Journal of AI and Data Mining
Vol 8, No 3, 2020, 439-449.
DOI: 10.22044/JADM.2019.7351.1875

Coordinate Exhaustive Search Hybridization Enhancing Evolutionary Optimization Algorithms

O. K. Erol[1],I. Eksin[2], A. Akdemir[3] and A. Aydınoglu[4]

1, 2, 4. Istanbul Technical University, Electric-Electronics Faculty, Control and Automation Dept., Maslak, Sariyer, Turkey.
3. Bogazici University, Engineering Faculty, Computer Engineering Dept., Bebek, Besiktas, Turkey.

Received 09 August 2018; Revised 08 February 2019; Accepted 02 March 2019
**Corresponding author:okerol@itu.edu.tr (O. K. Erol).*

Abstract
In general, all the hybridized evolutionary optimization algorithms use the routine "*first diversification and then intensification*" approach. In other words, these hybridized methods all begin with a global search mode using a highly random initial search population, and then switch to an intense local search mode at some stage. The population initialization is still a crucial point in the hybridized evolutionary optimization algorithms since it can affect the convergence speed and the quality of the final solution. In this work, we introduce a new approach by creating a paradigm shift that reverses the "diversification" and then "intensification" routines. Here, instead of starting with a random initial population, we first find a unique starting point by conducting an initial exhaustive search based on the coordinate exhaustive search local optimization algorithm only for a single-step iteration in order to collect a rough but some meaningful knowledge about the nature of the problem. Thus our main assertion is that this approach will ameliorate the convergence rate of any evolutionary optimization algorithm. In this work, we illustrate how one can use this unique starting point in the initialization of two evolutionary optimization algorithms including but not limited to the Big Bang-Big Crunch optimization and the Particle Swarm Optimization. The experiments performed on a commonly used benchmark test suite, which consists of mainly rotated and shifted functions, show that the proposed initialization procedure leads to a great improvement for the above-mentioned two evolutionary optimization algorithms.

Keywords: *Coordinate Exhaustive Search, Evolutionary Computation, Big Bang-Big Crunch Optimization Algorithm, Particle Swarm Optimization Algorithm, Hybridization, A-priori Knowledge Utilization.*

1. Introduction

There exist numerous optimization methods ranging from synthetically invented ones such as [1, 2] to nature-inspired complex algorithms [3-5]. Every algorithm, either synthetically invented or nature-inspired, tries to find one unique solution, which can be a number or a parameter that minimizes or maximizes a given function called the objective/cost/fitness function. If the problem involves some constraints, then the constrained optimization methods are used [6]. It is a well-known fact that there is no optimization method available to cope with every kind of objective function with the highest performance. However, there are algorithms that can be considered as "more general" than the others since they provide a reasonable computation power over a wide set of problems. The power of an algorithm is generally tested on a specific set of objective functions such as the ones given in the Congress on Evolutionary Computation [7], which is organized each year, where a vast number of shifted rotated functions possessing many local optima are used in the given benchmark test function bed.

It is expected that especially the evolutionary optimization algorithms find a global optimal point without being trapped by some local optima. A

common feature of these global optimization algorithms is that they always use some form of probabilistic function or operator. These algorithms may easily be stuck into the local optima if any randomness or probabilistic feature such as cross-over or mutation in Genetic Algorithms (GAs) or heating in Simulated Annealing is not included in the procedure. However, the inclusion of randomness or probabilistic features may create an extra complexity in the algorithms. For instance, the cross-over and mutation probability selection induces two extra parameters in GAs or the number of new parameters that has been adjusted or selected by the designer may go up to 20 or more as it is in Covariance Matrix Adaptation-Evolutionary Strategies (CMA-ESs) [8, 9]. In practice, when an algorithm possesses a high number of adjusting parameters, it naturally becomes more difficult to find a correct set of parameters, and thus more debatable favors its success. The algorithm must be simple with a low number of adjusting parameters, as is the case of Big Bang-Big Crunch (BB-BC) optimization and, to a certain extent, in Particle Swarm Optimization (PSO) algorithms [10]. The BB-BC optimization algorithm and its variants give a comparable optimization power in conjunction with many "state of art" algorithms requiring at most two adjusting parameters, which are the size of the universe and convergence rate, respectively [11].

Injecting some problem specific features into the optimization method will have a doping effect on these specific kinds of problems, while deteriorating the "general nature" of the algorithm. For example, the initial members can be generated according to a priori knowledge. If the optimal point is supposed to be within a limited region of the entire search space, then the members of the new generation are tried to be located within this region either at the initialization stage or after the application of mutation operator [12, 13]. Modifying the specifics of an algorithm to a given problem naturally increases the expected performance of the algorithm for that given problem [14]. Moreover, an algorithm for tuning the parameters of the optimization algorithm itself has been presented [15]. Initialization focused evolutionary algorithm performance amelioration techniques have recently been well-summarized [16]. However, in general, it is hard to automate the modification process while preserving the generality.

The crucial role played by the initial population in a population-based heuristic optimization cannot be neglected. It not only affects the search for several iterations but also often has an influence on the final solution. If the initial population itself has some knowledge about the potential regions of the search domain, then it is quite likely to accelerate the rate of convergence of the optimization algorithm. Therefore, population initialization is a crucial task in evolutionary algorithms because it can affect the convergence speed and the quality of the final solution. One of the most interesting methods available for generating the initial population has been suggested in [17], where the initial population has been generated using the opposition-based rule. If it exists, knowledge-based initial population initialization has been suggested in [18].

If no information about the solution is available, then random initialization is the most commonly used method to generate the candidate solutions. The basic idea behind this increase in power is the motto "if an optimal point position is not known, some equally distributed candidates may have a better chance to get located near this optimal point". Although this simple initialization brings amelioration in the performance of the algorithms that follow the initialization, this cannot be considered as "a priori knowledge" utilization; instead, spreading the candidates all over the search space can be viewed as a proof of the "lack of any a priori knowledge".

In the literature, the efforts of using a-priori knowledge by the injection of some form of greedy and local search procedures are found within the evolutionary iterations usually towards the final stages of the algorithm but not at the initialization stage [19-21]. In the study [22], the authors proposed two schemes for the generation of the initial population for the Differential Evolution (DE) algorithm. These schemes are based upon Quadratic Interpolation (QI) and Non-linear Simplex Method (NSM) in conjugation with computer-generated random numbers. There, the idea is to construct a population that is biased towards the optimum solution right from the very beginning of the algorithm. However, initially, a random population is still to be assigned in this hybridized methodology.

In this work, we propose a reverse routine to the conventional "*first diversification and then intensification*" approach in solving the global optimization problems via evolutionary algorithms, and in a sense, introduce a paradigm shift. Instead of forming a random population, a unique and, in some cases, "near or exact optimal starting point" is found

out by conducting a simple Coordinate Exhaustive Search (CES) optimization algorithm, which is a simple discrete version of cyclic coordinate search presented in [23] as part of a deterministic local optimization strategy. One may still generate a pure random initial population or use any initial population forming methodology found in the literature, and later inject this unique candidate solution found using the CES methodology within this initial population. We first applied this reverse routine to BBBC and presented the results obtained in [24] with various CES versions as applied to CEC'05 test bed functions presented in the Congress on Evolutionary Computation-2005, which is composed of rather simpler functions compared to the CEC'17 test bed functions formed at the Congress on Evolutionary Computation-2017. Here, we adopted and modified this idea of the coordinate search as an initial step of the proposed methodology to PSO algorithm without sacrificing the global properties of the method, i.e. we used the point obtained via local CES as the "sub- or near-optimal starting point" in the initialization step of BB-BC optimization algorithm and within the initial random population spread of the PSO algorithm.

In Section 2, we give a brief basic background information over the BB-BC optimization and PSO algorithms, on which the new initialization approach is implemented. The details of the new methodology, which may be seen as initialization or hybridization of global search optimization algorithms (BB-BC and PSO) with a deterministic local search technique (CES), is covered in Section 3. In Section 4, we present the simulation results of the newly proposed exhaustive search initialization method executed on a very broad benchmark test functions and the statistical analysis of the results. Finally, in Section 5, we put forward an overall discussion of the proposed method and its variants.

2. Background for two evolutionary optimization algorithms

2.1. Basics of Big Bang-Big Crunch (BB-BC) optimization algorithm

The basic BB-BC optimization algorithm [25] is inspired by the big bang and big crunch theories on the origin of the universe. In the big bang phase, as the theory implies, a whole new universe is being populated from a unique point. This point is labeled as the center point. The basic idea behind this algorithm is that searching for the optimal solution near the "center of mass" point is beneficial, while keeping the chance to look far beyond this point preserves the global property of the algorithm. In particular, an element of the new population x_i is formed by moving from the center of mass x_c in a random direction u_i as:

$$x_i = x_c + \gamma_s u_i \quad (1)$$

We pick u_i to be distributed according to a Gaussian noise with zero mean and unity variance. The term γ_s, here, represents the explosion strength parameter that we can pick to tune diversification around the center of mass. It is expected that the population members will accumulate around the center point x_c. This "bang phase", which can be viewed as diversification around x_c, implies that the solution is near the center point but its exact location is not known. The initial center point is arbitrary and unknown at the very beginning of the algorithm. If the position of the optimal point can somehow be guessed or known as a priori, then a point within the vicinity of that location can be assigned as the initial center point. However, if no such information is available, then a random initial center point is assigned and the population is tried to be spread over the entire universe or the search space.

The crunching phase can be viewed as the survival of the best as in GAs. The superior feature of this step is that it generates a new member different from those existing in the population. This step is accomplished using the weighted average operation given in Eq. 2.

$$x_c = \frac{\sum f_i \, x_i}{\sum f_i} \quad (2)$$

where, the weights of the candidate points are their respective fitness measures labeled as f_i. As the value for f_i gets higher, its capability to attract the center point increases. The successive calling of the "bang" and "crunch" phases will conduct a thorough search to find the optimal point and the two equations 1 and 2 form a maximization operation by nature. One "bang phase" together with one "crunch phase" forms one step of the iteration.

A modification to the proposed algorithm can include an enforced localization around the center point by decreasing the diversification factor. This can be done by inserting the term β into Equation 1 given above, as follows:

$$x_i = x_c + \frac{\gamma_s u_i}{\beta} \quad (3)$$

In this case, as the iterations evolve, the algorithm will intensify its search around a specific point at the expense of spending more time around this point.

2.2 Basics of PSO Algorithm

PSO is a population-based stochastic optimization method introduced in [10], being inspired by the study on the bird flocking behavior. In the PSO algorithm, the potential solutions called particles are flown in the problem hyperspace. A particle changes its position with time, and this change of position of a particle is called velocity. In PSO, particles update their positions with an internal velocity, while keeping their own best values so as to use it and share it with the other particles. More vividly, PSO is initialized with a group of random particles, and then it searches for a position with an optimum fitness value. In every iteration, each particle updates its position based on the best position it has achieved so far, i.e. the position with the highest fitness, denoted by p_k^i, and the global best position achieved so far, i.e. the position with the highest fitness value among all particles, denoted by p_k^g. After finding the best two values, the entire particle updates its velocity and positions with the following equation:

$$v_{k+1}^i = w v_k^i + c_1 u(p_k^i - x_k^i) + c_2 u(p_k^g - x_k^i) \quad (4)$$

Particle i updates its velocity at iteration k+1, v_{k+1}^i as a weighted combination of its past velocity v_k^i, its local best solution p_k^i and the global best solution p_k^g, and x_k^i being the current position of agent i at iteration k. Moreover, the parameters c_1 and c_2 are constant weights, and u is a uniform random variable between 0 and 1. Larger values of the inertia constant w emphasizes on a global search for a particle, and smaller values for w mean a more localized search. This constant is usually determined by the maximum number of iterations or can be adapted with respect to the current iteration number. The parameter c_1 is called the cognitive constant and c_2 is the social constant. Note that for the velocity update above, each particle needs to measure the fitness value of itself and keeps the position that has yielded the best fitness value up to iteration $k + 1$ in its memory. Then the velocity of particle at iteration $k + 1$ determines the position p of the particle at iteration $k + 1$, as follows:

$$p_{k+1}^i = p_k^i + v_{k+1}^i \quad (5)$$

The initial positions of particles in the problem space is random at iteration $k = 0$. Moreover, there exist other versions of PSO algorithms that differ in the extent of the social information exchange, namely Individual Best Algorithm, Global Best Algorithm, and Local Best Algorithm.

3. Coordinate exhaustive search initialization to evolutionary optimization algorithms

In this section, we first introduce the basic principles of the Coordinate Exhaustive Search (CES) algorithm, and then provide a perspective of the initializing application of CES either as the unique starting point in the BB-BC optimization algorithm or as a seed in the PSO algorithm.

3.1. Coordinate Exhaustive Search (CES) Algorithm

Coordinate descent algorithms are non-linear iterative methods used for minimizing an objective function $f: X \to \mathcal{R}$. Here, the set X is a Cartesian product of the sets $X_1, \dots, X_N$ and X is a set in $\mathcal{R}^N$. Let x denote a generic point in this set, that is $x \in X \subseteq \mathcal{R}^N$. The value for x at its i^{th} coordinate is denoted by $x_i \in [xi_{min}, xi_{max}] \ \forall i = 1, \dots,$; here, xi_{min} and xi_{max} are the lower and upper limits of the related coordinates, respectively. In these algorithms, we fix all of the components of x to some value, except for the i^{th} component, and then minimize $f(x)$ with respect to x_i. This procedure is repeated over all components leading to an iterative algorithm.

There are two types of coordinate descent algorithms. In the first type, we minimize each component simultaneously, i.e. let $x^0 = [x_1^0, \dots, x_N^0] \in X$ be the initial value, and then the final configuration $x^* = [x_1^*, \dots, x_N^*] \in X$ is given by solving the following optimization problems:

$$x_i^* = argmin\, f(x_1^0, \dots, x_{i-1}^0, x_i, x_{i+1}^0, \dots x_N^0) \quad \forall i = 1, \dots, N \qquad x_i \in X_i \quad (6)$$

In the initialization step proposed in this work, we first discretize the coordinate space X if it is continuous and then apply the aforementioned algorithm to find the initial point of an evolutionary algorithm. The discretization of component i involves dividing the space X_i into m equal points. Note that given the finite set of possible values for each component, the minimization in the coordinate exhaustive search algorithms in Equations 1 and 2 can simply be done by evaluating the objective function f for each element of the finite set and picking the component value to be the one that yields the minimum objective over the m evaluations. The illustration of how the final best point is formed for initialization approach CES is given in figure 1 in a 2D Euclidean space for the sake of visualization.

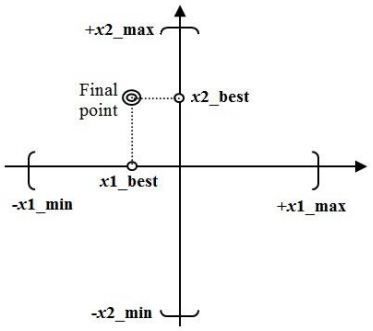

Figure 1. Illustration of Coordinate Exhaustive Search (CES).

3.2. Application of Coordinate Exhaustive Search (CES) Initialization to Plain BB-BC Optimization and PSO Algorithms

In this work, the local and global search preferences or diversification and intensification modes are reversed. Almost all the evolutionary computation techniques start with a highly diversified phase and then eventually switch to a local search after a certain convergence to the solution point. This switched status can be seen as an intensification phase of the algorithm. Here, contrary to the conventional or routine approach, we start with conducting a certain exhaustive search in a very gross manner, then switch to the diversification mode, and continue with the basic evolutionary algorithm.

Firstly, we apply the discrete version of simple CES algorithm for only one step or one shot to provide a base as an initialization seed to the basic BB-BC optimization algorithm. The BB-BC optimization algorithm is conducive to an initialization from a single point as the big-bang phase starts with a single point called the "center of mass". That is we select the point x^* found at the end of one shot coordinate exhaustive search as the initial center of mass x_c of the BB-BC algorithm. Next, we call the algorithm CES-BB-BC. When the exhaustive search is conducted on standard coordinate axes (the one that passes through the origin), the global optimum point may be found even at this initialization step without much effort since the origin is the solution point for some of the test function considered here. Next, we have applied CES methodology to another well-known evolutionary algorithm, which is the PSO algorithm, and that algorithm is called CES-PSO. The injection of the near optimal point x^* to any evolutionary search algorithm can be accomplished in many different ways. Here, we have assigned the near optimal point x^* found out at the CES stage to be the g_{best} before launching the PSO algorithm. One may also assign this near optimal point x^* calculated at the CES stage to some percentage of the initial population. We present the general flowchart of CES-BB-BC and CES-PSO in figure 2.

We can remark that there exists a natural conformity between these proposed coordinate exhaustive initialization algorithms and BB-BC evolutionary algorithm by itself since BB-BC algorithm makes its start from a unique point, namely "center of mass point" between two successive iterations, and the initialization stage provides the so-called best starting point. Thus if one wants to use this methodology with the other evolutionary algorithms, then one would have to take certain precautions to protect this valuable initial point in order not to get lost among the high number of individuals that has been formed randomly. A possible implementation can be accomplished by setting a portion of the initial members within the generation to this valuable point found by CES.

4. Simulation results of CES initialization approach on plain BBBC and PSO algorithms

4.1. Benchmark functions of CEC'17 and performance measures

We tested the new initialization approach on the basic BBBC and PSO evolutionary optimization algorithms using 15 functions selected from CEC'17, which are mainly composed of shifted, rotated, and hybrid functions. A complete list of these functions with their dimension and search space ranges together with their respective stopping criteria (Value To Reach or VTR) is given in table 1 for CEC'17.

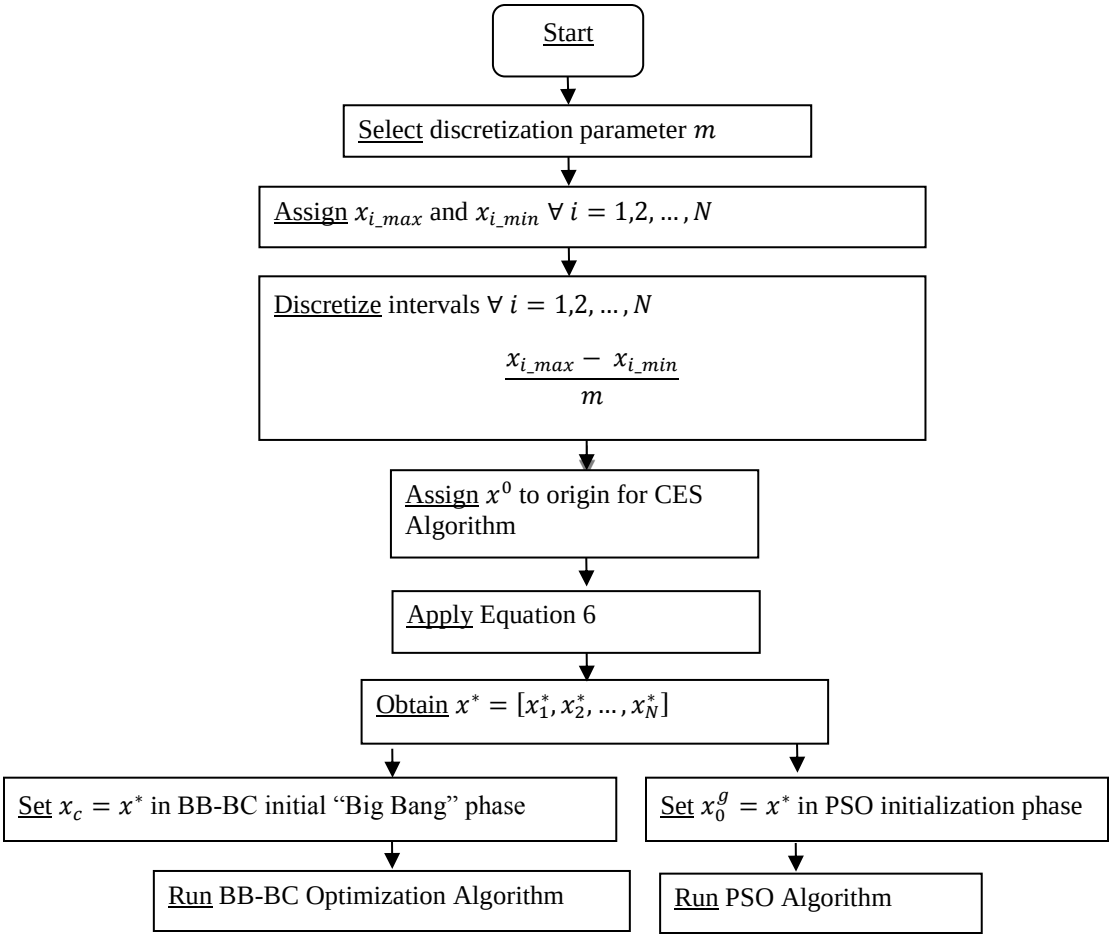

Figure 2. The flowchart of CES-BB-BC optimization and CES-PSO algorithms.

Table 1. List of CEC'17 benchmark functions.

Code	Function name	Dimensionality	Search Space	Value To Reach-VTR value
f1	Shifted and Rotated Sum of Different Power Function	10	[-100, 100]	200
f2	Shifted and Rotated Zakharov Function	10	[-100, 100]	300
f3	Shifted and Rotated Rosenbrock's Function	10	[-100, 100]	400
f4	Shifted and Rotated Rastrigin's Function	10	[-100, 100]	500
f5	Shifted and Rotated Expanded Schaffer's F6 Function	10	[-100, 100]	600
f6	Shifted and Rotated Lunacek Bi_Rastrigin Function	10	[-100, 100]	700
f7	Shifted and Rotated Non-Continuous Rastrigin's Function	10	[-100, 100]	800
f8	Shifted and Rotated Levy Function	10	[-100, 100]	900
f9	Hybrid Function 1 (N=3)	10	[-100, 100]	1100
f10	Hybrid Function 4 (N=4)	10	[-100, 100]	1400
f11	Hybrid Function 5 (N=4)	10	[-100, 100]	1500
f12	Hybrid Function 6 (N=4)	10	[-100, 100]	1600
f13	Hybrid Function 6 (N=5)	10	[-100, 100]	1700
f14	Hybrid Function 6 (N=5)	10	[-100, 100]	1900
f15	Hybrid Function 6 (N=6)	10	[-100, 100]	2000

The selected PSO start-up parameters are given in table 2. The start-up parameter selection is carried on according to the suggestions given in MATLAB code.

Table 2. PSO start-up values.

Parameter Name	Value
Bird in swarm	15
Number of quality in Bird	Dimension value from Table 1&2
Min-Max Range	Values from Table 1&2
Food availability	Objective function
Availability type	'min'
Velocity clamping factor	2
Cognitive constant (c_1)	2
Social constant (c_2)	2
Min Inertia weight	0.4
Max Inertia weight	0.9
Max iteration	100,000

The CEC'17 test suite has been run with 50×50 independent runs and 100 individuals within each generation. We run the algorithm for 50 times for each function, calculate the number of function calls for each run, and repeat this process for 50 times. Repeating the process for 50 times enables us to be more confident about the results. The statistical analysis section is devoted for showing the robustness of our results. The average of the number of function calls is taken at the end of all independent runs. The VTR values are the stopping criteria for each case, and they are set close enough to the respective optimum. We have tried to set the parameters of the evolutionary optimization methods as they are used in the competitions; however, this is not a limiting factor. Here, our main concern is about how the CES intrusion to the evolutionary optimization algorithms affects the convergence. We first compare the "plain" evolutionary optimization algorithm and CES-induced version of it. Therefore, we do not need to search for "optimal" setup parameters. Moreover, we do not make any change in the setup parameters between the "plain" and "CES-induced" versions of the evolutionary optimization algorithm in order to make a fair comparison.

It is obvious that small numbers for function calls (or Average Number of Function Call–ANFC) indicate a faster convergence; hence, it provides a measure for the success of the related algorithm. In that respect, the "winner" algorithm is the one that acquires the smallest value for the function calls for every test function. The "winner" of ANFCs and the AR/AAR are all highlighted in boldface. At the last row, AAR is given, which is calculated as the average of all AR's. This value gives us a rough idea about the overall improvement of the CES algorithm. For clarity, Amelioration Rates higher than 1 denote improvements over the plain methods. In all of the comparison tables, we have also used two other performance measures defined as "Amelioration Rate-AR" and "Average Amelioration Rate-AAR", which provide a more compact interpretation on the number function calls in comparing the plain and CES-induced versions of the evolutionary optimization methods. These are defined in Eqs. 7 and 8, respectively, as follow, where N_f denotes the number of functions for the selected testbed:

$$\text{AR}= \frac{ANFC\ of\ Original\ Methodology}{ANFC\ of\ CES\ Induced\ Methodology} \quad (7)$$

$$\text{AAR}= \frac{\Sigma_{i=1}^{Nf} AR_i}{number\ of\ functions} \quad (8)$$

4.1. Simulation Results and Statistical Analysis

Next, the simulation results given in table 3 provide us a comparison of the basic BB-BC and CES-BB-BC optimization algorithms on the test suite of CEC'17. It can easily be deduced from the examination of Table 3 that the CES-BB-BC optimization algorithm is more efficient in 12 cases out of 15 when compared to the plain-BBBC. This can be seen from the number of functions with Amelioration Rates higher than 1. However, since the overall success of the approach is more important for all the test function bed, one should consider the overall amelioration rate that is AAR. This shows us that overall, we have around 8% speed gain by incorporating CES into the design.

The net effect of using CES methodology on PSO algorithm on 15 test functions can be seen in Table 4. Note that PSO is limited to 100,000 function evaluation counts due to memory limitations. For the overall reduction of function calls, we again consider the AR values for each function, and AAR appears to be 1.14. This means that overall, reduction in function calls for PSO is 14% and even higher compared to the basic PSO. Evaluation of the improvement in the performance of the BB-BC and PSO optimization algorithms with CES initialization would be more accurate and righteous over the functions that PSO algorithm has "success", i.e. the solution

points that were obtained before reaching the function evaluation count limit of 100,000. For this reason, we have decided to choose 15 functions for which the algorithm terminates in both cases before reaching the upper bound.

Table 3. Comparison of plain BB-BC algorithm and CES-BB-BC on CEC'17 test suite.

Function Code	Average Number of Function Calls-ANFCs		Relative Standard Deviations		CES-BBBC Amelioration Rate-AR
	BB-BC	CES-BBBC	BBBC	CES-BBBC	BB-BC/CES-BBBC
f1	**55372**	56979	0.02	0.02	0,9718
f2	43238	**43230**	0.02	0.02	1,0002
f3	7804	**7661**	0.06	0.05	1,0188
f4	42333	**41189**	0.12	0.12	1,0278
f5	3115	**2437**	0.16	0.04	1,2783
f6	78079	81599	0.05	0.04	0,9569
f7	16385	**16302**	0.13	0.15	1,0051
f8	10187	10206	0.04	0.02	0,9981
f9	30119	**28586**	0.12	0.15	1,0543
f10	81031	**73102**	0.05	0.06	1,1085
f11	98387	**96913**	0.01	0.02	1,0152
f12	53138	**51003**	0.14	0.14	1,0419
f13	50685	**49796**	0.12	0.13	1,0178
f14	31916	**24806**	0.11	0.12	1,2866
f15	44483	**32201**	0.15	0.20	1,3814
AAR	-	-	**-**	**-**	**1.08**

Table 4. Comparison of CES-PSO with classical PSO algorithm on CEC'17 test suite.

Function Code	Average Number of Function Calls-ANFCs		Relative Standard Deviations		CES-PSO Amelioration Rate-AR
	PSO	CES-PSO	PSO	CES-PSO	PSO/CES-PSO
f1	66671	58714	0.02	0.02	1.1355
f2	64986	65170	0.01	0.01	0.9972
f3	16818	16223	0.04	0.07	1.0367
f4	40260	39997	0.04	0.04	1.0066
f5	1504	865	0.11	0.11	1.7379
f6	55198	56844	0.02	0.02	0.9711
f7	27148	27503	0.04	0.06	0.9871
f8	27990	27985	0.02	0.02	1.0002
f9	20333	19347	0.03	0.03	1.0510
f10	41541	45013	0.05	0.04	0.9229
f11	66294	56661	0.04	0.03	1.1700
f12	31747	54941	0.09	0.14	0.5778
f13	19150	12041	0.10	0.08	1.5904
f14	49340	44286	0.05	0.05	1.1141
f15	10964	5903	0.10	0.13	1.8574
AAR	-	-	-	-	1.1437

We have selected the stopping criteria as being in the 5% vicinity of VTR. Thus the algorithm terminates once it gets a value between the range [0.95×VTR, 1.05×VTR]. Here, one should keep in mind that only the relative convergence improvement in the related evolutionary optimization algorithm is tried to be noted because our main concern is to provide an amelioration for every evolutionary search algorithm at the start-up phase by imposing an initial knowledge by the proposed approach. Here, what we introduce is a new methodology that is a hybridization of the local and global search approaches in the operation of any evolutionary global optimization method. Therefore, we have not adopted the official CEC'17 stopping criterion, and we are not in need of comparing this methodology with any other global evolutionary optimization algorithm. We may easily conclude that this simple local search

initialization step for the basic evolutionary BB-BC optimization algorithm leads to about 8% reduction in the total number of function calls on the test bed of CEC'17, whereas the performance amelioration was about 20% for the CEC'05 test bed functions (Erol and Eksin 2017). Moreover, the results related to the CES initialization procedure applied to the basic PSO algorithm reveals that the total number of function calls reduce by 30% and 14% for the CEC'05 (Erol and Eksin 2017) and CEC'17 benchmark functions, respectively.

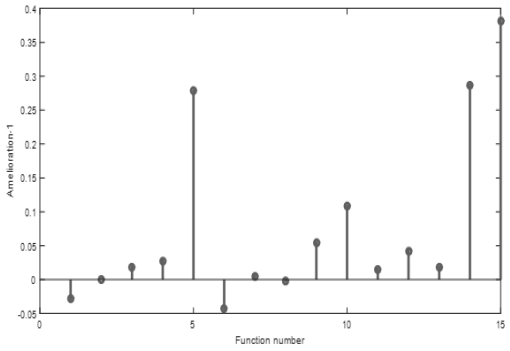

Figure 3(b). Illustration of the amelioration rate values related to PSO for each function shifted to 0.

To be confident about these results, we run the experiment for 50 times, where each run itself consisted of 50 independent trials of each function. Tables 3 and 4 show the statistical data associated with each function for BBBC and PSO, respectively. The relative standard deviation represents the deviation of each trial from the mean for each function. Thus lower values for relative standard deviation are preferable. The tables show that the RSTD values are quite low for all functions.

5. Discussion and conclusion

Almost all the evolutionary algorithms use the "*first diversification and then intensification routine*" approach, i.e. there exists a switch to a local search mode from a global search mode after a certain convergence limit is achieved for the optimum solution point. Here, we mainly propose a paradigm shift by introducing a reversed mode to this conventional approach. In other words, we first execute a rough "intensification" or a local optimization algorithm over the search space in order to get an insight and a-priori information about the function to be optimized, and then

Figures 3(a) and 3(b) illustrate the ameliorations obtained for each function after applying the CES algorithm. Graphs are scaled around 0 instead of 1 so that the positive values indicate improvements and the negative values correspond to a deterioration.

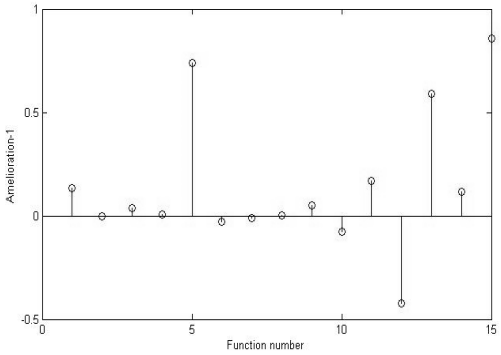

Figure 3(a). Illustration of the amelioration rate values related to BBBC for each function shifted to 0.

switch to the basic "diversification" or global evolutionary search phase. That is, what we propose here, is not a new evolutionary algorithm to compete with the existing ones but a new methodology beginning with a simple "*one-shot local search"* followed by a global search, and it is applicable to any evolutionary global optimization algorithm. Therefore, we only make a relative comparison between the pure global evolutionary search method and the one that uses a-priori deterministic search method, namely Coordinate Exhaustive Search (CES) as an initial hybridization step embedded to it.

In this respect, the idea of CES, which is a primitive local search algorithm, is adopted and modified as an initial step of the plain BB-BC evolutionary optimization method without sacrificing the global property of BB-BC methodology. In fact, there exists a natural conformity between the proposed coordinate exhaustive initialization algorithms and BB-BC optimization algorithm because the BB-BC optimization algorithm makes its start with a unique point called the "center of mass", and single shot CES algorithm provides this particular "best" starting point at the initialization stage. The plain BB-BC optimization method combined and enhanced by CES hybridization is tested on a vast number of test functions with distinct characteristics, which are taken from a widely

accepted benchmark test bed, namely CEC'17 functions. The overall number of function calls decreased up to 8% on the average when compared to the plain BB-BC optimization algorithm.

The same local CES hybridization approach is adapted to the well-known and widely used PSO algorithm. The results related to the CES hybridization procedure applied to the plain PSO algorithm has revealed that the total number of function calls has been reduced by 14% compared to the basic PSO algorithm for the same benchmark test functions.

Even for these heavily shifted, rotated, and compound functions of CEC'17, the proposed local CES hybridization approach as applied to two evolutionary optimization algorithms, namely BBBC and PSO has shown an average performance gain by a factor of more than 8% and 14%, respectively, including the initial search phase. As a final comment, we can say that this reverse mode approach can easily be adapted and applied to all evolutionary optimization algorithms with some slight modifications, and an improvement in the related algorithm is almost inevitable.

References

[1] Hooke, R. & Jeeves T. A. (1961). Direct search" solution of numerical and statistical problems. Journal of the Association for Computing Machinery (ACM), vol. 8, no. 2, pp. 212–229.

[2] Nelder, J. A. & Mead R. (1965). A simplex method for function minimization. Computer Journal 7, pp 308–313.

[3] Holland, J. H. (1975) Adaptation in natural and artificial systems. Ann Arbor, MI: The University of Michigan Press.

[4] Goldberg, D. E. (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley.

[5] Kirkpatrick, S., Gelatt, C. D. & Vecchi M. P. (1983). Optimization by simulated annealing. Science, vol. 220, no. 4598, pp. 671–680.

[6] Bartekas, D. P. (1996). Constrained optimization and Lagrange multiplier methods, Athena Scientific.

[7] Award, N. H., Ali, M. Z., Liang, J. J., Qu, B. Y. & Suganthan, P. N. (2016). Problem definitions and evaluation criteria for the CEC 2017 Special Session and Competition on single objective real-parameter numerical optimization, Tech. Report, Nanyang Technological University.

[8] Hansen, N., Müller, S. D. & Koumoutsakos, P. (2003). Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA–ES), Evolutionary Computation, vol. 11, no. 1, pp. 1–18.

[9] Hansen, N. (2006). The CMA evolution strategy: A comparing review, In J.A. Lozano, P. Larrañga, I. Inza and E. Bengoetxea (eds.). Towards a new evolutionary computation. Advances in estimation of distribution algorithms, pp. 75–102.

[10] Kennedy, J. & Eberhart, R. (1995). Particle swarm optimization, Proceedings of IEEE International Conference on Neural Networks, ISBN: 0-7803-2768-3/95, vol. 4, pp. 1942-1948.

[11] Genc, H. M., Eksin, I. & Erol, O. K. (2013). Big bang-big crunch optimization algorithm with local directional moves, Turk. J. Elec. Eng. & Comp. Sci., vol. 21, no. 5, pp. 1359-1375.

[12] Burke, E. K., Newall, J. P. & Weare, R. F. (1998). Initialization strategies and diversity in evolutionary timetabling, Evolutionary Computation Journal, vol 6.1, pp. 81-103.

[13] Cao, Y. & Wu, Q. H. (1997). Study of initial population in evolutionary programming, Proceedings of the European Control Conference.

[14] Pehlivanoglu, Y. V. (2013). A new particle swarm optimization method enhanced with a periodic mutation strategy and neural networks, IEEE Transactions on Evolutionary Computation, vol. 17, no. 3, pp. 436-452.

[15] Eiben, A. E. & Smit, S. K. (2011). Parameter tuning for configuring and analyzing evolutionary algorithms, Swarm and Evolutionary Computation, Elsevier, vol. 1, no.1, pp. 19-31.

[16] Kazimipour, B., Li, X., Qin, A. K. (2014). A review of population initialization techniques for evolutionary algorithms. IEEE Congress on Evolutionary Computation, DOI: 10.1109/CEC.2014.6900618, pp. 2585, 2592.

[17] Rahnamayan, S., Tizhoosh, H. R., & Salama, M. M. A. (2007). A novel population initialization method for accelerating evolutionary algorithms, Computers and Mathematics with Applications, vol. 53, pp. 1605-1614.

[18] Li, C., Chu, X., Chen, Y., & Xing, L. (2016). A knowledge-based technique for initializing a genetic algorithm. J. Intell. Fuzzy Syst. 2016, vol. 31, pp. 1145–1152.

[19] Chelouah, R. & Siarry, P. (2005). A hybrid method combining continuous tabu search and Nelder-Mead simplex algorithms for the global optimization of multiminima functions, European Journal of Operational Research, vol. 161, pp. 636-654.

[20] Wang, L., Xu, Y. & Li, L. (2011). Parameter identification of chaotic systems by hybrid Nelder-Mead simplex search and differential evolution algorithm, Expert Systems with Applications, vol. 38, pp. 3238-3245.

[21] Zahara, E. & Kao, Y. T. (2009). Hybrid Nelder-Mead simplex search and particle swarm optimization for constrained engineering design problems, Expert Systems with Applications, vol. 36, pp. 3880-3886.

[22] Musrrat A., Pant M. & Ajith A. (2013). Unconventional initialization methods for differential evolution, Applied Mathematics and Computation, vol. 219, no. 9, pp. 4474-4494.

[23] Luenberger D. G. (1984). Linear and Nonlinear Programming. Reading, MA:Addison-Wesley.

[24] Erol O. K. & Eksin I. (2017). Coordinate Exhaustive Search Initialization for Big Bang – Big Crunch Algorithm, Proceedings of ELECO 2017, 10th International Conference on Electrical and Electronics Engineering, Bursa, Turkey.

[25] Erol O. K. & Eksin, I. (2006) .A New Optimization Method: Big-Bang Big-Crunch, Advances in Engineering Software, Elsevier, vol. 37, no. 2, pp. 106-111.

نشریه هوش مصنوعی و داده کاوی

هماهنگی جستجوی جامع و تقویت هیبریداسیون الگوریتم‌های بهینه‌سازی تکاملی

A. Aydınoglu[4] و A. Akdemir[3] ،I. Eksin [2] ،O. K. Erol[*،1]

[۱ و۴] دانشگاه فنی استانبول، دانشکده الکترونیک، گروه کنترل، اتوماسیون،گروه کنترل و اتوماسیون، Maslak، Sariyer، ترکیه.

[۳] دانشگاه بوگازیچی، دانشکده مهندسی، گروه مهندسی کامپیوتر، بیبک، بشیکتاش، ترکیه.

ارسال ۲۰۱۸/۰۸/۰۹؛ بازنگری ۲۰۱۹/۰۲/۰۸؛ پذیرش ۲۰۱۹/۰۳/۰۲

چکیده:

به طور کلی، همه الگوریتم‌های بهینه‌سازی تکاملی ترکیبی از رویکرد معمول "اول متنوع سازی و سپس شدت گرفتن" استفاده می‌کنند. به عبارت دیگر، این روش‌های ترکیبی همه با یک حالت جستجوی جهانی با استفاده از یک جمعیت جستجوی اولیه بسیار تصادفی شروع می‌شوند و سپس در مرحله‌ای به حالت جستجوی محلی شدید تغییر می‌کنند. اولیه سازی جمعیت هنوز یک نکته اساسی در الگوریتم‌های بهینه‌سازی تکاملی ترکیبی است زیرا می‌تواند بر سرعت همگرایی و کیفیت راه حل نهایی تأثیر بگذارد. در این کار، با ایجاد یک تغییر پارادایم که معکوس روال "تنوع" و سپس "تشدید" است ، یک رویکرد جدید را معرفی می‌کنیم. در اینجا، به جای شروع با یک جمعیت اولیه تصادفی، ابتدا با انجام یک جستجوی جامع بر اساس الگوریتم بهینه‌سازی جستجوی محلی جامع، فقط برای یک تکرار تک مرحله‌ای، یک نقطه شروع منحصر به فرد را می‌یابیم تا بتوانیم دانش معنی داری در مورد ماهیت مسئله بدست آوریم. بنابراین ادعای اصلی ما این است که این روش سرعت همگرایی هر الگوریتم بهینه‌سازی تکاملی را بهبود می‌بخشد. در این کار، ما نشان می‌دهیم که چگونه می‌توان از این نقطه شروع منحصر به فرد در آغاز دو الگوریتم بهینه‌سازی تکاملی استفاده کنیم که به بهینه‌سازی Big Bang-Big Crunch و بهینه‌سازی ازدحام ذرات محدود نمی‌شوند. آزمایش‌های انجام شده بر روی یک مجموعه تست استاندارد، که از توابع عمدتا چرخشی و تغییر یافته تشکیل شده است، نشان می‌دهد که روش اولیه سازی پیشنهادی منجر به پیشرفت بزرگی برای دو الگوریتم بهینه‌سازی تکاملی فوق می‌شود.

کلمات کلیدی: هماهنگی جستجوی جامع، محاسبات تکاملی، الگوریتم بهینه‌سازی بحران Big Bang-Big، الگوریتم بهینه‌سازی ازدحام ذرات، هیبریداسیون، استفاده از دانش A-Priori.