Exploratory Power of the Harmony Search Algorithm: Analysis and Improvements for Global Numerical Optimization

Swagatam Das, Arpan Mukhopadhyay, Anwit Roy, Ajith Abraham, *Senior Member, IEEE*, and Bijaya K. Panigrahi, *Senior Member, IEEE*

***Abstract*—The theoretical analysis of evolutionary algorithms is believed to be very important for understanding their internal search mechanism and thus to develop more efficient algorithms. This paper presents a simple mathematical analysis of the explorative search behavior of a recently developed metaheuristic algorithm called harmony search (HS). HS is a derivative-free real parameter optimization algorithm, and it draws inspiration from the musical improvisation process of searching for a perfect state of harmony. This paper analyzes the evolution of the population–variance over successive generations in HS and thereby draws some important conclusions regarding the explorative power of HS. A simple but very useful modification to the classical HS has been proposed in light of the mathematical analysis undertaken here. A comparison with the most recently published variants of HS and four other state-of-the-art optimization algorithms over 15 unconstrained and five constrained benchmark functions reflects the efficiency of the modified HS in terms of final accuracy, convergence speed, and robustness.**

***Index Terms*—Explorative power, global optimization, harmony search (HS), particle swarm optimization (PSO), population variance.**

I. Introduction

In THE recent past, with the computational cost having been almost dramatically reduced, researchers all over the world are attracted toward the nature-inspired metaheuristics [1]–[4] on a regular basis to meet the demands of the complex, real-world optimization problems. Following this tradition, in 2001, Geem *et al.* proposed harmony search (HS) [5]–[7], a derivative-free metaheuristic algorithm, mimicking the improvisation process of music players. Since its inception, HS has successfully been applied to a wide variety of practical optimization problems like pipe-network design [8], structural optimization [9], the vehicle routing problem [10], the combined heat and power economic dispatch problem [11], the scheduling of a multiple dam system [12], and so on. A significant amount of research has already been undertaken to investigate the application of HS in solving difficult engineering optimization problems, as well as to improve the performance of HS by tuning its parameters and/or blending it with other powerful optimization techniques like particle swarm optimization (PSO) [4]. However, to the best of our knowledge, no significant research work has so far been reported in the context of the mathematical analysis of the underlying search mechanisms of HS. We believe that such an analysis may provide important guidelines to the researchers regarding the selection of control parameters for HS. It can also help us to understand the strong and weak points of this new algorithm that, within a short span of time, gained wide popularity among the researchers from diverse domains of science and engineering.

The efficiency of most evolutionary algorithms (EAs) depends on their extent of explorative and exploitative tendencies during the course of search. Exploitation means the ability of a search algorithm to use the information already collected and thus to orient the search more toward the goal, while exploration is the process that allows introduction of new information into the population. Exploration helps the algorithm to quickly search the new regions of a large search volume. Proper balance between these two characteristics results into enhanced performance [13]. Generally, EAs explore the search space by the (genetic) search operators, while exploitation is done via *selection* that promotes better individuals to the next generation.

In this paper, we focus on the evolution of the population variance of HS and its influence on the explorative power of the algorithm. We first find an analytical expression for the expected population variance of HS, taking inspiration from the works of Beyer [14], [15], who did a conceptually similar analysis for evolution strategies (ESs)/evolutionary programming (EP) [1]. We then draw a few important conclusions regarding the explorative power of HS by observing the change in expected population variance over generations with and without selection. Based on the analysis presented here, we propose a simple modification of the classical HS. In the modified HS, a control parameter known as the *distance bandwidth* (bw) has been made proportional to the standard deviation of the current population. This way, the proportionality constant provides us an extra control over the population variance, as well as the explorative power of HS over generations. Experimental results on a testbed of 20 well-known numerical benchmarks and one real-world optimization problem show that the modified HS can

Manuscript received May 19, 2009; revised February 5, 2010. This paper was recommended by Associate Editor J. Wang.

S. Das, A. Mukhopadhyay, and A. Roy are with the Department of Electronics and Telecommunication Engineering, Jadavpur University, Kolkata 700032, India (e-mail: swagatamdas19@yahoo.co.in; arpanmu@gmail.com; anwitju@gmail.com).

A. Abraham is with the Machine Intelligence Research Labs (MIR Labs), Scientific Network for Innovation and Research Excellence, Auburn, WA 98071-2259 USA (e-mail: ajith.abraham@ieee.org).

B. K. Panigrahi is with the Department of Electrical Engineering, Indian Institute of Technology, Delhi 110016, India (e-mail: bkpanigrahi@ee.iitd.ac.in).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2010.2046035

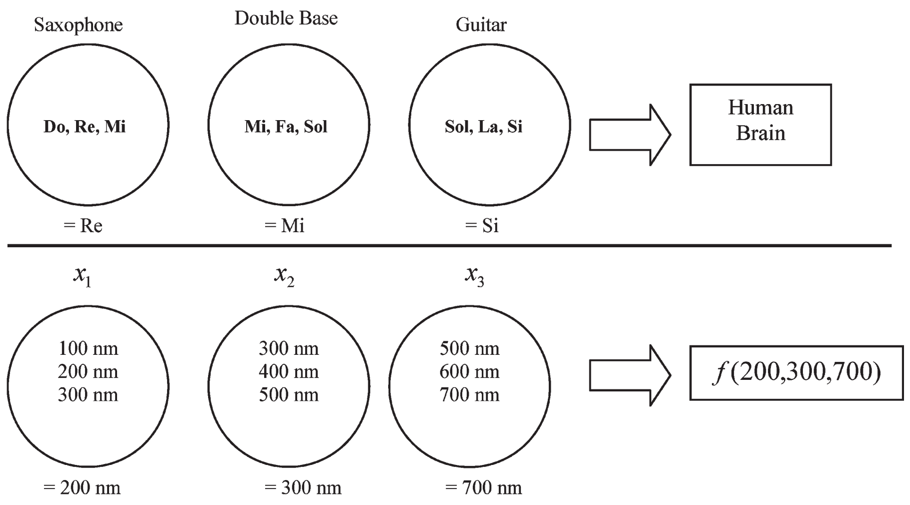


Fig. 1. Analogy between music improvisation and engineering optimization (figure adopted from [6]).

outperform the three most recently published variants of HS and can provide better or at least comparable results with respect to four powerful optimization algorithms of current interest, in a statistically meaningful way. At this point, we would like to mention that a preliminary version of Theorem 1 (to be presented in Section IV) with very limited simulation results appeared as a conference article in [16]. However, the present version has considerably been enhanced and differs in many respects from [16].

The rest of this paper is organized in the following way. Section II outlines the classical HS in a comprehensive style. Section III provides a brief survey on the present state-of-the-art research on HS and its applications. Section IV presents the main theoretical results and the corresponding improvement scheme for HS. Experimental settings and simulation strategies for comparing the modified HS with other population-based optimization algorithms are explained in Section V. Section VI presents and discusses the results of the comparative study. Finally, Section VII concludes the paper and unfolds a few important future research issues.

II. HS Metaheuristic Algorithm—An Overview

HS was devised as a new metaheuristic algorithm, taking inspiration from the music improvisation process, where musicians improvise their instruments' pitches searching for a perfect state of harmony. Although the estimation of a harmony is aesthetic and subjective, on the other hand, there are several theorists who have provided the standard of harmony estimation: Greek philosopher and mathematician Pythagoras (582–497BC) worked out the frequency ratios (or string length ratios with equal tension) and found that they had a particular mathematical relationship, after researching what notes sounded pleasant together. French composer Jean-Philippe Rameau (1683–1764) established the classical harmony theories in the book "*Treatise on Harmony*," which still form the basis of the modern study of tonal harmony [17].

The analogy between music improvisation and engineering optimization is illustrated in Fig. 1. Each music player (saxophonist, double bassist, and guitarist) can correspond to each decision variable (x_1, x_2, x_3) and the range of each music instrument (saxophone $=$ {Do, Re, Mi}; double bass $=$ {Mi, Fa, Sol}; and guitar $=$ {Sol, La, Si}) corresponds to the range of each variable value ($x_1 = \{100, 200, 300\}$; $x_2 = \{300, 400, 500\}$; and $x_3 = \{500, 600, 700\}$). If the saxophonist toots the note Re, the double bassist plucks Mi, and the guitarist plucks Si, their notes together make a new harmony (Re, Mi, Si). If the new harmony is better than the existing harmony, it is kept. Likewise, the new solution vector (200, 300, and 700 mm) is kept if it is better than the existing harmony in terms of the objective function value. The harmony quality is improved by practice after practice.

Similarly, in engineering optimization, each decision variable initially chooses any value within the possible range, together making one solution vector. If all the values of decision variables make a good solution, that experience is stored in each variable's memory, and the possibility of making a good solution is also increased next time. When a musician improvises one pitch, he (or she) has to follow any one of three rules: 1) playing any one pitch from his (or her) memory; 2) playing an adjacent pitch of one pitch from his (or her) memory; and 3) playing a totally random pitch from the possible range of pitches. Similarly, when each decision variable chooses one value in the HS algorithm, it follows any one of three rules: 1) choosing any one value from the harmony memory (HM), which is defined as memory considerations; 2) choosing an adjacent value of one value from the HM, which is defined as pitch adjustments; and 3) choosing a totally random value from the possible range of values, which is defined as randomization. According to the above algorithm concept, the HS metaheuristic algorithm consists of the following five steps [5], [6]:

Step 1) **Initialization of the optimization problem and algorithm parameters:** In the first step, the optimization problem is specified as follows:

$$\text{Minimize (or Maximize) } f(\vec{x})$$
$$\text{subjected to } x_i \in X_i, \quad i = 1, 2, \ldots, N. \qquad (1)$$

where $f(.)$ is a scalar objective function to be optimized; $\vec{x}$ is a solution vector composed of decision variables x_i; X_i is the set of possible range of values for each decision variable x_i (continuous decision variable), that is, ${}_Lx_i \leq X_i \leq {}_Ux_i$, where ${}_Lx_i$ and

${}_Ux_i$ are the lower and upper bounds for each decision variable, respectively, and N is the number of decision variables. In addition, the control parameters of HS are also specified in this step. These parameters are the HM size (HMS) i.e., the number of solution vectors (population members) in the HM (in each generation); the HM considering rate (HMCR); the pitch-adjusting rate (PAR); and the number of improvisations (NI) or stopping criterion.

Step 2) **HM initialization:** In this step, each component of each vector in the parental population (HM), which is of size HMS, is initialized with a uniformly distributed random number between the upper and lower bounds $[{}_Lx_i, {}_Ux_i]$, where $1 \le i \le N$. This is done for the ith component of the jth solution vector using the following equation:

$$x_i^j = {}_Lx_i + rand(0,1) \cdot ({}_Ux_i - {}_Lx_i) \tag{2}$$

where $j = 1, 2, 3 \ldots, \text{HMS}$, and $rand(0,1)$ is a uniformly distributed random number between 0 and 1, and it is instantiated anew for each component of each vector.

Step 3) **New harmony improvisation:** In this step, a new harmony vector $\vec{x}' = (x_1', x_2', x_3', x_4', \ldots, x_N')$ is generated based on three rules: 1) memory consideration; 2) pitch adjustment; and 3) random selection. Generating a new harmony is called 'improvisation.' In the memory consideration, the value of the first decision variable x_1' for the new vector is chosen from any of the values already existing in the current HM, i.e., from the set $\{x_1^1, \ldots, x_1^{\text{HMS}}\}$, with a probability HMCR. The values of the other decision variables $x_2', x_3', x_4', \ldots, x_N'$ are also chosen in the same manner. The HMCR, which varies between 0 and 1, is the rate of choosing one value from the previous values stored in the HM, while $(1 - \text{HMCR})$ is the rate of randomly selecting a fresh value from the possible range of values, i.e.,

$$x_i' \leftarrow \begin{cases} x_i \in \{x_i^1, x_i^2, x_i^3, \ldots, x_i^{\text{HMS}}\} \\ \quad \text{with probability HMCR} \\ x_i \in X_i \\ \quad \text{with probability } (1 - \text{HMCR}). \end{cases} \tag{3}$$

For example, an $\text{HMCR} = 0.80$ indicates that the HS algorithm will choose the decision variable value from historically stored values in the HM with an 80% probability or from the entire possible range with a 20% probability. Every component obtained by the memory consideration is further examined to determine whether it should be pitch adjusted. This operation uses the parameter PAR (which is the rate of pitch adjustment) as follows:

Pitch-Adjusting Decision for

$$x_i' = \begin{cases} x_i' \pm rand(0,1) \cdot bw \text{ with probability PAR} \\ x_i' \text{ with probability } (1 - \text{PAR}) \end{cases} \tag{4}$$

where bw is an arbitrary distance bandwidth (a scalar number), and $rand()$ is a uniformly distributed random number between 0 and 1. Evidently, step 3) is responsible for generating new potential variation in the algorithm and is comparable to mutation in standard EAs.

Step 4) **HM update:** If the new harmony vector $\vec{x}' = (x_1', x_2', x_3', x_4', \ldots, x_N')$ is better than the worst harmony in the HM, judged in terms of the objective function value, the new harmony is included in the HM, and the existing worst harmony is excluded from the HM. This is actually the selection step of the algorithm where the objective function value is evaluated to determine if the new variation should be included in the population (HM).

Step 5) **Check stopping criterion:** If the stopping criterion (maximum NI) is satisfied, the computation is terminated. Otherwise, steps 3) and 4) are repeated.

III. Related Works

HS was first proposed in 2001 by Geem *et al.* [5] (primarily for discrete search variables), and the authors demonstrated the superior performance of the algorithm by comparing it with a standard genetic algorithm (GA), an EP, and the generalized reduced gradient algorithm on the traveling salesperson problem, a 2-D constrained benchmark function, and a least-cost pipe network design problem. HS was subsequently extended by Lee and Geem to tackle continuous engineering optimization problems [6], [18]. Lee and Geem [6] reported superior performance of the algorithm in comparison to the state-of-the-art GA-based approaches proposed by Deb [19], Fogel [20], and Coello Coello *et al.* [21] on several constrained engineering optimization problems in two dimensions.

Geem added a new operation called *ensemble consideration* to the original HS algorithm structure in [22]. The new operation considers the relationship among decision variables, and the value of each decision variable can be determined from the strong relationship with other variables. In [23], Geem proposed a new stochastic derivative for discrete variables based on the HS algorithm.

Mahdavi *et al.* proposed an improved HS algorithm (IHS) [24] that employs a novel method generating new solution vectors with enhanced accuracy and convergence speed. The original HS keeps two of its control parameters PAR and *bw* fixed over generations. In IHS, both the parameters are dynamically changed with generations; more specifically, PAR is linearly increased, while bw is exponentially decreased between the predefined minimum and maximum values. IHS was found to outperform the classical HS over several benchmark problems. Omran and Mahdavi tried to improve the performance of HS by incorporating some ideas borrowed from the *g_best* PSO [4] into the algorithm. The new approach, called Global-best HS (GHS) [25], modifies the pitch-adjustment step of the HS such that the new harmony can mimic the best harmony in the HM, thus replacing the bw parameter altogether and adding a social dimension to HS. More recently, based on concepts that the better harmony vector should enjoy higher selection probability and that several new harmonies are generated in every iteration, Cheng *et al.* [26] developed another improved HS algorithm, called modified HS (MHS), which was found to be more efficient than the basic HS algorithm for slope stability analysis. Some of the very interesting applications of HS can be found in [27]–[36].

IV. Analytical Treatment

A. Computation of the Expected Population Variance

The explorative power of an EA expresses its capability to explore the search space. The evolution of the expected population variance over generations provides a measure of the explorative power of the algorithm. In the original HS algorithm, we do not have to deal with any population of vectors in step 3) (new harmony improvisation). Instead, a single new harmony vector is created in this step. However, for the sake of analysis, here, we assume that a population of vectors is created in its variation step (step 3), i.e., the step that is responsible for exploring new variations in the search space. After the selection step (step 4) of HS, the population variance may increase or decrease. To avoid any premature convergence or stagnation in the successive generations and to ensure that most of the regions in the search space have been explored, the variation operators must adjust the population variance such that it has a reasonable value from one generation to another. Thus, if the selection step decreases the population variance, the variation operators must necessarily increase it so as to achieve a proper balance between exploration and exploitation. For the sake of the analysis of the explorative power of an EA, we keep aside the selection steps involved in the algorithm and only consider the variation steps. Now, if we can show that the population variance over generations is increasing by applying only the variation operators, it can be inferred that the algorithm has good explorative power.

Thus, in step 3), i.e., the new harmony improvisation process, we consider a population of new harmonies instead of a single harmony. This is done for the sake of the analysis of the evolution of population variance. This new population is referred to as $\mathbf{Y} = [\vec{Y}^1, \vec{Y}^2, \ldots, \vec{Y}^{\text{HMS}}]$, where each vector $\vec{Y}^i$ is generated following the rules described in step 3).

Since, in HS, the perturbations are made independently for each decision variable, we can say that it will not be a loss of generality if we conduct our analysis for single-dimensional vectors, i.e., scalars. To do this, we will consider an initial population of scalar variables $x = \{x_1, x_2, x_3, \ldots, x_m\}$ with elements $(x_1 \in \Re)$, where we have taken HMS $= m$. The variance of the population x is given by

$$Var(x) = \frac{1}{m}\sum_{l=1}^{m}(x_l - \overline{x})^2 = \overline{x^2} - \overline{x}^2$$

where $\overline{x}$ = population mean and $\overline{x^2}$ = quadratic population mean. If the elements of the population are perturbed with some random numbers, $Var(x)$ will be a random variable, and $E(Var(x))$ will be a measure of the explorative power. The main analytical result in this context is expressed in the form of the following theorem.

Theorem 1: Let $x = \{x_1, x_2, x_3, \ldots, x_m\}$ be the current population of HS and $Y = \{Y_1, Y_2, \ldots, Y_m\}$ be the intermediate population obtained after the new harmony improvisation step. If HMCR is the HM consideration probability, PAR is the pitch-adjustment probability, bw is the arbitrary distance bandwidth, and we consider the allowable range for the decision variables (x_i) to be $\{x_{\min}, x_{\max}\}$, where $x_{\max} = a$ and $x_{\min} = -a$ with $a \in \Re$, then

$$E\left(Var(Y)\right) = \frac{(m-1)}{m} \cdot \left[\text{HMCR} \cdot Var(x) + \text{HMCR} \cdot (1-\text{HMCR}) \cdot \overline{x}^2 + \frac{1}{3} \cdot \text{HMCR} \cdot \text{PAR} \cdot bw^2 + \frac{a^2}{3} \cdot (1-\text{HMCR})\right]. \quad (5)$$

Proof: Here, $x = \{x_1, x_2, x_3, \ldots, x_m\}$ is the current scalar population. Therefore, the population mean is $\overline{x} = (1/m)\sum_{l=1}^{m} x_l$, and the quadratic population mean is $\overline{x^2} = (1/m)\sum_{l=1}^{m} x_l^2$. $Y = \{Y_1, Y_2, \ldots, Y_m\}$ is the intermediate population of scalars obtained after the new harmony improvisation step. Each element Y_l of the population Y is obtained as

$$Y_l \leftarrow \begin{cases} x_r, & \text{with probability HMCR} \cdot (1-\text{PAR}) \\ x_r + bw \cdot rand, & \text{with probability } 0.5 \cdot \text{HMCR} \cdot \text{PAR} \\ x_r - bw \cdot rand, & \text{with probability } 0.5 \cdot \text{HMCR} \cdot \text{PAR} \\ x_{\text{new}}, & \text{with probability } (1-\text{HMCR}) \end{cases}$$

where r is a uniformly chosen random number from the set $\{1, 2, \ldots, m\}$, x_{new} is a new random value in the allowable range $\{x_{\min}, x_{\max}\}$ or $\{-a, a\}$, and $rand$ is a uniformly chosen random number between 0 and 1. Since the index r is a uniformly distributed random variable with values in $\{1, 2 \ldots, m\}$, the probability $p_k = P(r = k) = (1/m)$, where k is a number within the set. Thus, x_r is a random variable, and

$$E(x_r) = \sum_{k=1}^{m} p_k \cdot x_k = \sum_{k=1}^{m} P(r=k) \cdot x_k = \frac{1}{m} \cdot \sum_{k=1}^{m} x_k = \overline{x} \quad (6)$$

with

$$E\left(x_r^2\right) = \sum_{k=1}^{m} p_k \cdot x_k^2 = \sum_{k=1}^{m} P(r=k) \cdot x_k^2 = \frac{1}{m} \cdot \sum_{k=1}^{m} x_k^2 = \overline{x^2}. \quad (7)$$

We now compute $E(Y_l)$ and $E(Y_l^2)$ by using the following lemma, the proof of which can be found in [37]. ■

Lemma 1 [37]: Let $Z_1, Z_2, \ldots, Z_k$ be the bounded random variables that are independent with respect to any discrete random variable V having the distribution $P(V = v_j) = p_j$, $j = 1, 2, \ldots, k,\ p_j \in [0, 1],\ \sum_{j=1}^{k} p_j = 1$. Then, the random variable

$$Z = \begin{cases} Z_1, & \text{with probability } p_1 \\ Z_2, & \text{with probability } p_2 \\ \vdots & \\ Z_k, & \text{with probability } p_k \end{cases}$$

has the mean

$$M(Z) = \sum_{p=1}^{k} p_j . M(Z_j). \quad (8)$$

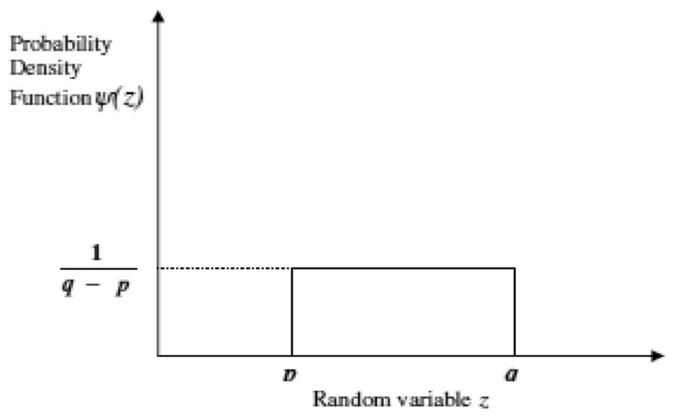


Fig. 2. Continuous uniform probability distribution.

Using the above lemma, we get the following expressions for $E(Y_l)$ and $E(Y_l^2)$:

$$\begin{aligned} E(Y_l) = {} & \text{HMCR} \cdot (1-\text{PAR}) \cdot E(x_r) + 0.5 \cdot \text{HMCR} \\ & \cdot \text{PAR} \cdot E(x_r + bw \cdot rand) + 0.5 \cdot \text{HMCR} \cdot \text{PAR} \\ & \cdot E(x_r - bw \cdot rand) + (1-\text{HMCR}) \cdot E(x_{\text{new}}) \quad (9) \\ E\left(Y_l^2\right) = {} & \text{HMCR} \cdot (1-\text{PAR}) \cdot E(x_r^2) + 0.5 \cdot \text{HMCR} \\ & \cdot \text{PAR} \cdot E(x_r + bw \cdot rand)^2 + 0.5 \cdot \text{HMCR} \\ & \cdot \text{PAR} \cdot E(x_r - bw \cdot rand)^2 + (1-\text{HMCR}) \\ & \cdot E\left(x_{\text{new}}^2\right). \quad (10) \end{aligned}$$

Therefore, now, we need to find out $E(x_{\text{new}})$ and $E(x_{\text{new}}^2)$. x_{new} is taken from the search range in the following way:

$$x_{\text{new}} = x_{\min} + rand(0,1) \cdot (x_{\max} - x_{\min})$$

where $rand(0,1)$ denotes the uniformly distributed random number lying between 0 and 1. Furthermore, we consider $x_{\min} = -a$, $x_{max} = a$, and $rand(0,1) = R$. Therefore, now

$$x_{\text{new}} = -a + 2 \cdot a \cdot R \quad (11)$$

$$x_{\text{new}}^2 = a^2 + 4 \cdot a^2 \cdot R^2 - 4 \cdot a^2 \cdot R. \quad (12)$$

The probability distribution of the random numbers is assumed to obey a continuous uniform probability density function, which is shown in Fig. 2. $\psi(z)$ is the continuous uniform probability distribution function. In this case, $p = 0$, $q = 1$, and $z = R$. Thus, $E(x_{\text{new}}) = -a + 2 \cdot a \cdot E(R)$ [from (11)]. Therefore, $E(R)$ is computed as follows:

$$E(R) = \int_0^1 R \cdot \psi(R) \cdot dR = \int_0^1 R \cdot dR = \left[\frac{R^2}{2}\right]_0^1 = \frac{1}{2}. \quad (13)$$

Therefore

$$E(x_{\text{new}}) = -a + 2 \cdot E(R) \cdot a = -a + 2a \cdot (1/2) - a + a = 0. \quad (14)$$

Furthermore, $E(x_{\text{new}}) = -a + 2 \cdot E(R) \cdot a = -a + 2a \cdot (1/2) - a + a = 0$ [from (12)], and

$$E(R^2) = \int_0^1 R^2 \psi(R) dR = \int_0^1 R^2 dR = \left[\frac{R^3}{3}\right]_0^1 = \frac{1}{3}. \quad (15)$$

Therefore

$$E\left(x_{\text{new}}^2\right) = a^2 - 4a^2 \cdot \frac{1}{2} + 4a^2 \cdot \frac{1}{3} = \frac{a^2}{3}. \quad (16)$$

Thus, from (9), we get

$$E(Y_l) = \text{HMCR} \cdot \overline{x} \quad (17)$$

and from (10), we get

$$E\left(Y_l^2\right) = \text{HMCR} \cdot \overline{x^2} + (1-\text{HMCR}) \cdot \frac{a^2}{3} + \text{HMCR} \cdot \text{PAR} \cdot \frac{bw^2}{3}. \quad (18)$$

We know that $E(Var(Y)) = E(\overline{Y^2}) - E(\overline{Y}^2)$; hence, we shall have to separately compute $E(\overline{Y^2})$ and $E(\overline{Y}^2)$. The mean-square value of the population is given by

$$\overline{Y^2} = \frac{1}{m} \sum_{k=1}^m Y_k^2.$$

Therefore

$$\begin{aligned} E(\overline{Y^2}) &= \frac{1}{m} \sum_{k=1}^m E\left(Y_k^2\right) \\ &= \frac{1}{m} \cdot \sum_{k=1}^m \left[\text{HMCR} \cdot \overline{x^2} + (1-\text{HMCR}) \cdot \frac{a^2}{3} + \text{HMCR} \cdot \text{PAR} \cdot \frac{bw^2}{3}\right] \quad \text{[by (18)]} \\ &= \text{HMCR} \cdot \overline{x^2} + (1-\text{HMCR}) \cdot \frac{a^2}{3} + \text{HMCR} \cdot \text{PAR} \cdot \frac{bw^2}{3}. \quad (19) \end{aligned}$$

Now, we need to determine $E(\overline{Y}^2)$. We know that the population mean is given as $\overline{Y} = (1/m)\sum_{k=1}^m Y_k$ We have

$$\overline{Y}^2 = \left\{\frac{1}{m} \sum_{k=1}^m Y_k\right\}^2 = \frac{1}{m^2}\left[\sum_{k=1}^m Y_k^2 + \sum_{k \neq l} Y_k \cdot Y_l\right].$$

Furthermore

$$\begin{aligned} E(\overline{Y}^2) &= E\left\{\frac{1}{m} \sum_{k=1}^m Y_k\right\}^2 = \frac{1}{m^2} E\left[\sum_{k=1}^m Y_k^2 + \sum_{k \neq l} Y_k \cdot Y_l\right] \\ &= \frac{1}{m} E\left[\frac{1}{m} \sum_{k=1}^m Y_k^2\right] + \frac{1}{m^2} E\left[\sum_{k \neq l} Y_k \cdot Y_l\right] \\ &= \frac{1}{m} \cdot E[\overline{Y^2}] + \frac{1}{m^2} \cdot \sum_{k \neq l} E(Y_k) \cdot E(Y_l) \end{aligned}$$

[as Y_k and Y_l are independent random variables $E(Y_k \cdot Y_l) = E(Y_k) \cdot E(Y_l)$]

$$\Rightarrow E(\overline{Y}^2) = \frac{1}{m} \cdot E(\overline{Y^2}) + \frac{1}{m^2} \cdot m \cdot (m-1) \cdot [E(Y_k)]^2$$

[since $E(Y_k) = E(Y_l)$]

$$\Rightarrow E(\overline{Y}^2) = \frac{1}{m} \cdot E(\overline{Y^2}) + \frac{1}{m} \cdot (m-1) \cdot [E(Y_k)]^2.$$

Therefore

$$\begin{aligned} E(Var(Y)) &= E(\overline{Y^2}) - E(\overline{Y}^2) \\ &= \left(1 - \frac{1}{m}\right) \cdot E(\overline{Y^2}) + \frac{(m-1)}{m} \cdot [E(Y_k)]^2 \\ &= \frac{(m-1)}{m} \cdot \left[E(\overline{Y^2}) - \{E(Y_k)\}^2\right]. \end{aligned} \tag{20}$$

Putting values from (17) and (19) in (20), we get

$$E(Var(Y)) = \frac{(m-1)}{m} \cdot \left[\text{HMCR} \cdot \overline{x^2} + (1 - \text{HMCR}) \cdot \frac{a^2}{3} + \text{HMCR} \cdot \text{PAR} \cdot \frac{bw^2}{3}\right] - \frac{(m-1)}{m} \cdot [\text{HMCR} \cdot \overline{x}]^2.$$

Simplifying further, we have

$$E(Var(Y)) = \frac{m-1}{m} \cdot \left[\text{HMCR} \cdot Var(x) + \text{HMCR} \cdot (1 - \text{HMCR}) \cdot \overline{x}^2 + \frac{1}{3} \cdot \text{HMCR} \cdot \text{PAR} \cdot bw^2 + \frac{a^2}{3} \cdot (1 - \text{HMCR})\right]$$

and the theorem is proved.

Lemma 1.1: If HMCR is chosen to be very high (i.e., very near to 1) and the distance bandwidth parameter (bw) is chosen to be proportional to the standard deviation of the current population (i.e., $bw \propto \sigma(x) = \sqrt{Var(x)}$), then the expected population variance (without fitness-based selection) can exponentially grow over generations.

Proof: The expected variance of the intermediate Y population (obtained after the improvisation process) is given by Theorem 1 in (5). Now, in (5), if we make HMCR ≈ 1, then the terms containing $\overline{x}$, $\overline{x}^2$, and a have very less contribution to the overall expected population variance. Hence, if we choose $bw = \sigma(x) = k \cdot \sqrt{Var(x)}$ (i.e., the standard deviation of the current population), the expression becomes

$$E(Var(Y)) \approx \frac{(m-1)}{m} \cdot \left[\text{HMCR} \cdot Var(x) + \frac{1}{3} \cdot \text{HMCR} \cdot \text{PAR} \cdot bw^2\right]$$

[neglecting the terms containing $(1 - \text{HMCR})$ as it becomes insignificantly small for HMCR $\to 1$]

$$\Rightarrow E(Var(Y)) = \frac{(m-1)}{m} \cdot \left[\text{HMCR} + \frac{1}{3} \cdot k^2 \cdot \text{HMCR} \cdot \text{PAR}\right] \cdot Var(x). \tag{21}$$

From (19), it is evident that if we do not include selection in the algorithm, then the expected variance of the gth population (x_g) becomes

$$E(Var(x_g)) = \left\{\frac{(m-1)}{m} \cdot \text{HMCR} \cdot \left[1 + \frac{1}{3} \cdot k^2 \cdot \text{PAR}\right]^g\right\} \cdot Var(x_0) \tag{22}$$

where x_0 is the initial population at generation $g = 0$. In (22), if we choose the values of the parameters HMCR, PAR, and k in such a way that the term within the second brackets becomes grater than unity, then we can expect an exponential growth of population variance. This growth of expected population variance over generations gives the algorithm a strong explorative power, which is essential for an EA. This completes the proof of Lemma 1.1. Now, consider the following numerical example.

Example 1: If $m = 10$, $\text{HMCR} = 1.00$, $k = 1.17$, and $\text{PAR} = 0.67$, we have

$$\left\{\frac{(m-1)}{m} \cdot \text{HMCR} \cdot \left(1 + \frac{1}{3} \cdot k^2 \cdot \text{PAR}\right)\right\} = 1.175167$$

and just after 100 generations, the factor

$$\left\{\frac{(m-1)}{m} \cdot \text{HMCR} \cdot \left(1 + \frac{1}{3} \cdot k^2 \cdot \text{PAR}\right)\right\}^{100} = 1.0216 \times 10^7$$

which implies a very large population variance as per (22).

B. EHS

Note that taking $bw \propto \sigma(x) \Rightarrow bw = k \cdot \sqrt{Var(x)}$ provides us an additional control over the explorative power of HS through the proportionality constant k. It is evident from (22) that, by choosing a suitable value of k, for a given set of values of HMCR and PAR, we can make the population variance of HS exponentially vary over generations without selection. Selection in an EA generally promotes exploitation and helps the solutions to converge to a specific point of the search space. The new scheme of tuning bw (by making it proportional to the current population variance) provides HS with high explorative power that, together with the exploitative behavior due to selection, can yield very good results on a wide variety of objective functions. From this point onward, this new variant of HS will be called explorative HS (EHS). Note that according to [11], the expected population variance for an ES without fitness-based selection and with a mutation based on the addition of a random vector (having mutually independent normally distributed components with zero mean and variance σ^2) and dominant recombination (global discrete recombination) is

$$E(Var(x_g)) = \left(1 - \frac{1}{m}\right)^g \cdot Var(x_0) + m \cdot \left[1 - \left(1 - \frac{1}{m}\right)^g\right] \cdot \sigma^2 \tag{23}$$

where m is the population size (analogous to HMS in HS). For a large m, (23) reduces to

$$E(Var(x_g)) \approx Var(x_0) + g \cdot \sigma^2. \tag{24}$$

Since, in (24), $((m-1)/m) \cdot [\text{HMCR} + (1/3) \cdot k^2 \cdot \text{HMCR} \cdot \text{PAR}]$ can be greater than 1 depending on the value

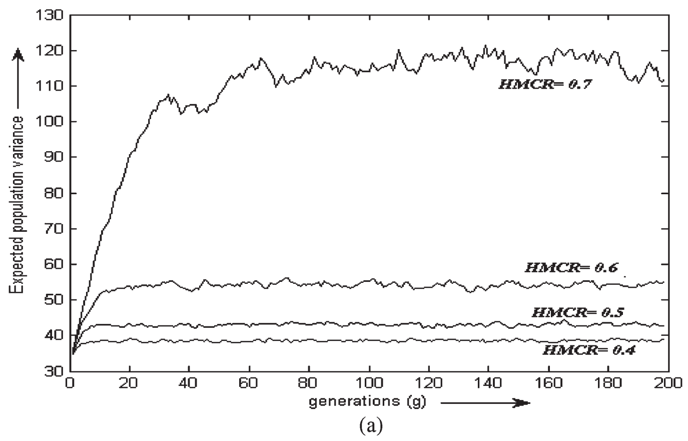


Fig. 3. Evolution of the expected population variance after HM improvisation process. (a) For actual HS. (b) As obtained from (5).

of k and this is more obvious for a large m as $(m-1)/m$ tends to 1, then it follows that the expected population variance (after applying the new HM improvisation) in the case of EHS can be greater than that in the case of the classical ES algorithm analyzed in [14], as the former may exponentially grow. Thus, the EHS algorithm has an explorative power greater than that of some of the classical ES algorithms. Note that the assumption $|_Lx_i| = |_Ux_i|$ was taken while proving Theorem 1 only to simplify the mathematical derivations and the closed-form expression for the evolution of the population variance. However, even if the absolute values of the lower and upper bound do not match, the basic assumptions derived on the nature of the evolution of the population variance remains valid, and EHS with the suggested adaptation rule for the *bandwidth* parameter performs much better as compared to the state-of-the-art HS variants. This is supported in Section VI by the results on constrained benchmarks f_{16} and f_{17}. In what follows, we shall extensively compare the performance of EHS with other state-of-the-art variants of HS, as well as with a few prominent swarm algorithms and EAs. The effect of varying the parameter k on the performance of EHS will also be investigated in Section VI-C.

C. Experimental Validation of Theorem 1 and Lemma 1.1

This section presents some computer simulation results to validate the results obtained in the previous sections. In Fig. 3, we compare the expected population variance plot and the theoretical variance plot [obtained from (5)] over generations for the classical HS. The expected population variance has been calculated by averaging the sample variance for all components and for 100 independent runs. In all the runs, the values of the parameters are chosen as follows: PAR $= 0.5$, HMS $= 10$, and $bw = 0.01$. The different curves in the plot are obtained for different values of the parameter HMCR. Note that for the plots shown in Fig. 3, the above parameter setting is not in accordance with Lemma 1.1. Therefore, instead of an exponential growth, the population variance gets saturated after a few generations. Fig. 3(a) and (b) shows the actual variance and the theoretical variance plots, respectively. Close correspondence of these two plots indicates the correctness of the expression (5) obtained through Theorem 1 in modeling the population variance of HS.

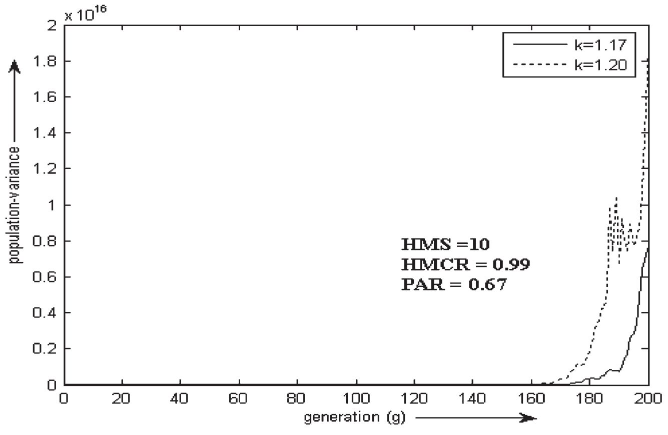
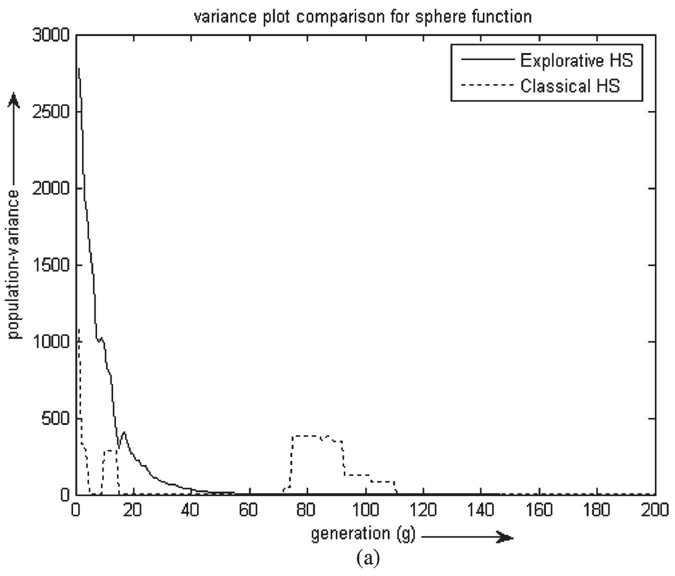


Fig. 4. Evolution of the expected population variance without selection and after choosing the parameter values in accordance with Lemma 1.1.

To justify the claim made in Lemma 1.1, the parameters HMCR, PAR, and k are chosen in such a way that the value of the expression $\{((m-1)/m) \cdot \text{HMCR} \cdot (1+(1/3) \cdot k^2 \cdot PAR)\}$ becomes greater than unity. Here, the value of PAR is kept constant at 0.5. The parameter HMCR is changed to meet the above criterion. While doing so, it is to be considered that the value of HMCR should be chosen to be very close to unity so that the dependence of the population variance on the population mean $(\overline{x})$ becomes negligible. HMS $= m$ is chosen to be 10 (which is the usual choice in HS community). Under these circumstances, the value of HMCR is conveniently chosen at 0.99, and k is kept at 1.17 and 1.20 (in this range, k gives very good results on benchmark functions, as discussed later in Section VI-C). The resultant evolution of population variance exhibited by HS is shown in Fig. 4. Fig. 5 shows an exponential growth of the expected population variance over generations. This supports the assertion made in Lemma 1.1. This exponential growth of population variance gives the algorithm a strong explorative power, which is essential for an EA.

Fig. 5(a) and (b) shows the evolution of the population variance over generations when selection is taken into account for two functions from our test suite provided in Table I, namely, the unimodal sphere model (f_1) and the multimodal generalized Rastrigin's function (f_5), in 15 dimensions, where



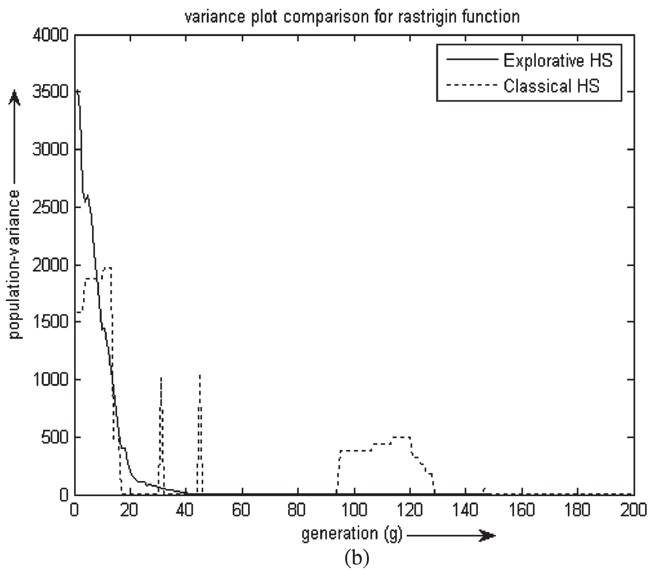


Fig. 5. Evolution of the expected population variance after HM improvisation process for the classical HS and EHS with selection. (a) Sphere model (f_1). (b) Generalized Rastrigin's function (f_5).

the search range for all the variables were kept in [−100, 100] for both the functions. We omit plots for rest of the functions for the sake of space economy and also considering the fact that the omitted plots show a more or less similar trend.

Note that, in HS-type selection, the new solution replaces the worst solution of the current generation provided that the fitness of the former is better. The selection process tends to remove the worse solutions that are far apart from the optima from the population and thereby promotes exploitation. The graphs indicate that, in EHS, the selection provides a good tradeoff to the high explorative power, which prevails during the earlier stages of the search. The population variance shows a steady fall toward zero, whence the solutions in the population converge to a small basin around the optima for EHS. However, for the classical HS, we see that the initial variance of the population is small (indicating a poor explorative power) and also the population variance shows fluctuating behavior over generations, a feature indicative of a poor tradeoff between exploration and exploitation.

V. Experimental Setup for Numerical Benchmarks

A. Unconstrained Benchmark Functions

We have used a testbed of 15 well-known unconstrained benchmark functions [38]–[40] to evaluate the performance of the EHS algorithm. The unconstrained benchmarks are briefly described in Table I. Note that the rotated and shifted functions are particularly challenging for many existing optimization algorithms. In case of rotations, when one dimension in the original vector $\vec{x}$ is changed, all dimensions of the rotated vector will be affected. Hence, the rotated function cannot be solved by just N 1-D searches. The composition functions are characterized by nonseparable search variables, rotated coordinates, and strong multimodality due to a huge number of local optima. They blend together the characteristics of different standard benchmarks.

B. Constrained Benchmark Functions

The functions f_{16}–f_{20}, listed below, are examples of GA-hard constrained optimization problems, extensively studied in [21] and [42]–[45], and in this paper, we have transformed them into an unconstrained one by adding a penalty term in the following way:

$$\psi_i(\vec{x}) = f_i(\vec{x}) + A_i \cdot \sum_{j=1}^{C_i} \max\{0, g_j(\vec{x})\} \tag{25}$$

where $i = 16, \ldots, 20$, C_i is the number of constraints with the ith benchnark, and A_i is the static penalty coefficient for the ith constrained benchmark problem.

The function f_{20} includes an equality constraint, which has been converted into an inequality constraint by using $|h(\vec{x}) - \delta| \leq 0$ using the degree of violation $\delta = 10^{-4}$. The values of the static penalty coefficients are given as follows [46]: $A_{16} = 0.5$, $A_{17} = 10^4$, $A_{18} = 1000$, $A_{19} = 500$, and $A_{20} = 10$. The following constrained optimization functions were considered in this paper.

1) $f_{16}(\vec{x}) = 5.\sum_{i=1}^{4} x_i - 5.\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i$, subject to $g_1(\vec{x}) = 2x_1 + 2x_2 + x_{10} + x_{11} - 10 \leq 0$, $g_2(\vec{x}) = 2x_1 + 2x_3 + x_{10} + x_{12} - 10 \leq 0$, $g_3(\vec{x}) = 2x_1 + 2x_3 + x_{11} + x_{12} - 10 \leq 0$, $g_4(\vec{x}) = -8x_1 + x_{10} \leq 0$, $g_5(\vec{x}) = -8x_2 + x_{11} \leq 0$, $g_6(\vec{x}) = -8x_3 + x_{12} \leq 0$, $g_7(x) = -2x_4 - x_5 + x_{10} \leq 0$, $g_8(\vec{x}) = -2x_6 - x_7 + x_{11} \leq 0$, and $g_9(\vec{x}) = -2x_8 - x_9 + x_{12} \leq 0$, where the bounds are $0 \leq x_i \leq 1 (i = 1, \ldots, 9)$, $0 \leq x_i \leq 100 (i = 10, 11, 12)$, and $0 \leq x_{13} \leq 1$. The global minimum value is -15 at $\vec{x} = (1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1)$.
2) $f_{17}(\vec{x}) = (x_1 + x_2 + x_3)$, subject to $g_1(\vec{x}) = -1 + 0.0025(x_4 + x_6) \leq 0$, $g_2(\vec{x}) = -1 + 0.0025(x_5 + x_7 - x_4) \leq 0$, $g_3(\vec{x}) = -1 + 0.01(x_8 - x_5) \leq 0$, $g_4 (\vec{x}) = -x_1x_6 + 833.3252x_4 + 100x_1 - 83\,333.33 \leq 0$, $g_5(\vec{x}) = -x_2x_7 + 1250x_5 + x_2x_4 - 1250x_4 \leq 0$, and $g_6(\vec{x}) = -x_3x_8 + 1\,250\,000 + x_3x_5 - 2500x_5 \leq 0$, where $100 \leq x_1 \leq 10\,000, 1000 \leq x_i \leq 10\,000 (i = 2, 3)$, and

TABLE I
UNCONSTRAINED BENCHMARK FUNCTIONS [38], [39]. IN FUNCTIONS f_{10} AND f_{11}, $\vec{z}=\vec{x}-\vec{o}$, WHERE $\vec{o}=\{o_1,\ldots,o_N\}$ IS THE SHIFTED GLOBAL OPTIMUM, AND FOR FUNCTIONS f_{12} AND f_{13}, $\vec{z}=M\cdot(\vec{x}-\vec{o})$, WHERE $\vec{o}=\{o_1,\ldots,o_N\}$ IS THE SHIFTED AND ROTATED GLOBAL OPTIMUM, AND M IS THE LINEAR TRANSFORMATION (ROTATION) MATRIX OBTAINABLE FROM [39]

Function Name	Expression	Search Range	Optimum Value
Sphere Function	$f_1(\vec{x})=\sum_{i=1}^{N} x_i^2$	$-100 \le x_i \le 100$	$\min(f_1)=f_1(0,\ldots,0)=0$
Generalized Rosenbrock's Function	$f_2(\vec{x})=\sum_{i=1}^{N-1}[100\,(x_{i+1}-x_i^{\,2})^2+(x_i-1)^2]$	$-30 \le x_i \le 30$	$\min(f_1)=f_1(1,\ldots,1)=0$
Step Function	$f_3(\vec{x})=\sum_{i=1}^{N}(\lfloor x_i+0.5\rfloor)^2$	$-100 \le x_i \le 100$	$\min(f_3)=f_3(\vec{p})=0$ such that $-\frac{1}{2}\le p_i \le \frac{1}{2}$ for $i=1,2,\ldots,N$
Rotated Hyper-Ellipsoid Function	$f_4(\vec{x})=\sum_{i=1}^{N}\left(\sum_{j=1}^{i} x_j\right)^2$	$-100 \le x_i \le 100$	$\min(f_4)=f_4(0,\ldots,0)=0$
Generalized Rastrigin's Function	$f_6(\vec{x})=\sum_{i=1}^{N}[x_i^2-10\cos(2\pi x_i)+10]$	$-5.12 \le x_i \le 5.12$	$\min(f_5)=f_5(0,\ldots,0)=0$
Generalized Griewank Function	$f_6(\vec{x})=\frac{1}{4000}\sum_{i=1}^{N} x_i^{\,2}-\prod_{i=1}^{N}\cos(\frac{x_i}{\sqrt{i}})+1$	$-600 \le x_i \le 600$	$\min(f_6)=f_6(0,\ldots,0)=0$
Ackley's Function	$f_7(\vec{x})=-20\exp\left(-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N}x_i^2}\right)-\exp\left(\frac{1}{N}\sum_{i=1}^{N}\cos(2\pi x_i)\right)+20+\exp(1)$	$-32 \le x_i \le 32$	$\min(f_7)=f_7(0,\ldots,0)=0$
Generalized Schwefel's Problem 2.26	$f_8(\vec{x})=\sum_{i=1}^{N}-x_i.\sin(\sqrt{\lvert x_i\rvert})$	$-500 \le x_i \le 500$	$\min(f_8)=f_8(420.9687,\ldots,420.9687)=-N\times 418.9829$
Weierstrass Function	$f_9(\vec{x})=\sum_{i=1}^{N}\left(\sum_{k=0}^{k\max}[a^k\cos(2\pi b^k(x_i+0.5))]\right)-N\sum_{k=0}^{k\max}[a^k\cos(2\pi b^k\cdot 0.5)]$ where $a=0.5, b=3, kmax=20$	$-0.5 \le x_i \le 0.5$	$\min(f_9)=f_9(0,\ldots,0)=0$
Shifted Rosenbrock's Function	$f_{10}(\vec{z})=\sum_{i=1}^{N-1}[100(z_{i+1}-z_i^{\,2})^2+(z_i-1)^2]+f\_bias_6$	$-100 \le x_i \le 100$	$\min(f_{10})=f\_bias_6=-390$
Shifted Rastrigin's Function	$f_{11}(\vec{z})=\sum_{i=1}^{N}[z_i^2-10\cos(2\pi z_i)+10]+f\_bias_9$	$-5 \le x_i \le 5$	$\min(f_{11})=f\_bias_9=-330$
Shifted Rotated High Conditional Elliptic Function	$f_{12}(\vec{x})=\sum_{i=1}^{N}(10^6)^{\frac{i-1}{n-1}}.z_i^2+f\_bias_3$	$-100 \le x_i \le 100$	$\min(f_{12})=f\_bias_3=-450$
Shifted Rotated Griewank's Function	$f_{13}(\vec{x})=\frac{1}{4000}\sum_{i=1}^{N}z_i^2-\prod_{i=1}^{N}\cos(\frac{z_i}{\sqrt{i}})+1+f\_bias_7$	$-100 \le x_i \le 100$	$\min(CF1)=120$
Rotated Hybrid Composition Function 1 (CF1)	marked as $f_{16}(\vec{x})$ in the CEC 2005 benchmark problem set [39] and is composed of two rotated Ackley's functions, two rotated sphere functions, two rotated Rastrigin's functions, two rotated Weierstrass functions, and two rotated Griewank's functions.	$-5 \le x_i \le 5$	$\min(CF1)=10$
Rotated Hybrid Composition Function 2 (CF2)	marked as $f_{18}(\vec{x})$ in the CEC 2005 benchmark problem set [39]. It has the same composition as that of CF1, however, the rotation and shift parameters are different	$-5 \le x_i \le 5$	$\min(CF2)=10$

$10 \le x_i \le 1000 (i=4,5,\ldots,8)$. The global minimum value is 7049.3307 at $\vec{x}=(579.3167, 1359.943, 5110.071, 182.0174, 295.5985, 217.9799, 286.4162, 395.5979)$.

3) $f_{18}(\vec{x})=x_1^2+x_2^2+x_1x_2-14x_1-16x_2+(x_3-10)^2+4(x_4-5)^2+(x_5-3)^2+2(x_6-1)^2+5x_7^2+7(x_8-11)^2+2(x_9-10)^2+(x_{10}-7)^2+45$, subject to $g_1(\vec{x})=-105+4x_1+5x_2-3x_7+9x_8\le 0$, $g_2(\vec{x})=10x_1-8x_2-17x_7+2x_8\le 0$, $g_3(\vec{x})=-8x_1+2x_2+5x_9-2x_{10}-12\le 0$, $g_4(\vec{x})=3(x_1-2)^2+4(x_2-3)^2+2x_3^2-7x_4-120\le 0$, $g_5(\vec{x})=5x_1^2+8x_2+(x_3-6)^2-2x_4-40\le 0$, $g_6(\vec{x})=x_1^2+2(x_2-2)^2-2x_1x_2+14x_5-6x_6\le 0$, $g_7(\vec{x})=0.5(x_1-8)^2+2(x_2-4)^2+3x_5^2-x_6-30\le 0$, and $g_8(\vec{x})=-3x_1+6x_2+12(x_9-8)^2-7x_{10}\le 0$, where $-10\le x_i\le 10 (i=1,2,\ldots,10)$. The global minimum value is 24.3062091 at $\vec{x}=(2.171996, 2.363683, 8.773926, 5.095984, 0.9906548, 1.430547, 1.321644, 9.828726, 8.280092, 8.375927)$.

4) $f_{19}(\vec{x})=(x_1-10)^2+5(x_2-12)^2+x_3^4+3(x_4-11)^2+10x_5^6+7x_6^2+x_7^4-4x_6x_7-10x_6-8x_7$, subject to $g_1(\vec{x})=-127+2x_1^2+3x_2^4+x_3+4x_4^2+5x_5\le 0$, $g_2(\vec{x})=-282+7x_1+3x_2+10x_3^2+x_4-x_5\le 0$, $g_3(\vec{x})=-196+23x_1+x_2^2+6x_6^2-8x_7\le 0$, and $g_4(\vec{x})=4x_1^2+x_2^2-3x_1x_2+2x_3^2+5x_6-11x_7\le 0$,

where $-10 \leq x_i \leq 10 (i = 1, 2, \ldots, 7)$. The global minimum value is 680.6300573 at $\vec{x} = (2.330499, 1.951372, -0.4775414, 4.365726, -0.6244870, 1.038131, 1.594227)$.

5) $f_{20}(\vec{x}) = (\exp(x_1x_2x_3x_4x_5))$, subject to $h_1(\vec{x}) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0$, $h_2(\vec{x}) = x_2x_3 - 5x_4x_5 = 0$, and $h_3(\vec{x}) = x_1^3 + x_2^3 + 1 = 0$, where $-2.3 \leq x_i \leq 2.3 (i = 1, 2)$, and $-3.2 \leq x_i \leq 3.2 (i = 3, 4, 5)$. The global minimum value is 0.0539498 at $\vec{x} = (-1.717143, 1.595709, 1.827247, -0.7636413, -0.763645)$.

C. Algorithms Compared and Their Parametric Setup for the Unconstrained Optimization Problems

The proposed EHS algorithm is compared with three state-of-the-art variants of HS called IHS [24], MHS [25], and GHS [26]. We omit results of the comparison with the classical HS to save space and also in consideration of the fact that, according to [24]–[26], GHS, MHS, and IHS perform much better than the classical HS over most of the benchmark instances. GHS and IHS use the same HMS = 30. Here, we disallow any kind of hand tuning of the parameters of the algorithms compared here over any benchmark problem. Hence, after performing a series of hand-tuning experiments, for IHS, we choose

$$\begin{aligned} \text{HMCR} &= 0.95 \quad \text{PAR}_{\min} = 0.35 \\ \text{PAR}_{\max} &= 0.99 \quad bw_{\min} = 1.00e-06 \\ bw_{\max} &= 1/20 \cdot (x_{\max} - x_{\min}) \end{aligned}$$

over all the unconstrained problems, as our preliminary experiments suggest that the IHS with this parametric setup provides uniformly good results over all the benchmarks presented here. Similarly, for GHS, we choose $\text{PAR}_{\min} = 0.01$, $\text{PAR}_{\max} = 0.99$, and HMCR = 0.9. The parameters for the MHS algorithm are set as those in [26], i.e., HMCR = 0.98, PAR = 0.1, $Nhm = 0.1 \times \text{HMS}$, $N\text{m}_1 = 500$, and $N\text{m}_2 = 200$.

Finally, for the proposed EHS, following values were selected: HMCR = 0.99, PAR = 0.33 (same as the classical HS), and $bw = k\sqrt{Var(x)}$ with $k = 1.17$.

The value of k was selected after a series of hand-tuning experiments to provide consistently good performance over all the benchmarks presented here. Section VI-C empirically investigates the effect of k on the performance of EHS. Our empirical experiments indicate that like IHS and GHS, EHS also is not quite sensitive to the choice of HMS, and no single choice is universally good over a large variety of problems. In general, HM resembles the short-term memory of a musician and should be small in size. However, since MHS uses a comparatively large HMS, we set the HMS for all algorithms to be equal to 50. This is done primarily to make the comparison fair enough, so that all HS variants may start from the same initial population over all the problems, and any difference in their performance may be attributed to their internal search operators.

Over the unconstrained optimization problems, we also compare the performance of EHS with that of four state-of-the-art evolutionary computing algorithms. These competitors are, respectively, known as LEA (an EA based on level-set evolution and Latin squares) [47], ALEP (EP with adaptive Lévy mutation) [48], CPSO-H$_6$ (a hybrid cooperative particle swarm optimizer) [49], differential evolution (DE) [50], [51], and G3 with PCX (a real coded GA with a generalized generation gap model and parent-centric recombination) [52]. For these five algorithms, we employ the best-suited parametric setup as available from their respective literatures. In the case of DE, we used the most widely used DE scheme known as DE/rand/1/bin with scale factor $F = 0.5$ and crossover rate $Cr = 0.9$. The detailed description of these algorithms can be found in the corresponding references, and we do not reiterate them here for the sake of space economy.

D. Algorithms Compared on the Constrained Optimization Problems

Apart from the classical HS, GHS, and IHS, we also compare the performance of EHS with two state-of-the-art EAs on the constrained optimization problems f_{16}–f_{20}. The first of these algorithms is the Runarsson–Yao (RY) [42] algorithm. It is a recently proposed method and obtains good performance on constrained optimization problems. The algorithm attempts to stochastically balance objective and penalty functions, i.e., via stochastic ranking, and presents a new view on penalty function methods in terms of the dominance of penalty and objective functions. The RY algorithm is used here with the probability parameter value $P_f = 0.475$ (for details, see [42]). The second competitor is a micro-GA (MGA) as a generalized hill-climbing operator for GA (GA-MGA) [43], which was also used to optimize the difficult constrained benchmarks. We will use these existing results for direct comparison in Section VI-B. In particular, the results for GA-MGA with the (35–35) fitness allocation scheme is used here, the details of which can be found in [43]. For testing over the constrained benchmarks, we keep the same parametric setup for the three HS variants, as described in Section V-B.

VI. Numerical Results and Discussions

A. Unconstrained Benchmark Functions

The comparative study presented on the unconstrained benchmarks focuses on the following performance metrics: 1) the quality of the final solution; 2) the convergence speed [measured in terms of the number of fitness function evaluations (FEs)]; and 3) the frequency of hitting the optima. For unconstrained benchmarks, an asymmetric initialization procedure was adopted here following [53] and [54].

1) Comparison of the Quality of the Final Solution: To judge the accuracy of different algorithms, we first let each of them run for a very long time over every benchmark function, until the number of FEs exceeds a given upper limit (fixed here to 4×10^5 for $N = 50$ dimensions). The mean and the standard deviation (within parentheses) of the best-of-run errors for 50 independent runs of each of the six algorithms are presented in Table II. Note that the best-of-the-run error corresponds to the absolute difference between the best-of-the-run value $f(\vec{x}_{\text{best}})$ and the actual optimum f^* of a particular objective function, i.e., $|f(\vec{x}_{\text{best}}) - f^*|$. The experiments reported here are for the number of dimensions $N = 50$ for functions f_1–f_{15}.

A nonparametric statistical test called Wilcoxon's rank sum test for independent samples [55], [56] is conducted at the 5% significance level in order to judge whether the results obtained with the best performing algorithm differ from the final results

TABLE II
AVERAGE ERROR AND STANDARD DEVIATION (IN PARENTHESIS) OF THE BEST-OF-RUN SOLUTIONS FOR 50 INDEPENDENT RUNS TESTED ON 15 UNCONSTRAINED BENCHMARK FUNCTIONS, EACH IN $N = 50$ DIMENSIONS. EACH RUN OF EACH ALGORITHM WAS CONTINUED UP TO 4×10^5 FEs

Func / Algorithms	f_1 (Sphere)	f_2 (Rosenbrock)	f_3 (Step)	f_4 (Rotated Hyper-Ellipsoid)	f_5 (Rastrigin)
CPSO-H_6	5.7534e-11† (2.1094e-11)	3.7263e-01† (1.8376e-01)	4.9283e-05† (2.0146e-05)	3.5712e-14† (7.2393e-13)	7.5712e-01† (1.2439e-01)
LEA	**6.8594e-15‡ (4.9847e-15)**	4.9274e-01† (8.7231e-01)	6.2713e-07 (8.5322e-07)	3.5358e-14† (1.4923e-10)	**2.5358e-17‡ (6.9563e-16)**
ALEP	6.3244e-04† (5.3432e-04)	2.8263e+01† (4.9826e+00)	1.1463e+00† (2.2437e+00)	1.4257e-02† (4.2926e-03)	6.0499e+00† (8.3452e-01)
G3 with PCX	4.82834e-05† (2.9387e-06)	6.2713e+00† (8.5223e-01)	4.9871e-03† (4.9127e-04)	3.7659e-05† (2.5936e-05)	1.5041e-01† (9.4235e-01)
DE/rand/1/bin	9.5340e-15‡ (3.2981e-15)	1.9325e+00† (2.8271e+00)	**4.8372e-08‡ (3.7263e-07)**	4.9384e-10† (1.4628e-08)	5.7362e-12 (3.8273e-14)
IHS	8.3392e-01† (4.8337e-02)	6.2761e+04† (3.8157e+03)	1.6472e+04† (2.6378e+03)	3.7552e+03† (9.3242e+02)	2.3468e+02† (2.8233e+01)
GHS	8.9437e-02† (1.0433e-02)	4.8610e+04† (2.7963e+03)	4.0361e+02† (2.9431e+00)	1.3482e+03† (6.5124e+01)	7.5463e+01† (4.9806e+00)
MHS	3.3574e-04† (2.4474e-04)	6.3198e+02† (4.8722e+01)	1.0333e+00† (1.3457e+00)	4.6308e+03† (1.2458e+03)	9. 6429e-01† (5.5854e-01)
EHS	3.0610e-12 (4.2572e-12)	**1.2094e-03 (3.8237e-04)**	1.9283e-07 (5.8672e-08)	**1.6362e-15 (4.1904e-15)**	6.3927e-12 (4.0475e-12)

Func / Algorithms	f_6 (Griewank)	f_7 (Ackley)	f_8 (Schwefel's Problem 2.26)	f_9 (Weierstrass)	f_{10} (Shifted Rosenbrock)
CPSO-H_6	5.6548e-02† (2.3031e-02)	1.7725e-09† (2.4893e-10)	1. 7685e-02‡ (4.9249e-03)	**8.5829e-14‡ (1.6792e-09)**	4.8274e+01† (3.9217e-01)
LEA	6.5132e-15† (1.7965e-13)	5.0520e-15 (7.638e-14)	**4.8865e-04‡ (2.4209e-06)**	2.9890e-12‡ (7.0836e-10)	1.7183e+00† (7.8713e-01)
ALEP	1.7382e-01† (4.093e-02)	3.71596e-02† (9.3228e-02)	1.4823e-01 (3.7743e-02)	9.6648e-05† (2.3331e-04)	7.9283e+02† (2.0635e+03)
G3 with PCX	6.8649e-04† (8.0353e-04)	3.4743e-06† (7.146e-06)	4.9274e-02‡ (7.9237e-04)	1.7984e-02† (6.8324e-03)	5.0982e+01† (3.5365e+02)
DE/rand/1/bin	3.4837e-14† (2.6367e-15)	**1.1253e-15 (3.4536e-14)**	1.4672e+01† (5.9283e+00)	1.8092e+01† (6.1253e+00)	2.4819e+01† (1.7365e+01)
IHS	1.9238e+02† (3.1569e+01)	1.1802e+01† (9.4657e-01)	5.9281e+01† (4.5331e+00)	1.6353e-03† (8.2304e-02)	7.2314e+05† (1.0882e+05)
GHS	4.9728e+01† (9.8384e+00)	8.76312e+00† (7.9245e-01)	1.1263e+02† (2.9041e+01)	9.5278e-01† (4.4272e-01)	4.0155e+07† (3.8080e+08)
MHS	1.0609e+00† (3.4546e-07)	9.3747e-06† (3.1042e-06)	7.6601e+00† (6.8161e+00)	7.6687e-05† (6.7367e-05)	3.6718e+06† (2.8266e+06)
EHS	**5.8924e-16 (6.0042e-15)**	2.0345e-15 (6.7564e-16)	1.7542e-01† (1.3425e-01)	6.0189e-09 (3.8271e-07)	**1.9273e-01 (4.0183e-01)**

Func / Algorithms	f_{11} (Shifted Rastrigin)	f_{12} (Shifted Rotated High Conditional Elliptic Function)	f_{13} (Shifted Rotated Griewank)	Hybrid Composite Function 1 (CF1)	Hybrid Composite Function 2 (CF2)
CPSO-H_6	3.0831e+00† (4.9291e-01)	8.0937e+03† (5.9284e+00)	1.8276e+00† (1.8276e+00)	1.2836e+02† (1.7004e+00)	7.5849e+02† (1.4642e+02)
LEA	5.9834e-02† (2.9931e-02)	2.1284e+03† (3.9435e+01)	1.3224e-02† (8.3000e-04)	1.1952e+02† (3.2841e+00)	5.1254e+02† (2.5865e+02)
ALEP	4.2632e+00† (2.5162e-01)	5.7352e+03† (3.1426e+00)	9.5343e-01† (2.8835e-03)	1.8723e+02† (8.9284e-01)	7.1672e+02† (2.8364e+02)
G3 with PCX	1.8284e+00† (7.5361e-01)	4.7271e+03† (8.1152e+01)	4.5624e-01† (2.8374e-02)	2.2411e+02† (1.8287e+01)	1.9142e+03† (2.643e+02)
DE/rand/1/bin	2.0468e+00† (3.8271e+00)	4.6729e+04† (2.6371e+01)	3.8279e-03† (2.6621e-03)	1.8371e+02† (1.1281e+02)	6.2317e+02† (1.5728e+02)
IHS	3.4409e+01† (1.8129e+00)	7.3722e+06† (1.7541e+04)	7.7171e+03† (8.3477e+01)	4.7381e+02† (1.8286e+01)	1.7423e+04† (5.8471e+03)
GHS	1.3821e+02† (2.1635e+01)	9.1354e+07† (8.6607e+06)	3.0574e+03† (9.7056e+01)	5.7583e+02† (7.6361e+01)	1.2043e+04† (1.0031e+01)
MHS	5.7898e+01† (8.0086e+00)	5.8944e+07† (3.5743e+07)	7.7929e+03† (2.7504e-02)	4.09323e+02† (1.7772e+01)	2.6472e+03† (8.2537e+02)
EHS	**1.0289e-03 (3.0189e-03)**	**1.3561e+03 (1.8382e+02)**	**8.5462e-04 (3.6172e-05)**	**1.1031e+02 (2.0293e+00)**	**4.3595e+02 (7.2617e+01)**

of rest of the competitors in a statistically significant way. In Table II, the mark † indicates that EHS performs statistically better than the corresponding algorithm as the P-values obtained with the rank sum test are less than 0.05 (5% significance level). On the other hand, the ‡ mark indicates that the corresponding algorithm is better than EHS.

Table II indicates that EHS performed better than the three most recently published HS variants over all the 15 benchmark instances. As revealed by Table II, over 10 out of 15 functions, EHS alone achieved the mean best final accuracy, beating all the competitor algorithms in a statistically significant manner (in 5% significance level). In two cases (f_5 and f_8), LEA outperformed EHS, which, however, managed to remain the third best algorithm (for f_5, the second best was DE/rand/1/bin, and for f_8, CPSO-H_6 occupied the second rank). DE/rand/1/bin remained another tough competitor of EHS, beating it statistically over functions f_1, f_3, and f_7. Furthermore, from Table II, we observe that, out of these three instances, for f_7, the differences of the final accuracy values obtained by EHS and the best algorithm, i.e., DE/rand/1/bin, are not statistically significant. CPSO-H_6 could beat EHS only in one case corresponding to function f_9. We also note that, in all benchmark instances, the performance of EHS is statistically superior to the performance of the three other state-of-the-art variants of HS. Since all the four HS variants start from the same initial population and maintained the same HMS, this difference in performance must be attributed to their internal search mechanisms, a fact that substantiates the usefulness of the modifications incorporated in EHS.

The convergence characteristics of the nine algorithms over eight representative benchmark instances in 50 dimensions have been shown in Fig. 6 in terms of the error (in logarithmic scale) of the median run of each algorithm versus the number of FEs. We omitted plots for all functions to save space and also in consideration of the fact that they display more or less the same trend. For the step function f_3, characterized by plateaus and discontinuity, owing to its higher explorative power, EHS maintained a steady convergence rate right from the start and

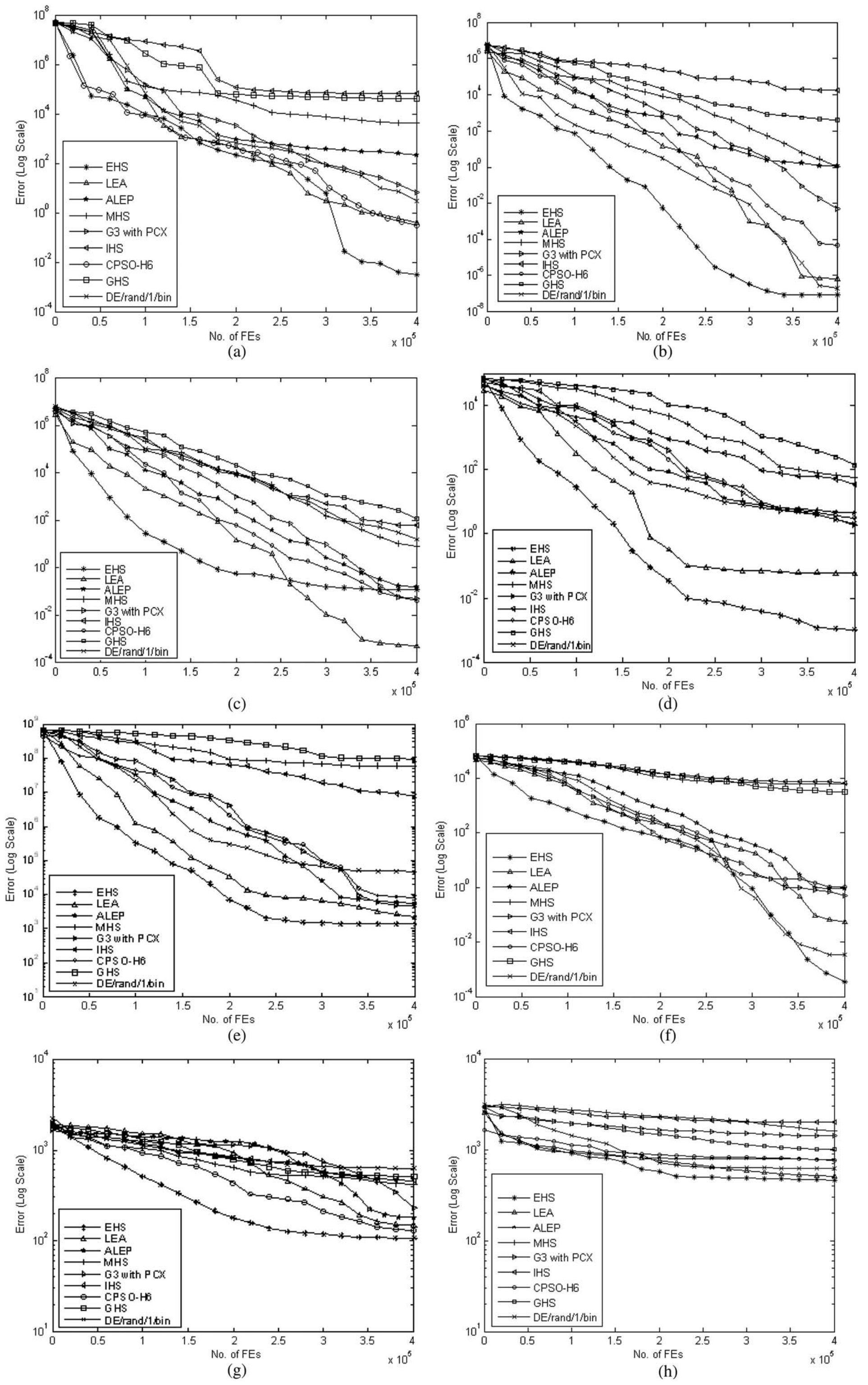


Fig. 6. Progress toward the optimum solution for the median run of nine algorithms over eight unconstrained test functions. (a) Generalized Rosenbrock's function (f_2). (b) Step function (f_2). (c) Generalized Schwefel's problem 2.26 (f_8). (d) Shifted Rastrigin's function (f_{11}). (e) Shifted and rotated hyperellipsoid (f_{12}). (f) Shifted and rotated Griewank's function (f_{13}). (g) Composite function 1 (CF1). (h) Composite function 2 (CF2).

finally finished at the lowest mean error, while the other two variants of HS showed a much slower convergence.

2) Comparison of the Convergence Speed and Success Rate: In order to compare the speeds of different algorithms, we select a threshold value of the error for each benchmark problem. For functions f_1 to f_9, this threshold is fixed at 10^{-5}; however, for more difficult functions (shifted, rotated, and composite) f_{10} to f_{15}, the threshold is set to 10^{-2}, in order to give a fair chance to all the metaheuristics compared. We run each algorithm on a function and stop as soon as the best error value determined by the algorithm falls below the predefined threshold or a maximum number of FEs is exceeded. Then, we

TABLE III
NUMBER OF SUCCESSFUL RUNS, MEAN NUMBER OF FEs, AND STANDARD DEVIATION (IN PARANTHESIS) REQUIRED TO CONVERGE TO THE THRESHOLD ERROR LIMIT OVER THE SUCCESSFUL RUNS FOR FUNCTIONS f_1 TO f_{15}

Func / Algorithms	f_1 (Sphere)	f_2 Rosenbrock)	f_3 (Step)	f_4 (Rotated Hyper-Ellipsoid)	f_5 (Rastrigin)
CPSO-H_6	50, 114563.25 (72653.44)	18, 176253.24 (34645.34)	40, 267474.25 (15327.08)	50, 134983.68 (14481.73)	21, 237583.67 (17432.82)
LEA	**50, 84628.60 (6583.23)**	26, 268931.52 (5712.65)	50, 273483.50 (13142.76)	50, 123924.16 (14028.47)	**50, 152483.50 (15622.76)**
ALEP	31,137822.63 (7198.45)	0	0	43, 129834.31 (10835.44)	7, 163586.82 (35271.78)
G3 with PCX	44, 157364.32 (21235.83)	4, 154563.25 (7653.44)	27, 312833.73 (29738.62)	26, 254982.74 (32432.87)	16, 236061.52 (14100.97)
DE/rand/1/bin	50, 87364.32 (13428.71)	6, 132763.70 (5582.48)	**50, 127512.34 (17238.49)**	**50, 131357.68 (17283.48)**	50, 178359.25 (5172.84)
IHS	0	0	0	25, 376093.80 (54827.57)	0
GHS	24, 136523.46 (17326.74)	0	0	15, 324782.17 (12283.49)	0
MHS	34, 121196.57 (6374.25)	0	0	33, 318769.67 (23692.58)	9, 284722.59 (15174.38)
EHS	50, 103932.64 (5450.492)	**29, 172228.72 (16473.45)**	50, 141029.75 (32732.68)	50, 186793.67 (12634.58)	50, 250039.62 (24831.23)
Func / Algorithms	**f_6 (Griewank)**	**f_7 (Ackley)**	**f_8 (Schwefel's Problem 2.26)**	**f_9 (Weierstrass)**	**f_{10} (Shifted Rosenbrock)**
CPSO-H_6	32, 257833.49 (15352.58)	50, 237263.92 (56432.45)	18, 231633.92 (64192.57)	**50, 137357.83 (18527.45)**	0
LEA	50, 169203.62 (27311.46)	50, 173945.56 (27465.26)	25, 285092.02 (71023.19)	50, 150923.56 (31364.29)	4, 276371.25 (17184.66)
ALEP	15, 345218.47 (27123.46)	23, 335411.45 (12893.56)	4, 232744.50 (27583.41)	43, 117232.25 (34812.67)	0
G3 with PCX	37, 263586.82 (55271.78)	46, 230272.50 (13642.289)	18, 233623.46 (11964.38)	32, 163778.58 (25385.31)	7, 287093.76 (14258.03)
DE/rand/1/bin	50, 133425.58 (14247.39)	50, 166783.62 (8835.56)	**50, 185562.84 (7382.48)**	0	9, 226358.31 (9470.37)
IHS	0	0	24, 308794.25 (46068.45)	34, 205472.02 (13109.56)	0
GHS	3, 272185.33 (10382.29)	4, 338392.75 (10231.48)	13, 145732.67 (2527.35)	20, 265978.50 (20741.27)	0
MHS	5, 368138.80 (6937.383)	42, 279372.87 (18242.03)	28, 326713.25 (11093.33)	42, 216279.52 (723.47)	0
EHS	**50, 124371.46 (11409.26)**	**50, 164722.34 (47212.38)**	41, 209183.67 (21253.24)	50, 170473.50 (23412.67)	**14, 206712.84 (46352.61)**
Func / Algorithms	**f_{11} (Shifted Rastrigin)**	**f_{12} (Shifted Rotated High Conditional Elliptic Function)**	**f_{13} (Shifted Rotated Griewank)**	**Hybrid Composite Function 1 (CF1)**	**Hybrid Composite Function 2 (CF2)**
CPSO-H_6	42, 328934.50 (45132.46)	0	42, 304982.64 (15182.67)	16, 191935.40 (3008.45)	2, 378273.50 (20938.47)
LEA	50, 86374.92 (5783.38)	6, 273922.67 (23814.25)	50, 138720.84 (20731.88)	11, 298521.54 (10832.41)	3, 354261.33 (18362.08)
ALEP	41, 197539.57 (14648.33)	2, 298342.22 (1421.68)	50, 206742.28 (18534.55)	15, 247234.72 (14451.72)	3, 366472.67 (16869.32)
G3 with PCX	35, 131453.20 (12873.66)	6, 2462744.69 (44583.41)	50,257362.57 (13417.34)	11, 378357.83 (23423.45)	0
DE/rand/1/bin	4, 301637.75 (51526.58)	1, 342495	50, 142735.64 (23718.59)	3, 354726.67 (44721.68)	0
IHS	**9, 358242.73 (20946.37)**	0	0	10, 350528.60 (13873.51)	0
GHS	0	0	0	7, 363728.43 (24279.53)	0
MHS	4, 234722.25 (13432.67)	0	0	11, 294812.82 (36173.52)	0
EHS	**50, 75834.68 (5942.76)**	**10, 209309.20 (17829.46)**	**50, 126473.93 (27268.45)**	**18, 213698.45 (34812.47)**	**4, 312673.25 (14824.32)**

note the number of FEs the algorithm takes. A lower number of FEs corresponds to a faster algorithm. Like the previous experiment (Section VI-A1), the maximum number of FEs for each function is kept at 4×10^5 for all the functions. Table III reports the number of runs (out of 50) that managed to find the optimum solution (within the given tolerance) without exceeding the maximum number of FEs, the mean number of FEs, and standard deviations (within parenthesis) required by the algorithms to converge within the prescribed threshold value. Entries marked as 0 indicate that no runs of the corresponding algorithm converged below the threshold objective function value without exceeding the upper limit of the number of FEs. Missing values of standard deviation in these tables also indicate a zero standard deviation. Table III shows that not only does EHS yield the most accurate results for nearly all the benchmark problems, but also it does so consuming the least amount of computational time. In addition, the number of runs that converge below a prespecified cutoff value is also greatest for EHS over most of the benchmark problems covered here. This indicates the higher robustness (i.e., the ability to produce similar results over repeated runs on a single problem) of the algorithm as compared to its other seven competitors.

TABLE IV
RESULTS OF THE COMPARATIVE STUDY ON FIVE CONSTRAINED NUMERICAL BENCHMARKS WHERE THE RESULTS FOR THE RY AND GA-MGA ALGORITHMS ARE OBTAINED FROM [42] AND [43], RESPECTIVELY

Func	Value type	EHS	IHS	GHS	RY	GA-MGA
f_{16}	Mean	**-15.000**	-14.9924	-14.9961	**-15.000**	**-15.000**
	Best	**-15.000**	-14.9956	-14.9989	**-15.000**	**-15.000**
	Worst	**-15.000**	-14.9918	-14.9946	**-15.000**	**-15.000**
	Std. Dev	**0.00**	0.00167	0.00235	**0.00**	**0.00**
	Mean No. of FEs (Std. Dev)	**38478.64 (1827.46)**	176390.82 (20193.57)	152 738.56 (11378.36)	350,000	350,000
	% of Trials giving Feasible Solutions	**100%**	70%	90%	**100%**	**100%**
f_{17}	Mean	**7264.854**	7708.648	7634.833	7559.613	7673.22
	Best	**7053.065**	7064.832	7061.465	7053.316	7055.15
	Worst	**7463.273**	11726.576	10736.537	8835.655	11860.88
	Std. Dev	**124.262**	904.731	815.36	530	888.4
	Mean No. of FEs (Std. Dev)	**68273.72 (3726.68)**	109294.84 (6351.48)	89283.07 (7728.41)	350,000	350,000
	% of Trials giving Feasible Solutions	**100%**	80%	90%	**100%**	90%
f_{18}	Mean	**24.363**	24.762	24.712	24.374	24.567
	Best	24.312	24.573	24.486	**24.307**	24.332
	Worst	**24.538**	25.174	25.095	24.642	25.177
	Std. Dev	**0.0526**	0.4712	0.3889	0.066	0.1615
	Mean No. of FEs (Std. Dev)	**67932.56**	217841.49 (10928.52)	117264.68 (9825.37)	350,000	350,000
	% of Trials giving Feasible Solutions	**100%**	90%	**100%**	**100%**	**100%**
f_{19}	Mean	**680.632**	721.472	694.738	680.656	**680.632**
	Best	680.630	680.685	680.658	680.630	**680.630**
	Worst	680.637	764.307	732.814	680.763	**680.635**
	Std. Dev	0.00133	34.732	21.652	0.036	**0.000924**
	Mean No. of FEs (Std. Dev)	**70936.48 (4838.77)**	192831.29 (26712.43)	90293.76 (7362.48)	350,000	350,000
	% of Trials giving Feasible Solutions	**100%**	70%	60%	**100%**	**100%**
f_{20}	Mean	**0.053986**	0.068372	0.067613	0.067543	0.054198
	Best	**0.053948**	0.056735	0.053978	0.053957	0.053951
	Worst	**0.054013**	0.276368	0.187264	0.216915	0.055813
	Std. Dev	**6.732×10^{-6}**	0.084	0.067	0.031	0.000629
	Mean No. of FEs (Std. Dev)	**28983.78 (3008.46)**	57583.30 (1272.84)	48723.42 (2957.03)	350,000	350,000
	% of Trials giving Feasible Solutions	**100%**	60%	90%	**100%**	**100%**

B. *Results for Constrained Benchmark Functions*

To further investigate the performance of the EHS algorithm, we use the five well-known constrained objective functions listed in Section V-A. In addition to comparing EHS with the two state-of-the-art HS variants, we also take into account two recent EAs—RY [42] and GA-MGA [43]—that were particularly devised for optimizing constrained objective functions. The numerical results for RY and GA-MGA have been taken from [42] and [43], respectively. Since Runarsson and Yao [42] used a termination criterion of 1750 generations (corresponding to 350 000 FEs) for the RY algorithm (GA-MGA results were also reported in [43] for termination criteria of 350 000 FEs) over the five benchmarks, in order to make the comparison fair, we compare both the qualities of their final solutions and the computational cost at the same value. Accordingly, the termination criterion of the three HS variants is that the quality of the best solution cannot further be improved in the successive 50 generations for each function. We reported the results for 100 independent runs of EHS, GHS, and IHS with different random seeds. In each run, the three HS variants start from the same initial population. The results for the RY and GA-MGA algorithms are for 30 and 100 independent trials for each function. Table IV reports the comparative performance of the five algorithms over the constrained benchmarks in terms of the following: 1) the mean function value of the n trials (Mean); 2) the best solution in n trials (Best); 3) the worst solution in the n trials (Worst); 4) the standard deviation of function values (Std. dev.); 5) the mean number of FEs with corresponding standard deviations, and 6) the percentage of trials that give feasible solutions, where $n = 100$ for EHS, GHS, IHS, and GA-MGA and $n = 30$ for the RY algorithm.

A detailed view of Table IV indicates that EHS outperforms both GHS and IHS on all the problem instances in terms of all the six metrics we considered. Table IV also compares EHS with RY. As can be seen, EHS and RY can find the exact global optimum in all the trials for f_{16}. For functions f_{17}, f_{18}, f_{19}, and f_{20}, the final solutions of EHS are better than or comparable to those of RY in terms of the six performance metrics. EHS gives a smaller standard deviation of the final objective function values than RY in these cases, and hence, EHS has a more stable solution quality.

For function f_{18}, the best solution found by RY (24.307) is better than that by EHS (24.312); however, the mean and worst solutions obtained with EHS are much better than those

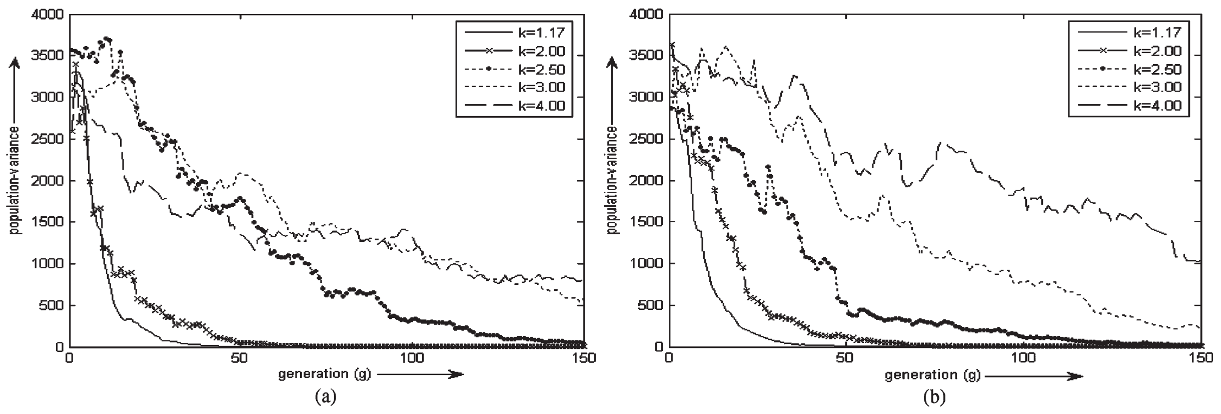


Fig. 7. Evolution of the expected population variance after HM improvisation process in EHS with selection for different values of k. (a) Sphere model (f_1). (b) Generalized Rastrigin's function (f_5).

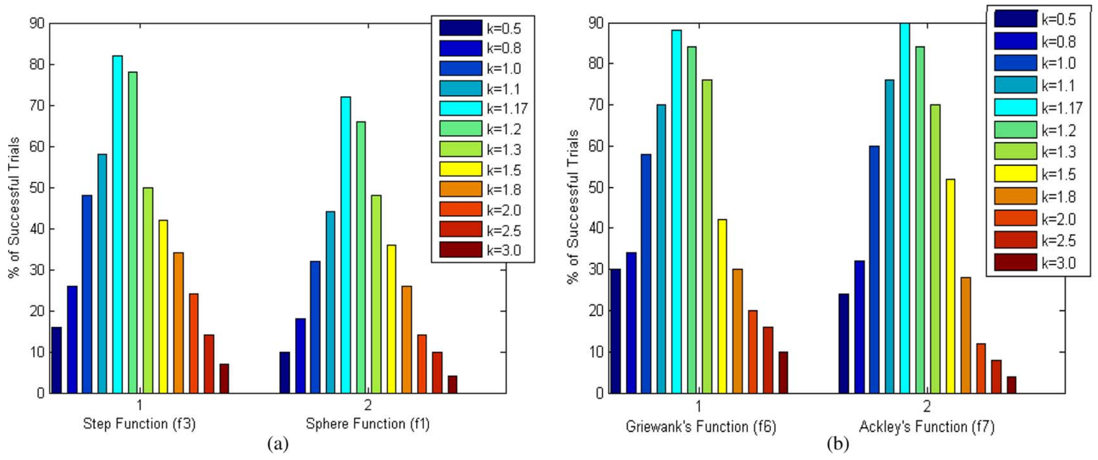


Fig. 8. Variation of the overall success rate of EHS with increasing neighborhood size (for four standard 50-D functions). Neighborhood sizes are indicated in the legend. (a) For step (f_3) and sphere (f_1) functions. (b) For Griewank (f_6) and Ackley's (f_7) functions.

obtained using the RY algorithm. As can be seen from Table IV, EHS and GA-MGA can find the exact global optimum in all trials for f_{16}. For functions f_{17}, f_{18}, and f_{20}, the solutions of EHS are better than those of GA-MGA. In particular, for f_{17}, the percentage of feasible solutions supplied by GA-MGA is 90%, while that of EHS is 100%. Only in case of function f_{19}, the best and worst solutions found by GA-MGA and the standard deviation of the function values were better than EHS, although both yielded an equal mean value over 100 trials. Note that the mean number of FEs per independent run of EHS is about 20 000 to 60 000 for all the functions, whereas that of RY and GA-MGA is 350 000. Therefore, EHS appears to be computationally more efficient than these two algorithms, at least over the test suite used here.

C. *Effect of the Parameter k on EHS Performance*

The proper selection of the proportionality constant k in the relation $bw = \sigma(x) = k \cdot \sqrt{Var(x)}$ for EHS affects the tradeoff between exploitation and exploration. For solving any given optimization problem, this selection remains an open problem. Some empirical guidelines may, however, be provided based on the fact that if the value of k is very high, then despite the selection, after only a few iterations, the population variance will increase so much that, due to overexploration, all the population members will be roaming near the boundary of the search volume. In that case, the convergence of the algorithm below a given threshold within a specified number of generations is nearly impossible. In fact, due to HS-type selection, it is observed that the best solution of the population will no longer improve after a few generations if the value of k is greater than 2.00. Fig. 7 shows the evolution of the expected population variance over generations for two representatives from the unconstrained test suite, i.e., functions f_1 (unimodal sphere model) and f_5 (generalized Rastrigin's function), in 30 dimensions, while rest of the functions show a similar trend. Here, we fix the maximum number of generations at 150 and observe that the HS population shows a trend of anticonvergence behavior due to overexploration for higher values of k. The graphs show a mean behavior of 50 independent trials over both functions.

Again, a too small value of k runs the risk of losing the diversity of the population, as the term within brackets $(\{((m-1)/m) \cdot \text{HMCR} \cdot (1+(1/3) \cdot k^2 \cdot \text{PAR})\})$ in (22) may become much smaller than 1, leading the population variance near zero after only a few iterations. Our experiments suggest that the value of k around 0.5–0.6 makes the performance of EHS comparable to that of the classical HS and inferior to IHS, MHS, and GHS due to the lack of sufficient explorative power.

In Fig. 8, we provide the overall success rate of the EHS algorithm for a k varying from 0.5 to 3.0 over four representative unconstrained functions f_1, f_3, f_6, and f_7 in 50 dimensions. We again do not iterate through the plots of all the functions to save space and time and also since they exhibit a similar trend. Since all the functions have their optima at the origin (0), we plot the percentage of runs that successfully yielded a final accuracy value below 10^{-5} for different k values. Fifty independent trials were executed in each case on each function. Thorough experimentation with all the test problems shows that keeping k around 1.2 (we kept it at 1.17) provides reasonably accurate results with high success rates over most of the benchmark problems covered here.

VII. Conclusion

This paper has presented a mathematical analysis of the evolution of population variance for the HS metaheuristic algorithm. The theoretical results indicate that the population variance of HS can be made to exponentially vary by making the distance bandwidth of HS proportional to the standard deviation of the current population. The proportionality constant provides us an additional control over the explorative power of HS for fixed values of other control parameters. An attractive property of the EHS algorithm proposed here is that it does not introduce any complex operations or additional burdens to the original simple HS framework in terms of FEs. The only difference from the original PSO is the way the parameter bw is adapted.

The new EHS algorithm has been compared with the most recently published HS variants and a few other well-known EAs and swarm-based algorithms over a testbed of 15 unconstrained and five constrained numerical benchmarks. The following performance metrics have been used: 1) solution quality; 2) speed of convergence; and 3) frequency of hitting the optimum. The EHS algorithm has been shown to always outperform the other three HS variants over all of the tested problems. Moreover, EHS has been found to meet or beat a state-of-the-art variant of PSO (CPSO-H_6), two standard real-coded EAs (LEA and G3 with PCX), and an improved version of EP (ALEP) in a statistically significant fashion.

Future research should address the issues like theoretically establishing an optimal range of values for k, which improves the performance of the algorithm over a wide range of functions. The parameter k may be made time varying or adapted in such a way that, during the early stages of search, exploration is emphasized, but during the later stages of search, exploitation is favored, depending on the nature of the fitness landscape. The parameter may even be self-adapted so that the algorithm itself determines the optimal value of k, capturing any special feature of the problem at hand. Other modifications to the algorithm leading to better tradeoffs between explorative and exploitative tendencies should also be investigated both empirically and theoretically. An analysis of the timing complexity of HS-type algorithms may also be worth considering in the future.

References

[1] T. Bäck, D. Fogel, and Z. Michalewicz, *Handbook of Evolutionary Computation*. London, U.K.: Oxford Univ. Press, 1997.

[2] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*. New York: Springer-Verlag, 2003.

[3] A. P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*. Hoboken, NJ: Wiley, 2006.

[4] J. Kennedy, R. C. Eberhart, and Y. Shi, *Swarm Intelligence*. San Francisco, CA: Morgan Kaufmann, 2001.

[5] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: Harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, Feb. 2001.

[6] K. S. Lee and Z. W. Geem, "A new metaheuristic algorithm for continuous engineering optimization: Harmony search theory and practice," *Comput. Methods Appl. Mech. Eng.*, vol. 194, no. 36–38, pp. 3902–3933, Sep. 2004.

[7] Z. W. Geem, Ed., *Music-Inspired Harmony Search Algorithm: Theory and Applications*. New York: Springer-Verlag, 2009, ser. Studies in Computational Intelligence Series.

[8] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "Harmony search optimization: Application to pipe network design," *Int. J. Model. Simul.*, vol. 22, no. 2, pp. 125–133, 2002.

[9] K. S. Lee and Z. W. Geem, "A new structural optimization method based on the harmony search algorithm," *Comput. Struct.*, vol. 82, no. 9/10, pp. 781–798, Apr. 2004.

[10] Z. W. Geem, K. S. Lee, and Y. Park, "Application of harmony search to vehicle routing," *Amer. J. Appl. Sci.*, vol. 2, no. 12, pp. 1552–1557, 2005.

[11] A. Vasebi, M. Fesanghary, and S. M. T. Bathaeea, "Combined heat and power economic dispatch by harmony search algorithm," *Int. J. Elect. Power Energy Syst.*, vol. 29, no. 10, pp. 713–719, Dec. 2007.

[12] Z. W. Geem, *Optimal Scheduling of Multiple Dam System Using Harmony Search Algorithm*. New York: Springer-Verlag, 2007, pp. 316–323.

[13] A. E. Eiben and C. A. Schippers, "On evolutionary exploration and exploitation," *Fundamenta Informaticae*, vol. 35, no. 1–4, pp. 1–16, Aug. 1998.

[14] H.-G. Beyer, "On the dynamics of EAs without selection," in *Proc. FOGA-5*, 1999, pp. 5–26.

[15] H.-G. Beyer, "On the explorative power of ES/EP-like algorithms," *Evol. Programming*, pp. 323–334, 1998.

[16] A. Mukhopadhyay, A. Roy, S. Das, and S. Das, "Population-variance and explorative power of harmony search: An analysis," in *Proc. 2nd MATEIT—Organized at C*, New Delhi, India, Sep. 26–28, 2008.

[17] R. Parncutt, *Harmony: A Psychoacoustical Approach*. New York: Springer-Verlag, 1989.

[18] Z. W. Geem and S. Kim, *Harmony Search Algorithms for Structural Design*. Philadelphia, PA: Old City Publishing, 2009.

[19] K. Deb, "Optimal design of a welded beam via genetic algorithms," *AIAA J.*, vol. 29, no. 1, pp. 2013–2015, 1991.

[20] D. B. Fogel, "A comparison of evolutionary programming and genetic algorithms on selected constrained optimization problems," *Simulation*, vol. 64, no. 6, pp. 399–406, 1995.

[21] Z. Michalewicz and M. Schoenauer, "Evolutionary algorithms for constrained parameter optimization problems," *Evol. Comput.*, vol. 4, no. 1, pp. 1–32, 1996.

[22] Z. W. Geem, "Improved harmony search from ensemble of music players," in *Proc. KES*, vol. 4251, *Lecture Notes on Artificial Intelligence,* B. Gabrys, R. J. Howlett, and L. C. Jain, Eds., 2006, pp. 86–93.

[23] Z. W. Geem, "Novel derivative of harmony search algorithm for discrete design variables," *Appl. Math. Comput.*, vol. 199, no. 1, pp. 223–230, May 2008.

[24] M. Mahdavi, M. Fesanghary, and E. Damangir, "An improved harmony search algorithm for solving optimization problems," *Appl. Math. Comput.*, vol. 188, no. 2, pp. 1567–1579, May 2007.

[25] M. G. H. Omran and M. Mahdavi, "Global-best harmony search," *Appl. Math. Comput.*, vol. 198, no. 2, pp. 643–656, May 2008.

[26] Y. M. Cheng, L. Li, T. Lansivaara, S. C. Chi, and Y. J. Sun, "An improved harmony search minimization algorithm using different slip surface generation methods for slope stability analysis," *Eng. Optim.*, vol. 40, no. 2, pp. 95–115, Feb. 2008.

[27] Z. W. Geem and J.-Y. Choi, "Music composition using harmony search algorithm," in *Proc. EvoWorkshops*, vol. 4448, *LNCS*, M. Giacobini, D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, and G. Weikum, Eds., 2007, pp. 593–600.

[28] Z. W. Geem, "Harmony search algorithm for solving sudoku," in *Proc. KES/WIRN*, vol. 4692, *LNAI,* B. Apolloni, R. J. Howlett, and L. Jain, Eds., 2007, pp. 371–378.

[29] Z. W. Geem, C. Tseng, and Y. Park, *Harmony Search for Generalized Orienteering Problem: Best Touring in China*. New York: Springer-Verlag, 2005, pp. 741–750.

[30] M. Fesanghary, E. Damangir, and I. Soleimani, "Design optimization of shell and tube heat exchangers using global sensitivity analysis and harmony search algorithm," *Appl. Thermal Eng.* [Online]. Available: http://dx.doi.org/10.1016/j.applthermaleng.2008.05.018

[31] A. R. Doodman, M. Fesanghary, and R. Hosseini, "A robust stochastic approach for design optimization of air cooled heat exchangers," *Appl. Energy*, vol. 86, no. 7/8, pp. 1240–1245, Jul./Aug. 2009.

[32] O. Zareia, M. Fesangharyb, B. Farshia, R. J. Saffarb, and M. R. Razfar, "Optimization of multi-pass face-milling via harmony search algorithm," *J. Mater. Process. Technol.*, vol. 29, no. 5, pp. 2386–2392, Mar. 2009, DOI:10.1016/j.jmatprotec.2008.05.029.

[33] M. Mahdavi, M. H. Chehreghani, H. Abolhassani, and R. Forsati, "Novel meta-heuristic algorithms for clustering web documents," *Appl. Math. Comput.*, vol. 201, no. 1/2, pp. 441–451, Jul. 2008.

[34] R. Forsati, A. T. Haghighat, and M. Mahdavi, "Harmony search based algorithms for bandwidth-delay-constrained least-cost multicast routing," *Comput. Commun.*, vol. 31, no. 10, pp. 2505–2519, Jun. 25, 2008.

[35] W. S. Jang, H. Kang, and B. H. Lee, "Hybrid simplex-harmony search method for optimization problems," in *Proc. IEEE CEC*, Jun. 1–6, 2008, pp. 4157–4164.

[36] L. Coelho and D. Bernerta, "An improved harmony search algorithm for synchronization of discrete-time chaotic systems," *Chaos Solitons Fractals*, vol. 42, no. 5, pp. 2526–2532, Sep. 2009, DOI:10.1016/j.chaos.2008.09.028.

[37] D. Zaharie, "Recombination operators for evolutionary algorithms," in *Proc. 27th Summer School, Appl. Math. Eng. Economics*, 2001.

[38] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.

[39] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," Nanyang Technol. Univ., Singapore, May 2005.

[40] Y. W. Shang and Y. H. Qiu, "A note on the extended Rosenbrock function," *Evol. Comput.*, vol. 14, no. 1, pp. 119–126, Mar. 2006.

[41] D. Whitley, D. Rana, J. Dzubera, and E. Mathias, "Evaluating evolutionary algorithms," *Artif. Intell.*, vol. 85, no. 1/2, pp. 245–276, Aug. 1996.

[42] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 4, no. 3, pp. 284–294, Sep. 2000.

[43] S. A. Kazarlis, S. E. Papadakis, J. B. Theochairs, and V. Petridis, "Microgenetic algorithms as generalized hill-climbing operators for GA optimization," *IEEE Trans. Evol. Comput.*, vol. 5, no. 3, pp. 204–217, Jun. 2001.

[44] S. Koziel and Z. Michalewicz, "Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization," *Evol. Comput.*, vol. 7, no. 1, pp. 19–44, 1999.

[45] Z. Michalewicz and G. Nazhiyath, "Genocop III: A co-evolutionary algorithm for numerical optimization problems with nonlinear constraints," in *Proc. 2nd IEEE Conf. Evol. Comput.*, 1995, vol. 2, pp. 647–651.

[46] L. Jiao, Y. Li, M. Gong, and X. Zhang, "Quantum-inspired immune clonal algorithm for global numerical optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 38, no. 5, pp. 1234–1253, Oct. 2008.

[47] Y. Wang and C. Dang, "An evolutionary algorithm for global optimization based on level-set evolution and Latin squares," *IEEE Trans. Evol. Comput.*, vol. 11, no. 5, pp. 579–595, Oct. 2007.

[48] C. Y. Lee and X. Yao, "Evolutionary programming using mutations based on the Lévy probability distribution," *IEEE Trans. Evol. Comput.*, vol. 8, no. 1, pp. 1–13, Feb. 2004.

[49] F. V. den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 225–239, Jun. 2004.

[50] R. Storn and K. V. Price, Differential evolution—A simple and efficient adaptive scheme for global optimization over continuous spaces, ICSI, Berkeley, CA, Tech. Rep. TR-95-012. [Online]. Available: http://http.icsi.berkeley.edu/~storn/litera.html

[51] K. Price, R. Storn, and J. Lampinen, *Differential Evolution—A Practical Approach to Global Optimization*. Berlin, Germany: Springer-Verlag, 2005.

[52] K. Deb, A. Anand, and D. Joshi, "A computationally efficient evolutionary algorithm for real-parameter optimization," *Evol. Comput.*, vol. 10, no. 4, pp. 371–395, Dec. 2002.

[53] D. Fogel and H.-G. Beyer, "A note on the empirical evaluation of intermediate recombination," *Evol. Comput.*, vol. 3, no. 4, pp. 491–495, 1995.

[54] P. J. Angeline, "Evolutionary optimization versus particle swarm optimization: Philosophy and the performance difference," in *Proc. 7th Int. Conf. Evol. Programming—Evolutionary Programming VII*, vol. 1447, *Lecture Notes in Computer Science*, 1998, pp. 84–89.

[55] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics*, vol. 1, no. 6, pp. 80–83, 1945.

[56] S. Garcıacute;a, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behavior: A case study on the CEC'2005 special session on real parameter optimization," *J. Heuristics*, vol. 15, no. 6, pp. 617–644, Dec. 2009, DOI: 10.1007/s10732-008-9080-4.

Swagatam Das received the B.E.Tel.E., M.E.Tel.E (control engineering specialization), and Ph.D. degrees from Jadavpur University, Kolkata, India, in 2003, 2005, and 2009, respectively.

He is currently an Assistant Professor with the Department of Electronics and Telecommunication Engineering, Jadavpur University. He is a coauthor of a research monograph on metaheuristic clustering techniques published by Springer in 2009. Currently, he is coauthoring a textbook on swarm and evolutionary computing, to be published from IEEE-Wiley press in early 2011. He has published more than 100 research articles in peer-reviewed journals and international conference proceedings. His current research interests include evolutionary computing, swarm intelligence, pattern recognition, bioinformatics, control systems engineering, and digital signal processing.

Dr. Das has been acting as a regular Reviewer for journals like *Pattern Recognition*, IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS, and IEEE TRANSACTIONS ON SMC—PART B and PART A. He serves as an Associate Editor of the *Information Sciences* (Elsevier) and as an Editorial Board Member of the *International Journal of Artificial Intelligence and Soft Computing* and *International Journal of Adaptive and Autonomous Communication Systems* (Inderscience publications).

Arpan Mukhopadhyay was born in West Bengal, India, in 1987. He received the B.E. degree in electronics and telecommunication engineering from Jadavpur University, Kolkata, India, in 2009. He is currently working toward the M.S. degree in communication engineering with the Indian Institute of Science, Bangalore, India.

His current area of research is wireless communication and wireless networks.

Anwit Roy was born in West Bengal, India, in 1987. He received the B.E. degree in electronics and telecommunication engineering from Jadavpur University, Kolkata, India, in 2009.

He is currently an Assistant Engineer with Cognizant Technology Solutions, Inc.

Ajith Abraham (M'96–SM'07) received the M.S. degree from Nanyang Technological University, Singapore, and the Ph.D. degree in computer science from Monash University, Melbourne, Vic., Australia.

He is currently coordinating the activities of the Machine Intelligence Research Labs (MIR Labs), Scientific Network for Innovation and Research Excellence, Auburn, WA, which has members from more than 60 countries. He has a worldwide academic experience with formal appointments in Monash University; Oklahoma State University, Stillwater, OK; Chung-Ang University, Seoul, Korea; Jinan University, Guangzhou, China; Rovira i Virgili University, Tarragona, Spain; Dalian Maritime University, Dalian, China; Yonsei University, Seoul, Korea; the Open University of Catalonia, Barcelona, Spain; the National Institute of Applied Sciences (INSA-Lyon), Lyon, France; and the Norwegian University of Science and Technology (NTNU), Trondheim, Norway. He serves the editorial board of several reputed International journals and has also guest edited 35 special issues on various topics. He has published more than 600 publications, and some of the works have also won best paper awards at international conferences. His research and development experience includes more than 20 years in the industry and academia. He works in a multidisciplinary environment involving machine intelligence, terrorism informatics, network security, sensor networks, e-commerce, Web intelligence, Web services, computational grids, data mining, and their applications to various real-world problems. He has given more than 35 plenary lectures and conference tutorials in these areas.

Dr. Abraham is a Senior Member of the IEEE Systems Man and Cybernetics Society, the IEEE Computer Society, the Institution of Engineering and Technology (U.K.), the Institution of Engineers Australia (Australia), etc. He is a Cochair of the IEEE Systems Man and Cybernetics Society Technical Committee on Soft Computing. He is actively involved in the Hybrid Intelligent Systems (HIS); Intelligent Systems Design and Applications (ISDA); Information Assurance and Security (IAS); and Next Generation Web Services Practices (NWeSP) series of international conferences, in addition to other conferences.

Bijaya K. Panigrahi (SM'06) received the Ph.D. degree from Sambalpur University, Sambalpur, India.

Since 2005, he has been an Assistant Professor with the Department of Electrical Engineering, Indian Institute of Technology (IIT) Delhi, New Delhi, India. Prior to joining IIT Delhi, he was a Lecturer with the University College of Engineering, Burla, Sambalpur, for 13 Years. His research areas include the study of advance signal processing techniques, computational intelligence algorithms, and their applications to the electrical engineering domain, particularly to the domain of power systems. His main research focuses on the development of advanced DSP tools and machine intelligence techniques for power quality studies, protection of power systems, etc. He also works in the area of application of evolutionary computing techniques (GA, PSO, clonal algorithm, ACO, bacterial foraging, harmony search, etc.) to solve the problems related to power system planning, operation, and control.