Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 3557 – 3566

doi:10.3906/elk-1810-74

HGAB3C: A new hybrid global optimization algorithm

Kamaljeet KAUR[1,*], Shakti KUMAR[2], Jyoti SAXENA[3]
[1]I.K. Gujral Punjab Technical University, Jalandhar, Punjab, India
[2]Department of Electronics and Communication Engineering, Panipat Institute of Engineering and Technology, Panipat, Haryana, India
[3]Department of Electronics and Communication Engineering, Giani Zail Singh Campus College of Engineering and Technology, Maharaja Ranjit Singh Punjab Technical University, Bathinda, Punjab, India

Received: 09.10.2018 • **Accepted/Published Online:** 24.05.2019 • **Final Version:** 18.09.2019

Abstract: This paper proposes a new optimization algorithm, namely HGAB3C, and presents its performance on the CEC-2014 test suite. In HGAB3C, simple genetic algorithms (GAs) and big bang-big crunch (BB-BC) are hybridized. The algorithm carries out global searches using a simple GA. In every generation the BB-BC algorithm is used to carry out local searches. The addition of local search has improved the capability of simple GAs significantly. The performance of the proposed algorithm is compared with 17 other optimization algorithms on all 30 functions of the CEC-2014 benchmark suite. It is observed that HGAB3C outperforms all other algorithms on 4 benchmark functions. For the 3 other functions, its performance equaled the best of the competing algorithms, which makes HGAB3C's performance best in a total of 7 benchmark functions. Out of the 18 competing algorithms, the proposed algorithm ranked second for the unmatched best mean error measure. For the best performance measure (number of functions giving unmatched best and equaled best mean error), the proposed algorithm was the third best. As far as the speed of convergence is concerned, the algorithm gave an unmatched best performance for the shifted Schwefel function (function 10 of CEC-2014 test bench). It obtained a mean error value of 0.00E+00, outperforming the previous best of 1.23E-03, converging to the target result in an average of 346.44 generations, which no other algorithm could achieve.

Key words: Optimization algorithm, HGAB3C, CEC-2014

1. Introduction

Engineering system complexity has grown many folds in the last few decades. Engineering system design is, in a way, a process of optimizing design parameters based on some specified objective function(s). Business decisions to maximize profits, data analysis of huge databases such as social media applications, metropolis vehicular traffic, supply-chain applications, and drug design, which are growing with exponential complexity, all call for newer quick and efficient optimization techniques. The demand for such optimization techniques today is greater than ever before. These increasing system complexities and larger databases are forcing researchers to search for new and better optimization techniques.

It has been observed that combining different optimization techniques with distinct features leads to better techniques with improved results for optimization problems. Kumar et al. [1] proposed a multipopulation big bang-big crunch algorithm and also evaluated its performance on the CEC-2014 test bench, finding it to be in the category of best algorithms. Zhu et al. [2] proposed an improved genetic algorithm (GA) with some

*Correspondence: lotus.k30@gmail.com

local search operators to make it efficient for dynamic shortest-path problems. An evolutionary algorithm based on covariance matrix learning and searching preference (CMLSP) proposed in [3] by Chen et al. discarded the traditional crossover and mutation methods and was able to generate high-quality solutions around a good solution. Elsayed et al. proposed the combination of three multioperator evolutionary algorithms into one algorithm named UMOEAS (united multioperator evolutionary algorithm) [4, 5], and it proved to be superior to single-operator evolutionary algorithms. In mean-variance mapping optimization (MVMO-SH), multiparent crossover was integrated in swarm intelligence [6] and it effectively tackled a variety of problems with varied mathematical properties and dimensions. MVMO with single parent–offspring approach has also been effectively used on the CEC-2014 test bench [7]. Erlich et al. carried out this integration with the intention of forcing the particles with the least fitness to explore other regions of the solution search space. Molina et al. proposed a region-based MA with a trade-off between exploration and exploitation in [8]. Hu et al. proposed a hybrid algorithm based on opposition-based learning (OBL) and adaptive differential evolution (ADE) named partial opposition-based learning and adaptive differential evolution (POBL-ADE) [9]. A variant of differential evolution with controlled restart was proposed in [10]. Another variant of DE, i.e. success-history based adaptive differential evolution (SHADE), was modified to incorporate linear population size reduction in L-SHADE [11] and proved to be a very good optimization algorithm on the CEC-2014 test bench. Li et al. also incorporated a pattern-driven local search operator in binary particle swarm optimization to improve its search efficiency [12]. Buzok et al. [13] and Yavuz et al. [14] also evaluated their hybrid algorithms, SHADE4 and SSEABC, respectively, on the CEC-2014 test suite and found them to be better than their parent algorithms. Thus, there is enough evidence available in the literature to suggest that an appropriate combination of two different algorithms for exploration and exploitation can lead to results better than the individual results that these algorithms have produced.

This proposed work is based upon the fact that the GA is quick to find a near global best solution due to its operators like crossover and mutation, but it may be unable to search the local best solution as the a simple GA does not have any operator that could fine-tune the results to reach the global best solution [15]. Big bang-big crunch (BB-BC) is a widely accepted algorithm that finds the best solution among the current population [16, 17]. The next generation is evolved around the current best (elite) of the present population. This causes the solution to improve with every generation. Hence, it offers a possibility to fine-tune the coarse solution found in earlier generations. In the work presented in this paper, a simple GA and BB-BC are hybridized into a single algorithm to achieve a fine-tuned global best solution whereby the idea is to explore the complete solution search space to find a solution around the global best by the GA and then improve it by exploiting the achieved solution using BB-BC. In this paper we implement the hybridized algorithm in MATLAB. We validate the proposed algorithm on the CEC-2014 test bench and it is found to give very encouraging results.

This paper has been divided into 4 sections. Section 1 has presented the motivation for this work. Section 2 proposes the new hybridized algorithm, Section 3 presents simulation results and discussions, and Section 4 concludes the paper.

2. Proposed algorithm

In this section we present the proposed algorithm. In a simple GA, the creation of each new generation attempts to generate new fitter solutions. For every generation created by a simple GA, a local search carried out by the BB-BC algorithm attempts to further improve the fitness of the current generation. As stated earlier, in order to keep algorithm simple and effective, we chose a simple GA as our main algorithm to be fine-tuned for local searches by the BB-BC algorithm. The proposed algorithm is given below.

2.1. Nomenclature and variables/constants used

Variables
Nind: No. of individuals in the population.
Ngen: No. of maximum generations.
Fobj: Objective function that calculates the error/fitness for the given candidate solution.
Pop: Initial random population for GA and BB-BC.
G_itr: No. of generations for GA.
B_itr: No. of generations for BB-BC.
GenID: Current generation number.
Constants (for 10-dimensional CEC-14 benchmark functions)
Dim: No. of variables in desired solution = 10.
LB: Array containing lower bound of each solution variable = [-100 -100 -100 -100 -100 -100 -100 -100 -100 -100].
UB: Array containing upper bound of each solution variable = [100 100 100 100 100 100 100 100 100 100 100].
Fobj: Objective function that calculates the error/fitness for the given candidate solution.
Fcor: Correction function that ensures that the generated solution is within the stipulated lower and upper bounds (LB and UB), meeting all constraints. This function ensures that the updated value of each candidate is within the range of -100 to 100.
Target: Target error = 10^{-9}; error below this target value is considered to be 0.

2.2. Algorithm

Step 1: Initialization:
Create an initial population ‘Pop’ of ‘Nind’ individuals randomly within the constrained range of -100 to 100.
Step 2: Evaluate the fitness of each individual of the population:
Evaluate error (fitness) of the initial population for the minimization (maximization) problem and store the best individual as Elite and its error as EliteErr (EliteFitness). All the functions in the CEC-14 benchmark suite are minimization problems. Hence, in this paper, the error value of each individual is found and the individual with minimum error is selected as Elite. The error value corresponding to the selected Elite individual is EliteErr.
Step 3: Iterative process
GenID = 1
For i = 1 to Ngen
Phase 1: GA
For gi = 1to G_itr
Rank the population according to its fitness (less error leads to a more fit individual).
Select individuals for breeding. There are several methods of sampling to choose fit individuals for sampling. In this algorithm stochastic universal sampling has been used for selection.
Select individuals for breeding using stochastic universal sampling.
Perform crossover and mutation on selected individuals to generate offspring.
Evaluate fitness of each offspring.

Reinsert offspring into the current population to generate a new population for the next generation.
End For
Save the best fit individual generated in Phase1 as Elite and its error as EliteErr.
Phase 2: BB-BC
For bi = 1to B_itr
Compute the fitness of each candidate solution in the current population.
Compute the center of mass using Eq. (1) or select the best fit individual as the center of mass:

$$x = \frac{\sum_{i=1}^{N} \frac{1}{f_i} x_i}{\sum_{i=1}^{N} \frac{1}{f_i}}, \tag{1}$$

where f_i is the fitness of the ith candidate solution, i.e. x_i, and N is the number of individuals in a population, i.e. Nind. In this work, the best fit individual is considered as the center of mass.
Evolve a new generation around every Elite as follows:
For k = 1 to Nind
Using Eq. (2), calculate new candidate solutions around the center of mass/elite by adding or subtracting a normal random number whose value decreases as the iterations elapse.

$$x_{new}(k) = Elite + \frac{lr}{m} \tag{2}$$

End For
(l and m are proportional to the upper limit of the parameter and number of iterations, respectively, and r is an array of random numbers between 0 and 1 and has the same length as that of a candidate solution/Elite.)
x_{new} is the new candidate solution.
GenID = GenID + 1;
End For
End For

3. Performance evaluation: simulation results and discussion

In order to evaluate the performance of the proposed HGAB3C algorithm we implemented the algorithm in MATLAB and simulated its performance on 30 CEC-2014 benchmark functions with 10 dimensions only. We used core i5 @ 3.2GHz with 4GB RAM-based desktop computers for evaluating the performance. We conducted 25 trials for each function. The performance of the proposed algorithm on the CEC-2014 test suite is shown in Table 1. We compared the performance of HGAB3C with 17 other algorithms as given in [18, 19]. This comparison is shown in Table 2.

The detailed comparative performance of the 18 algorithms is shown in Table 2, while Table 3 summarizes the category-wise comparative performance of HGAB3C with 17 other algorithms. From Table 3 we observe that HGAB3C gives the best performance for 7 of the 30 functions of the CEC-2014 test bench. We further observe that out of these 7, for 4 functions i.e. function numbers 10, 20, 21, and 27, the proposed algorithm gave the unmatched best performance. It obtained the unmatched best performance for function numbers 10, 20, 21, and 27 with mean error of 0, 1.78E-02, 3.88E-02, and 1.31E-01, respectively, beating the earlier best of 1.23E-03, 5.59E-02, 4.08E-01, and 1.54E-01, respectively. For 3 other functions, i.e. function numbers 3, 8, and 26, it matched the best performance given by other algorithms, as well. In the overall comparison we find that the UMOEAS algorithm ranks first with the best performance for 11 of the 30 functions. However, UMOEAS could

Table 1. Performance of HGAB3C algorithm on CEC-2014 test bench.

Function number	Best	Median	Worst	Mean	Standard deviation	Earlier best mean
1	4.64E-08	5.83E-08	6.41E-08	5.76E-08	4.27E-09	0.00E+00
2	1.40E-01	1.66E-01	1.78E-01	1.65E-01	1.11E-02	0.00E+00
3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
4	0.00E+00	0.00E+00	3.48E+01	1.11E+01	1.66E+01	0.00E+00
5	1.84E-09	2.00E+01	2.00E+01	1.76E+01	6.63E+00	1.64E-07
6	5.50E-06	3.00E-03	9.84E-01	1.70E-01	3.49E-01	0.00E+00
7	4.68E-02	7.38E-02	8.60E-01	1.36E-01	2.08E-01	0.00E+00
8	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
9	1.99E+00	2.98E+00	5.99E+00	3.54E+00	1.70E+00	1.65E+00
10	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.23E-03
11	3.48E+00	2.68E+01	2.52E+02	6.13E+01	6.96E+01	2.98E+00
12	6.59E-04	6.95E-04	6.95E-04	6.94E-04	7.35E-06	0.00E+00
13	1.44E-02	1.65E-02	1.54E-01	2.54E-02	2.97E-02	9.44E-03
14	1.40E-03	5.40E-03	9.80E-03	5.67E-03	2.37E-03	5.68E-04
15	2.80E-01	5.04E-01	9.30E-01	5.36E-01	1.90E-01	3.30E-01
16	9.72E-02	1.08E+00	2.54E+00	1.15E+00	7.11E-01	1.93E-01
17	1.90E-03	1.24E+01	4.00E+01	9.50E+00	1.07E+01	1.40E+00
18	1.33E-02	3.00E+00	1.56E+01	3.68E+00	3.54E+00	7.83E-01
19	1.95E-02	2.93E-02	1.01E+00	1.55E-01	3.21E-01	7.66E-02
20	2.90E-03	4.60E-03	7.72E-02	1.78E-02	2.61E-02	5.59E-02
21	4.05E-05	4.29E-04	3.14E-01	3.88E-02	1.03E-01	4.08E-01
22	4.20E-03	5.53E-02	4.16E-01	1.42E-01	1.53E-01	4.41E-02
23	3.29E+02	3.29E+02	3.29E+02	3.29E+02	2.32E-13	2.00E+02
24	1.08E+02	1.14E+02	1.18E+02	1.13E+02	2.54E+00	1.06E+02
25	1.13E+02	1.21E+02	2.01E+02	1.32E+02	2.66E+01	1.01E+02
26	1.00E+02	1.00E+02	1.00E+02	1.00E+02	4.09E-02	1.00E+02
27	8.70E-03	9.36E-02	5.95E-01	1.31E-01	1.22E-01	1.53E-01
28	3.57E+02	3.69E+02	4.79E+02	3.75E+02	3.15E+01	2.00E+02
29	2.23E+02	2.48E+02	2.69E+02	2.51E+02	1.23E+01	1.96E+02
30	4.59E+02	4.84E+02	5.78E+02	4.88E+02	2.67E+01	2.00E+02

give only 2 unmatched best performances, and for the other 9 functions its performance was equaled by other algorithms as well. PB3C is ranked second. We rank HGAB3C, LSHADE, and DE_b6e6rlwithrestart third, followed by CMSLSP, SOO, SOO+BOBYQA, GaAPDE, and RMA-LSChCMA. Considering the unmatched best criteria as performance metrics we rank PB3C as the best algorithm with 7 unmatched best performances. We find that HGAB3C ranks second with 4 unmatched best performances and UMOEAS and LSHADE are ranked third with both having 2 unique best performances. It is also evident from Table 3 that the best performance of HGAB3C is distributed across all 4 categories of test functions. UMOEAS and LSHADE are

Table 2. Comparison of HGAB3C with 17 other algorithms on CEC-2014 benchmark functions.

S. no.	Algorithm	F1	F2	F3	F4	F5	F6
1	NRGA	2.79E+04	9.15E+02	1.52E+03	1.54E+01	1.96E+01	2.45E+00
2	FWA_DM	5.01E+03	1.34E-04	1.88E-09	1.41E+00	2.00E+01	7.06E-01
3	UMOEAS	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.68E+01	0.00E+00
4	SOO + BOBYQA	4.57E+03	3.60E-02	5.84E+03	0.00E+00	2.00E+01	2.00E-03
5	SOO	8.81E+06	6.34E+00	6.64E+03	6.78E-01	2.00E+01	2.00E-03
6	RSDE	0.00E+00	0.00E+00	0.00E+00	2.81E+00	1.92E+01	5.29E-02
7	POBL_ADE	1.62E+04	2.27E+03	5.74E-04	2.55E+01	1.91E+01	1.04E+00
8	FERDE	2.37E+00	6.29E-05	1.35E-03	0.00E+00	1.91E+01	8.89E-01
9	FCDE	0.00E+00	0.00E+00	0.00E+00	1.84E+01	2.03E+01	3.57E+00
10	DE_b6e6rlwithrestart	0.00E+00	0.00E+00	0.00E+00	1.13E+00	1.85E+01	0.00E+00
11	CMLSP	1.77E-07	1.12E-15	1.06E-04	3.34E-15	1.69E+01	6.20E-02
12	GaAPADE	0.00E+00	0.00E+00	0.00E+00	3.07E+01	1.97E+01	1.48E-01
13	OptBees	7.84E+02	9.88E-03	9.21E-01	2.69E+00	2.00E+01	3.02E+00
14	LSHADE	0.00E+00	0.00E+00	0.00E+00	2.94E+01	1.42E+01	1.75E-02
15	RMA_LSCh_CMA	0.00E+00	0.00E+00	1.03E-07	8.50E-02	1.37E+01	1.48E-04
16	MVMO	4.95E-04	7.10E-09	9.86E-11	9.55E+00	1.66E+01	3.45E-03
17	PB3C	4.66E+01	1.44E+01	6.27E-09	3.72E-04	1.64E-07	1.96E-02
18	HGAB3C	5.76E-08	1.65E-01	0.00E+00	1.11E+01	1.76E+01	1.70E-01
S. no.	Algorithm	F7	F8	F9	F10	F11	F12
1	NRGA	2.03E-01	5.59E+00	8.69E+00	1.19E+02	5.76E+02	1.24E-01
2	FWA_DM	9.48E-02	2.54E-01	6.01E+00	1.59E+00	3.72E+02	4.25E-02
3	UMOEAS	0.00E+00	0.00E+00	2.73E+00	3.74E-01	1.44E+02	0.00E+00
4	SOO + BOBYQA	4.90E-02	1.89E+01	8.96E+00	1.30E+02	3.49E+02	0.00E+00
5	SOO	4.90E-02	1.89E+01	8.96E+00	1.30E+02	3.49E+02	0.00E+00
6	RSDE	3.55E-02	6.61E-01	8.52E+00	6.84E+01	2.91E+02	2.21E-01
7	POBL_ADE	1.63E-01	7.81E+00	7.63E+00	1.53E+02	2.08E+02	2.69E-01
8	FERDE	1.88E-02	0.00E+00	5.64E+00	3.67E-02	7.55E+01	1.23E-01
9	FCDE	1.96E-01	1.61E+01	2.10E+01	2.92E+02	6.61E+02	7.73E-01
10	DE_b6e6rlwithrestart	1.69E-02	0.00E+00	4.90E+00	1.23E-03	1.97E+02	2.93E-01
11	CMLSP	0.00E+00	2.07E+00	1.66E+00	1.96E+02	1.53E+02	3.03E-02
12	GaAPADE	3.16E-03	0.00E+00	3.38E+00	1.52E-01	1.83E+02	1.40E-01
13	OptBees	1.56E-01	1.16E-13	2.08E+01	2.19E+02	3.93E+02	1.30E-01
14	LSHADE	3.04E-03	0.00E+00	2.35E+00	8.57E-03	3.21E+01	6.82E-02
15	RMA_LSCh_CMA	0.00E+00	0.00E+00	3.32E+00	7.68E+00	2.01E+01	1.65E-02
16	MVMO	1.86E-02	6.69E-15	3.49E+00	2.14E+00	9.63E+01	4.22E-02
17	PB3C	6.64E-02	1.99E+00	1.79E+00	2.50E-01	2.98E+00	8.41E-03
18	HGAB3C	1.36E-01	0.00E+00	3.54E+00	0.00E+00	6.13E+01	6.94E-04
S. no.	Algorithm	F13	F14	F15	F16	F17	F18
1	NRGA	1.58E-01	2.54E-01	1.02E+00	2.75E+00	1.61E+04	7.42E+03
2	FWA+DM	1.21E-01	2.14E-01	7.75E-01	1.76E+00	2.55E+02	2.52E+01
3	UMOEAS	9.44E-03	1.10E-01	6.67E-01	1.53E+00	8.48E+00	7.84E-01
4	SOO + BOBYQA	3.00E-02	1.30E-01	4.20E-01	2.52E+00	4.23E+02	3.95E+03
5	SOO	3.00E-02	1.30E-01	4.40E-01	2.52E+00	3.12E+06	1.29E+04
6	RSDE	1.28E-01	1.36E-01	9.83E-01	2.23E+00	4.77E+01	2.00E+00
7	POBL_ADE	1.31E-01	2.60E-01	7.12E-01	1.41E+00	2.57E+02	3.32E+01
8	FERDE	1.16E-01	9.36E-02	6.73E-01	1.53E+00	8.23E+00	2.73E+00
9	FCDE	3.56E-01	3.47E-01	1.59E+00	3.19E+00	3.03E+02	2.27E+01
10	DE_b6e6rlwithrestart	1.28E-01	1.11E-01	8.32E-01	1.87E+00	1.40E+00	6.21E-01
11	CMLSP	2.73E-02	1.89E-01	8.97E-01	1.56E+00	3.13E+02	3.09E+01
12	GaAPADE	6.01E-02	9.42E-02	6.06E-01	1.98E+00	9.91E+00	2.23E-01
13	OptBees	4.16E-01	3.69E-01	2.44E+00	2.64E+00	6.84E+02	3.35E+01
14	LSHADE	5.16E-02	8.14E-02	3.66E-01	1.24E+00	9.77E-01	2.44E-01
15	RMA_LSCh_CMA	3.29E-02	1.27E-01	4.72E-01	1.05E+00	7.83E+01	5.22E+00
16	MVMO	3.55E-02	8.91E-02	4.35E-01	1.45E+00	9.36E+00	7.83E-01
17	PB3C	1.69E-02	5.68E-04	3.32E-01	1.93E-01	2.43E+01	2.52E+00
18	HGAB3C	2.54E-02	5.67E-03	5.36E-01	1.15E+00	9.50E+00	3.68E+00

Table 2. Contunied.

S. no.	Algorithm	F19	F20	F21	F22	F23	F24
1	NRGA	2.09E+00	1.72E+03	4.82E+03	3.76E+01	3.29E+02	1.31E+02
2	FWA_DM	1.30E+00	1.34E+01	9.46E+01	3.41E+01	3.30E+02	1.27E+02
3	UMOEAS	2.00E-01	3.71E-01	5.40E-01	2.45E-01	3.30E+02	1.08E+02
4	SOO + BOBYQA	5.50E-01	6.93E+03	1.94E+03	1.27E+02	2.00E+02	1.16E+02
5	SOO	5.50E-01	9.36E+03	2.47E+04	1.27E+02	2.00E+02	1.16E+02
6	RSDE	1.03E+00	7.22E-01	1.21E+00	1.17E+01	3.30E+02	1.19E+02
7	POBL_ADE	2.09E+00	1.26E+01	1.03E+02	3.00E+01	3.29E+02	1.24E+02
8	FERDE	5.09E-01	1.70E+00	8.54E+00	3.24E+00	3.30E+02	1.15E+02
9	FCDE	2.46E+00	1.78E+01	1.48E+02	2.75E+01	3.30E+02	1.37E+02
10	DE_b6e6rlwithrestart	1.42E-01	5.59E-02	7.87E-01	1.54E-01	3.30E+02	1.12E+02
11	CMLSP	1.25E+00	1.99E+01	3.64E+01	8.95E+01	2.02E+02	1.10E+02
12	GaAPADE	2.57E-01	4.32E-01	5.09E-01	3.25E+00	3.30E+02	1.09E+02
13	OptBees	9.33E-01	8.96E+00	5.71E+01	1.70E+01	2.72E+02	1.37E+02
14	LSHADE	7.73E-02	1.85E-01	4.08E-01	4.41E-02	3.30E+02	1.08E+02
15	RMA_LSCh_CMA	7.66E-02	8.06E+00	4.93E+01	8.48E+00	3.30E+02	1.08E+02
16	MVMO	1.58E-01	3.13E-01	1.94E+00	2.63E-01	3.30E+02	1.09E+02
17	PB3C	2.75E-01	1.52E+00	3.32E+00	1.76E+00	2.75E+02	1.06E+02
18	HGAB3C	1.55E-01	1.78E-02	3.88E-02	1.42E-01	3.29E+02	1.13E+02
S. no.	Algorithm	F25	F26	F27	F28	F29	F30
1	NRGA	1.84E+02	1.00E+02	2.81E+02	4.77E+02	4.13E+02	1.73E+03
2	FWA_DM	1.79E+02	1.00E+02	3.21E+02	3.47E+02	2.12E+02	3.94E+02
3	UMOEAS	1.26E+02	1.00E+02	2.55E+01	3.13E+02	1.96E+02	2.34E+02
4	SOO + BOBYQA	1.39E+02	1.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02
5	SOO	1.45E+02	1.00E+02	2.00E+02	2.00E+02	2.00E+02	2.00E+02
6	RSDE	1.30E+02	1.00E+02	9.13E+01	3.87E+02	2.13E+02	5.05E+02
7	POBL_ADE	1.86E+02	1.00E+02	2.56E+02	4.23E+02	3.55E+05	6.38E+02
8	FERDE	1.36E+02	1.00E+02	3.66E+02	3.66E+02	3.18E+02	5.35E+02
9	FCDE	1.84E+02	1.00E+02	4.75E+01	4.57E+02	3.41E+04	8.67E+02
10	DE_b6e6rlwithrestart	1.21E+02	1.00E+02	6.16E+01	3.63E+02	2.18E+02	4.67E+02
11	CMLSP	1.28E+02	1.00E+02	4.11E+01	2.80E+02	2.00E+02	2.16E+02
12	GaAPADE	1.64E+02	1.00E+02	8.97E+01	3.83E+02	2.22E+02	4.67E+02
13	OptBees	1.46E+02	1.00E+02	7.42E+00	3.07E+02	2.20E+02	3.89E+02
14	LSHADE	1.33E+02	1.00E+02	5.81E+01	3.81E+02	2.22E+02	4.65E+02
15	RMA_LSCh_CMA	1.75E+02	1.00E+02	1.85E+02	3.89E+02	2.27E+02	5.85E+02
16	MVMO	1.16E+02	1.00E+02	1.72E+01	3.61E+02	1.81E+02	4.92E+02
17	PB3C	1.01E+02	1.00E+02	1.54E-01	3.57E+02	2.28E+02	4.97E+02
18	HGAB3C	1.32E+02	1.00E+02	1.31E-01	3.75E+02	2.51E+02	4.88E+02

ranked at number 3 with each giving a unique best performance for 2 of the 30 functions. With one unique best performance each, DE_b6e6rlwithrestart, CMLSP, SOO, SOO+BOBYQA, and MVMO are together ranked thereafter.

Table 4 presents the convergence times for the proposed HGAB3C algorithm. For function no. 10, the proposed algorithm proves to be the fastest. It converged to the target error of less than 10^{-9} value of objective function in an average of 346.44 generations in 65.4 s mean time only, which no other algorithm could match. Function 10 is a multimodal, separable function with a large number of local optima. The second better local optimum is far from the global optimum. The GA and BB-BC both are good at evaluating optima for unimodal as well as multimodal topologies. Furthermore, we used BB-BC specifically for enhancing the local search ability and hence HGAB3C, which is a hybrid of the two, was able to explore the optimum for function no. 10 with the best possible speed. Table 5 shows the experimental results of 20 runs of function no. 10 with varying GA and BB-BC generations. It clearly indicates that if the number of iterations in the GA and BB-BC phase is 50 the HGAB3C converges faster. However, if the number of generations in any of the constituting algorithms is reduced, the convergence rate is reduced.

Table 3. Performance comparison of HGAB3C with 17 other algorithms on CEC-14 bench mark functions.

S. No.	Algorithm	Unmatched best	Equal best	Overall best	Functions with best performance	C1	C2	C3	C4
1	PB3C	7	1	8	5, 11, 14, 15, 16, 24, 25, 26	X	5	X	3
2	HGAB3C	4	3	7	3, 8, 10, 20 ,21, 26, 27	1	2	2	2
3	UMOEAS	2	9	11	1, 2, 3, 4, 6, 7, 8, 12, 13, 26 ,29	3	6	X	2
4	LSHADE	2	5	7	1, 2, 3, 8, 19, 22, 26	3	1	2	1
5	DE_b6e6rlwithrestart	1	6	7	1, 2, 3, 6, 8, 17, 26	3	2	1	1
6	CMLSP	1	4	5	2, 4, 7, 9, 26	1	3	X	1
7	SOO	1	4	5	12, 23, 26, 28, 30	X	1	X	4
8	SOO+BOBYQA	1	4	5	12, 23, 26, 28, 30	X	1	X	4
9	MVMO	1	3	4	3, 7, 18, 26	1	1	1	1
10	GaAPDE	X	5	5	1, 2, 3, 8, 26	3	1	X	1
11	RMA-LSCh-CMA	X	5	5	1, 2, 7, 8, 26	2	2	X	1
12	RSDE	X	4	4	1, 2, 3, 26	3	X	X	1
13	FCDE	X	4	4	1, 2, 3, 26	3	X	X	1
14	OptBees	X	2	2	7, 26	X	1	X	1
15	NRGA	X	1	1	26	X	X	X	1
16	FWA-DM	X	1	1	26	X	X	X	1
17	POBL_ADE	X	1	1	26	X	X	X	1
18	FERDE	X	1	1	26	X	X	X	1

Table 4. Convergence performance of HGAB3C.

Function number	Average generations	Minimum generations	Maximum generations	Average convergence time
3	1782.44	800	3686	2.74E+02
4	35480	4656	100000	5.16E+03
8	284.64	200	397	4.20E+01
10	346.44	230	488	6.54E+01

Table 5. Convergence performance of HGAB3C for function no. 10.

Number of generations in GA phase	Number of generations in BB-BC phase	Average convergence time (s)	Increase in convergence time	Average iterations required for convergence	Increase in required iterations
50	50	47.05	-	275.7	-
100	1	540.64	11.5 times	2.33E+03	8.4 times
1	100	689.75	14.7 times	6.31E+03	22.9 times

It is observed that each algorithm does not perform equally well in all four categories of functions [5]. Category 1 (C1) is a group of unimodal functions and has the following three functions: rotated high conditioned elliptic function, rotated bent cigar function, and rotated discus function. Many algorithms, i.e. UMOEAS, LSHADE, DE_b6e6rlwithrestart, GaAPDE, RSDE, and FCDE, have achieved the target in this entire category. This implies there are many options to choose from if the objective function belongs to the category of unimodal functions. In category 2 (C2) of 13 simple multimodal functions, UMOEAS, PB3C, and CMLSP provide the best solution for 6, 5, and 3 functions, respectively, and are followed by HGAB3C with the best performance for 2 functions in this category. Readers may refer to [5] for a list of functions, their categories, and names. In category 3 (C3) with 6 functions in all, no algorithm is able to perform best for more than 2 functions. HGAB3C and LSHADE performed best for 2 functions in C3 each, followed by DE_b6e6rlwithrestart and MVMO with the best performance for 1 function each. From the above discussion, it is evident that HGAB3C has performed best for 1 or 2 functions of each category, which implies that it can be applied to problems with varied mathematical properties.

4. Conclusions

This paper has proposed a new hybrid optimization algorithm named HGAB3C. The algorithm was implemented in MATLAB and its performance was evaluated on the CEC-2014 test suite. The performance of HGAB3C was compared with 17 other algorithms found in the literature. Among the 17 other algorithms, it performed best in 7 of the 30 benchmark functions, ranking third overall. Out of the 7 best performing functions, HGAB3C gave an unmatched best performance for 4 of the functions. HGAB3C obtained an unmatched best performance for function numbers 10, 20, 21, and 27 with mean errors of 0, 1.78E-02, 3.88E-02, and 1.31E-01, respectively, beating the earlier best of 1.23E-03, 5.59E-02, 4.08E-01, and 1.54E-01, respectively. For the unique best performance measure, the proposed algorithm ranks second, superseded by the PB3C algorithm. In the case of the other 3 functions that gave best performances as well, a few other algorithms have also performed equally well. For instance, for function number 3, UMOEAS, LSHADE, DE_b6e6rlwithrestart, MVMO, GaAPDE, RSDE, and FCDE were also able to achieve the same target value of the objective function as achieved by HGAB3C. Similarly for function number 8, along with HGAB3C, LSHADE, DE_b6e6rlwithrestart, GaAPDE, and RMA-LSCh-CMA also achieved the set target. Almost all algorithms were found to perform equally well for function number 26. As far as convergence time is concerned, for the shifted Schwefel function (function number 10), no other algorithm except HGAB3C could converge and achieve the mean target error of less than 10^{-9} in an average of 346.44 generations only.

References

[1] Kumar S, Walia SS, Singh A. Parallel big bang-big crunch algorithm. International Journal of Advanced Computing 2013; 46 (3): 1330-1334.

[2] Zhu X, Luo W, Zhu T. An improved genetic algorithm for dynamic shortest path problems. In: IEEE 2014 Congress on Evolutionary Computation; Beijing, China; 2014. pp. 2093-2100. doi: 10.1109/CEC.2014.6900496

[3] Chen L, Zheng Z, Liu HL, Xie S. An evolutionary algorithm based on covariance matrix learning and searching preference for solving CEC 2014 benchmark problems. In: IEEE 2014 Congress on Evolutionary Computation; Beijing, China; 2014. pp. 2672-2677. doi: 10.1109/CEC.2014.6900594

[4] Elsayed SM, Sarker RA, Essam DL, Hamza NM. Testing united multi-operator evolutionary algorithms on the CEC2014 real-parameter numerical optimization. In: IEEE 2014 Congress on Evolutionary Computation; Beijing, China; 2014. pp. 1650-1657. doi: 10.1109/CEC.2014.6900308

[5] Elsayed SM, Sarker RA, Essam DL. United multi-operator evolutionary algorithms. In: 2014 IEEE Congress on Evolutionary Computation; Beijing, China; 2014. pp. 1006-1013. doi: 10.1109/CEC.2014.6900237

[6] Erlich I, Rueda JL, Wildenhues S, Shewarega F. Evaluating the mean-variance mapping optimization on the IEEE CEC 2014 test suite. In: 2014 IEEE Congress on Evolutionary Computation; Beijing, China; 2014. pp. 1625-1632. doi: 10.1109/CEC.2014.6900516

[7] Erlich I, Rueda JL, Wildenhues S, Shewarega F. Solving the IEEE-CEC 2014 expensive optimization test problems by using single-particle MVMO. In: 2014 IEEE Congress on Evolutionary Computation; Beijing, China; 2014. pp. 1084-1091. doi: 10.1109/CEC.2014.6900517

[8] Molina D, Lacroix B, Herrera F. Influence of regions on the memetic algorithm for the CEC'2014 Special Session on Real-Parameter Single Objective Optimisation. In: 2014 IEEE Congress on Evolutionary Computation; Beijing, China; 2014. pp. 1633-1640. doi: 10.1109/CEC.2014.6900536

[9] Hu Z, Bao Y, Xiong T. Partial opposition-based adaptive differential evolution algorithms: evaluation on the CEC 2014 benchmark set for real-parameter optimization. In: 2014 IEEE Congress on Evolutionary Computation; Beijing, China; 2014. pp. 2259-2265. doi: 10.1109/CEC.2014.6900489

[10] Polakova R, Tvrdik J, Bujok P. Controlled restart in differential evolution applied to CEC2014 benchmark functions. In: 2014 IEEE Congress on Evolutionary Computation; Beijing, China; 2014. pp. 2230-2236. doi: 10.1109/CEC.2014.6900632

[11] Tanabe R, Fukunaga AS. Improving the search performance of SHADE using linear population size reduction. In: 2014 IEEE Congress on Evolutionary Computation; Beijing, China; 2014. pp. 1658-1665. doi: 10.1109/CEC.2014.6900380

[12] Li Y, Tian X, Jiao L, Zhang X. Biclustering of gene expression data using particle swarm optimization integrated with pattern-driven local search. In: 2014 IEEE Congress on Evolutionary Computation; Beijing, China; 2014. pp. 1367-1373. doi: 10.1109/CEC.2014.6900323

[13] Bujok P, Tvrdik J, Polakova R. Evaluating the performance of SHADE with competing strategies on CEC 2014 single-parameter test suite. In: IEEE 2016 Congress on Evolutionary Computation; Vancouver, Canada; 2016. pp. 5002-5009. doi: 10.1109/CEC.2016.7748322

[14] Yavuz G, Aydin D, Stützle T. Self-adaptive search equation-based artificial bee colony algorithm on the CEC 2014 benchmark functions. In: IEEE 2016 Congress on Evolutionary Computation; Vancouver, Canada; 2016. pp. 1173-1180. doi: 10.1109/CEC.2016.7743920

[15] Mitchell M. An Introduction to Genetic Algorithms. A Bradford Book, Cambridge, MA, USA: MIT Press, 1998.

[16] Erol OK, Eksin I. A new optimization method: big bang - big crunch. Advances in Engineering Software 2006; 37 (2): 106-111.

[17] Kripka M, Kripka RM. Big crunch optimization method. In: Rio de Janeiro International Conference on Engineering Optimization; Rio de Janeiro, Brazil; 2008. pp 1-5.

[18] Kumar S, Singh A, Walia S. Parallel big bang big crunch global optimization algorithm: performance and its applications to routing in WMNs. Wireless Personal Communications 2018; 100 (4): 1601-1618.

[19] Liang JJ, Qu BY, Suganthan PN. Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization. Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore. 2013.