ISSN 22773061

A New Compound Swarm Intelligence Algorithms for Solving Global Optimization Problems

Ibrahim M. Hezam[1] Osama Abdel Raouf[2] Mohey M. Hadhoud[3]
[1]Department of Mathematics& computer, Faculty of Education, Ibb University,Yemen.
E-mail: Ibrahizam.math@gmail.com
[2]Department of Operations Research, Faculty of Computers & Information, Minufiya University, Egypt.
E-mail: osamaabd@hotmail.com
[3]Department of Information Technology, Faculty of Computers & Information, Minufiya University, Egypt.
E-mail: mmhadhoud@yahoo.com

ABSTRACT

This paper proposes a new hybrid swarm intelligence algorithm that encompasses the feature of three major swarm algorithms. It combines the fast convergence of the Cuckoo Search (CS), the dynamic root change of the Firefly Algorithm (FA), and the continuous position update of the Particle Swarm Optimization (PSO). The Compound Swarm Intelligence Algorithm (CSIA) will be used to solve a set of standard benchmark functions. The research study compares the performance of CSIA with that of CS, FA, and PSO, using the same set of benchmark functions. The comparison aims to test if the performance of CSIA is Competitive to that of the CS, FA, and PSO algorithms denoting the solution results of the benchmark functions.

Keywords: Hybrid Swarm Intelligence, Cuckoo Search, Firefly Algorithm, Particle Swarm Optimization, Global optimization.

Council for Innovative Research
Peer Review Research Publishing System
Journal: INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY
Vol 10, No 9.
editor@cirworld.com
www.cirworld.com, member.cirworld.com

ISSN 22773061

1. INTRODUCTION

Global optimization is an important task in most scientific and engineering problems. In global optimization problem, it is difficult to obtain an optimal solution due to time complexity. Accordingly Metaheuristics became important in such situations for their independence on a specific problem. Metaheuristics are optimization approaches which make use of the best solution improved iteratively to the next search. For example, finding optimal solutions for nonlinear, non-differentiable fractional objective functions is very difficult to deal with. The complexity of these problems makes it impossible to search for all possible optimal exact solutions. So is the search for any solution near the optimal solution. The metaheuristic optimization is the best for finding near optimal. Recently growing popularity the hybridization of different algorithmic concepts has been to obtain better performing systems that exploit and combine the advantages of the individual pure strategies, that is, hybrids are believed to benefit from synergy. In fact, choosing an adequate combination of multiple algorithmic concepts is often the key to achieving top performance in solving many hard optimization problems. [6] combined two nature inspired algorithms and introduced the CS/PSO algorithm. Cuckoo birds are aware of each other positions and make use of swarm intelligence in PSO in order to reach for better solutions. [15] combined the differential evolution (DE) and cuckoo search (CS) algorithm to solve the uninhabited combat air vehicle UCAV path planning problem. DE is applied to optimize the process of selecting cuckoo of the CS model during the process of cuckoo in nest updating. [14] proposed hybrid optimization algorithm of PSO and CS. By CS-PSO, the search area of PSO was extended, and the defect of PSO is easily fall into point of local extremum that was improved. [13] proposed a hybrid algorithm which combines the merits of Ant Colony Optimization (ACO) and Cuckoo Search for Job scheduling. The major problem in the ACO is that, the ant will walk through the path where the chemical substances called pheromone is deposited. This acts as if it lures the artificial ants. Cuckoo search can perform the local search more efficiently and there is only a single parameter apart from the population size. It minimizes the makespan and the scheduling can be used in scientific computing and high power computing. [7] introduced a modify firefly algorithm and use this algorithm with cellular learning automata. [1] combines the standard Firefly Algorithm (FA) with the evolutionary operations of Differential Evolution (DE) method to improve the searching accuracy and information sharing among the fireflies. [3] proposed ant colony optimization (ACO) and firefly algorithm (FFA) algorithm for constrained optimization problems, The methodology of the proposed algorithm is introduced based on a parallel mechanism of ACO and FFA for updating the solutions of ACO-FFA. [4] presents the evolutionary hybrid genetic-firefly algorithm for the optimization of complex problems and to search global solution more precisely.

The purpose of this paper is to encompound the search features of three different metaheuristic algorithms, cuckoo search (CS), firefly algorithm (FA) and Particle Swarm Optimization (PSO). Where the cuckoo birds experience new places (random walk) utilizing firefly algorithm strategy instead of Lévy flight. In the proposed algorithm cuckoo birds will also be aware of each other positions utilizing PSO swarm communication technique to search for a better solution.

The remainder of this paper is organized as follows. Section 2; introduce the standard cuckoo search. The standard firefly algorithm is reviewed in section 3. In Section 4, introduces the standard particle swarm optimization. A Compound Swarm Intelligence Algorithm (CSIA) is presented in Section 5. In section 6, Benchmark functions with discussion are introduced. Finally, Section 7 is the concluding part of the paper.

2. CUCKOO SEARCH ALGORITHM (CS) [5, 9, 10, 17]

The Cuckoo search algorithm is a Meta heuristic search algorithm which has been proposed recently by Yang and Deb (2009) and it was based on the following idealized rules:

• Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest.

• The best nests with high quality of eggs (solutions) will carry over to the next generations.

• The number of available host nests is fixed, and a host can discover an alien egg with a probability $p_a \in [0,1]$. In this case, the host bird can either throw the egg away or abandon the nest so as to build a completely new nest in a new location.

Cuckoo search algorithm

Begin

Objective function $f(x), x = (x_1, x_2, ..., x_d)^T$;

Initial a population of n host nests $x_i (i = 1, 2, ..., d)$

while (t < MaxGeneration) or (stop criterion);

Get a cuckoo (say i) randomly

and generate a new solution by Lévy flights;

Evaluate its quality/fitness; F_i

Choose a nest among n (say j) randomly;

if ($F_i > F_j$),

Replace j by the new solution;

end

Abandon a fraction (P_a) of worse nests

[and build new ones at new locations via Lévy flights];

Keep the best solutions (or nests with quality solutions);

Rank the solutions and find the current best;

end while

Post process results and visualization;

End

when generating new solutions $x_i(t+1)$ for the ith cuckoo, the following Lévy flight is performed

$$x_i^{(t+1)} = x_i^{(t)} + \alpha \oplus Levy(\lambda) \quad (2)$$

where $\alpha > 0$ is the step size, which should be related to the scale of the problem of interest. The product $\oplus$ means entry-wise multiplications. In this research work, we consider a Lévy flight in which the step-lengths are distributed according to the following probability distribution

$$Levy u = t^{-\lambda}, 1 < \lambda \leq 3$$

which has an infinite variance. Here the consecutive jumps/steps of a cuckoo essentially form a random walk process which obeys a power-law step length distribution with a heavy tail.

3. FIREFLY ALGORITHM (FA) [11, 17, 16]

The Firefly Algorithm was developed by Yang (2008) and it was based on the following idealized behavior of the flashing characteristics of fireflies:

• All fireflies are unisex so that one firefly is attracted to other fireflies regardless of their sex;

• Attractiveness is proportional to their brightness, thus for any two flashing fireflies, the less bright one will move towards the brighter one. The attractiveness is proportional to the brightness and they both decrease as their distance increases. If no one is brighter than a particular firefly, it moves randomly;

• The brightness or the light intensity of a firefly is affected or determined by the landscape of the objective function to be optimized.

The operation of the Firefly Algorithm is as follows:

Step 1: Define Objective function $f(x), x = (x_1, x_2, ..., x_d)^T$; and Generate initial population of fireflies placed at random positions within the n-dimensional search space, x_i. Define the light absorption coefficient γ.

Step 2: Define the Light Intensity of each firefly, L_i, as the value of the cost function for x_i.

Step 3: For each firefly, x_i, the light Intensity, L_i, is compared for every firefly x_j $j \in \{1, 2, ..., d\}$

Step 4: If L_i is less than L_j, then move firefly x_i towards x_j in n-dimensions. The value of attractiveness between flies varies relatively the distance *r* between them:

$$x_i^{t+1} = x_i^t + \beta_\circ e^{-\gamma r_{ij}^2}(x_j^t - x_i^t) + \alpha \varepsilon_i^t \quad (3)$$

where $\beta_\circ$ is the attractiveness at *r=0* the second term is due to the attraction, while the third term is randomization with the vector of random variables εi being drawn from a Gaussian distribution. $\alpha \in [0,1]$. The distance between any two fireflies *i* and *j* at x_i *and* x_j can be regarded as the Cartesian distance $r_{ij} = \|x_i - x_j\|_2$.

Step 5: Calculate the new values of the cost function for each fly, x_i, and update the Light Intensity, *Li*.

Step 6: Rank the fireflies and determine the current 'best'.

Step 7: Repeat Steps 2 to 6 until definite termination conditions are met, such as a pre-defined number of iterations or a failure to make progress for a fixed number of iterations.

4. PARTICLE SWARM OPTIMIZATION (PSO) [2, 8, 11, 12]

Particle swarm optimization is a population based stochastic optimization technique developed by Eberhart and Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling.

The characteristics of PSO can be represented as follows:

x_i^k The current position of the particle *i* at iteration *k*; v_i^k The current velocity of the particle *i* at iteration *k*;

y_i^k The personal best position of the particle *i* at iteration *k*; $\tilde{y}_i^k$ The neighborhood best position of the particle.

The velocity update step is specified for each dimension $j \in 1,\ldots,N_d$, hence, $v_{i,j}$ represents the *jth* element of the velocity vector of the ith particle. Thus, the velocity of particle *i* is updated using the following equation:

$$v_i^k(t+1) = wv_i^k(t) + c_1 r_1(t)(y_i(t) - x_i(t)) + c_2 r_2(t)(\tilde{y}_i(t) - x_i(t)) \quad (4)$$

where *w* is weighting function, $c_{1,2}$ are weighting coefficients, $r_{i,2}(t)$ are random numbers between 0 and 1. The current position (searching point in the solution space) can be modified by the following equation:

$$x_i^{k+1} = x_i^k + v_i^{k+1} \quad (5)$$

The detailed operation of particle swarm optimization is given below:

Step 1: Initialize parameters and population.

Step 2: Initialization. Randomly set the position and velocity of all particles, within pre-defined ranges. And on D dimensions in the feasible space (i.e.it satisfies all the constraints)

Step 3: Velocity Updating. At each iteration, velocities of all particles are updated according to equation (4). After updating, v_i^k should be checked and maintained within a pre-specified range to avoid aggressive random walking.

Step 4: Position Updating. Assuming a unit time interval between successive iterations, the positions of all particles are updated according to equation (5). After updating, x_i^k should be checked and limited within the allowed range.

Step 5: Memory updating. Update y_i^k and $\tilde{y}_i^k$ when condition is met.

$$y_i^k(t+1) = \begin{cases} y_i^k(t) & \text{if } f\left(x_i^k(t+1)\right) \geq f\left(y_i^k(t)\right) \\ x_i^k(t+1) & \text{if } f\left(x_i^k(t+1)\right) < f\left(y_i^k(t)\right) \end{cases} \quad (6)$$

where *f(x)* is the objective function subject to maximization.

Step 6: Termination Checking. Repeat Steps 2 to 4 until definite termination conditions are met, such as a pre-defined number of iterations or a failure to make progress for a fixed number of iterations.

5. THE PROPOSED ALGORITHM

In this proposed algorithm, cuckoo bird will be able to perform stochastic behaviour (random walk) using the strategy of firefly algorithm according to equation (3) instead of using Lévy Flight movement. Also the cuckoo birds will be able to communicate them to inform each other from their position and help each other to emigrate to a better place. Each cuckoo bird will record the best personal experience as pbest during its own life. In addition, the best pbest among all the birds is called gbest. The cuckoo birds' communication is established through the pbest and gbest. They update their position using these parameters along with the velocity of each swarm member. The update rule for cuckoo (i's) position is carried out according to equations (4,5).

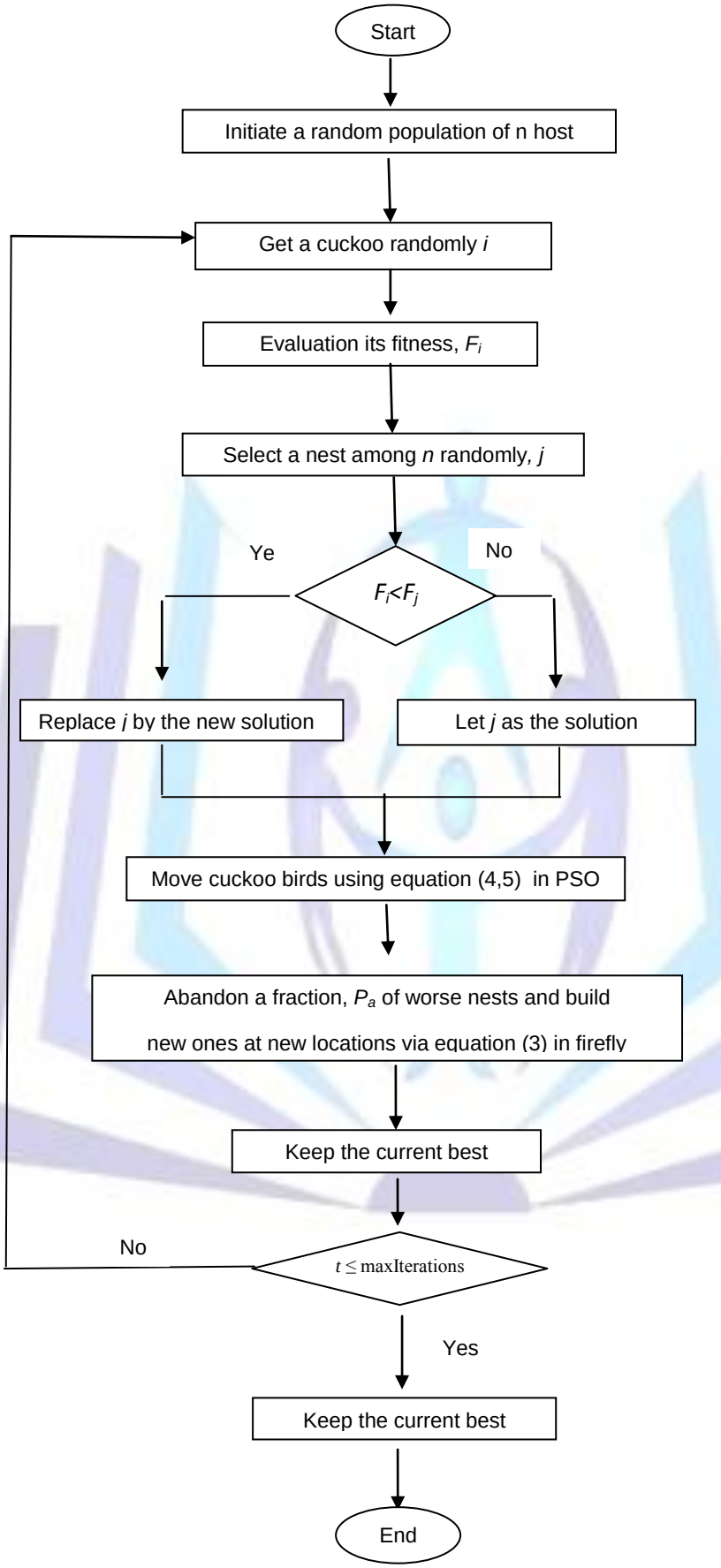


Fig. (1). Flowchart of The Compound Swarm Intelligence Algorithm (CSIA).

The pseudo-code of the CSIA is presented as bellow:

Begin

Objective function $f(x), x = (x_1, x_2, ..., x_d)^T$;

Initial a population of n host nests $x_i (i = 1, 2, ..., d)$

while (t < MaxGeneration) or (stop criterion);

Get a cuckoo (say *i*) randomly

and generate a new solution ***by equation (3)in firefly algorithm;***

Evaluate its quality/fitness; F_i

Choose a nest among *n* (say *j*) randomly;

if $(F_i > F_j)$,

Replace *j* by the new solution;

end

Move cuckoo birds using equation (4) and (5);

Abandon a fraction (P_a) of worse nests

[and build new ones at new locations ***via equation (3)in firefly algorithm***;

Keep the best solutions (or nests with quality solutions);

Rank the solutions and find the current best;

end while

Post process results and visualization;

End

6. ILLUSTRATIVE TEST FUNCTIONS WITH DISCUSSION

The following ten benchmark functions which are shown in table (1) were collected from literature to demonstrate the efficiency and robustness of the proposed algorithms. The numerical results of the proposed algorithms are compared with the traditional cuckoo search algorithm, firefly algorithm, and particle swarm optimization illustrated in Tables (2). The simulation parameter settings results of CS,FA,and PSO which are the same used for CSAI algorithm are as follows:

CS	number of nests *n=50*, Discovery rate of alien eggs/solutions p_a=0. 25;
FA	population size : 50, *α* (randomness): 0.25, minimum value of *β: 0.20, γ* (absorption): 1.0
PSO	population size of 50, the inertia weight was set to change from 0.9 (wmax) to 0.4 (warming) over the iterations. Set c_1 *:0.12* and c_2 *:1.2.*

The algorithms have been implemented by MATLAB R2011 on core (TM) i3 to 2.27 GHz processor.

Table (1). The Benchmark functions

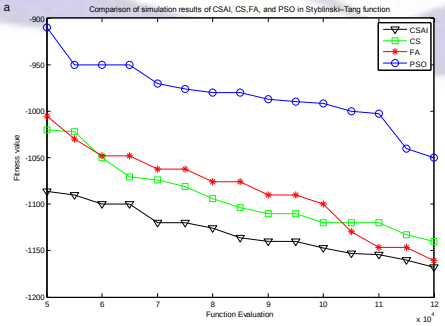
Function	Formulation	Dimention	f_{min}	Range
F_1 (Styblinski–Tang function)	$\frac{\sum_{i=1}^{n} x_i^4 - 16x_i^2 + 5x_i}{2}$	30	-39.16599D	$[-5,5]^D$
F_2(Powell function)	$\sum_{i=1}^{n/k} (x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} + x_{4i})^2 + (x_{4i-2} + x_{4i-1})^4 + 10(x_{4i-3} + 10x_{4i})^4$	24	0	$[-4,5]^D$

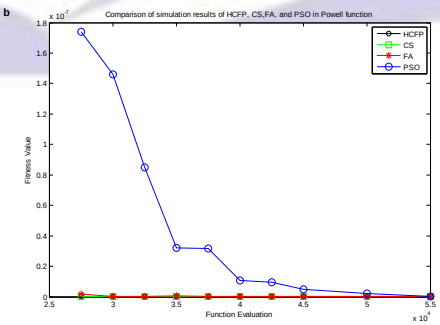
F_3(Schaffer function N:2)	$0.5+\frac{\sin\left(\sin\left(x_1^2+x_2^2\right)\right)-0.5}{\left(1+0.001\left(x_1^2+x_2^2\right)\right)^2}$	2	0	[-100,100]
F_4(Schaffer function N:4)	$0.5+\frac{\cos\left(\sin\left\|x_1^2+x_2^2\right\|\right)-0.5}{\left(1+0.001\left(x_1^2+x_2^2\right)\right)^2}$	2	0.292579	[-100,100]
F_5(Drop-wave function)	$\frac{1+\cos\left(12\sqrt{x_1^2+x_2^2}\right)}{2+0.5\left(x_1^2+x_2^2\right)}$	2	-1	[-5.12,5.12]
F_6(De-Jong function N:5)	$\left(0.002+\sum_{i=1}^{25}\frac{1}{i+\left(x_1-a_{1i}\right)^6+\left(x_2-a_{2i}\right)^6}\right)$ $a=\begin{pmatrix}-32 & -16 & 0 & 16 & 32 & -32\ldots 16 & 32\\ -32 & -32 & -32 & -32 & -16\ldots 32 & 32\end{pmatrix}$	2	1	[-65.536,65.536]
F_7(Gear function)	$\left(\frac{1}{6.931}-\frac{\lfloor x_1\rfloor\lfloor x_2\rfloor}{\lfloor x_3\rfloor\lfloor x_4\rfloor}\right)$	4	2.7E-012	$[12,60]^D$
F_8(Pathological function)	$\sum_{i=1}^{n-1}\frac{\sin^2\left(\sqrt{x_{i+1}^2+x_i^2}\right)-0.5}{0.001\left(x_i-x_{i+1}\right)^4+0.5}$	2 20	-1.9960079	$[-100,100]^D$
F_9(SineEnvelope function)	$-\sum_{i=1}^{n-1}\left[\frac{\sin^2\left(\sqrt{x_{i+1}^2+x_i^2}-0.5\right)}{\left(0.001\left(x_{i+1}^2+x_i^2\right)+1\right)^2}+0.5\right]$	2 20	0	$[-100,100]^D$
F_{10}(Shekel function)	$\sum_{i=1}^{m}\frac{1}{c_i+\sum_{j=1}^{4}\left(x_j-a_{ji}\right)^2}$	4 m=10	-10.5364	$[10,10]^D$

Table (2). Result comparisons on 10 test functions.

Function		CS	FA	PSO	CSAI
F_1 (Styblinski–Tang function)	Best	-1.14E+003	-1.161E+003	-1.050E+003	**-1.168E+003**
	Mean	-1.1E+003	-1.085E+003	-9.82E+002	**-1.129E+003**
	StdDev.	37.66E+00	45.68E+00	35.24E+002	26.02E+002
	Time(s)	63	59.9	5.4	66.9
F_2(Powell function)	Best	3.99E-013	5.27E-013	1.74E-010	**1.67E-013**
	Mean	2.44E-011	2.98E-010	4.9E-008	**2.27E-011**
	StdDev.	3.47E-011	5.40E-010	6.36E-008	2.758E-011
	Time(s)	112	83.3	61.2	129
F_3(Schaffer function N:2)	Best	**0.00E+00**	6.25E-013	2.68E-012	**0.00E+00**
	Mean	1.42E-011	5.27E-011	5.33E-011	**0.00E+00**
	StdDev.	3.06E-011	8.81E-011	5.19E-011	0.00E+00
	Time(s)	116	243	24	143

F_4(Schaffer function N:4)	Best	0.5000009	0.500001	**0. 500E+00**	**0. 500E+00**
	Mean	0.500037	0.500044	0.500011	**0.500E+00**
	StdDev.	3.29E-005	3.26E-005	2.82E-005	0.00E+00
	Time(s)	126	201	111	155
F_5(Drop-Wave function)	Best	**-1.00E+00**	-0.999999999	-0.9999995	**-1.00E+00**
	Mean	-0.99999992	-0.99999997	-0.99997	**-1.00E+00**
	StdDev.	1.9E-007	2.19E-008	3.02E-005	0.00E+00
	Time(s)	100	118	23	158
F_6(De-Jong function N:5)	Best	**0.998003838**	0.9980038378	12.6705058	**0.9980038378**
	Mean	1.044446848	1.143360007	12.6705058	**0.998003838**
	StdDev.	1.1E-002	3.3E-002	2.22E-011	4.3E-010
	Time(s)	96	136	19	130
F_7(Gear function)	Best	**2.7E-012**	**2.7E-012**	2.3E-011	**2.7E-012**
	Mean	1.04E-010	1.2E-010	1.12E-009	**2.7E-012**
	StdDev.	2.7E-010	3.02E-010	9.31E-010	0.00E+00
	Time(s)	65	137	30	95
F_8(Pathological function)	Best	**-6.01E+00**	-7.2E-001	-1.4E-003	**-6.02E+00**
	Mean	-5.77E+00	-1.33E-001	-3.633E-004	**-5.89E+00**
	StdDev.	2.2E-001	3.24E-001	7.21E-004	7.06E-002
	Time(s)	154	137	28	199
F_9(SineEnvelop e function)	Best	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Mean	**1.41E-004**	3.78E-004	5.57E-004	1.65E-004
	StdDev.	2.54E-004	3.99E-004	3.65E-004	3.12E-004
	Time(s)	5	8	3.5	10
F_{10}(Shekel function)	Best	--10.5364098	-10.53640	-10.53429	**-10.5364E+00**
	Mean	-10.536322	-10.5363872	-10.530721	**-10.5364E+00**
	StdDev.	1.77E-004	9.27E-002	4.08E-005	0.00E-00
	Time(s)	124	60	64	95





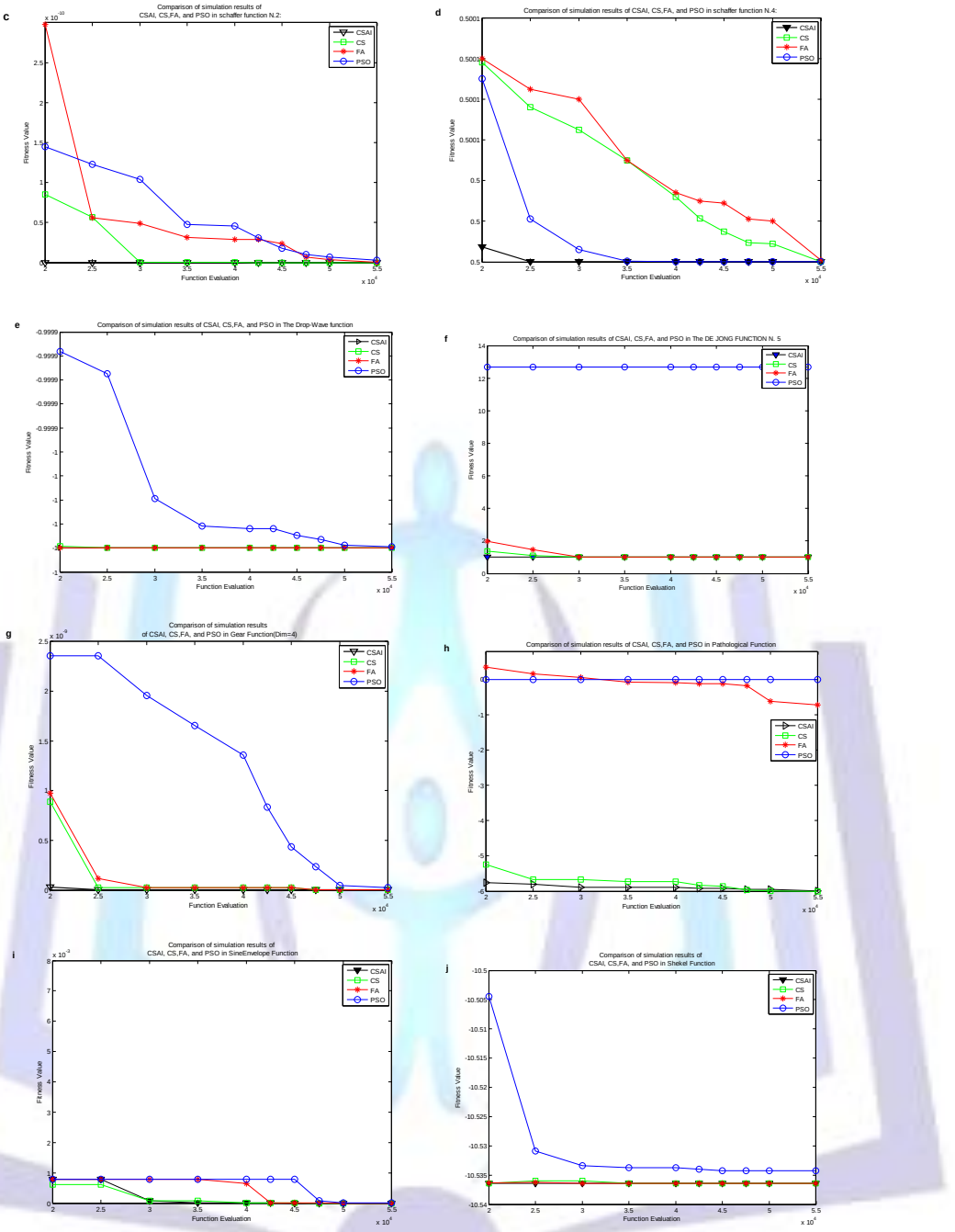


Fig.(2) Convergence performance on the 10 test functions. (a) f_1, (b) f_2, (c) f_3, (d) f_4, (e) f_5, (f) f_6, (g) f_7, (h) f_8, (i) f_9, (j) f_{10}.

ISSN 22773061

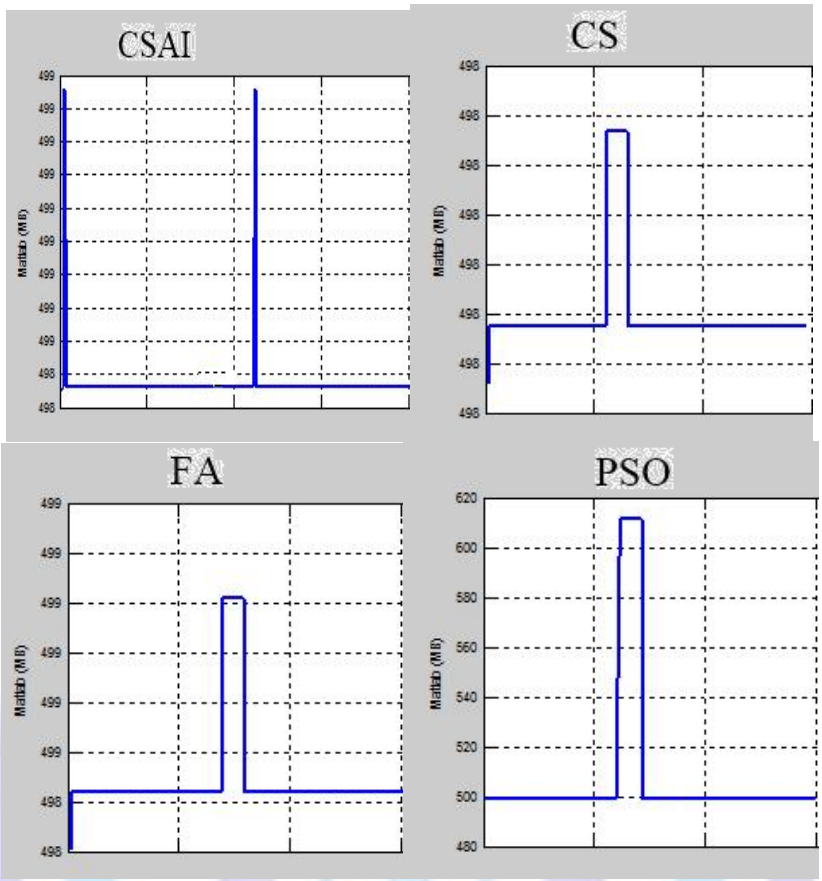


Fig.(3) Memory usage indicator

The detailed accuracy performance concerning the solution of CSAI, CS, FA, and PSO listed in Table (2), is measured in terms of the best, mean, and standard deviation values of the solutions obtained by 20 independent runs. A comparative study was carried out between the three scoped swarm intelligence algorithms along with the proposed CSAI algorithm. The comparison is based on three different measures; the obtained optimization value, the convergence time and the amount addressed memory resources. First refereeing to the obtained optimization value the proposed CSAI algorithm managed to explore new solution areas that benchmark problem results using metaheuristic algorithm couldn't reach that could be clearly noticed from f_8 in table(2) when an optimal value of -6.02 was reached while all other metaheuristic technique in table(1) gave an optimal of -1.99. The optimization value results for the rest functions indicates a better achievement for CSAI algorithm. Boldface figures in the table indicates the best result(s) among the algorithms.

Unfortunately, the convergence time of the CSAI was not such an achievement since it could be noticed from table (2) that it take a longer time to converge compared to the three rest algorithms. This may be referred to the time consumed in the communication between swarm member using the PSO technique.

Finally, comparing the four algorithms according to memory usage reveals that the three algorithms CS,FA, and CSAI almost utilizes the same memory amount of 498-499 as shown in fig(3) while PSO algorithm takes an amount of 615 as shown in the same fig. That is to say that the modifications in the proposed algorithm did not affect memory utilization.

7. CONCLUSIONS

A new hybrid algorithm combining the search feature of CS,FA, and PSO algorithms was implementation and tested. The proposed CSAI algorithm proved the capability of exploring new solution area. Using a set of ten benchmark functions that could be solved mathematically.The CSAI algorithm gave the best and closest solutions match when compared to CS, FA, and PSO algorithms. The proposed algorithm utilized more time to converge compared to CS, FA, PSO algorithms but still have the same level of memory usage. The algorithm features and capabilities will evidently become clearer when handling large scale problems having irregular solution space.

REFERENCE

[1] Abdullah, A. et al. 2012. A new hybrid firefly algorithm for complex and nonlinear problem. *Distributed Computing and Artificial Intelligence*. Springer. 673–680.

[2] Bratton, D. and Kennedy, J. 2007. Defining a standard for particle swarm optimization. *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE* (2007), 120–127.

[3] El-Sawy, A.A. et al. 2012. A Novel Hybrid Ant Colony Optimization and Firefly Algorithm for Solving Constrained Engineering Design Problems. *Journal of Natural Sciences and Mathematics*. 6, (2012).

[4] Farook, S. and Raju, P.S. Evolutionary Hybrid Genetic-Firefly Algorithm for Global Optimization.

[5] Gandomi, A.H. et al. 2013. Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems. *Engineering with Computers*. 29, (2013), 17–35.

[6] Ghodrati, A. and Lotfi, S. 2012. A hybrid CS/PSO algorithm for global optimization. *Intelligent Information and Database Systems*. Springer. 89–98.

[7] Hassanzadeh, T. and Meybodi, M.R. 2012. A new hybrid algorithm based on Firefly Algorithm and cellular learning automata. *20th Iranian Conference on Electrical Engineering (ICEE)* (2012), 628–633.

[8] Hezam, I.M. and AbdelRaouf, O. 2013. Particle Swarm Optimization Approach For Solving Complex Variable Fractional Programming Problems. *International Journal of Engineering*. 2, (2013).

[9] Hezam, I.M. and AbdelRaouf, O.M.M.H. 2013. Solving Fractional Programming Problems Using Metaheuristic Algorithms Under Uncertainty. *International Journal of Advanced Computing*. 46, (2013), 1261–1270.

[10] Hezam, I.M. and Raouf, O.A. 2013. Employing Three Swarm Intelligent Algorithms for Solving Integer Fractional Programming Problems. *International Journal of Scientific and Engineering Research (IJSER)*. 4, Issue 7, (2013), 191–198.

[11] Jones, K.O. and Boizanté, G. 2011. Comparison of Firefly algorithm optimisation, particle swarm optimisation and differential evolution. *Proceedings of the 12th International Conference on Computer Systems and Technologies* (2011), 191–197.

[12] Lee, K.Y. and El-Sharkawi, M.A. 2008. *Modern heuristic optimization techniques: theory and applications to power systems*. Wiley-IEEE press.

[13] Raju, R. et al. 2013. Hybrid Ant Colony Optimization and Cuckoo Search Algorithm for Job Scheduling. *Advances in Computing and Information Technology*. Springer. 491–501.

[14] Wang, F. et al. 2011. Hybrid optimization algorithm of PSO and Cuckoo Search. *Artificial Intelligence, Management Science and Electronic Commerce (AIMSEC), 2011 2nd International Conference on* (2011), 1172–1175.

[15] Wang, G. et al. 2012. A hybrid meta-heuristic DE/CS algorithm for UCAV path planning. *Journal of Information and Computational Science*. 5, (2012), 4811–4818.

[16] Yang, X.-S. 2010. *Engineering optimization: an introduction with metaheuristic applications*. Wiley.

[17] Yang, X.-S. 2011. *Nature-inspired metaheuristic algorithms*. Luniver Press.