International Journal of Innovations in Engineering and Technology (IJIET)
http://dx.doi.org/10.21172/ijiet.151.03

Chaotic Cheetah Chase Algorithm

M. Goudhaman[1], S. Sasikumar[2], N. Vanathi[3]
[1]*Research Scholar - CSE, Saveetha School of Engineering,*
[2]*Faculty of CSE, Saveetha Engineering College*
[3]*Faculty of Science & Humanity, KCG College of Technology*

Abstract- The Cheetah Chase Algorithm (CCA) is a recently developed meta-heuristic optimization algorithm which is based on the hunting mechanism of Cheetah. Similarly to other meta-heuristic algorithms, the main problem faced by CCA is slow convergence speed. So to enhance the global convergence speed and to get better performance, this paper introduces chaos theory into CCA optimization process. Various chaotic maps are considered in the proposed chaotic CCA (CCCA) methods for tuning the main parameter of CCA which helps in controlling exploration and exploitation. The proposed CCCA methods are benchmarked on twenty well-known test functions. The results prove that the chaotic maps (especially Tent map) are able to improve the performance of CCA.
Keywords: Cheetah Chase Algorithm, Meta-heuristic algorithm, Chaos, Chaotic maps.

I. INTRODUCTION

In many optimization problems, it is required to find the optimal solution to a given problem under highly complex constraints in a reasonable amount of time. Generally, modern intelligent methods are used to deal with these types of optimization problems. There are various methods that are proposed in order to solve these problems but they are insufficient to produce better results. In the past few decades, meta-heuristic optimization algorithms have achieved a lot of attention in scientific communities with significant developments, especially for solving many complex optimization problems. Prior to meta-heuristic algorithms, Hill-Climbing, Random Search and Simulated Annealing (SA), were the traditional algorithms used to solve the optimization problems (Yang, X. S. 2010). Traditional algorithms start their search from a single point and require gradient information that consumed a lot of time to reach the global optima (Sivanandam, S. N. 2007). Due to their limited pertinence and intricacy of constraints, these algorithms were not very effective for solving real world applications like localization problem (Arora, S., & Singh, S. 2017), economical optimization (Gao, X. Z., et.al. 2010), structural optimization problems (Gandomi, A. H., et. al. 2013) and engineering design problems (Coello, C. A. C. 2000) which involve different constraints to be gratified.

Basically, meta-heuristic algorithms impersonate biological or physical phenomenon to handle complex real world optimization problems. Unlike classical techniques, these meta-heuristic algorithms are mostly derivation-free (Yang, X. S. 2010). Due to their stochastic nature, meta- heuristic algorithms have superior abilities to avoid local optima entrapment. These algorithms can be applied to various fields due to their simplicity, flexibility, robustness, and efficiency (Sivanandam, S. N. 2007).Some of the most prominent nature inspired meta-heuristic algorithms developed so far are Particle Swarm Optimization (PSO) (Eberhart, R., & Kennedy, J. 1995; Kennedy, J. 2011), Artificial Bee Colony (ABC) (Dorigo, M., & Di Caro, G. 1999; Dorigo, M., & Gambardella, L. M. 1997), Firefly Algorithm (FA) (Yang, X. S. 2010), Biogeography-Based Optimization algorithm (BBO) (Simon, D. 2008), Bat Algorithm (Gandomi, A. H., et. al. 2013; Tsai, P. W. 2015), Krill-Herd (KH) (Gandomi, A. H., & Alavi, A. H. 2012), Cat Swarm Optimization (Shu-Chuan et al. 2006),Grey Wolf Optimizer (GWO) (Mirjalili, S., & Lewis, A. 2014),Ant Lion Optimizer (ALO) (Mirjalili, S. 2015),Butterfly Optimization Algorithm (BOA) (Arora, S., & Singh, S. 2015)and most recently Whale Optimization Algorithm (CCA)(Mirjalili, S., & Lewis, A. 2016).

Above all, the most challenging task encountered in the development of any meta-heuristic algorithm is to find a proper balance between exploration and exploitation due to the stochastic nature of the optimization process (Mirjalili, S., & Lewis, A. 2016). The exploration phase helps the optimizer to globally explore the search space as extensively as possible. Also, the population faces some abrupt changes in this phase. In contrast, the exploitation phase involves the refinement of the promising solutions obtained from the exploration phase. Here, the population encounters small abrupt changes (Alba, E., & Dorronsoro, B. 2005).

CCA is a recently developed nature-inspired meta-heuristic that imitates the social behavior of Cheetah (Goudhaman.M, 2018). This algorithm is inspired by the process of hunting and chasing of Cheetah to capture its prey with the parameters of high speed, velocity and greater accelerations. The method includes three main steps of hunting, i.e., encompassing prey, searching for prey and chasing the prey. It has been proved that this algorithm is able to show very competitive results compared to other meta-heuristic algorithms in solving various real world problems. To further enhance the performance this paper introduces chaos theory into CCA.

With the development of the non-linear dynamics chaos theory has been extensively utilized in various applications (Pecora, L. M., &Carroll, T. L. 1990). Chaos theory is related to the study of chaotic dynamical systems that are highly sensitive to initial conditions and includes infinite unstable periodic motions. In order to improve performance, chaos has been employed in various meta-heuristic algorithms, which results in better convergence speed and avoidance from local optima entrapment (Kellert, S. H. 1994). Although it appears to be stochastic, providing chaotic behavior does not crucially need stochastic. Deterministic systems are also able to show chaotic behaviors. Earlier, chaos theory has been utilized by various meta-heuristic algorithms such as genetic algorithm (Li-Jiang, Y., & Tian-Lun, C. 2002), harmony search (Alatas, B. 2010), PSO (Liu, B. et.al. 2005),ABC (Alatas, B. 2010), FA (Yang, X. S. et.at. 2013), KH(Wang, G. G. et.al. 2014), BOA (Arora, S., & Singh, S. 2017) and GWO(Kohli, M., & Arora, S. 2017) to enhance the performance of the algorithms by tuning certain parameters.
The aim of this paper is to introduce Chaotic Cheetah Chase Algorithm (CCA) based methods in which different chaotic systems are used to replace the critical parameter of CCA that helps to switch the local and global searching ability of CCA. A subset of unimodal and multimodal benchmark functions have been employed in order to evaluate the proposed CCCA.
The rest of the paper is organized as follows. Review of CCA is presented in Section 2. The chaotic maps that describe chaotic sequences for CCA are described in Section 3. In Section 4, the proposed CCCA have been presented. The experimental results have been described in Section 5. Finally, the conclusions and future work have been discussed in Section 6.

1.1 Cheetah Chase Algorithm (CCA)

The Cheetah is a giant and energetic civet that was once found all through Asia, Africa and certain places of Europe. Cheetahs are one of Africa's most energetic predators and are most famous for their monstrous speed when in a chase. Equipped for achieving speeds of more than 60mph for minimum span of time, Cheetah is the speediest land vertebrate on the earth. The Cheetah is one of a kind among Africa's civets principally on the grounds that they are most dynamic amid the day, which keeps away from rivalry for nourishment from other substantial predators like Lions and Hyenas that chase amid the cooler night. The Cheetah has outstanding visual perception thus chases utilizing sight by first stalking its prey from between 10 to 30 meters away, and after that pursuing it when the time is correct.[17][18]
The light and thin body of the cheetah influences it to appropriate to short, dangerous blasts of speed, hasty acceleration, and a capability to execute extraordinary alters in course while moving at speed. These behaviours represent unique features of the cheetah's capability to capture fast moving prey.
Cheetahs can start from 0 miles for per hour to 65 miles per hour in only 3.5 seconds. Cheetahs can achieve a best speed anyplace in the middle of 60 and 70 miles per hour, varies on the size of cheetah. But, the fascinating thing is that cheetahs can just run that quick for 20 to 30 seconds. Along these lines, they can't maintain that speed for long circumstances. What is the reason they can't run that quick for long? All things considered, in light of the fact that keeping up that speed for any more extended than 20-30 seconds could have an exceptionally negative impact on their organs, and the cheetah could experience the ill effects of extraordinary over-effort and over-warming.

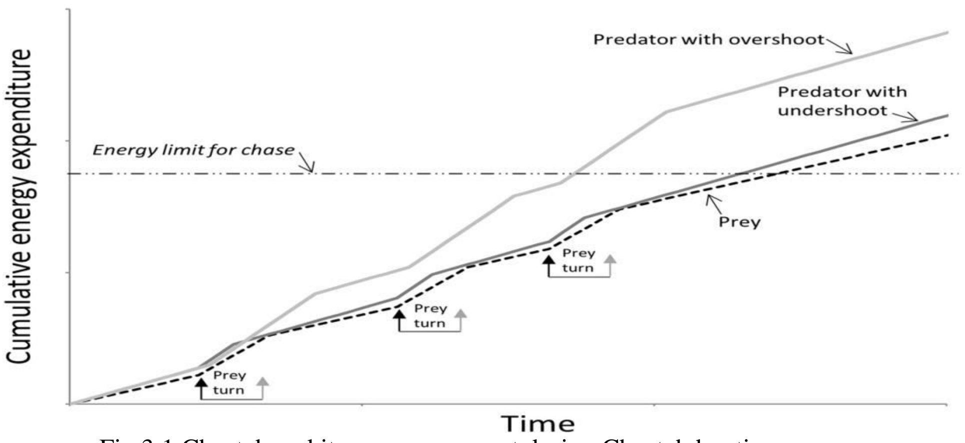


Fig 3.1 Cheetah and its prey movement during Cheetah hunting process

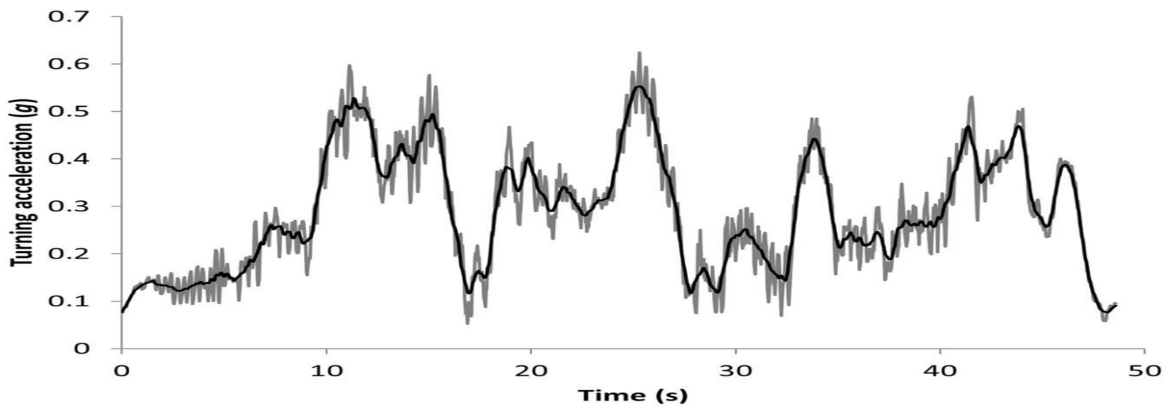


Fig 3.2 Graphical sum of the static heave and sway acceleration axes during cheetah hunting its prey.

Elliot et al., (1977) gave an applied model to prey securing by earthly carnivores depicting four noteworthy components; look, stalk, assault, and stifle. Of these, the assault is the most power-requesting (Williams et al., 2014), ordinarily including complex fast moves, supported by obviously entangled behavioural alternatives for the both predators and prey.

The speediest land vertebrate, the cheetah, can fasten from a standing start to 95 km/h in only three seconds, which compares to an acceleration of 8.8 m/s2. Cheetahs can just keep up their quickest pace (111 km/h) for roughly 400 m before their body overheats and their muscles start to tire and create lactic corrosive from fatigue.

```
Step 1: Initialization of the parameters like speed, velocity, acceleration, time and distance.
Step 2: Initialize total time, tc and total distance travelled dc.
Step 3: While (t < Tmax), // Multiple numbers of iterations based on number of Cheetah's.
Step 4: At start node measure the parameters like speed (sc), velocity (vc ) and acceleration (ac).
// Start of the chase Cheetah can accelerate to top speed for first few seconds.
Step 5: Move on to next node and update parameters.
Step 6: If prey captured by Cheetah
        Step 6.1: Estimate the total time, tc = tc1 + tc2.,
    Where tc1 is time to get the cheetah to accelerate to top speed.
        tc2 is that travel certain distance at top speed
        Step 6.2: Estimate the total distance dc = dc1 + dc2
        Where dc1 and dc2 are the distances went in times tc1 and tc2 respectively.
        Step 6.3: Estimate distance the prey can go in time tc, dp = dp1 + dp2,
        Step 6.4: Estimate the maximum distance travelled by prey dmax= dc – dp.
    Else
        Repeat step 4.
Step 8: End while.
Step 9: Select the best possible shortest path node details with other parameters Speed (sc), Velocity (vc ) and Acceleration (ac)..
Step 10: Post-process and Visualization.
```

The pseudo code of the Cheetah Chase Algorithm

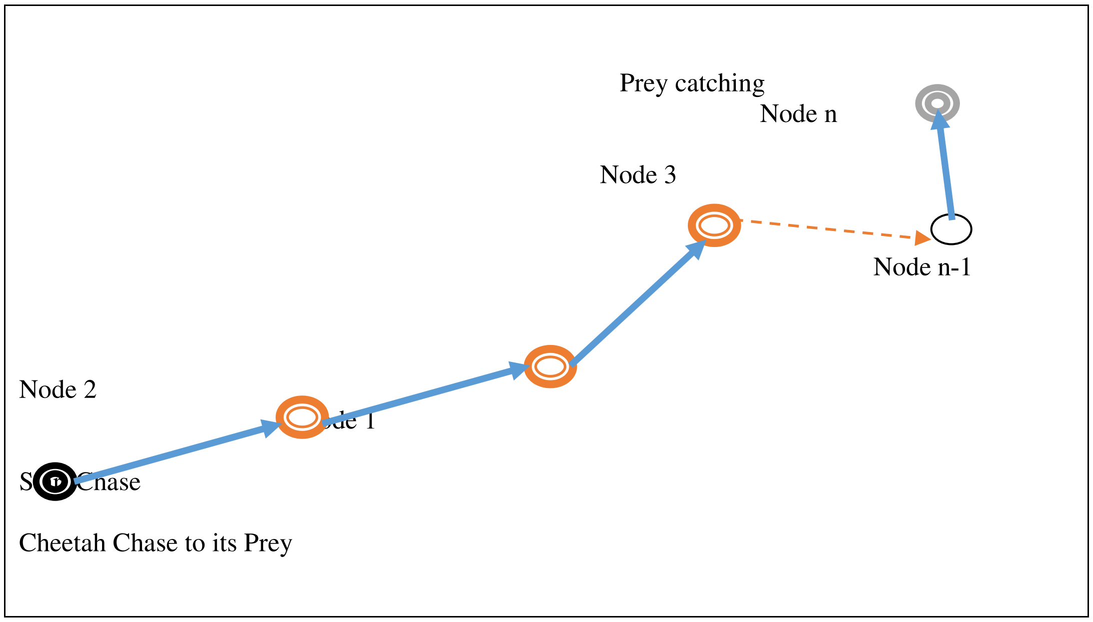
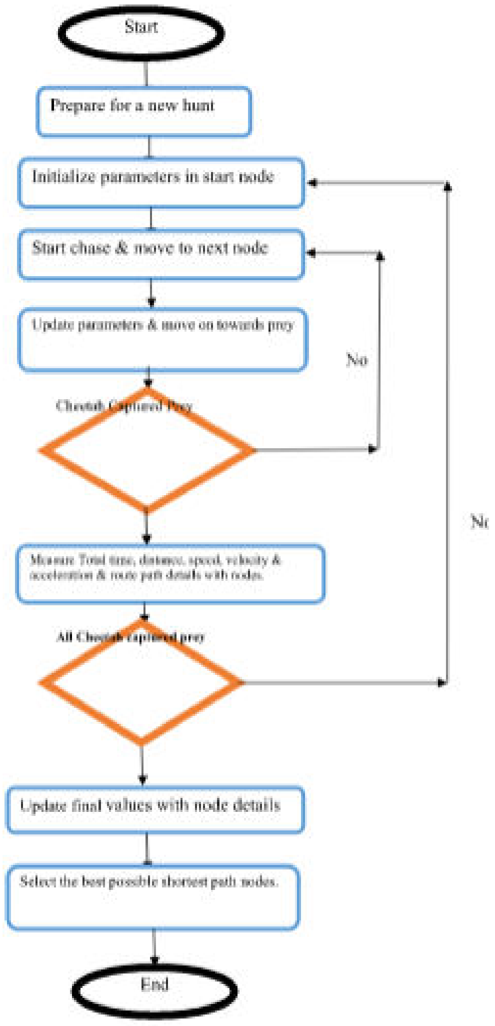


Figure 3.3: Cheetah Chase to its Prey


Start
Prepare for a new hunt
Initialize parameters in start node
Start chase & move to next node
Update parameters & move on towards prey
No
Cheetah Captured Prey
Nc
Measure Total time, distance, speed, velocity & acceleration & route path details with nodes.
All Cheetah captured prey
Update final values with node details
Select the best possible shortest path nodes.
End

Flowchart of Cheetah Chase Algorithm

II. CHAOTIC MAPS

In this section, we present ten 1-D non-invertible chaotic maps that are utilized to produce chaotic sets. This set of chaotic maps with initial point 0.7 has been chosen with different behaviors. The initial point can be chosen any number between 0 and 1 (or depends on the range of chaotic map). However, it should be noted that the initial value may have significant impacts on the fluctuation pattern of some of the chaotic maps (Saremi, S.2014). Recently, a lot of chaotic maps were discovered primarily by mathematicians and physicians applicable to different domains of human activity. In line with this, the majority of these were applied to different algorithms for solving the various real-world problems. Therefore, the more applicable
chaotic maps tackling the optimization algorithms are used in the current research work as shown in Table 1 (Kohli, M., & Arora, S. 2017;Gandomi, A. H et al. 2013; Gandomi, A. H & Yang X. S. 2014).

III. CHAOTIC CHEETAH CHASE ALGORITHM (CCCA)

In spite of having good convergence rate, CCA still cannot perform that better in finding the global optima which affect the convergence rate of the algorithm. So, to reduce this affect and to improve its efficiency, CCCA algorithm is developed by introducing chaos in CCA algorithm itself. Generally, chaotic comes from the word 'chaos' which means the property of a complex system whose behavior is so unpredictable, and map means mapping or associating chaos behavior in the algorithm with some parameter using a function. Due to the ergodicity and non- repetition properties of chaos, it can perform overall searches at higher speeds compared to the stochastic searches that basically rely on probabilities (Santos Coelho, L., & Mariani, V. C. 2008). Chaotic maps are the maps which show complex and dynamic behavior in the non-linear systems (Pecora, L. M., &Carroll, T. L. 1990).

Table 1 Chaotic Maps

S. No	Name	Chaotic map
1	Logistic	xi+1 = axi(1-xi)
2	Cubic	xi+1 = axi(1-x_i^2)
3	Sine	xi+1 = $\frac{a}{4}$ sin (πxi)
4	Sinusoidal	xi+1 = a x_i^2 sin (πxi)
5	Singer	xi+1 = μ(7.86xi – 23.31 x_i^2 + 28.75 x_i^3 - 13.302875 x_i^4), μ = 1.07
6	Circle	xi+1 = mod(xi + b - ($\frac{a}{2\pi}$ sin (2πxk), 1)
7	Iterative	xi+1 = sin (a$\pi/$xi)
8	Tent	xi+1 = {xi / 0.7 xi < 0.7 {10/3 (1 – xi)} xi $\geq$ 0.7
9	Piecewise	xi+1 = xi / P 0 ≤ xi < P xi – P / 0.5 - P P ≤ xi < 0.5 P = 0.4 1 – P - xi / 0.5 – P0.5 ≤ xi < 1 - P 1 - xi / P0.5 1- P ≤ xi < 1
10	Gauss/mouse	xi+1 = {$\frac{1}{1}$ / mod(xi , 1)} xi = 0, otherwise 1/ xi mod(1) = 1/xi – [1/xi]

Due to their dynamic behavior, chaotic maps have been widely acknowledged in the field of optimization which helps optimization algorithm in exploring the search space more vigorously and globally (Yang, D., Li, G., & Cheng, G. 2007).In almost all meta-heuristic algorithms with stochastic components, randomness is achieved by using probability distributions. It can be advantageous to replace such randomness by using chaotic maps. In order to introduce chaos in optimization algorithm, different chaotic maps having different mathematical equations are used that are listed in Table 1. Since last decade, chaotic maps have been extensively appreciated in the field of optimization algorithms in exploring the search space more dynamically and globally. In consonance with different human's domain a large variation of chaotic maps designed by physicians, researchers and mathematicians are available in the optimization field (He, D., He, C. 2001). Out of all these, ten most significant uni-dimensional chaotic maps (Gandomi, A. H., & Yang, X. S. 2014) have been employed in the present work to tackle CCCA, details of which are given in Table 1.

The convergence rate of CCA has been positively influenced by utilizing chaotic maps as these maps persuade chaos in the feasible region which is predicted only for very short initial time and is stochastic for a longer period of time (Yang, D.,& Cheng, G.2007). Pseudo code of the proposed CCCA algorithm is illustrated in Algorithm 2.

The optimization procedure of the proposed CCCA is also presented in the form of flowchart given in Fig 1. The very first step of the flowchart implicates the stochastic initialization of population of whales. Then, a respective chaotic map is chosen to be mapped with the algorithm along with the initialization of its first chaotic number and a variable (Gandomi, A. H., & Yang, X. S. 2014). After this the parameters of the CCCA algorithm involved in controlling the exploration and exploitation mechanism specifically a, A, C, l and p are initialized which are same as in CCA. Also, the chaotic number of the chaotic map is initialized to adjust the parameter ‘p’ of CCA which is highlighted in Fig 1.In the next step, fitness of all the whales initialized in the search space is evaluated using the various standard benchmark functions. The whale with the highest fitness is assumed to be the current best search agent. The current best search agent will keep updating its position using Eq. (1), when the value of control parameter A< 1. Similarly, when the value of parameter A≥ 1, a random whale is chosen and the position of the current best search agent is updated using Eq. (8) if there is a new best search agent than the last one. Sequentially the fitter whale will keep updating its position and at the end may get the first position as optimal solution. The value of parameter ‘p’ is also updated along with the course of iterations using Eq. (3) and Eq. (4). At the end of the last iteration, the best search agent will be considered as the most optimal solution by the CCCA algorithm.

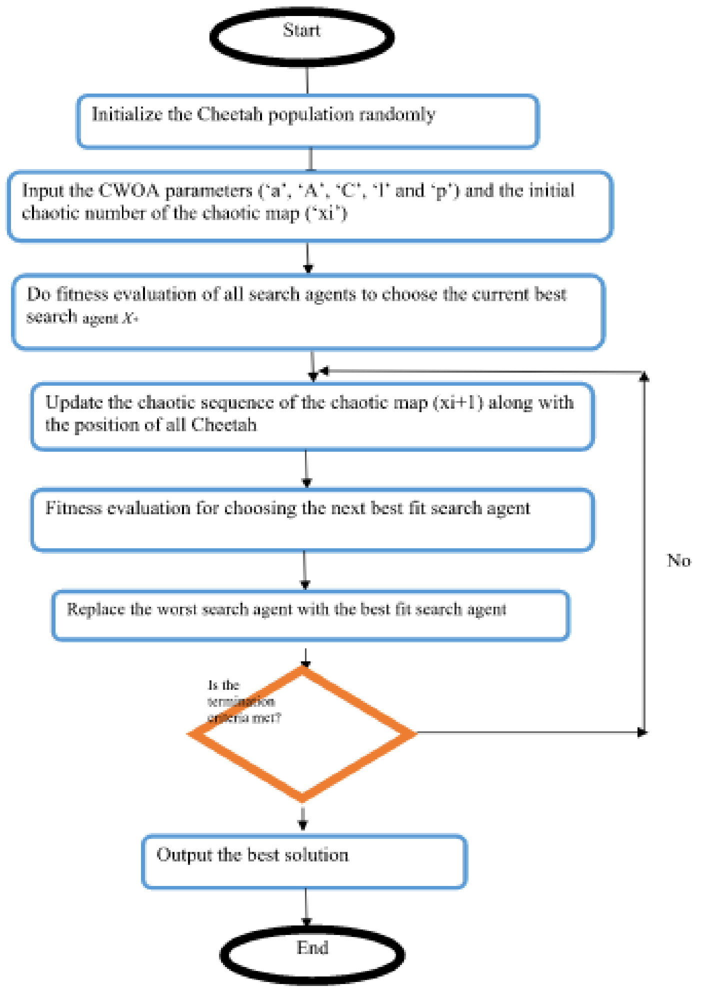

Start
Initialize the Cheetah population randomly
Input the CWOA parameters (‘a’, ‘A’, ‘C’, ‘l’ and ‘p’) and the initial chaotic number of the chaotic map (‘xi’)
Do fitness evaluation of all search agents to choose the current best search agent X*
Update the chaotic sequence of the chaotic map (xi+1) along with the position of all Cheetah
Fitness evaluation for choosing the next best fit search agent
Replace the worst search agent with the best fit search agent
Is the termination criteria met?
No
Output the best solution
End

Fig. 1. Flowchart of optimization procedure of CCCA

```
Initialize the generation counter t and randomly initialize the whale's population Xi(i=1, 2,…,n)
Evaluate the fitness of each search agent to find the best search agent X*
Initialize the value of the chaotic mapx0 randomly
while (t < maximum number of iterations)
Update the chaotic number using the respective chaotic map equation for each search agent
Update a, A, C, l and p
if1 (p < 0.5)
if2( | A | < 1)
Update the position of the current search agent by the Eq. (1)
else if2 ( | A | ≥ 1)
Select a random search agent (Xrand)
Update the position of the current search agent by Eq. (8)
end if2
else if1 (p ≥ 0.5)
Update the position of the current search by the Eq. (5)
end if1
end for
Check if any search agent goes beyond the search space and amend it
Calculate the fitness of each search agent
Update X* if there is a better solution
t=t+1
end while
return X*
```

Algorithm 2. Pseudo-code of CCCA algorithm

IV. EXPERIMENTAL RESULTS AND DISCUSSIONS

In order to measure and test the performance of every novel optimization algorithm, the algorithm must deal with some well-defined test functions. In this section, various experiments on optimization benchmark problems are implemented to verify the performance of the proposed meta-heuristic CCCA method. Twenty well-known benchmark functions (Digalakis, J. G., 2001, Yao, X., 1999) have been utilized in order to check the performance of CCCA. These functions are divided into two categories: unimodal and multimodal. Unimodal benchmark functions have single optima and they are well suited for benchmarking exploitation. In contrast, multimodal benchmark functions have more than one optima that makes them more challenging than unimodal functions. One of the optima is called global optima and the rest are called local optima. Avoiding the local optima and determining the global optimum should be the main characteristics of any powerful meta-heuristic algorithm. Therefore, the multimodal benchmark functions are responsible for testing exploration and avoiding the entrapment in local optima. Note that the minima of most of the unimodal and multimodal benchmark functions is 0 except some functions ,i.e., F10, F15, F16 and F20. The properties of unimodal and multimodal benchmark functions are listed in Table 2, where Dim indicates the dimension of the function, and Range is the boundary of the function's search space. The performance of CCCA with different chaotic maps and their results have been discussed in Section 5.1. Also, the qualitative analysis and the statistical testing of the results have been clearly described in Sections 5.2 and 5.3 respectively.
For the results of various CCCA's, the population size of the whales is taken 30 and 50 iterations are performed. The results are averaged over thirty independent runs.CCCA1 to CCCA10 utilize Logistic, Cubic, Sine, Sinusoidal, Singer, Circle, Iterative, Tent, Piecewise and Gauss/mouse maps, respectively as shown in Table 1. It can be seen from the Table 3 that CCCA6 andCCCA7 algorithms shows worse results as compared to CCA algorithm. This shows that, Circle and Iterative chaotic maps are not able to improve the performance of CCA algorithm. In contrast, CCCA1, CCCA2, CCCA3, CCCA4, CCCA5, CCCA8, CCCA9 and CCCA10 algorithms shows much better results as compared to CCA algorithm. In other words, Logistic, Cubic, Sine, Sinusoidal, Singer, Tent, Piecewise and Gauss/mouse chaotic maps are able to enhance the performance of CCA algorithm successfully. From the results of Table 3, it is proved that the Tent–based CCA algorithm yields the best results on all the test functions. The p values depicted that this supremacy is statistically significant. It can also be seen from the table that the Cubic, Sine, Piecewise and Tent chaotic maps yield the best results in more than thirteen test functions. On the other hand, Circle and Iterative maps have provided worse results in more than thirteen test functions.

S. No	Bench Mark Function	Formula	Dim	Range	Optimal Value
F1	Sphere	fx = $\sum_{i=1}^{n} x$2i	30	[-100,100]	0
F2	Beale	fx = (1.5 – xi + xi xi+1)2 + (2.25 – xi + xi x_{i+1}^2)2 +(2.25 – xi + xi x_{i+1}^3)2	2	[-4.5,4.5]	0
F3	Cigar	fx = $x_i^2 + \sum_{i=2}^{n} x$2i	30	[-100,100]	0
F4	Step	fx = $\sum_{i=1}^{n-1}(\lvert xi\rvert + 0.5)$2	30	[-100,100]	0
F5	QuarticNoise	fx = $\sum_{i=1}^{n-1} x_i^4 + N(0,1)$2	30	[-1.28,1.28]	0
F6	Bohachevsky	fx=x_i^2+2.0 x_{i+1}^2-0.3cos(3$\pi xi)\cos(4\pi x_{i+1})$+0.7	2	[-100,100]	0
F7	Ackley	fx = - 20 exp(0.02) $\sqrt{1}$ / D $\sum_{i=1}^{D} x_i^2 - \exp\ (\frac{1}{D}\sum_{i=1}^{D} Cos\ (2\pi(xi$) + 20 +exp	30	[-32.768,32.768]	0
F8	Griewank	fx = $\frac{1}{4000}\sum_{i=1}^{n-1}(x_i - 100)^2) - \prod_{i=1}^{n-1}\cos(\frac{xi-100}{\sqrt{i-1}}) + 1$	30	[-600,600]	0
F9	Levy	fx = sin2($\pi\omega 1$) + $\sum_{i=1}^{D-1}(\omega i - 1)$2 [1+ 10sin2($\pi\omega i + 1$)] + ($\omega d - 1)2\ [1 + sin2(2\pi\omega d]$	30	[-10,10]	0
F10	Michalewitz	fx = $\sum_{i=1}^{n}(\sin(xi)\ sin20(\frac{ix_i^2}{\pi}$)2m	10	[-100,100]	0
F11	Rastrigin	fx =(x_i^2 - 10 cos 2πx_i + 10)	30	[-5.12,5.12]	0
F12	Alpine	fx = $\sum_{i=1}^{n}\lvert x_i \sin(x_i) + 0.1\ x_i\rvert$	30	[-100,100]	0
F13	Schaffer	fx = ($(x_i^2 + x_{i+1}^2)^{1/4}$(50 $(x_i^2 + x_{i+1}^2)^{0.1} + 1$	2	[-100,100]	0
F14	Rosenbrock	fx = 100($x_{i+1} - x_i^2)^2 + (1.0 - x_i)^2$	30	[-10,10]	0
F15	Easom	fx = cos ($x_i)\cos(x_{i+1})\exp(-(x_i - \Omega)^2\quad(-(x_{i+1} - \Omega)^2$)	2	[-100,100]	-1
F16	Shubert	fx = ($\sum_{i=1}^{5} i\cos(i+1)\,x_i + i) + \sum_{i=1}^{5} i\cos(i+1)\,x_{i+1} + i)$	2	[-10,10]	-186.73
F17	Schwefel 1.2	$f_x = \sum_{i=0}^{n-1}\ \{\sum_{j=0}^{j<1} x_i$} 2	30	[-10,10]	0
F18	Schwefel 2.21	$f_x = \max(\lvert x_i\rvert)$	30	[-10,10]	0
F19	Schwefel 2.22	$f_x = \sum_{i=1}^{n-1}\lvert x_i\rvert + \prod_{i=1}^{n-1}\lvert x_i\rvert$	30	[-10,10]	0
F20	Schwefel 2.26	$f_x = -\sum_{i=0}^{n-1}(x_i \sin\sqrt{x_i})$	30	[-500,500]	-12569.5

4.1. Qualitative analysis

Qualitative analysis on different benchmark functions for further effective evaluation of the performance of CCCA has also been done. The line graphs of convergence of various benchmark functions using CCCA algorithm have been shown in Figs. 2-6, which helps to analyze the convergence rate of the algorithm more evidently. To clearly notice and analyze the convergence curves of CCCA on various chaotic maps the graphs have been plotted on 50 iterations. Table 3 shows that on average CCCA8 (Tent map) performs better than other methods on nine of the benchmarks when searching for function minimum. CCCA2, CCCA3 and CCCA7 (Cubic, Sine, and Iterative respectively) are the second best maps performing best on seven out of twenty benchmarks. CCCA1, CCCA4, CCCA5, and CCCA10 (Logistic, Sinusoidal, Singer and Gauss/mouse respectively)are the third most effective and have shown the best performance on six benchmarks. The values shown in these figures are the average function optimum achieved from thirty runs. Here, all the values are true function values.

Fig.2 shows the values obtained by the ten chaotic maps on the F01 Sphere function, which is also known as de jong’s function and has a single global value F01min=0, therefore it is easy to solve. From Fig.2 CCCA9 (Piecewise map) has the fastest convergence rate towards the global solution and overtakes all other methods. CCCA6and CCCA7 (Circle and Iterative maps respectively) fail to find the global value within the maximum number of iterations.

Fig.3 displays the function values for the F03Cigar function. CCCA8 (Tent map) shows the fastest convergence rate and overtakes all other methods. It can also be seen that only CCCA6 and CCCA7 (Circle and Iterative respectively) are inferior to CCCA8 (Tent map). Otherwise, all other maps are very close to CCCA8 in showing a very good convergence rate.

Fig.4 illustrates the values achieved for the ten methods when using the F08 Greiwank function. F08 has a strange property, as it is much easier to solve for higher dimensions than lower dimensions (Liang, J. J., & Baskar, S. 2006). From Fig. 4 all the maps have shown the fastest convergence rate towards the global optimum than CCA except CCCA6, CCCA7, and CCCA10 (Circle, Iterative and Gauss/mouse maps respectively).

Fig. 5 shows the functions values for the F13 Schaffer function, which is a unimodal function. From Fig. 5, CCCA6 (Circle map) performs better than other nine methods while
CCCA7 (Iterative map) performs second best in this function.

Fig. 6 reveals the function values for the F18 Schwefel 2.21 function. At first glimpse, it is obvious that CCCA8 has the fastest convergence rate towards the global solution. CCCA8
(Tent map) reaches the optimal solution significantly earlier than other methods. From Fig. 6, it is also illustrated that CCCA9 and CCCA10 (Piecewise and Gauss/mouse maps respectively) are only inferior to CCCA8 (Tent map) and perform second best in this unimodal function.

Considering the results shown in Fig. 2-6, it can be concluded that CCCA has superior performance than CCA. Further, CCCA8 (Tent map) have provided superior results on nine benchmark functions as compared to CCA.

4.2. Statistical testing

To evaluate the performance of meta-heuristic algorithms, statistical tests should be conducted (Derrac, J. et.al. 2011). Specifically, it is not adequate to compare algorithms based on the mean and standard deviation values (García, S. et.al. 2009), and a statistical test is necessary to prove that a proposed new algorithm presents a significant improvement compared to other algorithms. In order to judge whether the results of the algorithms differ from each other in a statistically significant way, a nonparametric statistical test, Wilcoxon’s rank-sum test (Wilcoxon, F. 1945) is carried out at 5% significance level. The average (mean) and standard deviation (SD) of the best solutions obtained in the last iteration are reflected in Table 3. The p values calculated in the Wilcoxon’s rank-sum are given in the results as well. In the tables, N/A indicates “not applicable”, meaning that the corresponding algorithm could not be compared with itself in the rank-sum test. Generally, it is considered that p values < 0.05 can be considered as sufficient indication against the null hypothesis. Note that the best results are highlighted in bold face and p values > 0.05 are underlined. Generally speaking, the results of the chaotic maps on all the benchmark functions follow the order of Tent < Cubic Sine Piecewise < Logistic Singer Sinusoidal Gauss/mouse < Circle Iterative. Note that the ‘ ’ sign signifies an approximately equal number of successful results of different chaotic maps in many benchmark functions. This comparison shows that the Tent map shows the best (minimum) results, whereas the Circle and the Iterative maps provide the worst (maximum) results. The underlying reason behind the better performance of CCCA using Tent chaotic map is that it provides better exploration and local optima avoidance capability. In other words, the Tent map bring different patterns of search behaviour for whales which results in showing higher exploration capability. The results of convergence curves also prove that the superior exploration of the Tent map does not have a negative impact on the exploitation. To sum up, the results show that the Tent chaotic map shows very effective results and can successfully enhance the convergence rate of CCCA.

Table 3 Results of 10 chaotic maps on all benchmark functions on CCCA

Sphere	Mean	Std. Dev.	p Values	Beale	Mean	Std. Dev.	p Values
CCA	2.58E-52	7.98E-52	0.005	CCA	1.62E-01	3.41E-01	0.386
CCCA1	3.00E-69	6.10E-69	0.445	CCCA1	4.14E-03	6.46E-03	0.241
CCCA2	1.11E-67	3.51E-67	0.799	CCCA2	9.93E-03	2.49E-02	0.093
CCCA3	1.80E-70	3.41E-70	0.959	CCCA3	8.94E-04	8.36E-04	N/A
CCCA4	2.00E-69	5.606E-69	0.721	CCCA4	5.57E-03	1.44E-02	0.878
CCCA5	2.95E-70	4.45E-70	0.721	CCCA5	5.83E-01	5.16E-01	0.005
CCCA6	1.64E+08	2.23E+08	0.005	CCCA6	7.24E-01	2.32E-01	0.005
CCCA7	1.14E+08	1.43E+08	0.005	CCCA7	7.24E-01	2.32E-01	0.005
CCCA8	3.26E-70	7.52E-70	0.445	CCCA8	2.24E-01	4.67E-01	0.028
CCCA9	1.68E-70	2.58E-70	N/A	CCCA9	1.80E-01	3.68E-01	0.013
CCCA10	2.78E-70	7.71E-70	0.508	CCCA10	9.75E-01	3.98E-01	0.005

Step	Mean	Std. Dev.	p Values	Quadrtic Noise	Mean	Std. Dev.	p Values
CCA	0.00E+00	0.00E+00	N/A	CCA	3.05E-02	3.37E-02	0.799
CCCA1	0.00E+00	0.00E+00	N/A	CCCA1	4.08E-02	4.59E-02	0.386
CCCA2	0.00E+00	0.00E+00	N/A	CCCA2	3.60E-02	2.64E-02	0.059
CCCA3	0.00E+00	0.00E+00	N/A	CCCA3	3.34E-02	2.39E-02	0.169
CCCA4	0.00E+00	0.00E+00	N/A	CCCA4	1.95E-02	1.83E-02	0.878
CCCA5	0.00E+00	0.00E+00	N/A	CCCA5	3.09E-02	2.50E-02	0.386
CCCA6	2.91E+04	1.40E+04	0.005	CCCA6	2.40E+00	1.12E+00	0.005
CCCA7	2.81E+04	6.24E+03	0.005	CCCA7	1.82E+00	6.20E-01	0.005
CCCA8	0.00E+00	0.00E+00	N/A	CCCA8	1.91E-02	1.54E-02	N/A
CCCA9	0.00E+00	0.00E+00	N/A	CCCA9	4.47E-02	3.51E-02	0.047
CCCA10	0.00E+00	0.00E+00	N/A	CCCA10	2.11E-02	2.99E-02	0.575

Ackley	Mean	Std. Dev.	p Values	Greiwank	Mean	Std. Dev.	p Values
CCA	-1.44E-16	0.00E+00	N/A	CCA	0.00E+00	0.00E+00	N/A
CCCA1	-1.44E-16	0.00E+00	N/A	CCCA1	0.00E+00	0.00E+00	N/A
CCCA2	-1.44E-16	0.00E+00	N/A	CCCA2	0.00E+00	0.00E+00	N/A
CCCA3	-1.44E-16	9.86E-21	N/A	CCCA3	0.00E+00	0.00E+00	N/A
CCCA4	-1.4E-16	9.86E-21	N/A	CCCA4	0.00E+00	0.00E+00	N/A
CCCA5	-1.4E-16	8.05E-21	N/A	CCCA5	0.00E+00	0.00E+00	N/A
CCCA6	1.81E+01	1.11E+00	0.005	CCCA6	1.41E+02	6.63E+01	0.005
CCCA7	1.78E+01	1.86E+00	0.005	CCCA7	1.04E+02	4.36E+01	0.005
CCCA8	-1.44E-16	0.00E+00	N/A	CCCA8	0.00E+00	0.00E+00	N/A
CCCA9	-1.44E-16	0.00E+00	N/A	CCCA9	0.00E+00	0.00E+00	N/A
CCCA10	-1.44E-16	0.00E+00	N/A	CCCA10	0.00E+00	0.00E+00	N/A

Michalewitz	Mean	Std. Dev.	p Values	Rastrigin	Mean	Std. Dev.	p Values
CCA	-3.04E+00	7.21E-01	0.013	CCA	0.00E+00	0.00E+00	N/A
CCCA1	-2.98E+00	3.67E-01	0.005	CCCA1	0.00E+00	0.00E+00	N/A
CCCA2	-2.98E+00	7.03E-01	0.009	CCCA2	0.00E+00	0.00E+00	N/A
CCCA3	-2.96E+00	6.15E-01	0.005	CCCA3	0.00E+00	0.00E+00	N/A
CCCA4	-2.77E+00	5.21E-01	0.005	CCCA4	0.00E+00	0.00E+00	N/A
CCCA5	-3.02E+00	8.82E-01	0.009	CCCA5	0.00E+00	0.00E+00	N/A
CCCA6	-4.27E+00	7.78E-01	0.386	CCCA6	2.54E+02	4.56E+01	0.005
CCCA7	-4.40E+00	8.75E-01	N/A	CCCA7	2.55E+02	4.71E+01	0.005
CCCA8	-2.51E+00	5.05E-01	0.005	CCCA8	0.00E+00	0.00E+00	N/A

CCCA9	-2.94E+00	6.06E-01	0.007	CCCA9	0.00E+00	0.00E+00	N/A
CCCA10	-2.68E+00	7.53E-01	0.009	CCCA10	0.00E+00	0.00E+00	N/A

Schaffer	Mean	Std. Dev.	p Values	Rosenbrock	Mean	Std. Dev.	p Values
CCA	5.13E-26	9.66E-26	0.005	CCA	2.88E+01	2.44E-02	N/A
CCCA1	4.37E-29	1.10E-28	0.005	CCCA1	2.88E+01	4.59E-02	N/A
CCCA2	1.43E-28	3.62E-28	0.005	CCCA2	2.88E+01	3.61E-02	N/A
CCCA3	5.17E-29	1.06E-28	0.005	CCCA3	2.88E+01	2.48E-02	N/A
CCCA4	1.39E-28	4.39E-28	0.005	CCCA4	2.88E+01	1.73E-02	N/A
CCCA5	4.05E-29	1.27E-28	0.005	CCCA5	2.88E+01	2.91E-02	N/A
CCCA6	4.04E-37	1.16E-36	N/A	CCCA6	4.61E+05	5.17E+05	0.005
CCCA7	2.24E-36	7.05E-36	0.093	CCCA7	1.95E+05	2.18E+05	0.005
CCCA8	2.19E-30	5.37E-30	0.005	CCCA8	2.88E+01	3.78E-02	0.317
CCCA9	2.79E-31	4.49E-31	0.005	CCCA9	2.88E+01	1.45E-02	0.317
CCCA10	8.37E-29	1.71E-28	0.005	CCCA10	2.88E+01	2.46E-02	0.317

Shubert	Mean	Std. Dev.	p Values	Schwefel 1.2	Mean	Std. Dev.	p Values
CCA	-7.17E+01	1.42E+01	0.108	CCA	7.90E-184	0.00E+00	0.005
CCCA1	-6.09E+01	1.27E+01	0.059	CCCA1	9.85E-210	0.00E+00	0.005
CCCA2	-7.62E+01	1.12E+01	0.074	CCCA2	2.44E-206	0.00E+00	0.005
CCCA3	-7.18E+01	1.23E+01	0.074	CCCA3	2.76E-206	0.00E+00	0.005
CCCA4	-7.59E+01	1.42E+01	0.074	CCCA4	9.80E-210	0.00E+00	0.005
CCCA5	-7.24E+01	1.12E+01	0.074	CCCA5	4.51E-206	0.00E+00	0.005
CCCA6	-7.84E+01	2.04E+01	N/A	CCCA6	8.60E-290	0.00E+00	0.646
CCCA7	-7.64E+01	1.92E+01	1.000	CCCA7	1.14E-292	0.00E+00	N/A
CCCA8	-7.81E+01	1.01E+01	0.110	CCCA8	6.25E-208	0.00E+00	0.005
CCCA9	-6.92E+01	1.22E+01	0.074	CCCA9	2.58E-209	0.00E+00	0.005
CCCA10	-7.50E+01	9.88E+00	0.074	CCCA10	5.54E-206	0.00E+00	0.005

Schwefel 2.22	Mean	Std. Dev.	p Values	Schwefel 2.26	Mean	Std. Dev.	p Values
CCA	1.93E-32	4.96E-32	0.005	CCA	-1.14E+04	1.62E+03	N/A
CCCA1	1.07E-40	2.21E-40	0.241	CCCA1	-1.03E+04	-1.03E+04	0.314
CCCA2	3.31E-41	7.46E-41	N/A	CCCA2	-9.38E+03	2.72E+03	0.074
CCCA3	1.55E-39	4.31E-39	0.059	CCCA3	-9.35E+03	2.26E+03	0.086
CCCA4	5.56E-39	1.74E-38	0.386	CCCA4	-8.58E+03	1.64E+03	0.017
CCCA5	1.04E-40	1.19E-40	0.037	CCCA5	-1.11E+04	1.10E+03	0.386
CCCA6	1.17E+03	2.10E+02	0.005	CCCA6	-5.59E+03	1.30E+02	0.005
CCCA7	1.17E+03	3.80E+02	0.005	CCCA7	-7.21E+03	2.38E+03	0.008
CCCA8	1.15E+03	1.59E+02	0.005	CCCA8	-8.45E+03	1.46E+03	0.011
CCCA9	4.55E-41	6.96E-41	0.203	CCCA9	-8.46E+03	2.15E+03	0.013
CCCA10	1.67E-40	3.95E-40	0.386	CCCA10	-9.88E+03	1.90E+03	0.028

Cigar	Mean	Std. Dev.	p Values	Bohachevsky	Mean	Std. Dev.	p Values
CCA	7.82E-57	1.83E-56	0.005	WOA	-5.55E-17	0.00E+00	N/A
CCCA1	2.49E-71	7.77E-71	0.114	CWOA1	-5.55E-17	0.00E+00	N/A
CCCA2	8.57E-71	2.52E-70	0.059	CWOA2	-5.55E-17	0.00E+00	N/A
CCCA3	4.63E-72	1.28E-71	0.285	CWOA3	-5.55E-17	0.00E+00	N/A
CCCA4	1.68E-72	4.97E-72	0.386	CWOA4	-5.55E-17	0.00E+00	N/A
CCCA5	5.49E-74	1.58E-73	0.575	CWOA5	-5.55E-17	0.00E+00	N/A
CCCA6	1.26E+04	8.88E+03	0.005	CWOA6	1.76E-01	3.09E-01	0.109
CCCA7	1.07E+04	5.56E+03	0.005	CWOA7	1.76E-01	3.09E-01	0.043
CCCA8	4.64E-74	1.26E-73	N/A	CWOA8	-5.55E-17	0.00E+00	N/A
CCCA9	2.59E-73	7.42E-73	0.445	CWOA9	-5.55E-17	6.13E-21	N/A

CCCA10	5.26E-73	1.23E-72	0.575	CWOA10	-5.55E-17	0.00E+00	N/A

Levy	Mean	Std. Dev.	p Values	Alpine	Mean	Std. Dev.	p Values
CCA	2.24E+00	4.58E-01	N/A	CCA	5.11E-31	8.75E-31	0.005
CCCA1	2.37E+00	9.38E-01	0.003	CCCA1	9.74E-39	2.99E-38	0.386
CCCA2	2.41E+00	3.73E-01	0.003	CCCA2	3.30E-39	9.78E-39	0.059
CCCA3	2.37E+00	9.38E-01	0.003	CCCA3	3.30E-39	9.78E-39	0.646
CCCA4	2.54E+00	4.94E-01	0.003	CCCA4	6.17E-40	1.44E-39	0.386
CCCA5	2.63E+00	5.68E-01	0.003	CCCA5	8.91E-42	1.06E-40	N/A
CCCA6	2.41E+00	3.73E-01	0.003	CCCA6	7.22E+03	2.15E+03	0.005
CCCA7	6.25E+01	3.15E+01	0.003	CCCA7	6.76E+03	1.28E+03	0.005
CCCA8	2.60E+00	3.55E-01	0.003	CCCA8	4.34E-39	1.33E-38	0.959
CCCA9	2.45E+00	3.61E-01	0.003	CCCA9	3.95E-40	6.09E-40	0.203
CCCA10	2.43E+00	4.18E-01	0.003	CCCA10	1.08E-39	2.58E-39	0.333

Easom	Mean	Std. Dev.	p Values	Schwefel 2.21	Mean	Std. Dev.	p Values
CCA	-8.76E-01	3.09E-01	N/A	CCA	1.10E-18	3.48E-18	0.005
CCCA1	-4.71E-01	4.99E-01	0.047	CCCA1	2.97E-29	8.70E-29	0.959
CCCA2	-3.33E-01	4.49E-01	0.028	CCCA2	1.11E-31	2.31E-31	0.575
CCCA3	-6.12E-01	4.49E-01	0.139	CCCA3	4.07E-30	7.94E-30	0.017
CCCA4	-6.07E-01	4.32E-01	0.047	CCCA4	1.52E-30	3.93E-30	0.575
CCCA5	-4.58E-01	4.92E-01	0.037	CCCA5	1.42E-28	4.49E-28	0.169
CCCA6	0.00E+00	0.00E+00	0.005	CCCA6	8.31E+01	1.06E+01	0.005
CCCA7	-2.00E-01	4.22E-01	0.028	CCCA7	8.63E+01	7.44E+00	0.005
CCCA8	-3.73E-01	4.84E-01	0.009	CCCA8	1.07E-31	1.46E-31	N/A
CCCA9	-3.70E-01	4.82E-01	0.059	CCCA9	1.11E-30	2.26E-30	0.333
CCCA10	-8.37E-01	3.01E-01	0.241	CCCA10	1.24E-30	2.52E-30	0.445

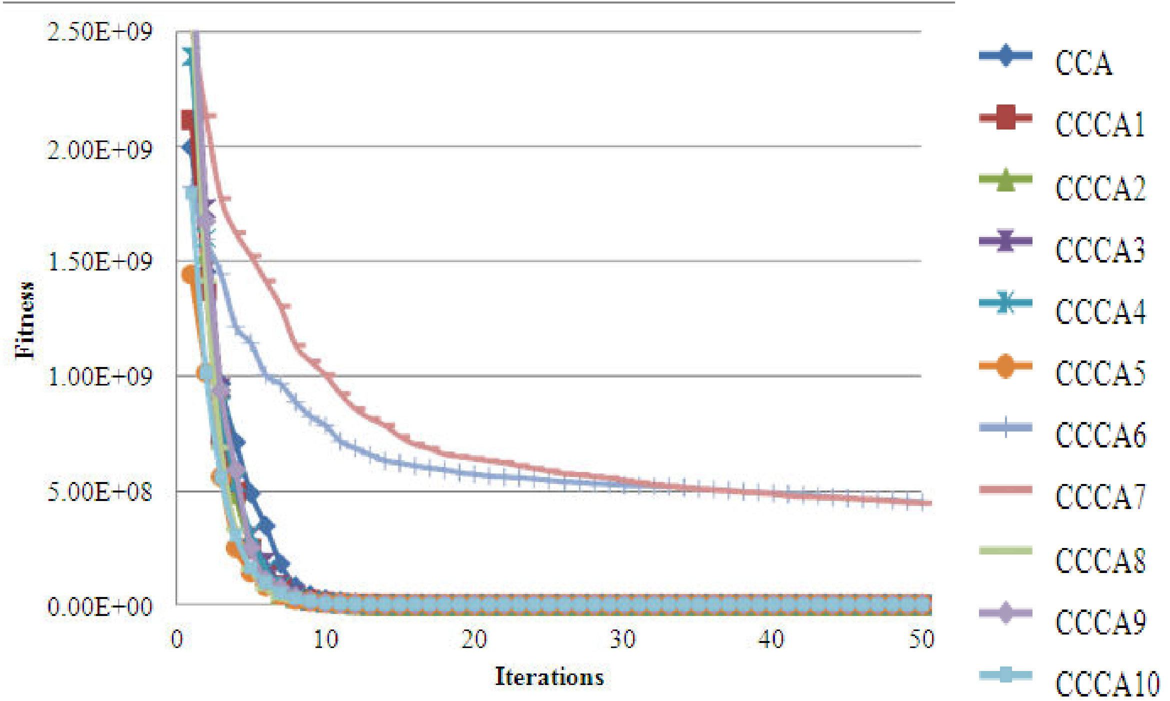


Fig.2. Performance comparisons on the F01 Sphere function.

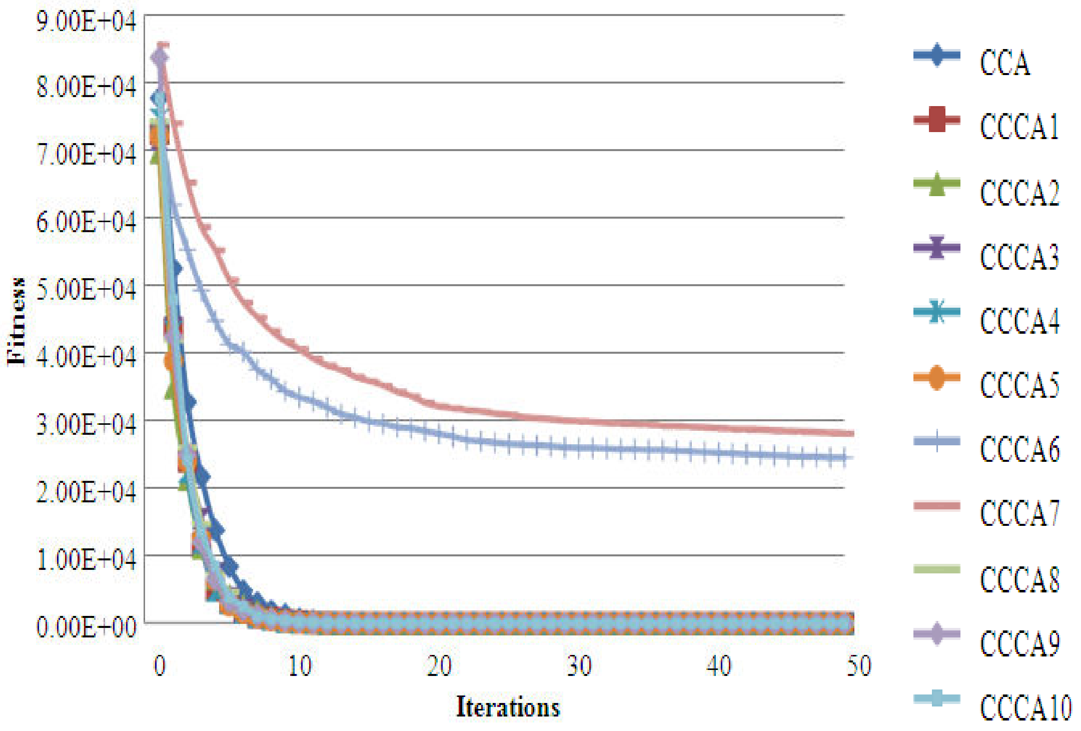


Fig.3. Performance comparison on the F03 Cigar function.

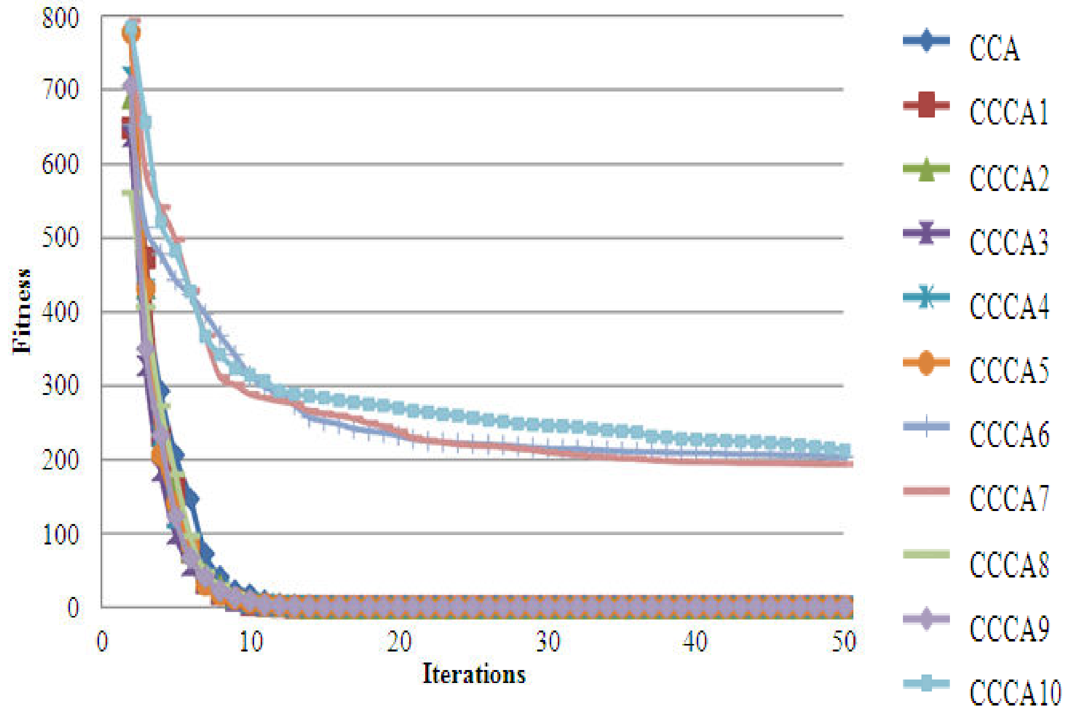


Fig.4. Performance comparison on the F08 Greiwank function.

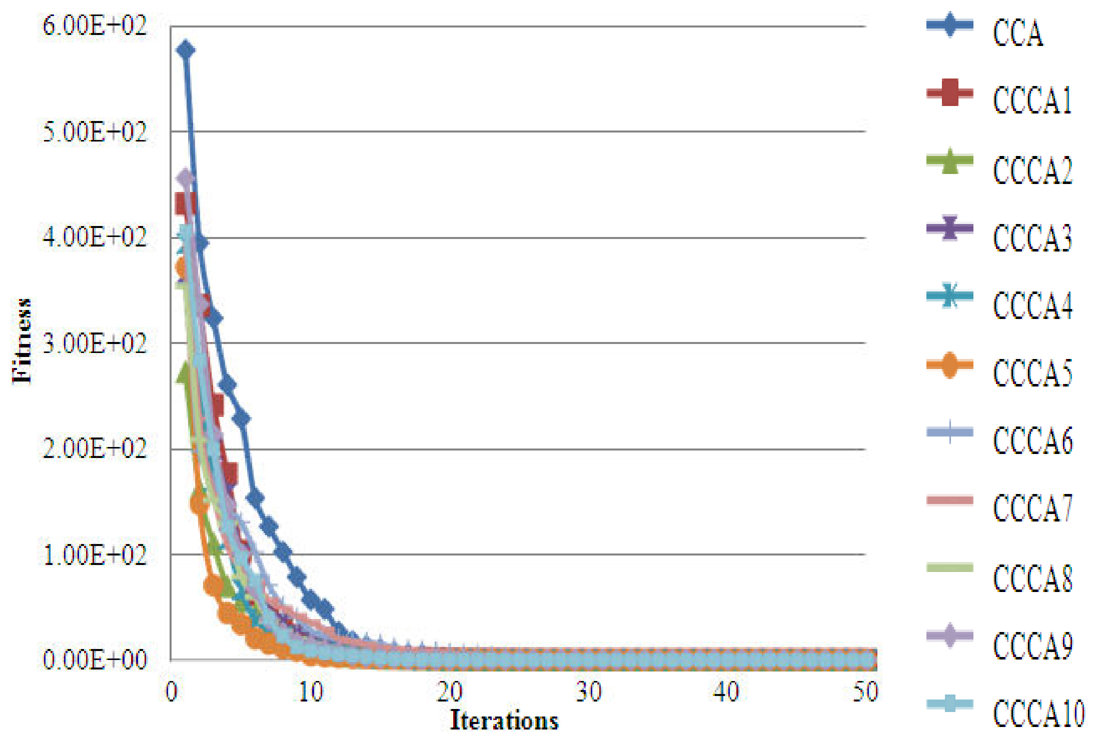


Fig.5. Performance comparison on the F13 Schaffer function.

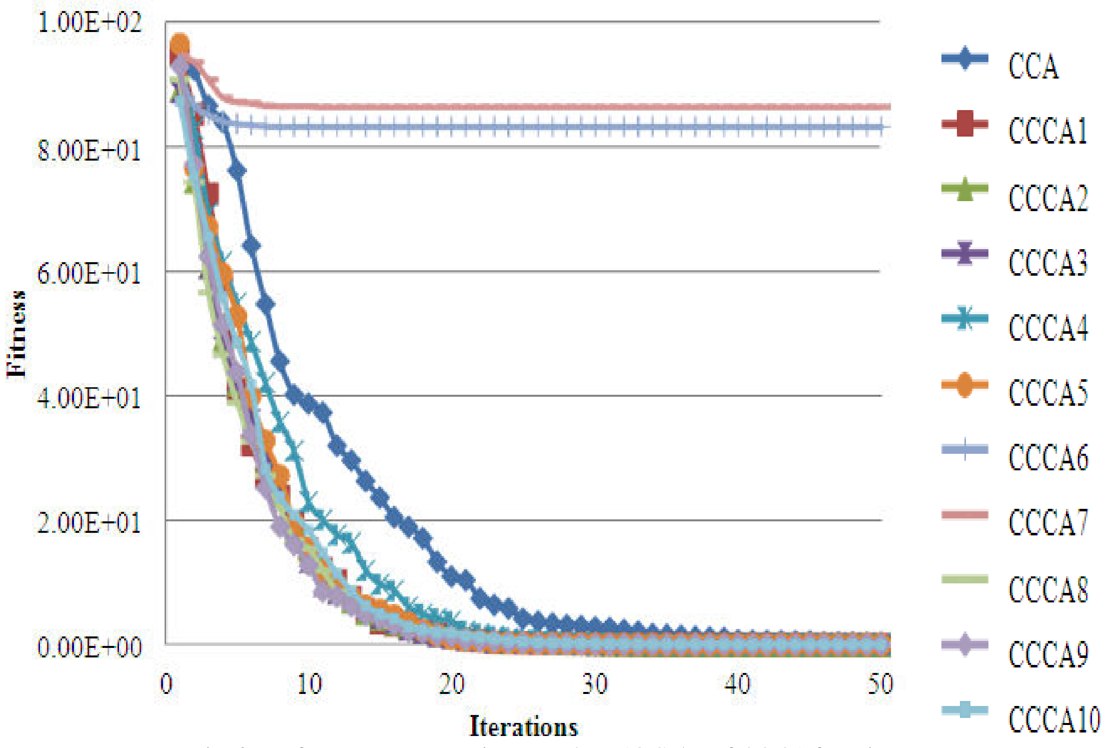


Fig.6. Performance comparison on the F18 Schwefel 2.21 function.

V. CONCLUSIONS AND FUTURE SCOPE

In the presented paper chaos theory and Cheetah Chase Algorithm (CCA) are hybridized in order to design an improved chaotic Cheetah Chase Algorithm (CCCA). To adjust the key parameter, p, of CCA, a wide variety of chaotic maps has been utilized. Twenty benchmark functions dividing into multimodal and unimodal problems have been employed in order to compare and verify the performance of CCCAs. Generally speaking, the results proved that chaotic maps are able to significantly improve the performance of CCA. Among all the chaotic maps, the Tent map has considerably enhanced the performance of CCA. The chaos induced by the chaotic maps in the search space is the main reason behind the superior performance of CCCA. The chaos helps the controlling parameter to find the optimal solution more quickly and thus refine the convergence rate of the algorithm. For future work, it would be interesting to employ CCCA algorithm for solving real-world engineering problems.

VI. REFERENCES

[1] Alatas, B. (2010). Chaotic harmony search algorithms. Applied Mathematics and Computation, 216(9), 2687-2699.
[2] Alatas, B. (2010). Chaotic bee colony algorithms for global numerical optimization. Expert Systems with Applications, 37(8), 5682-5687.
[3] Alba, E., & Dorronsoro, B. (2005). The exploration/exploitation tradeoff in dynamic cellular genetic algorithms. IEEE Transactions on Evolutionary Computation, 9(2), 126-142.
[4] Aljarah, I., Faris, H., & Mirjalili, S. (2016). Optimizing connection weights in neural networks using the whale optimization algorithm. Soft Computing, 1-15.
[5] Arora,Singh,(2015).Butterfly algorithm with Heavy Flights for global optimization. In Signal Processing, Computing and Control (ISPCC), 2015 International Conference on (pp. 220-224). IEEE.
[6] Arora, S., &Singh, S. (2017). An improved butterfly optimization algorithm with chaos. Journal of Intelligent & Fuzzy Systems, 32(1), 1079-1088.
[7] Arora, S., & Singh, S. (2017). Node Localization in Wireless Sensor Networks Using Butterfly Optimization Algorithm. Arabian Journal for Science and Engineering, 1-11.
[8] Coello, C. A. C. (2000). Use of a self-adaptive penalty approach for engineering optimization problems. Computers in Industry, 41(2), 113-127.
[9] Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation, 1(1), 3-18.
[10] Digalakis, J. G., & Margaritis, K. G. (2001). On benchmarking functions for genetic algorithms. International journal of computer mathematics, 77(4), 481-506.
[11] Dorigo, M., & Di Caro, G. (1999). Ant colony optimization: a new meta-heuristic. In Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on (Vol. 2, pp. 1470-1477). IEEE.
[12] Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactions on evolutionary computation, 1(1), 53-66.
[13] Dos Santos Coelho, L., & Mariani, V. C. (2008). Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization. Expert Systems with Applications, 34(3), 1905-1913.
[14] Eglese, R. W. (1990). Simulated annealing: a tool for operational research.European journal of operational research, 46(3), 271-281.
[15] Eberhart, R., & Kennedy, J. (1995). Particle swarm optimization, in Proceedings of IEEE International Conference on Neural Networks, 1995, pp. 1942-1948
[16] Geem, Z. W., Kim, J. H., & Loganathan, G. V. (2001). A new heuristic optimization algorithm: harmony search. Simulation, 76(2), 60-68.
[17] Gandomi, A. H., Yang, X. S., Talatahari, S., & Alavi, A. H. (2013). Firefly algorithm with chaos. Communications in Nonlinear Science and Numerical Simulation, 18(1), 89-98.
[18] Gandomi, A. H., & Alavi, A. H. (2012). Krill herd: a new bio-inspired optimization algorithm. Communications in Nonlinear Science and Numerical Simulation, 17(12), 4831-4845.
[19] Gandomi, A. H., Yang, X. S., Alavi, A. H., & Talatahari, S. (2013). Bat algorithm for constrained optimization tasks. Neural Computing and Applications, 22(6), 1239-1255.
[20] Gandomi, A. H., & Yang, X. S. (2014). Chaotic bat algorithm. Journal of Computational Science, 5(2), 224-232.
[21] Gao, X. Z., Wang, X., Ovaska, S. J., & Xu, H. (2010). A modified harmony search method in constrained optimization. International Journal of Innovative Computing, Information and Control, 6(9), 4235-4247.
[22] García, S., Molina, D., Lozano, M., & Herrera, F. (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization.Journal of Heuristics, 15(6), 617-644.
[23] Goudhaman.M, "Cheetah Chase Algorithm (CCA): A Nature inspired metaheuristic algorithm" IJET, International Journal of Engineering & Technology, VOLUME 7, ISSUE 3, January/2018 PAGE NO: 1804-1811
[24] Goudhaman.M, Vanathi. N, Sasikumar.S, (2018), Semantic Approach for Dynamic Shortest Path Problem (SPP) By Cheetah Chase Algorithm (CCA), IAETSD Journal For Advanced Research In Applied Sciences Volume 5, Issue 8, August/2018 ISSN NO: 2394-8442, PAGE NO:229-237, UGC Indexed Journal - August 2018.
[25] https://cheetah.org/about-the-cheetah/
[26] http://c21.phas.ubc.ca/article/cheetah-chase
[27] Karaboga, D., & Akay, B. (2009). A comparative study of artificial bee colony algorithm. Applied mathematics and computation, 214(1), 108-132.
[28] Kaveh, A. (2017). Sizing Optimization of Skeletal Structures Using the Enhanced Whale Optimization Algorithm. In Applications of Metaheuristic Optimization Algorithms in Civil Engineering (pp. 47-69). Springer International Publishing.
[29] Kellert, S. H. (1994). In the wake of chaos: Unpredictable order in dynamical systems. University of Chicago press.
[30] Kennedy, J. (2011). Particle swarm optimization. In Encyclopedia of machine learning (pp. 760- 766). Springer US.
[31] Kohli, M., & Arora, S. (2017). Chaotic grey wolf optimization algorithm for constrained optimization problems. Journal of Computational Design and Engineering.

[32] Liang, J. J., Qin, A. K., Suganthan, P. N., & Baskar, S. (2006). Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE transactions on evolutionary computation, 10(3), 281-295.
[33] Li-Jiang, Y.,& Tian-Lun, C. (2002). Application of chaos in genetic algorithms. Communications in Theoretical Physics, 38(2), 168.
[34] Liu, B., Wang, L., Jin, Y. H., Tang, F., & Huang, D. X. (2005). Improved particle swarm optimization combined with chaos. Chaos, Solitons & Fractals, 25(5), 1261-1271.
[35] Mirjalili, S., & Lewis, A. (2013). S-shaped versus V-shaped transfer functions for binary particle swarm optimization. Swarm and Evolutionary Computation,9, 1-14.
[36] Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. Advances in Engineering Software, 69, 46-61.
[37] Mirjalili, S. (2015). The ant lion optimizer. Advances in Engineering Software,
[38] Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. Advances in Engineering Software, 95, 51-67.
[39] Mafarja, M. M., & Mirjalili, S. (2017). Hybrid Whale Optimization Algorithm with Simulated Annealing for Feature Selection. Neurocomputing.
[40] Pecora, L. M., & Carroll, T. L. (1990). Synchronization in chaotic systems.Physical review letters, 64(8), 821.
[41] Tsai, P. W., Zhang, J., Zhang, S., Istanda, V., Liao, L. C., & Pan, J. S. (2015). Improving swarm intelligence accuracy with cosine functions for evolved bat algorithm. Journal of Information Hiding and Multimedia Signal Processing, 6, 1194-1202.
[42] Reddy, P. D. P., Reddy, V. V., & Manohar, T. G. (2017). Whale optimization algorithm for optimal sizing of renewable resources for loss reduction in distribution systems.
[43] Renewables: Wind, Water, and Solar, 4(1), 3.
[44] Saremi, S., Mirjalili, S., & Lewis, A. (2014). Biogeography-based optimisation with chaos. Neural Computing and Applications, 25(5), 1077-1097.
[45] Simon, D. (2008). Biogeography-based optimization. IEEE transactions on evolutionary computation, 12(6), 702-713.
[46] Sivanandam, S. N., & Deepa, S. N. (2007). Principles of Soft Computing. John Wiley & Sons.
[47] Shu-Chuan C., Pei-Wei T. and Jeng-Shyang P. (2006), "Cat warm Optimization", 9th Pacific Rim International Conference on Artificial Intelligence, LNAI 4099, pp. 854-858.
[48] Storn, R., & Price, K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, 11(4), 341-359.
[49] Talatahari, S., Azar, B. F., Sheikholeslami, R., & Gandomi, A. H. (2012). Imperialist competitive algorithm combined with chaos for global optimization.Communications in Nonlinear Science and Numerical Simulation, 17(3), 1312-1319.
[50] Wang, G. G., Guo, L., Gandomi, A. H., Hao, G. S., & Wang, H. (2014). Chaotic krill herd algorithm. Information Sciences, 274, 17-34.
[51] Wilcoxon, F. (1945). Individual comparisons by ranking methods. Biometrics bulletin, 1(6), 80- 83.
[52] Yang, D., Li, G., & Cheng, G. (2007). On the efficiency of chaos optimization algorithms for global optimization. Chaos, Solitons & Fractals, 34(4), 1366-1375.
[53] Yang, X. S. (2010). Nature-inspired metaheuristic algorithms. Luniver press.
[54] Yang, X. S. (2010). Firefly algorithm, Levy flights and global optimization. InResearch and development in intelligent systems XXVI (pp. 209-218). Springer London.
[55] Yang, X. S. (2010). A new metaheuristic bat-inspired algorithm. Nature inspired cooperative strategies for optimization (NICSO 2010), 65-74.
[56] Yang, D., Li, G., & Cheng, G. (2007). On the efficiency of chaos optimization algorithms for global optimization. Chaos, Solitons & Fractals, 34(4), 1366-1375.
[57] Yao, X., Liu, Y., & Lin, G. (1999). Evolutionary programming made faster. IEEE Transactions on Evolutionary computation, 3(2), 82-102.