A MULTISWARM BASED FIREFLY ALGORITHM IN DYNAMIC ENVIRONMENTS

Shadi Mashhadi Farahani
Department of electronic, Computer and IT
Islamic Azad University
Qazvin, Iran
Farahani.ce@gmail.com

Babak Nasiri
Department of electronic, Computer and IT
Islamic Azad University
Qazvin, Iran
Nasiri_babak@yahoo.com

Mohammad Reza Meybodi
Department of Computer engineering and IT
Amirkabir University of Technology
Tehran, Iran
mmeybodi@aut.ac.ir

Abstract— **Many real-world problems are dynamic, requiring an optimization algorithm which is able to continuously track a changing optimum over the time. In this paper, it is explored a new Multiswarm approach based on firefly algorithm. The main idea is to split the population of particles into a set of interacting swarms. Each swarm interacts locally by an exclusion parameter and globally through anti-convergence operator. In order to improve local search, it is used a new model of Firefly algorithm that can track Gbest of related swarm well. In addition when environment change occurs, last information make local search worse, it is proposed an approach to replace population with a new one that it is more probable to be in better position and tracks global best. This paper derives guidelines for setting the involved parameters and evaluates Multiswarm based firefly algorithm on a scenario of the multimodal dynamic, Moving Peaks Benchmark (MPB). Results are compared with other particle swarm optimization (PSO) and evolutionary algorithm approaches from the literature, showing that proposed algorithm significantly outperforms previous approaches. Simulation results show better performance and accuracy than mentioned algorithms.**

Keywords-Firefly Algorithm; Multiswarm; Dynamic Environments; Moving Peak Benchmark.

I. INTRODUCTION

Uncertainty is a clear issue in problems in real-world. Some of the approaches to address these uncertainties are evolutionary algorithms (EAs) and swarm intelligence. Recently, the application of Pso[1] in dynamic problems has also been explored [1], [2], [3] and [4]. Similar to EAs and Pso, Firefly algorithm is one of the swarm intelligence algorithms that need to be adapted for optimal results on dynamic optimization problems. This is due to the following reasons:

1) Outdated mem**ory:** if the environment of problem changes, the information stored in the memory, may no longer be true and may actually misguide the search.

2) Diversity loss: in normal operation, the swarm contracts around the best solution found during the optimization. As has been demonstrated, the time taken for a partially converged swarm to re-diversify, find the shifted peak, and then to re-converge is quite deleterious for performance [5].

The simplest way to address environment change is to assume a change as an entrance to new optimization problem that must be solved. If there is enough time, this approach will be suitable, but in most cases the time is too short to optimize again and quickly. One effort is to use current information to enter into new optimization problem. For example we can use neighborhoods information to address the new problem. So if we don't have enough time to solve new optimization time, the best point is to use existing information to act in new environment. But which information is needed to keep? And how this information will be used for doing search quickly? The other problem that is in most optimization problem, when algorithm is converged to a point, it will lose its diversity and this makes environment inconsistent, so in order to solve the dynamic optimization problem we must pay attention to increase diversity.

Various adaptations to PSO have been suggested to tackle the difficulties mentioned above. Most of them assume that the time of change in the environment is known to the algorithm, or can be detected, e.g., re-evaluation of the objective function [3]. The problem of outdated memory is usually solved by setting each particle's memory position to its current position (i.e., erasing the memory) [3]. Blackwell and Branke [3] introduced a version of a Multiswarm PSO, with the aim of maintaining a multitude of swarms on different peaks. This approach has been inspired by multipopulation EA approaches like the self-organizing scouts developed by Branke[6], which have shown to give excellent results on the tested problems. In the Multiswarms approach, population is split to several swarms that are response to track peaks. A part of population clusters around any local optimum it may discover, and remains close to this optimum for further exploration. The remainder of the population continues to search for new local optima, and the process is repeated if any more local optima are found. This

[1] Particle Swarm Optimization

978-1-4577-0174-0/11/$26.00 ©2011 IEEE

technique seems to work well for dynamic functions consisting several peaks, where the dynamism is expressed by small change in locations of peaks, width and height. To track the optimum in such environment, the algorithm has to be able to follow a moving peak, and to jump to another peak when the peak heights change in a way that makes a previously non-optimal peak the highest peak. In order to allow each sub swarm to track its peak, in [2], the Multiswarm idea has also been combined with the quantum particle idea to sustain diversity within a swarm.

Firefly algorithm is a kind of evolutionary computing algorithms that was introduced by Yang in 2008 and 2009. This algorithm was improved for optimizing some static problems [7] and [8]. In this paper we proposed a new Firefly algorithm based on Multiswarm for dynamic problems. One of the disadvantages of the Firefly algorithm is slow convergence speed and missing the best place for each agent. In addition Proposed algorithm can address outdated memory by replacing some part of population with new population and it is defined a new operator to keep diversity of population and direct agents to global best. Finally it is proposed some self adaptive parameters to get better results. Proposed algorithm is tested on MPB and is compared with SPSO [9], Cellular PSO [10], Mcpso [11], MQSO [11], RSPSO [12], RPSO [13]. They are compared by their offline errors which are the issue to compare for such algorithms in dynamic environments [13]. Simulation results show that proposed algorithm has good performance in dealing with dynamic problems.

The rest of this paper is organized as follows: it outlines Multiswarm technique in section I and then describes Firefly algorithm in section II. Section III expresses proposed algorithm and analyze it. Experimental settings and results are presented in section IV. Section V concludes the paper.

II. FIREFLY ALGORITHM AND MULTISWARM

In this section we will describe Multiswarm technique and provide a brief introduction to Firefly algorithm.

A. *Multiswarm*

The main idea of the Multiswarm technique is to divide the population into a number of sub swarms, with the aim of positing each of those sub swarms on different, promising peaks of the landscape. However, simply breaking up the neighborhoods and dividing the global swarm into a number of independent swarms are unlikely to be effective since the swarms would not interact. [11] proposes some approaches called Exclusion and Anti-convergence to solve this problem. Exclusion is a local interaction between colliding swarms, preventing swarms from setting on the same peak. If a swarm is divided into a number of sub swarms, it may happen that particles from different swarms cluster around a single peak. This is undesirable since the motivation behind a Multiswarm approach is to posit different swarms on different peaks. No single swarm is able to move closer to the peak and optimization would cease. In order to prevent this, we make a simple competition among swarms that are close to each other. The winner is the swarm with the best function value. The loser is expelled and reinitialized in search space; the winner remains. Swarms can be considered to be close to each other when their Gbest of swarms are within an exclusion radius r_{excl}. Exclusion thus constitutes local interaction among colliding swarms.

Anti-convergence is an information sharing interaction among all swarms as a global interaction in the Multiswarm algorithm, with the aim of allowing new peaks to be detected. As each swarm converges to a peak neutral particles orbit chaotically around the peak.[5] The spatial extent of the neutral sub swarm is therefore a suitable criterion for swarm convergence; swarm convergence is thus defined as the case of the neutral swarm size being less than a convergence radius r_{conv} and multi swarm convergence occurs when all swarms have been converged. But if the number of swarms is less than the number of peaks in the fitness landscape, and all swarms have converged, the system will have lost its peak-detection capability; Therefore it was introduced anti-convergence operator in [11]. When all swarms have converged, anti-convergence expels the worst swarm from its peak and reinitializes it in the search space. As a result there is at least one swarm watching out for new peaks.

B. *Firelfy algorithm*

The Firefly algorithm was developed by the author [7] [8] and it is based on idealized behavior of the flashing characteristics of fireflies. For simplicity, we can summarize these flashing characteristics in the following three rules:

- All fireflies are unisex, so that one firefly is attracted to other fireflies regardless of its 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 moves randomly.
- The brightness of a firefly is affected or determined by the landscape of the objective function to be optimized [8] [14].

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 $I(x) \sim f(x)$. 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 [13] [15].

The light intensity $I(r)$ varies with distance r monotonically and exponentially. That is:

$$I = I_0 e^{-\gamma r} \qquad (1)$$

Where 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) [15].

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

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 [16]. In general β_0 ∈[0,1] should be used [7].

The value of γ determines the variation of attractiveness with increasing distance from communicated firefly. In general γ ∈[0,10] could be suggested [3].

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

$$x_i = x_i + \beta_0 e^{-\gamma r_{ij}^2}(x_j - x_i) + \alpha\, \varepsilon_i \qquad (3)$$

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] [16]. In [15] uses an Lévy distribution instead of Gaussian one.

III. PROPOSED ALGORITHM

In proposed algorithm there are some predefined swarms that are responsible to cover peaks and to track them in changing environment. It is possible that local optimum change to global one when a change occurs. This paper uses Multisawrm technique to cover each peak by a swarm. As mentioned above swarms interact with each other by Exclusion and Anti-convergence operator and it is introduced r_{excl} and r_{conv} to implement these two operators. Indeed when all swarms have been converged and the number of peaks is more than the number of swarms, the swarm with the worst fitness will reinitialize in search space to find another local optimum. Multiswarm, Exclusion and Anti-convergence are used exactly like [11]. To keep diversity in each swarm, quantum and charged particles are used. Every swarm has a particle with the best fitness value that called Gbest. Quantum particles are update by Firefly algorithm movement and charged particles exposure in a radius of the best particle in each swarm. In general proposed algorithm can be shown in following steps.

A. *Behavior of Firefly algorithm*

Due to slow convergence speed of Firefly algorithm and trapping into several local optimums, in this paper, a new behavior is introduced that improves performance of Firefly algorithm. In proposed algorithm all of the quantum particles in each swarm will move to Gbest if there was no any better firefly in their neighbors. This behavior improves speed of convergence. Also to cover any deviations in firefly movement, It is used an angle. This angle makes movement of fireflies predictable and gives a direction to each firefly. In order to move each firefly, kind of quantum, this paper uses Eq (5) and Eq (6). Also for adjusting track motions each particle is multiplied in a constant value called *s*. By this technique, diversity in each swarm is kept. When all fireflies are moved, the best firefly will update. In addition to change step length for each firefly, and have better local search in last iterations, we used an approach to adapt step length in each iteration. It is used a weight coefficient ($weight$) that depends on number of iterations. In order to prevent shrinking this value, it is applied a high and low limitation called max_{weight} and min_{wieght}.

$$weight = \frac{max_{iteration} - itr}{max_{iteration}} * (max_{weight} - min_{wieght}) + min_{weight} \quad (4)$$

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

$$x_i = x_i + \theta \qquad (6)$$

B. *Exploiting the seacrh space*

To augment exploitation property in our algorithm, a Try_number parameter is introduced that all Gbest in each swarm, exploits a part of search space Try_number of times and tries new positions. This part of search space is a radius of that Gbest (r_{cloud}), but if their next place is better than before they will move. Gbest motion is determined by Eq. (7).This idea improves convergence speed by number of evaluation and it can decreases reduction of diversity after environment changes. But it should be mentioned that, choosing size of r_{cloud} and Try_number value are considerable. If radius is small, local search will improve but on the other hand decreases step length and if its value increases algorithm can explore the search space.

$$X_{i,j} = Gbest_{i,j} + (R_j \times r_{cloud}) \qquad (7)$$

In proposed algorithm the value of radius is determined by step length of peaks. Actually this parameter solves decreasing of diversity after environmental changing problem. In order to achieve this purpose the size of r_{cloud} is half of the severity and then it will be reduced by a self-adaptive manner. After quantum fireflies has been exploited a part of search space, algorithm evaluates success rate. Success rate is determined by Eq (8).

Success rate = Total number of reads to succeed / Try_number (8)

Then a compaction coefficient is computed by:

$$CF_i = CF_{min} + ((1 - CF_{min}) \times (S_i)) \quad (9)$$

Where S_i is success rate of firefly i in current iteration, CF_i is compaction coefficient swarm i and CF_{min} is low limit of compaction coefficient that is a positive number and less than one. So compaction coefficient is a number between [CF_{min},1] and it can reduce $r_{\_cloud}$ in every iteration by Eq (10) to increase the ability of doing local search to get better result.

$$r_{cloud,i}(t+1) = r_{cloud,i}(t) \times CF_i \qquad (10)$$

In dynamic environment frequency of environment change is defined by evaluation number. Without considering this topic, however if Try_number increases, the ability of doing search improves and algorithm can get better results in less iteration but when number of evaluations increase too , before getting better results environment is changed. To deal with this problem, Try_number for Gbest that has better fitness in swarms, is more than the other Gbest. This approach causes local search get strong around best found place. On the other hand by choosing a good value for Try_number, the number of evaluation will be decreased.

Environment changing is determined by evaluating a firefly and if the fitness of that firefly is changed, adaptive parameter will set to its initial value and it is proposed that the worst swarm replace with a swarm that is placed around the best firefly. Pseudo code of proposed algorithm is shown in figure (1).

```
for EACH PARTICLE
  set initial value of parameters
  Randomly initialize position
  evaluate f(x)
end
repeat
Anti convergence operator
exclusion operator
test for change
   if change occur
      reevaluate each particle
      replace the worst swarm with a swarm that is placed in a cloud of global best
      reinitialize r_cloud
   for each swarm
      for each particle in swarm
   // update particles based on proposed Firefly algorithm
   apply Equationuation (5)
   apply Equationuation (6)
      end
   end
   for each swarm i
      for cnt =1 to Try_number
         update gbest_i  based on Equation 7
         if f(qbest_i) > f(gbest_i) then
            gbest_i  = qbest_i
      Endfor
   Endfor
   update r_cloud based on Equationuation 10
   until stopping criterion is met
```

Figure 1 Pseudo code of proposed algorithm

IV. PARAMETER SETTINGS AND RESULTS

Proposed algorithm is tested on Moving Peaks Benchmark that has ever been used for dynamic problems. Results are collected by parameters of table 1 that shows second scenario of MPB [14] which the most famous scenario in this benchmark. Within MPB problem, the optima can be varied by three features, i.e., the location height and width of peaks.

Table 1 peak parameters

Parameter	value
Peak numbers	Between 1 to 200
Change frequency	5000
Height severity	7.0
Width severity	1.0
Peak shape	cone
Basic function	no
Shift length, s	1.0
Number of dimensions, D	5
Correlation coefficient, λ	0
The range of peaks	[0,100]
Height of peak range	[30,70]
Width of peak range	[1,12]
Initial height value	50

In all simulations the numbers of swarms are equal to number of peaks except when number of peaks is 100 and 200 that this parameter is set to 10, because in these two cases, number of population is too much and speed of convergence gets slower. Try_number is 5 and this parameter for the best swarm is 50. Initial value of the r_{cloud} is half of the peak step length and s parameter is 0.82 and θ=π/4. Parameters of firefly algorithm is set to α = 0.005, γ=1 and β_0=1. r_{conv} and r_{excl} are set exactly like [11] and $CF_{min}$is set to 0.6. Results of proposed algorithm are listed in table 2.

All results are the average of offline error of 30 independent runs with different random seeds. Simulation results (offline error and standard error) on MPB with 100 times environment change on different number of peaks are shown in table 2.

As shown in simulation results, proposed algorithm has better performance than the others. When the number of swarms is more than the number of peaks, swarms compete on peaks and finally they eliminate each other from . Also when the numbers of peaks are more than swarm number, some of the peaks aren't covered by any swarms so results get worse than before.

But anti convergence strategy improves this problem however it can't prevent decrease in performance.

Table 2 offline error and standard error of different peaks number with frequency 5000

Peak number	Cellular pso	FMSO	MQSO	MCQSO	SPSO	RPSO	PROPOSED ALGORITHM
1	2.55(0.12)	3.44(0.11)	3.81(0.34)	4.93(0.17)	2.64(0.10)	0.56(0.04)	**0.72(0.10)**
5	1.68(0.11)	2.94(0.07)	1.90(0.14)	2.07(0.08)	2.15(0.07)	12.22(0.76)	**0.98(0.11)**
10	1.78(0.05)	3.11(0.06)	1.80(0.10)	2.08(0.07)	2.51(0.09)	12.98(0.48)	**1.15(0.61)**
20	2.61(0.07)	3.36(0.06)	2.96(0.12)	2.64(0.07)	3.21(0.07)	12.79(0.54)	**1.89(0.13)**
30	2.93(0.08)	3.28(0.05)	3.36(0.12)	2.63(0.08)	3.64(0.07)	12.35(0.62)	**2.20(0.08)**
40	3.14(0.08)	3.26(0.04)	3.70(0.14)	2.67(0.07)	3.85(0.08)	11.37(0.41)	2.42(0.22)
50	3.26(0.08)	3.22(0.05)	3.76(0.14)	2.65(0.06)	3.86(0.08)	11.34(0.29)	2.67(0.20)
100	3.41(0.07)	3.06(0.04)	4.12(0.14)	2.49(0.04)	4.01(0.07)	9.73(0.28)	2.48(0.44)
200	3.40(0.06)	2.84(0.03)	4.09(0.13)	2.44(0.04)	3.82(0.05)	8.90(0.19)	2.54(0.095)

V. CONCLUSION

In this paper it is proposed a new model for optimizing in dynamic environment which is based on Firefly algorithm, and results are compared with several known method on Moving Peak Benchmark. Proposed model follows Multiswarm approach to cover all search space and interact with each other by exclusion and anti-convergence operator. In each swarm local search happens by firefly algorithm but for moving to global best and increasing speed of convergence, it is defined a directed movement and angle for each particle. This approach improves Firefly algorithm behavior and increases its convergence speed. To exploit search space and adapting swarms when an environment changes, a self-adaptive approach is applied to find better position for each particle in swarm and the worst swarm is replaced by a swarm with better position by maintaining position. Simulation Results show better performance than mentioned algorithm but when number of peaks increases algorithm misses its ability to get better results.

REFERENCES

[1] T. Blackwell, "swarms in dynamic environments", in Proc. Genetic and Evol. Comput. Conf., vol. 2723, E. Cantu-Paz, Ed., 2002, pp. 1-12.

[2] T.Blakwell and J. Branke, "Multi-swarm optimization in dynamic environments," in Applications of Evolutionary Computing. Ser.

[12] S. Bird and X. Li, "Using regression to improve local convergence", in Proc. IEEE Congr. Evol. Comput.,2007, pp. 592-599.

[13] Y. Shi and R. Eberhart, "A Modified Particle Swarm Optimization", In IEEE International Conference on Evolutionary Computation Proceedings, 1998, pp. 69-73, Anchorage.

[14] S. Yang and C. Li, "A clustering Particle Swarm Optimizer for Locating and Tracking Multiple Optima in Dynamic Environments", in IEEE Transaction on Evolutionary Computation, 2010, pp. 1-16.

[15] X.-S. Yang, "Firefly Algorithm, Levy Flights and Global Optimization", Research and Development in Intelligent Systems

Lecture Notes in Computr science, G. R. Raidl et al., Eds. Berlin, Germany: Springer-Verlag, vol. 3005, 2004, pp.489-500.

[3] A. Carlisle and G. Dozier, "Adapting particle swarm optimization to dynamic environments", in Proc. Int Conf. Artif. Intell, 2002, pp 429-434.

[4] J. P. Coelho, P. B. De Moura Oliveira, and J. E. Boaventura Cunha, "Non-linear concentration control system design using a new adaptive particle swarm optimizer", in Proc. Portuguese Conf. Autom, Control, 2002, pp 132-137.

[5] T. M. Blackwell, "particles swarms and population diversity II: Experiments. In: J. Branke , editor, GECCO Worlshop on Evolutionary Algorithms for Dynamic Optimization Problems, 2003, pp. 14-18.

[6] J. Branke, "Evolutionary Optimization in Dynamic Environments", Norwell, MA: Kluwer, 2001.

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

[8] Yang, X. S., "Firefly algorithms for multimodal optimization", in: Stochastic Algorithmlgorithms:Foundations and Appplications (Eds O. Watanabe and T. eugmann), SAGA 2009, LectureNotes in Computer Science, 5792, Springer-Verlag, Berlin, 2009, pp. 169-178.

[9] D. Parrott and X. Li "Locating and Tracking Multiple Dynamic Optima by A particle Swarm Model Using Speciation", in IEEE Transaction on Evolutionary Computation, vol. 10, No. 4, 2006, pp. 440-458.

[10] B. Hashemi and M. R. Meybodi, "Cellular PSO: A PSO for Dynamic Environments", in advances in Computation and Intelligense, Lecture Notes in Computer Science, vol. 5821, 2009, pp. 422-433.

[11] T. M. Blackwell and J. Branke, :Multiswarm, Exclusion and Anti-Convergence in Dynamic Environment", in IEEE Transaction on Evolutionary Computing, vol. 10, No. 4, 2006, pp. 459-472.

XXVI (Eds M. Bramer, R. Ellis, Petridis), Springer London, 2010, pp. 209-218.

[16] Yang, X. S., "Firefly Algorithm, Stochastic Test Functions and Design Optimisation", Int. J. Bio- Inspired Computation, Vol. 2, No. 2, 2010, pp.78--84 .

[17] Yazdani, D., Nasiri, B. and Meybodi, M. R., (2011) "A New Adaptive Optimization Method for Dynamic Environment based on Particle Swarm Optimization", Proceedings of the 16th Annual CSI Computer Conference (CSI 2011), University of Sharif, Tehran, Iran.