Islamic Azad University

Journal of

Optoelectronical Nanostructures

Winter 2021 / Vol. 6, No. 1

Application of Classical Bird Swarm Learning Algorithm as a Method of Optimization in Nanotechnology Systems

Abdorreza Asrar[1], Milad Yasrebi [*,1]

[1] Faculty of Naval Aviation, Malek Ashtar University of Technology, Iran

(Received 26 Dec. 2020; Revised 29 Jan. 2021; Accepted 20 Feb. 2021; Published 15 Mar. 2021)

Abstract: There can be no doubt that nanotechnology will play a major role in our future technology. Computer science offers more opportunities for quantum and nanotechnology systems. Soft Computing techniques such as swarm intelligence, can enable systems with desirable emergent properties. Optimization is an important and decisive activity in structural designing. The inexpensive requirement in memory and computation suits well with nanosized autonomous agents whose capabilities may be limited by their size. To apply in nanorobot control, a modification of PSO algorithm is required. Using birds' classical conditioning learning behavior in this paper, particles will learn to perform a natural conditional behavior towards an unconditioned stimulus. Particles in the problem space are divided into multiple categories and if any particle finds the diversity of its category in a low level, it will try to move towards its best personal experience. We also used the idea of birds' sensitivity to the space in which they fly and tried to move the particles more quickly in improper spaces so that they would depart the spaces. On the contrary, we reduced the particles' speed in valuable spaces in order to do more search. The proposed method was implemented in MATLAB software and compared to similar results. It was shown that the proposed method finds a good solution to the problem regardless of nondeterministic functions or stochastic conditions.

Keywords: Nanotechnology, Quantum, Swarm Algorithm, Optimization, Cost, Speed, Particle, Standard Deviation

1. INTRODUCTION

As the development of nanotechnology progresses in several disciplines including physics, chemistry, biology and material science, computer scientists must be aware of their roles and brace themselves for the greater advancement of nanotechnology in the future. More recently computer science has become involved in nanotechnology. Such research is wide ranging and includes: software engineering, networking, Internet security, image processing, virtual

[*] Corresponding author. Email: miladyasrebi@gmail.com

reality, human–machine interface, artificial intelligence, and intelligent systems. Most work focuses on the development of research tools. For example, computer graphics and image processing have been used in nanomanipulators that provide researchers an interactive system interface to scanning-probe microscopes, which allow us to investigate and manipulate the surface at atomic scales [1, 2]. Also in medical science have been used a new artificial intelligence algorithm for delivering Nano-robots to the cancer area [3]. Apart from genetic algorithms and other evolutionary algorithms that have promising potential for a variety of problems (including automatic system design for molecular nanotechnology [4]), another emerging technique is swarm intelligence, which is inspired by the collective intelligence in social animals such as birds. These systems usually consist of a population of simple agents that interact locally with each other and with their environment. Although usually no centralized control imposes the agents' behaviors on them, their local interactions lead to the emergence of a public behavior. Many researchers have successfully applied swam intelligence techniques to real-world problems including complex control systems in manufacturing plants and air traffic control [5-8]. With some modifications towards nanotechnology characteristics, these techniques can be applied to control a swarm of a trillion Nano assemblers or nanorobots [9]. It is anticipated that soft computing methods such as this algorithm will overcome concerns about implications of nanotechnology, and prevent the notorious scenario of self-replicating nanorobots multiplying uncontrollably [10]. Recent work by the authors has focused on the use of swarm intelligence for physical nanotechnology applications, where these kinds of severe communication restrictions are common [11,12]. Particle Swarm Optimization (PSO) is a computation- oriented random intelligence and a general optimization method based on population proposed in 1995 [13]. The idea of this algorithm was inspired by birds' social, attacking behavior to search for food. Due to its unique search mechanism, simple concept, computational efficiency, and ease of implementation, swarm algorithm is used widely in many areas of engineering optimization. In Particle Swarm Algorithm, the word "particle" refers to the population members which have a lower mass and smaller volume (small or low-volume random mass). It is the concept of going towards a better mode of behavior with speed and acceleration. Each particle in the swarm represents a solution in a multi-dimensional space with four vectors, its current position, the best position found so far, the best position found by its neighbors so far and its speed, and it sets its position in the search space based on the best position reached by itself (pbest) and the best position reached by its neighbors (gbest) during the search process. Some applications of the algorithm can be referred to in [14], [15] and [16] that used the SF method to calculate the

trajectory in the binary search space to solve the knapsack problem (KP). Meanwhile, to solve some problems, a combination of PSO with guided local search, and some concepts derived from genetic algorithm (GA) can be used [17]. Also adopt particle swarm optimization algorithm (PSO) and one state of the art method (invasive weed optimization algorithm from literature) to solve Multiple traveling salesman problem [18].

A CABC was presented by [19] with discrete coding for TSP. This algorithm was proposed by [20]. Harmony Search Algorithm is a meta-heuristic algorithm in the natural process of musical performance that searches a good condition during jazz improvisation. To find harmony in a pleasant piece of jazz music (a proper condition), improvisation searches as a prescribed mood by the standard of aesthetics, and in fact as the optimization process looks for a general solution (a proper condition) found by the determined objective function. Bees algorithm was first introduced by [21] on a mathematical function. In this algorithm, a D-dimensional bee vector includes the problem variables (solution). Bee Algorithm (BA) was first used for adjoined optimization functions and for scheduling tasks [22] and then for binary data clustering [23]. Imperialist Competitive Algorithm is an evolutionary optimization method inspired by imperialist competition imperial and is derived from imperialist behavior in an efforts to overcome the colonies. Like other evolutionary algorithms, ICA starts with initial population and is divided into two separate types: Colonists and imperialists (those with the best objective function values). All these colonies, according to his power, are shared among the imperialists. There is a reverse relationship between the power and the costs of any state; i.e. more powerful imperialists will be more dominant [24]. Other algorithms have been presented in recent years, most of which were in line with improving famous optimization algorithms, such as GOA algorithm[25], DSA algorithm [26] and BMO algorithm [27]. sinDE algorithm [28], JOA algorithm [29], CPSO [30], SAPSO [31] SFPSO [32] and D-PSO-C algorithm were presented using a dynamic multi-population method to improve the particle swarm algorithm [33].

In order to imitate the physical collective intelligence in social insects, we have proposed this algorithm, which adding these features:

1- Use the idea of flying several groups of birds while flying.

2- Using a population-based mechanism of elitism using an instinctive behavior of birds.

3- Using the new velocity equation for particles.

4- Using the equation of new motion for particles.

5- Using the idea of competence for different parts of the problem space based on the instinctive behavior of the bird

6- Using the learning behavior of the classic conditioning of birds in creating a

suitable movement mechanism in the problem area.
The bird swarm algorithm is widely used in many areas of engineering optimization due to its unique search mechanism, simple concept, and computational efficiency. It is also a good algorithm for static problems and requires low memory consumption.
This paper is organized as follows: in Section 1 an introduction and previous works will be provided; the proposed method will be presented in Section 2; results of the simulation will be discussed in Section 3; and Section 4 will deal with conclusion.

2. THE PROPOSED ALGORITHM

Due to using a number of bird swarms in our proposed method, the determined population was divided into the intended categories. In general we divided the birds into categories, and implemented the proposed method's mechanism in each category and in the entire categories. We finally determined the solutions produced in the entire swarms as the optimal solution.

Fig 1. Bird Swarm Movement

In the first phase, we will define the range of variables in each population. The reason is to divide the variables of each population in certain ranges and allow them to move in that range.

$$n\,Var\ section = x$$

(1)

In this equation, we will define the determined number in the variable x. In this simulation, the number of x equals 10; that is to say we will assume the range of variables to be 10. Now the length of each range is calculated by the equation 3-4.

$$L = \frac{(Var\ Max - Var\ Min)}{n\ Var\ section} \tag{2}$$

In this equation, *Var Max* and *Var Min* are the upper and lower limits, and *n Var section* is the number of variable ranges. In the next step, the population is divided into several parts or categories.

$$n\ Population\ section = x \tag{3}$$

In this simulation we divided the population into 4 categories. Therefore, equation 3-5 can be written as:

$$n\ Population\ section = 4 \tag{4}$$

To define the parameters, a threshold parameter is defined to determine the standard deviation of bird swarms. At the stage of generating the population with chaos and as large as the population determined based on the upper limit, we create the lower limit and the number of variables to generate the initial population.

$$Particle(i).Position = GenerateDataWithChaos(Var\ Max, Var\ Min, Var\ Size) \tag{5}$$

In this equation, *Var Size* is the number of the population, and *GenerateDateWithChaos* is also used because we want to create unique random numbers and the particles' locations in the search space should not be duplicated and regulate the primary population and improve the results. Chaos theory is an interdisciplinary theory stating that, within the apparent randomness of chaotic complex systems, there are underlying patterns, interconnectedness, constant feedback loops, repetition, self-similarity, fractals, and self-

organization. [34] The produced population is inserted in the cost function and the cost of particles will be obtained.

$$Particle(i).Cost = Fitness(Particle(i).Position) \tag{6}$$

We also determine where in the space the variables of each population are located. We divided the dimensions of the problem space into sections so that the space could be classified based on the value of the area. Thus, it was necessary to determine in which areas were any variables of the population.

$$\begin{aligned} &for\ i = 1{:}nVar \\ &\ for\ j = 1{:}nVar\ Section \\ &\quad if\ VarMin + (j-1) * L \leq \\ &\quad Particle(i).Position(j)\ \ and\ \ Particle(i).Position(j) < \\ &\quad (VarMin + j * L) \rightarrow Space(i,j) = j \end{aligned} \tag{7}$$

Birds that have good energy and good nutrition, will have a better performance in flight. We modeled this behavior of birds and removed the particles with inappropriate conditions from the problem space when deciding on the initial population and replaced them with the same number of random population. To do so, we first arranged the initially generated population in an array according to their costs.

$$Cost\ Array = Sort(Particle(i).Cost) \tag{8}$$

And from the end of the array, the specified number of replacements were done. In this study, we replaced one third of the arranged costs of the particles from the end of the array.

$$\begin{aligned} &for\ i = 1{:}index\left(\frac{2}{3} \times nPopulation\right){:}end \\ &\quad Particle(i).Position = \\ &\quad unifrnd(Var\ Max, Var\ Min, Var\ Size) \end{aligned} \tag{9}$$

The cost of new population is estimated, and the particle location is the local optimum for each particle in the first evaluation.

$$PBest = (Particle(i).Position) \tag{10}$$

The general optimum is the best value in the entire particles. For the static functions in this study, the minimum value was the best cost.

$$GBest = min(Particle(i).Cost) \tag{11}$$

After determining the general optimum of the entire particles, the general optimum of each category will be determined separately. To do this, for each category we will initially put the costs of the particles in an array.

$$for\ i = 1{:}nPopulation\ Section$$
$$C = (Particle.Cost) \tag{12}$$

$$T = \frac{(i-1)\times Population\ Number}{nPopulation\ Section} + 1 \tag{13}$$

We specify the population of each category in parameter *T* and put them in parameter *C* in order to separately compare the population in each category.

$$C = C(T) \tag{14}$$

Now the minimum cost per category will be determined as the general optimum of that category.

$$GBest(i) = min(C) \tag{15}$$

We used the idea of birds; sensitivity to the environment, and after dividing the land dimensions into several sections for variables, rated each space on the grounds that every time a particle was placed in a space and it was also the optimum. We added the velocity of particles in the space by 1 unit.

$$\begin{aligned} &for\ i = 1{:}nVar\ Section \\ &\ if\ GBest\ in\ \ i\ \rightarrow Space\ Rate(i) = Space\ Rate(i) + 1 \end{aligned} \tag{16}$$

In the next stage, the standard deviation of each category was measured in order to be able to implement our idea of classical conditioning learning behavior.

$$\begin{aligned} &for\ i = 1{:}nPopulation\ Section \\ &\ if\ STD(i) \geq Threshold \\ &\quad C1 = 0.9 \times C1 \quad C2 = 1.1 \times C2 \end{aligned} \tag{17}$$

$$\begin{aligned} &Else \\ &\quad C1 = 1.1 \times C1 \quad C2 = 0.9 \times C2 \end{aligned} \tag{18}$$

In this equation, the coefficients before and after the local optimum are used in the velocity equation. According to the above equations if the standard deviation of the category is high, we can lead the particle towards the general optimum and make it far from the local optimum by placing a number higher than 1 for the coefficient before the general optimum and a number lower than 1. Conversely, if the diversity or standard deviation of the category is low, we can put a less-than-1 coefficient before the local optimum and a more-than-1 coefficient before the general optimum to send the particle away from the general optimum and lead it towards the local optimum. This is done because if the diversity of the bird swarm is greater than the defined threshold, as the particles have searched in a larger space, going towards its general optimum might be more beneficial.

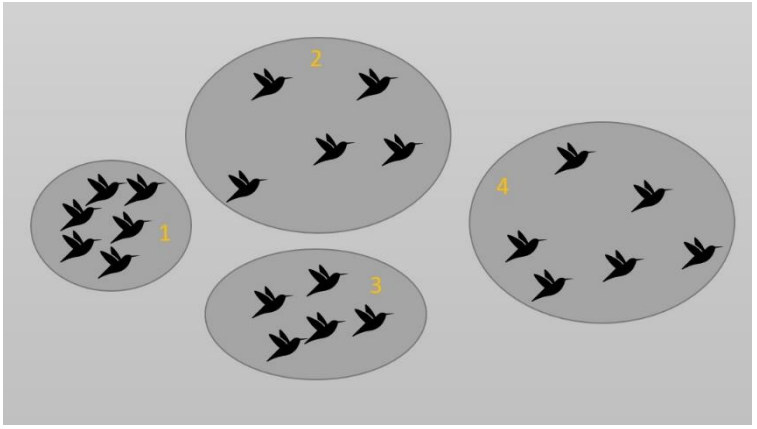


Fig 2. Variation of the categories

In Figure 2 it is shown that categories 1 and 4 have the least and the most diversity, respectively. In categories 1 and 2 the particles mainly move towards a local optimum while in Figure 2 the particles in categories 3 and 4, according to the defined mechanism in this study, will mainly move towards the general optimum. We will define this idea in the form of classical conditioning learning behavior of birds for particles. According to the learned behavior, birds learn to move more eagerly towards the general optimum whenever they feel that the diversity of their categories is high.

The particle velocity equation will be set as follows:

$$
\begin{aligned}
&for\ i = 1{:}nPopulation \\
&\ for\ j = 1{:}nVar \\
&\quad Velocity(i) = \left(1 - \frac{Space\ Rate\ (1,j,Space\ Rate(i,j))}{sum(Space\ Rate(1,j,x))}\right) \times w \times \\
&\quad Particle(i).Velocity + c1 \times rand(Var\ Size) \times \\
&\quad (Particle(i).PBest - Particle(i).Position) + c2 \times \\
&\quad rand(Var\ Size) \times (Particle(i).GBest - \\
&\quad Particle(i).Position)
\end{aligned}
\tag{19}
$$

In this equation, $\left(1 - \frac{Space\ Rate\ (1,j,Space\ Rate(i,j))}{sum(Space\ Rate(1,j,x))}\right)$ is used to control the speed of the particles in high-value and low-value spaces. This part of the velocity equation produces lower numbers for the particles in the spaces with

higher values. Multiplying this lower number by inertia weight will reduce the velocity. Through this mechanism, the particles will move at a slower rate and do more search in the more valuable space. *w* is the inertia weight used to determine the impact of the previous speed on the current one, $Particle(i).Velocity$ is the current speed of the particle, *c1* is the coefficient that specifies the ratio of particles' tendency towards the local optimum, $rand(Var\ Size)$ is a random number for non-linear motion of the particle, $Particle(i).PBest$ is the local optimum or the best personal experience and the best place a particle has had so far. $Particle(i).Position$ is the current location of the particle, *c2* is the coefficient that specifies the ratio of particles' tendency towards the general optimum, and $Particle(i).GBest$ is the general optimum or the best particle in the entire population of the birds. With this equation, if the particles that are going to move in the next step find their standard deviation and category diversity to be high, they will move towards the general optimum, and if the category diversity is low and the particles involved in a category are searching in a smaller space, they will try to incline towards the local optimum or their best personal experience. Determining the particle velocity equation, we can specify the motion equation in accordance with equation 20.

$$Particle(i).NewPosition = Particle(i).Position + Particle(i).Velocity \tag{20}$$

In this equation, $Particle(i).NewPosition$ is the next location of the particle while $Particle(i).Position$ and $Particle(i).Velocity$ are the particle's current location and velocity, respectively.

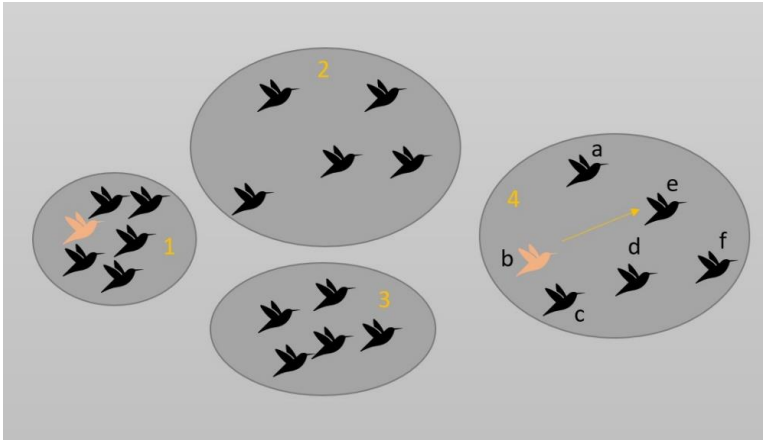


Fig 3. Particle motion mechanism within each category

According to Figure 3, assuming that particle *e* is the general optimum of category 4, particle *b* in category 4 wants to move in the next step and due to the high diversity of the categories tries to set its velocity equation so that it will move toward the general optimum of the category. This is done by setting the number of coefficients *c1* and *c2*. Figure 5-3 shows how to value the problem space. The parts marked in bold have greater value and probably more general optimums have been found in the evaluations in these areas. In the proposed method we try to move the particles in these spaces at a slower speed so that the particles could do more search in these areas. The first part of the equation will apply the speed of the proposed method of this mechanism.

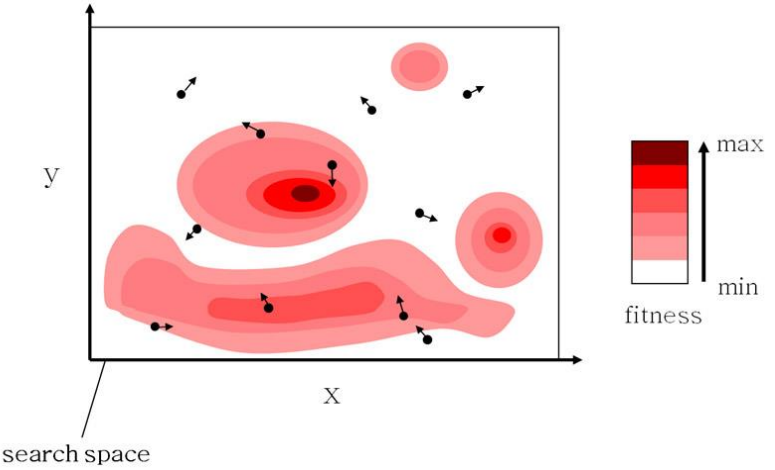


Fig 4- how to value the problem space

After the particle motion according to equation 3-23 and getting located in a new place, they will be put in the cost function again and the cost of the particles will be obtained. The local and general optimums of each category will be determined again. The general optimum will be in accordance to the previous equations and finding the local optimum or the best personal experience of each particle can be achieved through equation 21.

$$if\ Cost(Particle(i).NewPosition) < Pbest(i) \rightarrow Pbest(i) = Particle(i).NewPosition \tag{21}$$

In this equation $Cost(Particle(i).NewPosition)$ is the cost of the bird's new place and *Pbest (i)* is the local optimum of the bird. Earlier, it was said that in the first stage, the local optimum of any bird is its initial place, and from the second iteration, the local optimum of each particle is updated according to equation 22. In the end, the number of evaluations determined by comparing the general optimums of the total categories will obtain the best cost.

$$for\ i = 1{:}nPopulation\ Section \\ Best\ Cost = min\ (Gbest(Population\ Section(i))) \tag{22}$$

Regarding the condition of stopping, it should be said that the following ways are available:
1- A certain number of iterations
2- Achieve a decent threshold
3- A number of iterations that do not change the competence (for example, if after 10 iterations the competency was constant and did not improve).
4- The last way is based on the density of aggregation around the optimal point. Thus, we say that if 80% of the particles were at a distance of less than 20% from the best answer, the algorithm must be stopped.
We use a certain number of iterations in this algorithm. Using any of the above methods will not have much effect on the results, because the proposed algorithm on static problems will be very accurate.

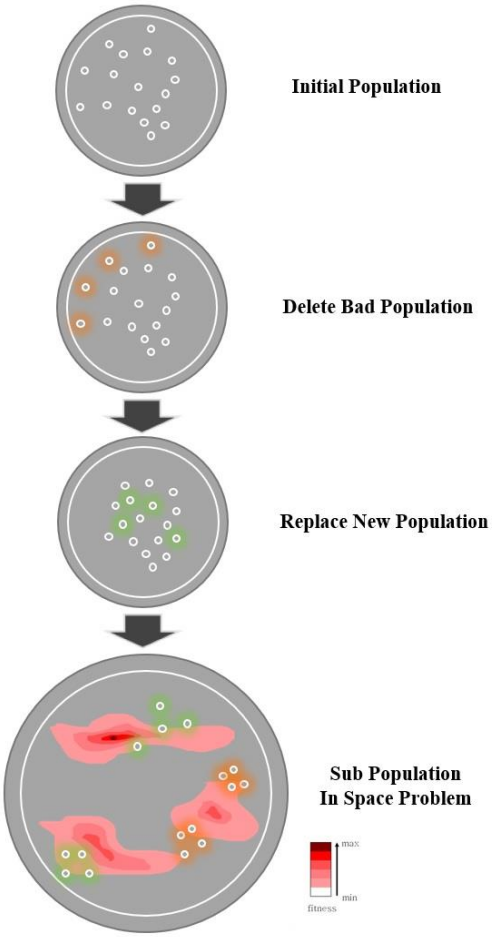


Fig 5 – General Diagram of the Proposed Method

Figure 5 shows the general diagram of the proposed method. In this figure, first the initial random population is generated and then a third of the bad population will be removed. In the next step, a random population as large as the removed one will replace it. The population is then divided into a number of particle categories. In this figure, the particles in the red category will mainly move towards their personal experience due to the low diversity. This motion that will be combined with a random and free motion may change the same category into a category with higher diversity in the next assessments. Due to the higher diversity, the particles in the green category will converge to the general optimum of their category. In this figure, the particles in high-value spaces identified in bold will move slower in order to do more search in those points.

Here, the conditioned stimulus of bird swarm diversity and the natural work that the birds do as they see the condition is to go to the local or general optimums of the categories.

1. Generate the initial population randomly.
2. Insert the initial population in the cost function and calculate the cost of the particles.
3. Arrange the particles in an array based on their costs.
4. Remove stage 3 and 1/3 of the particles respectively from the arranged array and the end of it and replace them with random particles.
5. Put the new population in the cost function and calculate the cost of the particles.
6. Divide the particles into several categories.
7. Specify the variable areas of each population.
8. For each category, put the initial place of the particles as their local optimum.
9. Specify the general optimum of each category as its best local optimum and add one unit to the value of the particle's space.
10. Set the values of parameters *C1* and *C2* based on standard deviation and diversity of the categories.
11. For each particle, update the velocity equation are based on the speed of the space at which they are.
12. Calculate the equation of particles' motion.
13. Move the population to a new location.
14. Calculate the cost of the particles for new locations of the particles.
15. If the new location of the particles is better than their local optimum, put the new location as a local optimum.
16. Find the general optimum for each category.
17. Select the best general optimum of the categories as the best cost.
18. If the current iteration is less than the determined iteration, go to Step 5; otherwise, go to Step 19.
19. The end

Fig 6- pseudo-code of the proposed method

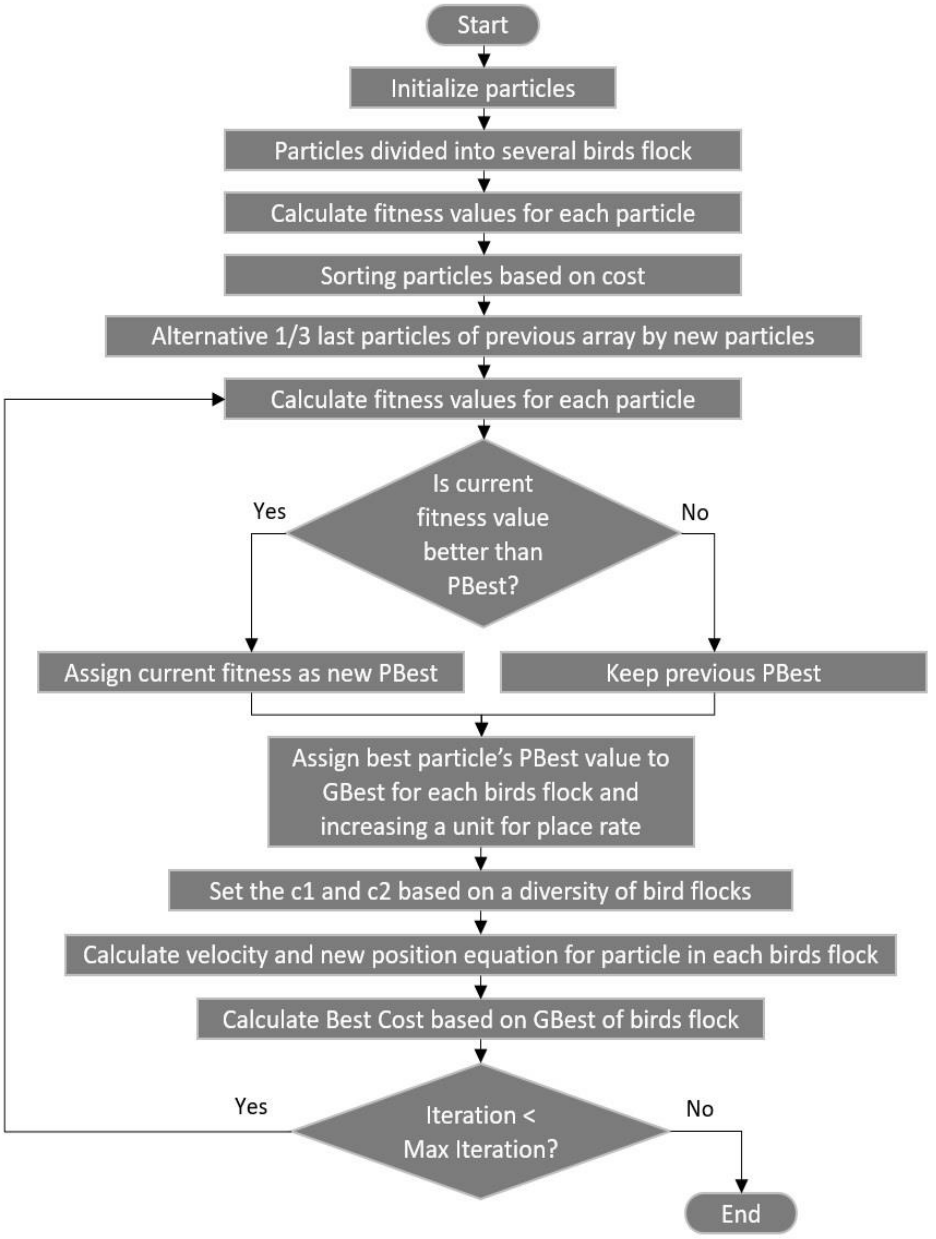


Fig 7. Flowchart of the proposed method

3. SIMULATION AND RESULTS

We implemented the proposed method in MATLAB software and assessed the results in the series of CEC 2005 tests were used.

The proposed method was compared to the following methods: Drop Rain [35], CS [36], TLBO [37], DSA algorithm [26] and BMO [27]. We tested these methods on the test series CEC 2005 [38]. This comparison was done based on the average best cost (Mean) and the standard deviation (StdDev) of averagely 25 times and on the basis of 300000 evaluations with 30 variables and a population of 40.

Table 1. Results of comparison with newer algorithms on 26 functions based on the mean best cost and standard deviation

	Criteria	Rain Drop	CS	TLBO	DSA	BMO	Proposed
F1	**Mean**	2.63E-05 ‡	**0§**	3.64E-27 ‡	1.23E-28 ‡	1.23E-28 ‡	**0**
	StdDev	5.15E-05	**0**	5.82E-27	**0**	0	**0**
F2	**Mean**	8.46E+02‡	1.40E-03‡	1.79E-09‡	1.56E+03‡	1.36E+00‡	1.88E-09
	StdDev	1.83E+02	2.49E-03	4.50E-09	5.64E+02	1.45E+00	4.69E-09
F3	**Mean**	1.90E+06‡	2.06E+06‡	1.05E+06‡	1.19E+07‡	5.48E+06‡	1.01E+06
	StdDev	6.20E+05	6.83E+05	6.64E+05	5.55E+06	2.42E+06	6.72E+05
F4	**Mean**	1.39E+04‡	1.51E+03‡	2.73E+02‡	8.46E+03‡	3.87E+03‡	1.67E+02
	StdDev	2.93E+03	1.19E+03	8.31E+02	2.01E+03	4.32E+03	7.78E+03
F5	**Mean**	9.51E+03‡	2.96E+03‡	3.24E+03‡	3.24E+03‡	4.22E+03‡	1.49E+03
	StdDev	1.89E+03	7.17E+02	7.64E+02	4.99E+02	1.30E+03	6.87E+02
F6	**Mean**	1.16E+02‡	1.52E+01‡	3.13E+01‡	5.18E+01‡	4.42E+01‡	1.33E+01
	StdDev	3.10E+01	2.10E+01	4.51E+01	3.37E+01	4.91E+01	2.56E+01
F7	**Mean**	6.70E-01‡	3.24E-03‡	2.77E-02‡	2.28E-02‡	2.15E-02‡	3.17E-03
	StdDev	1.76E-01	5.52E-03	4.12E-02	8.43E-03	1.66E-02	5.22E-03
F8	**Mean**	2.03E+01‡	2.09E+01‡	2.09E+01‡	2.09E+01‡	2.05E+01‡	2.01E+01
	StdDev	9.33E-02	5.83E-02	5.01E-02	5.15E-02	5.58E-02	8.44E-02
F9	**Mean**	8.03E+00‡	2.21E+01‡	9.78E+01‡	0§	3.27E+00‡	**0**
	StdDev	2.28E+00	4.72E+00	2.38E+01	0	2.57E+00	**0**
F10	**Mean**	3.37E+02‡	1.64E+02‡	1.18E+02‡	8.93E+01‡	6.79E+01‡	4.38E+01
	StdDev	6.85E+01	3.85E+01	3.33E+01	2.03E+01	1.88E+01	2.23E+01

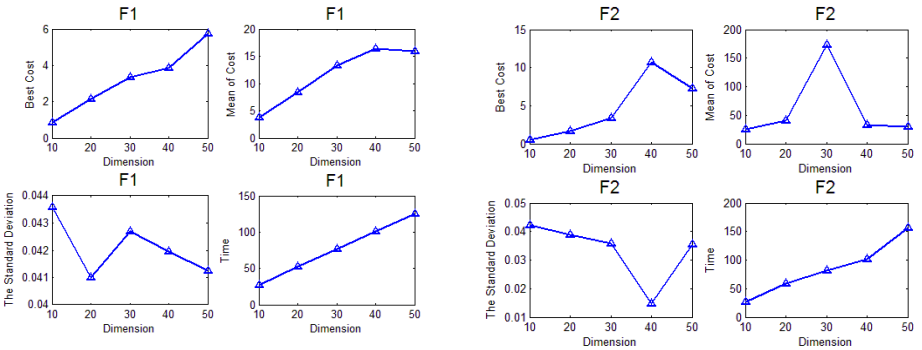
F11	**Mean**	3.32E+01‡	2.94E+01‡	3.02E+01‡	2.71E+01‡	2.52E+01‡	2.33E+01
	StdDev	3.02E+00	1.32E+00	5.30E+00	2.10E+00	3.76E+00	4.39E+00
F12	**Mean**	1.06E+04‡	2.59E+04‡	9.49E+03‡	2.09E+04‡	1.13E+04‡	9.82E+03
	StdDev	9.22E+03	7.13E+03	1.13E+04	5.03E+03	5.85E+03	3.53E+04
F13	**Mean**	1.96E+00‡	5.96E+00‡	4.62E+00‡	1.83E+00‡	2.14E+00‡	1.03E+00
	StdDev	4.26E-01	1.32E+00	1.42E+00	1.25E-01	5.29E-01	5.45E-01
F14	**Mean**	1.29E+01‡	1.30E+01‡	1.28E+01‡	1.29E+01‡	1.22E+01‡	1.19E+01
	StdDev	3.92E-01	2.10E-01	3.99E-01	2.43E-01	6.65E-01	7.43E-01
F15	**Mean**	4.12E+02‡	2.85E+02‡	4.62E+02‡	3.96E+01‡	3.04E+02‡	2.55E+01
	StdDev	1.98E+02	7.52E+01	5.62E+01	2.38E+01	1.14E+02	2.38E+01
F16	**Mean**	4.15E+02‡	2.05E+02‡	2.55E+02‡	1.63E+02‡	1.19E+02‡	1.07E+02
	StdDev	1.59E+02	5.72E+01	1.35E+02	3.77E+01	3.98E+01	6.98E+01
F17	**Mean**	4.76E+02‡	2.36E+02‡	2.61E+02‡	2.26E+02‡	1.45E+02‡	1.22E+02
	StdDev	8.39E+01	5.84E+01	1.59E+02	3.48E+01	7.51E+01	6.53E+01
F18	**Mean**	1.04E+03‡	9.09E+02‡	9.34E+02‡	9.09E+02‡	9.06E+02‡	8.79E+02
	StdDev	8.90E+01	1.96E+00	3.78E+01	1.48E+00	1.23E+00	1.73E+00
F19	**Mean**	1.05E+03‡	9.09E+02‡	9.51E+02‡	9.10E+02‡	9.06E+02‡	7.29E+02
	StdDev	5.16E+01	1.63E+00	2.98E+01	1.28E+00	1.75E+00	1.88E+00
F20	**Mean**	1.06E+03‡	9.09E+02‡	9.47E+02‡	9.09E+02‡	9.06E+02‡	9.04E+02
	StdDev	8.00E+01	1.96E+00	2.30E+01	1.58E+00	1.79E+00	1.11E+00
F21	**Mean**	1.13E+03‡	5.12E+02‡	1.01E+03‡	5.00E+02‡	1.09E+03‡	4.79E+02
	StdDev	2.80E+02	6.00E+01	3.14E+02	9.21E-14	4.49E+00	5.33E-14
F22	**Mean**	1.22E+03‡	9.20E+02‡	9.38E+02‡	9.38E+02‡	8.64E+02‡	8.51E+02
	StdDev	6.95E+01	2.80E+01	3.88E+01	1.53E+01	3.08E+01	4.76E+01
F23	**Mean**	1.16E+03‡	5.66E+02‡	1.12E+03‡	5.34E+02‡	1.10E+03‡	5.17E+02
	StdDev	2.33E+02	1.08E+02	1.97E+02	3.70E-04	3.59E+00	3.66E-04
F24	**Mean**	1.15E+03‡	6.20E+02‡	2.66E+02‡	2.00E+02‡	9.42E+02‡	1.44E+02
	StdDev	4.24E+02	3.60E+02	2.29E+02	6.27E-13	4.04E+00	6.83E-13
F25	**Mean**	1.18E+03‡	2.13E+02‡	4.49E+02‡	2.21E+02‡	2.17E+02‡	2.00E+02
	StdDev	4.06E+02	1.67E+00	4.31E+02	2.32E+00	1.70E+00	3.69E+00

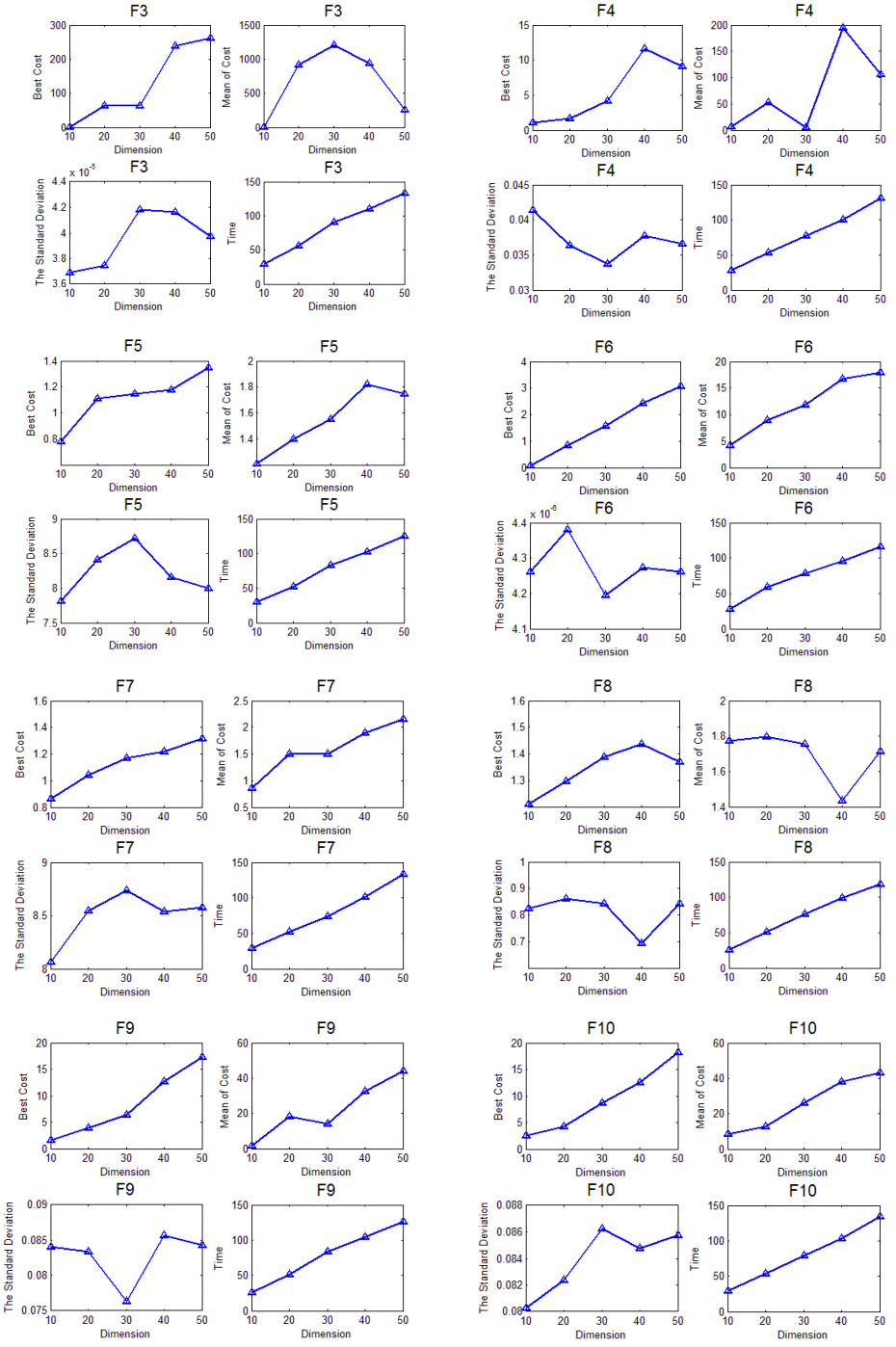
علامت "‡" **یعنی عملکرد الگوریتم پیشنهادی بهتر بوده، علامت"†" یعنی عملکرد پیشنهادی بدتر بوده و علامت "§" به معنی عملکرد یکسان بوده است.**

Friedman Test

	4.21	4.29	3.85	4.53	3.11	4.12
p-value	3.38E-06					
Statistic	40.320					

Results of the comparison between the proposed algorithm and new methods showed that in all 25 functions the proposed algorithm had better quality. In functions F1 and F9 the proposed method jointly with another algorithm could have the best performance. In function F1, the CS algorithm and the proposed approach had the best performance due to reaching the optimal point zero. The same process was found in the proposed algorithm and DSA algorithm in function F9. The standard deviation values of the proposed method was often high in most comparisons and this indicates the ability of algorithms search diversity in the problem space. The statistical test applied on the results of Table 4-3 shows that the values of the results obtained by the methods had no significant relationship with the random process. Figure 10 shows the proposed method diagram for 14 tested functions on different values of dimensions or number of variables. The results showed that as the number of dimensions increased and the problem became more complicated to resolve, the value of the best cost will be ascending. The diagrams were applied for 20 populations, 10000 times assessments and 10, 20, 30, 40 and 50 variables.




F3
Best Cost
Dimension
F3
Mean of Cost
Dimension
F3
The Standard Deviation
x 10^-5
Dimension
F3
Time
Dimension
F4
Best Cost
Dimension
F4
Mean of Cost
Dimension
F4
The Standard Deviation
Dimension
F4
Time
Dimension
F5
Best Cost
Dimension
F5
Mean of Cost
Dimension
F5
The Standard Deviation
Dimension
F5
Time
Dimension
F6
Best Cost
Dimension
F6
Mean of Cost
Dimension
F6
The Standard Deviation
x 10^-5
Dimension
F6
Time
Dimension
F7
Best Cost
Dimension
F7
Mean of Cost
Dimension
F7
The Standard Deviation
Dimension
F7
Time
Dimension
F8
Best Cost
Dimension
F8
Mean of Cost
Dimension
F8
The Standard Deviation
Dimension
F8
Time
Dimension
F9
Best Cost
Dimension
F9
Mean of Cost
Dimension
F9
The Standard Deviation
Dimension
F9
Time
Dimension
F10
Best Cost
Dimension
F10
Mean of Cost
Dimension
F10
The Standard Deviation
Dimension
F10
Time
Dimension

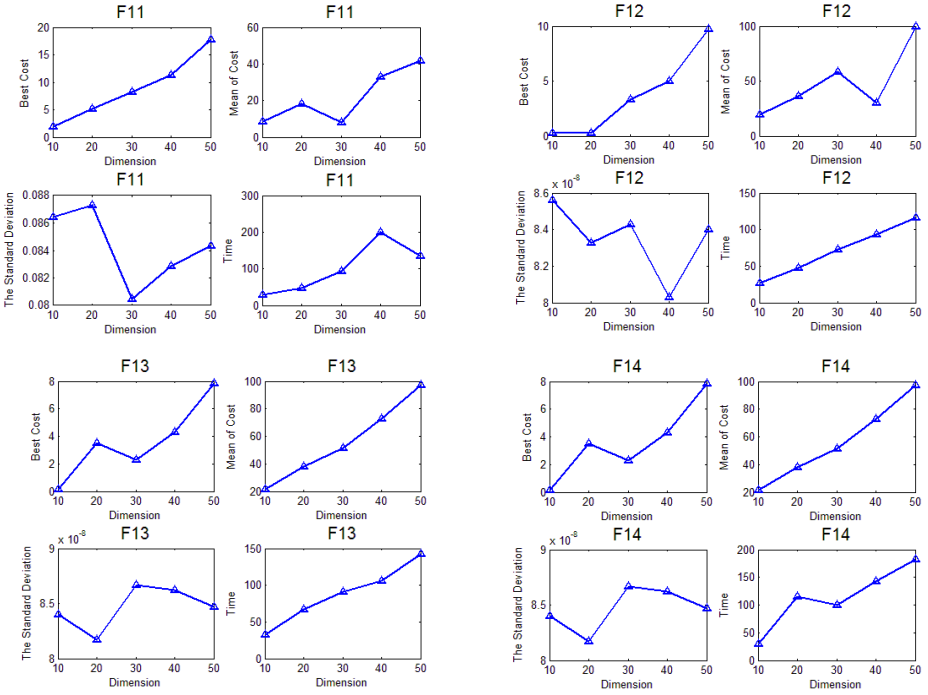


Fig 8. The diagram for the best cost, mean cost, standard deviation and the implementation time of the proposed method on 14 test functions

4. CONCLUSION

Nowadays, the algorithms inspired by the nature are helping human beings in many scientific studies. These algorithms were generated by inspiration from personal and social behaviors available in the nature and have shown that there is a disciplined and purposeful structure in the nature. Using the instinctive behaviors of birds in this paper, we provided an algorithm that could perform more accurately, more targeted and more controlled than the basic swarm algorithm in a problem space. We modeled the classical conditioning learning behavior and implemented a normal task based on a natural stimulant for each particle in the search space. On this basis, each particle learned that when facing with low variation in its category space, it should move towards its optimal personal experience, and if the category was high-diversity, it should incline towards the general optimum of its category.

We also used the assumption that low-energy birds would face problems in the flight space, and generated the initial population according to the elite particles. In the third idea of our algorithm, we valued the problem space based

on the merits and tried to apply some changes in velocity equations of the particles so that the particles' velocity in valuable spaces would be low in order to conduct more search. Conversely, the particles were moved more quickly in low-value spaces to get far from those spaces more rapidly. The simulation was carried out in MATLAB software and the results of the proposed method in the series of CEC 2005 tests were used. As computer scientists who are interested in the field of nanotechnology, our current work involves building systems that consist of a large number of particles automatically forming into a designed structure. It was shown that the proposed method finds a good solution to the problem regardless of nondeterministic functions or stochastic conditions.

It is anticipated that models such as these will lead to successful bottom–up nanotechnology systems in the future.

REFERENCES

[1] R.L. Haupt, S. Ellen Haupt, *Practical genetic algorithms*. (2004).

[2] S. Boyd, S.P. Boyd, L. Vandenberghe, *Convex Optimization*, Cambridge University Press, (2004)

[3] D. Ezzat, S. Amin, H.A. Shedeed, & M. F. Tolba, *A new nano-robots control strategy for killing cancer cells using quorum sensing technique and directed particle swarm optimization algorithm*. In International Conference on Advanced Machine Learning Technologies and Applications (pp. 218-226). Springer, Cham. (2019, March)

[4] W. Sun, Y. Yuan., *Optimization Theory and Methods: Nonlinear Programming, Springer Science + Business Media, LLC Press*, (2006)

[5] H. D. Phan, K. Ellis, J. C. Barca, & A. Dorin, *A survey of dynamic parameter setting methods for nature-inspired swarm intelligence algorithms*. Neural Computing and Applications, 1-22. E. H. Miller, A note on reflector arrays, IEEE Trans. Antennas Propagat., to be published. (2020).

[6] S. Hazra, & P. K. Roy, *Newly-Developed Swarm Intelligence Algorithms Applied to Renewable Energy-Based Load Dispatch Real-World Problems*. In Handbook of Research on Advancements of Swarm Intelligence Algorithms for Solving Real-World Problems (pp. 1-26). IGI Global (2020).

[7] X. Gong, L. Liu, S. Fong, Q. Xu, T. Wen, & Z. Liu, *Comparative Research of Swam Intelligence Clustering Algorithms for Analyzing Medical Data*. IEEE Access, 7, 137560-137569 (2019).

[8] J. P. Devarajan, & T. P. Robert, *Swarm Intelligent Data Aggregation in Wireless Sensor Network*. International Journal of Swarm Intelligence Research (IJSIR), 11(2), 1-18. (2020)

[9] Y. C. Lee, & J. Y. Moon, *Bio-Nanorobotics: Mimicking Life at the Nanoscale*. In Introduction to Bionanotechnology (pp. 93-114). Springer, Singapore. R. J. Vidmar. On the use of atmospheric plasmas as electromagnetic reflectors. IEEE Trans. Plasma Sci. [Online]. 21(3) (1992, Aug.) 876–880. Available: http://www.halcyon.com/pub/journals/21ps03-vidmar (2020)

[10] E. Mendez-Flores, E. M. Martinez-Galicia, J. D. J. Lozoya-Santos, R. Ramirez-Mendoza, R. Morales-Menendez, I. Macias-Hidalgo, A. & Molina-Gutierrez. *Self-Balancing Robot Control Optimization Using PSO*. In 2020 5th International Conference on Control and Robotics Engineering (ICCRE) (pp. 7-10). IEEE (2020, April).

[11] M. Li, N. Xi, Y. Wang, & L. Liu. *Progress in nanorobotics for advancing biomedicine*. IEEE Transactions on Biomedical Engineering. (2020)

[12] A. M. R. Kabir, D. Inoue, & A. Kakugo. *Molecular swarm robots: recent progress and future challenges*. Science and Technology of Advanced Materials, (just-accepted). (2020).

[13] J. Kennedy, R. C. Eberhart. *Particle Swarm Optimization*, Proceedings of the 4th IEEE International Conference on Neural Networks, pp. 1942-1948.(1995)

[14] N. F. Wan, L. Nolle, *Solving a multi-dimensional knapsack problem using hybrid particle*. 23rd European Conference on Modelling and Simulation. (2008)

[15] K. B. Deep, *A socio-cognitive particle swarm optimization for multi-dimensional*. First International Conference on Emerging Trends in Engineering and, pp. 355–360. (2008).

[16] X. Shen, Y. Li, C. Chen, J. Yang, D. Zhang. *Greedy continuous particle swarm optimisation algorithm for the knapsack problems.* International Journal of Computer Applications in Technology 44 (2), 37–144. (2012).

[17] H. S. Lopes, L. S. Coelho. *Particle swarn optimization with fast local search for the blind traveling salesman problem*. International Conference on Hybrid Intelligent Systems, pp. 245–250. (2005)

[18] H. Zhou, M. Song, W. Pedrycz,.*A comparative study of improved GA and PSO in solving multiple traveling salesmen problem*. Applied Soft Computing, 64, 564-580. (2018).

[19] A. Banharnsakun, B. Sirinaovakul, T. Achalakul. *Job shop scheduling with the best-so-far ABC*. Engineering Applications of Artificial Intelligence 25 (3), pp. 583–593. (2012)

[20] D. Karaboga, B. Gorkemli, *A combinatorial artificial bee colony algorithm for traveling salesman problem*. International Symposium on Intelligent Systems and Applications, pp. 50–53. (2011)

[21] Z. Geem, J. Kim, G. Loganathan. *A new heuristic optimization algorithm: Harmony search.Simulation*, 60. (2001)

[22] D. T. Pham, A. Ghanbarzadeh, E. Koc, S. Otri, S. Rahim, M. Zaidi, *The bees algorithm.* Technical note, Cardiff University, UK: Manufacturing Engineering Center. (2005)

[23] D. T. Pham, S. Otri, A. Afify, M. Mahmuddin, H. Al-Jabbouli, *Data clustering using the bees algorithm*. 40thCIRPInternational Seminar on Manufacturing Systems, p. p. s.p. (2007)

[24] D. Pham, E. Koc, J. Lee, J. Phrueksanant, *Using the bees algorithm to schedule jobs for a machine*. Proceedings of Eighth International Conference on Laser Metrology, pp. 430–439, CMM and Machine. (2007)

[25] S. Hazra, & P. K. Roy,. *Newly-Developed Swarm Intelligence Algorithms Applied to Renewable Energy-Based Load Dispatch Real-World Problems*. In Handbook of Research on Advancements of Swarm Intelligence Algorithms for Solving Real-World Problems (pp. 1-26). IGI Global. (2020)

[26] P. Civicioglu. *Transforming geocentric cartesian coordinates to geodeticcoordinates by using differential search algorithm.* Comput, Geosciuk, vol. 46,no. 9, pp. 229-247, Sep. (2012)

[27] A. Gandomi, *Bird mating optimizer: An optimization algorithm inspired by birdmating strategies*. Commun Nonlinear Sci, vol. 19, no. 4, pp. 1213-1228, Apr. (2014)

[28] A. Draa, S. Bouzoubia, L. Boukhalfa, *A sinusoidal differential evolution algorithmfor numerical optimisation*, Appl. Soft Comput. 27 (2015) 99–126. (2015).

[29] G. Sun, R. Zhao, Y. Lan, *Joint operations algorithm for large-scale global optimization*. Applied Soft Computing, 38: 1025-1039. (2016).

[30] X. Yan, F. He, N. Hou, & H. Ai. *An efficient particle swarm optimization for large-scale hardware/software co-design system*. International Journal of Cooperative Information Systems, 27(01), 1741001. (2018)

[31] B. Tang, Z. Zhu, H. S. Shin, A. Tsourdos, & J. Luo. *A framework for multi-objective optimisation based on a new self-adaptive particle swarm optimisation algorithm*. Information Sciences, 420, 364-385. (2017)

[32] H. Su, Z. Fu, & Z. Wen. *SFPSO algorithm-based multi-scale progressive inversion identification for structural damage in concrete cut-off wall of embankment dam*. Applied Soft Computing, 84, 105679. (2019)

[33] X. Xu, Y. Tang, J. Li, C. C. Hua, X. P. Guan, *Dynamic multi-swarm particle swarmoptimizer with cooperative learning strategy*, Appl. Soft Comput. 29, 169–183. (2015).

[34] I. G. Tsoulos, A. Tzallas, & E. Karvounis, *Improving the PSO method for global optimization problems*. Evolving Systems, 1-9. (2020)

[35] M. Yasrebi, A. E. Baghban, H. Parvin, H., & M. Mohammadpour Optimisation *inspiring from behaviour of raining in nature: droplet optimisation algorithm.* International Journal of Bio-Inspired Computation, 12(3), 152. (2018).

[36] X. S. Yang, S. Deb. S. Cuckoo, *search via Levy fligh*ts, *in Proc*. NaBIC 2009, IEEE Publications, pp. 210-214, Dec. 2009. 18 / Information Sciences XX 1–22 19. (2014)

[37] E. Rashedi, H. Nezamabadi-pour, S, Saryazdi. G*SA: A Gravitational Search Algorithm, Inform. Sciences*, vol. 179, no. 13, pp. 2232-2248. (2009)

[38] P. N. Suganthan, N. Hansen, J. J. Liang, *Problem definitions and evaluation criteria for the CEC 2005 Special Session on Real Parameter Optimizatio*n, Nanyang Technological University, Singapore, Tech. Rep, May. 2005[Online].

Available: http:// www3.ntu.edu.sg/home/EPNSugan/index f iles/CEC-05/Tech- Repot-May-30-05.pdf. (2005).