Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 135652, 11 pages
http://dx.doi.org/10.1155/2014/135652

Research Article

A Convergent Differential Evolution Algorithm with Hidden Adaptation Selection for Engineering Optimization

Zhongbo Hu,[1,2] Shengwu Xiong,[1] Zhixiang Fang,[3] and Qinghua Su[2]

[1] *School of Computer Science and Technology, Wuhan University of Technology, Wuhan, Hubei 430070, China*
[2] *School of Mathematics and Statistics, Hubei Engineering University, Xiaogan, Hubei 432000, China*
[3] *State Key Laboratory for Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430072, China*

Correspondence should be addressed to Zhongbo Hu; huzbdd@126.com

Received 27 November 2013; Revised 25 January 2014; Accepted 27 February 2014; Published 30 March 2014

Academic Editor: Gongnan Xie

Many improved differential Evolution (DE) algorithms have emerged as a very competitive class of evolutionary computation more than a decade ago. However, few improved DE algorithms guarantee global convergence in theory. This paper developed a convergent DE algorithm in theory, which employs a self-adaptation scheme for the parameters and two operators, that is, uniform mutation and hidden adaptation selection (haS) operators. The parameter self-adaptation and uniform mutation operator enhance the diversity of populations and guarantee ergodicity. The haS can automatically remove some inferior individuals in the process of the enhancing population diversity. The haS controls the proposed algorithm to break the loop of current generation with a small probability. The breaking probability is a hidden adaptation and proportional to the changes of the number of inferior individuals. The proposed algorithm is tested on ten engineering optimization problems taken from IEEE CEC2011.

1. Introduction

Differential evolution (DE) is a population-based stochastic real-parameter algorithm for continuous optimization problems firstly introduced by [1, 2]. Recently, several detailed surveys of advances in DE were conducted (i.e., [3, 4]). These, along with the competitions in the 1996–2012 IEEE *International Conferences on Evolutionary Computation* (CEC), showed that DE is one of the most powerful stochastic optimizers. In the last two decades, many outstanding DE variants have been proposed. Few of these algorithms are based on convergence theory; therefore few of them, like Elite Genetic algorithm, guarantee convergence (all the "converge," "convergent," or "convergence" of this paper mean the convergence in probability) in the theory for any continuous optimization problems regardless of initial populations. However, theoretical studies of stochastic algorithms are attracting the attention of a greater number of researchers. IEEE CEC 2013 held a special session focusing on the theoretical foundations of bioinspired computation. Aiding algorithm design is one of the most important purposes of theoretical studies.

As to the convergence theory researches in DE, two trends are gradually becoming apparent. One is designing improved DE algorithms based on theoretical foundations. Reference [5] performed a mathematical modelling and convergence analysis of continuous multiobjective differential evolution (MODE) using certain simplifying assumptions. This work was extended by [6]. Reference [7] proposed a differential evolution Markov chain algorithm (DE-MC) and proved that its population sequence is a unique joint stationary distribution. Reference [8] presented a convergent DE using a hybrid optimization strategy and a transform function and proved its convergence by the Markov process. Reference [9] presented a DE-RW algorithm that applied a random-walk mechanism to the basic DE variants (the convergence of DE-RW was not proved, but it can be easily proved by Theorem 2 in Section 4). The other trend involves the study of convergence theory as an aid to DE algorithm design. Reference [10] established asymptotic convergence behaviour of a basic DE (DE/rand/1/bin) by applying the concepts of Lyapunov stability theorems. The analysis is

based on the assumption that the objective function has the following two properties: (1) the objective function has the second-order continual derivative in the search space, and (2) it possesses a unique global optimum within the range of search. These researches show that the enhancement of population diversity is an efficient route to the development of globally convergent DE algorithms. However, a difficult task associated with increased diversity is how to remove the extra inferior individuals generated during the evolving process.

This paper proposes a self-adaptive convergent DE algorithm with a hidden adaptation selection, named SaCDEhaS. This paper then proves that the SaCDEhaS guarantees global convergence in probability. The SaCDEhaS algorithm is formed by integrating the basic DE with two extra operators, that is, a uniform mutation operator and a hidden adaptation selection (haS) operator. The parameters' self-adaptation and use of the uniform mutation operator both enhance the diversity of populations. The proposed haS operator automatically eliminates some inferior individuals that are generated as part of enhancing the population diversity. Experimental results and comparison studies of all the bound-constrained and unconstrained optimization problems posed by the CEC 2011 continuous benchmark functions for engineering optimization [11] show that (1) the uniform mutation and haS operators improve algorithm performance, and (2) SaCDEhaS is competitive with the top four DE variants on the CEC 2011 competition.

The rest of this paper is organized as follows. Section 2 briefly introduces the basic DE algorithm. Section 3 describes in detail the proposed algorithm. Section 4 gives a sufficient condition for the convergence of modified DE and proves the global convergence of the proposed algorithm. Numerical experiments are then presented in Section 5, followed by conclusions in Section 6.

2. Basic Differential Evolution Algorithm

DE is arguably one of the most powerful stochastic real-parameter optimization algorithms. And it is used for dealing with continuous optimization problems [12, 13]. This paper supposes that the objective function to be minimized is $f(\vec{x})$, $\vec{x} = (x_1, \ldots, x_D) \in \Re^D$, and the feasible solution space is $\Psi = \prod_{j=1}^{j=D}[L_j, U_j]$, where L_j and U_j are the lower and upper boundary values of x_j, respectively. The basic DE works through a simple cycle of mutation, crossover, and selection operators after initialization. There are several main variants of DE [1, 2]. This paper uses the DE/rand/1/bin strategy; the strategy is most commonly used in practice. It can be described in detail as follows.

Initialization. Like any other evolutionary algorithms, DE starts with initializing a population of NP D-dimensional vectors representing the potential solutions (individuals) over the optimization search space. We will symbolize each individual by $\vec{x}_i^g = (x_{i,1}^g, x_{i,2}^g, \ldots, x_{i,D}^g)$, for $i = 1, \ldots, NP$, where $g = 0, 1, \ldots, g_{\max}$ is the current generation and $g_{\max}$ is the maximum number of generations. The initial population (at $g = 0$) should be sufficiently scaled to cover the search space as much as possible. Generally, the initialization of population is carried out by generating uniformly randomizing vectors within the search space. We can initialize the jth dimension of the ith individual according to

$$x_{i,j}^0 = L_j + \text{rand}(0,1) \cdot \left(U_j - L_j\right), \tag{1}$$

where rand(0, 1) is a uniformly distributed random number defined in [0, 1] (the same below).

Mutation__DE/rand/1/∗. For each target vector $\vec{x}_i^g$, DE creates a donor vector $\vec{v}_i^g$ by the mutation operator. The mutation operator of DE/rand/1/∗ can be formulated as follows:

$$\vec{v}_i^g = \vec{x}_{r_1}^g + F\left(\vec{x}_{r_2}^g - \vec{x}_{r_3}^g\right), \quad r_1 \neq r_2 \neq r_3 \neq i. \tag{2}$$

Here the indices $r_1, r_2, r_3 \in \{1, 2, \ldots, m\} \setminus \{i\}$ are uniformly random integers mutually different and distinct from the loop index i. And $F \in (0, 1]$ is a real parameter, called mutation or scaling factor.

If the element values of the donor vector $\vec{v}_i$ exceed the prespecified upper bound or lower bound, we can change the element values by the *periodic mode* rule as follows:

$$v_{i,j} = \begin{cases} U_j - \left(L_j - v_{i,j}\right) \% \left|U_j - L_j\right| & \text{if } v_{i,j} < L_j \\ L_j + \left(v_{i,j} - U_j\right) \% \left|U_j - L_j\right| & \text{if } v_{i,j} > U_j. \end{cases} \tag{3}$$

Crossover__DE/∗/bin. This paper uses the binomial crossover operator, which generates the trial vector $\vec{u}_i$ by mixing elements of the donor vector $\vec{v}_i$ with the target vector $\vec{x}_i$ as follows:

$$u_{i,j}^g = \begin{cases} v_{i,j}^g, & \text{if rand}(0,1) \leq \text{CR or } j = j_{\text{rand}} \\ x_{i,j}^g, & \text{otherwise.} \end{cases} \tag{4}$$

Here $\text{CR} \in (0, 1)$ is the probability of crossover and j_{rand} is a random integer in $[1, n]$.

Selection. The selection operator determines which one between the target and the trial vector survives to the next generation. The selection operator for minimization problems can be formulated as

$$\vec{x}_i^{g+1} = \begin{cases} \vec{u}_i^g, & \text{if } f\left(\vec{u}_i^g\right) < f\left(\vec{x}_i^g\right) \\ \vec{x}_i^g, & \text{otherwise.} \end{cases} \tag{5}$$

This means that $\vec{x}_i^{g+1}$ equals the trial vector $\vec{u}_i^g$ if and only if $\vec{u}_i^g$ is a better cost function value than $\vec{x}_i^g$; otherwise, the parent individual $\vec{x}_i^g$ is retained to the next generation.

3. Convergent DE Algorithm with Hidden Adaptation Selection

The proposed SaCDEhaS algorithm is formed by integrating the basic DE with a self-adaptive parameter control strategy and an extra trial vector generation strategy, that is, the uniform mutation operator, and using the haS operator instead of the selection operator of the basic DE.

3.1. Self-Adaptive Parameter Control Strategy. The parameter control strategies of DE have been extensively investigated over the past decade and some developments have been reported. Generally, we distinguish between two forms of setting parameter values: parameter tuning and parameter control. The former means that the user must find suitable values for the parameters by tuning the algorithm and then running the algorithm with those fixed parameters. Reference [1] indicated that a reasonable value for NP could be chosen between $5 \times D$ and $10 \times D$ and the effective range of F is usually between 0.4 and 1. A value of CR = 0.9 has been found to work well across a large range of problems [14].

The latter, that is, parameter control, means that the values of the parameters are changed during the run. References [15, 16] categorised the change into three classes: deterministic parameter control, adaptive parameter control, and self-adaptive parameter control. Reference [17] presented a randomly self-adaptive parameter control strategy; their experimental results showed that the SaDE algorithm with randomly self-adaptive parameter control strategy was better than, or at least comparable to, the basic DE algorithm and several competitive evolutionary algorithms reported in the literature. In particular, the self-adaptive strategy does not increase the time complexity compared to the basic DE algorithm. The parameter control strategy is formulated as follows:

$$F_i^{g+1} = \begin{cases} F_l + \text{rand}\,(0,1) * F_u, & \text{if rand}\,(0,1) < \tau_1 \\ F_i^g, & \text{otherwise,} \end{cases} \tag{6}$$

$$\text{CR}_i^{g+1} = \begin{cases} \text{rand}\,(0,1), & \text{if rand}\,(0,1) < \tau_2 \\ \text{CR}_i^g, & \text{otherwise.} \end{cases} \tag{7}$$

Here F_l and F_u are the lower and upper limits of F and both lie in $[0, 1]$. τ_1 and τ_2 are two new parameters. Reference [17] used $\tau_1 = \tau_2 = 0.1$, $F_l = 0.1$, and $F_u = 0.9$. Then the parameter F belongs in $[0, 0.9]$ while the CR belongs in $[0, 1]$.

In a word, a good parameter setting or a good parameter control strategy can benefit significantly the performance of DE. Combining the merits of the parameter tuning and the parameter control, the proposed SaCDEhaS algorithm initializes $F_i^0 = 0.6$, $\text{CR}_i^0 = 0.9$ and then employs the randomly self-adaptive strategies (6), (7) to change the parameters F and CR, respectively. For SaCDEhaS, a good parameter setting during the initial phase benefits its convergence rate, while the application of the randomly self-adaptive strategies during the middle and the latter phases is helpful to keep its diversity.

3.2. Trial Vector Generation Strategies

3.2.1. Uniform Mutation Operator. After the classical DE crossover operator, SaCDEhaS creates a *variation vector* $\vec{w}_i^g$ corresponding to each trial vector $\vec{u}_i^g$ by the uniform mutation operator. Uniform mutation runs independently on each element of each trial vector. Each element is replaced by a feasible solution randomly generated with an auxiliary convergence probability (P_{ac}), which is a control parameter taking a small value. The description of uniform mutation is as follows:

$$\vec{w}_i^g = \begin{cases} \vec{w}_r & \text{rand}\,(0,1) \le P_{\text{ac}} \\ \vec{u}_i^g & \text{otherwise.} \end{cases} \tag{8}$$

Here $w_{r,j} = L_j + \text{rand}(0,1) \cdot (U_j - L_j)$, L_j and U_j are the lower and upper boundary values of x_j, respectively. From the formula of the uniform mutation, the variation vector $\vec{w}_i^g$ equals the trail vector $\vec{u}_i^g$ with probability $(1 - P_{\text{ac}})$ and equals a uniformly distributed random vector in the feasible region with P_{ac}.

3.2.2. Hidden Adaptation Selection Operator. After uniform Mmutation operator, SaCDEhaS has the following hidden adaptation selection operator (haS) instead of the selection operator of the basic DE. The haS operator determines which one between the target $\vec{x}_i^g$ and the variation vector $\vec{w}_i^g$ survives to the next generation or to break the current loop with the probability P_{ac}. We formulate the haS operator as follows:

$$\vec{x}_i^{g+1} = \begin{cases} \vec{w}_i^g, & \text{if } f\left(\vec{w}_i^g\right) < f\left(\vec{x}_i^g\right) \\ \vec{x}_i^g, & \text{if } f\left(\vec{w}_i^g\right) \ge f\left(\vec{x}_i^g\right), \\ & \text{rand}\,(0,1) > P_{\text{ac}} \\ \text{break}, & \text{otherwise,} \end{cases} \tag{9}$$

$$i = 1, 2, \ldots, NP.$$

Here "break" denotes an execution breaking out of the loop from i to NP.

3.3. Pseudocode and Observation. The pseudocode of SaCDEhaS algorithm (SaCDEhaS/rand/1/bin) is shown in Algorithm 1.

Observations of SaCDEhaS. (i) How to achieve the convergence: the uniform mutation operator makes each generation population ergodic, which assists the algorithm with elitist selection to reach convergence. The theoretical proof of convergence for SaCDEhaS will be shown in the following section.

(ii) How to understand haS operator: the haS operator has the following characteristics.

(1) The "otherwise" in the expression (9) means $f(\vec{w}_i^g) \ge f(\vec{x}_i^g)$ and $\text{rand}(0,1) \le P_{\text{ac}}$. That is to say, for every inferior variation (trial) solution $\vec{w}_i^g$, the haS operator forces SaCDEhaS to break the current loop with a low probability P_{ac}. So the actual probability of breaking the current loop is higher if there are more inferior solutions in the variation population. That is to say, the actual probability is a hidden adaption and proportional to the number of inferior solutions in each target population.

(2) The haS operator remains the current best solution of the target population to the next generation. In fact, the better individuals, function values of which are calculated, are remained greedily to the next generation according to the first two formulas in expression (9). Those current trial individuals, which are abandoned by the "break" strategy in the

initialize parameters NP, F, CR, P_{ac} and population X
while! termination_condition **do**
  for $g = 0$ to g_max
    for $i = 0$ **to** NP
      update F, CR according to formulas (6) and (7)
      $\vec{v}_i^g = mutation\_\text{DE/rand/1}\ (X, F)$
      $\vec{u}_i^g = crossover\_\text{DE/}^*\text{/bin}\ (\vec{x}_i^g, \vec{v}_i^g, CR)$
      $\vec{w}_i^g = uniform\_mutation\ (\vec{u}_i^g, P_{ac})$
      $\vec{x}_i^g = connotatively\_adaptive\_selection\ (\vec{x}_i^g, \vec{w}_i^g, P_{ac})$
      $i = i + 1$
    end for
    $g = g + 1$
  end for
end while

Algorithm 1: Pseudocode of SaCDEhaS (SaCDEhaS/rand/1/bin).

haS operator, do not waste computing overhead to calculate their function values during the program's execution. And the corresponding target individuals are remained to the next generation regardless of their function values. That is to say, the individual, which is confirmed as the current best solution by calculating function values, must be survived to the next generation.

(3) Since the parameters will be regenerated by formulas (6) and (7) after the "break" strategy, one could think of the haS as a triggering strategy of breaking the current loop and regenerating new parameters. Individuals of a target population can be divided into two parts, that is, the previous individuals and the later individuals, by the triggering time. It is obvious that the previous individuals in a population have a greater probability to be updated than the later individuals. In fact, the previous individuals serve two purposes: one is to provide learning information which determines whether or not the current loop is stopped, and the other is to be candidate solutions benefitting the algorithm's search. Unlike a simple regeneration strategy of new parameters, the haS operator does not abandon the previous individuals. This can speed up the convergence without increasing extra computing overhead.

(iii) How to achieve the tradeoff between exploration and exploitation: obviously, population diversity is enhanced by using the uniform mutation operator and the self-adaptive technology for parameters. Meanwhile, however, more inferior solutions may be generated. SaCDEhaS employs the proposed haS operator to minimize the negative influence made by enhancing population diversity. The minimization of the negative influence can promote the balance between the exploration and exploitation ability on some level. In fact, the "break" strategy in haS operator is designed based on the randomly self-adaptive strategy. If the randomly generated parameters are not good and generate many inferior solutions, the loop will have a higher probability to be stopped and new parameters will be randomly generated in the next generation. In the process, the previous individuals' information of a population is used to determine whether the loop continues or breaks.

(iv) How to estimate the computing overhead: comparing with the basic DE, SaCDEhaS has an extra computing overhead to generate random values in three operators, that is, in the self-adaptive parameter control strategy, in the uniform mutation operator, and in the haS operator. However, the computing overhead of generating random values is smaller than an objective function evaluation. So [18] suggests algorithms to estimate their computing overhead by setting function evaluation times (FEs). As shown in Table 2, the convergence speed of SaCDEhaS is quicker than that of SaDE within the same FEs.

4. Proof of Global Convergence for the Proposed Algorithm

Different definitions of the convergence exist for analysing asymptotic convergence of algorithms. The following definition of convergence, that is, convergence in probability, is used in this paper.

Definition 1. Let $\{X(t), t = 0, 1, 2, \ldots\}$ be a population sequence associated with a random algorithm. The algorithm has global convergence in probability for a certain optimization problem, if and only if

$$\lim_{t \to \infty} p\left\{X(t) \cap S_\delta^* \neq \emptyset\right\} = 1, \tag{10}$$

where δ is a small positive real, S_δ^* denotes an expanded optimal solution set, $S_\delta^* = \{\vec{x} \mid |f(\vec{x}) - f(\vec{x}^*)| < \delta\}$, and $\vec{x}^*$ is an optimum of the objective function $f(\vec{x})$.

Several important theorems for the global convergence of evolutionary algorithms (EAs) have been presented. Rudolph [19] generalized convergence conditions for binary and Euclidean search space to a general search space. Under the convergence condition, the EAs with an elitist selection strategy converge to the global optimum. The measure associated with a Markovian kernel function, which needs to be calculated in the convergence condition, seems not to be very convenient. He and Yu [20] introduced several convergence conditions for EAs. The convergence conditions are based on certain probability integral of the offspring entering the

Table 1: Summary of problems with boundary constrains.

Problem	Description	Dimension	Cur.Best
f01	Parameter estimation for frequency-modulated sound waves	6	$0.000000E+00$
f02	Lennard-Jones Potential	30	$-2.842253E+01$
f03	Bifunctional catalyst blend optimal control problem	1	$1.151489E-05$
f04	Optimal control of a nonlinear stirred tank reactor	1	$1.377076E+01$
f05	Tersoff potential function minimization problem, Si(B)	30	$-3.684537E+01$
f06	Tersoff potential function minimization problem, Si(C)	30	$-2.916612E+01$
f07	Spread spectrum radar polyphase code design	20	$5.000000E-01$
f10	Circular antenna array design problem	12	$-2.184253E+01$
f12	Messenger: spacecraft trajectory optimization problem	26	$6.710520E+00^{**}$
f13	HaSsini 2: spacecraft trajectory optimization problem	22	$8.398688E+00$

"Cur.Best" denotes the best function values obtained within 1.5×10^5 FEs on the CEC 2011 competition.
** Marks the fact that the best value is obtained by the proposed algorithm SaCDEhaS.

optimal set. Perhaps, the most convenient theorem for proving the global convergence of DE variants is the one recently presented by Hu et al. [21]. It just needs to check whether or not the probability of the offspring in any subsequence population entering the optimum solution set is big enough. The theorem can be described in detail as follows.

Theorem 2 (see [21]). *Let $\{X(t), t = 0, 1, 2, \ldots\}$ be a population sequence of a DE variant with a greedy selection operator. In the t_k^{th} target population $X(t_k)$, there exists at least one individual $\vec{x}$, which corresponds to the trial individual $\vec{u}$, such that*

$$p\{\vec{u} \in S_\delta^*\} \geq \zeta(t_k) > 0, \tag{11}$$

and the series $\sum_{k=1}^{\infty} \zeta(t_k)$ diverges; then the DE variant holds global convergence.

Where $\{t_k, k = 1, 2, \ldots\}$ denotes any subsequence of natural number set, $p\{\vec{u} \in S_\delta^*\}$ is the probability of $\vec{u}$, locating in the optimal solution set S_δ^*, $\zeta(t_k)$ is a small positive real depending on t_k.

The series $\sum_{k=1}^{\infty} \zeta(t_k)$ diverging means that the probability $p\{\vec{u} \in S_\delta^*\}$ is large enough. That is to say, if the probability of $\vec{u}$ entering into the optimal set, in a certain subsequence population, is large enough, the DE variant with elitist selection holds global convergence.

Conclusion. SaCDEhaS converges to the global optima set of continuous optimization problems, regardless of the initial population distribution.

Proof. From Theorem 2, it is needed to prove that SaCDEhaS satisfies the following two characteristics.

(i) The Selection Operator of SaCDEhaS Is Greedy. SaCDEhaS algorithm uses the haS selection operator. According to the characteristic (2) of the "*How to understand haS operator*" in Section 3, we can know that the haS selection operator can remain greedily the current best solution to the next generation.

(ii) The Probability of Trial Individuals Entering into the Optimal Solution Set is Large Enough. According to formula (8), the probability of uniform mutation operator generating a uniformly distributed random vector equals P_{ac}. So the probability

$$p\{\vec{u} \in S_\delta^*\} \geq 1 - \left(1 - P_{\text{ac}} \cdot \frac{\mu(S_\delta^*)}{\mu(\Psi)}\right)^{NP} > 0, \tag{12}$$

where $\mu(\cdot)$ denotes the measure of a set. Now we set $\zeta(t) \equiv 1-(1-P_{\text{ac}}\cdot(\mu(S_\delta^*)/\mu(\Psi)))^{NP}$; then the series $\sum_{t=1}^{\infty} \zeta(t)$ diverges.

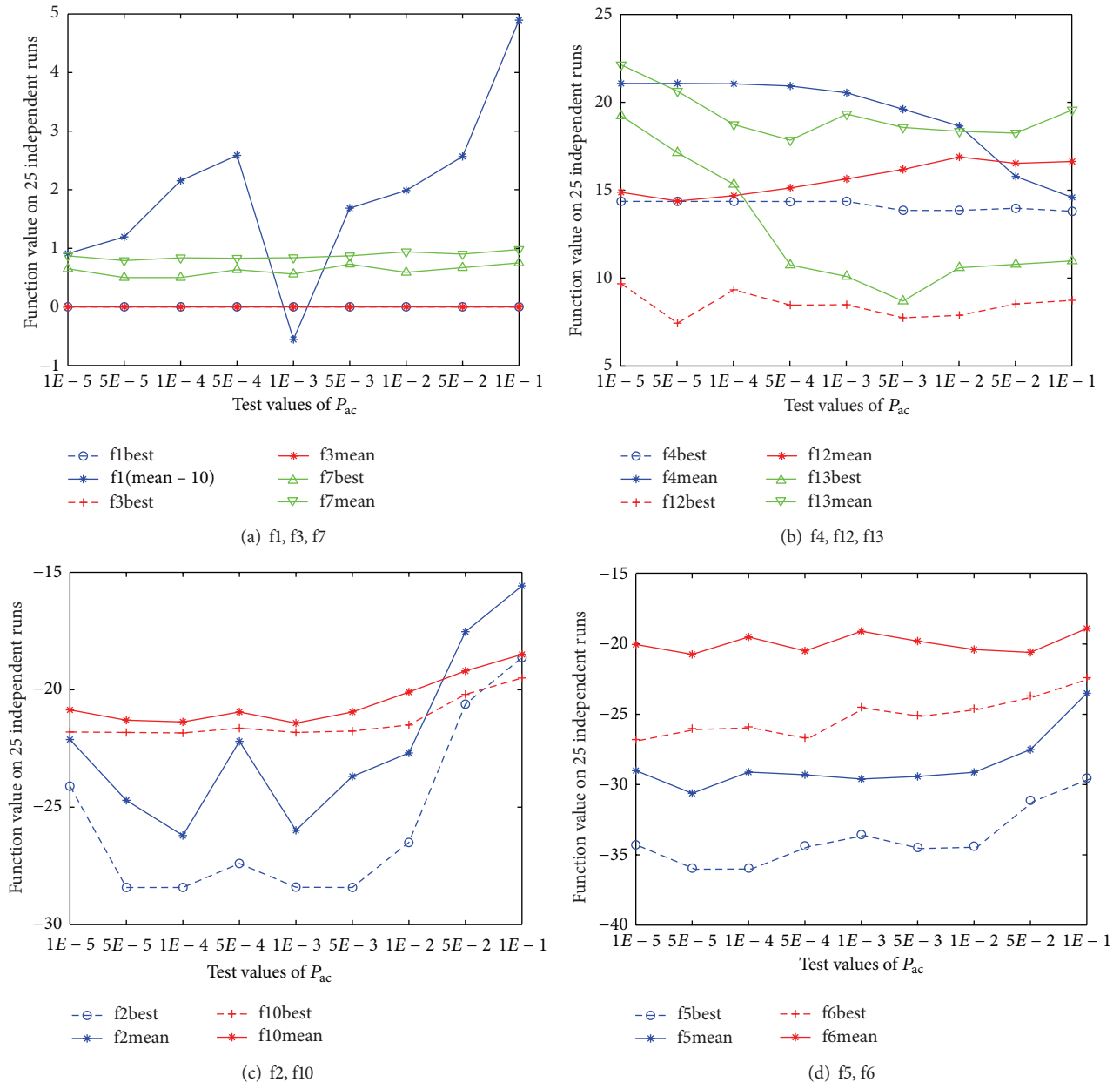
So, according to Theorem 2, we can get that the conclusion holds. □

Observation of SaCDEhaS's Convergence. In fact, like the elite genetic algorithm, SaCDEhaS satisfies the classical convergent model characterized by two aspects: one is the retention of the current best solution, and the other is the ergodicity of the population. Especially, the population ergodicity makes the algorithm have an ability of escaping the local optima. The uniform mutation operator makes SaCDEhaS satisfy the second characteristic, while the haS operator makes it meet the first characteristic.

5. Numerical Experiment

In this section, the performance of SaCDEhaS is tested on the benchmark function set proposed for the *Testing Evolutionary Algorithms on Real-world Numerical Optimization Problems* CEC 2011 Special Session [22]. Experiments are conducted on two comparative studies: (1) between SaCDEhaS and a SaDE algorithm which removes the uniform mutation and haS operators, and (2) between SaCDEhaS and the top four DE variants in the CEC 2011 competition. The first comparison study was implemented to show the effect of the uniform mutation and haS operators. The second comparison was to demonstrate the promising performance of the proposed SaCDEhaS algorithm.

5.1. Problem Definitions and Evaluation Criteria. The benchmark consisted of thirteen engineering optimization problems. Of these, T08, T09, and T11 have equality or inequality



(a) f1, f3, f7

(b) f4, f12, f13

(c) f2, f10

(d) f5, f6

Figure 1: Test pattern of P_{ac} on all 10 problems within 1.5×10^5 Fes.

constraints imposed, and the other 10 problems are related to bound constrained optimization (or unconstrained optimization). These are summarized in Table 1.

According to the evaluation criteria given in the technical report [22], researchers must report the mean, best, and worst objective function values obtained over 25 independent runs after executing their algorithms for 5.0×10^4, 1.0×10^5, and 1.5×10^5 function evaluations (FEs).

5.2. Configuration of Parameters. In our experiments, SaCDEhaS and SaDE used the same population size. Depending on the dimensions of the problems, we set the population sizes for the 10 problems, in order, at 50, 250, 10, 10, 100, 80, 150, 80, 150, and 150. In order to determine the value of auxiliary convergence probability P_{ac}, the experiments tested the performance of SaCDEhaS on all ten optimization problems with different P_{ac} values, that is, $\{1E-5, 1E-5, 5E-5, 1E-4, 5E-4, \ldots, 1E-1\}$. For each parameter P_{ac} value and each problem, SaCDEhaS run 25 times independently within 1.5×10^5 FEs, and the best value and the mean value of the 25 runs were recorded. According to all experimental results, we can get that the optimal P_{ac} were $1E-3$, $1E-4$, $1E-1$, $1E-1$, $5E-5$, $1E-5$, $5E-5$, $1E-4$, $5E-5$, and $5E-3$ in order.

As shown in Figure 1, those lines of f3 and f7 were horizontal, while those lines of the other eight problems were

Table 2: The best and mean values achieved via SaCDEhaS and SaDE algorithms.

Pro.	Alg.	FEs = $5E4$		FEs = $1E5$		FEs = $1.5E5$	
		Best	Mean	Best	Mean	Best	Mean
T01	SaDE	$0.0000E+00$	$1.1995E+01$	$0.0000E+00$	$1.1995E+01$	$0.0000E+00^*$	$1.1995E+01$
	SaCDEhaS	$0.0000E+00$	$9.4453E+00$	$0.0000E+00$	$9.4453E+00$	$0.0000E+00^*$	$9.4453E+00$
T02	SaDE	$-2.5473E+01$	$-1.7971E+01$	$-2.8422E+01$	$-2.4597E+01$	$-2.8423E+01^*$	$-2.5210E+01$
	SaCDEhaS	$-2.6199E+01$	$-1.8424E+01$	$-2.8371E+01$	$-2.4879E+01$	$-2.8423E+01^*$	$-2.6207E+01$
T03	SaDE	$1.1515E-05$	$1.1515E-05$	$1.1515E-05$	$1.1515E-05$	$1.1515E-05^*$	$1.1515E-05$
	SaCDEhaS	$1.1515E-05$	$1.1515E-05$	$1.1515E-05$	$1.1515E-05$	$1.1515E-05^*$	$1.1515E-05$
T04	SaDE	$1.4750E+01$	$2.0686E+01$	$1.4750E+01$	$2.0686E+01$	$1.4750E+01$	$2.0686E+01$
	SaCDEhaS	$1.3815E+01$	$1.6805E+01$	$1.3771E+01$	$1.4971E+01$	$1.3771E+01^*$	$1.4562E+01$
T05	SaDE	$-2.9655E+01$	$-2.4186E+01$	$-3.2728E+01$	$-2.7236E+01$	$-3.3879E+01$	$-2.9448E+01$
	SaCDEhaS	$-3.0337E+01$	$-2.4001E+01$	$-3.5503E+01$	$-2.8854E+01$	$-3.5962E+01$	$-3.0634E+01$
T06	SaDE	$-1.6865E+01$	—	$-2.3470E+01$	—	$-2.4051E+01$	—
	SaCDEhaS	$-1.8954E+01$	$-1.6577E+01$	$-2.3229E+01$	$-1.8566E+01$	$-2.6803E+01$	$-2.0694E+00$
T07	SaDE	$6.8042E-01$	$1.1232E+00$	$5.0000E-01$	$9.5787E-01$	$5.0000E-01^*$	$8.4480E-01$
	SaCDEhaS	$5.6924E-01$	$1.2145E+00$	$5.0000E-01$	$8.7375E-01$	$5.0000E-01^*$	$7.9041E-01$
T10	SaDE	$-2.1774E+01$	$-2.1167E+01$	$-2.1780E+01$	$-2.1250E+01$	$-2.1780E+01$	$-2.1299E+01$
	SaCDEhaS	$-2.1725E+01$	$-2.1122E+01$	$-2.1834E+01$	$-2.1345E+01$	$-2.1835E+01$	$2.6919E-01$
T12	SaDE	$9.4639E+00$	$1.4507E+01$	$8.8193E+00$	$1.3967E+01$	$8.7939E+00$	$1.3823E+01$
	SaCDEhaS	$7.4100E+00$	$1.4350E+01$	$7.0985E+00$	$1.3966E+01$	$6.7105E+00^*$	$2.9544E+00$
T13	SaDE	$1.0625E+01$	$1.8921E+01$	$1.0129E+01$	$1.8378E+01$	$1.0126E+01$	$1.8248E+01$
	SaCDEhaS	$9.3750E+00$	$1.9608E+01$	$8.6945E+00$	$1.8808E+01$	$8.6626E+00$	$4.3153E+00$

"—" denotes that the value is larger than $1E+16$. *Marks that the current best value is obtained.

Table 3: Sign Test of experimental results in Table 2.

	Neg. dif.	Pos. dif.	Tie	Total	P value
On Best Value	3	18	9	30	0.001
On Mean Value	6	20	4	30	0.011

"Neg. dif." and "Pos. dif." denote the number of the negative and positive differences, respectively. "P value" denotes the probability value supporting the null hypothesis.

crooked. This indicated that the parameter P_{ac} is insensitive to the problems f3, f7 and sensitive to the other problems.

5.3. Comparison of SaCDEhaS and SaDE. The results over 25 independent runs for the SaCDEhaS and SaDE algorithms were recorded in the four Tables (Tables 5, 6, 7, and 8) in the appendix. In these tables, the best, median, worst, and mean function values and standard deviations were shown for FEs of 5.0×10^4, 1.0×10^5, and 1.5×10^5, respectively. The comparative aspects extracted in Table 2 included the best and mean values from the tests of SaCDEhaS and SaDE. Sign Test [23], which is a popular statistical method to compare the overall performances of algorithms, was then used to analyze the best values and the mean values in Table 2. As shown in Table 3, the probability value of supporting the null hypothesis of Sign Test on the mean values equaled 0.011, while the probability on the best values was 0.001, which was less than the significance level 0.05. So we can reject the null hypothesis; that is to say, the overall performance of SaCDEhaS is better than SaDE.

In addition, from Table 2, the following are comments on notable aspects of performance of the two algorithms.

(i) In problem T04, SaDE began to stagnate from the first stage when FEs are 5.0×10^4 (the mean and best values were always equal to $2.0686E+01$ and $1.4750E+01$, resp.). However, that did not happen to SaCDEhaS in any of the problems (in the three stages, the mean values were $1.6805E+01$, $1.4971E+01$, and $1.4562E+01$, resp., and at the second stage, SaCDEhaS achieved the current best value $1.3771E+01$).

(ii) SaCDEhaS achieved the current best values on six of the 10 problems (T1, T2, T3, T4, T7, and T12), while SaDE achieved them on only four (T1, T2, T3, and T7). The current best values are marked with star in Table 2. For problem T12 in particular, the minimum value achieved by SaCDEhaS was less than any other reported values in the CEC 2011 competition.

The above analyses indicate that SaCDEhaS outperforms SaDE on the benchmark set. This shows that the employment of the uniform mutation and haS operators benefits the algorithm.

Table 4: The best and mean values achieved via SaCDEhaS and four improvement DE algorithms within 1.5×10^5 FEs.

Test Fun.		SaCDEhaS	SAMODE	DE-RHC	Adap.DE	ED-DE-MA
T01	Best	$0.000000E+00^*$	$0.000000E+00^*$	$5.02E-20^*$	$0.000000E+00^*$	$0.0000E+00^*$
	Mean	$9.445299E+00$	$1.212025E+00$	$8.91E+00$	$3.852582E+00$	$0.0000E+00$
T02	Best	$-2.842253E+01^*$	$-2.842253E+01^*$	$-2.84E+01^*$	$-2.842253E+01^*$	$-2.8423E+01^*$
	Mean	$-2.487860E+01$	$-2.706978E+01$	$-2.64E+01$	$-2.678273E+01$	$-2.6225E+01$
T03	Best	$1.151489E-05^*$	$1.151489E-05^*$	$1.15E-05^*$	$1.151489E-05^*$	$1.1515E-05^*$
	Mean	$1.151489E-05$	$1.151489E-05$	$1.15E-05$	$1.151489E-05$	$1.1515E-05$
T04	Best	$1.377076E+01^*$	$1.377076E+01^*$	$2.00E+01$	$1.432911E+01$	$1.3771E+01^*$
	Mean	$1.497052E+01$	$1.394069E+01$	$2.10E+01$	$1.831217E+01$	$1.3774E+01$
T05	Best	$-3.596196E+01$	$-3.684393E+01$	$-3.69E+01$	$-3.684537E+01^*$	$-3.6895E+01$
	Mean	$-2.885426E+01$	$-3.359474E+01$	$-3.58E+01$	$-3.381816E+01$	$-3.3702E+01$
T06	Best	$-2.680279E+01$	$-2.916612E+01^*$	$-2.92E+01^*$	$-2.916612E+01^*$	$-2.9166E+01^*$
	Mean	$-2.030931E+01$	$-2.763470E+01$	$-2.91E+01$	$-2.583055E+01$	$-2.6809E+01$
T07	Best	$5.000000E-01^*$	$5.000000E-01^*$	$9.51E-01$	$5.000000E-01^*$	$5.1940E-01$
	Mean	$7.904189E-01$	$8.166238E-01$	$1.15E+01$	$5.000000E-01$	$1.1875E+00$
T10	Best	$-2.183457E+01$	$-2.182166E+01$	$-2.05E+01$	$-2.180845E+01$	$-2.1832E+01$
	Mean	$-2.136730E+01$	$-2.165891E+01$	$-1.83E+01$	$-2.095834E+01$	$-2.1421E+01$
T12	Best	$6.710520E+00^*$	$6.943215E+00$	$1.58E+01$	$1.239837E+01$	$9.9697E+00$
	Mean	$1.381452E+01$	$1.106747E+01$	$1.90E+01$	$2.119566E+01$	$1.4436E+01$
T13	Best	$8.662559E+00$	$8.610634E+00$	$1.42E+01$	$8.621241E+00$	$1.4034E+01$
	Mean	$1.854369E+01$	$1.099524E+01$	$2.00E+01$	$1.253713E+00$	$1.6241E+01$

*Marks that the current best value is obtained.

Table 5: The function values achieved via SaCDEhaS for test problems (T01–05).

FEs		T01	T02	T03	T04	T05
5E4	Best	$0.000000E+00$	$-2.619919E+01$	$1.151489E-05$	$1.381528E+01$	$-3.033732E+01$
	Median	$1.094227E+01$	$-1.859087E+01$	$1.151489E-05$	$1.536424E+01$	$-2.338909E+01$
	Worst	$2.016705E+01$	$-8.547038E+00$	$1.151489E-05$	$2.082830E+01$	$-2.139462E+01$
	Mean	$9.445299E+00$	$-1.842434E+01$	$1.151489E-05$	$1.680539E+01$	$-2.400113E+01$
	Std.	$6.526244E+00$	$4.593860E+00$	$0.000000E+00$	$2.554046E+00$	$2.250130E+00$
1E5	Best	$0.000000E+00$	$-2.837096E+01$	$1.151489E-05$	$1.377076E+01$	$-3.550334E+01$
	Median	$1.094227E+01$	$-2.727041E+01$	$1.151489E-05$	$1.434257E+01$	$-2.827621E+01$
	Worst	$2.016705E+01$	$-1.200074E+01$	$1.151489E-05$	$1.911801E+01$	$-2.508282E+01$
	Mean	$9.445299E+00$	$-2.487860E+01$	$1.151489E-05$	$1.497052E+01$	$-2.885426E+01$
	Std.	$6.526244E+00$	$4.034637E+00$	$0.000000E+00$	$1.321138E+00$	$2.685251E+00$
1.5E5	Best	$0.000000E+00$	$-2.842253E+01$	$1.151489E-05$	$1.377076E+01$	$-3.596196E+01$
	Median	$1.094227E+01$	$-2.744682E+01$	$1.151489E-05$	$1.410267E+01$	$-3.042736E+01$
	Worst	$2.016705E+01$	$-1.200229E+01$	$1.151489E-05$	$1.753947E+01$	$-2.528275E+01$
	Mean	$9.445299E+00$	$-2.620730E+01$	$1.151489E-05$	$1.456192E+01$	$-3.063443E+01$
	Std.	$6.526244E+00$	$3.355820E+00$	$0.000000E+00$	$9.726881E-01$	$2.326170E+00$

5.4. Comparison of SaCDEhaS with Other Improved DE Variants. The CEC 2011 competition results are available on the homepage of Suganthan [11]. Four of the top six algorithms belong to the DE family. SAMODE [24] turned out to be the best of these, followed by DE-RHC [25], Adap. DE [26], and ED-DE algorithm [27]. In order to fully evaluate the performance of the proposed SaCDEhaS algorithm, we now compare it with the four top algorithms, for all the box-constrained global optimization problems of the benchmark function set for the CEC 2011 Special Session. Table 4 shows the best and mean values in the test instances for the SaCDEhaS, SAMODE, DE-RHC, Adap. DE, and ED-DE algorithm.

From Table 4, we comment on the performance of these algorithms as follows.

(i) SaCDEhaS obtained the current best values in six of the 10 problems (T1, T2, T3, T4, T7, and T12), while SaCDEhaS, SAMODE, DE-RHC, Adap. DE, and ED-DE obtained the current best values in six (T1, T2, T3, T4, T6, and T7), four (T1, T2, T3, and T6), six (T1, T2, T3, T5, T6, and T7), and five problems (T1, T2, T3, T4,

Table 6: The function values achieved via SaCDEhaS for test problems (T06, 07, 10, 12, 13).

FEs		T06	T07	T10	T12	T13(150, $5E-3$)
$5E4$	Best	$-1.895433E+01$	$5.692406E-01$	$-2.172491E+01$	$7.410013E+00$	$9.374999E+00$
	Median	$-1.659496E+01$	$1.151573E+00$	$-2.134481E+01$	$1.473892E+01$	$2.078702E+01$
	Worst	$-1.320473E+01$	$1.815681E+00$	$-1.921022E+01$	$2.024237E+01$	$2.534618E+01$
	Mean	$-1.657655E+01$	$1.214532E+00$	$-2.112168E+01$	$1.434990E+01$	$1.960803E+01$
	Std.	$1.338433E+00$	$3.291907E+00$	$6.140559E-01$	$2.789373E+00$	$4.562423E+00$
$1E5$	Best	$-2.322918E+01$	$5.000000E-01$	$-2.183449E+01$	$7.098511E+00$	$8.694516E+00$
	Median	$-1.846216E+01$	$8.538244E-01$	$-2.137892E+01$	$1.449314E+01$	$1.922921E+01$
	Worst	$-1.482346E+01$	$1.380470E+00$	$-2.065842E+01$	$2.017225E+01$	$2.515676E+01$
	Mean	$-1.856559E+01$	$8.737516E-01$	$-2.134460E+01$	$1.396631E+01$	$1.880750E+01$
	Std.	$1.957858E+00$	$2.038954E-01$	$2.879415E-01$	$2.876316E+00$	$4.369647E+00$
$1.5E5$	Best	$-2.680279E+01$	$5.000000E-01$	$-2.183457E+01$	$6.710520E+00$	$8.662559E+00$
	Median	$-1.946277E+01$	$8.082074E-01$	$-2.138188E+01$	$1.435571E+01$	$1.901252E+01$
	Worst	$-1.678506E+01$	$1.044074E+00$	$-2.065842E+01$	$2.017119E+01$	$2.515676E+01$
	Mean	$-2.030931E+01$	$7.904189E-01$	$-2.136730E+01$	$1.381452E+01$	$1.854369E+01$
	Std.	$2.069445E+00$	$1.644569E-01$	$2.691909E-01$	$2.954389E+00$	$4.315291E+00$

Table 7: The function values achieved via SaDE for test problems (T01–05).

FEs		T01	T02	T03	T04	T05
$5E4$	Best	$0.000000E+00$	$-2.547346E+01$	$1.151489E-05$	$1.474962E+01$	$-2.965491E+01$
	Median	$1.230606E+01$	$-1.651099E+01$	$1.151489E-05$	$2.107781E+01$	$-2.339049E+01$
	Worst	$2.069471E+01$	$-1.017972E+01$	$1.151489E-05$	$2.410711E+01$	$-2.124083E+01$
	Mean	$1.199518E+01$	$-1.797113E+01$	$1.151489E-05$	$2.068582E+01$	$-2.418576E+01$
	Std.	$6.105013E+00$	$4.391106E+00$	$3.000000E-16$	$2.138599E+00$	$2.181301E+00$
$1E5$	Best	$0.000000E+00$	$-2.842249E+01$	$1.151489E-05$	$1.474962E+01$	$-3.272771E+01$
	Median	$1.230606E+01$	$-2.602234E+01$	$1.151489E-05$	$2.107781E+01$	$-2.584935E+01$
	Worst	$2.069471E+01$	$-1.819737E+01$	$1.151489E-05$	$2.410711E+01$	$-2.371424E+01$
	Mean	$1.199518E+01$	$-2.459705E+01$	$1.151489E-05$	$2.068582E+01$	$-2.723624E+01$
	Std.	$6.105013E+00$	$3.451965E+00$	$3.000000E-16$	$2.138599E+00$	$2.564945E+00$
$1.5E5$	Best	$0.000000E+00$	$-2.842253E+01$	$1.151489E-05$	$1.474962E+01$	$-3.387944E+01$
	Median	$1.230606E+01$	$-2.648390E+01$	$1.151489E-05$	$2.107781E+01$	$-2.856424E+01$
	Worst	$2.069471E+01$	$-1.820793E+01$	$1.151489E-05$	$2.410711E+01$	$-2.575636E+01$
	Mean	$1.199518E+01$	$-2.529978E+01$	$1.151489E-05$	$2.068582E+01$	$-2.944785E+01$
	Std.	$6.105013E+00$	$3.419653E+00$	$3.000000E-16$	$2.138599E+00$	$2.562794E+00$

and T6), respectively. The success rate of SaCDEhaS achieving the current best values was not less than that of the other four algorithms.

(ii) SaCDEhaS found the current best value of $6.710520E+00$ in problem T12, which was lower than the value ($6.943215E+00$) obtained on the CEC 2011 competition.

(iii) However, considering only the mean values, SaCDEhaS did not dominate. The best-performing algorithm was SAMODE.

We can state from the above that the proposed SaCDEhaS algorithm is promising when compared to similar algorithms.

6. Conclusion

This paper presents a self-adaptive convergent DE algorithm, SaCDEhaS, for continuous engineering optimization problems. SaCDEhaS is formed by integrating the basic DE with a self-adaptive parameter control strategy and two extra operators, that is, uniform mutation and hidden adaptation selection (haS) operators. The haS operator automatically controls the algorithm to break the current loop at a low probability. The breaking probability is proportional to the number of inferior individuals of every trial population. To some degree, this alleviates the negative effects associated with enhancement of the diversity. The theoretical proof of this paper demonstrated that these improvements enhance the diversity of populations and guarantee the algorithm's global convergence in probability.

Table 8: The function values achieved via SaDE for test problems (T06, 07, 10, 12, 13).

FEs		T06	T07	T10	T12	T13
5*E*4	Best	−1.686549*E* + 01	6.804249*E* − 01	−2.177440*E* + 01	9.463884*E* + 00	1.062508*E* + 01
	Median	−1.173179*E* + 01	1.042188*E* + 00	−2.136835*E* + 01	1.545941*E* + 01	2.000271*E* + 01
	Worst	—	1.738322*E* + 00	−1.798882*E* + 01	1.901426*E* + 01	2.434862*E* + 01
	Mean	—	1.123239*E* + 00	−2.116746*E* + 01	1.450659*E* + 01	1.892074*E* + 01
	Std.	—	3.236158*E* − 01	8.397006*E* − 01	2.463023*E* + 00	3.421109*E* + 00
1*E*5	Best	−2.346968*E* + 01	5.000000*E* − 01	−2.177983*E* + 01	8.819336*E* + 00	1.012878*E* + 01
	Median	−1.456034*E* + 01	9.747225*E* − 01	−2.141695*E* + 01	1.452319*E* + 01	1.934751*E* + 01
	Worst	—	1.444171*E* + 00	−1.799526*E* + 01	1.865872*E* + 01	2.343274*E* + 01
	Mean	—	9.578726*E* − 01	−2.125042*E* + 01	1.396713*E* + 01	1.837846*E* + 01
	Std.	—	2.224497*E* − 01	8.395020*E* − 01	2.467759*E* + 00	3.401275*E* + 00
1.5*E*5	Best	−2.405059*E* + 01	5.000000*E* − 01	−2.177985*E* + 01	8.793885*E* + 00	1.012562*E* + 01
	Median	−1.495897*E* + 01	8.691929*E* − 01	−2.146233*E* + 01	1.437097*E* + 01	1.905806*E* + 01
	Worst	—	1.274259*E* + 00	−1.825312*E* + 01	1.861175*E* + 01	2.328276*E* + 01
	Mean	—	8.448028*E* − 01	−2.129898*E* + 01	1.382334*E* + 01	1.824810*E* + 01
	Std.	—	1.715784*E* − 01	8.041711*E* − 01	2.485077*E* + 00	3.370184*E* + 00

The experimental studies were carried out on the all bound-constrained and unconstrained optimization problems in the CEC 2011 competition, which included 10 global engineering optimization problems. Comparison between SaCDEhaS and SaDE (which did not use the uniform mutation and haS operators) indicated the advantages of simultaneously using these two operators in SaCDEhaS. SaCDEhaS was also compared with the top four DE variants in the CEC 2011 competition. The results indicate the competitiveness of the proposed SaCDEhaS.

In the proposed algorithm, SaCDEhaS, the additional auxiliary convergence probability P_{ac} was used. Tests of the effects of different sets of P_{ac} on the benchmarks indicated that the performance of SeCDEhaS is sensitive to P_{ac} when finding the best set of parameters. We will investigate the adaptive or self-adaptive strategy of the parameter P_{ac} in future work.

Conflict of Interests

The authors declare that they have no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant nos. 61170202 and 11301163, the Fundamental Research Funds for the Central Universities under Grant no. 2012-YB-19, and the Key Project of Chinese Ministry of Education under Grant no. 212109.

References

[1] R. Storn and K. Price, "A simple and efficient adaptive scheme for global optimization over continuous spaces," Tech. Rep. TR-95-012, ICSI, Berkeley, Calif, USA, 1995.

[2] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[3] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.

[4] F. Neri and V. Tirronen, "Recent advances in differential evolution: a survey and experimental analysis," *Artificial Intelligence Review*, vol. 33, no. 1-2, pp. 61–106, 2010.

[5] F. Xue, A. C. Sanderson, and R. J. Graves, "Modeling and convergence analysis of a continuous multi-objective differential evolution algorithm," in *Proceedings of the IEEE Congress on Evolutionary Computation (IEEE CEC '05)*, vol. 1, pp. 228–235, IEEE Press, Edinburgh, UK, September 2005.

[6] F. Xue, A. C. Sanderson, and R. J. Graves, "Multi-objective differential evolution: algorithm, convergence analysis, and applications," in *Proceedings of the IEEE Congress on Evolutionary Computation (IEEE CEC '05)*, vol. 1, pp. 743–750, Edinburgh, UK, September 2005.

[7] C. J. F. Ter Braak, "A Markov Chain Monte Carlo version of the genetic algorithm Differential Evolution: easy Bayesian computing for real parameter spaces," *Statistics and Computing*, vol. 16, no. 3, pp. 239–249, 2006.

[8] Y. T. Zhao, J. Wang, and Y. Song, "An improved differential evolution to continuous domains and its convergence," in *Proceedings of the 1st ACM/SIGEVO Summit on Genetic and Evolutionary Computation (GEC '09)*, pp. 1061–1064, Shanghai, China, June 2009.

[9] Z. Zhan and J. Zhang, "Enhance differential evolution with random walk," in *Proceedings of the 14th International Conference on Genetic and Evolutionary Copputation Conference Companion (GECCO '12)*, pp. 1513–1514, Philadelphia, Pa, USA, 2012.

[10] S. Ghosh, S. Das, A. V. Vasilakos, and K. Suresh, "On convergence of differential evolution over a class of continuous functions with unique global optimum," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 42, no. 1, pp. 107–124, 2012.

[11] P. N. Suganthan, http://www.ntu.edu.sg/home/epnsugan/.

[12] Z. Hu, Q. Su, S. Xiong, and F. Hu, "Self-adaptive hybrid differential evolution with simulated annealing algorithm for numerical optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '08)*, pp. 1189–1194, Hong Kong, China, June 2008.

[13] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 55–66, 2011.

[14] J. Montgomery and S. Chen, "An analysis of the operation of differential evolution at high and low crossover rates," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '10)*, pp. 881–888, Barcelona, Spain, July 2010.

[15] A. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 124–141, 1999.

[16] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*, Natural Computing Series, Springer, Berlin, Germany, 2003.

[17] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[18] P. N. Suganthan, N. Hansen, J. J. Liang et al., "Problem definition and evalution criteria for the CEC 2005 special session on real-parameter optimization," Tech. Rep., Nanyang Technological University, Singapore, 2005.

[19] G. Rudolph, "Convergence of evolutionary algorithms in general search spaces," in *Proceedings of the IEEE International Conference on Evolutionary Computation (CEC '96)*, pp. 50–54, Nagoya, Japan, May 1996.

[20] J. He and X. Yu, "Conditions for the convergence of evolutionary algorithms," *Journal of Systems Architecture*, vol. 47, no. 6, pp. 601–612, 2001.

[21] Z. Hu, S. Xiong, Q. Su, and X. Zhang, "Sufficient conditions for global convergence of differential evolution algorithm," *Journal of Applied Mathematics*, vol. 2013, Article ID 193196, 14 pages, 2013.

[22] S. Das and P. N. Suganthan, "Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems," Tech. Rep., Competition on Testing Evolutionary Algorithms on Real-World Numerical Optimization Problems, New Orleans, La, USA, 2010.

[23] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.

[24] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Differential evolution with multiple strategies for solving CEC2011 real-world numerical optimization problems," in *Proceedings of the IEEE Congress of Evolutionary Computation (CEC '11)*, pp. 1041–1048, New Orleans, La, USA, June 2011.

[25] A. LaTorre, S. Muelas, and J.-M. Peña, "Benchmarking a hybrid DE-RHC algorithm on real world problems," in *Proceedings of the IEEE Congress of Evolutionary Computation (CEC '11)*, pp. 1027–1033, New Orleans, La, USA, June 2011.

[26] M. Asafuddoula, T. Ray, and R. Sarker, "An adaptive differential evolution algorithm and its performance on real world optimization problems," in *Proceedings of the IEEE Congress of Evolutionary Computation (CEC '11)*, pp. 1057–1062, New Orleans, La, USA, June 2011.

[27] Y. Wang, B. Li, and K. Zhang, "Estimation of distribution and differential evolution cooperation for real-world numerical optimization problems," in *2011 IEEE Congress of Evolutionary Computation (CEC '11)*, pp. 1315–1321, New Orleans, La, USA, June 2011.