International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2016): 79.57 | Impact Factor (2015): 6.391

Outlander Algorithm Based on Integrated Aggressive Selection Method

Mashar Cenk Gençal[1], Mustafa Oral[2]

[1] Cukurova University, Computer Engineering Department, Balcali Sarıcam, Adana 01330, Turkey

Abstract: ***Numerous algorithms are utilized in optimization problems. One of the most commonly used methods to find optimum points of a given function is Genetic Algorithms, which stochastically select individuals from the population. The aim of genetic algorithms is to gradually approximate to the optimum points by choosing the better individuals in each iteration. Thus, having a good selection method is significantly important issue in genetic algorithms. In this paper, a new selection method, which is an improved version of Integrated Aggressive Selection Method, is introduced. The performance of the new method is compared with four methods that were previously proposed by us; Aggressive, Integrated Aggressive, Non-Aggressive and Integrated Non-Aggressive selection methods, and with the most commonly used standard selection methods; Roulette Wheel, Linear Ranking and Tournament by utilizing variety of benchmark functions. It is observed that newly proposed "Outlander" algorithm delivers the overall best performance results comparing to the other selection methods for both unimodal and multimodal optimization problems.***

Keywords: Genetic algorithms, Outlander, Selection methods, Integrated Aggressive, Aggressive, Non-Aggressive, Integrated Non-Aggressive, Tournament

1. Introduction

Genetic Algorithm, introduced by Holland in 1975 [1], is the larger class of Evolutionary Algorithms. It is a heuristic search method inspired from Darwin's Evolution Theory. Since it offers robust search in problem space, the idea has influenced many researchers [2]. The algorithm provides a solution or a group of solutions that are optimal or near-optimal for the problem. GA accomplishes this by embodying various bio-inspired searching instruments such as; recombination, mutation and most importantly selection. It is a rule of thumb that a good search algorithm has to have to important features; exploration and exploitation. It is really important to maintain these two features throughout the searching process. Many metaheuristic search algorithms such as PSO [3], ACO [4] and even GA, abolish exploration after a certain time in order to converge to an optimum point and exploit that area to improve the optimal solution. However, this may result with getting stuck to a local optimum point, and missing the global one.

Neither crossover nor mutation operators provide sustainable exploration all along the searching process. Even though these two operators contribute to exploration at the early stages of searching, they eventually help to exploitation. Selection, on the other hand, is an important operation to improve the quality of next generation. By promoting better individuals to produce more offspring, the quality of future generations is expected to be higher than the past ones. A good selection algorithm should maintain diversity in a population. However, as the population converges to an optimum point, the diversity will, eventually, be lost unless some other measures are taken.

In order to solve an optimization problem with GA, we should be able to create a solution suggestion to the problem and be able to evaluate the quality of the candidate solutions. As long as we can create and evaluate goodness of candidate solutions, GA can be employed to solve an optimization problem.

GA starts with randomly creating an initial population that is a collection of candidate solutions (individuals). Since, the individual creation process is stochastic, quality of initial population can be in any grade. Sometimes, with the help of a priori to the problem, tailored initial populations can be designed. However, this effort may never pay off. Totally randomized process usually results with unpredictably good solutions.

The fitness (goodness) values of each individual in the population are computed by a fitness evaluation function that is defined specifically to the problem by the developer. These fitness values are utilized for the selection step of GA. For instance, in selection step, the individual having better fitness values should be advantageous to pass their genes to the next generations through recombination/crossover step [5]. In crossover, offspring are obtained by combining the genes of the parents through a mating process. Offspring are thought to be possessing good traits of their ancestors in order to converge to a good solution. Predominantly good solutions overpower the mate selection process; therefore, extra cautions have to be considered. Mutation whose purpose is maintaining diversity and preventing premature convergence [5] is applied to some of the offspring by altering their genetic structures. Afterwards, the next generation of the population is created by combining a proportion of offspring and members of the current population. The algorithm returns to the selection phase by using the new generation and repeats the steps until the termination criteria are met; either the optimum solution is found or maximum number of generations are reached. The algorithmic steps of standard genetic algorithm are given in Figure 1.

Selection algorithms have to be designed carefully by taking into account favoring the individuals with good traits for mating and maintaining the diversity of individuals. However, usually the latter is sacrificed for convergence. We propose a new algorithm called "Outlander" that promotes better individuals, allows convergence and most importantly maintains exploration abilities throughout the search process. The inspiring work that leads to "Outlander" is given in section three, and the proposed algorithm is described in detail in section four. Optimization performance of "Outlander" is tested and discussed in section five.

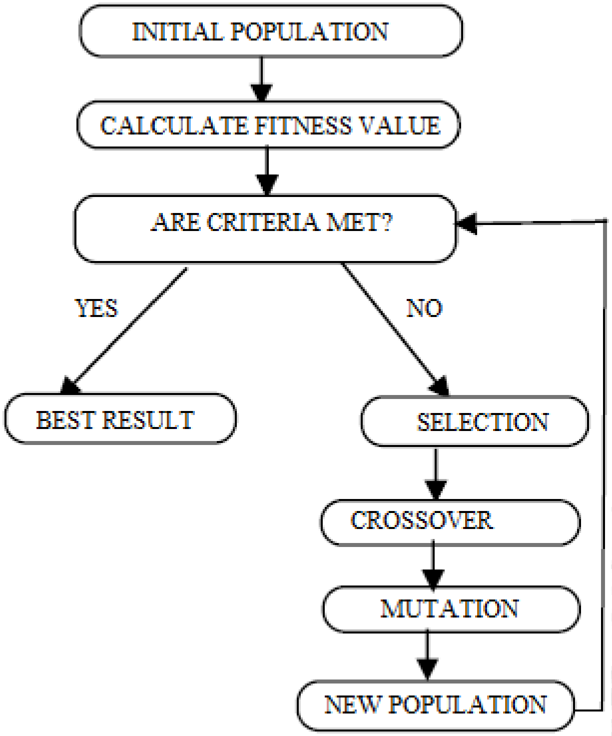

Figure 1: The process of a standard genetic algorithm

2. Literature Review

Roulette Wheel (RW) is the first selection method for GAs [1]. Even if implementation of RW is straight-forward, this method has some disadvantages such as; individuals having high fitness value dominates the population too early, no selection pressure for some cases, etc.

Linear ranking (LR) was introduced by Baker in 1989 [6]. The idea was to eliminate the main problems of RW, which are premature convergence and selection pressure. Even if LR tries to solve these problems, the method still encounters with some other issues: reducing selection pressure may cause to converge slowly, population must be sorted in each iteration, and unfortunately, it is not suitable for parallel processing since it requires a global ranking for all individuals.

Another well-known selection method is Tournament. Tournament selection was analyzed, and published by Blickle in 1995 [7]. The benefits of Tournament selection can be listed as; having less time complexity, O(n), being suitable for parallel programming, and being capable of adjusting selection pressure. One disadvantage of Tournament selection is the possibility of being trapped up in local minima if the fitness function has many local minimums.

Goldberg proposed Boltzmann Tournament selection which is inspired by Simulated Annealing (SA) that utilizes Boltzmann distribution and the temperature value to converge global optima in optimization problems [8]. Instead of paying attention the cooling process of SA, he actually used Boltzmann distribution in order to have a population which gradually improves towards the best solution.

For multimodal optimization problems in GAs, Restricted Tournament Selection was presented by Harik [9]. In this method, he added a window around randomly chosen mates in tournament selection. That window whose size is predefined shows the neighborhood of a selected individual. As in tournament selection, two individuals are randomly chosen, however, they did not mate as usual. Instead, a local search has been carried out within the 'windows' of the chosen individuals. Hence, it is expected that each mate would be improvised if there is a better solution found within the neighborhood.

Matsui introduced Correlative Tournament Selection to protect population diversity [10]. While the first parent is chosen by standard Tournament selection, the second parent is decided by correlative tournament selection. The algorithm randomly takes some number of individuals from the population. Then, it compares the values of these individuals that obtained by correlation function. An individual is awarded as second parent if the individual has better value than comparing other selected individuals.

De La Maza and Tidor offered Boltzmann selection [11]. The idea of the algorithm is based on the temperature value which controls selection pressure. The algorithm starts with high temperatures to make selection pressure low which gives selection chance not only to the best individuals, but also all the other individuals. Thereafter, it progressively decreases the temperature to increase selection pressure. This procedure facilitates the algorithm to investigate all search space at the beginning, then it is primarily focused to the better areas which were already found.

Local selection method was presented by Pohlheim [12]. The method is based on local neighborhood. In this method, each individual in a population can only mate with an individual that is located in its pre-defined neighborhood. However, it was observed that the method is unfeasible for small populations.

Goh et al. proposed Sexual selection that influenced by Darwinian approach on sexual selection [13]. Initially, the algorithm specifies females and males in a population. Females are decided either randomly or according to the specific knowledge of the problem. Moreover, male selection is performed as in standard tournament selection. The algorithm carries on until all females are mated.

The algorithm of High-Low fit selection is initiated by sorting individuals in the population [14]. After that, it divides the population into two sub groups: individuals having high fitness and individuals having low fitness. The method randomly selects two parents from each group, and mates them.

Yalkın and Korkmaz proposed a neural network based selection method [15]. First of all, the algorithm works as a standard GA. While working on, the algorithm collects data for selection mechanism. The process of collecting data is proceeded until appropriate data is obtained. Then, selection method is utilized. While the first mate is chosen by using tournament selection, the second mate is selected by using the neural network.

Another method, which is used in standard GAs, is age based replacement [16]. Instead of using better individuals having high fitness value, this method selects individuals according to their ages. The age of an individual means that the total time of the individual maintaining in the population. When a new individual participates to the population, the new individual is replaced with the oldest individual. However, this method may cause to lose the best individuals in the population [16].

3. Inspiring Works

3.1 Aggressive Selection Method

Basically, Aggressive Selection Method (ASM) is a method where an individual can only mate with another individual having better or equal fitness value comparing to individual itself [17]. For example, if the fitness value of an individual is 5, then, this individual can only mate with the individual having fitness value which must be less than 5 in minimization problems. Pseudo code of the algorithm is as follows;

```
count = 1
   while count <= population size
      Randomly choose first and second mate
      Determine costs of first and second mate

      If cost (second mate) <= cost (first mate)
         Second mate is chosen
         count = count+1;
      Else
         Reject mating
      end
end
```

3.2 Integrated Aggressive Selection Method

Integrated Aggressive Selection method (IASM) is hybridization of ASM and Tournament [17]. In this algorithm, the first mate is chosen by standard Tournament selection, and the second mate is selected by ASM. The process of IASM is described as follows:

```
count = 1
   while count <= population size
      Choose first mate via tournament
      Randomly choose second mate
      Determine costs of first and second mate
      If cost (second mate) <= cost (first mate)
         Individuals mate
         count = count+1;
      Else
         Reject mating
      end
end
```

The motivation behind ASM and IASM algorithms is to progressively improve population while permitting only the better individuals to qualify crossover step. For instance, it is assumed that the fitness values of four individuals in a population are respectively A, B, C and D where D<C<B<A. In a minimization problem, it is clear that D is the best individual in the population. Table 1 demonstrates that the worst individuals can only mate with better individuals while D can mate with all members of the population. Hence, it is apparently figured out that D has more chance than others for mating:

Table 1: Mating permissions of individuals in a population

Fitness Values	*1. (A)*	*2. (B)*	*3. (C)*	*4. (D)*
1. (A)	*Allowed*	*Allowed*	*Allowed*	*Allowed*
2. (B)	*Rejected*	*Allowed*	*Allowed*	*Allowed*
3. (C)	*Rejected*	*Rejected*	*Allowed*	*Allowed*
4. (D)	*Rejected*	*Rejected*	*Rejected*	*Allowed*

However, these methods, ASM and IASM, can cause transaction and time losses because of the rejection mechanism. That's why, the improved versions of ASM and IASM, NASM and INASM, respectively, were proposed [18]. In the improvised methods, a kind of Metropolis Algorithm [19] is employed to moderate the rigid selection scheme of ASM and IASM. Less rejections are achieved with enhanced searching capabilities.

3.3 Non-Aggressive Selection Method

Non-Aggressive Selection Method (NASM) is an improved version of ASM [18]. The only difference between ASM and NASM is that the second mate can be selected even if it does not have better fitness value than the first mate has. The algorithm works as follows:

```
count = 1
   while count <= population size
      Randomly choose first and second mate
      Determine costs of first and second mate
      Calculate the probability
      If cost (second mate) <= cost (first mate)
            Individuals mate
            count = count+1;
      Else
         If random value < the probability
            Individuals mate
            count = count+1;
         Else
         Reject mating
      end
end
```

3.4 Integrated Non-Aggressive Selection Method

Integrated Non-Aggressive Selection method (INASM), is the combination of Tournament and NASM selection methods [18]. In INASM, the first mate is selected by using standard Tournament selection, and the second mate is chosen by IASM. Pseudo code of INASM is;

```
count = 1
  while count <= population size
    Choose first mate via tournament
    Randomly choose second mate
    Determine costs of first and second mate
    Calculate the probability
    If cost (second mate) <= cost (first mate)
          Individuals mate
        count = count+1;
    Else
      If random value < the probability
        Individuals mate
        count = count+1;
      Else
        Reject mating
    end
  end
```

4. Proposed Method

The dazzling performance of Integrated Aggressive Selection Method (IASM), that uses extremely naïve approach to the selection problem is encouraging to attempt further improvements. Previous studies [17], [18] showed that selecting both mates by using Tournament selection is not as effective as selecting only one mate through Tournament. Incorporating random mating with Tournament, IASM proved to be better than its predecessors and the commonly used selection methods. While strict mating may cause early convergence to a local extremum, relaxed mate selection process provides greater search abilities to an evolutionary algorithm. One main dis-advantage of IASM is the transaction time loss due to the rejecting to mate. Every rejection has to be compensated with another mating attempt that may, still, result rejection. To overcome this problem, a new algorithm, "Outlander", is proposed for the selection of second mate.

As in IASM, the algorithm chooses the first mate by using standard Tournament selection. The difference occurs in selecting the second mate. In IASM, if the cost of second mate that is randomly selected from the population is lower than the first mate, they are allowed to mate, otherwise the first mate rejects the mating procedure. However, the proposed method does not give any rejection chance to the first mate. If the first mate rejects the second mate, a matchmaker suggests an "***outlander***" that is not in the population and randomly created just for the current mating process. The qualities of outlander are never evaluated and matchmaker's decision is undisputable. In this case, the first mate is forced to mate with an outlander. The pseudo-code for the algorithm works as follows:

```
count = 1
  while count <= population size
    Choose first mate via tournament
    Randomly choose second mate
    Determine costs of first and second mate
    If cost (second mate) <= cost (first mate)
      Second mate is chosen
    Else
      Create a new individual as the second mate
    End
    count = count+1;
end
```

5. Test Functions

A collection of continuous benchmark functions [20] that are grouped into two classes; unimodal and multimodal, are widely accepted by the researchers in the field. Unimodal class consists of sensitive functions that accomplish to converge slowly to the global extremum. Multimodal class includes functions having more than one local extremum. The performances of the proposed algorithm and its rivals are evaluated by using the benchmark test functions given in Table 2.

Table 2: Benchmark functions and their classes

Function	*Definition*	*Class*
f_1	*Ackley*	*Multimodal*
f_2	*Axis Parallel Hyper-Ellipsoid*	*Unimodal*
f_3	*Branins*	*Multimodal*
f_4	*De Jong*	*Unimodal*
f_5	*Goldstein-Price*	*Unimodal*
f_6	*Langermann*	*Multimodal*
f_7	*Rastrigin*	*Multimodal*
f_8	*Rosenbrock's Valley*	*Unimodal*
f_9	*Schwefel*	*Multimodal*
f_{10}	*Sum of Different Powers*	*Unimodal*

In Ackley's function, test area is usually restricted to -32.768 $\leq x_i \leq$ 32.768, i=1,...,n. Its global minimum f_1 = 0 is obtainable for x_i=0, i=1,…,n.

In Axis Parallel Hyper-Ellipsoid function, test area is usually restricted to -5.12 $\leq x_i \leq$ 5.12, $i = 1, \dots, n$. Its global minimum f_2 = 0 is obtainable for $x_i = 0, i = 1, \dots, n$.

In Branins's function, global minimum f_3 = 0,397887 is obtainable for three points: (-π, 12.275), (π, 2.275), (9.42478, 2.475).

In De Jong's function, test area is usually restricted to -5.12 $\leq x_i \leq$ 5.12, $i = 1, \dots, n$. Its global minimum f_4 = 0 is obtainable for $x_i = 0, i = 1, \dots, n$.

In Goldstein-Price's function, test area is usually restricted to -2 $\leq x_1 \leq$ 2, -2 $\leq x_2 \leq$ 2. f_5= 3 is obtainable at the point (0,-1).

In Langermann's function, local minimums are unevenly distributed.

In Rastrigin's function, test area is usually restricted to -5.12 $\leq x_i \leq$ 5.12, $i = 1, \ldots, n$. Its global minimum f_7 = 0 is obtainable for $x_i = 0, i = 1, \ldots, n$.

In Rosenbrock's Valley function, test area is usually restricted to -2.048 $\leq x_i \leq$ 2.048, $i = 1, \ldots, n$. Its global minimum f_8 = 0 is obtainable for $x_i = 0, i = 1, \ldots, n$.

In Schwefel's function, test area is usually restricted to -500 $\leq x_i \leq$ 500, $i = 1, \ldots, n$. Its global minimum f_9 = -418.9829n is obtainable for $x_i = 420.9687, i = 1, \ldots, n$.

In Sum of Different Powers function, test area is usually restricted to -1$\leq x_i \leq$1, $i = 1, \ldots, n$. Its global minimum $f_{10} = 0$ is obtainable for $x_i = 0, i = 1, \ldots, n$.

6. Results and Discussions

The comparative performance analysis of the proposed algorithm is carried out with the most commonly used standard selection methods; RW, LR and Tournament, and its predecessors; ASM, IASM, NASM, INASM. The tests were performed by using the same parameters for each algorithm such that crossover probability is 0.7, mutation probability is 0.05, and population sizes are 50 and 100. In order to decrease the effects of randomization and to increase the reliability and accuracy of the tests, each test is repeated 25 times with 25 different random seeds. Reported values are the mean, median, and standard deviation of the best results obtained in these 25 runs.

Table 3a-3j show the performance results of tested algorithms. Medians of 25 runs are utilized to decide which method is better comparing others. Means are not used since if one of the runs produces extremely high error value, it dominates the results of other runs. Standard deviation of these runs expresses the dispersion. For instance, if the standard deviation is small, it means that best results obtained from the runs are very close to each other.

Table 4 summarizes the best performing algorithms for each test function as well as overall success of each method. The most striking result is that the proposed algorithm, "Outlander", has the best overall performance for both unimodal functions (F2, F4, F5, F8 and F10) and multimodal functions (F1, F3, F6, F7 and F9). "Outlander" algorithm in small population size produced the best performances in three out of five unimodal functions. Even though, it resulted with the best performances in two out of five unimodal functions, that is the highest achieved ratio amongst the all functions. In both population size cases, "Outlander" is the clear winner with five out of ten cases for unimodal functions.

Its performance is even better for multimodal functions. "Outlander" produced the best results in four out of five cases for small population size and three out of five cases for large population size. In total, seven out of ten cases it was the best performing algorithm.

As we all know, multimodal functions are more challenging than unimodal functions. Forcing individuals to mate with "Outlanders" increases the searching abilities of genetic algorithm. It performs better, especially in small population sizes, that is quite advantageous in terms of computation and memory costs.

Table 3-a: Test results for Ackley function

f_1	Pop size = 50		Pop size = 100	
Method	Median	St. Dev.	Median	St. Dev.
ASM	1,03E+00	1,19E+00	3,39E-01	7,14E-01
NASM	3,14E-01	9,17E-01	6,75E-02	8,26E-01
IASM	1,15E-01	1,07E+00	4,62E-02	7,25E-01
INASM	1,95E-01	8,57E-01	**5,58E-06**	4,52E-02
OA	2,44E-01	4,04E-01	2,51E-02	8,79E-02
RW	1,61E-01	3,63E-01	1,87E-02	1,70E-01
LR	**8,93E-02**	8,43E-01	7,95E-03	3,03E-01
Tour.	2,31E-01	8,53E-01	4,51E-03	2,32E-01

Table 3-b: Test results for Axis Parallel Hyper-Ellipsoid

f_2	*Pop size = 50*		*Pop size = 100*	
Method	*Median*	*St. Dev.*	*Median*	*St. Dev.*
ASM	*2,84E-02*	*7,26E-01*	*1,17E-02*	*1,67E-01*
NASM	*4,64E-03*	*1,08E-01*	*1,92E-03*	*3,12E-02*
IASM	*6,02E-03*	*1,61E-01*	***2,68E-08***	*3,78E-03*
INASM	*1,46E-03*	*7,01E-02*	*1,92E-06*	*2,54E-02*
OA	***1,26E-03***	*6,02E-03*	*6,67E-05*	*1,30E-03*
RW	*2,24E-02*	*7,47E-02*	*1,21E-03*	*3,56E-02*
LR	*2,82E-03*	*5,18E-02*	*3,43E-04*	*2,74E-02*
Tour.	*1,29E-02*	*1,66E-01*	*5,91E-05*	*1,76E-02*

Table 3-c: Test results for Branins function

f_3	*Pop size = 50*		*Pop size = 100*	
Method	*Median*	*St. Dev.*	*Median*	*St. Dev.*
ASM	*4,05E-01*	*8,24E-02*	*3,99E-01*	*6,59E-03*
NASM	*3,99E-01*	*4,22E-02*	***3,98E-01***	*2,83E-03*
IASM	***3,98E-01***	*3,03E-02*	***3,98E-01***	*2,27E-03*
INASM	***3,98E-01***	*7,94E-03*	***3,98E-01***	*1,57E-04*
OA	***3,98E-01***	*8,28E-02*	***3,98E-01***	*2,93E-03*
RW	***3,98E-01***	*4,95E-02*	***3,98E-01***	*1,12E-02*
LR	*3,99E-01*	*1,19E-02*	***3,98E-01***	*1,83E-03*
Tour.	*4,03E-01*	*5,64E-02*	***3,98E-01***	*5,96E-03*

Table 3-d: Test results for De Jong function

f_4	*Pop size = 50*		*Pop size = 100*	
Method	*Median*	*St. Dev.*	*Median*	*St. Dev.*
ASM	*1,39E-02*	*1,86E-02*	*9,06E-04*	*8,38E-03*
NASM	*2,59E-04*	*6,05E-03*	*1,51E-05*	*4,59E-04*
IASM	*4,91E-04*	*1,10E-02*	*1,62E-08*	*3,38E-04*
INASM	*1,06E-04*	*3,27E-03*	***7,75E-11***	*2,37E-05*
OA	*1,72E-04*	*8,37E-04*	*2,61E-05*	*8,29E-05*
RW	*1,22E-03*	*1,46E-02*	*6,93E-05*	*1,13E-03*
LR	***8,05E-05***	*1,53E-03*	*3,94E-06*	*5,04E-04*
Tour.	*1,26E-03*	*7,73E-03*	*5,40E-06*	*4,53E-04*

Table 3-e: Test results for Goldstein-Price function

f_5	*Pop size = 50*		*Pop size = 100*	
Method	*Median*	*St. Dev.*	*Median*	*St. Dev.*
ASM	*3,49E+00*	*1,41E+00*	*3,04E+00*	*5,51E-01*
NASM	*3,11E+00*	*9,59E+00*	***3,00E+00***	*7,80E-01*
IASM	*3,11E+00*	*6,16E+00*	***3,00E+00***	*3,05E+00*
INASM	*3,01E+00*	*1,09E-01*	***3,00E+00***	*1,41E-02*
OA	***3,00E+00***	*2,32E-02*	***3,00E+00***	*9,57E-03*

Paper ID: ART20178369 DOI: 10.21275/ART20178369

RW	*3,03E+00*	*9,27E+00*	***3,00E+00***	*5,40E+00*
LR	*3,04E+00*	*3,35E+00*	*3,01E+00*	*3,12E+00*
Tour.	*3,19E+00*	*1,01E+01*	*3,01E+00*	*8,35E-02*

Table 3-f: Test results for Langermann function

f_6	*Pop size = 50*		*Pop size = 100*	
Method	*Median*	*St. Dev.*	*Median*	*St. Dev.*
ASM	*-4,04E+00*	*4,83E-01*	*-4,09E+00*	*3,70E-01*
NASM	*-4,02E+00*	*5,63E-01*	*-4,12E+00*	*8,43E-02*
IASM	*-4,02E+00*	*4,23E-01*	***-4,10E+00***	*1,01E-01*
INASM	*-4,08E+00*	*6,87E-02*	***-4,10E+00***	*3,88E-01*
OA	***-4,10E+00***	*5,20E-02*	*-4,13E+00*	*2,13E-02*
RW	*-3,03E+00*	*6,90E-01*	*-3,66E+00*	*3,95E-01*
LR	*-4,04E+00*	*6,97E-01*	*-4,12E+00*	*2,05E-01*
Tour.	*-4,03E+00*	*3,79E-01*	*-4,08E+00*	*4,67E-02*

Table 3-g: Test results for Rastrigin function

f_7	*Pop size = 50*		*Pop size = 100*	
Method	*Median*	*St. Dev.*	*Median*	*St. Dev.*
ASM	*1,47E+00*	*6,52E-01*	*1,01E+00*	*7,11E-01*
NASM	*9,96E-01*	*4,18E-01*	*9,95E-01*	*6,13E-01*
IASM	*2,08E-01*	*4,08E-01*	***0,00E+00***	*2,67E-01*
INASM	*1,22E-01*	*3,94E-01*	*8,42E-07*	*1,98E-01*
OA	***0,00E+00***	*9,47E-02*	***0,00E+00***	*0,00E+00*
RW	*3,12E-02*	*3,90E-01*	*1,65E-02*	*7,14E-02*
LR	*1,44E-01*	*4,41E-01*	*1,30E-11*	*3,33E-01*
Tour.	*1,39E-01*	*4,43E-01*	*2,87E-02*	*4,96E-02*

Table 3-h: Test results for Rosenbrock's Valley function

f_8	*Pop size = 50*		*Pop size = 100*	
Method	*Median*	*St. Dev.*	*Median*	*St. Dev.*
ASM	*3,25E-02*	*1,07E-01*	*7,07E-03*	*6,28E-02*
NASM	*1,31E-02*	*9,97E-02*	*7,49E-03*	*6,21E-02*
IASM	*1,21E-02*	*1,22E-01*	*3,26E-04*	*8,82E-02*
INASM	*4,06E-02*	*1,63E-01*	*1,39E-02*	*9,45E-02*
OA	***1,73E-03***	*7,22E-03*	***6,91E-05***	*5,07E-04*
RW	*6,00E-02*	*1,35E-01*	*1,79E-02*	*1,91E-01*
LR	*4,21E-02*	*1,69E-01*	*1,44E-02*	*1,36E-01*
Tour.	*5,02E-02*	*1,33E-01*	*6,96E-03*	*8,21E-02*

Table 3-i: Test results for Schwefel function

f_9	*Pop size = 50*		*Pop size = 100*	
Method	*Median*	*St. Dev.*	*Median*	*St. Dev.*
ASM	*-7,16E+02*	*9,34E+01*	*-8,20E+02*	*6,71E+01*
NASM	*-8,25E+02*	*8,01E+01*	*-8,36E+02*	*4,24E+01*
IASM	*-8,37E+02*	*8,43E+00*	***-8,38E+02***	*1,62E+00*
INASM	***-8,38E+02***	*1,41E+01*	***-8,38E+02***	*1,91E-01*
OA	***-8,38E+02***	*1,59E+00*	***-8,38E+02***	*1,52E-01*
RW	*-6,84E+02*	*9,49E+01*	*-6,78E+02*	*8,51E+01*
LR	***-8,38E+02***	*1,39E+01*	***-8,38E+02***	*7,01E-01*
Tour.	***-8,38E+02***	*3,33E+00*	*-8,20E+02*	*6,71E+01*

Table 3-j: Test results for Sum of Different Powers function

f_{10}	*Pop size = 50*		*Pop size = 100*	
Method	*Median*	*St. Dev.*	*Median*	*St. Dev.*
ASM	*3,34E-06*	*2,10E-04*	*8,39E-07*	*2,75E-05*
NASM	*7,36E-08*	*1,28E-05*	*7,15E-09*	*1,17E-06*
IASM	*1,08E-07*	*2,55E-05*	*1,86E-09*	*6,43E-07*
INASM	*1,04E-07*	*1,25E-06*	***5,65E-12***	*9,79E-08*
OA	*3,67E-08*	*2,08E-06*	*2,54E-09*	*4,00E-07*
RW	*5,42E-07*	*1,21E-05*	*1,85E-07*	*5,13E-06*
LR	***2,47E-08***	*3,15E-05*	*4,66E-10*	*2,77E-07*
Tour.	*2,15E-06*	*1,62E-05*	*9,91E-10*	*2,54E-07*

Table 4: The Best Performing Algorithm in Each Test Methods

		Selection Methods															
		ASM		NASM		IASM		INASM		OA		RW		LR		Tour.	
		L	H	L	H	L	H	L	H	L	H	L	H	L	H	L	H
Unimodal	f_2						✓			✓							
	f_4							✓						✓			
	f_5				✓		✓		✓	✓	✓		✓				
	f_8									✓	✓						
	f_{10}								✓					✓			
Sub Total		0	0	0	1	0	**2**	1	**2**	**3**	2	0	1	**2**	0	0	0
Total		0		1		2		3		**5**		1		2		0	
Multimodal	f_1								✓					✓			
	f_3				✓	✓	✓	✓	✓	✓	✓	✓	✓		✓		✓
	f_6						✓		✓	✓							
	f_7						✓			✓	✓						
	f_9						✓	✓	✓	✓	✓			✓	✓	✓	
Sub Total					1	1	**4**	2	**4**	**4**	3	1	1	2	2	1	1
Total		0		1		5		6		**7**		2		4		2	
Grand Total		0		2		7		9		**12**		3		6		2	

7. Conclusions

Selection mechanism of an evolutionary algorithm has a great impact on the optimization performance. Selection algorithms suffer from dominant individuals as well as population diversity loss. In order to maintain exploration-exploitation balance, diversity should be maintained and mating of individuals with better traits should be promoted right from the beginning of the search process until to the end. A new selection algorithm “Outlander” is proposed to satisfy these needs. The performance tests revealed that “Outlander” is the best selection algorithm amongst the ones tested in this work. This encourages us to carefully inspect the selection behavior of “Outlander” and improve it further in the future

8. Acknowledgement

This work was supported by Research Fund of Çukurova University (project no: 8463), Turkey.

References

[1] J. Holland, Adaptation in Natural and Artificial Systems: An introductory analysis with applications to biology, control, and artificial intelligence, The University of Michigan Press, viii 183 pp., 1975.
[2] D.E. Goldberg, Genetic Algorithms in Search Optimization and Machine Learning, Addison-Wesley Longman Publishing Co., 1989.
[3] R.J. Kennedy, Eberhart, “Particle swarm optimization”, Proceedings of IEEE International Conference on Neural Networks IV, 1995.
[4] M. Dorigo, M.G. Luca, " Ant Colony system: A Cooperative learning approach to the Travelling Salesman Problem”, IEEE transaction on evolutionary computation, Vol. 1, No. 1, 1997
[5] M. Mitchell, An Introduction to Genetic Algorithms, A Bradford Book The MIT Press, 1996.
[6] J.J. Grefenstette, J.E. Baker, “How Genetic Algorithms Work: a Critical Look at Implicit Parallelism”, Proceedings of the Third International Conference on Genetic Algorithms, pages 20-27, 1989.
[7] T. Blickle, L. Thiele, A Comparison of Selection Schemes used in Genetic Algorithms, TIK Report No. 11, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH), 1995.
[8] D.E. Goldberg, A note on Boltzmann Tournament selection for genetic algorithms and population-oriented simulated annealing. Complex Systems 4.4, pages 445-460, 1990.
[9] G.R. Harik, Finding Multimodal Solutions Using Restricted Tournament Selection. ICGA., 1995.
[10] K. Matsui, “New selection method to improve the population diversity in genetic algorithms. In: Systems, Man, and Cybernetics”, IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on. IEEE, p. 625-630, 1999.
[11] De La Maza, B. Tidor, “Increased Flexibility in Genetic Algorithms: The Use of Variable Boltzmann Selective Pressure to Control Propagation”, Proceeding of ORSA Conference: Computer Science and Operations Research: New Developments in Their Interfaces, 1991.
[12] H. Pohlheim, The Multipopulation Genetic Algorithm: Local selection and Migration, Technical report, Technical University Ilmenau.M, 1995.
[13] K.S. Goh, A. Lim, B. Rodrigues, Sexual selection for genetic algorithms. Artificial Intelligence Review, 19(2), 123-152, 2003
[14] A. Elnima, E. Elamin, A proposed genetic algorithm selection method, 2006.
[15] C. Yalkın, E.E. Korkmaz, “Neural Network World: A Neural Network Based Selection Method for Genetic Algorithms”, International Journal on Non-Standard Computing and Artificial Intelligence, 495-510 pp., 2012.
[16] A.E. Eiben, J.E. Smith, Introduction to Evolutionary Computing, Springer, 2015.
[17] M. Oral, M.C. Gençal, “A Novel Selection Methods for Genetic Algortihms”, Proceeding of IMSEC: International Mediterranean Science and Engineering Congress, 254-259 pp., 2016.
[18] M. Oral, M.C. Gençal, “Genetik Algoritma İçin İyileştirilmiş Saldırgan ve Bütünleşik Saldırgan Seçim Yöntemleri”, Proceeding of UBMK: 2nd International Conference on Computer Sciences and Engineering, 320-325 pp., 2017.
[19] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, Equation of State Calculation by Fast Computing Machines, J. of Chem. Phys., 21, 1087-1091, 1953
[20] M. Molga, C. Smutnicki, Test Functions for Optimization Needs, 2005.