Some hybrid models to improve Firefly algorithm performance

SH. MASHHADI FARAHANI[1], A. AMIN ABSHOURI[2], B. NASIRI[3], M. R. MEYBODI[4]

[1]Department of electronic, Computer and IT, Islamic Azad University
Iran, Qazvin
Email: Farahani.ce@gmail.com

[2]Department of electronic, Computer and IT, Islamic Azad University
Iran, Qazvin
Email: A_amin1980@yahoo.com

[3]Department of electronic, Computer and IT, Islamic Azad University
Iran, Qazvin
Email: nasiri.babak@qiau.ac.ir

[4]Department of Computer engineering and IT, Amirkabir University of technology
Iran, Tehran
Email: Mmeybodi@aut.ac.ir

ABSTRACT

Firefly algorithm is one of the evolutionary optimization algorithms, and is inspired by the behavior of fireflies in nature. Though efficient, its parameters do not change during iterations, which is also true for particle swarm optimization. This paper propose a hybrid model to improve the FA algorithm by introducing learning automata to adjust firefly behavior, and using genetic algorithm to enhance global search and generate new solutions. We also propose an approach to stabilize firefly movement during iterations. Simulation results show better performance and accuracy than standard firefly algorithm.

Keywords: Firefly algorithm, Genetic algorithm, Learning Automata, Optimization.

1. INTRODUCTION

The meaning of optimization is finding a parameter in a function that makes a better solution. All of suitable values are possible solutions and the best value is optimum solution [1]. Often to solve optimization problems, optimization algorithms are used. Classification of optimization algorithm can be carried out in many ways. A simple way is looking at the nature of the algorithms, and this divides the algorithms into two categories: deterministic algorithm, and stochastic algorithms. Deterministic algorithms follow a rigorous procedure, and its path and values of both design variables and the functions are repeatable. For stochastic algorithms, in general we have two types: heuristic and metaheuristic. Nature-inspired metaheuristic algorithms are becoming powerful in solving modern global optimization problems. All metaheuristic algorithms use certain tradeoff between randomization and local search [2], [3], and [4].

Heuristic algorithms that have ever defined are inspired by nature. These strong algorithms are used for solving NP-hard problems such as travelling salesman problem (TSP) [2]. Optimization algorithms cover all maximization and minimization problem. These kinds of algorithms work on a population of solutions and always search optimum solutions [5]. One of these heuristic algorithms is firefly algorithm that is inspired by firefly behavior in nature. Fireflies are one of the most special and fascinating creatures in nature. These nocturnal luminous insects of the beetle family lampyridae, inhabit mainly tropical and temperate regions, and their population is estimated at around 1900 species [6]. They are capable of producing light thanks to special photogenic organs situated very close to the body surface behind a window of translucent cuticle [7].

Bioluminescent signals are known to serve as elements of courtship rituals, methods of prey attraction, social orientation or as a warning signal to predators. The phenomenon of firefly glowing is an area of continuous research considering both its biochemical and social aspects [8] [9]. Mechanism of firefly communication via luminescent flashes and their synchronization has been imitated effectively in various techniques of wireless network design [10] and mobile robotics [11].

In order to improve Firefly algorithm in static problems, an approach has been proposed [12], which includes a Lévy flight Firefly algorithm introducing a new distribution to change the movement of Firefly algorithm.

Firefly algorithm is powerful in local search but sometimes it may trap into several local optimums as result it cannot search globally well. Firefly algorithm parameters may not change by the time during iterations. Two parameters of the algorithm are attractiveness coefficient and randomization coefficient. The values are crucially important in determining the speed of the convergence and the behavior of FA algorithm.

Learning Automata are adaptive decision-making devices that operating in an unknown random environment and progressively improve their performance via a learning process. It has been used successfully in many applications such as call admission control in cellular networks [13] and [14], capacity assignment problems [15], Adaptation of back propagation parameter [16], and Determination of the number of hidden units for three layers neural networks [17]. In order to make adaptive parameters of firefly and deal with fixed or random parameter, one of the proposed algorithms is to set parameters of firefly by means of two Learning Automata; one Learning Automata for absorption coefficient and another one for randomization coefficient.

Genetic algorithm searches the solution space of a function through the use of simulated evolution, i.e. the survival of the fitness strategy. In general, the fitness individuals of any population tend to reproduce and survive to the next generation, thus improves successive generations. These algorithms do the search by its special operation [5]. In order to enhance global search and generate new solutions in Firefly algorithm, one of the proposed algorithms is combination of genetic algorithm with firefly algorithm as a new generation which may find better solutions and make a balance between global and local search. Also it can get rid of trapping in to several local optimums. To stabilize fireflies movement, it is proposed a new model that uses Guassian distribution to direct fireflies to global best better. Proposed approaches are tested on five standard benchmarks that have ever been used to evaluate optimization algorithm in static continuous problems. Simulation results show better performance and accuracy than standard Firefly algorithm, PSO algorithm and derivatives of PSO.

In the rest of the paper the following materials are provided. Section 2 gives a brief introduction to standard Firefly algorithm, Learning Automata and Genetic algorithm. The proposed algorithms are given in section 4. Experiments settings and results are presented in section 5. Section 6 concludes the paper.

2. Definition of basic concepts

This section introduces applied algorithms including; Firefly algorithm, explores standard Firefly algorithm, Learning Automata and Genetic algorithm.

2.1. Standard Firefly Algorithm

The Firefly algorithm was developed by Xin-She Yang [3], [18] and it is based on idealized behavior of the flashing characteristics of fireflies. For simplicity, we can summarize these flashing characteristics as the following three rules:
All fireflies are unisex, so that one firefly is attracted to other fireflies regardless of their sex.
Attractiveness is proportional to their brightness, thus for any two flashing fireflies, the less bright one will move towards the brighter one. The attractiveness is proportional to the brightness and they both decrease as their distance increases. If no one is brighter than a particular firefly, it will move randomly.
The brightness of a firefly is affected or determined by the landscape of the objective function to be optimized [19], [12].
Assume continuous optimization problem where the task is to minimize cost function f(x) for $x \in S \subset R^n$ i.e. find x^* such as:

$$f(x^*) = \min_{x \in S} f(x) \qquad (1)$$

For solving an optimization problem by Firefly algorithm iteratively, there is a swarm of m agents (fireflies) and x_i represents a solution for firefly i in whereas $f(x_i)$ denotes its cost.
Initially all fireflies are dislocated in S (randomly or employing some deterministic strategy). $S_k(k = 1, \dots, d)$ In the d dimensions should be determined by the actual scales of the problem of interest. For simplicity we can assume that the attractiveness of a firefly is determined by its brightness or light intensity which in turn is associated with the encoded objective function. In the simplest case for an optimization problem, the brightness I of a firefly at a particular position x can be chosen as$I(x) \alpha f(x)$. However, the attractiveness β is relative, it should vary with the distance r_{ij} between firefly i and firefly

j. As light intensity decreases with the distance from its source and light is also absorbed in the media, so we should allow the attractiveness to vary with degree of absorption [19], [12].
The light intensity $\mathrm{I(r)}$ varies with distance r monotonically and exponentially. That is:

$$\mathrm{I} = \mathrm{I_0 e^{-\gamma r}}, \qquad (2)$$

Where $\mathrm{I_0}$ the original light intensity and γ is is the light absorption coefficient. As firefly attractiveness is proportional to the light intensity seen by adjacent fireflies, we can now define the attractiveness β of a firefly by Eq (3) [4], [19].

$$\beta = \beta_0 e^{-\gamma r^2} \qquad (3)$$

Where r is the distance between each two fireflies and β_0 is their attractiveness at r= 0 i.e. when two fireflies are found at the same point of search space S [7], [11]. In general $\beta_0 \in [0,1]$ should be used and two limiting cases can be defined: when $\beta_0 = 0$, that is only non-cooperative distributed random search is applied and when $\beta_0 = 1$ which is equivalent to the scheme of cooperative local search with the brightest firefly strongly determining other fireflies positions, especially in its neighborhood[3].
The value of γ determines the variation of attractiveness with increasing distance from communicated firefly. Using γ=0 corresponds to no variation or constant attractiveness and conversely setting γ→∞ results in attractiveness being close to zero which again is equivalent to the complete random search. In general $\gamma \in [0,10]$ could be suggested [3].
It is worth pointing out that the exponent γr can be replaced by other functions such as γr^m when $m > 0$. The distance between any two fireflies i and j at x_i and x_j can be Cartesian distance in Eq (4).

$$r_{ij} = \|x_i - x_j\|_2 = \sqrt{\sum_{k=1}^{d}(x_{i,k} - x_{j,k})^2} \qquad (4)$$

The firefly i movement is attracted to another more attractive (brighter) firefly j is determined by:

$$x_i = x_i + \beta_0 e^{-\gamma r_{ij}^2}(x_j - x_i) + \alpha\varepsilon_i, \qquad (5)$$

Where the second term is due to the attraction, while the third term is randomization with the vector of random variable ε_i being drawn from a Gaussian distribution and (α Є [0,1]) [20], [19]. In [12] uses a Lévy distribution instead of Gaussian one. Schematically, the Firefly algorithm can be summarized as the pseudo code in pseudo code 1.

```
1.  Objective function f(x), x=(x1,x2,…,xd)^T
2.  Initialize a population of fireflies $x_i (i = 1,2,..,n)$
3.  Define light absorption coefficient γ
4.  While (t<MaxGeneration)
5.     For i=1:n    (all n fireflies)
6.        For j=1:i
7.           Light intensity $I_i$ at $x_i$ is determined by $f(x_i)$
8.           If ($I_i > I_j$)
9.               Move firefly i towards j in all d dimensions    (Apply Eq (5))
10.          Else
11.              Move firefly i randomly
12.          End if
13.       Attractiveness varies with distance r via exp[$-\gamma r^2$]        ($\beta = \beta_0 e^{-\gamma r_{ij}^2}$)
14.       Evaluate new solutions and update light intensity
15.       End for j
16.    End for i
17.    Rank the fireflies and find the current best
18. End while
19. Postprocess results and visualization.
```

Pseudo code 1- standard Firefly Algorithm

2.2. Learning Automata

Learning Automata is adaptive decision-making devices operating on unknown random environments [21]. The Learning Automaton has a finite set of actions and each action has a certain probability (unknown for the automaton) of getting rewarded by the environment of the automaton. The aim is to learn choosing the optimal action (i.e. the action with the highest probability of being rewarded) through repeated interaction on the system. If the learning algorithm is chosen properly, then the iterative process of interacting on the environment can be made to select the optimal action.
Figure 1 illustrates how a stochastic automaton works in feedback connection with a random environment. Learning Automata can be classified into two main families: fixed structure learning automata and variable structure learning automata (VSLA) [22]. In the following, the variable structure learning automata is described.

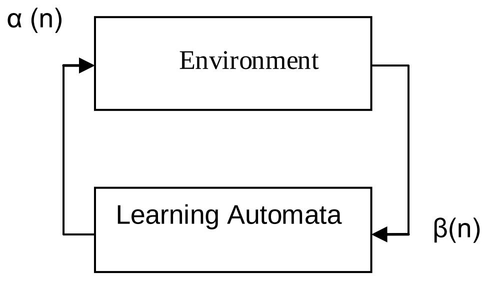


Figure 1- The interaction between learning automata and environment

Variable structure of Learning Automata can be shown by a quadruple { α , β, p, T } where α={α1, α2, ..., α_r } which is the set of actions of the automaton, β={β_1, β_2,…, β_m} is its set of inputs; p={p_1, ..., p_r} is probability vector for selection of each action, and p(n +1) = T[α(n),β(n), p(n)] is the learning algorithm. If β = {0,1}, then the environment is called P-Model. If β belongs to a finite set with more than two values, between 0 and 1, the environment is called Q-Model and if β is a continuous random variable in the range [0, 1] the environment is called S-Model. Let a VSLA operate in a SModel environment. A general linear schema for updating action probabilities when action i is performed is given by:

$$P_i(n+1) = P_j(n) + a.(1 - p_i(n)) - (b.\beta_i(n).pi(n)$$
$$P_j(n+1) = P_j(n) + a.(1 - \beta_i(n)).Pj(n) + (b.\beta_i(n))[1/(r-1) - p_j(n)] \quad \forall j \ j \neq i \qquad (6)$$

Where a and b are reward and penalty parameters. When a=b, the automaton is called S-LR-P. If b=0 and 0<b<<a<1, the automaton is called S-LR-I and S-LRϵP, respectively [22].

2.3. Genetic Algorithm

Genetic algorithms maintained and manipulate a family or population of solutions and implement a survival of the fitness strategy in their search for better solutions. This provides an implicit as well as explicit parallelism that allows for the exploitation of several promising areas of the solution space at the same time. The implicit parallelism is due to the schema theory developed by Holland, while the

explicit parallelism arises from the manipulation of a population of points. The evaluation of the fitness of these points is easy to accomplish in parallel [5].
Genetic algorithms have been shown to solve linear and non-linear problems by exploring all regions of the state space and exponentially exploiting promising areas through Mutation, Crossover and Selection operations applied to individuals in the populations [5].
The use of Genetic algorithm requires the determination of six fundamental issues: chromosome representation, selection function, and the genetic operators making up the reproduction function, the creation of the initial population, termination criteria, and the evaluation function.
For any genetic algorithms a chromosome representation is needed to describe each individual in the population of interest. The representation scheme determines how the problem is structured in the GA and also determines the genetic operators that are used. Each individual or chromosome is made up of a sequence of genes from certain alphabet. An alphabet could consist of binary digits (0 and 1), floating point numbers, integers, symbols (i.e., A, B, C, D), matrices, etc. [23].
Genetic algorithm provides the basic search mechanisms of the GA. Selection functions that are used for GA are: tournament selection, elitism and ranking. For any kind of algorithms and their application one of these kind of that is used. The operators are used to create new solutions based on existing solutions in the population. There are two basic types of operators, Crossover and Mutation. Crossover takes two individuals and produces two new individuals while Mutation alters one individual to produce a single new solution. The application of these two basic types of operators and their derivation depends on the chromosome representation used. In [24] all of the kind of these operators is introduced.
The most common method is to generate solutions for the entire population randomly. However, since GAs can iteratively improve existing solutions, the beginning population can be seeded with potentially good solutions, with the remainder of the population being randomly generated solutions. Starting and termination can define as maximum iteration, not getting better solutions and convergence and arriving threshold [5], [25]. Pseudo code of GA is shown in pseudo code 2.

```
1. Gen=0
2. Pop (gen) =randomly initialized population
3. Evaluate fitness of each individual in Pop(gen)
4. While termination condition = false do begin
5.    Gen = Gen+1;
6.    Select Pop (gen) from Pop (gen-1) based on fitness
7.    Apply genetic operators to Pop (gen)
8.    Evaluate fitness of each individual in Pop(gen)
9. end
```

Pseudo code 2- Standard Genetic Algorithm

3. Proposed Algorithms

Previous studies have shown that problem based tuning of parameters in Firefly algorithm is a key factor to find the optimum solution accuracy and efficiently. As mentioned above one of the considerable issue is to improve exploration property to do global search in the landscape and prevent to miss better position. In this section, new Learning Automata schemes and Genetic algorithm based Firefly algorithm is presented. The proposed algorithms are classified in three

classes: in the first class, named LA-FF (Learning Automata-Firefly Algorithm), an adaptive parameter is presented, in the second class, named HGA-FF (Hybrid Genetic and Firefly Algorithm), we can improve global search and fireflies share their information in both populations. Last class applies Gaussian distribution and a directed movement for each firefly in landscape.
In entire the optimization algorithms, initial parameters are constant from beginning to end of iterations and their best values can be achieved by going through try and error. In order to deal with varying optimization problems which can be applied to the real world problems, it is better to introduce a way to adapt parameters by the time. In standard Firefly algorithm, movement of firefly is defined by α parameter in third part of Eq (5) that is a random step with fixed length. Due to firefly's movement, the algorithm will miss better local search capabilities. In proposed algorithm, it is defined a coefficient for α that depends on iteration and it always produce a value less than one. This coefficient is determined by:

$$\mathrm{W}_{\mathrm{itr}} = \mathrm{X} + \frac{(\mathrm{itr}_{\max} - \mathrm{itr})^{\mathrm{n}}}{(\mathrm{itr}_{\max})^{\mathrm{n}}} + (\mathrm{Y} - \mathrm{X}), \qquad (7)$$

Where n>=1. In Eq (7), weight of $\mathrm{W}_{\mathrm{itr}}$ is defined based on current iteration number and the last number of iteration. The values that produce by this equation is between X and Y, and reduces by the time. Because α Є [0, 1], so X=0 and Y=1. n could be a linear or non-linear coefficient and $\mathrm{itr}_{\max}$ is maximum number of iteration and itr is iteration i [31]. This causes that α fixed coefficient changes by the time and step length shrinks and the algorithm can get better result in local search.

3.1. LA-FF (Learning Automata –Firefly Algorithm)

In this section, we proposed two new firefly algorithms based on Learning Automata. In the proposed algorithms like firefly algorithm, there is population of firefly that each has an initial position.
In standard firefly algorithm, for any two flashing fireflies, the less bright one will move towards the brighter, which includes a percentage of randomness. The initial percentage value and other parameter such as attractiveness coefficient is crucially important in determining the speed of the convergence and how the FA algorithm behaves. In mode standard, the parameters are fixed from beginning to end. Alpha determines random percentage in firefly moving. It includes value between zero to one. Attractiveness coefficient is named Gamma. The parameter varies between zero to extreme. If absorption coefficient is close to zero, then β0= β and this corresponds to a special case of particle swarm optimization. Besides, if absorption coefficient is close extremely, this is the case where the fireflies fly in a very foggy region randomly.
In this section, we proposed two algorithms based on learning automata. In the first proposed algorithm Firefly-LA that is called LA1-FF, absorption coefficient is set by one Learning Automata. The second proposed algorithm that is called LA2-FF, the firefly parameters are set by two Learning Automatas, one Learning Automata represents absorption coefficient and the other Learning Automata represents randomization coefficient.

3.1.1 LA1-FF

In LA1-FF algorithm, absorption coefficient is set by one Learning Automata.
This Learning Automata actually is set according to [9]; it means, in each iteration, the value of parameter is set as:

I. Is unchanged
II. Is increased by multiplying a number greater than one

III. Is increased by multiplying a number smaller than one. This number is generated Based on the current repetition number by formula 7.

In other words, in any iteration, the learning automaton functions three actions; decrease, increase and unchanged gamma parameter, therefore, any action determines the parameter value related this value in previous iteration. In first action, this parameter value is equal to this in previous iteration. In second action, this parameter in current iteration is increased by multiplying a number greater than one. In the third action, to decrease this parameter; the value in previous iteration is multiplied by a number which is smaller than one.

In these model, probabilities of learning automata actions change based on 6 states, in the learning automata's problem, environment with input α change probability of any action as follows:

a. If we observe improvement in more than 83% of the population, then selection probability of this action increases three times and selection probabilities of other actions decrease so;
b. If the observed improvement is between 66-83% of population, then selection probability of this action increases two times and selection probabilities of other actions decrease so;
c. If the observed improvement is between 50-66% of population, then selection probability of this action increases and selection probabilities of other actions decrease;
d. If we observe improvement in less than 16% of the population, then selection probability of this action decreases three times and selection probabilities of other actions increase so;
e. If the observed improvement is between 16-33% of population, then selection probability of this action decreases two times and selection probabilities of other actions increase so;
f. If the observed improvement is between 33-50% of population, then selection probability of this action decreases and selection probabilities of other actions increase.

Learning Automata environment considers P model and it is applied from learning algorithm $p(n+1)=T[\alpha (n),\beta(n),p(n)]$. Increasing or decreasing of selection probability is according to equation 6.

3.1.2 LA2-FF

In other proposed algorithm Firefly-LA, LA1-FF algorithm, the firefly parameters are set by means of two Learning Automata, one Learning Automata represents absorption coefficient and second Learning Automata represents randomization coefficient. Both of Learning Automata actually are set according to [9]; it means, in any iteration, the value of any parameter is set as:

I. Is unchanged;
II. Is increased by multiplying a number greater than one;
III. Is increased by multiplying a number smaller than one. This number is generated Based on the current repetition number by formula 7.

Actually both of learning automata of the algorithm act like the first Firefly-algorithm.

Pseudo code of LA-FF is shown in pseudo code 3.

```
1.  Objective function F(x), x=(x1,…,xd)T
2.  Initialize a population of fireflies xi(i=1,2,…,n)
3.  Define light absorption coefficient $\gamma$
4.  **While** (t< Max Generation)
5.      Initialize the LA
6.      **For** i=1:n all n fireflies
7.          Light intensity Ii at Xi is determined by f(xi)
8.          The LA selects an action for gamma
```

```
9.        The LA selects an action for alpha
10.       If(Ij>Ij)
11.           Move firefly i toward j in all d dimensions
12.       End if
13.       Attractiveness varies with distance r via exp[-r]
14.       Evaluate  new solutions and update light intensity
15.       Evaluate action and return a reinforcement signal β
16.       Update probability vector using learning rule
17.     End for j
18.     Rank the fireflies and find the current best
19. End while
20. Post process result and visualization
```

Pseudo code 3 – LA-FF Algorithm

3.2. HGA-FF (Hybrid Genetic algorithm and Firefly algorithm)

Another discussed issue of Firefly algorithm is weakness in exploring search space. In order to deal with this problem, we used nature of Genetic algorithm for more desirable global search. Genetic algorithm is suitable in exploring search space and find new better solutions. In this new model we use a co-evolutionary algorithm that each one of them has their own populations. At the end of each iteration they swap their populations. In this new model, GA does search globally and find better solutions than last iteration and create new population. After finishing, GA firefly algorithm will search locally in created population of GA. We can show the steps of new HGA-LL as follows:

```
1. Initialize two populations for genetic and firefly algorithm
2. Run firefly and genetic algorithm concurrently
3. Evaluate new solution
4. Check termination criteria
5. If  termination criteria = false
6.     Change populations
7. else
8.     Go to step 2
9. End if
```

Pseudo code 4- HGA-FF Algorithm

In proposed model, in GA steps, new generations which are produced by new algorithm cost better than their parents; then, this causes next iterations to have better solutions. As mentioned above, we can use different selection methods, so in this model we use tournament technique.

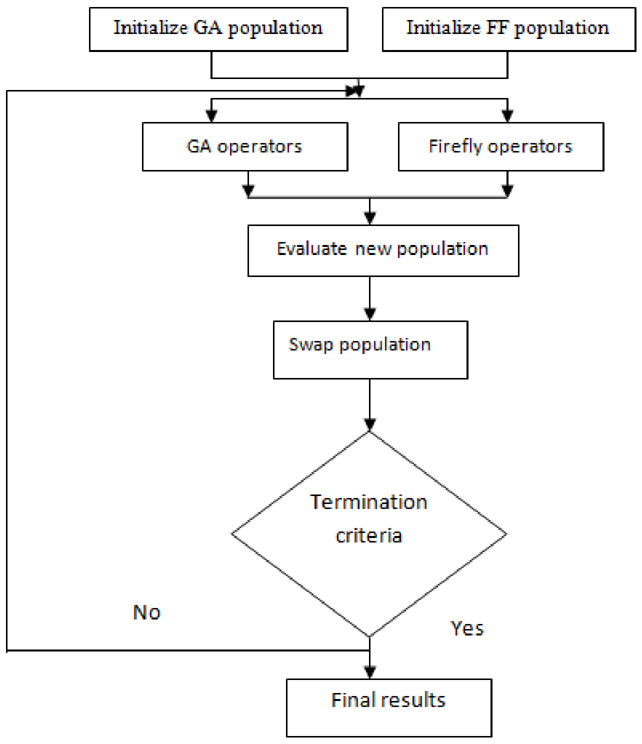


Figure 2 Flowchart of new model

3.3. GD-FF (Gaussian Distribution on Firefly's behavior)

3.3.1. Social behavior

Random walk is a random process which consists of taking a consecutive random step series of consecutive random steps. Here the step size or length in a random walk can be fixed or varying. If the step length obeys the Gaussian distribution, the random walk becomes the Brownian motion [31]. In standard Firefly algorithm, agents move by just a predefined movement that guides them to better position in their neighborhood. In order to move all fireflies in a same manner, it is used random walk concepts to move all of the agents based on a Gaussian distribution. In proposed algorithm, at the end of each iteration, it is introduced normal Gaussian distribution that is determined by:

$$p = f(x|\mu, \delta) = \frac{1}{\delta\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\delta^2}} \qquad (8)$$

Where x is an error between best solution and fitness value of firefly *i*:

$$x = f(g_{best}) - f(x_i) \qquad (9)$$

μ is mean and δ is standard deviation. Because of using standard normal distribution, it is set to μ=0 and δ=1. Then a random number will be drawn from this Gaussian distribution that is related to each firefly probability (p). Social behavior of fireflies is introduced by:

$$x_i = x_i + \alpha * (1-p) * rand() \qquad (10)$$

Where α is firefly parameter that is adjusted by adaptive parameter approach in [31]. But if the new position of firefly i cause better fitness for that special firefly, it will move to that new position.

3.3.2. Directed movement

In addition in standard Firefly algorithm, firefly movement is based on light intensity and comparing it between each two fireflies. Thus for any two fireflies, the less bright one will move towards the brighter one. If no one is brighter than a particular firefly, it will move randomly. In proposed algorithm this random movement is directed, and that firefly moves towards best solution with better cost in that iteration. The firefly i movement is attracted to best solution that is more attractive (brighter). This causes that if there was no local best in each firefly’s neighborhood; they move towards best solution and make better position for each firefly for next iteration and they get more near to global best. Firefly’s movement in this model is exactly similar to Eq (5) in standard Firefly algorithm.

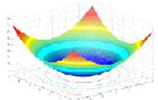
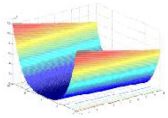
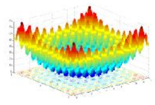
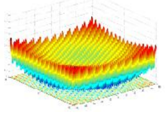
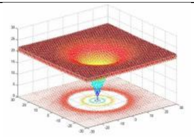
```
1.  Objective function f(x), x = (x1, ..., xd)$^T$
2.  initialize a population of fireflies $x_i (i = 1,2,..,n)$
3.  Define light absorption coefficient $\gamma$
4.  While (t<MaxGeneration)
5.     for $i = 1:n$  (all n fireflies)
6.        for $j = 1:i$
7.           Light intensity $I_i$ at $x_i$ is determined by $f(x_i)$
8.           If ($I_i > I_j$)
9.              Move firefly i towards j in all d dimensions
10.          Else
11.             Move firefly i towards best solution in that iteration
12.          End if
13.          Attractiveness varies with distance r via $\exp[-\gamma r^2]$ ($\beta = \beta_0 e^{-\gamma r_{ij}^2}$)
14.       End for j
15.    End for i
16.    Rank the fireflies and find the current best
17.    Define normal distribution
18.    for $k = 1:n$ all n fireflies
19.       Draw a random number from defined distribution
20.       And apply Eq. 8.
21.       Evaluate new solution(new_cost(k))
22.       If((new_cost(k)<cost(i))&&(new_cost(k)<last_cost_iteration(k)))
23.          Move firefly i towards current best
24.       End if
25.    End for k
26. End while
27. Postprocess results and visualization
```

Pseudo code 5- GD-FF Algorithm

4. Experimental Setting & RESULTS

Performance of the proposed Firefly models is tested on a number of benchmark functions (table 1) which have been extensively used [26]. The benchmark functions include two unimodal functions, Rosenbrock and Sphere, and three multimodal functions, Rastrigin, Griewank and Ackley. The Rstrigin function has many local optima around the global optima and no correlation among its variables. The Ackely function is the only function, which introduces correlation between its variables. Table 1 shows the values that have been used for the dimension of these functions, feasible bounds, the range of the corresponding initial position of the fireflies, and the goal for each function that has to be achieved by the algorithms [27], [28], [29] and [30].

Table 1 standard test functions

	Function	Range	Function figure
F1	**Sphere**= $\sum_{i=1}^{n} x_i^2$	±100	
F2	**Rosenbrock**= $\sum_{i=1}^{n-1}\left(100(\mathrm{x}_{\mathrm{i}+1}-\mathrm{x}_{\mathrm{i}}^{2})^2+(\mathrm{x}_{\mathrm{i}}\text{-}1)^2\right)$	±50	
F3	**Rastrigin**= $\sum_{i=1}^{n}(x_i^2-10\cos(2\pi x_i)+10)$	±5.12	
F4	**Griewank**= $\sum_{i=1}^{\mathrm{n}}\left(\frac{\mathrm{x}_{\mathrm{i}}^2}{4000}\right)-\prod_{\mathrm{i}=1}^{\mathrm{n}}\cos\left(\frac{\mathrm{x}_{\mathrm{i}}}{\sqrt{\mathrm{i}}}\right)+1$	±600	
F5	**Ackley**=$20+e-20*e^{-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}}-e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)}$	±32	

In entire proposed algorithm population size is set to 30, and all of the fireflies are located in search space randomly. All of the results are in 10, 20 and 30 dimensions. The interval of this random search space is limited to the function that is applied for testing the proposed algorithm. Results are the mean fitness of the best values founded in 30 times separated run with independent seeds. In order to make a normal attractiveness, $\boldsymbol{\beta_0}$ is set to 1 and for standard Firefly algorithm to do local search, γ is set to 1, initially and $\mathbf{itr_{max}}$ =3000.

In LA-FF algorithm, its parameters are set as follows: reward and penalty coefficients are 0.1. Three α in LA for gamma can be adjusted by initial number 5 and three α in LA for alpha can be adjusted by initial number 0.5. In Eq(7), value of x is considered 9.0, y=1 and n=5. In any iteration, α is multiplied with 1.003, alpha may not be more than 1 and gamma may not be more than 10 since it was stated about standard mod, gamma varies between 0 to10 and alpha varies 0 to1.

In HGA-FF algorithm, adaptive approach is applied to set parameters by the time. Initial value of α is 0.7 and it changes by Eq (7) and shrinks to a part of search space at the end of each iteration to improve local search. In Eq (7), the value of the n is 2. By tuning these parameters in Eq(7), we define a weight for α coefficient that shrinks step length in each part of search space. In Genetic algorithm step, we use tournament selection technique for selecting each individual and they are shown by value between 0 and 1. The probability of Mutation and Crossover operators are 0.4 and 0.5 in order. Running number of Firefly and Genetic algorithm is 100 simultaneously. By introducing this model, we make a balance between local and global search. Genetic algorithm can find better new solution and by firefly algorithm we can search the found new population locally well. In GD-FF algorithm initial value of α is 0.7 and value of n changes by dimension of each fireflies and its value determines by:

$$n= 10 \text{ ^ } (- \text{dimension}) \qquad (10)$$

Simulation results are shown in table 2 and they are compared by standard Firefly and PSO algorithm. Derivation of PSO that have ever been proposed is added for higher accuracy in comparison.

Table 2 average and standard deviation of best found result in 30 times run

F	Dim			**Average (standard deviation)**						
		Standard Pso	**PSO-TVIW**	**MPSO-TVAC**	**HPSO-VAC**	**Standard firefly**	**HGA-FF**	**LA1-FF**	**LA2-FF**	**GD-FF**
F1	10	0.001	0.001	0.001	0.001	1.45 (0.39)	**0.29 (0.062)**	**1.6940e-004 (0.0014)**	**4.82e-007 (9.7e-008)**	**3.59e-056 (1.96e-055)**
	20	0.001	0.001	0.001	0.001	4.23 (0.52)	**1.69 (0.21)**	**8.8267e-004 (0.0040)**	**2.61e-006 (4.08e-007)**	**3.96 e-035** (2.63 e-035))
	30	0.001	0.001	0.001	0.001	7.08 (062)	**3.67 (0.20)**	**0.0033 (0.0123)**	**4.82e-007 (9.7e-008)**	**7.04 e -033 (7.98 Ee-033)**
F2	10	21.71 (40.16)	16.21 (14.98)	1.23 (4.32)	1.92 (4.33)	24.64 (3.76)	**2.16 (1.01)**	**1.85 (2.65)**	**1.02 (1.28)**	**5.55 (.99)**
	20	52.21 (148.32)	42.73 (72.61)	22.73 (30.63)	20.79 (12.77)	152.7 (22.31)	**14.91 (2.22)**	**8.441 (7.12)**	**13.66 (5.17)**	**18.33 (1.06)**
	30	77.61 (81.17)	61.78 (48.93)	34.22 (36.42)	10.41 (44.84)	335.16 (35.46)	**21.08 (2.18)**	**32.512 (8.12)**	**26.24 (7.14)**	**29.62 (1.76)**

F3	10	2.33 (2.30)	2.1184 (1.56)	1.78 (2.79)	0.039 (0.061)	4.80 (1.67)	**4.97 (2.22)**	**0.0600 (0.0102)**	**4.8 (3.27)**	**3.38 (2.61)**
	20	13.81 (3.50)	16.36 (4.42)	11.13 (0.91)	0.235 (0.126)	11.49 (2.05)	**10.95 (3.98)**	**0.0624 (0.0128)**	**10.5 (2.76)**	**5.87 (4.29)**
	30	6.65 (21.82)	24.346 (6.32)	50.06 (21.14)	1.903 (0.894)	26.13 (2.81)	**17.21 (3.98)**	**0.0635 (0.0119)**	**16.54 (5.54)**	**10.38 (7.13)**
F4	10	0.16 (0.10)	0.092 (0.021)	0.0056 (0.05)	0.057 (0.045)	1.04 (0.001)	**0.03 (0.01)**	**4.64e-006 (3.25e-005)**	**4.26e-005 (8.12e-004)**	**6.227e-008 (9.93e-008)**
	20	0.25 (0.13)	0.1212 (0.52)	0.0348 (0.13)	0.018 (0.005)	1.56 (0.004)	**0.11 (0.15)**	**2.16e-005 (8.33e-005)**	**0.004 (0.01)**	**1.7199e-007 (1.94e-007**
	30	0.0678 (0.24)	0.1486 (0.12)	0.0169 (0.77)	0.023 (0.004)	1.93 (0.003)	**0.16 (0.02)**	**1.93e-004 (9.85e-005)**	**0.01 (0.01)**	**1.5784e-006 (1.4680e-006)**
F5	10	0.41 (1.42)	0.238 (1.812)	0.537 (0.23)	0.092 (0.014)	1.75 (0.01)	**0.67 (0.9359)**	**0.0607 (0.0098)**	**3.94e-004 (1.11e-004)**	**8.9410e-015 (3.08e-015)**
	20	0.57 (3.10)	0.318 (1.118)	0.369 (2.73)	0.117 (0.025)	1.78 (0.01)	**2.20 (0.6641)**	**0.624 (0.0105)**	**0.01 (8.9e-004)**	**3.11e-014 (6.77e-015)**
	30	1.89 (2.21)	0.63 (2.065)	1.32 (2.58)	0.07 (0.01)	1.81 (0.01)	**2.71 (0.38)**	**0.0639 (0.0098)**	**0.15 (9.3e-04)**	**1.3204e-014(5.25e-015)**

As shown in table 2, proposed algorithms have better result than standard Firefly algorithm that expresses they can improve its performance. Each one of the proposed algorithm has its special behavior in solving static problems so they follow optimum value in related function in a different manner.

As you can see in table 2 LA1-FF, LA2-FF and GA-FF have a better performances than standard Firefly algorithm. LA2-FF acts better than LA1-FF on all of the functions, except Griewank, since there are many local optimums in Griewank that they are alike and near together. Therefore if any firefly moves with random steps like standard firefly algorithm will act better than adaptive step.

Also, it should be mentioned that LA-FFs have better results in Sphere, Griewank and Ackley than GA-FF; because LA-FFs function according to a learning behavior, and fireflies move based on their experience in last iterations. Learning Automata always gives desirable information about search space to algorithm, which results in engendering high accuracy. But Genetic algorithm tries to do search over all landscape and achieve new better solution regardless of any experience in landscape. Due to figure of Griewank and Ackley, which are complex with several identical and near local optimum, it can be observed LA-FF algorithms can do search well.

In comparison of GD-FF algorithm with derivatives of PSO algorithm and the other proposed algorithms, it functions well and its performance and accuracy are high. This algorithm omits randomization movement in search space and direct fireflies to best solution; this behavior causes to have better result in all dimensions in each test functions. Movement of fireflies in GD-FF model is based on Gaussian distribution that is composed of fireflies' fitness in population.

In this paper, because of more enhanced performance of GD-FF than others, some plots of its operation are shown on test functions in figures 3 to 5. This figure shows influence of GD-FF behavior on a best firefly to direct it to optimum value.

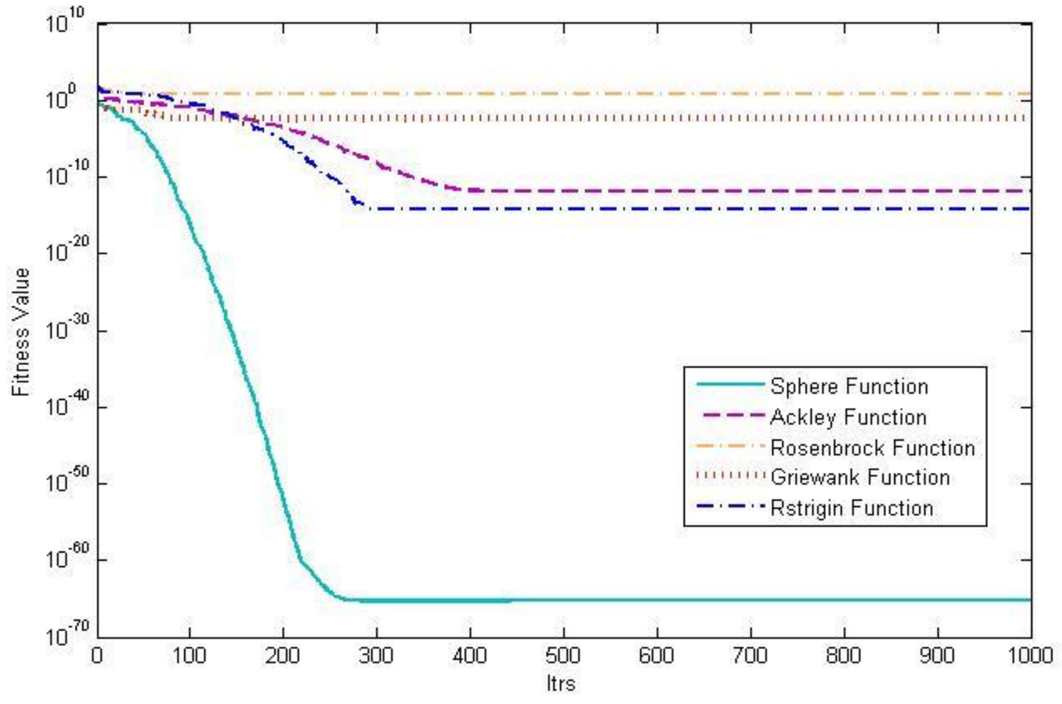


Figure 3 Comparison of simulation results of GD-FF in all functions in 10 dimensions.

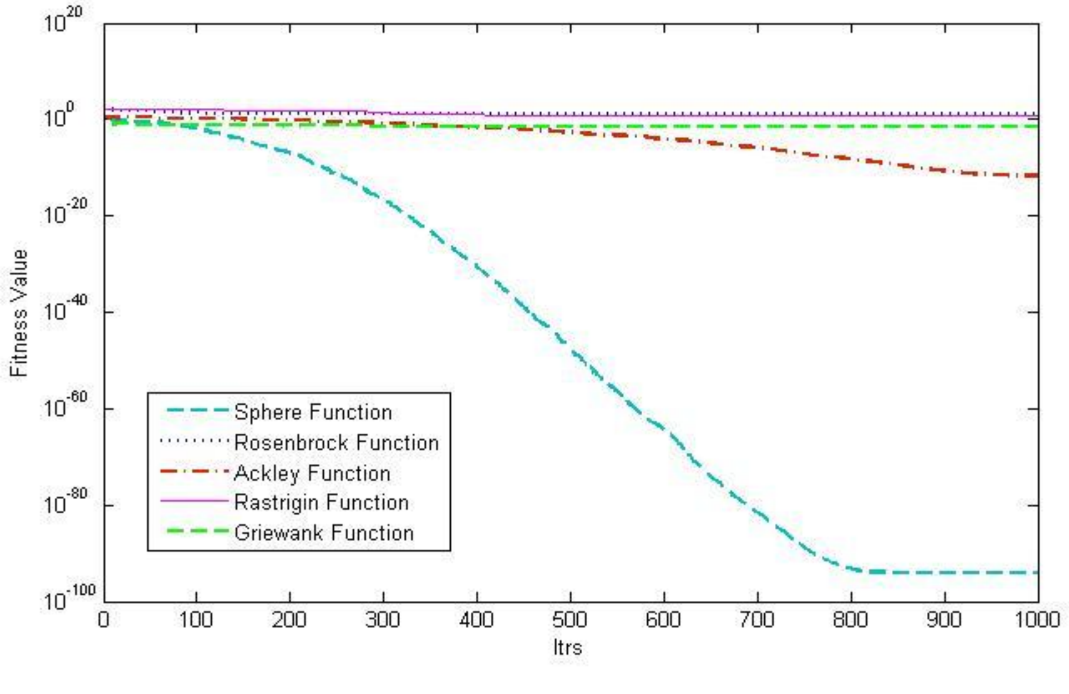


Figure 4 Comparison of simulation results of GD-FF in all functions in 20 dimensions.

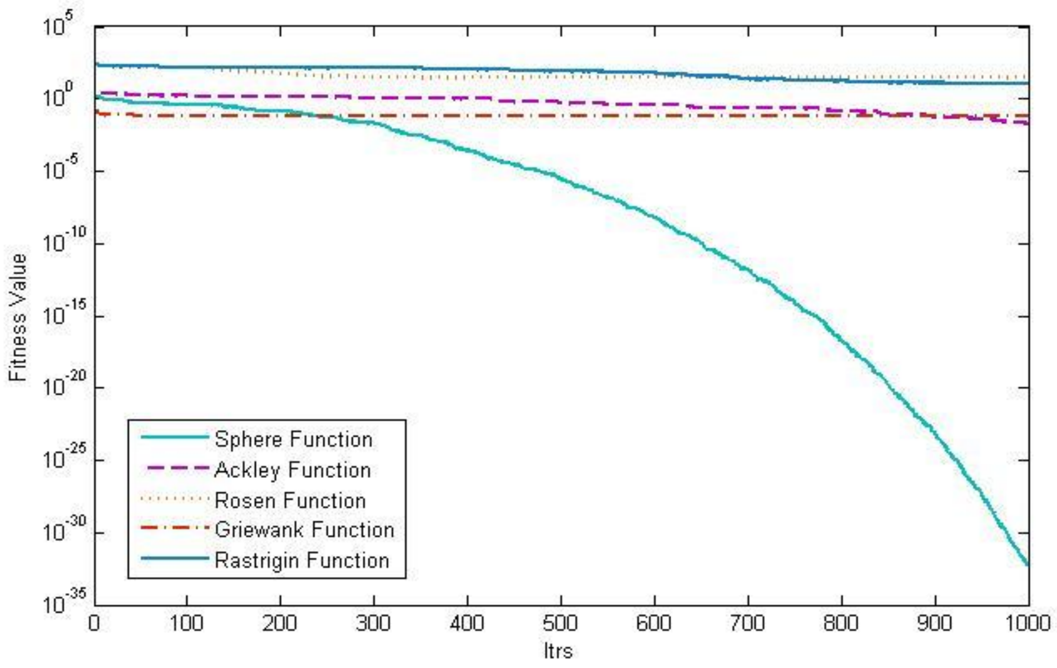


Figure 5 Comparison of simulation results of GD-FF in all functions in 30 dimensions.

As shown in figure 3 to 5, GD-FF functions on each firefly to guide them to optimum value as well. It follows a descending behavior to reach better fitness and approach to optima. This figure shows, GD-FF can get rid of local optima well; further, the problem of trapping into several local optima is solved in this proposed algorithm. Also, we need to note that in Rosenbrock function due to its figure, algorithm loses best fitness in the primary iterations but after some iterations later, it can get rid of this problem and follow best solution to have better results.
Comparison of the standard Firefly algorithm and GD-FF algorithm is shown in figures 6 to 11.

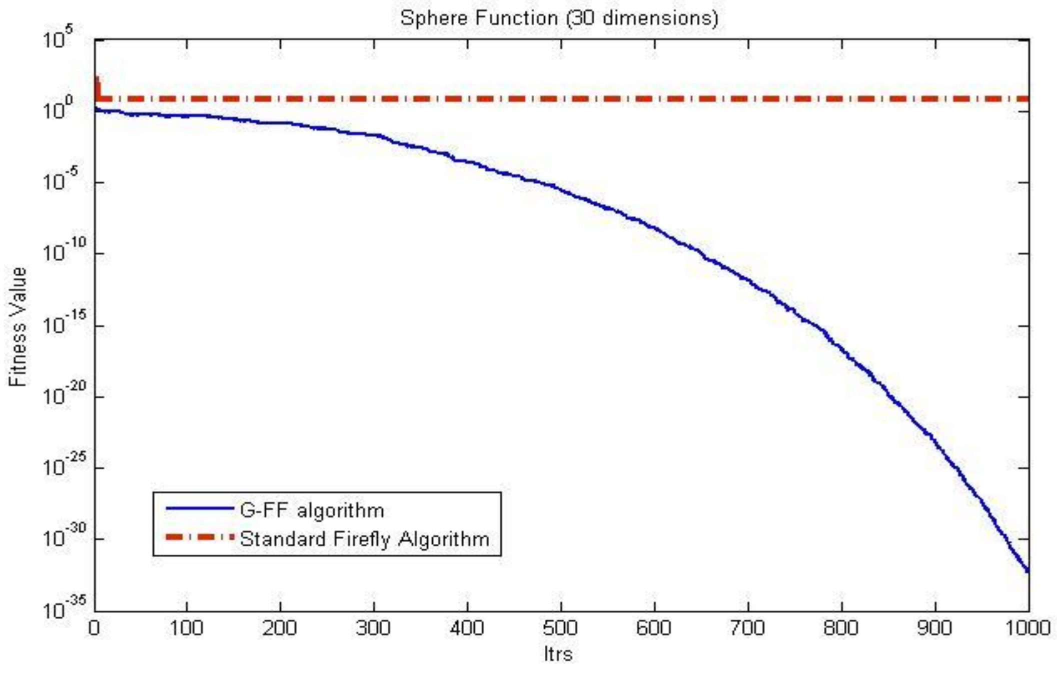


Figure 6 Comparison of GD-FF and standard Firefly algorithm behavior for sphere function

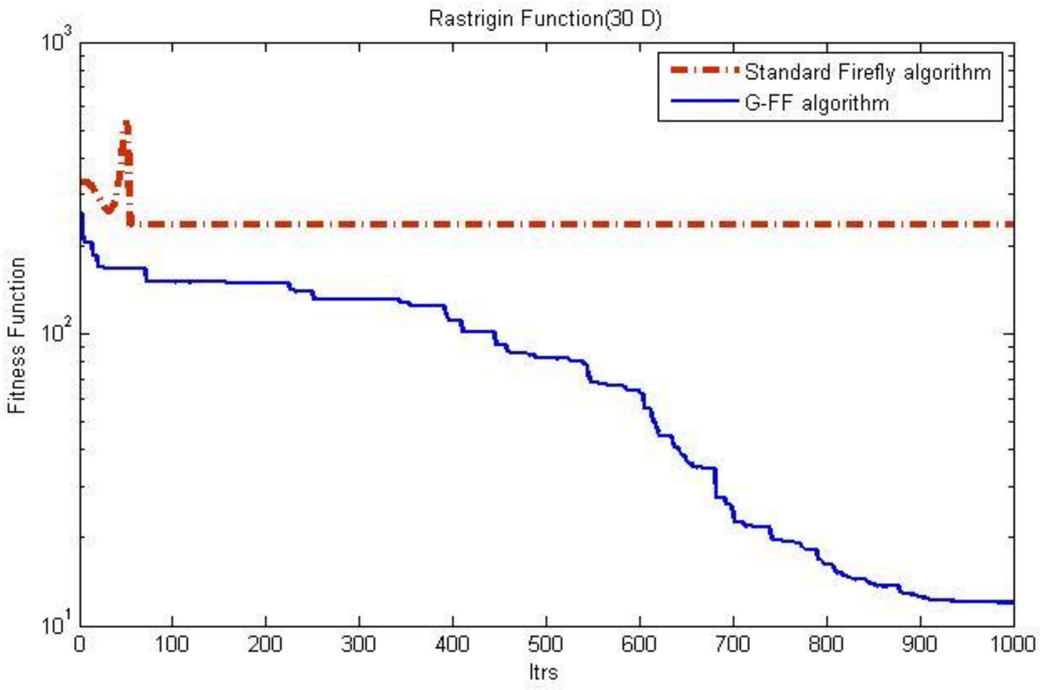


Figure 7 Comparison of GD-FF and standard Firefly algorithm behavior for Rastrigin function

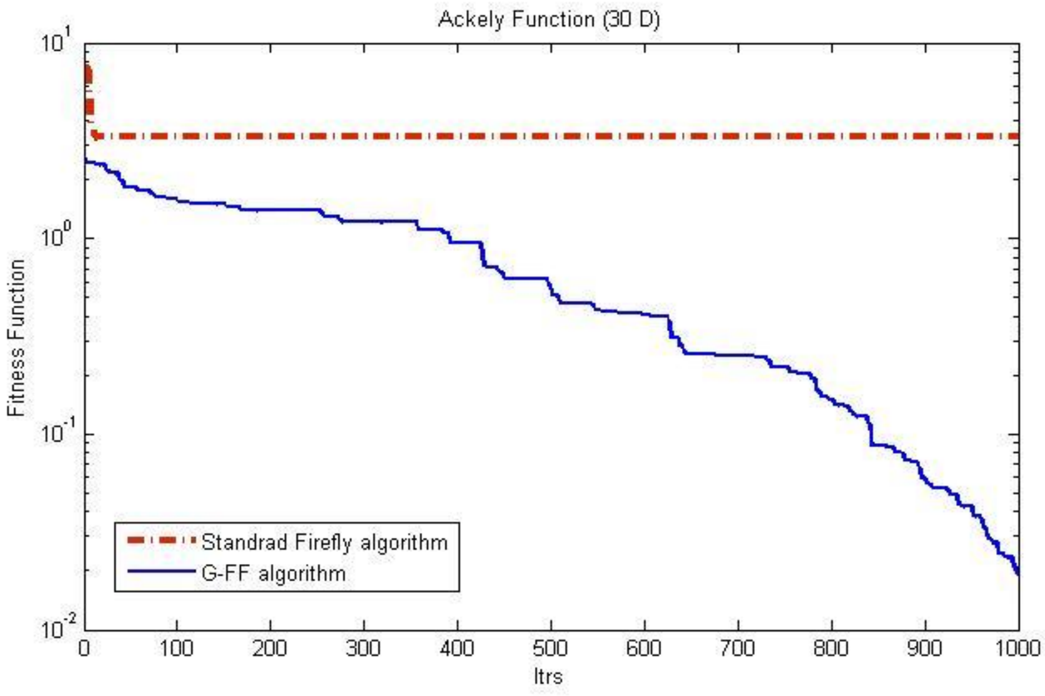


Figure 8 Comparison of GD-FF and standard Firefly algorithm behavior for Ackley function

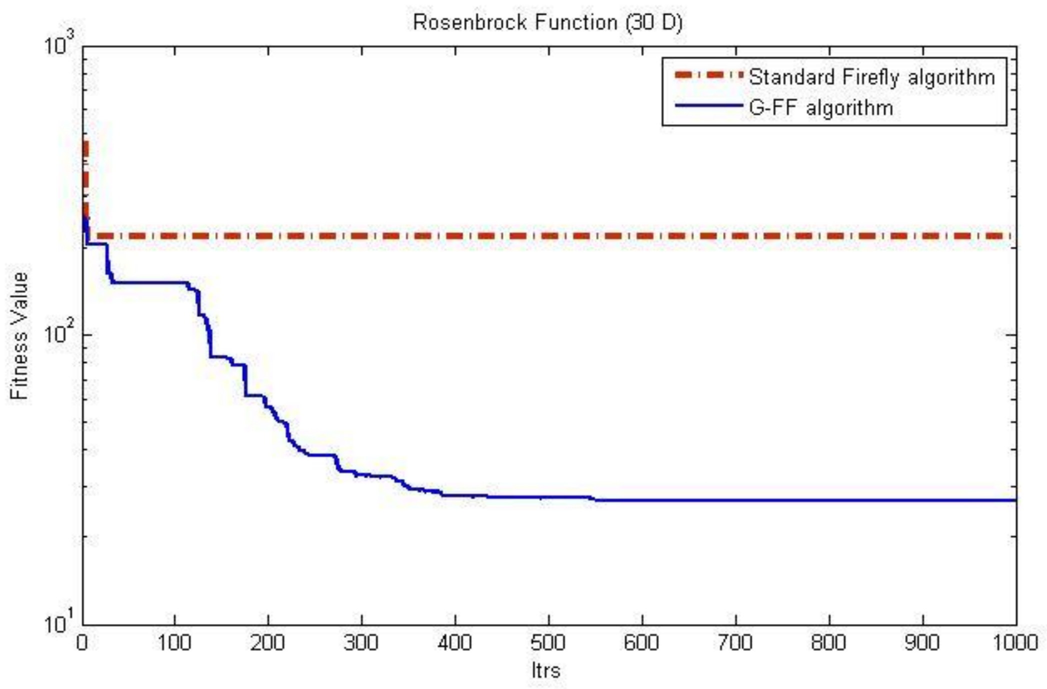


Figure 9 Comparison of GD-FF and standard Firefly algorithm behavior for Rosenbrock function

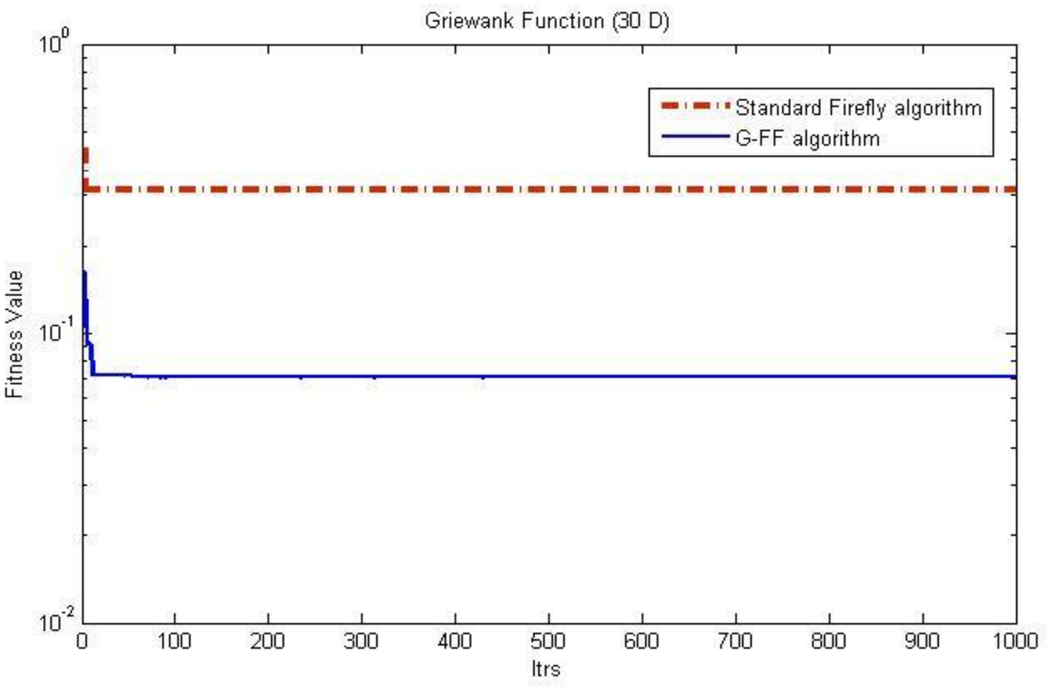


Figure 8 Comparison of GD-FF and standard Firefly algorithm behavior for Griewank function

4. DISCUSSION AND CONCLUSION

In this paper, we studied some disadvantages of Firefly algorithm such as unchanged parameters during iterations and trapping into several local optimums. Three classes of algorithms are proposed to improve performance of Firefly algorithm. In the first class, Learning Automata is used to adapt firefly's parameter by the time. This approach chooses better parameters in different situations and can adapt parameter base on landscape. In addition, to improve global search and get rid of trapping into several local optima, it is introduced Genetic algorithm to explore search space and find new better solutions. By tuning the Genetic algorithm, we made a balance between exploration and exploitation properties. The Last approaches use random walk theory to make a social behavior for all of fireflies and they move based on a Gaussian distribution. At the end, to prevent missing better position, the random movement will be omitted when there isn't any good better neighborhood for a firefly. Experimental results show that proposed algorithms are highly competitive with standard Firefly and PSO algorithm and some other derivatives of PSO.

5. REFERENCES

[1] Y. Liu and K. M. Passino, 2005, Swarm Intelligence: A Survey. In Proc. of _th International Conference of Swarm Intelligence.

[2] Baeck, T., Fogel, D. B., Michalewicz, Z, 1997, Handbook of Evolutionary Computation. Taylor & Francis.

[3] Yang, X. S., 2008, Nature-Inspired Metaheuristic Algorithms. Luniver Press.

[4] Yang, X. S., 2010, Engineering Optimization: An Introduction with Metaheuristic Applications. Wiley & Sons, New Jersey.

[5] Christopher R. Houck, Jeffery A.Jines and Michael G.Kay, 1996, A Genetic Algorithm for Function Optimization: A Matlab mplementation.

[6] Encyclopedia Britannica, 2009, Firefly, In: Encyclopedia Britannica. Ultimate Reference Suite, Chicago: Encyclopedia Britannica.

[7] Babu, B.G., Kanna, M, 2002, Lightning bugs. Resonance 7(9), 49-55.

[8] Fraga, H, 2008, Firefly luminescence, a historical perspective and recent developments, journal of Photochemical & Pgotobiological Science, 146-158.

[9] Lewis, S., Cratsley, C,2008, Flashsigned evolution, matechoice and predation fireflies, Annual Review of Entomology, 293-321.

[10] Leiden frost, R, Elmenriech, W, 2008, Establishing wireless timetriggered communication using a firefly clock synchronization approach, In Proceeding of the 2008 international Workshop on Intelligent Solutions in Embeded Systems, 1- 18.

[11] Krishnand, K, Ghose, and D, 2006, Glowworms swarm based optimization algorithm for multimodal functions with collective robotics applications, Multiagent and Grid Systems, 209-222.

[12] X.-S. Yang, 2010, Firefly Algorithm, Levy Flights and Global Optimization", Research and Development in Intelligent Systems XXVI (Eds M. Bramer, R. Ellis, M. Petridis), Springer London, pp. 209-218.

[13] H. Beigy and M. R. Meybodi, 2005, An Adaptive Call Admission Algorithm for Cellular Networks. Journal of Computer and Electrical Engineering, vol. 31, no. 2, pp. 132-151.

[14] H. Beigy and M. R. Meybodi,2002, Call Admission Control in Cellular Mobile Networks: A Learning Automata Approach. in Eurasia-Ict 2002: Information and Communication Technology, Lecture Notes in Computer Science, vol. 2510, 450-457.

[15] B. J. Oommen and T. D. Roberts, 2000, Continuous Learning Automata Solutions to the Capacity Assignment Problem. IEEE Transactions on Computers, vol. 49, no. 6, 608-620.

[16] M. R. Meybodi and H. Beigy, 2002, A Note on Learning Automata Based Schemes for Adaptation of Bp Parameters. Journal of Neurocomputing, vol. 48, no. 4, 957-974.

[17] H. Beigy and M. R. Meybodi, 2009, A Learning Automata Based Algorithm for Determination of the Number of Hidden Units for Three Layers Neural Networks. International Journal of Systems Science 40,101-108.

[18] Yang, X. S, 2009, Firefly algorithms for multimodal optimization, in:Stochastic Algorithms Foundations and Applications (Eds O. Watanabe and T.Zeugmann), SAGA 2009, Lecture Notes in computer Science, 5792, Springer-Verlag, Berlin, 169-178.

[19] Yang, X. S., 2010, Firefly Algorithm Stochastic Test Functions and Design Optimization. Int. J. Bio-Inspired Computation, vol.2, No. 2, 78-84.

[20] Szymon LÃ ukasik and SÃlawomirZ ak, 2009, Firefly algorithm for Continuous Constrained Optimization Tasks. In proceeding of Springer, 169-178.

[21] K. S. Narendra and M. A. L. Thathachar, 1989, Learning Automata: an Introduction: Prentice-Hall Inc.

[22] B. Masoumi and M. R. Meybodi, 2010, Learning Automata based Multi-agent System Algorithms for Finding Optimal Policies in Markov Games. Asian Journal of Control.

[23] Michalewics, Z, 1994, Genetic algorithms + Data Structures = Evolution Programs. Al Series, Springer-Verleag, New York.

[24] Jean-Yves-Potvin, 1996, Genetic algorithms for travelling salesman problem, Anals of Operations research, 339-370.

[25] Darrwll Whitley, 1993, A genetic algorithm tutorial.

[26] S. Janson, M. Middendorf, 2005, A hierarchical particle swarm optimizer and its adaptive variant, IEEE Transactions on Systems, Man and Cybernetics, Part B 35 1272–1282.

[27] F. van den Bergh, A.P. Engelbrecht, 2001, Effects of swarm size on cooperative particle swarm optimizers. in: Genetic and Evolutionary Computation Conference, San Francisco, CA, USA, 892–899.

[28] R. Mendes, J. Kennedy, J. Neves, 2004, The fully informed particle swarm. simpler, maybe better, IEEE Transactions of Evolutionary Computation 8, 204–210.

[29] D Bratton, J. Kennedy, 2007, Defining a standard for particle swarm optimization. in: IEEE Swarm Intelligence Symposium, Honolulu, Hawaii, USA, 120–127.

[30] R. Mendes, J. Kennedy, J. Neves, 2003, Avoiding the pitfalls of local optima: how topologies can save the day, in: 12th Conference Intelligent Systems Application to Power Systems (ISAP2003), Lemnos, Greece.

[31] Yazdani, M. and Meybodi, M. R., 2010, AFSA-LA:A new Model for Optimization. Proceedings Of the 15th annual CSI Computer Conference(CSICC10), Tehran, Iran, Feb. 20-22.