Volume: 14 Issue: 06 Pages: 5696 - 5704 (2023) ISSN: 0975-0290

# CIDO: Chaotically Initialized Dandelion Optimization for Global Optimization

**Sinem Akyol**
Department of Software Engineering, Firat University, Elazig-23000
Email: sakyol@firat.com
**Muhammed Yildirim**
Department of Computer Engineering, Malatya TurgutOzal University, Malatya-44000
Email:muhammed.yildirim@ozal.edu.tr
**Bilal Alatas**
Department of Software Engineering, Firat University, Elazig-23000
Email: balatas@firat.com

-------------------------------------------------------------------**ABSTRACT**-------------------------------------------------------------
**Metaheuristic algorithms are widely used for problems in many fields such as security, health, engineering. No metaheuristic algorithm can achieve the optimum solution for all optimization problems. For this, new metaheuristic methods are constantly being proposed and existing ones are being developed. Dandelion Optimizer, one of the most recent metaheuristic algorithms, is biology-based. Inspired by the wind-dependent long-distance flight of the ripening seed of the dandelion plant. It consists of three phases: ascending phase, descending phase and landing phase. In this study, the chaos-based version of Chaotically Initialized Dandelion Optimizer is proposed for the first time in order to prevent Dandelion Optimizer from getting stuck in local solutions and to increase its success in global search. In this way, it is aimed to increase global convergence and to prevent sticking to a local solution. While creating the initial population of the algorithm, six different Chaotically Initialized Dandelion Optimizer algorithms were presented using the Circle, Singer, Chebyshev, Gauss/Mouse, Iterative and Logistic chaotic maps. Two unimodal (Sphere and Schwefel 2.22), two multimodal (Schwefel and Rastrigin) and two fixed-dimension multimodal (Foxholes and Kowalik) quality test functions were used to compare the performances of the algorithms. When the experimental results were analyzed, it was seen that the Chaotically Initialized Dandelion Optimizer algorithms gave successful results compared to the classical Dandelion Optimizer.**

Keywords -**Metaheuristic Algorithms, Dandelion Optimizer, Chaos, Global Optimization**
--------------------------------------------------------------------------------------------------------------------------------------------
Date of Submission: October 28, 2022 Date of Acceptance: April 10, 2023
--------------------------------------------------------------------------------------------------------------------------------------------

## I. INTRODUCTION

The aim of optimization is to search for the best between all solution candidates for the interested problem under specific constraints. Although the solution methodologies in classical optimization algorithms mostly depend on the variable types (real, integer, etc.), objective and constraint functions (non-linear, linear, etc.) used in the problem modeling, their effectiveness also depend on the solution method. Also, classical optimization algorithms do not provide applicable general solution strategies for problem formulations that consist of different types of decision variables, constraint functions, and objectives functions. Therefore, metaheuristic optimization algorithms are suggested. These techniques have gained considerable popularity due to their high computational efficiency and simplicity in their transformation. General purposed metaheuristic algorithms can be categorized into eight groups: swarm-based, biology-based, physics-based, chemistry-based, social-based, music-based, mathematics-based, and sports-based [1][2]. Hybridization of these algorithms can also be categorized as a different group [3].
A perennial herb belonging to the Asteraceae family, the dandelion is officially known as Herbataraxaci. These plants have heads that can grow to be more than 20 cm tall, and they resemble inflorescences [4]. The wind transfers their mature seeds to new locations where they will nourish life. The crown hairs, which are crucial for the dissemination of the seeds, delay their landing so that the wind can carry them farther. The most emblematic plant that depends on the wind for seed dispersion is the dandelion. Under the correct circumstances, its seed can travel tens of kilometers in the wind [5]. The two main parameters that influence the dispersal of dandelion seeds are wind velocity and climate. When determining whether a seed is traveling over long or short distances, wind speed is used [6]. Weather affects the ability of dandelions to grow near or far by controlling whether dandelion seeds can fly.
Dandelion seeds go through three stages. In the first stage, the ascent stage, a vortex forms on the dandelion seed and rises under the influence of drag in sunny and windy weather. In contrast, there is no vortex on the seeds when it is raining. In this situation, only local searches are possible. In the second stage, the descending stage, the seeds fall continuously after reaching a certain height. Dandelions eventually reproduce through seed, which at this stage lands randomly under the influence of wind and weather. Dandelion improves its populations by passing its seeds through these three stages to the next generation. Dandelion Optimization, one of the most recent metaheuristic algorithms, has been developed inspired by these three stages [7].
In this study, chaotic version of Dandelion Optimizer is proposed for the first time in order to increase its performance. CIDO algorithms were developed using the initial population chaotic maps. Circle, Singer, Chebyshev, Gauss/Mouse, Iterative and Logistic maps were used as chaotic maps. In the second part, classical DO is introduced

and its pseudo code is given. In the third section, the proposed CIDO is explained. Information is given about the chaotic maps used. In the fourth chapter, performance comparisons were made using 4 unimodal, 4 multimodal and 4 fixed dimension multimodel test functions. In the fifth chapter, the conclusion part, the study was concluded.

## II. DANDELION OPTIMIZER (DO)

Similar to other nature-inspired meta-heuristic algorithms, in the DO algorithm, each dandelion seed is assumed to represent a candidate solution and the initial population is expressed as in Equation 1.

$$population = \begin{bmatrix} x_1^1 & \cdots & x_1^{Dim} \\ \vdots & \ddots & \vdots \\ x_{pop}^1 & \cdots & x_{pop}^{Dim} \end{bmatrix} \quad (1)$$

Here, $pop$ represents the population number and $Dim$ represents the size of the variable. Equation 2 illustrates the formulation of the $i$th candidate solution, $x_i$, which is produced randomly between the given problem's upper bound ($UB$) and lower bound ($LB$).

$$x_i = rand \times (UB - LB) + LB \quad (2)$$

$i$ is an integer between 1 and $pop$ and $rand$ is a random number between 0 and 1.
At the time of initiation, the individual with the best fitness becomes the first elite individual to be considered the most feasible location for the seed of dandelion to develop. Equation 3 displays the mathematical equation of the first selection using the minimum value as an example. Here find specifies two directories with equal values [7].

$$f_{best} = \min(f(X_i))$$
$$x_{elite} = x(find(f_{best} == f(X_i))) \quad (3)$$

### II. I. ASCENT STAGE

Dandelion seeds must ascend to a particular height during the ascension phase before they may separate from their parents. It rises to different heights due to factors such as wind speed and air humidity. At this stage, the air is divided into two states.
**State 1:** On a clear day, it is reasonable to suppose that wind speeds follow the lognormal distribution $lnY \sim N(\mu, \sigma^2)$. The probability that dandelion seeds will wind up in remote places rises under this distribution because the random numbers are more spread along $Y$- axis. Therefore, DO performs the discovery process in this case. Dandelion seeds are randomly dispersed around search space by the wind. A dandelion seed's rate of germination is influenced by the speed of the wind. The farther the seeds are dispersed and the higher the dandelions fly, the stronger the wind. The wind speed constantly modifies the vortices on dandelion seeds to cause them to rise spirally [7]. Equation 4 contains the mathematical expression that describes this circumstance.

$$x_{t+1} = x_t + \alpha \times v_x \times v_y \times lnY(x_s - x_t) \quad (4)$$

The dandelion seed's location in this iteration is shown by the value of $x_t$. In the $t$ iterations of the search space, $x_s$ denotes the spot that was arbitrarily selected. Equation 5 refers to randomly generated positions [7].

$$x_s = rand(1, Dim) \times (UB - LB) + LB \quad (5)$$

$lnY$ represents the lognormal distribution subject to $\mu = 0$ and $\sigma^2 = 1$ and it is mathematically defined as shown in Equation 6. The mathematical formula for $lnY$, which stands for a lognormal distribution subject to $\mu = 0$ and $\sigma^2 = 1$, is given in Equation 6.

$$lnY = \begin{cases} \frac{1}{y\sqrt{2\pi}} \exp\left[-\frac{1}{2\sigma^2}(lny)^2\right] & y \geq 0 \\ 0 & y < 0 \end{cases} \quad (6)$$

In this case, $y$ stands for typical normal distribution $N(0,1)$. $\alpha$ is an adjustable parameter that determines the size of the search step, and Equation 7 provides its mathematical expression.

$$\alpha = rand() \times (\frac{1}{T^2}t^2 - \frac{2}{T}t + 1) \quad (7)$$

The lift component coefficients of a dandelion caused by the separated vortex motion are represented by the variables $v_x$ and $v_y$ in Equation 4. Equation 8 is used to calculate the force on variable dimensions. $\theta$ is a number in the range $[-\pi, \pi]$ [7].

$$r = \frac{1}{\varepsilon^{\theta}}$$
$$v_x = r \times cos\theta \quad (8)$$
$$v_y = r \times sin\theta$$

**State 2:** Dandelion seeds struggle to rise effectively with the wind on wet days due to humidity, air resistance, and other variables. In this instance, the seeds of dandelion gain from their surrounding community. The mathematical definition corresponding to this situation is given in Equation 9.

$$x_{t+1} = x_t \times k \quad (9)$$

$k$ is used to organize a dandelion's local search area. Equation 10 shows the calculation of domains.

$$q = \frac{1}{T^2-2T+1}t^2 - \frac{2}{T^2-2T+1}t + 1 + \frac{1}{T^2-2T+1} \quad (10)$$
$$k = 1 - rand() \times q$$

To ensure the convergence of the population to the ideal individual at the end of the iterations, the parameter $k$ gradually approaches 1 [7]. Equation 11 illustrates the mathematical formulation of dandelion seeds in the rising stage.

$$x_{t+1} = \begin{cases} x_t + \alpha \times v_x \times v_y \times lnY(x_s - x_t) & randn < 1.5 \\ x_t \times k & else \end{cases} \quad (11)$$

$randn$ is a random number following the standard normal distribution.

### II.II. DESCENDING STAGE

The DO algorithm prioritizes exploration at this point. Dandelion seeds typically fall to the ground after ascending a given amount. Brownian motion is utilized in DO to replicate the trajectory of moving dandelions. In the iterative updating process, it is simple for individuals to traverse more search populations since the Brownian motion goes by a normal distribution at all changes. The average position data following the rising stage is utilized to reflect the stability of the dandelion landing. This makes it easier for the population as a whole to grow into vibrant communities.

Equation 12 provides the mathematical expression for this stage.

$$x_{t+1} = x_t - \alpha \times \beta_t \times (x_{mean\_t} - \alpha \times \beta_t \times x_t) \quad (12)$$

This equation, which uses a random number drawn from the norm distribution, describes the Brownian motion. The mathematical expression of $x_{mean\_t}$, which denotes the population's average location in the $i$th iteration, is given in Equation 13 [7].

$$x_{mean\_t} = \frac{1}{pop}\sum_{i=1}^{pop} x_i \quad (13)$$

## II.III. LANDING STAGE

The DO algorithm concentrates on the exploitation steps in this section. The dandelion seed decides where to land at random in the first two stages. It is intended that the method converges to the overall optimal solution as the iterations advance gradually. The approximate position where the dandelion seeds will thrive the easiest is the best option that can be reached in this manner. The expertise of current elites is used by search agents in their local communities in order to closely approximate the global optimum. The population's evolution will eventually lead to the discovery of the global optimal solution. This behavior is expressed by Equation (14).

$$x_{t+1} = x_{elite} + levy(\lambda) \times \alpha \times (x_{elite} - x_t \times \delta) \quad (14)$$

$x_{elite}$ represents the optimal dandelion seed position in the $i$th iteration. The $levy(\lambda)$ calculated using Equation 15 represents the Levy flight [7].

$$levy(\lambda) = s \times \frac{\omega \times \sigma}{|t|^{\frac{1}{\beta}}} \quad (15)$$

Here, $\beta$ is a random number between [0,2], in this study, the value of $\beta$ is taken as 1.5. $s$ is a constant and its value is 0.01. $\omega$ and $t$ are random numbers between [0,1]. The mathematical formula of $\sigma$ is shown in Equation 16.

$$\sigma = \left(\frac{\Gamma(1+\beta) \times \sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}}\right) \quad (16)$$

The value of the constant $\beta$ is used as 1.5. $\delta$ is a linearly increasing value between [0,2] and is computed as shown in Equation 17.

$$\delta = \frac{2t}{T} \quad (17)$$

## II.IV. RUNNING THE DO ALGORITHM

In DO, the optimization process is initiated by the creation of random vector groups in the search space. Dandelion seeds then carry out three phases of iterative optimization: ascending, descending, and landing. Each seed of dandelion was placed in ascending order from top to bottom based on its fitness, using the smallest value as an example. The elite member of the population's next generation is the individual with the lowest fitness, and the ranking population serves as the beginning population for following iteration. This sorting technique is helpful for passing down reliable information. The algorithm sets the maximum number of iterations before completing the optimization process [7]. Figure 1 depicts the flowchart diagram for the suggested DO algorithm.

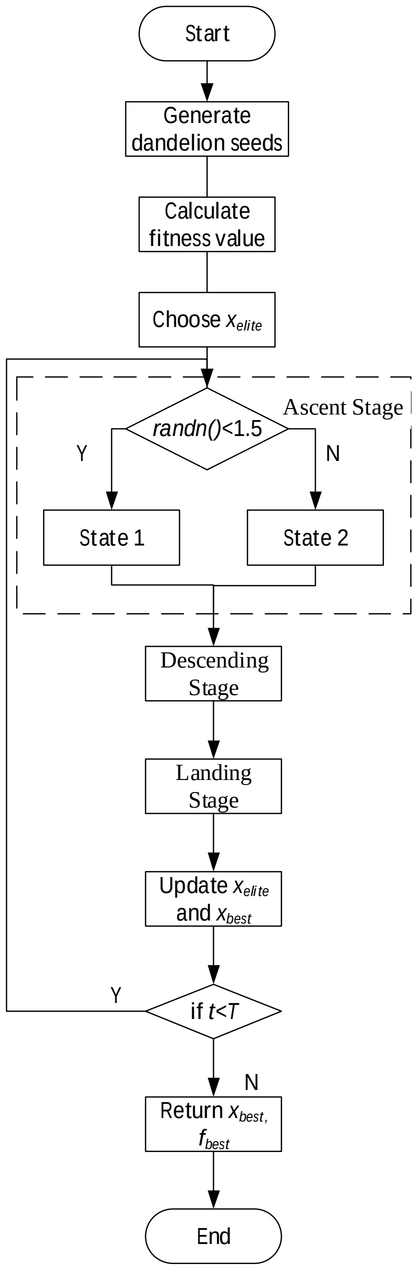


**Fig 1.** Pseudocode of the DO algorithm

## III. CHAOTICALLY INITIALIZED DANDELION OPTIMIZER (CIDO)

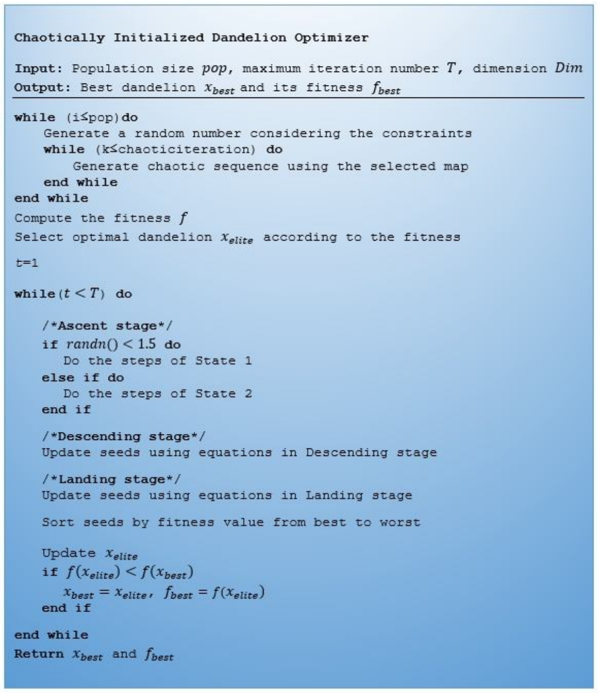
Generating long-term random sequences with good uniformity is essential in numerical analysis, sampling, decision making, and especially metaheuristic optimization. Successful generation of random numbers reduces computation and storage time to obtain the preferred accuracy. Such created sequences may be "random" enough for specific application, but not random enough for other applications [8][9]. Recently, random number sequences have been replaced by chaotic number sequences and give better results in many complex real-world problems. Chaotic number sequences have increased in popularity due to their theoretical spread spectrum properties, unpredictability, and their associated ergodic properties [10].

The initial population is created using random sequences in the classic dandelion optimizer. The dandelion optimizer's randomized parameters may have an impact on how effective the algorithm is. The convergence rate might be lower since it might not be able to cover a worldwide search. The aim of this study is to reveal the effect of the chaotic sequence with different chaotic maps on the success of dandelion optimizer for the interested problems. The initiation step of the dandelion optimizer can affect the success of convergence. Thus, in the initialization step of dandelion optimizer, different chaotic systems are used instead of random number sequences. Therefore, a chaotically initiated dandelion optimizer, called CIDO, is proposed in this study. In this way, the global convergence of dandelion optimizer was tried to be improved and kept away from being trapped in local solutions. Instead of the equation in Equation 5 used to create the initial population in the dandelion optimizer algorithm, chaotic maps are used.

The first rand value is considered as $R^0$ and after this rand value for the determination of initial values of dimensions of search agents Equation 18 is used.

$$R^{t+1} = ChaoticMap_{type}(R^t) \tag{18}$$

generates chaotic numbers according to the first initial random number ($R^0$) for the first iteration. type shows the type of the used chaotic map. Thus, $R^{t+1}$generates chaotic numbers according to the initial values of the selected chaotic map for the initial search agents. The pseudo-code of the proposed method is presented in Figure 2.


```
Chaotically Initialized Dandelion Optimizer

Input: Population size $pop$, maximum iteration number $T$, dimension $Dim$
Output: Best dandelion $x_{best}$ and its fitness $f_{best}$
------------------------------------------------------------------------
while (i≤pop)do
    Generate a random number considering the constraints
    while (k≤chaoticiteration) do
        Generate chaotic sequence using the selected map
    end while
end while
Compute the fitness $f$
Select optimal dandelion $x_{elite}$ according to the fitness

t=1

while($t < T$) do

    /*Ascent stage*/
    if $randn() < 1.5$ do
       Do the steps of State 1
    else if do
       Do the steps of State 2
    end if

    /*Descending stage*/
    Update seeds using equations in Descending stage

    /*Landing stage*/
    Update seeds using equations in Landing stage

    Sort seeds by fitness value from best to worst

    Update $x_{elite}$
    if $f(x_{elite}) < f(x_{best})$
       $x_{best} = x_{elite}$, $f_{best} = f(x_{elite})$
    end if

end while
Return $x_{best}$ and $f_{best}$
```

**Fig 2.** The pseudo-code of the CIDO algorithm

The chaotic maps and equations used in the study are given in Table 1.

**Table 1.** Chaotic maps and equations used in the study

| Chaotic Map | Formula |
|---|---|
| **Circle map** | $X_{n+1} = X_n + b - \left(\frac{a}{2\pi}\right)\sin(2\pi X_n)\, mod$ |
| **Singer map** | $X_{n+1} = u \times (7.86 \times X_n - 23.31 \times (X_n^2) + 28.75 \times (X_n^3) - 13.302875 \times X_n^4)$ |
| **Chebyshev map** | $X_{n+1} = \cos(n \times arccos(X_n))$ |
| **Gauss/mouse map** | $X_{n+1} = \exp(-\alpha X_n^2) + \beta_1$ |
| **Iterative map** | $X_{n+1} = \sin\left(\frac{a\pi}{X_n}\right), a = 0.7$ |
| **Logistic map** | $X_{n+1} = r x_n (1 - X_n)$ |

Figure 3 shows the chaotic numbers generated from the chaotic maps used in the study.

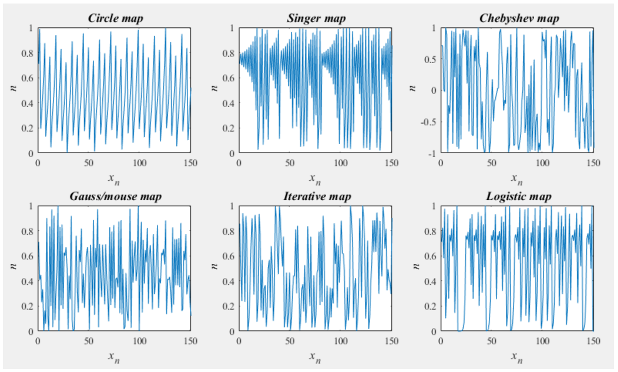


**Fig3.** Chaotic numbers generated from the chaotic maps used

## IV. EXPERIMENTAL RESULTS

To measure the performance of the proposed CIDO algorithm, four unimodal (Sphere, Rosenbrock, Schwefel 2.22, and Step), four multimodal (Schwefel, Ackley, Rastrigin, and Griewank) and four fixed-dimension multimodal (Foxholes, Kowalik Six-hump camel back and Branin), a total of twelve benchmark test functions were used. It is typical to use specific standard benchmark functions even in the most recent examples of these tests with the implicit (but unproven) assumption that the difficulty of these benchmark functions generally equates to that of real-world applications. Even some of these benchmark functions are marketed as being exceptionally difficult.These benchmark test functions and their explanations used are presented in Table 2.

**Table 2.** Benchmark test functions

| | Function | Definition | Range | Min | Problem Dimension |
|---|---|---|---|---|---|
| Unimodal | Sphere | $f_1(x) = \sum_{i=1}^{n} x_i^2$ | $-100 \le x_i \le 100$ | 0 | 30 |
| | Schwefel 2.22 | $f_2(x) = \sum_{i=0}^{N} \lvert x_i \rvert + \prod_{i=0}^{N} \lvert x_i \rvert$ | $-10 \le x_i \le 10$ | 0 | 30 |
| | Rosenbrock | $f_3(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x^2)^2 + (x_i - 1)^2)$ | $-30 \le x_i \le 30$ | 0 | 30 |
| | Step | $f_4(x) = \sum_{i=1}^{n} (\lvert x_i + 0.5 \rvert)^2$ | $-100 \le x_i \le 100$ | 0 | 30 |
| Multimodal | Schwefel | $f_5(x) = n \times 418.9829 + \sum_{i=1}^{n} -x_i \sin(\sqrt{x_i})$ | $-500 \le x_i \le 500$ | $-418.9829 \times n$ | 30 |
| | Rastrigin | $f_6(x) = \sum_{i=1}^{N} [x_i^2 - 10\cos(2\pi x_i) + 10]$ | $-5.12 \le x_i \le 5.12$ | 0 | 30 |
| | Ackley | $f_7(x) = -20exp\left(-0.2\sqrt{\frac{1}{2}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} cos2\pi xi\right) + 20 + e$ | $-32 \le x_i \le 32$ | 0 | 30 |
| | Griewank | $f_8(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ | $-600 \le x_i \le 600$ | 0 | 30 |
| Fixed-dimension multimodal | Foxholes | $f_9(x) = \left(\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{i,j})^6}\right)^{-1}$ | $-65.536 \le x_i \le 65.536$ | 1 | 2 |
| | Kowalik | $f_{10}(x) = \sum_{i=1}^{11} \left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$ | $-5 \le x_i \le 5$ | 0.00030 | 4 |
| | Six-hump camel back | $f_{11}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$ | $-5 \le x_i \le 5$ | -1.03163 | 2 |
| | Branin | $f_{12}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)cosx_1 + 10$ | $[-5.10] \times 0.15$ | 0.398 | 2 |

In this study, the population number was taken as 30 for all quality test functions. The ending condition of the algorithm was determined as the number of iterations reaching 500. For each benchmark test function and chaotic map, the algorithm is run 20 times and the best, worst, average, and standard deviation of the results are given. The results obtained for the Sphere, Schwefel 2.22, Rosenbrock and Step unimodal benchmark test functions are shown in Table 3.

**Table 3.** Results obtained for Unimodal benchmark test functions

| | | Classical DO | Circle CIDO | Singer CIDO | Chebyshev CIDO | Gauss CIDO | Iterative CIDO | Logistic CIDO |
|---|---|---|---|---|---|---|---|---|
| Sphere | Best | 2.005E-06 | 1.124E-06 | 1.804E-06 | 1.811E-06 | 1.446E-06 | 1.736E-06 | 3.066E-06 |
| | Average | 1.253E-05 | 1.001E-05 | 1.135E-05 | 8.810E-06 | 1.017E-05 | 1.019E-05 | 1.037E-05 |
| | Worst | 5.473E-05 | 2.902E-05 | 4.523E-05 | 3.007E-05 | 2.256E-05 | 3.073E-05 | 2.007E-05 |
| | Standard Deviation | 1.142E-05 | 8.179E-06 | 1.161E-05 | 6.736E-06 | 5.772E-06 | 7.151E-06 | 5.431E-06 |
| | | | | | | | | |
| Schwefel 2.22 | Best | 8.463E-04 | 8.381E-04 | 6.729E-04 | 5.818E-04 | 4.299E-04 | 6.542E-04 | 6.373E-04 |
| | Average | 1.716E-03 | 1.711E-03 | 1.614E-03 | 1.599E-03 | 1.471E-03 | 1.625E-03 | 1.350E-03 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Worst | 2.926E-03 | 3.151E-03 | 2.831E-03 | 3.144E-03 | 2.613E-03 | 3.318E-03 | 2.068E-03 |
| | Standard Deviation | 6.377E-04 | 6.028E-04 | 6.320E-04 | 5.770E-04 | 6.463E-04 | 7.883E-04 | 3.690E-04 |
| | | | | | | | | |
| **Rosenbrock** | Best | 2.537E+01 | 2.501E+01 | 2.538E+01 | 2.561E+01 | 2.420E+01 | 2.516E+01 | 2.290E+01 |
| | Average | 3.782E+01 | 3.492E+01 | 2.690E+01 | 3.472E+01 | 3.188E+01 | 2.725E+01 | 3.078E+01 |
| | Worst | 1.784E+02 | 1.204E+02 | 2.921E+01 | 1.211E+02 | 1.230E+02 | 2.935E+01 | 9.779E+01 |
| | Standard Deviation | 3.596E+01 | 2.480E+01 | 7.457E-01 | 2.398E+01 | 2.148E+01 | 7.477E-01 | 1.584E+01 |
| | | | | | | | | |
| **Step** | Best | 6.834E-06 | 5.249E-06 | 6.925E-06 | 5.048E-06 | 7.565E-06 | 5.272E-06 | 6.344E-06 |
| | Average | 1.858E-05 | 1.593E-05 | 1.733E-05 | 1.622E-05 | 1.676E-05 | 1.742E-05 | 1.833E-05 |
| | Worst | 3.197E-05 | 2.824E-05 | 2.966E-05 | 2.652E-05 | 2.752E-05 | 3.849E-05 | 3.114E-05 |
| | Standard Deviation | 8.373E-06 | 5.381E-06 | 7.290E-06 | 5.130E-06 | 6.385E-06 | 8.150E-06 | 7.320E-06 |

When Table 3 is examined, the best results are obtained when the initial population for the Sphere test function is created using the Circle chaotic map. Looking at the average, the CIDO algorithm, which was created using the initial population Chebyshev chaotic map, gave the best result. While the worst result is obtained from the classical DO, it is seen that the best value is obtained from the Logistic map in terms of standard deviation. Looking at Schwefel 2.22, it is seen that the best result is obtained from the CIDO algorithm, whose initial function is created using the Gaussian chaotic map. The average best result and the best standard deviation were taken from Logistic CIDO. In the Rosenbrock test function, it is seen that the best result is obtained from the Logistic map. When the average and standard deviation are examined, it is seen that the best result is given by Singer map, and the worst result in the study is obtained from classical DO. When the results for the step test function are examined, it is observed that the best results and standard deviations are obtained from Chebyshev CIDO, and the best results are obtained from Circle CIDO in the average.A graphical representation of the results obtained from the unimodal benchmark test functions is given in Figure 4.

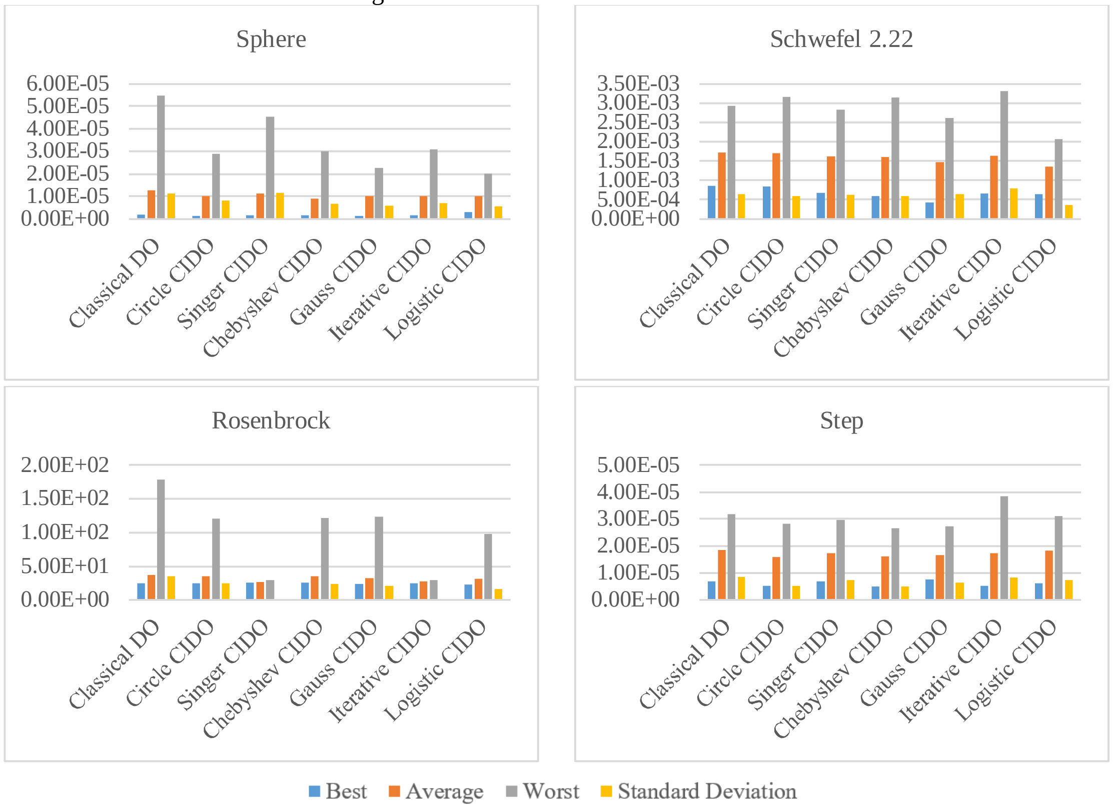


**Fig 4.**Graphical representation of results from unimodal benchmark test functions

The results obtained when Schwefel, Rastrigin, Ackley, and Griewank are run for the multimodal quality test functions are given in Table 4. Accordingly, the best result for the Schwefel function and the best result on average were obtained from the Singer chaotic map. In the standard deviation, the best result was obtained from the CIDO, whose initial population was created using the Chebyshev chaotic map. Looking at the results for the Rastrigin function, while the Logistic map gave the best result, Chebyshev gave the best average result and the best standard deviation value. When the results obtained for the Ackley test function are examined, the Logistic map gives the best result, while the Circle map gives the best average result and standard deviation. Finally, for the Griewank test function, Circle CIDO gave the best result, Logistic CIDO gave the best average result, and Singer CIDO gave the best standard deviation value. The worst result was given by the classical DO algorithm.A graphical representation of the results obtained from the multimodal benchmark test functions is given in Figure 5.

**Table 4.** Results obtained for Multimodal benchmark test functions

| | | **Classical DO** | **Circle CIDO** | **Singer CIDO** | **Chebyshev CIDO** | **Gauss CIDO** | **Iterative CIDO** | **Logistic CIDO** |
|---|---|---|---|---|---|---|---|---|
| **Schwefel** | Best | -7.961E+03 | -8.827E+03 | -9.623E+03 | -8.194E+03 | -8.078E+03 | -9.008E+03 | -8.909E+03 |
| | Average | -7.448E+03 | -7.536E+03 | -8.900E+03 | -7.838E+03 | -7.488E+03 | -7.884E+03 | -8.217E+03 |
| | Worst | -6.351E+03 | -6.302E+03 | -8.233E+03 | -7.393E+03 | -6.782E+03 | -6.558E+03 | -6.984E+03 |
| | Standard Deviation | 4.160E+02 | 6.948E+02 | 4.167E+02 | 2.367E+02 | 4.502E+02 | 6.088E+02 | 5.109E+02 |
| | | | | | | | | |
| **Rastrigin** | Best | 9.063E+00 | 1.994E+00 | 7.021E+00 | 3.001E+00 | 1.010E+00 | 2.997E+00 | 9.965E-01 |
| | Average | 2.678E+01 | 1.704E+01 | 2.360E+01 | 1.689E+01 | 1.923E+01 | 2.643E+01 | 2.704E+01 |
| | Worst | 8.613E+01 | 6.446E+01 | 5.425E+01 | 3.421E+01 | 4.296E+01 | 7.706E+01 | 6.500E+01 |
| | Standard Deviation | 2.000E+01 | 1.446E+01 | 1.476E+01 | 1.037E+01 | 1.220E+01 | 1.891E+01 | 2.005E+01 |
| | | | | | | | | |
| **Ackley** | Best | 5.122E-04 | 4.370E-04 | 3.894E-04 | 1.990E+01 | 3.120E-04 | 4.365E-04 | 2.876E-04 |
| | Average | 8.230E-04 | 6.747E-04 | 7.862E-04 | 1.994E+01 | 7.243E-04 | 7.599E-04 | 7.587E-04 |
| | Worst | 1.103E-03 | 9.259E-04 | 1.148E-03 | 1.997E+01 | 2.134E-03 | 1.219E-03 | 1.335E-03 |
| | Standard Deviation | 1.589E-04 | 1.572E-04 | 2.312E-04 | 1.635E-02 | 3.764E-04 | 2.143E-04 | 2.587E-04 |
| | | | | | | | | |
| **Griewank** | Best | 1.761E-05 | 3.883E-06 | 1.375E-05 | 2.566E-05 | 1.250E-05 | 9.634E-06 | 1.466E-05 |
| | Average | 1.756E-02 | 1.192E-02 | 1.323E-02 | 1.084E-02 | 1.390E-02 | 1.249E-02 | 1.068E-02 |
| | Worst | 7.290E-02 | 4.328E-02 | 3.464E-02 | 6.391E-02 | 3.986E-02 | 6.432E-02 | 4.734E-02 |
| | Standard Deviation | 2.278E-02 | 1.414E-02 | 1.105E-02 | 1.543E-02 | 1.328E-02 | 1.575E-02 | 1.369E-02 |

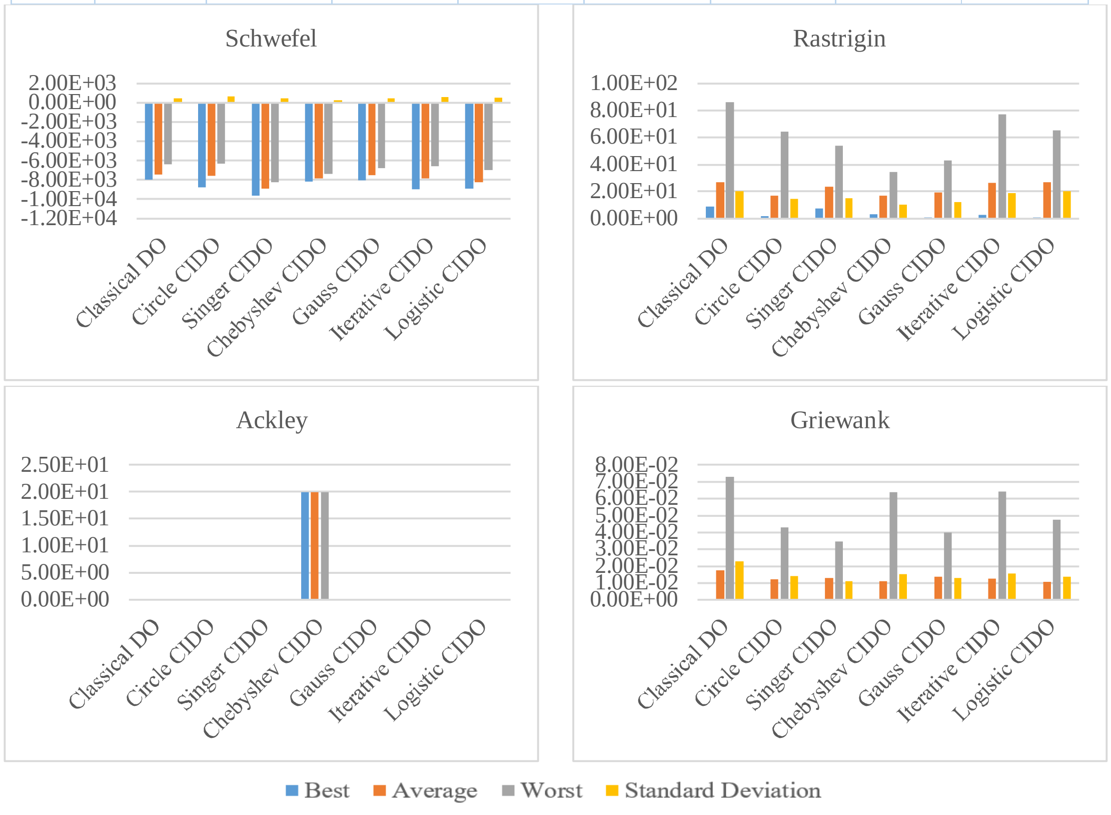


**Fig 5.**Graphical representation of results from multimodal benchmark test functions

The results obtained for Foxholes, Kowalik, Six-hump and Branin fixed-dimension multimodal benchmark test functions are given in Table 5. Accordingly, all algorithms gave similar results for the Foxholes quality test function. Looking at the average, it is seen that the best results are obtained from the CIDO algorithms that create the initial population with Circle, Singer and Iterative chaotic maps. For the Kowalik quality test function, while the algorithms give close values for the best result, the best value for the average best result is obtained from the CIDO algorithm using the Singer chaotic map. When the results obtained for the Six-Hump Camel Back quality test function are examined, all the results seem to be close to each other. However, it is seen that the best results are obtained from Singer and Iterative in the mean, and the best value is taken from the Iterative chaotic map when the standard deviation is examined. For the Branin test function, it is seen that Singer, Gauss and Iterative CIDO give the best results. It is seen that the best average result and standard deviation are obtained from the Chebyshev CIDO algorithm.

**Table 5.** Results obtained for Fixed-dimension Multimodal benchmark test functions

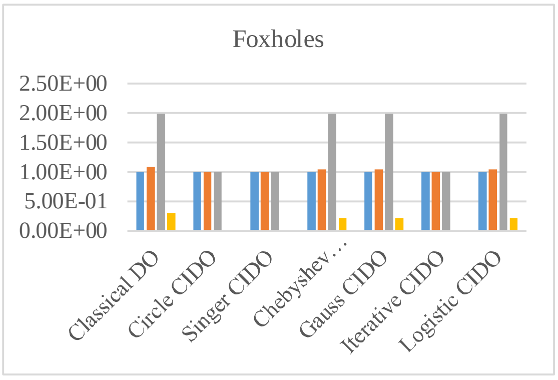
| | | **Classical DO** | **Circle CIDO** | **Singer CIDO** | **Chebyshev CIDO** | **Gauss CIDO** | **Iterative CIDO** | **Logistic CIDO** |
|---|---|---|---|---|---|---|---|---|
| **Foxholes** | Best | 9.980E-01 | 9.980E-01 | 9.980E-01 | 9.980E-01 | 9.980E-01 | 9.980E-01 | 9.980E-01 |
| | Average | 1.097E+00 | 9.980E-01 | 9.980E-01 | 1.048E+00 | 1.048E+00 | 9.980E-01 | 1.048E+00 |
| | Worst | 1.992E+00 | 9.980E-01 | 9.980E-01 | 1.992E+00 | 1.992E+00 | 9.980E-01 | 1.992E+00 |
| | Standard Deviation | 3.060E-01 | 1.669E-15 | 1.328E-15 | 2.223E-01 | 2.223E-01 | 7.562E-15 | 2.223E-01 |
| | | | | | | | | |
| **Kowalik** | Best | 3.075E-04 | 3.075E-04 | 3.075E-04 | 3.075E-04 | 3.075E-04 | 3.075E-04 | 3.075E-04 |
| | Average | 5.715E-04 | 5.331E-04 | 4.047E-04 | 5.080E-04 | 5.464E-04 | 5.076E-04 | 5.547E-04 |
| | Worst | 1.594E-03 | 1.223E-03 | 6.137E-04 | 1.223E-03 | 1.223E-03 | 1.594E-03 | 1.223E-03 |
| | Standard Deviation | 3.219E-04 | 2.722E-04 | 1.116E-04 | 2.635E-04 | 3.008E-04 | 2.835E-04 | 3.127E-04 |
| | | | | | | | | |
| **Six-hump** | Best | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 |
| | Average | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 |
| | Worst | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 | -1.032E+00 |
| | Standard Deviation | 8.165E-13 | 2.660E-12 | 8.360E-13 | 1.324E-12 | 2.313E-12 | 6.055E-13 | 1.005E-12 |
| | | | | | | | | |
| **Branin** | Best | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 |
| | Average | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 |
| | Worst | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 | 3.979E-01 |
| | Standard Deviation | 3.532E-11 | 2.302E-11 | 3.564E-11 | 1.977E-11 | 5.107E-11 | 2.049E-11 | 2.376E-11 |

A graphical representation of the results obtained from the fixed-dimension multimodal benchmark test functions is given in Figure 6.

## V. CONCLUSION

The Dandelion Optimizer algorithm is one of the most recent metaheuristic algorithms. As with other general-purpose metaheuristic algorithms, DO must be prevented from getting stuck at local points to achieve fast global convergence and accuracy rates. For this, in this study, chaotic maps were used instead of random numbers in the creation of the initial population, which is the first step of DO. Algorithms with initial populations created using Circle, Singer, Chebyshev, Gauss, Iterative and Logistic chaotic maps were run for a total of twelve quality test functions, four of which are unimodal (Sphere, Rosenbrock, Schwefel 2.22, and Step), four of which are multimodal (Schwefel, Ackley, Rastrigin, and Griewank) and four of which are fixed-dimension multimodal (Foxholes, Kowalik, Six-hump camel back, and Branin).

When the experimental results are examined, it is seen that the CIDO algorithms with the initial population created using chaotic maps get more efficient results compared to the classical DO. The chaotic version of the algorithm can be effectively used for many optimization problems in the future. Also, different chaotic maps can be adapted to increase the efficiency of the algorithm. Hybrid or multi-purpose ones can also be developed and used in different problems.



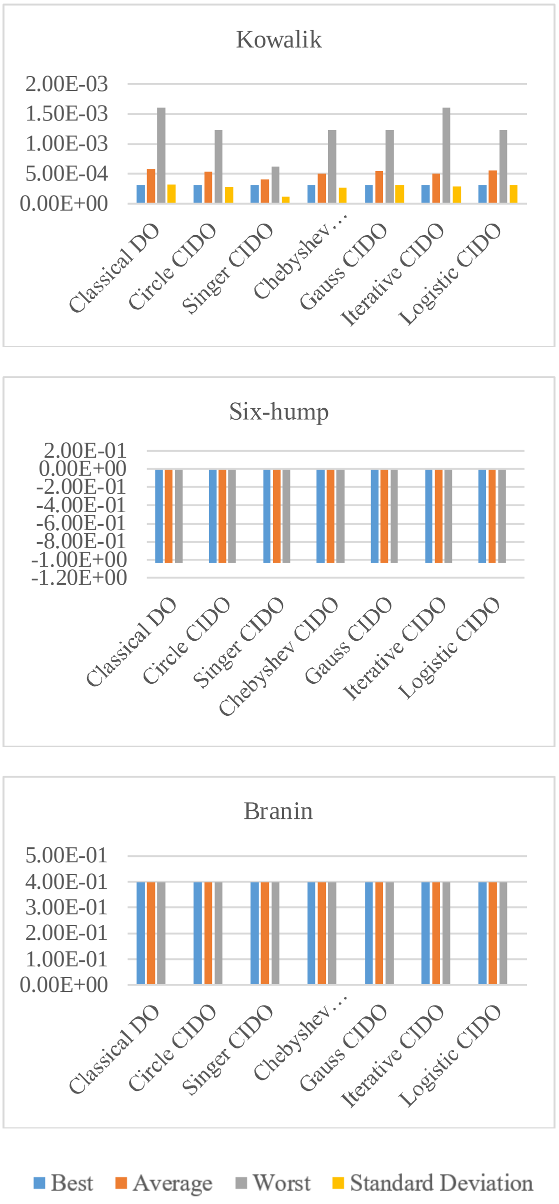


**Fig 6.**Graphical representation of results from fixed-dimension multimodal benchmark test functions

## REFERENCES

[1] Akyol, S., Alatas, B., Sentiment classification within online social media using whale optimization algorithm and social impact theory based optimization, *Physica A: Statistical Mechanics and its Applications*, 540, 2020,123094.

[2] Nasr, M., Farouk, O., Mohamedeen, A., Elrafie, A., Bedeir, M., &Khaled, A. Benchmarking meta-heuristic optimization,*International Journal of Advanced Networking and Applications*, 11, 6, 2020, 4451-4457.

[3] Akyol, S., A new hybrid method based on Aquila optimizer and tangent search algorithm for global optimization, *Journal of Ambient Intelligence and Humanized Computing*, 2022, 1-21.

[4] Meng, Q.A., Wang, Q., Zhao, K., Wang, P., Liu, P., Liu, H., Jiang, L., Hydroactuated configuration alteration of fibrous dandelion pappi: *Toward self-controllable transport behavior*, 2016.

[5] Cummins, C., Seale M., Macente A., Certini D., Mastropaolo E., Viola I.M., Nakayama N. A., Separated vortex ring underlies the flight of the dandelion, *Nature*, 562, 7727, 2018, 414-418.

[6] Soons, M.B., Heil, G.W., Nathan, R., Katul, G.G., Determinants of long-distance seed dispersal by wind in grasslands, *Ecology*, 85, 11, 2004, 3056-3068.

[7] Zhao, S., Zhang, T., Ma, S., & Chen, M., Dandelion Optimizer: A nature-inspired metaheuristic algorithm for engineering applications, Engineering Applications of Artificial Intelligence, 114, 2022, 105075.

[8] Cheng, M. Y., Tran, D. H., & Cao, M. T., Chaotic initialized multiple objective differential evolution with adaptive mutation strategy (CA-MODE) for construction project time-cost-quality trade-off, *Journal of Civil Engineering and Management*, 22, 2, 2016, 210-223.

[9] Alatas, B., Akin, E., & Ozer, A. B., Chaos embedded particle swarm optimization algorithms, *Chaos*, Solitons & Fractals, 40, 4, 2009, 1715-1734.

[10] Akyol, S., Alatas, B., Chaotically Initiated Flower Pollination Algorithm for Search and Optimization Problems. *II International Conference on Engineering and Natural Sciences (ICENS)*, 2016, 2797-2803.

## Biographies and Photographs

Dr. Sinem Akyol received her B.S. degree in Computer Engineering from Ege University in 2009. She received her M.S. degree in Electrical and Electronics Engineering from Munzur University in 2013 and Ph.D. degree in Computer Engineering from Firat University in 2018, in Elazig, Turkey. Currently, she is an assistant professor of Software Engineering at Firat University, Turkey. Her research interests include artificial intelligence, data mining, social network analysis, metaheuristic optimization, and machine learning.

**Dr. Muhammed Yildirim** received his Ph.D. degree in Computer Engineering from the University of Firat, in Elazig, Turkey. Currently, Muhammed Yildirim is an assistant professor of Computer Engineering at Malatya Turgut Ozal University, Turkey, where he is conducting research activities in the areas of artificial intelligence, optimization and image processing.

**Prof. Dr. Bilal** Alatas received his B.S. and M.S. degrees in Computer Engineering from Firat University in 2001 and 2003, respectively. He received Ph.D. degree from Firat University in 2007. Currently, he works as a Professor of Software Engineering at this department. He served as the chair of department of computer engineering at Munzur University during 2010-2014. He is the founder head of the Computer Engineering Department of Munzur University and Software Engineering Department of Firat University. His research interests include artificial intelligence, data mining, social network analysis, metaheuristic optimization, and machine learning. Dr. Alatas has published over 200 papers in many well-known international journals, proceedings of the refereed conferences, and books since 2001. Over 4850 citations of his works have been reported in the Google Scholar.