International Journal of Enhanced Research in Science Technology & Engineering, ISSN: 2319-7463
Vol. 4 Issue 3, March-2015, pp: (264-268), Impact Factor: 1.252, Available online at: www.erpublications.com

Study of Selection on Performance of Genetic Algorithms

Manoj Kr. Mahto[1], Mr. Lokesh Kumar[2]

[1]M. Tech (CSE), RIEM, Rohtak
[2]HOD, CSE Dept. RIEM, Rohtak

ABSTRACT: Genetic Algorithms are biologically inspired optimisation algorithms. Performance of genetic algorithms largely depends on type of genetic operators – Selection, Crossover, Mutation and Replacement used in it. The paper focuses on different selection and replacement operators. Selection operator is used to select the individuals from a population to create a mating pool which will participate in reproduction process. Replacement operator decides which individuals stay in a population and which are replaced by removing or replacing some offspring or parent individuals. The paper focuses on three selection operators – Roulette Wheel Selection, Rank Selection and Annealed Selection. Annealed selection is the new approach which blends the exploitative nature of roulette wheel and exploratory nature of rank selection. Generational and μ+λ replacement operators are implemented in this paper. Implementation is carried out using MATLAB code on two test problems – Benchmark TSP Eil51 problem and Benchmark DeJong's Sphere Function (F1). The paper compares the performance of genetic algorithm using The paper compares the performance of genetic algorithm using these three selection approaches with generational replacement and μ+λ replacement. The results are optimistic and clearly demonstrate that the genetic algorithm with μ+λ replacement is better than the one with generational replacement. Out of the three selection operators, annealed selection outperforms the other two.

Keywords: genetic algorithm, replacement, selection.

1. INTRODUCTION

Genetic algorithms are random search algorithms that were defined as adaptive heuristic search algorithms based on the evolutionary ideas of natural selection and natural genetics by David Goldberg [1]. Genetic algorithms are applicable to wide range of problems pertaining to non-linear programming, stochastic programming, optimisation and combinatorial problems. Genetic algorithm mimics the natural evolution process. It iteratively transforms a population of fixed-length strings. The strings of artificial genetic systems are analogous to chromosomes in biological systems. Total package of strings is called structure and is analogous to genotype. The structures decode to form a particular parameter set, solution alternative or point, which correspond to phenotype. The chromosome is composed of genes, each gene describing a parameter of optimisation problem. The alphabets on strings are referred to as genes and the values of genes are called alleles [2]. Each chromosome has a fitness value associated with it. A typical genetic algorithm is composed of three main operators – Selection, Crossover and Mutation.

A genetic algorithm operates on population of constant size. An initial population of individuals is generated randomly or heuristically. Selection operator is used to improve the quality of the population by selecting the fittest individuals to form the mating pool. Crossover operator takes two individuals with higher fitness values and randomly chooses the position and length of the portion to be exchanged and performs this operation at either single or multiple points. Mutation introduces new genetic structures in the population by randomly modifying some of the genes, helping the search algorithm to escape from local optimum by reaching new points in the search space. Current generation of individuals is replaced by newly generated offsprings by the specific replacement strategy. Genetic algorithms are stochastic iterative algorithms, so the algorithm iterates till maximum number of generations is reached or the cycle of genetic algorithm continues until the optimal solution is achieved [3].

When a new generation of offsprings is produced, the next question is which of these newly generated offsprings would move forward to the next generation and would replace which chromosomes of the current generation. The answer to this question is based on Darwin's principle of "Survival of Fittest" [4]. So better fit individuals have more chances to survive and carried forward to next generation leaving behind the less fit ones. The process of forming next generation of individuals by replacing or removing some offsprings or parent individuals is done by replacement operator. This process in evolution is known as replacement scheme [3].

In this paper, the focus is to study the performance of genetic algorithms by changing replacement strategies and selection method. The paper is organized in the following sections.

2. RELATED WORK

In 1966, Fogel stated in his work related to evolutionary programming that each individual produces one offspring and best half from the parent and offspring populations are selected to form the new population. This replacement technique involves overlapping of the two populations as parents and their offsprings constantly compete with each other for survival 4]. Cavicchio mentioned the use of Pre-selection schemes to preserve diversity. The best Preselection scheme is that if a child has higher fitness than the worse parent, it replaces the parent and is also known as Parental Replacement technique. It implies selection before regular selection [5].

The (μ+1) approach was the first steady state replacement strategy introduced by Rechenberg in 1973 and had parent population greater than one ($\mu > 1$). All parents participated to produce one offspring and one offspring is eliminated in each generation. Rechenberg calculated the convergence rates of two model functions and postulated his 1/5 success rule. The (μ+1) approach could not self adapt to different step sizes, so was later not used in evolutionary methods [6]. Grefenstette focused on behavior of standard genetic algorithm on certain class of non-stationary environment. His empirical study confirmed that larger generation gap value improved performance [7].

Whitley introduced GENITOR in which worst λ individuals were deterministically replaced every iteration. This led to very rapid improvements in the mean population fitness, but in certain cases also lead to premature convergence as the population focused on fittest member currently present [8]. Goldberg & Deb analysed GENITOR and observed that it has high selective pressure even if parents are selected randomly and suggested that deletion of worst individuals was major factor in selection intensity [9]. Syswerda compared generational and steady state genetic algorithms with fitness proportional selection of parents and several replacement methods. Syswerda showed that generation gap had no effect on the allocation of copies to strings [10].

Smith and Vavak considered both random and worst fit replacement strategy. They observed that replacing the oldest member or replacing randomly may result in loss of optimal value. They noted that the loss can be corrected simply by using an elitist replacement strategy that the best individual in current generation survives to the next [11]. Kay Weise & Scott D Goodwin proposed an intermediate selection strategy called Keep Best reproduction with a motive that both parents pass on their good genetic material to their children. They tested their proposition on Travelling Salesman Problem and found that this technique outperforms standard generational replacement. This technique also benefits from higher mutation rates in contrast to generational replacement [12].

3. SELECTION AND REPLACEMENT

Selection operation is the primary operation in genetic algorithm. It is used to choose the best fit individuals in the population to create the mating pool. Individuals in the mating pool will participate in further genetic operations to create the next generation of population. The next generation of population is created with a hope to reach the optimal solution. Selection of individuals in the population is fitness dependent and is done using different algorithms [9]. Selection chooses more fit individuals in analogy to Darwin's theory of evolution – survival of fittest [4]. Too strong selection would lead to sub-optimal highly fit individuals and too weak selection may result in too slow evolution [13]. There are many methods in selecting the best chromosomes such as roulette wheel selection, rank selection, steady state selection and many more.

Replacement is the last step in breeding step of any genetic algorithm cycle. It is used to decide which individuals stay or get replaced in a population [3]. Basically, there are two kinds of replacement strategies for maintaining the population – generational replacement and steady state replacement. In generational replacement, entire population of genomes is replaced at each generation. In elitism, complete population of genome is replaced except for the best member of each generation which is carried over to next generation without modification [14]. In this case, generations are non-overlapping. Steady state replacement involves overlapping population in which only a small fraction of the population is replaced during each iteration. In a steady state replacement, new individuals are inserted in the population as soon as they are created [15]. The paper reviews roulette wheel selection, rank selection and annealed selection operator [16] and then effect of selection and two different replacement strategies on performance of genetic algorithm.

A. Roulette Wheel Selection

Roulette wheel is the simplest selection technique. In this technique, all the chromosomes in the population are placed on the roulette wheel according to their fitness value [2,9,17]. Each individual is assigned a segment of roulette wheel whose size is proportional to the value of the fitness of the individual. The bigger the fitness value is, the larger the segment is. Then, the virtual roulette wheel is spinned. The individual corresponding to the segment on which roulette wheel stops, is then selected. The process is repeated until the desired number of individuals is selected. Individuals with higher fitness have more probability of selection. It can possibly miss the best individuals of a population at certain times. There is no guarantee that good individuals will find their way into next generation. Roulette wheel selection uses exploitation technique in its approach. Rank Selection sorts the population first according to fitness value and ranks them. Then every chromosome is allocated selection probability with respect to its rank [18]. Individuals are selected as per their selection probability. Rank selection is an explorative technique of selection. Rank selection prevents too quick convergence and differs from roulette wheel selection in terms of selection pressure. Rank selection overcomes the scaling problems like stagnation or premature convergence. Ranking controls selective pressure by uniform method of scaling across the population. Rank selection behaves in a more robust manner than other methods[8,19].

B. Annealed Selection

The annealed selection approach is to move the selection criteria from exploration to exploitation so as to obtain the perfect blend of the two techniques. In this method, fitness value of each individual is computed as per the current generation number. Selection pressure is changed with changing generation number and new fitness contribution, FXi of each individual is computed. Selection probability of each individual is computed on the basis of FXi. The annealed selection operator computes fitness of individual depending on the current number of generation as under [16]:

$$FXi = Fi / ((ngen+1) - nogen) \qquad (1)$$

where FXi is fitness of individual i computed using annealed selection, Fi is fitness of individual as computed by the fitness function, ngen is total number of generations and nogen is current generation number.

C. Generational Replacement

In generational replacement, entire population of chromosomes is replaced by new set of chromosomes at each generation [3]. Two consecutive generations are non-overlapping using this replacement. Module for Generational Replacement is :

```
Replace(P,S,n)

//S is set of chromosomes generated as offspring //P is parent generation of chromosomes
P:=S
End
```

E.μ+λ Re

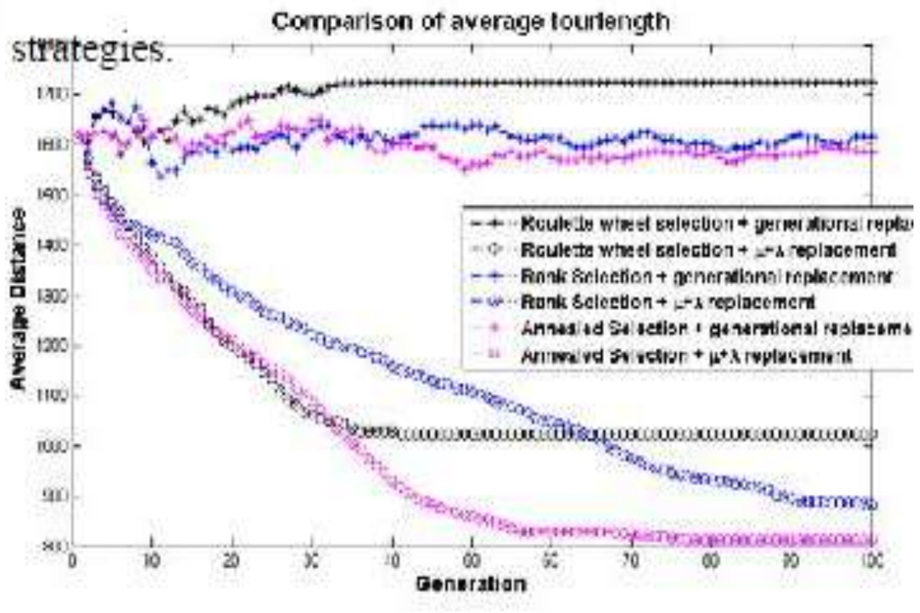
Procedure: GA(fitfn, n, r, m,ngen)

```
//fitfn is fitness function to evaluate chomosomes
//n is the population size in each generation (say 100)
//r is fraction of population generated by crossover (say 0.7)
//m is the mutation rate ( say 0.01)
//ngen is total number of generations
P := generate n individuals at random // initial generation is generated randomly i:=1
//define the next generation S of size nwhile i <=ngen do
{ //Selection step:
        L:= Select(P,n)        //  n/2 individuals of P will be selected using          any of the three
selection methods
        //Crossover step:
S:= Crossover(L,n,r)            // Generates n chromosomes with crossover probability r.
//Mutation step:
         Mutation(S,m)        //Inversion of chromosomes with mutation rate m
//Replacement step:
         Replace(P,S,n)        //Replaces old population using any of the            two replacement
strategies
pb(i):=min(fitfn(P)) // store best individual in population
i:=i+1
}
```

4. IMPLEMENTATION AND OBSERVATION

In this paper, genetic algorithm is developed using MATLAB code for two test problems – Benchmark TSP- Eil51 problem and Benchmark sphere function (F1). The Travelling Salesman Problem (TSP) is to find the shortest tour or Hamiltonian path through a set of N vertices so that each vertex is visited exactly once [20]. The code uses PMX crossover [2] for Benchmark TSP problem and other factors like initial population, population size, number of generations, crossover and mutation probability are kept constant to compare the performance of genetic algorithm in various cases of selection and replacement.

Figure 1 and Figure 2 illustrate the comparison of average and minimum tour length respectively of Eil51 TSP problem for 100 generations. Figure 3 and Figure 4 illustrate the comparison of average and minimum tour length selection wise using column chart for the two replacement strategies.



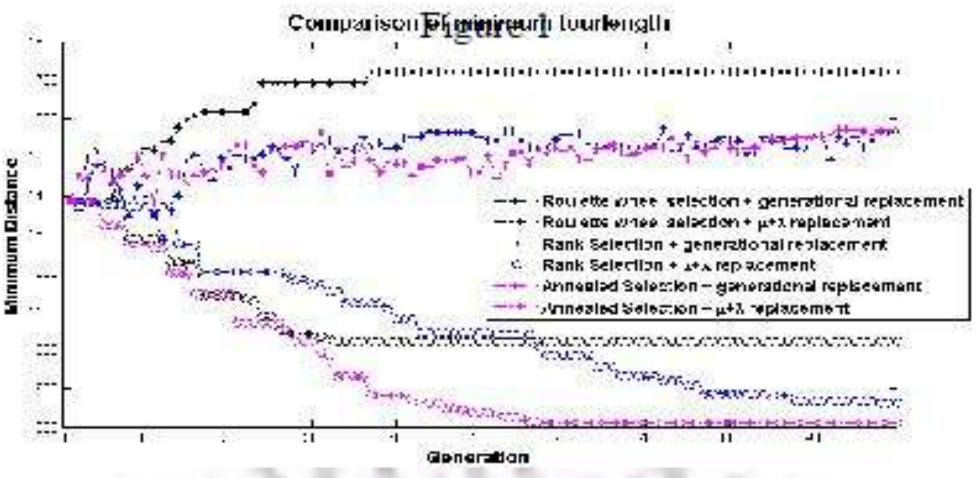


Figure 1

Figure 1 and Figure 2 clearly state that the pink lines for Annealed Selection show better results than the blue and black lines. It is quite noticeable from Figure 3 and Figure 4 that μ+λ replacement shows better result in each case of selection. Rank Selection had exploratory nature as it is continuously looking for new solutions and Roulette wheel selection found better chromosomes in early runs of generation and converged earlier than Rank Selection. In case of Annealed Selection, early runs of generation depicted exploration and with increasing number of current generation, it had exploiting nature and converged to find the better solution. This is due to increasing selection pressure in each generation as its performance is dependent on the current number of generation. There is less pressure on selection in early generations, so it had exploratory nature. Selection pressure increased turning the exploratory nature gradually into exploiting nature as the number of current generation increased.

5. CONCLUSION

In this paper, the performance of annealed selection operator [17] that has been proposed by the authors is analysed in combination with two different replacement methods – generational replacement and μ+λ replacement. The results clearly demonstrated that the genetic algorithm implementing μ+λ replacement is better than the one implementing generational replacement. Out of the three selection operators, annealed selection outperformed roulette wheel selection and rank selection. It has been experimentally proved that annealed selection operator has a blend of exploration as well as exploitation and gives better results in both replacement strategies. The experiments have been conducted of two different types of test problems – Benchmark TSP and Benchmark DeJong's Sphere function and the results confirmed optimistic results in both the cases in favour of Annealed selection and μ+λ replacement.

REFERENCES

[1] J.Holland, Adaptation in natural and artificial systems, University of Michigan Press, Ann Arbor, 1975.
[2] D.E. Goldberg, Genetic algorithms in search, optimisation, and machine learning, Addison Wesley Longman, Inc., ISBN 0-201-15767-5, 1989.
[3] S.N. Sivanandam, S.N. Deepa, Introduction to Genetic Algorithms, Springer, ISBN 9783540731894, 2007.
[4] Fogel D.:, Evolutionary Computation, IEEE Press, 1995.
[5] D.J. Cavicchio, Adaptive search using simulated evolution, Unpublished doctoral dissertation, University of Michigan, Ann Arbor, 1970.
[6] I. Rechenberg, Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Frommann-Holzboog, Stuttgart, 1973.
[7] John J. Grefenstette, Genetic algorithms for changing environments, Parallel Problem Solving from Nature, 2 (Reinhard Manner and Bernard Manderick, eds.). Amsterdam: North Holland, pp 137-144, 1992.
[8] D. Whitley, "The GENITOR algorithm and selection pressure: why rank-based allocation of reproductive trials is best", Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann, pp 116-121, 1989.
[9] D.E. Golberg and K.Deb, "A comparative analysis of selection schemes used in genetic algorithms", Foundations of Genetic Algorithms, San Mateo, CA, Morgan Kaufmann, pp69-93, 1991.
[10] G.Syswerda, A study of reproduction in generational and steady state genetic algorithms.