Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.18576/amis/100337

Cuckoo Search and Genetic Algorithm Hybrid Schemes for Optimization Problems

Mohamed Abdel-Baset[1,*] *and Ibrahim Hezam*[2]

[1] Department of Operations Research, Faculty of Computers and Informatics, Zagazig University, El-Zera Square, Zagazig, Sharqiyah, Egypt.
[2] Department of computer, Faculty of Education, Ibb University, Ibb city, Yemen.

Received: 3 Nov. 2015, Revised: 3 Feb. 2016, Accepted: 4 Feb. 2016
Published online: 1 May 2016

Abstract: In this paper, two hybrid schemes using cuckoo search algorithm and genetic algorithm are proposed. In the two hybrid schemes, the algorithm consists of two phases in the first phase, CS (or GA) explores the search space. In the second phase, to improve global search and get rid of trapping into several local optima. The novel hybrid algorithms are applied to solve 15 benchmark functions chosen from literature. The simulation results and comparison with classical CS and GA algorithms confirm the effectiveness of the proposed algorithms in solving various benchmark optimization functions.

Keywords: Hybrid optimization; Cuckoo search; Genetic algorithm; Unconstrained optimization.

1. Introduction

Optimization is a field of applied mathematics that deals with finding the extremal values of a function in a domain of definition, subject to various constraints on the variable values [1], such problems are classified in two classes: unconstrained and constrained problems. Solving global optimization problems has made great gain from the interest in the interface between computer science and operations research [2, 3]. There are two categories of optimization techniques: exact and heuristic. Exact strategies guarantee the optimal solution will be found, and work well for many problems. However, for complex problems or ones with a very large number of decision variables, exact strategies may require very high computational costs [2]. A large amount of real-world problems fall in this category of complex problems, and in order to solve them in a reasonable amount of time a different approach is needed [2]. For these problems, meta-heuristic algorithms are considered as efficient tools to obtain optimal solutions [4]. Two important characteristics of meta-heuristics are intensification and diversification. Intensification intends to use the information from the current best solutions. This process searches around the neighborhood of the current best solutions and selects the best candidates. Diversification, also called exploration, guarantees that the algorithm can explore the search space more efficiently, often by randomization. The essential step guarantees that the system can jump out of any local optima and can generate new solutions as diversely as possible [5]. These methods have received remarkable attentions, as they are known to be derivative free, robust and often involve a small number of parameter tunings [4]. However, applying such single methods is sometimes too restrictive, especially for high dimensional and nonlinear problems. This is because these methods usually require a substantially huge amount of computational times and are frequently trapped in one of the local optima. Recently, different methods combining meta-heuristics with local search methods is a practical remedy to overcome the drawbacks of a slow convergence rate and random constructions of meta-heuristics [6]. In these hybrid methods, local search strategies are inlaid inside meta-heuristics in order to guide them, especially in the vicinity of local minima, and overcome their slow convergence especially in the final stage of the search. In recent years, it has become evident that the concentration on a sole meta-heuristic is rather restrictive. A combination of a meta-heuristic with other optimization techniques, a so-called hybrid meta-heuristic, can provide a more efficient behavior and a higher flexibility when dealing with real world and

* Corresponding author e-mail: analyst_mohamed@yahoo.com

large-scale problems. This is because hybrid meta-heuristics utilize the strengths of different algorithms [6, 7, 9–17]. In general, hybrid meta-heuristic approaches can be classified as either collaborative combinations or integrative combinations [8]. Recently, Yang [17] developed a new cuckoo search algorithm (CS) which was inspired by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds (often other species). In this paper, the two hybrid schemes are composed of standard CS and GA are presented. The proposed algorithms, CS-GA and GA-CS for solving unconstrained global optimization problems. Both variants are tested on a set of unconstrained problems. The experimental results show that the accuracy and speed performance of the proposed algorithms in comparison with the other state-of-the-art algorithms. The rest of this paper is organized as follows: in Section 2, a review of the basic of CS is given. GA is then presented in Section 3. The proposed algorithms are discussed in Section 4. Benchmark problems and corresponding experimental results are given in Section 5, while conclusions are given in Section 6.

2. Cuckoo Search Algorithm

The CS algorithm is a Meta-heuristic search algorithm, which has been proposed recently by Yang and Deb [17], it was based on the following idealized rules: (1) each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest. (2) The best nests with high quality of eggs (solutions) will be carried over to the next generations. (3) The number of available host nests is fixed, and a host can discover an alien egg with a probability. In this case, the host bird can either throw the egg away or abandon the nest to build a completely new nest in a new location. The main steps of the cuckoo search algorithm are summarized in Algorithm 1. When generating new solutions for the ith cuckoo, the following Lvy flight is performed:

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

where is the step size, which should be related to the scale of the problem of interest. The product means the entry-wise multiplications. We consider a Lvy flight in which the step-lengths are distributed according to the following probability distribution

$$Levy\, u = t^{-\lambda}, 1 < \lambda \leq 3 \quad (2)$$

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

Algorithm 1: Pseudo code for Cuckoo search algorithm

Define Objective function $f(x)$, $x = (x_1, x_2, ..., x_d)$
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

3. Genetic Algorithm

GA which proposed in the early 1970s [4] is a stochastic global search method that mimics the metaphor of natural biological evaluation. GA is started with an initial population of individuals (generation) which are generated randomly. Every individual (chromosome) encodes a single possible solution to the problem under consideration. The fittest individuals are chosen by ranking them according to a pre-defined fitness function, which is evaluated for each member of this population. The individuals with high fitness values, therefore represent a better solution to the problem than individuals with lower fitness values. Following this initial process, the crossover and mutation operations are used where the individuals in the current population produce the children (offspring). These children are assigned fitness scores. After selection, crossover and mutation have been applied to the initial population, a new population will have been formed and the generational counter is increased by one. This process of selection, crossover and mutation is continued until a termination condition is reached [4]. The structure of the GA is shown by the following pseudo-code

4. The Proposed Algorithm for Unconstrained Optimization Problem

The proposed CS-GA and GA-CS algorithms are collaborative combinations of the CS and GA techniques. In these hybrids, in the first step, CS (or GA) explores the search place in order to either isolate the most promising region of the search space. In the second step, to improve global search and avoid trapping into local optima, it is introduced GA (or CS) to explore search space (starting with the solution obtained by CS (or GA)) and find new

Algorithm 2: Pseudo code for Genetic algorithm

```
Define Objective function f(x), x = (x1, x2, ..., xd)
Encode the solution into chromosomes (binary strings)
Define fitness F ( F α f (x) for maximization)
Generate the initial population
Initial probabilities of crossover (pc) and mutation (pm)
  while (t < MaxGeneration) or (stop criterion);
        Generate new solution by crossover and mutation
        if pc>rand, Crossover; end if
        if pm>rand, Mutate; end if
        Accept the new solutions if their fitness increase
        Select the current best for a new generation
        (elitism)
  end while
Decode the results and visualization
End
```

Algorithm 3: Pseudo code for Hybrid CS-GA

```
Define Objective function f(x), x = (x1, x2, ..., xd)
Initial a population of n host nests xi (i = 1,2,...,d)
Define the cuckoo search parameters Pa
Define Genetic algorithm parameters pc, pm
begin CS
    while (t < MaxGeneration) or (stop criterion);
    Get a cuckoo (say i) randomly and generate a new
    solution by Lévy flights;
    Evaluate its quality/fitness; Fi
    Choose a nest among n (say j ) randomly;
    if (Fi> Fj),
    Replace j by the new solution;
    end
    Abandon a fraction (Pa) 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
     Final best population of nests;
End begin CS
Begin GA
    i=0
    Initial population P(0) = Final best population of nests
    Evaluate P(0) fitness
    while (t < MaxGeneration) or (stop criterion); do
    i=i+1
    Select P(i) from P(i − 1)
    Recombine P(i) with crossover probability pc
    Mutate P(i) with mutation probability pm
    Evaluate P(i) fitness
    end while
    Rank the chromosomes, find the current best and save
    Post process results and visualization
end begin GA
```

better solutions. The structure of the hybrid CS-GA is shown by the following pseudo-code as:

In analogical manner the hybrid GA-CS is introduced in the first step, GA explores the search place in order to generate solutions and then uses them as an initial population for CS. Thus, the CS will start with a population, which is closer to optimal solution. Further, CS will be obtained the best model parameters vector. The structure of the hybrid GA-CS is shown by the following pseudo-code as:

Algorithm 4: Pseudo code for GA-CS

```
Define Objective function f(x), x = (x1, x2, ..., xd)
Initial a population of n host nests xi (i = 1,2,...,d)
Define the cuckoo search parameters
Define Genetic algorithm parameters pc, pm
while (t < MaxGeneration) or (stop criterion);
          Begin GA
          i=0
          Initial population P(0) = Final best population of
          nests
          Evaluate P(0) fitness
          while (t < MaxGeneration) or (stop criterion); do
          i=i+1
          Select P(i) from P(i − 1)
          Recombine P(i) with crossover probability pc
          Mutate P(i) with mutation probability pm
          Evaluate P(i) fitness
          end while
   end while
   Rank the chromosomes, find the current best and
   save
   Post process results and visualization
   end begin GA
For i=1:n
Get a cuckoo (say i) randomly and generate a new
solution by Lévy flights;
Evaluate its quality/fitness; Fi
Choose a nest among n (say j ) randomly;
if (Fi> Fj),
          Replace j by the new solution;
end
```

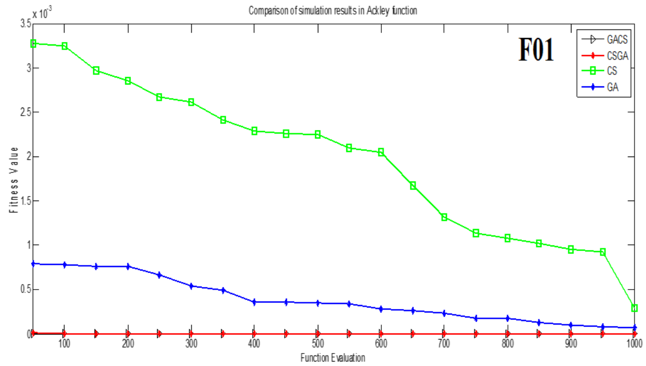
5. Numerical Results

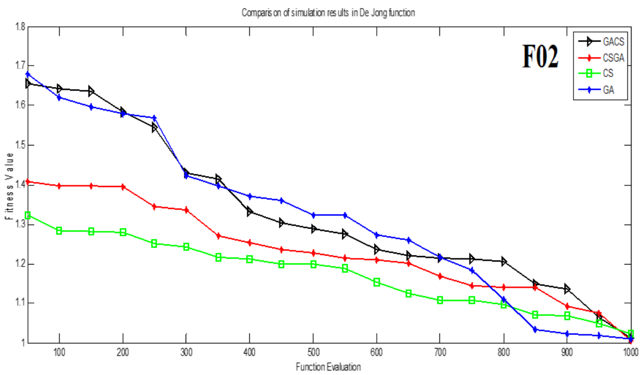
In this section, we will carry out numerical simulation based on some well-known optimization problems [9, 13, 17] to investigate the performances of the proposed algorithms. The functions name with global optimum, search ranges and initialization ranges of the test functions are presented in Table 1. In these problems, the essential parameters of CS are number of nests n=100, discovery rate of alien eggs/solutions pa=0. 25. And probability of crossover is 0.85. Which are the same used for GA-CS and CS-GA algorithms. The results of hybrid algorithms are conducted from 20 independent runs for each problem. All the experiments were performed on a Windows 7 Ultimate 64-bit operating system; processor

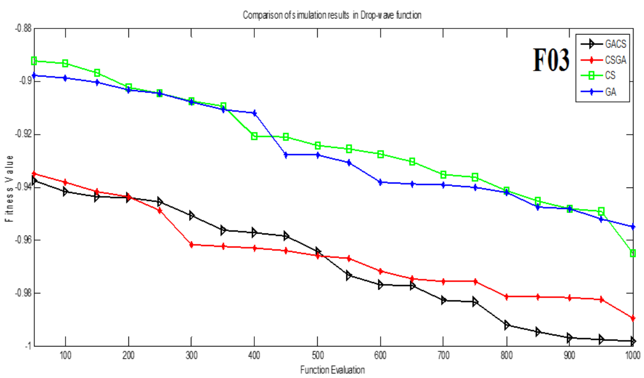
Intel Core i7 760 running at 2.81 GHz; 6 GB of RAM and code was implemented in C#.

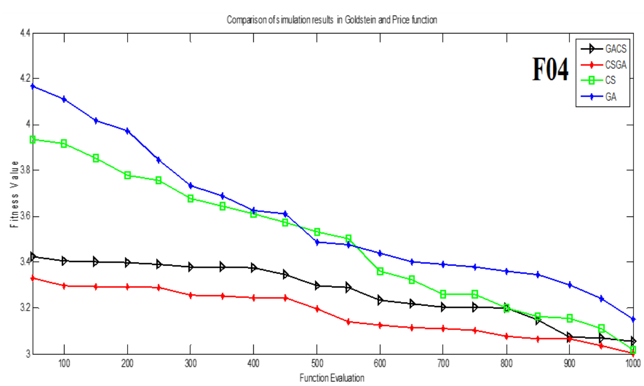
Table 1: The Benchmark functions.

ID	Function Name	Formulation	Global minimum
F01	Ackley	$20+e-20\exp(-0.2\sqrt{\frac{\sum_{i=1}^{N} x_i^2}{N}})-\exp(\sqrt{\frac{\sum_{i=1}^{N} x_i^2\cos(2\pi x_i)}{N}})$ $-32 \le x_i \le 32$	0
F02	De Jongfunction N:5	$\left(0.002+\sum_{i=1}^{25}\frac{1}{i+(x_1-a_{1i})^6+(x_2-a_{2i})^6}\right)$ $a=\begin{pmatrix}-32 & -16 & 0 & 16 & 32 & -32 \ldots 16 & 32\\ -32 & -32 & -32 & -32 & -16 \ldots 32 & 32\end{pmatrix}$ $-65.536 \le x_1, x_2 \le 65.536$	1
F03	Drop-wave function (dimensions=2)	$\frac{1+\cos\left(12\sqrt{x_1^2+x_2^2}\right)}{2+0.5\left(x_1^2+x_2^2\right)}$ $-5.12 \le x_1, x_2 \le 5.12$	-1
F04	Goldstein and Price	$\left[1+(x_1+x_2+1)^2\left(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2\right)\right]$ $\times\left[30+(2x_1-3x_2)^2\left(18-32x_1+12x_1^2+48x_2-36x_1x_2+2\right.\right.$ $-2 \le x_1, x_2 \le 2$	3
F05	Griewank	$\frac{1}{4000}\sum_{i=1}^{N} x_i^2-\prod_{i=1}^{N}\cos(\frac{x_i}{\sqrt{i}})+1$ $-600 \le x_i \le 600$	0
F06	Hezam function (*dimensions=2*)	$-\frac{1}{1+\lvert z^n-\tan z\rvert}$ $z \in C, n=20;\ z=x_1+ix_2 \in [-2,2]$	-1
F07	Himmelblau	$\left(x_1^2+x_2-11\right)^2+\left(x_2^2+x_1-7\right)^2$	0
F08	Powell function (*dimensions=24*)	$\sum_{i=1}^{n/k}\left(x_{4i-3}+10x_{4i-2}\right)^2+5\left(x_{4i-1}+x_{4i}\right)^2$ $+\left(x_{4i-2}+x_{4i-1}\right)^4+10\left(x_{4i-3}+10x_{4i}\right)^4$ $-4 \le x_i \le 5$	0
F09	Rastrigrin	$\sum^{N}(x_i^2-10\cos(2\pi x_i)+10)$ $-100 \le x_i \le 100$	0
F10	Rosenbrock	$\sum_{i=1}^{N-1}(100(x_{i+1}-x_i^i)^2+(x_i-1)^2)$ $-100 \le x_i \le 100$	0
F11	Rotated hyper-ellipsoid	$\sum_{i=1}^{N}(\sum_{j=1}^{i} x_j)^2$ $-100 \le x_i \le 100$	0
F12	Schwefel	$418.9829N-\sum_{i-1}^{N}(x_i \sin(\sqrt{\lvert x_i\rvert}))$ $-500 \le x_i \le 500$	0
F13	SineEnvelope function (*dimensions=2*)	$-\sum_{i=1}^{n-1}\left[\frac{\sin^2\left(\sqrt{x_{i+1}^2+x_i^2}-0.5\right)}{\left(0.001\left(x_{i+1}^2+x_i^2\right)+1\right)^2}+0.5\right]$; *n=20* $-100 \le x_i \le 100$	0
F14	Sphere	$\sum_{i=1}^{N} x_i^2;\ -100 \le x_i \le 100;$	0
F15	Styblinski–Tang function (*dimensions=30*)	$\frac{\sum_{i=1}^{n} x_i^4-16x_i^2+5x_i}{2}$ $-5 \le x_i \le 5$	-39.16599*D*





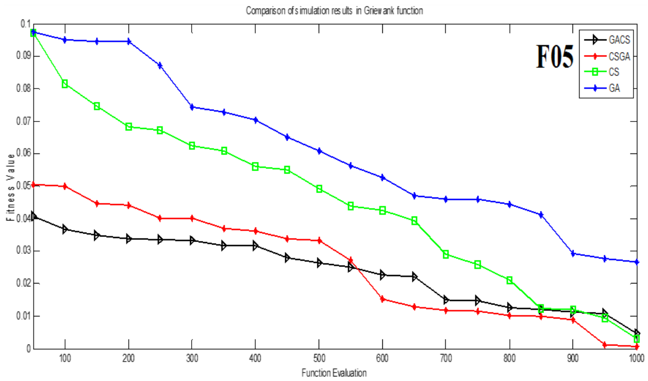


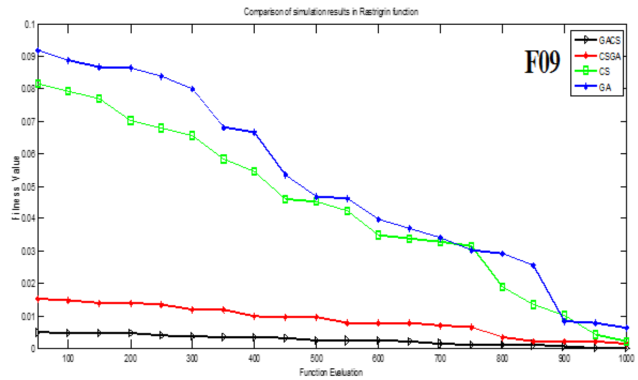


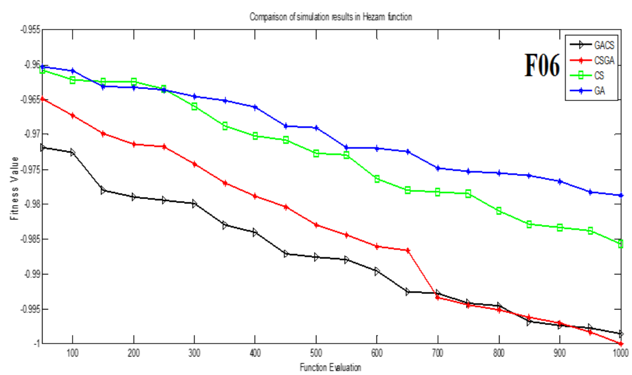
The results from the GA-CS and CS-GA (see Table 2), show that the proposed hybrid schemes achieve better to pure CS and pure GA solutions. The results show that CS-GA and GA-CS are robust and competitive with the state-of the-art well-known evolutionary algorithms. It is clear that the performance of GA-CS and CS-GA are

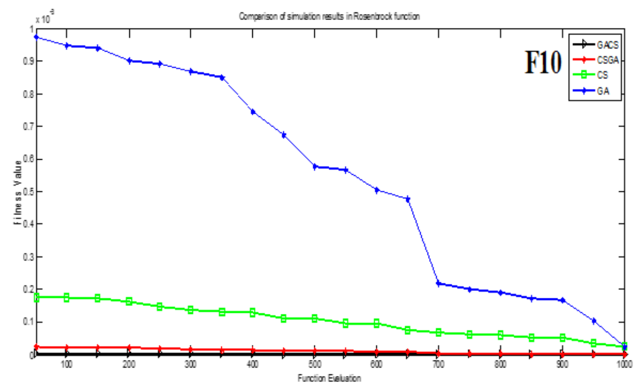
Table 2: The optimal solution results of proposed algorithms and other algorithms.

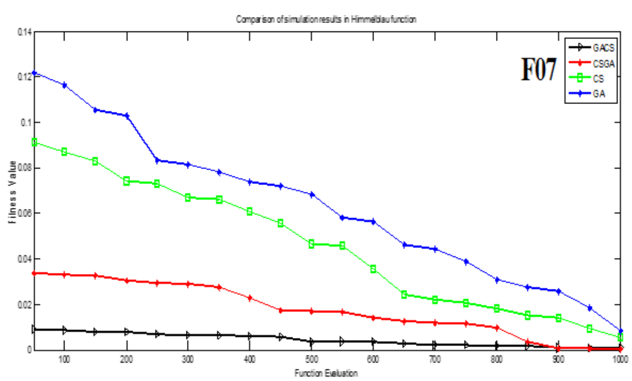
Test Problem	Algorithm	Min	Max	Mean	Standard Deviation
F01	GA	6.4354e-008	9.0832e-004	0.000386110054093954	0.000251240844284281
	CS	7.0857e-0010	3.3956e-003	0.00196796704502479	0.000868387090205297
	CS-GA	**0**	9.9876e-006	**4.25076756664906e-06**	3.55535898617486e-06
	GA-CS	**0**	**9.3947e-006**	5.92756737453864e-06	2.31161529528762e-06
F02	GA	1.0064	1.6861	0.215469063487366	0.215469063487366
	CS	1.0024	**1.336**	1.17401349318838	0.0889985723631675
	CS-GA	**1**	1.4248	**1.23250238912314**	0.117452228657367
	GA-CS	**1**	1.6669	1.32766631650774	0.196845428932942
F03	GA	-0.96244	-0.89286	-0.926222939215064	0.0196413522573634
	CS	-0.96523	-0.88591	-0.923889868025335	0.0205793851127676
	CS-GA	**-1**	-0.92882	-0.965332563028172	0.0162360093085712
	GA-CS	**-1**	**-0.9362**	**-0.968650362917142**	0.0211800929150197
F04	GA	3.0108	4.2177	3.58716193554105	0.299819748973087
	CS	3.0096	4.0294	3.48120716678074	0.287049866880929
	CS-GA	**3**	**3.3486**	**3.17621137665972**	0.103160903013155
	GA-CS	**3**	3.4321	3.27466752019290	0.123278227517195
F05	GA	0.0037957	0.10117	0.0614782171440947	0.0235238040874096
	CS	0.001209	0.09874	0.0455830028746040	0.0263075567952727
	CS-GA	**0**	0.051161	0.0259777253362568	0.0168043827889759
	GA-CS	**0**	**0.041966**	**0.0240884217298361**	0.0106109821995857
F06	GA	-0.98131	-0.95873	-0.969787297721470	0.00607110529490644
	CS	-0.98613	-0.95926	-0.972978606526178	-0.972978606526178
	CS-GA	**-1**	-0.96478	-0.983463971235532	0.0113404574610371
	GA-CS	**-1**	**-0.96723**	**-0.987177505706453**	0.00842659714661514
F07	GA	**0**	0.12651	0.0629646893745502	0.0331783713969257
	CS	**0**	0.091854	0.0456594004043365	0.0284317032914567
	CS-GA	**0**	0.034018	0.0176663168865541	0.0115764358456526
	GA-CS	**0**	**0.0090118**	**0.00437901892556019**	0.00278119978232517
F08	GA	0.00011	0.010047	0.00540115636170945	0.00291827118840213
	CS	0.00011153	0.0096895	0.00515239079563775	0.00515239079563775
	CS-GA	8.7855e-009	**0.0063907**	**0.00344470143045183**	0.00166670888303499
	GA-CS	**0**	0.0087066	0.00467837945513165	0.00244745306664402
F09	GA	1.8489e-007	0.095061	0.0508823938677554	0.0290039163250799
	CS	2.9937e-009	0.091618	0.0435289424139179	0.0254254717941823
	CS-GA	**0**	0.015309	0.00868587854151802	0.00460870098307971
	GA-CS	**0**	**0.005710**	**0.00265857687336894**	0.00159012700601643
F10	GA	**0**	9.8743e-004	0.000549536235969843	0.000333169454859424
	CS	**0**	1.8423e-004	0.000103788911748824	4.89366886202407e-05
	CS-GA	**0**	2.4159e-005	1.11133858886066e-05	8.01119267542348e-06
	GA-CS	**0**	**4.8473e-006**	**2.11962809878174e-06**	1.47715471346944e-06
F11	GA	1.0316e-024	0.022088	0.0103833877269048	0.00673219452903841
	CS	5.7574e-028	0.03811	0.0194430677475940	0.0128919811815733
	CS-GA	**0**	0.0076472	0.00456504614399630	0.00211001284923032
	GA-CS	**0**	**0.0068709**	**0.00308962054445953**	0.00170377174312602
F12	GA	**0**	0.034796	0.0116010542205684	0.0116010542205684
	CS	**0**	0.026021	0.00746364541025885	0.00746364541025885
	CS-GA	**0**	**0.004148**	0.00203648776623796	0.00113269138852288
	GA-CS	**0**	0.005086	**0.00196362935549940**	0.00196362935549940
F13	GA	**0**	0.09654	0.0511310578334745	0.0269771870632130
	CS	**0**	0.08954	0.0400286350453206	0.0207198732780366
	CS-GA	**0**	**0.05841**	**0.0300948043200226**	0.0206816914282432
	GA-CS	**0**	0.07168	0.0328732247722852	0.0220191813692026
F14	GA	4.2005e-006	9.8120e-003	0.00541505610742572	0.00303969722212693
	CS	7.9725e-008	8.7661e-003	0.00429541152658520	0.00249176461642322
	CS-GA	**0**	**2.6763e-005**	**1.19590057893899e-05**	6.77761778750693e-06
	GA-CS	**0**	5.6714e-004	0.000236981205455750	0.000188661537398281
F15	GA	-11748.964	-740.5938	-6479.59794521835	3785.56882331539
	CS	-11749.797	-761.5086	-6646.27539761730	2793.70836672638
	CS-GA	**-11750**	-795.2994	-6289.90529099173	3530.21389600095
	GA-CS	**-11750**	**-828.6905**	**-7337.97371888534**	2477.16321287490

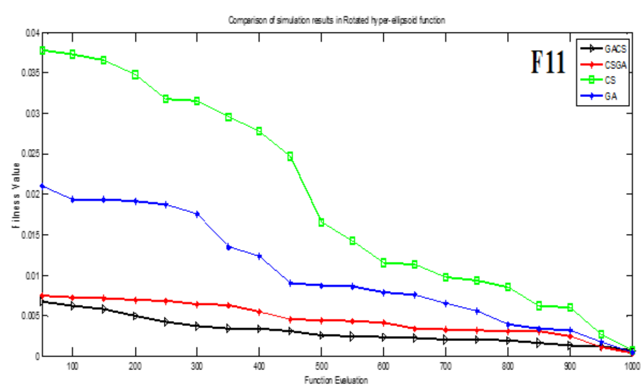

Comparison of simulation results in Griewank function
F05
GACS
CSGA
CS
GA
Fitness Value
Function Evaluation

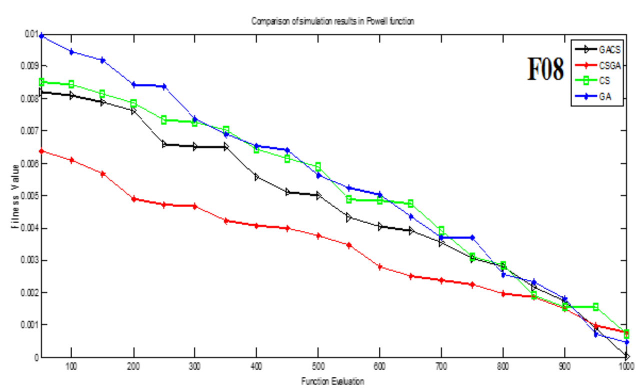

Comparison of simulation results in Rastrigin function
F09
GACS
CSGA
CS
GA
Fitness Value
Function Evaluation

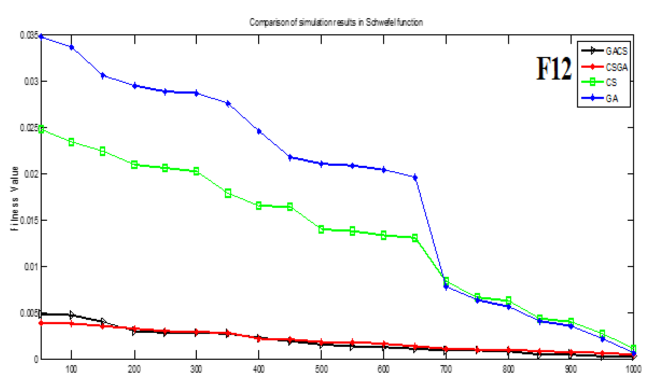

Comparison of simulation results in Hezam function
F06
GACS
CSGA
CS
GA
Fitness Value
Function Evaluation


Comparison of simulation results in Rosenbrock function
F10
GACS
CSGA
CS
GA
Fitness Value
Function Evaluation


Comparison of simulation results in Himmelblau function
F07
GACS
CSGA
CS
GA
Fitness Value
Function Evaluation


Comparison of simulation results in Rotated hyper-ellipsoid function
F11
GACS
CSGA
CS
GA
Fitness Value
Function Evaluation


Comparison of simulation results in Powell function
F08
GACS
CSGA
CS
GA
Fitness Value
Function Evaluation


Comparison of simulation results in Schwefel function
F12
GACS
CSGA
CS
GA
Fitness Value
Function Evaluation

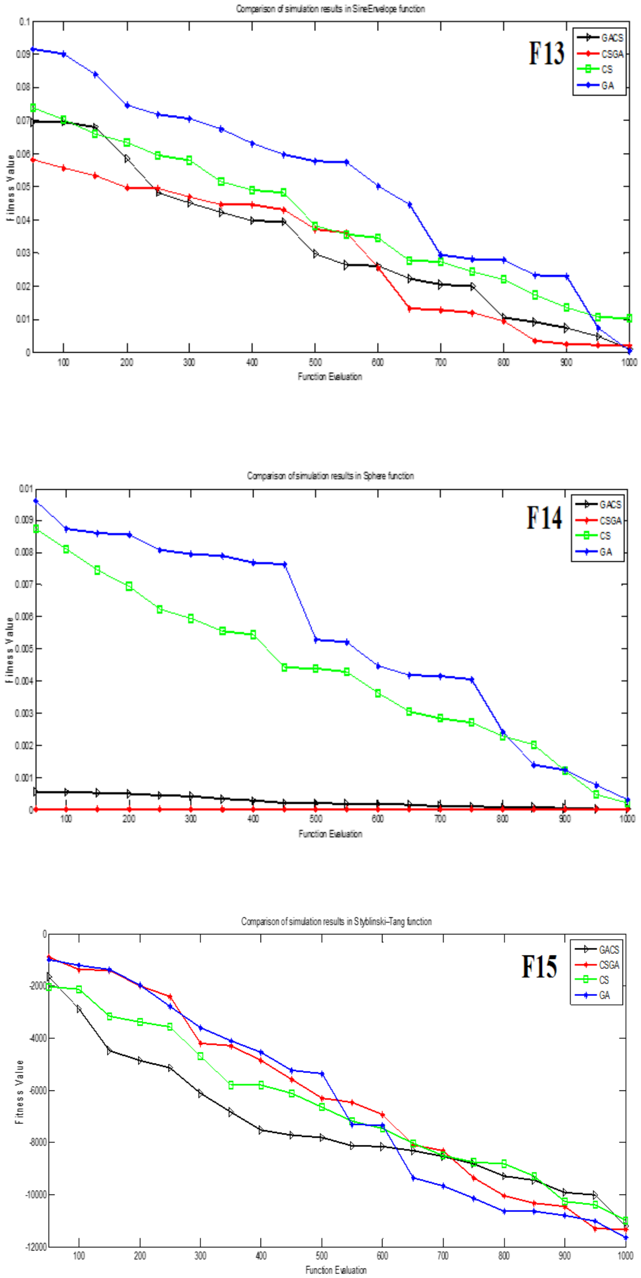


Figure 1:F-01:F15 the convergence rate of the function error values on 15 functions.

significantly superior to all the present algorithms for all functions according to the experimental results shown in figure 1. The mean and the difference between the best value and worst value of the result obtained by GA-CS were small compared to the results we have obtained from other algorithms in functionsF03, F05, F06, F07, F09, F10, F11 and F15. While in F01, F02, F04, F08, F13 and F14, the mean and the difference between the best value and worst value of the result obtained by CS-GA were small compared to the results we have obtained from other algorithms. In general, the performance of GA-CS and CS-GA are highly competitive with other algorithms.

6. Conclusions

In the present study, two hybrid meta-heuristic CS-GA and GA-CS algorithms were proposed, which are collaborative combinations of the CS and GA techniques. CS-GA and GA-CS algorithms have been employed to solve unconstrained optimization problems. GA-CS and CS-GA algorithms have been validated using fifteen benchmark mathematical functions. Several simulation examples have been completed to verify the weight of the planned algorithm. A comparison of pure CS, pure GA and hybrids CS-GA and GA-CS algorithms were done. The results have demonstrated the superiority of the CS-GA and GA-CS algorithms to finding the solution. The results indicate that CS-GA and GA-CS algorithms are more accurate, reliable and efficient at finding global optimal solution than are other algorithms.

Acknowledgements

The authors would like to thank the anonymous reviewers for their careful reading of this paper and for their helpful comments.

References

[1] S.S. Rao and S. Rao, Engineering optimization: theory and practice, John Wiley & Sons, 2009.
[2] L. Liberti and N. Maculan, Global optimization: from theory to implementation, Springer Science & Business Media, 2006.
[3] O.A. Raouf and I.M. Hezam, Solving Fractional Programming Problems based on Swarm Intelligence, Journal of Industrial Engineering International, vol. 10, 2014, pp. 110.
[4] X.-S. Yang, Engineering optimization: an introduction with metaheuristic applications, John Wiley & Sons, 2010.
[5] R.S. Parpinelli and H.S. Lopes, New inspirations in swarm intelligence: a survey, International Journal of Bio-Inspired Computation, vol. 3, 2011, pp. 116.
[6] G. Wang and L. Guo, A novel hybrid bat algorithm with harmony search for global numerical optimization, Journal of Applied Mathematics, vol. 2013, 2013.
[7] O. Roeva, Genetic Algorithm and Firefly Algorithm Hybrid Schemes for Cultivation Processes Modelling, Transactions on Computational Collective Intelligence XVII, Springer, 2014, pp. 196211.
[8] V. Atanassova, S. Fidanova, I. Popchev, and P. Chountas, Generalized nets, ACO algorithms and genetic algorithms, Proceedings in Mathematics Monte Carlo Methods and Applications, De Gruyter, 2012, pp. 3946.

[9] M. Abdel-Baset and I.M. Hezam, An Effective Hybrid Flower Pollination and Genetic Algorithm for Constrained Optimization Problems, Advanced Engineering Technology and Application An International Journal, vol. 4, 2015, pp. 27 2734.
[10] M. Abdel-Basset and I. Hezam. ”An Improved Flower Pollination Algorithm for Ratios Optimization Problems.” Applied Mathematics & Information Sciences Letters An International Journal, vol. 3, No. 2 , 2015, pp 83-910.
[11] G. Kanagaraj, S. Ponnambalam, and W. Lim, Application of a hybridized cuckoo search-genetic algorithm to path optimization for PCB holes drilling process, Automation Science and Engineering (CASE), 2014 IEEE International Conference on, IEEE, 2014, pp. 373378.
[12] W. Lim, G. Kanagaraj, and S. Ponnambalam, A hybrid cuckoo search-genetic algorithm for hole-making sequence optimization, Journal of Intelligent Manufacturing, 2014, pp. 113.
[13] I.M.H. Mohamed Abdel-Baset, An Improved Flower Pollination Algorithm Based on Simulated Annealing for Solving Engineering Optimization Problems, Asian Journal of Mathematics and Computer Research, vol. 3, 2015, pp. 149170.
[14] S.A. Pradeep and R. Manavalan, Analysis of Cuckoo Search with Genetic Algorithm for Image Compression, International Journal of Engineering Research, vol. 2, 2013, pp. 386392.
[15] M. Rashad, A. Keshk, M. El-Dosuky, and M. Kamal, Genetic Cuckoo Optimization Algorithm (GCOA), International Journal of Computer Applications, vol. 90, 2014, pp. 712.
[16] H. Zheng, Y. Zhou, and P. Guo, Hybrid genetic-cuckoo search algorithm for solving runway dependent aircraft landing problem, Research Journal of Applied Sciences, Engineering and Technology, 2013, pp. 21362140.
[17] A.H. Gandomi, X.-S. Yang, and A.H. Alavi, Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems, Engineering with Computers, vol. 29, 2013, pp. 1735.

Mohamed A. Metwally Received his B.Sc., M.Sc and the Ph.D in information technology from Zagazig University. He is a lecturer in the operations Research Department, Faculty of Computers and Informatics, Zagazig University. His current research interests are Optimization, Operations Research, Data Mining, Computational Intelligence, Applied Statistics and Decision support systems. He is also a reviewer in different international journals and conferences. He has published more than 70 articles in international journals and conference proceedings.

Ibrahim Hezam obtained his B.Sc. in 2003 in Mathematics from IBB University, IBB, Yemen; His M.Sc. in Pure Mathematics (Operations Research) from Helwan University, Egypt, in 2011. Currently, he is a Ph.D. Student in Department of Operations Research and Decision Support, Faculty of Computers & Information, Menoufia University, Egypt. His research interests are metaheuristic optimization methods, artificial intelligence, and decision support systems.