Araştırma Makalesi
Adıyaman Üniversitesi Mühendislik Bilimleri Dergisi
17 (2022) 366-383

AN IMPROVED ARTIFICIAL ATOM ALGORITHM WITH THE OPERATOR OF SHUFFLED FROG LEAPING ALGORITHM

Ayşe Erdoğan Yıldırım[1*]

[1]Fırat Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Elazığ, 23119, Türkiye
Geliş Tarihi/Received Date: 01.03.2022 Kabul Tarihi/Accepted Date: 18.04.2022 DOI: 10.54365/adyumbd.1080995

ABSTRACT

Artificial Atom Algorithm is an optimization technique that developed inspired by nature. This algorithm used for both continues problems and discrete problems in previous studies. In this study, an arrangement that would increase the success of this algorithm was envisaged. For this purpose, the ionic bond function of Artificial Atom Algorithm has been improved benefiting an algorithmic step of Shuffled Frog Leaping Algorithm. As a result of the updates, the search space was narrowed for the ionic bond operator. Thus, the state of getting away from the solution in each iteration was prevented. The success of Improved Artificial Atom Algorithm was tested with benchmark functions. Experimental results for the proposed method were interpreted comparatively.

Keywords: *Artificial atom algorithm, Improved artificial atom algorithm, Shuffled frog leaping algorithm, Optimization, Benchmark problems.*

ADAPTİF KURBAĞA SIÇRAMA ALGORİTMASININ OPERATÖRÜ İLE GELİŞTİRİLMİŞ YAPAY ATOM ALGORİTMASI

ÖZET

Yapay Atom Algoritması, doğadan ilham alınarak geliştirilmiş bir optimizasyon tekniğidir. Bu algoritma önceki çalışmalarda hem sürekli problemler hem de ayrık problemler için kullanılmıştır. Bu çalışmada bu algoritmanın başarısını artıracak bir düzenleme öngörülmüştür. Bu amaçla, Yapay Atom Algoritmasının iyonik bağ işlevi, Adaptif Kurbağa Sıçrama Algoritmasının algoritmik bir adımından yararlanılarak geliştirilmiştir. Güncellemeler sonucunda iyonik bağ operatörü için arama alanı daraltılmıştır. Böylece her iterasyonda çözümden uzaklaşma durumu önlenmiştir. Geliştirilmiş Yapay Atom Algoritmasının başarısı, kıyaslama fonksiyonları ile test edilmiştir. Önerilen yöntem için deneysel sonuçlar karşılaştırmalı olarak yorumlanmıştır.

Anahtar Kelimeler: *Yapay atom algoritması, Geliştirilmiş yapay atom algoritması, Adaptif kurbağa sıçrama algoritması, Optimizasyon, Kıyaslama problemleri.*

1. Introduction

Optimization can be defined as objective function making minimum or maximum for specific problems. Optimization problems investigate parameter values that provide the minimum or maximum values for objective function. Various optimization algorithms are used to solve these problems. These algorithms are divided into two groups in terms of the used methods. These are deterministic and heuristic optimization algorithms [1]. Some of the heuristic optimization algorithms are Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Sapling Growing up Algorithm, Bee Colony Optimization (BCO), Ant Colony Algorithm (ACO), Differential Evolution Algorithm (DE), Parliamentary Optimization Algorithm, Uniform Big Bang - Chaotic Big Crunch Optimization,

* [1]e-posta: aerdoganyildirim@gmail.com ORCID ID: https://orcid.org/0000-0002-8983-8410 (Sorumlu Yazar)

Artificial Chemical Reaction Algorithm, Cricket Algorithm, Bat Algorithm, Harmony Search (HS), Shuffled Frog Leaping Algorithm and Artificial Atom Algorithm (A3) [2-15].

GA is the most well-known heuristic method that aims to increase the fitness values of chromosomes by using basic operators such as selection, crossover, mutation [16]. PSO is an algorithm developed by modeling the feature of mimicking the behavior of the leader of fish and bird flocks. According to the algorithm, the position value of the one with the best fitness value is used to update the values of the others [17]. Saplings Growing up Algorithm consists of saplings planting, matching, branching and grafting operators. By planting saplings, it is ensured that a uniformly distributed solution set is formed. With matching, genetic information is exchanged between saplings [4]. Branching and grafting operators have been combined and named as mix-develop since they cause too much divergence from the solution [18]. BCO is an intuitive method that models the processes of searching the environment for a substance called nectar, which bees need to find in order to make honey in their hives, and informing the bees in the hive when they find it. If the neighbor is better than the memory, the new one overwrites the old search region [5]. It is thought that the change in the amount of a substance called pheromone secreted by ants when they use a road helps to guide the ants coming from behind and to choose the shortest path. ACO was developed by modeling this situation. The functioning of the algorithm depends on the amount of pheromone. The pheromone amount of the candidate with the best fitness function is updated [6]. DE is a population-based algorithm similar to the genetic algorithm. DE is more efficient for continuous data. Algorithm operators (crossover, mutation) are applied to each chromosome in the population one by one [19]. The Parliamentary Optimization method is designed by modeling systems in which there is an assembly and parties competing with each other, and the parties have candidates competing with each other within their own groups. Accordingly, the unification process at the end of the in-group and out-group race determines the result of the selection [20]. The Big Bang-Big Crunch Algorithm is based on the Big Bang theory and is inspired by the events thought to have occurred when the universe was formed. The algorithm mimics the bursting and shrinking processes by random distribution and random aggregation of solutions. The center of mass of the particles coming together is calculated. The algorithm is continued with new bursts and contractions. The algorithm terminates when the center of mass no longer changes [21]. Uniform Big Bang - Chaotic Big Crunch Optimization is designed to improve the performance of Big Bang - Big Crunch Optimization and it proposes efficient methods to overcome premature convergence [9]. Artificial Chemical Reaction Algorithm was developed by modeling chemical reactions. First, the molecules that will react are determined. These molecules undergo certain reactions. Accordingly, a chemical stability test is performed. As a result of this test, molecules with good values are preserved, while those with bad values are discarded [10]. The cricket algorithm was created by modeling the ability of crickets to predict the temperature of the air by flapping their wings. The loudness of the crickets gives the value of the fitness function of the optimization problem. Air temperature, speed and frequency of sound are used to update the positions of the crickets. If the fitness value is better at the new location, this value is taken to the population [11]. The Bat Algorithm is an optimization algorithm inspired by the behavior of bats to determine the direction and distance of an object by making use of the echolocation of the sound [12]. The HS algorithm is a music-based optimization method that aims to find the best result by using random selection, harmony memory consideration and pitch adjustment operators [13].

Optimization algorithms aim to find the global optimum, not to find the local optimal solution. The heuristic methods make this by improving the obtained solutions iteratively. In most heuristic methods, while improving candidate solutions, part of the candidate solution set is randomly refreshed or new elements are included in the set [22]. However, rather than production a completely new value, searching a solution using old solutions can yield more effective results.

In this article, it was aimed to achieve more successful results for optimization problems by making improvements on the ionic bond operator of A3. Therefore, A3 was improved by utilizing one of the new frog generation operators of Shuffled Frog Leaping Algorithm. The success of the proposed algorithm was evaluated in comparison with benchmark problems.

The contributions of this study to science:

- The A^3 algorithm, which has an important place among meta-heuristic algorithms, was developed by utilizing the new frog generation operator of the Shuffled Frog Leaping Algorithm. Thus, the random search space used by the ionic bond operator is narrowed down at each step. In this way, the performance of the A^3 algorithm has been improved.

- With the improvement made on A^3, the rate of reaching of the algorithm the global optimum result with lower iteration numbers in the optimization problems has been increased.

- The algorithm has been applied to 11 benchmark test functions. In most of the test functions, the global optimum result has been reached or quite approached optimum result.

- The algorithm has been compared with the meta-heuristic methods that are widely used in the literature and the studies carried out in recent years. The results reveal the superiority of the Improved A^3 method over other methods for benchmark functions.

In Section 2, firstly, the general features of meta-heuristic algorithms are mentioned. Then, it is given the information about Shuffled Frog Leaping Algorithm and A^3. Section 3 describes the changes made to improve A^3. In Section 4, the benchmark problems which are utilized to prove the success of the proposed method are given. In addition, parameter settings for the algorithms and application results in comparative are also given with tables and figures. In Section 5, the studies are summarized and the results are interpreted.

2. Meta-heuristic Algorithms

Heuristic algorithms are methods which do not guarantee to always put forth the same and optimum result. These algorithms have been developed to keep the run-time and accuracy rate in balance. Meta-heuristic algorithms choose and use methods that are considered to be more advantageous between different heuristic algorithms. Meta-heuristic algorithms have been developed by inspired from nature and environment, such as animal species, atoms and chemical reactions, sports branches, behaviors or movements by modeling [22-23].

2.1. Shuffled Frog Leaping Algorithm

Eusuff and Lansey [14] design Shuffled Frog Leaping Algorithm as a combination of Memetic Algorithm and Particle Swarm Optimization. It is a meta-heuristic algorithm, inspired by the evolutionary development of a frog which has researched for food and jumped in a swamp. The basic logic of the algorithm is based to make improvements on a population consisting of frogs. This population is divided into sections called memeplex which represents different frog species. Frogs improve affected by frogs in their own memeplex. When the number of memetic evolutions reaches a certain value, data is exchanged between different memeplexes and new memeplexes are created. In this way, the algorithm tries to reach local and global optimal solutions. The algorithm terminates by applying a certain number of local search and mixing operations [24].

The algorithmic operation of the Shuffled Frog Leaping Algorithm is as follows:

1. A random research space, which is consisted of candidate solutions, each of which is called as a frog, is created.
2. The objective function values are calculated for each frog.
3. The research space is ranked from the best solution to the worst solution.
4. The research space is divided into sections called memeplexes.
5. Starting from the best, the frogs are selected as many as the number of memeplexes and they are distributed to each memeplex. This process is repeated for all frogs.
6. Then, the local search operation is performed in each memeplex.

a) The local search process aims to improve the worst element in each memeplex. For this purpose, the worst frog is removed from the memeplex.
b) Instead, a new one is included into the memeplex according to the following formula.

$$x_{new} = x_{old} + (x_{localbest} - x_{worst})^* \text{ rand} \quad (1)$$

c) If the new frog is no better than the old, the following formula is applied.

$$x_{new} = x_{old} + (x_{globalbest} - x_{worst})^* \text{ rand} \quad (2)$$

d) If the new frog is no better than the old, the new frog is randomly produced.

7. After the local search, repeat from Step 3 until the termination condition is met [14].

The flow diagram of Shuffled Frog Leaping Algorithm is given in Fig. 1.

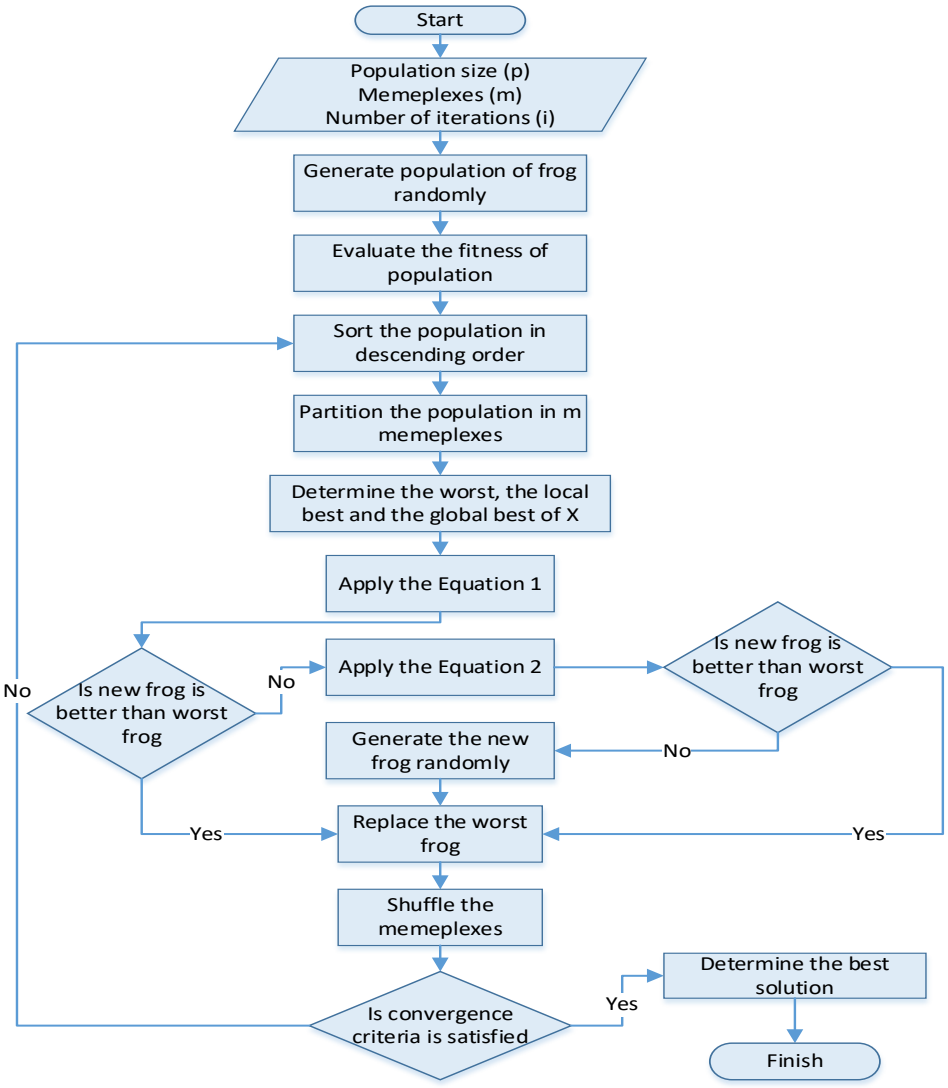


Fig. 1. The flow diagram of Shuffled Frog Leaping Algorithm

2.2. Artificial Atom Algorithm

Artificial Atom Algorithm (A^3) which is inspired by chemical reactions, is a meta-heuristic method. This algorithm carries out operations on electrons, atoms and an atom set. The algorithm uses

two algorithmic operators. A^3 aims to maintain and replicate best solutions with the covalent bond operator. It tries to achieve the global optimal value with the ionic bond operator. For this purpose, first covalent bond then ionic bond is applied to the randomly generated atom set within its boundaries. Obtained new atoms are evaluated using the objective function. In addition, electrons are evaluated in terms of their effect on the objective function. Subsequently, electrons and atoms are ordered in themselves. This process continues with the number of iterations [15]. A^3 provides computational time advantage, especially for benchmark problems [25]. The flow diagram of A^3 is given in Fig. 2.

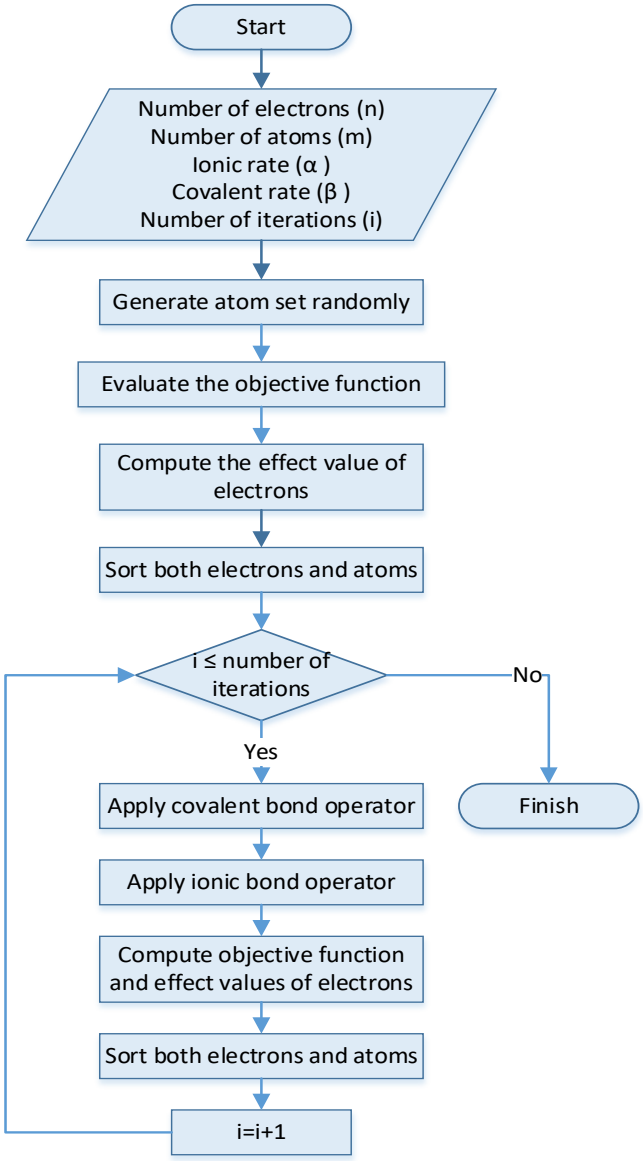


Fig. 2. The flow diagram of Artificial Atom Algorithm (A^3)

3. Improved Artificial Atom Algorithm

The success of A^3 on benchmark problems and discrete problems has been shown in previous studies [15, 26-30]. It has known that A^3 has achieved quite fast results especially in benchmark problems [25]. It seems to have been influenced by replication principle of the covalent bond operator of A^3 on this case. According to this principle, the better ones are replicated in the atom set. While the ionic bond operator regenerates a portion of the atom set with random elements, even if it is positive to

achieve the global best, but it prevents the search space to narrow. Therefore, in this article, the idea of improving the A^3 is proposed. It has been found appropriate to make this improvement on the ionic bond operator of A^3. For this improvement, it was inspired by an algorithmic step of Shuffled Frog Leaping Algorithm.

The ionic bond function generates random numbers within boundaries of the attributes for the ionic area of the atom set. This operator is shown in Algorithm 1 [15].

Algorithm 1: Ionic Bond Operator

$IonicBond\ (A, m, n, \beta)$
  $j \leftarrow 1, \dots, m$ // m : Number of atoms
    $i \leftarrow \beta n + 1, \dots, n$ // β : Covalent rate
      // n : Number of electrons
      $A_j[i] \leftarrow L_i + \eta * (U_i - L_i)$
// $A_j[i] \in AtomSet$
// η : A randomly generated number between 0 and 1
// U_i : The upper limit for i
// L_i : The lower limit for i

Shuffled Frog Leaping Algorithm also generates random numbers in the frog generation stage. However, it benefits from the best and worst results (frogs) when generating random numbers. For this, it uses following formula [24].

$$x_{new} = x_{old} + (x_{best} - x_{worst}) * \text{rand} \tag{3}$$

Therefore, the advantages of these two heuristic algorithms were evaluated and the ionic bond operator of A^3 was improved. Thus, the ionic bond function was modified as follows (Algorithm 2):

Algorithm 2: Improved Ionic Bond Operator

$IonicBond\ (A, m, n, \beta)$
  $j \leftarrow 1, \dots, m$ // m : Number of atoms
    $i \leftarrow \beta n + 1, \dots, n$ // β : Covalent rate
      // n : Number of electrons
      $temp \leftarrow |A_j[i_{old\_local\_best}] - A_j[i_{old\_worst}]| * (L_i + \eta * (U_i - L_i))$
      $X = IsInRange(temp, L_i, U_i)$
      $if(X == true)$
        $A_j[i] = temp$
      end
// $A_j[i] \in AtomSet$
// η : A randomly generated number between 0 and 1
// U_i : The upper limit for i
// L_i : The lower limit for i

With the update, it is aimed to narrow the search space with the ionic bond function over time. Otherwise, with the ionic bond function, candidate solutions that causative to remove from optimal results are included from the wide search space in the atom set. This progress helps A^3 achieve optimum more quickly.

4. Experimental Results

As a result of the improvement in A^3, it was tested using benchmark functions to see how the performance changed. For this purpose, six benchmark problems have been identified. The functional definitions of these problems and the bounds of the parameters are given in Table 1 [31].

Table 1. Used benchmark functions in the application

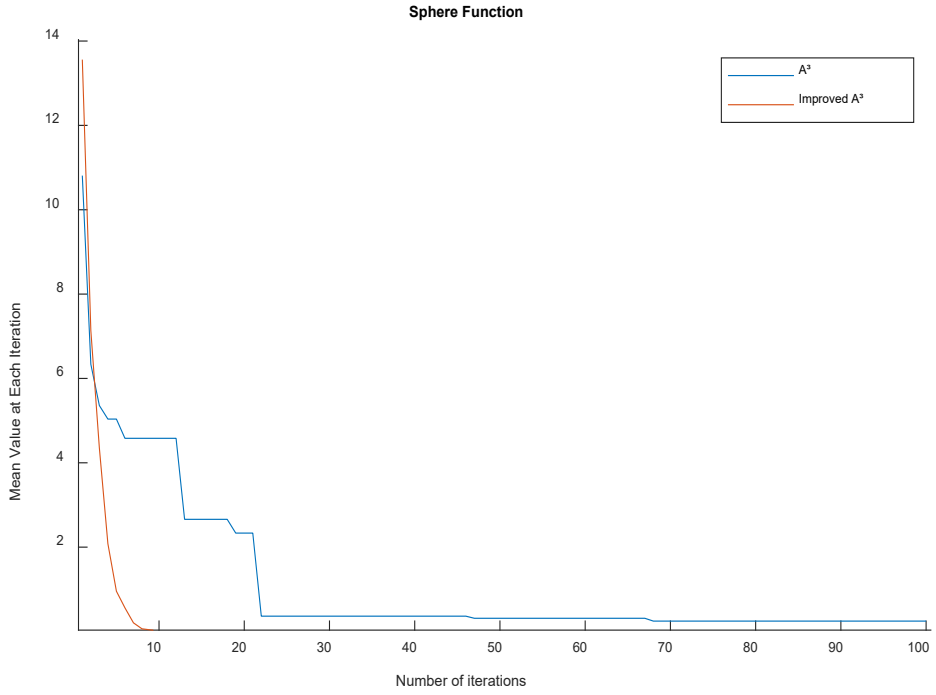
Functions	Definition of Functions	Bounds
Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$	$-5.12 \leq x_i \leq 5.12$
Rotated Hyper-Ellipsoid	$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{i} x_j^2$	$-100 \leq x_i \leq 100$
Parallel Hyper-Ellipsoid	$f(x) = \sum_{i=1}^{n} i x_i^2$	$-5.12 \leq x_i \leq 5.12$
Rastrigin (Rotated Rastrigin)	$f(x) = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)]$	$-5.12 \leq x_i \leq 5.12$
Griewank	$f(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$-600 \leq x_i \leq 600$
Quartic	$f(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1)$	$-1.28 \leq x_i \leq 1.28$

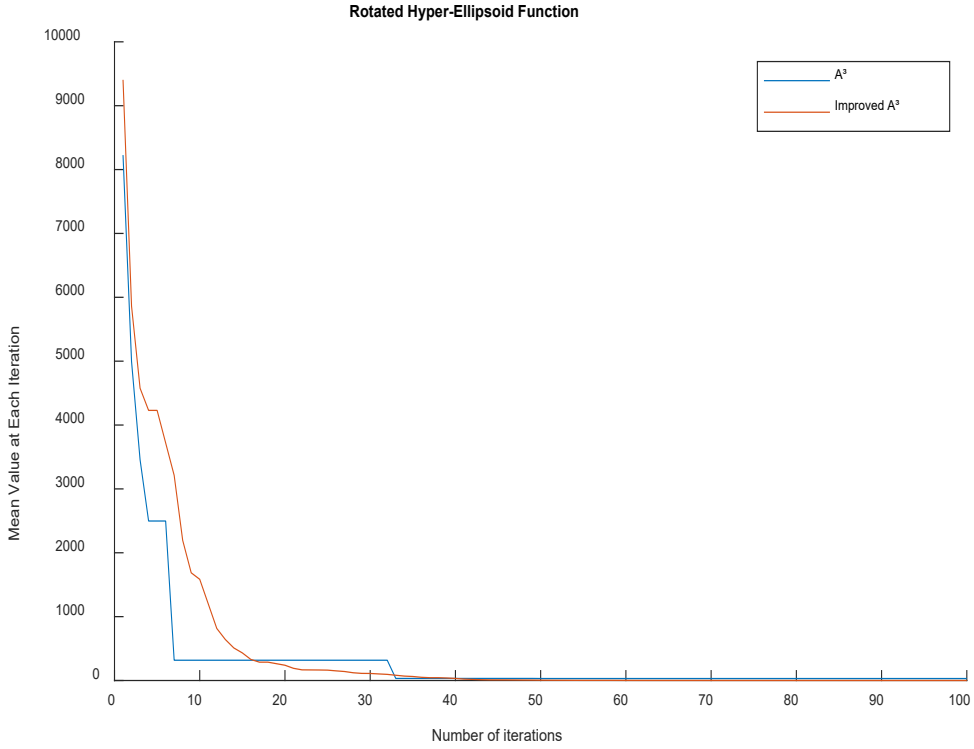
In the solution of these benchmark problems for A^3 and Improved A^3, the determined parameter settings are presented in Table 2. The same parameter values have been used for A^3 and Improved A^3.

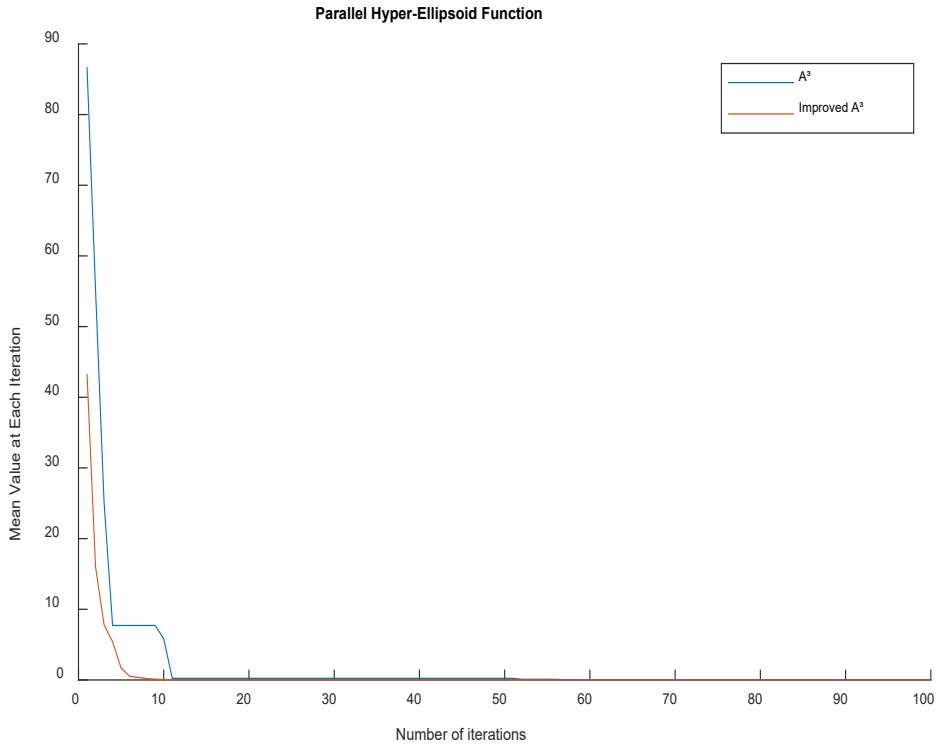
Table 2. Parameter settings of A^3 and Improved A^3 for benchmark problems

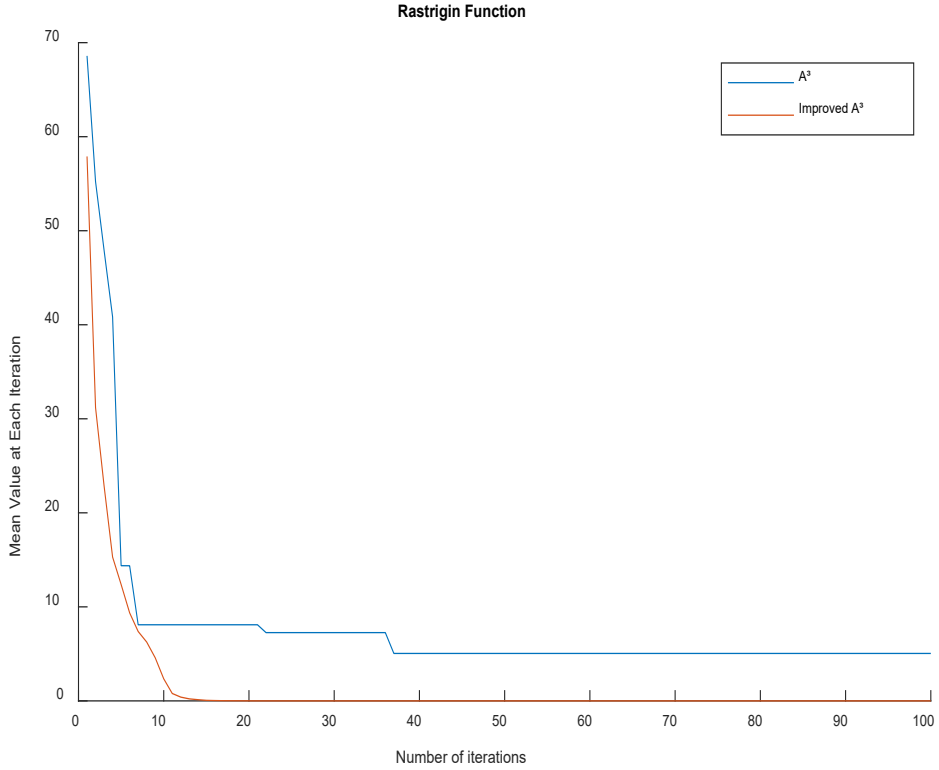
Parameter	Value
Number of electrons	50
Number of atoms	10
Number of iterations	100/1000
Covalent rate (β)	0.6
Ionic rate (α)	0.4

In Fig. 3, it is seen changes that occur in the mean values at each iteration with A^3 and Improved A^3 applications for Sphere, Rotated Hyper-Ellipsoid, Parallel Hyper-Ellipsoid, Rastrigin, Griewank and Quartic functions.


Sphere Function
A³
Improved A³
Mean Value at Each Iteration
Number of iterations
14
12
10
8
6
4
2
10
20
30
40
50
60
70
80
90
100


Rotated Hyper-Ellipsoid Function
A³
Improved A³
Mean Value at Each Iteration
Number of iterations
10000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
0
10
20
30
40
50
60
70
80
90
100


Parallel Hyper-Ellipsoid Function
A³
Improved A³
Mean Value at Each Iteration
Number of iterations
0
10
20
30
40
50
60
70
80
90
100


Rastrigin Function
A³
Improved A³
Mean Value at Each Iteration
Number of iterations
0
10
20
30
40
50
60
70
80
90
100

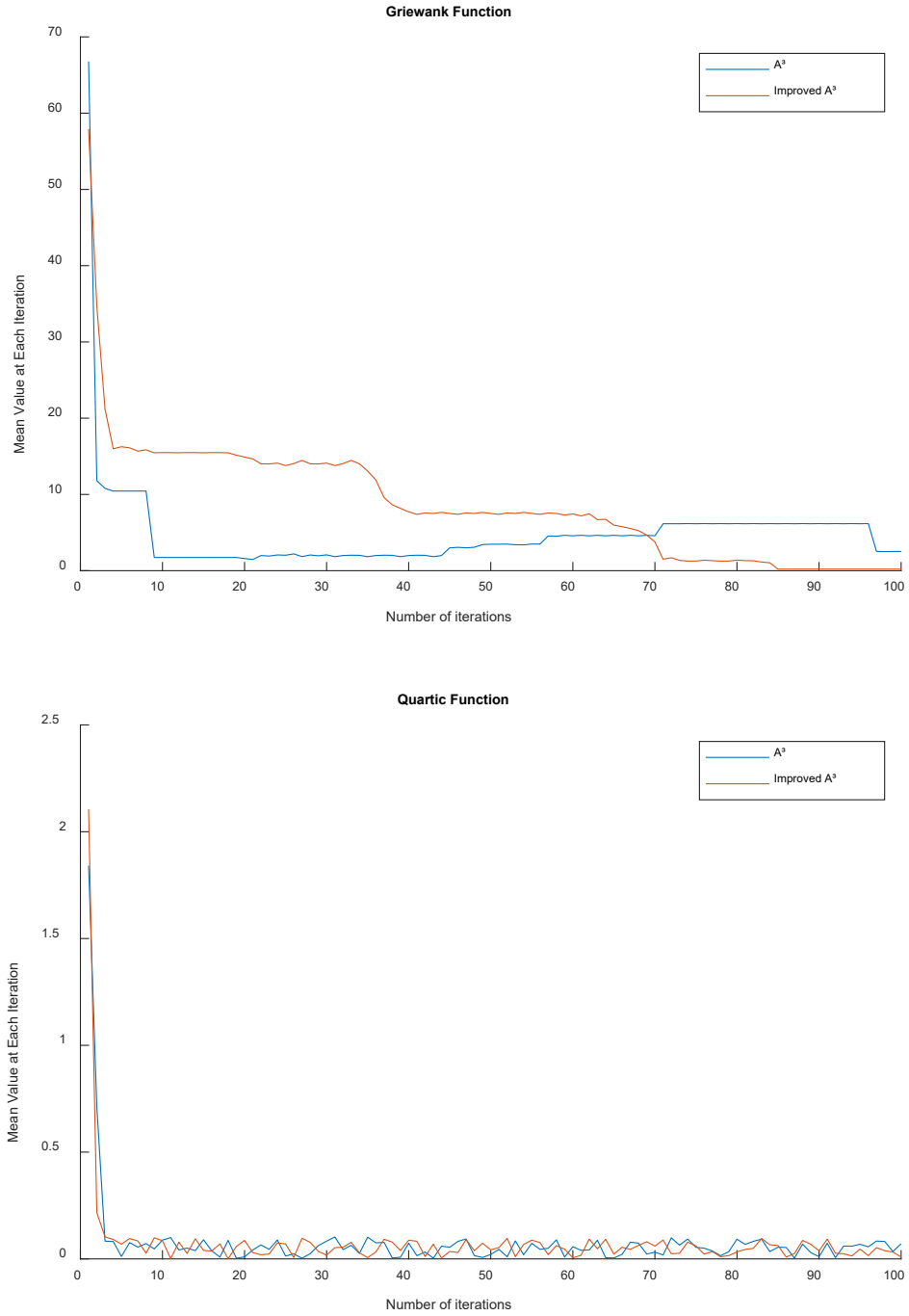


Fig. 3 Application results of A^3 and Improved A^3 for benchmark problems

The performance of the A^3 and Improved A^3 with the operator of the Shuffled Frog Leaping Algorithm for the selected six benchmark problems were compared. The best results were obtained by running both algorithms 20 times. These comparative results are given in Table 3. In addition, global optimum values for these benchmark problems are also shown in Table 3. In addition, the best result shows the best result obtained by running the algorithms 20 times. The mean result shows the average result obtained by running the algorithms 20 times. Here, it should be taken into account that the number of iterations is kept very low in order to show how quickly the proposed method reaches the optimum result.

Table 3. The experimental results for A^3 and Improved A^3 applications to benchmark problems

Benchmark Problems	Global Min	Iteration Number	A^3			Improved A^3		
			Mean result	Best result	Run-time (s)	Mean result	Best result	Run-time (s)
Sphere	0.0	100	0.2145	1.788E-02	0.0100	**1.0022E-34**	**2.0171E-37**	0.1389
		1000	0.0022	1.897E-04	0.0604	**0.0**	**0.0**	1.3753
Rotated Hyper-Ellipsoid	0.0	100	642.931	149.672	0.0122	**5.4277E-06**	**7.6720E-08**	0.1322
		1000	6.6968	0.0335	0.0852	**3.1842E-82**	**2.3666E-91**	1.2256
Parallel Hyper-Ellipsoid	0.0	100	5.1782	0.8609	0.0109	**1.0356E-31**	**1.4878E-34**	0.1370
		1000	0.0301	0.0018	0.0623	**0.0**	**0.0**	1.3276
Rastrigin (Rotated Rastrigin)	0.0	100	20.305	14.4373	0.0122	**0.0**	**0.0**	0.1350
		1000	0.7855	0.0226	0.0841	**0.0**	**0.0**	1.3195
Griewank	0.0	100	8.6828	1.8784	0.0160	**0.9270**	**7.0538E-07**	0.1240
		1000	1.5229	0.0193	0.1198	**0.0**	**0.0**	1.2114
Quartic	0.0	100	3.006E-04	8.329E-09	0.0108	**3.2809E-141**	**1.7267E-147**	0.1577
		1000	1.598E-07	2.337E-11	0.0802	**0.0**	**0.0**	1.4469

It has been observed that the proposed method achieves optimal results for all selected comparison problems, except for the Rotated Hyper-Ellipsoid problem. Furthermore, when A^3 and Improved A^3 are compared, it is seen that Improved A^3 is quite advantageous in terms of best results; on the other hand, A^3 is advantageous in terms of run-time performance.

The Improved A^3 method was compared with Genetic Algorithm (GA), Ant Colony Optimization (ACO), Bee Colony Optimization (BCO), Particle Swarm Optimization (PSO), Differential Evolution (DE), Harmony Search (HS) methods that are frequently used in the literature. Additionally, it was compared to the first version of the Artificial Atom Algorithm (A^3). The best results were obtained by running all algorithms 20 times.

Parameter settings of the Improved A^3 and other compared algorithms are given in Table 4. The maximum number of iterations for all methods is determined as 1000. In addition, for all algorithms, it is defined the number of population and decision variable as 10.

Table 4. The parameter settings for compared algorithms

Algorithm	Parameter	Value
BCO	Trial limit (L)	60
	Acceleration coefficient upper bound (a)	1
DE	Lower bound of scaling factor (β_{min})	0.2
	Upper bound of scaling factor (β_{max})	0.8
	Crossover probability (P_c)	0.2
GA	Crossover probability (P_c)	0.7
	Extra range factor for crossover (γ)	0.4
	Mutation probability (P_m)	0.3
	Mutation rate (mu)	0.1
HS	Harmony memory consideration rate ($HMCR$)	0.9
	Pitch adjustment rate (PAR)	0.1
	Fret width (bandwidth) (Fw)	0.02*($V_{max}-V_{min}$)
	Fret width damp ratio (Fw_{damp})	0.995
PSO	Inertia weight (w)	1
	Inertia weight damping ratio (w_{damp})	0.99
	Personal learning coefficient (c_1)	1.5
	Global learning coefficient (c_2)	2.0
ACO	Intensification factor (selection pressure) (q)	0.5
	Deviation-distance ratio (ζ - zeta)	1
A^3	Covalent rate (β)	0.6
	Ionic rate (α)	0.4
Improved A^3	Covalent rate (β)	0.6
	Ionic rate (α)	0.4

As seen in Table 5, benchmark problems were used to compare the Improved A^3 with BCO, DE, GA, HS, PSO, ACO and A^3. In addition, in Table 5, the Improved A^3 and the other meta-heuristic

methods were presented comparatively in terms of mean cost, best cost and average run-time when the algorithm ran 20 times.

Table 5. The comparative results of various heuristic methods with the Improved A^3

Algorithm	Max Iter=1000 Population=10 Variable=10	Sphere	Rotated Hyper-Ellipsoid	Parallel Hyper-Ellipsoid	Rastrigin (Rotated R.)	Griewank	Quartic
BCO	**Mean Cost**	0.1005	185.214	0.3882	39.5201	1.3720	0.0785
	Best Cost	5.921e-02	90.6409	2.417e-01	25.626	1.1154	4.176e-02
	Run Time (s)	0.5206	0.5193	0.4997	0.5166	0.5084	0.4933
DE	**Mean Cost**	2.759e-07	6.247e-04	9.530e-04	1.3393	0.0183	0.0071
	Best Cost	7.232e-22	2.725e-16	4.450e-31	9.977e-01	1.370e-03	3.181e-03
	Run Time (s)	0.2661	0.2735	0.2763	0.2756	0.2600	0.2857
GA	**Mean Cost**	6.409e-05	0.2205	7.230e-04	0.0247	0.1105	0.0122
	Best Cost	5.305e-05	2.050e-2	8.699e-05	1.667e-02	8.944e-02	3.675e-03
	Run Time (s)	0.4736	0.5076	0.4544	0.5006	0.4952	0.5083
HS	**Mean Cost**	1.375e-07	2.781e-04	6.371e-07	0.0755	0.0604	0.0189
	Best Cost	9.563e-08	1.326e-04	4.871e-07	1.553e-05	4.689e-02	6.411e-03
	Run Time (s)	0.4779	0.4724	0.4794	0.4656	0.4884	0.4614
PSO	**Mean Cost**	3.488e-09	9.892e-07	1.809e-09	16.2178	0.1318	0.0084
	Best Cost	2.313e-14	3.924e-13	2.844e-11	7.9597	2.701e-02	2.327e-03
	Run Time (s)	0.2925	0.2919	0.2762	0.2926	0.2749	0.2557
ACO	**Mean Cost**	1.735e-43	3.281e-40	1.194e-41	16.3966	0.2978	0.0078
	Best Cost	4.194e-44	7.173e-41	2.300e-44	6.0006	5.974e-02	2.385e-03
	Run Time (s)	0.4687	0.4051	0.3943	0.4207	0.4199	0.4827
A^3	**Mean Cost**	4.570e-04	0.3677	0.0028	0.0655	0.9393	3.763e-10
	Best Cost	1.429e-04	1.5361e-04	1.2899e-03	1.870e-03	7.532e-03	1.427e-12
	Run Time (s)	0.0131	0.0130	0.0130	0.0154	0.0193	0.0132

Improved A^3	**Mean Cost**	**8.488e-295**	2.7445e-38	**2.178e-272**	**0.0**	**0.0129**	**0.0**
	Best Cost	**9.561e-314**	**4.5743e-47**	**0.0**	**0.0**	**0.0**	**0.0**
	Run Time (s)	**0.3128**	**0.3152**	**0.3196**	**0.3747**	**0.3213**	**0.3213**

Table 5. Continue

Table 5. reveals that the Improved A^3 is the most successful method, reaching the optimum cost for the problems other than Sphere and Rotated Hyper-Ellipsoid, and obtaining the best results compared to the other seven meta-heuristic methods. In addition, it is seen that the proposed method reached the optimum results for most benchmark problems in reasonable time. Furthermore, it is observed that the Improved A^3 is only slower than PSO, DE and the first version of A^3, in terms of run-time, and faster than other methods compared. This small loss in terms of run-time provided a great advantage in terms of cost. The results reveal that the Improved A^3 is a very useful method for the optimization test functions.

In addition to previous comparisons, Table 6 compares the success of Improved A^3 in test functions with a recent study in the literature. For this purpose, 7 test functions that are widely used and applicable to the Improved A^3 method have been determined. Accordingly, the results obtained for the Improved A^3 and the variants of Krill Held algorithm based on Van der Corput sequence (VcKH), Faure sequence (FaKH), and Sobol sequence (ScKH) are given in Table 6 comparatively. For this study, the number of parameters was determined as 10 or 30 (10 for Michalewicz, 30 for others), the population size was 50, and the maximum number of iterations was 300 [32].

Table 6. Comparative results of the Improved A^3 method with the variants of the Krill Herd Algorithm

Function	**KH**		**VcKH**		**FaKH**		**ScKH**		**Improved A^3**	
	Best	Mean	Best	Mean	Best	Mean	Best	Mean	Best	Mean
Sphere	1.74E-02	2.55E-02	1.56E-02	2.34E-02	1.31E-02	2.48E-02	1.63E-02	2.66E-02	**4.2448 E-14**	**4.1396 E-12**
Rastrigin	5.97E +00	1.24E +01	3.46E-04	7.39E-04	3.16E-06	2.16E-05	3.52E-04	9.76E-04	**0.0**	**0.0**
Griewank	4.51E-02	8.21E-02	2.52E-02	6.00E-02	4.00E-02	5.89E-02	4.91E-02	8.52E-02	**1.0436 E-14**	**1.05E-02**
Quartic	3.59E-02	8.49E-02	2.10E-03	9.14E-03	2.02E-03	1.46E-02	3.73E-04	1.79E-02	**0.0**	**0.0**
Rosenbrock	2.12E +01	7.49E +01	5.30E-05	1.53E-04	6.07E +00	1.19E +01	2.61E +01	3.07E +01	1.97E-01	3.05E-01
Schwefel	-7.74E +03	-5.48E +03	-1.26E +04	-1.26E +04	-1.26E +04	-1.26E +04	-7.30E +03	-5.21E +03	**-8.22E-01**	**-3.21E-02**
Michalewicz	-2.41E +01	-2.12E +01	-2.41E +01	-2.02E +01	-2.36E +01	-2.08E +01	-2.40E +01	-2.07E +01	**-1.09E +00**	**-3.04E +00**

The results in Table 6 show that the VcKH method achieves better results for the Rosenbrock test function. In contrast, Table 6 reveals that the Improved A^3 method is the most successful method in terms of the best and average results for the other 6 test functions. In addition, the fact that it reaches the global optimum value for Rastrigin and Quartic test functions even in a small iteration, such as 300 iterations, shows the superiority of the Improved A^3 method. Furthermore, the Improved A^3 method has approached the global optimum for Sphere and Griewank test functions compared to other methods.

In Table 7, the Improved A^3 method is presented in comparison with four methods based on the Modified Levy Flight Distribution Algorithm (LFD) [33]. The Improved A^3 method was tested with the same parameter values using 7 test functions preferred in the relevant study in the literature. In this study, the number of parameters was determined as 30, the population size was 50, and the maximum number of iterations was 1000. The results were compared in terms of best and mean values [33].

Table 7. Comparative results of the Improved A^3 method with the variants of the Modified Levy Flight Distribution Algorithm

Function	**Basic LFD**		**LFDNM-S1**		**LFDNM-S2**		**LFDNM-S3**		**Improved A^3**	
	Best	Mean	Best	Mean	Best	Mean	Best	Mean	Best	Mean
Sphere	8.0419 E−08	1.5220 E−07	1.9756 E−08	3.6282 E−08	1.5254 E−09	1.1919 E−08	1.7217 E−44	1.4697 E−42	**1.0325 E-50**	**4.3990 E-48**
Schwefel 2.22	1.8925 E−04	3.0625 E−04	9.9580 E−05	1.3472 E−04	2.2527 E−05	3.7902 E−05	4.7253 E−29	2.0705 E−27	**3.3940 E-85**	**6.6181 E-84**
Rosenbrock	27.7517	27.8977	26.5653	27.0939	23.4629	23.6612	0.2515	0.4302	**0.1852**	**0.3365**
Step	0.5739	1.1480	0.1592	0.5076	6.8912 E−07	2.3949 E−06	5.6266 E−08	1.7721 E−07	4.0394 E-05	7.8422 E-05
Schwefel 2.26	−4.824E+03	−4.3960E+03	−8.3413E+03	−7.8148E+03	−8.9003E+03	−7.6089E+03	−9.8013E+03	−8.2171E+03	−7.541 E+03	−7.541 E+03
Rastrigin	1.5259 E−05	2.8745	2.9721 E−06	0.4463	2.0053 E−06	0.0665	1.2108 E−11	2.2578 E−11	**0.0**	**0.0**
Griewank	1.8918 E−07	3.7917 E−07	4.3454 E−08	1.0317 E−07	1.4857 E−08	4.5251 E−08	4.1078 E−14	5.8538 E−14	**9.9920 E-16**	**2.2204 E-17**

Table 7 shows that the Improved A^3 method has reached the global optimum value for Rastrigin function. In addition, the Improved A^3 method obtained better results than the other compared methods for Sphere, Schwefel 2.22, Rosenbrock, Rastrigin and Griewank functions in terms of best and mean values. Only for Step and Schwefel 2.26 functions, LDF variants seem to get better results than Improved A^3. Among these, it was seen that the results of LDF variants and Improved A^3 for Schwefel 2.26 function were quite close.

In addition to these comparisons, in the study of Bingöl and Yıldırım (2021) using the Salp Swarm Algorithm, it is seen that although they used the low parameter number (number of parameter was determined as 2), they could not obtain better results than the Improved A^3 method with 30 parameters [34].

5. Conclusions

A^3 is an intuitive algorithm that contributes to the solution of optimization problems [15,25]. With the ionic bond function of A^3, a part of the atom set is renewed using the large data set in the search space. This case may lead to a deviation from the solution. In this article, it is aimed that overcomes this problem. For this reason, the ionic bond operator of A^3 has been improved by benefiting an operator of Shuffled Frog Leaping Algorithm.

When the ionic bond operator of A^3 was improved, it was inspired by the operator, which was used to add a new frog into the memeplex in the Shuffled Frog Leaping Algorithm. Thus, the ionic bond operator was carried out by narrowing the search space between the best and worst results in atom set. Consequently, it was expected to provide a better optimization with Improved A^3.

The proposed method was first tested comparatively on 6 benchmark problems. It was compared to the 7 different meta-heuristic algorithm. For all these problems, Improved A^3 produced more successful results than A^3 and the other heuristics. Furthermore, Improved A^3 achieved the optimum results for these 6 problems except Rotated Hyper-Ellipsoid. In addition, it was seen to be advantageous in terms of the number of iterations over A^3 and many other meta-heuristic algorithms [25]. The improved A^3 method is the only algorithm that reaches best result for the most test functions in the number of iterations determined among the compared algorithms. In terms of run-time, it is faster than all comparable methods except for DE, PSO and the first version of A^3. Considering the performance-runtime balance, Improved A^3 is clearly superior to all other methods. In addition to these applications, two comparison were made with the variants of the Krill Held Algorithm [32] and Levy Flight Distribution Algorithm [33] in recent studies in the literature for 7 selected test functions. In the first of these comparison, the Improved A^3 method achieved the best results for all test functions except the Rosenbrock function. It has also proven to be far superior to all of the Kh, VcKH, FaKH and ScKH methods, reaching the global best for the Rastrigin and Quartic methods. In the second of these comparison, the Improved A^3 method achieved the best results for all test functions except the Step and Schwefel 2.22 test functions. In addition, the proposed method has achieved great success by achieving the global best result for the Rastrigin function. The comparison results clearly show the superiority of the Improved A^3 method, both according to the current studies in the literature and according to the results obtained with the 7 most known meta-heuristic algorithms.

With the improvement in A^3, the successful position of the algorithm among meta-heuristic algorithms is further strengthened. In future studies, it is possible to solve engineering design problems or discrete problems using Improved A^3 and evaluate whether the algorithm is also successful for these problems.

References

[1] Yıldırım AE. Yapay atom algoritması ve ayrık problemlere uygulanması (in Turkish). PhD. Thesis, Institute of Natural and Applied Science, Malatya: Inonu University, Turkey; 2018.

[2] Pérez J, Valdez F, Castillo O. Bat algorithm comparison with genetic algorithm using benchmark functions. In: Castillo O, Melin P, Pedrycz W, Kacprzyk J. (eds) Recent Advances on Hybrid Approaches for Designing Intelligent Systems. Studies in Computational Intelligence, 547, Springer, Cham; 2014

[3] Onwubolu GC, Clerc M. Optimal path for automated drilling operations by a new heuristic approach using particle swarm optimization. Int J Prod Res 2004; 42(3), 473–491.

[4] Karcı A. Theory of saplings growing up algorithm. In: Beliczynski B, Dzielinski A, Iwanowski M, Ribeiro B. (Ed.), Adaptive and Natural Computing Algorithms. Lecture Notes in Computer Science, 4431, 450-460, Springer, Berlin, Heidelberg; 2007.

[5] Wong LP, Low MYH, Chong CS. A bee colony optimization algorithm for traveling salesman problem. Proceedings of Second Asia International Conference on Modelling & Simulation (AMS), Kuala Lumpur, Malaysia; 2008.
[6] Dorigo M, Stutzle T. Ant colony optimization. MIT Press, Cambridge, MA; 2004.
[7] Karcı A. Differential evolution algorithm and its variants. Anatolian Journal of Computer Sciences 2017; 2(1), 10-14.
[8] Kızıloluk S, Alataş B. Automatic mining of numerical classification rules with parliamentary optimization algorithm. Advances in Electrical and Computer Engineering 2015; 15(4), 17-24.
[9] Alataş B. Uniform big bang - chaotic big crunch optimization. Communications in Nonlinear Science and Numerical Simulation 2011; 16, 3696-3703.
[10] Alataş B, Özer, AB. Mining of generalized interesting classification rules with artificial chemical reaction optimization algorithm. Journal of the Faculty of Engineering and Architecture of Gazi University 2017; 32, 101-118.
[11] Canayaz M, Karcı A. Cricket behaviour-based evolutionary computation technique in solving engineering optimization problems. Journal of Applied Intelligence 2016; 44, 362-376.
[12] Yang XS, Gandomi AH. Bat algorithm: a novel approach for global engineering optimization. Journal of Engineering Computations 2012; 29, 464-483.
[13] Lee KS, Geem ZW. A new structural optimization method based on the harmony search algorithm. Journal of Computers and Structures 2004; 82(9-10), 781-798.
[14] Eusuff M, Lansey K, Pasha F. Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. Journal of Engineering Optimization 2006; 38, 129-154.
[15] Yıldırım AE, Karcı A. Applications of artificial atom algorithm to small-scale traveling salesman problems. Soft Computing 2018; 22(22), 7619-7631.
[16] Sastry K, Goldberg D, Kendall G. Genetic Algorithms. In: Burke EK, Kendall G. (eds) Search Methodologies. Springer, Boston, MA; 2005.
[17] Jain NK, Nangia U, Jain J. A Review of Particle Swarm Optimization. J. Inst. Eng. India Ser. B 2018; 99, 407–411.
[18] Demir M, Karcı A, Özdemir M. Fidan gelişim algoritması yardımı ile DNA motiflerinin keşfi (in Turkish). Çankaya University Journal of Science and Engineering 2011; 8, 51-62.
[19] Mallipeddi R, Suganthana PN, Panb QK, Tasgetiren MF. Differential evolution algorithm with ensemble of parameters and mutation strategies. Applied Soft Computing 2011; 11(2), 1679-1696.
[20] Altunbey F, Alataş B. Overlapping community detection in social networks using parliamentary optimization algorithm. International Journal of Computer Networks and Applications 2015; 2, 12-19.
[21] Erol OK, Eksin I. A new optimization method: Big Bang–Big Crunch. Advances in Engineering Software 2006; 37, 106-111.
[22] Karaboğa D. Yapay Zekâ Optimizasyon Algoritmaları (in Turkish), Nobel Publisher, Istanbul, Turkey; 2017.
[23] Canayaz M. Cırcır böceği algoritması: Yeni bir meta-sezgisel yaklaşım ve uygulamaları (in Turkish). PhD. Thesis, Institute of Natural and Applied Science, Malatya: Inonu University, Turkey; 2015.
[24] Yetkin, M. Metasezgisel algoritmaların Jeodezi'de kullanımı (in Turkish). Journal of Geomatic Research 2016; 1(1), 8-13.
[25] Altun S, Varjovi MH, Karcı, A. Performance comparison of different optimization methods under the same conditions. Proceedings of 3nd International Conference on Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey; 2018.
[26] Yıldırım AE, Karcı A. Solutions of travelling salesman problem using genetic algorithm and atom algorithm. Proceedings of 2nd International Eurasian Conference on Mathematical Sciences and Applications (IECMSA), Sarajevo, Bosnia and Herzegovina; 2013.
[27] Canayaz M, Demir M. Veri kümelemede yapay atom algoritması ve cırcır böceği algoritmasının karşılaştırılmalı analizi. Proceedings of 4th International Symposium on Innovative Technologies in Engineering and Science (ISITES), Antalya, Turkey; 2016.

[28] Yıldırım AE, Karcı A. Group elevator control optimization using artificial atom algorithm. Proceedings of 2nd International Conference on Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey; 2017.
[29] Yıldırım AE, Karcı A. Application of traveling salesman problem for 81 provinces in Turkey using artificial atom algorithm. Proceedings of 7th International Conference on Advanced Technology & Science (ICAT), Antalya, Turkey; 2018.
[30] Yıldırım AE, Karcı A. Application of three bar truss problem among engineering design optimization problems using artificial atom algorithm. Proceedings of 3nd International Conference on Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey; 2018.
[31] Hussain K, Salleh MNM, Cheng S, Naseem R. Common benchmark functions for metaheuristic evaluation: A review. International Journal of Informatics Visualization 2017; 1, 218-223.
[32] Agushaka OJ, Ezugwu AE-S. Influence of initializing krill herd algorithm with low-discrepancy sequences. IEEE Access 2020; 8, 210891.
[33] Dündar A, İzci D, Ekinci S, Eker E. A novel modified lévy flight distribution algorithm based on nelder-mead method for function optimization. DUJE 2021; 12(3), 487-497.
[34] Bingöl H, Yıldırım M. Global optimizasyon için sürü tabanlı bir yaklaşım salp sürü algoritması. Fırat Üniversitesi Fen Bilimleri Dergisi 2021; 33, 51-59.