Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2012, Article ID 860681, 15 pages
doi:10.1155/2012/860681

Research Article

An Elite Decision Making Harmony Search Algorithm for Optimization Problem

Lipu Zhang,[1] Yinghong Xu,[2] and Yousong Liu[3]

[1] *Department of Mathematics, Zhejiang A&F University, Zhejiang 311300, China*
[2] *Department of Mathematics, Zhejiang Sci-Tech University, Zhejiang 310018, China*
[3] *State Key Laboratory of Software Engineering, Wuhan University, Hubei 430072, China*

Correspondence should be addressed to Lipu Zhang, zhanglipu@shu.edu.cn

Received 5 April 2012; Revised 26 May 2012; Accepted 10 June 2012

Academic Editor: Ricardo Perera

This paper describes a new variant of harmony search algorithm which is inspired by a well-known item "elite decision making." In the new algorithm, the good information captured in the current global best and the second best solutions can be well utilized to generate new solutions, following some probability rule. The generated new solution vector replaces the worst solution in the solution set, only if its fitness is better than that of the worst solution. The generating and updating steps and repeated until the near-optimal solution vector is obtained. Extensive computational comparisons are carried out by employing various standard benchmark optimization problems, including continuous design variables and integer variables minimization problems from the literature. The computational results show that the proposed new algorithm is competitive in finding solutions with the state-of-the-art harmony search variants.

1. Introduction

In 2001, Geem et al. [1] proposed a new metaheuristic algorithm, harmony search (HS) algorithm, which imitates the behaviors of music improvisation process. In that algorithm, the harmony in music is analogous to the optimization solution vector, and the musicians improvisations are analogous to local and global search schemes in optimization techniques. The HS algorithm does not require initial values for the decision variables. Furthermore, instead of a gradient search, the HS algorithm uses a stochastic random search that is based on the harmony memory considering rate and the pitch adjusting rate so that derivative information is unnecessary. These features increase the flexibility of the HS algorithm and have led to its application to optimization problems in different areas including music composition [2], Sudoku puzzle solving [3], structural design [4, 5], ecological conservation [6],

and aquifer parameter identification [7]. The interested readers may refer the review papers [8–10] and the references therein for further understanding.

HS algorithm is good at identifying the high performance regions of the solution space at a reasonable time but gets into trouble in performing local search for numerical applications. In order to improve the fine-tuning characteristic of HS algorithm, Mahdavi et al. [11] discussed the impacts of constant parameters on HS algorithm and presented a new strategy for tuning these parameters. Wang and Huang [12] used the harmony memory (HM) (set of solution vectors) to automatically adjust parameter values. Fesanghary et al. [13] use sequential quadratic programming technique to speed up local search and improve precision of the HS algorithm solution. Omran and Mahdavi [14] proposed a so-called the global best HS algorithm, in which concepts from swarm intelligence are borrowed to enhance the performance of HS algorithm such that the new harmony can mimic the best harmony in the HM. Also, Geem [15] proposed a stochastic derivative for discrete variables based on an HS algorithm to optimize problems with discrete variables and problems in which the mathematical derivative of the function cannot be analytically obtained. Pan et al. [16] used the good information captured in the current global best solution to generate new harmonies. Jaberipour and Khorram [17] described two HS algorithms through parameter adjusting technique. Yadav et al. [18] designed an HS algorithm which maintains a proper balance between diversification and intensification throughout the search process by automatically selecting the proper pitch adjustment strategy based on its HM. Pan et al. [19] divided the whole HM into many small-sized sub-HMs and performed the evolution in each sub-HM independently and thus presented a local-best harmony search algorithm with dynamic subpopulations. Later on, the excellent ideas of mutation and crossover strategies used in [19] were adopted in designing the differential evolution algorithm and obtained perfect result for global numerical optimization by Islam et al. [20].

Considering that, in political science and sociology, a small minority (elite) always holds the most power in making the decisions, that is, elite decision making. One could image that the good information captured in the current elite harmonies can be well utilized to generate new harmonies. Thus, in our elite decision making HS (EDMHS) algorithm, the new harmony will be randomly generated between the best and the second best harmonies in the historic HM, following some probability rule. The generated harmony vector replaces the worst harmony in the HM, only if its fitness (measured in terms of the objective function) is better than that of the worst harmony. These generating and updating procedures repeat until the near-optimal solution vector is obtained. To demonstrate the effectiveness and robustness of the proposed algorithm, various benchmark optimization problems, including continuous design variables and integer variables minimization problems, are used. Numerical results reveal that the proposed new algorithm is very effective.

This paper is organized as follows. In Section 2, a general harmony search algorithm and its recently developed variants will be reviewed. Section 3 introduces our method that has "Elite-Decision-Making" property. Section 4 presents the numerical results for some well-known benchmark problems. Finally, conclusions are given in the last section.

2. Harmony Search Algorithm

In the whole paper, the optimization problem is specified as follows:

$$\text{Minimize } f(x), \quad \text{subject to } x_i \in X_i, \quad i = 1, 2, \ldots, N, \tag{2.1}$$

where $f(x)$ is an objective function, x is the set of each decision variable (x_i), N is the number of decision variables, and X_i is the set of the possible range of values for each decision variable, that is $x_i^L \le X_i \le x_i^U$ and x_i^L and x_i^U are the lower and upper bounds for each decision variable, respectively.

2.1. The General HS Algorithm

The general HS algorithm requires several parameters as follows:

HMS: harmony memory size,

HMCR: harmony memory considering rate,

PAR: pitch adjusting rate,

bw: bandwidth vector.

Remarks. HMCR, PAR, and bw are very important factors for the high efficiency of the HS methods and can be potentially useful in adjusting convergence rate of algorithms to the optimal solutions. These parameters are introduced to allow the solution to escape from local optima and to improve the global optimum prediction of the HS algorithm.

The procedure for a harmony search, which consists of Steps 1–4.

Step 1. Create and randomly initialize an HM with HMS. The HM matrix is initially filled with as many solution vectors as the HMS. Each component of the solution vector is generated using the uniformly distributed random number between the lower and upper bounds of the corresponding decision variable $[x_i^L, x_i^U]$, where $i \in [1, N]$.

The HM with the size of HMS can be represented by a matrix as

$$\mathrm{HM} = \begin{pmatrix} x_1^1 & x_2^1 & \cdots & x_N^1 \\ x_1^2 & x_2^2 & \cdots & x_N^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{\mathrm{HMS}} & x_2^{\mathrm{HMS}} & \cdots & x_N^{\mathrm{HMS}} \end{pmatrix}. \tag{2.2}$$

Step 2. Improvise a new harmony from the HM or from the entire possible range. After defining the HM, the improvisation of the HM, is performed by generating a new harmony vector $x' = (x_1', x_2', \ldots, x_N')$. Each component of the new harmony vector is generated according to

$$x_i' \longleftarrow \begin{cases} x_i' \in \mathrm{HM}\ (:, i) & \text{with probability HMCR,} \\ x_i' \in X_i, & \text{with probability 1-HMCR,} \end{cases} \tag{2.3}$$

where HMCR is defined as the probability of selecting a component from the HM members, and (1-HMCR) is, therefore, the probability of generating a component randomly from the possible range of values. Every x_i' obtained from HM is examined to determine whether it

should be pitch adjusted. This operation uses the PAR parameter, which is the rate of pitch adjustment as follows:

$$x_i' \longleftarrow \begin{cases} x_i' \pm \text{rand}[0,1] \times \text{bw} & \text{with probability PAR,} \\ x_i', & \text{with probability 1-PAR,} \end{cases} \tag{2.4}$$

where rand$[0,1]$ is the randomly generated number between 0 and 1.

Step 3. Update the HM. If the new harmony is better than the worst harmony in the HM, include the new harmony into the HM and exclude the worst harmony from the HM.

Step 4. Repeat Steps 2 and 3 until the maximum number of searches is reached.

2.2. The Improved HS Algorithm

To improve the performance of the HS algorithm and eliminate the drawbacks associated with fixed values of PAR and bw, Mahdavi et al. [11] proposed an improved harmony search (IHS) algorithm that uses variable PAR and bw in improvisation step. In their method, PAR and bw change dynamically with generation number as expressed below:

$$\text{PAR}(\text{gn}) = \text{PAR}_{\min} + \frac{\text{PAR}_{\max} - \text{PAR}_{\min}}{\text{MaxItr}} \times \text{gn}, \tag{2.5}$$

where PAR(gn) is the pitch adjusting rate for each generation, $\text{PAR}_{\min}$ is the minimum pitch adjusting rate, $\text{PAR}_{\max}$ is the maximum pitch adjusting rate, and MaxItr and gn is the maximum and current search number, respectively. We have

$$\text{bw}(\text{gn}) = \text{bw}_{\max} e^{c \times \text{gn}}, \tag{2.6}$$

where

$$c = \frac{\log(\text{bw}_{\min}/\text{bw}_{\max})}{\text{MaxIter}}. \tag{2.7}$$

Numerical results reveal that the HS algorithm with variable parameters can find better solutions when compared to HS and other heuristic or deterministic methods and is a powerful search algorithm for various engineering optimization problems, see [11].

2.3. Global Best Harmony Search (GHS) Algorithm

In 2008, Omran and Mahdavi [14] presented a GHS algorithm by modifying the pitch adjustment rule. Unlike the basic HS algorithm, the GHS algorithm generates a new harmony vector x' by making use of the best harmony vector $x^{\text{best}} = \{x_1^{\text{best}}, x_2^{\text{best}}, \ldots, x_n^{\text{best}}\}$ in the HM. The pitch adjustment rule is given as follows:

$$x_j' = x_k^{\text{best}}, \tag{2.8}$$

where k is a random integer between 1 and n. The performance of the GHS is investigated and compared with HS. The experiments conducted show that the GHS generally outperformed the other approaches when applied to ten benchmark problems.

2.4. A Self-Adaptive Global Best HS (SGHS) Algorithm

In 2010, Pan et al. [16] presented a SGHS algorithm for solving continuous optimization problems. In that algorithm, a new improvisation scheme is developed so that the good information captured in the current global best solution can be well utilized to generate new harmonies. The pitch adjustment rule is given as follows:

$$x_j' = x_j^{\text{best}}, \tag{2.9}$$

where $j = 1, \ldots, n$. Numerical experiments based on benchmark problems showed that the proposed SGHS algorithm was more effective in finding better solutions than the existing HS, HIS, and GHS algorithms.

3. An Elite Decision Making HS Algorithm

The key differences between the proposed EDMHS algorithm and IHS, GHS, and SGHS are in the way of improvising the new harmony.

3.1. EDMHS Algorithm for Continuous Design Variables Problems

The EDMHS has exactly the same steps as the IHS with the exception that Step 3 is modified as follows.

In this step, a new harmony vector $x' = (x_1', x_2', \ldots, x_N')^T$ is generated from

$$x_i' \longleftarrow \begin{cases} x_i' \in [\mathrm{HM}(s,i), \mathrm{HM}(b,i)] & \text{with probability HMCR,} \\ x_i' \in \mathrm{X}_i, & \text{with probability 1-HMCR,} \end{cases} \tag{3.1}$$

where $\mathrm{HM}(s,i)$ and $\mathrm{HM}(b,i)$ are the ith element of the second-best harmony and the best harmony, respectively.

3.2. EDMHS Algorithm for Integer Variables Problems

Many real-world applications require the variables to be integers. Methods developed for continuous variables can be used to solve such problems by rounding off the real optimum values to the nearest integers [14, 21]. However, in many cases, rounding-off approach may result in an infeasible solution or a poor suboptimal solution value and may omit the alternative solutions.

In EDMHS algorithm for integer programming, we generate the integer solution vector in the initial step and improvise step, that is, each component of the new harmony vector is generated according to

$$x_i' \longleftarrow \begin{cases} x_i' \in \text{round}([\text{HM}(s,i), \text{HM}(b,i)]) & \text{with probability HMCR,} \\ x_i' \in X_i, & \text{with probability 1-HMCR,} \end{cases} \tag{3.2}$$

where round$(*)$ means round off for $(*)$. The pitch adjustment is operated as follows:

$$x_i' \longleftarrow \begin{cases} x_i' \pm 1 & \text{with probability PAR,} \\ x_i', & \text{with probability 1-PAR.} \end{cases} \tag{3.3}$$

4. Numerical Examples

This section is about the performance of the EDMHS algorithm for continuous and integer variables examples. Several examples taken from the optimization literature are used to show the validity and effectiveness of the proposed algorithm. The parameters for all the algorithm are given as follows: HMS = 20, HMCR = 0.90, $\text{PAR}_{\min} = 0.4$, $\text{PAR}_{\max} = 0.9$, $\text{bw}_{\min} = 0.0001$, and $\text{bw}_{\max} = 1.0$. In the processing of the algorithm, PAR and bw are generated according to (2.5) and (2.6), respectively.

4.1. Some Simple Continuous Variables Examples

For the following five examples, we adopt the same variable ranges as presented in [4]. Each problem is run for 5 independent replications, the mean fitness of the solutions for four variants HS algorithm, IHS, SGHS, SGHS, and EDMHS, is presented in tables.

4.1.1. Rosenbrock Function

Consider the following:

$$f(x) = 100\left(x_2 - x_1^2\right)^2 + (1 - x_1)^2. \tag{4.1}$$

Due to a long narrow and curved valley present in the function, Rosenbrock function [4, 22] is probably the best known test case. The minimum of the function is located at $x^* = (1.0, 1.0)$ with a corresponding objective function value of $f(x^*) = 0.0$. The four algorithms were applied to the Rosenbrock function using bounds between −10.0 and 10.0 for the two design variables x_1 and x_2. After the 50,000 searches, we arrived at Table 1.

Table 1: Four HS algorithms for Rosenbrock function.

Variables	IHS	GHS	SGHS	EDMHS
x_1	1.0000028617324386	0.9913653798835682	1.0000082201314386	0.9999992918896516
x_2	1.0000062226347253	0.9837656861940776	1.0000169034081148	0.9999985841159521
$f_1(x)$	0.0000000000331057	0.0001667876726056	0.0000000000890147	**0.0000000000005014**

Table 2: Four HS algorithms for Goldstein and Price function I.

Variables	IHS	GHS	SGHS	EDMHS
x_1	0.0000043109765698	−0.0108343859912985	−0.0000010647548017	−0.0000022210968748
x_2	−0.9999978894568922	−1.0091267108154769	−1.0000037827893109	−1.0000008657021768
$f(x)$	3.0000000046422932	3.0447058568657721	3.0000000055974083	**3.0000000011515664**

4.1.2. Goldstein and Price Function I (with Four Local Minima)

Consider the following:

$$f(x) = \left\{1 + (x_1 + x_2 + 1)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right\} \times \left\{30 + (2x_1 - 3x_2)^2\left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right\}. \tag{4.2}$$

Goldstein and Price function I [4, 13, 23] is an eighth-order polynomial in two variables. However, the function has four local minima, one of which is global, as follows: $f(1.2, 0.8) = 840.0$, $f(1.8, 0.2) = 84.0$, $f(-0.6, -0.4) = 30$, and $f^*(0.0, 1.0) = 3.0$ (global minimum). In this example, the bounds for two design variables (x_1 and x_2) were set between −5.0 and 5.0. After 8000 searches, we arrived at Table 2.

4.1.3. Eason and Fenton's Gear Train Inertia Function

Consider the following:

$$f(x) = \frac{1}{10}\left\{12 + x_1^2 + \frac{1 + x_2^2}{x_1^2} + \frac{x_1^2x_2^2 + 100}{(x_1x_2)^4}\right\}. \tag{4.3}$$

This function [4, 24] consists of a minimization problem for the inertia of a gear train. The minimum of the function is located at $x^* = (1.7435, 2.0297)$ with a corresponding objective function value of $f^*(x) = 1.744152006740573$. The four algorithms were applied to the gear train inertia function problem using bounds between 0.0 and 10.0 for the two design variables x_1 and x_2. After 800 searches, we arrived at Table 3.

Table 3: Four HS algorithms for Eason and Fenton's gear train inertia function.

Variables	IHS	GHS	SGHS	EDMHS
x_1	1.7434541913586368	1.7131403370902785	1.7434648607226395	1.7434544417399731
x_2	2.0296978640691021	2.0700437540873073	2.0296831598594332	2.0296925490097708
$f(x)$	1.7441520055927637	1.7447448145676987	**1.7441520056712445**	1.7441520055905921

Table 4: Four HS algorithms for Wood function.

Variables	IHS	GHS	SGHS	EDMHS
x_1	0.9367413185752959	0.9993702652662329	0.9917327966129160	1.0001567183702584
x_2	0.8772781982936317	0.9987850979456709	0.9835814785067265	1.0003039053776117
x_3	1.0596918740170123	0.9993702652662329	1.0081526992384837	0.9998357549633209
x_4	1.1230215213184420	0.9987850979456709	1.0164353912102084	0.9996725376532794
$f(x)$	0.0136094062872233	0.0000602033138483	0.0002433431550602	**0.0000001061706105**

4.1.4. Wood Function

Consider the following:

$$f(x) = 100\left(x_2 - x_1^2\right)^2 + (1 - x_1)^2 + 90\left(x_4 - x_3^2\right)^2 + \left(1 - x_3^2\right)^2 + 10.1\left[(x_2 - 1)^2 + (x_4 - 1)^2\right] + 19.8(x_2 - 1)(x_4 - 1). \tag{4.4}$$

The Wood function [4, 25] is a fourth-degree polynomial, that is, a particularly good test of convergence criteria and simulates a feature of many physical problems quite well. The minimum solution of the function is obtained at $x^* = (1, 1, 1, 1)^T$, and the corresponding objective function value is $f^*(x) = 0.0$. When applying the four algorithms STO the function, the four design variables, x_1, x_2, x_3, x_4, were initially structured with random values bounded between −5.0 and 5.0, respectively. After 70,000 searches, we arrived at Table 4.

4.1.5. Powell Quartic Function

Consider the following:

$$f(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4. \tag{4.5}$$

The second derivative of the Powell quartic function [4, 26] becomes singular at the minimum point, it is quite difficult to obtain the minimum solution (i.e., $f^*(0, 0, 0, 0) = 0.0$) using gradient-based algorithms. When applying the EDMHS algorithm to the function, the four design variables, x_1, x_2, x_3, x_4, were initially structured with random values bounded between −5.0 and 5.0, respectively. After 50,000 searches, we arrived at Table 5.

It can be seen from Tables 1–5, comparing with IHS, GHS, and SGHS algorithms, that the EDMHS produces the much better results for four test functions. Figures 1–5 present

Table 5: Four HS algorithms for Powell quartic function.

Variables	IHS	GHS	SGHS	EDMHS
x_1	−0.0383028653671760	−0.0256621703960072	0.0334641210434073	−0.0232662093056917
x_2	0.0038093414837046	0.0023707007810820	−0.0033373644857512	0.0023226342970439
x_3	−0.0195750968208506	−0.0199247989791340	0.0159748222727847	−0.0107227792768697
x_4	−0.0195676609811871	−0.0199247989791340	0.0160018633328343	−0.0107574107951817
$f(x)$	0.0000046821615160	0.0000070109937353	0.0000024921236096	**0.0000005715572753**

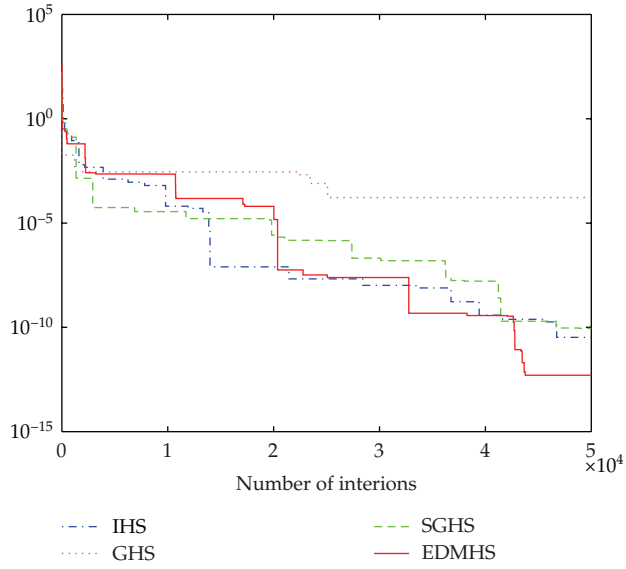

Figure 1: Convergence of Rosenbrock function.

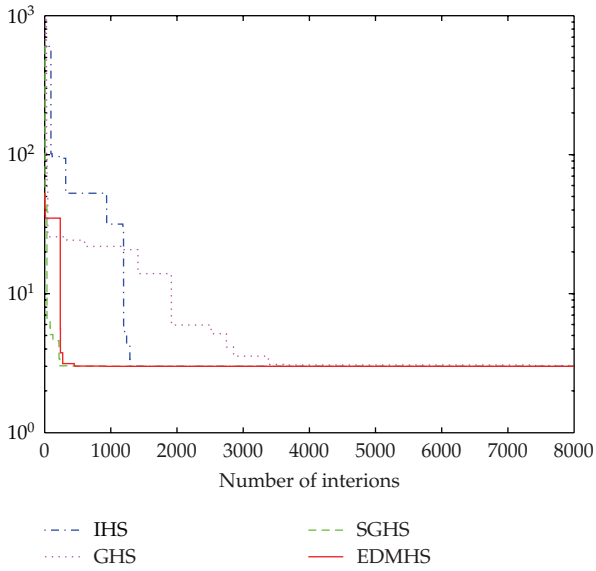

Figure 2: Convergence of Goldstein and Price function I.

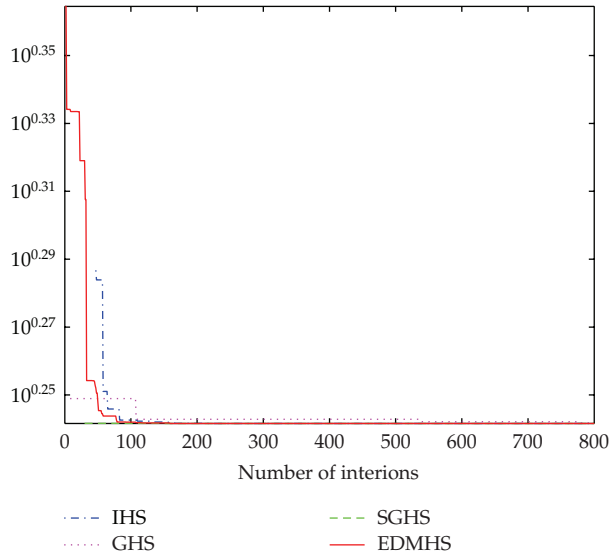

Figure 3: Convergence of Eason and Fenton function.

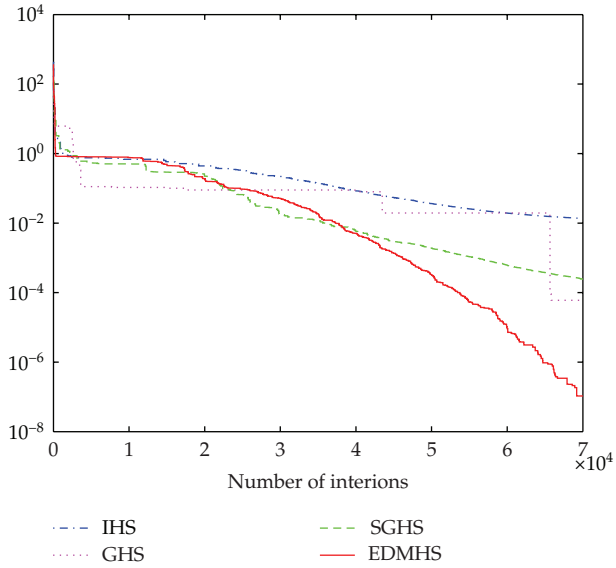

Figure 4: Convergence of Wood function.

a typical solution history graph along iterations for the five functions, respectively. It can be observed that four evolution curves of the EDMHS algorithm reach lower level than that of the other compared algorithms. Thus, it can be concluded that overall the EDMHS algorithm outperforms the other methods for the above examples.

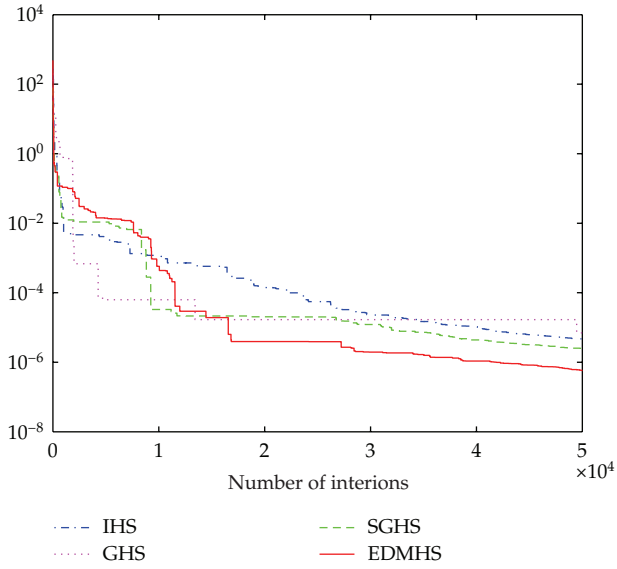

Figure 5: Convergence of Powell quartic function.

4.2. More Benchmark Problems with 30 Dimensions

To test the performance of the proposed EDMHS algorithm more extensively, we proceed to evaluate and compare the IHS, GHS, SGHS, and EDMHS algorithms based on the following 6 benchmark optimization problems listed in CEC2005 [27] with 30 dimensions.

(1) Sphere function:

$$f(x) = \sum_{i=1}^{n} x_i^2, \tag{4.6}$$

where global optimum $x^* = 0$ and $f(x^*) = 0$ for $-100 \le x_i \le 100$.

(2) Schwefel problem:

$$f(x) = -\sum_{i=1}^{n} \left(x_i \sin\left(\sqrt{|x_i|} \right) \right), \tag{4.7}$$

where global optimum $x^* = (420.9687, \ldots, 420.9687)$ and $f(x^*) = -12569.5$ for $-500 \le x_i \le 500$.

(3) Griewank function:

$$f(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1, \tag{4.8}$$

where global optimum $x^* = 0$ and $f(x^*) = 0$ for $-600 \le x_i \le 600$.

Table 6: AE and SD generated by the compared algorithms.

Problem	IHS AE	IHS SD	GHS AE	GHS SD	SGHS AE	SGHS SD	EDMHS AE	EDMHS SD
Sphere	$1.27e-07$	$1.79e-07$	$1.41e-02$	$2.36e-02$	$5.30e-07$	$1.27e-06$	$1.07e-07$	$1.31e-07$
Schwefel	$4.83e-01$	$7.06e-01$	$2.11e-02$	$3.01e-02$	$7.70e-01$	$1.15e-00$	$7.03e-01$	$1.61e-00$
Griewank	$1.18e-01$	$1.87e-01$	$8.83e-02$	$1.57e-01$	$8.02e-03$	$1.05e-02$	$1.02e-02$	$1.51e-02$
Rastrigin	$9.72e-01$	$1.18e-00$	$1.09e-02$	$2.05e-02$	$1.12e-00$	$1.43e-00$	$1.48e-00$	$1.93e-00$
Ackley	$5.11e-01$	$6.06e-01$	$2.05e-02$	$2.83e-02$	$2.13e-01$	$2.98e-01$	$3.34e-01$	$3.85e-01$
Rosenbrock	$3.37e+01$	$4.08e+01$	$6.77e+01$	$8.97e+01$	$3.46e+01$	$3.80e+01$	$3.17e+01$	$4.02e+01$

(4) Rastrigin function:

$$f(x) = \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i) + 10 \right), \tag{4.9}$$

where global optimum $x^* = 0$ and $f(x^*) = 0$ for $-5.12 \le x_i \le 5.12$.

(5) Ackley's function:

$$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{n} x_i^2} \right) - \exp\left(\frac{1}{30}\sum_{i=1}^{n} \cos(2\pi x_i) \right) + 20 + e, \tag{4.10}$$

where global optimum $x^* = 0$ and $f(x^*) = 0$ for $-32 \le x_i \le 32$.

(6) Rosenbrock's Function:

$$f(x) = \sum_{i=1}^{n-1} \left(100\left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right), \tag{4.11}$$

where global optimum $x^* = (1, \ldots, 1)$ and $f(x^*) = 0$ for $-5 \le x_i \le 10$.

The parameters for the IHS algorithm, HMS = 5, HMCR = 0.9, $\mathrm{bw}_{\max} = (x_j^U - x_j^L)/20$, $\mathrm{bw}_{\min} = 0.0001$, $\mathrm{PAR}_{\min} = 0.01$, and $\mathrm{PAR}_{\max} = 0.99$ and for the GHS algorithm, HMS = 5, HMCR = 0.9, $\mathrm{PAR}_{\min} = 0.01$, and $\mathrm{PAR}_{\max} = 0.99$.

Table 6 presents the average error (AE) values and standard deviations (SD) over these 30 runs of the compared HS algorithms on the 6 test functions with dimension equal to 30.

4.3. Integer Variables Examples

Six commonly used integer programming benchmark problems are chosen to investigate the performance of the EDMHS integer algorithm. For all the examples, the design variables, $x_i, i = 1, \ldots, N$, are initially structured with random integer values bounded between −100 and 100, respectively. Each problem is run 5 independent replications, each with approximately 800 searches, all the optimal solution vector are obtained.

4.3.1. Test Problem 1

Consider the following:

$$f_1(x) = \left(9x_1^2 + 2x_2^2 - 11\right)^2 + \left(3x_1^2 + 4x_2^2 - 7\right)^2, \tag{4.12}$$

where $x^* = (1,1)^T$ and $f_1(x^*) = 0$, see [14, 21, 28].

4.3.2. Test Problem 2

Consider the following:

$$f_2(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - x_3)^4 + 10(x_3 - x_4)^4, \tag{4.13}$$

where $x^* = (0,0,0,0)^T$ and $f_2(x^*) = 0$, see [14, 21, 28].

4.3.3. Test Problem 3

Consider the following:

$$f_3(x) = 2x_1^2 + 3x_2^2 + 4x_1x_2 - 6x_1 - 3x_2, \tag{4.14}$$

where

$$x_1^* = (4,-2)^T, \qquad x_2^* = (3,-2)^T, \qquad x_3^* = (2,-1)^T \tag{4.15}$$

and $f_3(x^*) = 0$, see [14, 21, 29].

4.3.4. Test Problem 4

Consider the following:

$$f_4(x) = x^T x, \tag{4.16}$$

where $x^* = (0,0,0,0,0)^T$ and $f_4(x^*) = 0$, see [14, 21, 30].

4.3.5. Test Problem 5

Consider the following:

$$f_5(x) = -(15, 27, 36, 18, 12)x + x^T \begin{pmatrix} 35 & -20 & -10 & 32 & -10 \\ -20 & 40 & -6 & -31 & 32 \\ -10 & -6 & 11 & -6 & -10 \\ 32 & -31 & -6 & 38 & -20 \\ -10 & 32 & -10 & -20 & 31 \end{pmatrix} x, \tag{4.17}$$

where $x^* = (0, 11, 22, 16, 6)^T$ and $x^* = (10, 12, 23, 17, 6)^T$ with $f_5(x^*) = -737$, see [21, 28].

4.3.6. Test Problem 6

Consider the following:

$$f_6(x) = -3803.84 - 138.08x_1 - 232.92x_2 + 123.08x_1^2 + 203.64x_2^2 + 182.25x_1x_2, \tag{4.18}$$

where $x^* = (0, 1)^T$ and $f_6(x^*) = -3833.12$, see [21, 28].

5. Conclusion

This paper presented an EDMHS algorithm for solving continuous optimization problems and integer optimization problems. The proposed EDMHS algorithm applied a newly designed scheme to generate candidate solution so as to benefit from the good information inherent in the best and the second best solution in the historic HM.

Further work is still needed to investigate the effect of EDMHS and adopt this strategy to solve the real optimization problem.

Acknowlegdments

The research is supported by the Grant from National Natural Science Foundation of China no. 11171373 and the Grant from Natural Science Foundation of Zhejiang Province no. LQ12-A01024.

References

[1] 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, 2001.

[2] Z. W. Geem and J. Y. Choi, "Music composition using harmony search algorithm," in *Proceedings of the Applications of Evolutionary Computing*, pp. 593–600, April 2007.

[3] Z. Geem, "Harmony search algorithm for solving sudoku," in *Knowledge-Based Intelligent Information and Engineering Systems*, pp. 371–378, Springer.

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

[5] M. P. Saka, "Optimum geometry design of geodesic domes using harmony search algorithm," *Advances in Structural Engineering*, vol. 10, no. 6, pp. 595–606, 2007.

[6] Z. Geem and J. Williams, "Ecological optimization using harmony search," in *Proceedings of the American Conference on Applied Mathematics*, pp. 24–26, 2008.

[7] M. T. Ayvaz, "Simultaneous determination of aquifer parameters and zone structures with fuzzy c-means clustering and meta-heuristic harmony search algorithm," *Advances in Water Resources*, vol. 30, no. 11, pp. 2326–2338, 2007.

[8] Z. W. Geem, "Harmony search applications in industry," *Soft Computing Applications in Industry*, vol. 226, pp. 117–134, 2008.

[9] Z. Geem, *Music-Inspired Harmony Search Algorithm: Theory and Applications*, vol. 191, Springer, 2009.

[10] G. Ingram and T. Zhang, "Overview of applications and developments in the harmony search algorithm," *Music-Inspired Harmony Search Algorithm*, vol. 191, pp. 15–37, 2009.

[11] M. Mahdavi, M. Fesanghary, and E. Damangir, "An improved harmony search algorithm for solving optimization problems," *Applied Mathematics and Computation*, vol. 188, no. 2, pp. 1567–1579, 2007.

[12] C. M. Wang and Y. F. Huang, "Self-adaptive harmony search algorithm for optimization," *Expert Systems with Applications*, vol. 37, no. 4, pp. 2826–2837, 2010.

[13] M. Fesanghary, M. Mahdavi, M. Minary-Jolandan, and Y. Alizadeh, "Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 197, no. 33-40, pp. 3080–3091, 2008.

[14] M. G. H. Omran and M. Mahdavi, "Global-best harmony search," *Applied Mathematics and Computation*, vol. 198, no. 2, pp. 643–656, 2008.

[15] Z. W. Geem, "Novel derivative of harmony search algorithm for discrete design variables," *Applied Mathematics and Computation*, vol. 199, no. 1, pp. 223–230, 2008.

[16] Q.-K. Pan, P. N. Suganthan, M. F. Tasgetiren, and J. J. Liang, "A self-adaptive global best harmony search algorithm for continuous optimization problems," *Applied Mathematics and Computation*, vol. 216, no. 3, pp. 830–848, 2010.

[17] M. Jaberipour and E. Khorram, "Two improved harmony search algorithms for solving engineering optimization problems," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, pp. 3316–3331, 2010.

[18] P. Yadav, R. Kumar, S. Panda, and C. Chang, "An intelligent tuned harmony search algorithm for optimisation," *Information Sciences*, vol. 196, pp. 47–72, 2012, http://dx.doi.org/10.1016/j.ins.2011.12.035.

[19] Q. Pan, P. Suganthan, J. Liang, and M. Tasgetiren, "A local-best harmony search algorithm with dynamic subpopulations," *Engineering Optimization*, vol. 42, pp. 101–117, 2010.

[20] S. Islam, S. Das, S. Ghosh, S. Roy, and P. Suganthan, "An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 42, no. 2, pp. 482–500, 2012.

[21] E. Laskari, K. Parsopoulos, and M. Vrahatis, "Particle swarm optimization for integer programming," in *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 2, pp. 1582–1587.

[22] H. H. Rosenbrock, "An automatic method for finding the greatest or least value of a function," *The Computer Journal*, vol. 3, pp. 175–184, 1960.

[23] A. A. Goldstein and J. F. Price, "On descent from local minima," *Mathematics of Computation*, vol. 25, pp. 569–574, 1971.

[24] E. D. Eason and R. G. Fenton, "A comparison of numerical optimization methods for engineering design," *Journal of Engineering for Industry*, vol. 96, no. 1, pp. 196–200, 1974.

[25] A. Colville, I. B. M. C. N. Y. S. Center, and I. B. M. C. P .S. Center, *A Comparative Study on Nonlinear Programming Codes*, IBM Corporation, Philadelphia Scientific Center, 1970.

[26] A. Conn, K. Scheinberg, and P. Toint, "On the convergence of derivative-free methods for unconstrained optimization," in *Approximation Theory and Optimization: Tributes to MJD Powell*, pp. 83–108, 1997.

[27] P. Suganthan, N. Hansen, J. Liang et al., "Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization," Tech. Rep. 2005005, Nanyang Technological University, Singapore, 2005.

[28] A. Glankwahmdee, S. Judith, and L. Gary, "Unconstrained discrete nonlinear programming," *Engineering Optimization*, vol. 4, no. 2, pp. 95–107, 1979.

[29] S. S. Rao and S. S. Rao, "Engineering Optimization: Theory and Practice," John Wiley & Sons, Hoboken, NJ, USA, 2009.

[30] G. Rudolph, "An evolutionary algorithm for integer programming," in *Proceedings of the 3rd Conference on Parallel Problem Solving from Nature (PPSN '94)*, pp. 139–148, Jerusalem, Israel, October 1994.