population is stepped through until the target value is reached. The expected value of an individual is that fitness divided by the actual fitness of the population. A slice of the roulette wheel is assigned to each individual, the size of the slice being proportional to the individual's fitness. Here N is the number of individuals in the population so the wheel is spun N times. On each spin, the individual under the wheel's marker is selected to be parents for the next generation.

This method is implemented as follows:

1. Total expected value of the individuals in the population is summed. Let it be T.
2. Repeat N times:

- A random integer „r" between o and T is Chosen.
- Loop through the individuals in the population, summing the expected values, until the sum is greater than or equal to „r". The individual whose expected value puts the sum over this limit is the one selected.

Roulette wheel selection is easier to implement but is noisy. The rate of evolution depends on the variance of fitness's in the population.

Crossover

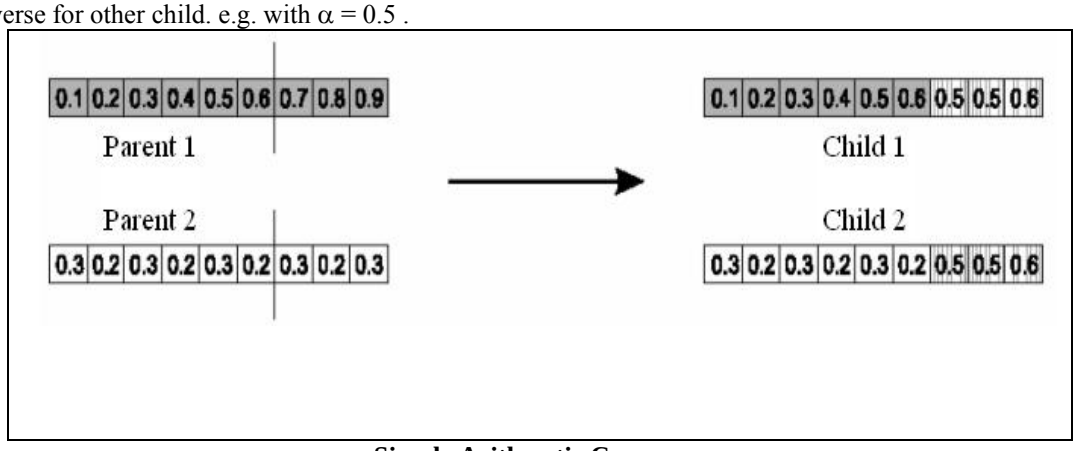
Crossover is the process of taking two parent chromosomes and producing from them a number of offspring's. After the selection (reproduction) process, the population is filled with better individuals. Reproduction makes exact copies of good strings but does not create new ones. Crossover operator is applied to the mating pool with the hope that it creates a better offspring [7].

Crossover is a recombination operator that works in three steps:
i. The reproduction operator selects at random a pair of two individual strings for the mating/crossover.
ii. A cross site or cross point is selected at random along the string length.

- iii. Finally, the position values are swapped between the two strings following the cross site.
- **Simple Arithmetic crossover**
- Let the two Parents be: $\langle x_1,\ldots,x_n \rangle$ and $\langle y_1,\ldots,y_n \rangle$
- Pick random gene *(k)* after this point mix the values.
- Child 1 is:

$$\left\langle x_1,\ldots,x_k,\alpha\cdot y_{k+1}+(1-\alpha)\cdot x_{k+1},\ldots,\alpha\cdot y_n+(1-\alpha)\cdot x_n\right\rangle$$

- Reverse for other child. e.g. with $\alpha = 0.5$.



Simple Arithmetic Crossover

Mutation

After crossover, the strings are subjected to mutation. Mutation prevents the algorithm to be trapped in a local minimum. Mutation plays the role of recovering the lost genetic materials as well as for randomly distributing genetic information.

Mutation is viewed as a background operator to preserve genetic diversity in the population. Mutation helps escape from local minima's trap and maintains diversity in the population. It also keeps the gene pool well stocked, and thus ensuring periodicity.
Mutation of a bit involves flipping a bit, changing 0 to 1 and vice-versa. There are various types of mutation schemes such as:

- Flipping
- Interchanging
- Reversing
- Uniform mutation &
- Non-uniform mutation.

Uniform Mutation

- General scheme of floating point mutations

$$\bar{x} = \langle x_1, \ldots, x_l \rangle \rightarrow \bar{x}' = \langle x_1', \ldots, x_l' \rangle$$
$$x_i, x_i' \in [LB_i, UB_i]$$

- Uniform mutation:

$$x_i' \text{ drawn randomly (uniform) from } [LB_i, UB_i]$$

- Analogous to bit-flipping (binary) or random resetting (integers).

A. Parameters

With an encoding, a fitness function, and operators in hand, the GA is ready to enter in action. But before doing that, the user has to specify a number of parameters such as population size, selection rate, and operator probabilities [LOB00]. For our problem, the population size chosen is around 100 approximately. With selection rate depending upon the various selection schemes employed. Operator probabilities are chosen in order to maintain the population diversity. For crossover the probability is 100% i.e. with every iteration/algorithm run crossover is performed always. Whereas the mutation operation performed after every five consecutive algorithm runs.

B. Initialization method & stopping criteria

The last steps of applying a GA are the specification of an initialization method and stopping criteria. The genetic algorithm is usually initialized with a population of random individuals, but sometimes a fraction of the population is initialized with previously known (good) solutions [LOB00].

Following the initialization step, each individual is evaluated according to the user's specified fitness function. Thereafter, the GA simulates evolution on the artificial population of solutions using operators that imitate the survival-of-the-fittest and principles of natural genetics such as recombination and mutation

4. Conclusion

Function optimization/maximization is the main point of discussion in this dissertation. Actually this dissertation is based on implementing De Jong's function1 i.e. sphere model using different selection techniques used in Genetic algorithm and making a comparison of them based on the fitness values of function at different number of iterations.

The main point around which all this work revolves is the different selection techniques employed for running this algorithm. All other parameters are kept constant, except the three selection techniques (Roulette wheel, Random selection and Best Fit/Elitist). Termination criteria chosen is the number of iterations and the function reaching its maximum value. In future, further other different selection techniques can be employed for maximizing this function. And other prospect in future can be, changing other parts of the genetic algorithm keeping a particular selection technique fixed which might finally show even better results as compared to that came now. And further a larger size of chromosomes could be used for better results.

References

[1] H.Nazif(2009). *A Genetic Algorithm on Single Machine Scheduling Problem to Minimise Total Weighted Completion Time*. European Journal of Scientific Research,Vol.35 No.3, pp.444-452
[2] Garey, M. & Johnson, D. (1979). *Computers and Intractability: a theory of NP- Completeness.* W.H.Freeman, San Francisco Proceedings of the Second International Conference on Genetic Algorithms, pp. 252-256
[3] Back, T., Fogel, David B. & Michalewicz, Z. (Eds.) (1997). *Handbook of Evolutionary Computation*. Computational Intelligence, Library Oxford University Press in cooperation with the Institute of Physics Publishing / CRC Press, Bristol, New York, ring bound edition. ISBN: 0-7503-0392-1, 978-0-75030-392-7, 0-7503-0895-8, 978-0-75030-895-3. [Also available online at:http://books.google.de/books?id=n5nuiIZvmpAC]
[4] Back, T. (1996). *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press US. ISBN: 0-1950-9971-0, 978-0-19509-971-3.[Also available online at: http://books.google.de/books?id=EaN7kvl5coYC]
[5] Tomassini, M. (1999). *Parallel and Distributed Evolutionary Algorithms: A Review*. In Miettinen, K., Makela, M., & Periaux, J. (Eds.), *Evolutionary Algorithms in Engineering and Computer Science* (pp. 113 - 133). Chichester: J. Wiley and Sons.
[6] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Boston: Addison-Wesley
[7] Sivanandam, S. N. & Deepa, S. N. (2008). *Introduction to Genetic Algorithms*. Springer
[8] Davis, L. (1991). *Handbook of Genetic Algorithm*. Von Nostrand Reinhold, Newyork.
[9] Lobo, Fernando Miguel (2000).*The parameter-less genetic algorithm: rational and automated parameter selection for simplified genetic.*