A new improved harmony search algorithm for continuous optimization problems

Yang Lin
International School of SICT
Shandong Institute of Commerce and Technology
Jinan 250103, China
Luckyl8325@163.com

XING Chang-ming
School of further education
Shandong University of Finance and Economics
Jinan 250014, China
xingchm@126.com

Abstract—**Harmony Search (HS) is a new meta-heuristic algorithm, which can be used to solve lots of NP-hard problems. A new improved harmony search algorithm based on the current global information (IGHS) is presented for solving continuous optimization problems. IGHS employs a novel method for generating new solution vectors, which using the current global optimum in the harmony memory. In order to avoid premature and enhance global search ability, IGHS disturbs the current global optimum at a certain probability. The computational simulations and comparisons are carried out by employing 6 benchmark problems from literature. The results show that the proposed IGHS algorithm is more effective in finding better solutions when compared to HS and other related algorithms.**

Keywords-harmony search; Evolutionary algorithms; Global optimization; Continuous optimization

I. INTRODUCTION

Inspired by the behaviors of music improvisation process, a new intelligent optimization algorithm has been recently developed by Geem and named harmony search (HS). In the HS algorithm, the decision variable x_{ij} is analogous to the tonality of the musical instrument, the solution vector $X_i = (x_{i0}, x_{i1}, \ldots\ldots, x_{in})$ is analogous to the harmony in music, and the local and global search schemes are analogous to musician's improvisations. In comparison to other algorithm such as genetic algorithm (GA) and particle swarm optimization (PSO), the HS algorithm is easy to implement, fast convergence and better robustness [2, 3]. Therefore, the HS algorithm has captured much attention and has been successfully applied to solve a wide range of practical optimization problems, such as multi-extremum function optimization[4], design optimization of water distribution networks[5], pipe network design[6] and no-idle flow shop scheduling problems[7].

The HS algorithm is good at identifying the high performance regions of the solution space at a reasonable time, but it is easy to get into trouble in performing local search for numerical optimization applications[2]. Then, In order to improve the fine-tuning characteristic and the convergence rate of HS algorithm, some modified variants were developed. Mahdavi et al. [2] presented an improved HS algorithm, denoted as IHS, by introducing a strategy to dynamically adjust the PAR and bw parameters. Omran and Mahdavi [8] proposed a global best HS algorithm, denoted as GHS, by borrowing the concept from the particle swarm optimization. Pan et al.[4] developed a self-adaptive global best harmony search algorithm, denoted as SGHS, which employs a new improvisation scheme and an adaptive parameter tuning methods. Their experiments showed that these variants could find better solutions when compared to the basic HS algorithm. In this paper, a new improved harmony search algorithm based on the current global information is introduced, denoted by IGHS, which disturbs the current global optimum at a small probability for avoiding the algorithm into local minima. Computational experiments and comparisons show that the IGHS algorithm outperforms the existing HS, IHS and GHS algorithms when applied to 6 standard continuous optimization problems.

II. THE ORIGINAL HARMONY SEARCH ALGORITHM

Harmony search (HS) algorithm was developed in an analogy with music improvisation process where musicians improvise the pitches of their instruments to obtain a perfect harmony. In the harmony search algorithm, each solution vector is called a "harmony" and represented by an n-dimensional real-valued vector. HMS harmony vectors, denoted by harmony memory (HM), are randomly generated. Then a new candidate harmony X_{new} is generated by an improvisation scheme. Finally, the worst harmony vector X_w in the current HM is replaced by X_{new} if the fitness value of X_{new} is better than the fitness value of X_w. The above process is repeated until a termination criterion is met. The specific process of HS algorithm is briefly described below.

Step1. Initialize the problem and algorithm parameters.

Given the optimization problem: $\min f(X),\ s.t.\ x_i \in [L_i, U_i], i = 1, 2, \ldots, n$, where $f(X)$ is a continuous function; X is n-dimensional real vector, and L_i and U_i are the lower and upper bounds of the variable x_i, respectively.

The parameters of the HS algorithm is initialized also, which including the harmony memory size (HMS), harmony memory consideration rate (HMCR), pitch adjusting rate (PAR), distance bandwidth (BW), the total number of function evaluations(N) or stopping criterion, and the bounds of the decision variable ($[L_i, U_i]$).

Step2. Initialize the harmony memory.
The $\mathrm{HM} = [\mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_\mathrm{m}]$ is filled with m randomly

generated solution vectors, where m=HMS, $X_i = (x_{i0}, x_{i1}, \ldots\ldots, x_{in})$, $i = 1,2,\ldots,m$,and $m = HMS$. The x_{ij} is the jth harmony vector in the X_i, which is generated as follows.

$x_{ij} = L_j + (U_j - L_j) * r$, where r is a uniform random number between 0 and 1.

Step3. Improvise a new harmony.

A new harmony vector, $X_{new} = (x_{new0}, x_{new1}, \cdots, x_{newn})$, is generated based on three rules: a memory consideration, a pitch adjustment and a random selection. The specific operations are given as follows.

```
for(j=1 to n ) do
   if($r_1$<HMCR) then
      $x_{new\,j}$=$x_{rndj}$ where rnd$\in$(1,2,...,HMS)
      if($r_2$<PAR)then
         $x_{new\,j}$=$x_{new\,j}$$\pm r_3\times$BW where $r_1,r_2,r_3\in(0,1)$
      endif
   else
      $x_{new\,j}$=$L_j+r\times(U_j-L_j)$,where r$\in$(0,1)
   endif
endfor
```

Step4. Update harmony memory.

If the new harmony vector X_{new} is better than the worst harmony X_w in the current HM, the HM will be updated. That is, if $f(X_{new}) < f(X_w)$, X_w will be replaced by X_{new} and excluded from the HM.

Step5. Check stopping criterion

If the stopping criterion (i.e. maximum number of the function evaluations) is satisfied, return the best harmony vector X_b in the current HM, otherwise go back to step 3.

III. IMPROVED HARMONY SEARCH

In order to improve the performance of HS, some variant of HS were developed. A few major improvements are described as follows.

A. IHS algorithm

The IHS algorithm applies the same improvisation scheme as the basic HS algorithm. The key difference between IHS and HS is that IHS uses variables PAR and bw in Step 3. The parameter PAR and BW change dynamically with generation number according to Eq. (1) and Eq.(2).

$$PAR_k = PAR_{\min} + \frac{(PAR_{\max} - PAR_{\min})}{NI} \times k \qquad (1)$$

$$BW_k = BW_{\max} e^{\left(\frac{\ln(\frac{BW\min}{BW\max})}{NI} \times t\right)} \qquad (2)$$

Where PAR_k and BW_k is the pitch adjustment rate and the bandwidth for each generation k respectively.

B. GHS algorithm

Compared with the HS algorithm, the GHS algorithm used a different improvisation scheme, which borrowed the concepts from swarm intelligence. In the GHS algorithm, the best harmony vector $X_b = (x_{b0}, x_{b1}, \ldots, x_{bn})$ in the current HM is used for generating a new harmony vector X_{new}. The jth harmony vector $X_{new\,j}$ in the X_{new} is generated as follows.

$x_{new\,j} = x_{bk}$, where k is a random integer between 1 and n.

In addition, the GHS algorithm employs the dynamic adjusting function for the PAR parameter as in Eq. (1), and eliminates the difficulties in selecting the parameter BW. The specific GHS algorithm could be found in the ref.[8].

Due to limited space, other variants of the HS algorithm can not be introduced one bye one.

IV. THE PROPOSED IGHS ALGORITHM

Inspired by the ref.[4] and ref.[8], a new variant of the HS algorithm, denoted by IGHS, is developed in this section. The IGHS algorithm modified the improvisation schema based on the GHS algorithm. The details of the IGHS algorithm are described as follows.

In the GHS algorithm, the new vector X_{new} is filled with the any dimension harmony vector in the best harmony vector X_b randomly, which may generate infeasible solutions with high probability. To avoid this phenomenon, the pitch adjustment rule $x_{new\,j} = x_{Bk}, j = 1,2,\cdots,n$ is modified to $x_{new\,j} = x_{Bj}, j = 1,2,\cdots,n$, That is the decision variable in X_{new} is assigned by the corresponding decision variable in X_b. Unlike the SGHS, in order to ensure convergence speed and prevent falling into local minima, the best harmony vector is disturbed with small probability $HMCR * PAR$. The parameter BW is retained when disturbing the best harmony vector. The HS algorithm has an important feature that the new harmony vector is generated by the cooperation of all individuals. For keeping this feature to the IGHS algorithm, the new harmony vector is assigned by the corresponding decision variable in the current HM with probability $HMCR * (1 - PAR)$.

In addition, the parameters PAR and BW could employ the fixed values as the HS algorithm or employ the dynamic adjusting function as in Eq. (1) and Eq.(2), respectively. Below, the IGHS with fixed PAR and BW is denoted as IGHS1, and the IGHS with dynamic PAR and BW is denoted as IGHS2. The improvise schema of the IGHS algorithm is described as follows.

```
for(j=1 to n ) do
    if($r_1$<HMCR) then
        if($r_2$<PAR)then
            $x_{newj}=x_{bj} \pm r_3*BW$    $r_1,r_2,r_3 \in (0,1)$
        else
            $x_{newj}=x_{rndj}$ where rnd$\in$(1,2,…,HMS)
        endif
    else
        $x_{newj}=L_j+r\times(U_j-L_j)$,where r$\in$(0,1)
    endif
endfor
```

V. Experimental results

To test the performance of the different HS algorithm, using Java Programming, an extensive experiment is provided based on computer with Pentium dual-core 2.8 HGz CPU and 2.0G Memory. The six numerical optimization problems used in our empirical studies is shown in table 1. Where the second column represents the range of the variable x_i.

Table 1 Test functions

Function	x_i
$f_1=\sum_{i=1}^{n} x_i^2$	[-100,100]
$f_2=\sum_{i=1}^{n}(\sum_{j=1}^{i} x_j)^2$	[-65,65]
$f_3=\sum_{i=1}^{n-1}(100\times(x_{i+1}-x_i^2)^2+(x_i-1)^2)$	[-2.048,2.048]
$f_4=\sum_{i=1}^{n}[x_i^2-10\times\cos(2\pi\times x_i)+10]$	[-5.12,5.12]
$f_5=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	[-600,600]
$f_6=-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2})-\exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i)+20+e$	[-32,32]

In order to test the robustness of these algorithms for different problems, the above functions were implemented in 30 and 100 dimensions respectively. And the default parameters of all algorithms are listed in Table 2.

Table 2 Parameter settings for these algorithms.

methods	HMS	HMCR	PAR	PAR_{min}	PAR_{max}	BW	BW_{min}	BW_{max}	NI
HS	5	0.9	0.3	-	-	0.01	-	-	50000
IHS	5	0.9	-	0.01	0.99	-	0.0001	$(U_i-L_i)/20$	50000
GHS	5	0.9	-	0.01	0.99	-	-	-	50000
IGHS1	5	0.95	0.3	-	-	0.01	-	-	50000
IGHS2	5	0.95		0.01	0.99	-	0.00001	$(U_i-L_i)/20$	50000

The mean best result (MB) and standard deviations (SD) over 30 replications for 30-dimensional and 100-dimensional test functions is recorded in tables 3 and 4, respectively. The statistically significant best solutions have been shown in bold.

Table 3 Comparing the algorithm results (n=30)

	HS	IHS	GHS	IGHS1	IGHS2
f_1	0.000187	0.000712	0.000010	0.000103	**0.000000**
	0.000032	0.000644	0.000022	0.000001	**0.000000**
f_2	4297.816457	4343.653320	5146.176259	6396.779	**0.000002**
	1362.148438	1062.106222	6348.792556	1926.06	**0.000001**
f_3	340.297100	624.323216	49.669203	**36.509484**	39.778402
	266.691353	559.847363	59.161192	**26.822640**	30.100683
f_4	1.390625	3.499144	**0.008629**	0.030016	0.044148
	0.824244	1.182907	**0.015277**	0.099191	0.242561
f_5	1.119266	1.120992	0.102407	0.437046	**0.007983**
	0.041207	0.040887	0.175640	0.199025	**0.008317**
f_6	1.130004	1.893394	0.020909	0.011151	**0.000296**
	0.407044	0.314610	0.021686	0.036951	**0.000028**

Table 4 Comparing the algorithm results (n=100)

	HS	IHS	GHS	IGHS1	IGHS2
f_1	1.4496E+04[9]	1.5373E+04[9]	1.6361E+04[9]	2268.54	**2189.87**
	1.2675E+03	1.2652E+03	1.0244E+03	0.3235E+03	**0.3163E+03**
f_2	215052.904398	213812.584732	321780.353575	**113912.3838**	115549.2522
	28276.375538	28305.249583	39589.041160	**13060.97082**	14942.30952
f_3	16675172.184717	17277654.059718	2598652.617273	**1119.516423**	773.513165
	3182464.488466	2945544.275052	915937.797217	**707.137520**	824.252690
f_4	343.497796	343.232044	80.657677	**117.060471**	121.1126
	27.245380	25.149464	30.368471	**11.387451**	10.832564
f_5	195.592577	204.291518	54.252289	20.641514	**20.535526**
	24.808359	19.157177	18.600195	3.049441	**2.907198**
f_6	13.857189	13.801383	8.767846	**6.730084**	6.803974

0.284945	0.530388	0.880066	**0.267981**	0.208818

As shown in the table 3 and table4, besides to the 30-dimension function f_4, the convergence accuracy of IGHS algorithm is better than other algorithms. Especially in the classic complex problem, the multi-dimensional Rosenbrock optimization f_3, IGHS algorithm improve about 4 orders of magnitude than other algorithms. In addition, comparing the HS and IHS, IGHS1 and IGHS2, we can see that the IHS algorithm is not always better than the HS algorithm, and the IGHS is not always better than the IGHS1 too. However, optimizing the structure of the HS algorithm, changing the generation way of the new harmony vector could improve the performance of the HS algorithm. Due to space limitations, the convergence speed comparison of the algorithm will be discussed in other separate literature.

VI. CONCLUSIONS

This paper presented a new variant of the HS algorithm. The new algorithm based on the current global information in the current HM, and is named IGHS. Based on the GHS, a novel designed scheme to generate a candidate solution is applied for preventing the generation of infeasible solutions with high probability. If the best harmony vector is assigned to the new solution, the new solution will be disturbed with small probability $HMCR*PAR$, which could avoid the prematurely and enhance the global search ability. Numerical experiments based on benchmark problems showed that the proposed IGHS algorithm is simple, easy to implement, and more effective in finding better solutions than other related algorithms.

ACKNOWLEDGMENT

Authors acknowledge the financial support offered by the research project of SICT under the grant NO. 2011xjkj04. This research is also partially supported by the Shandong outstanding young scientist research award fund under Grant NO. 2010BSE14022 and the research project of Shandong University of finance.

REFERENCES

[1] Z.W. Geem, J.H. Kim, G.V. Loganathan, A new heuristic optimization algorithm: harmony search [J], Simulations. 2001, 76: 60–68.

[2] M. Mahdavi, M. Fesanghary, E. Damangir, An improved harmony search algorithm for solving optimization problems [J], Appl. Math. Comput. 2007,188 :1567–1579.

[3] K.S. Lee, Z.W. Geem, A new meta-heuristic algorithm for continuous engineering optimization, harmony search theory and practice [J], Comput. Methods Appl. Mech. Eng. 2005,194 :3902-3933.

[4] Pan Quan-ke, Suganthan P. N., Tasgetiren MF. A Self-Adaptive Global Best Harmony Search Algorithm for Continuous Optimization Problems [J]. Applied mathematics and computation, 2010, 206:830-848

[5] GEEM Z. Optimal cost design of water distribution networks using harmony search [J]. Engineering Optimization,2006,38(3):259-280.

[6] GEEM Z, KIM J, LOGANATHAN G. Harmony search optimization application to pipe network design[J]. International Journal of Model Simulation,2002,22(2):125-133.

[7] Wu lei, Pan Quan-ke, Sang Hong-yan, Pan Yu-xia, Harmony search algorithms for no-idle flow shop scheduling problems[J], Computer Integrated Manufacturing Systems(in Chinese), 2009,15(10):1960-1967.

[8] M.G.H. Omran, M. Mahdavi, Global-best harmony search[J], Applied mathematics and computation. 198 (2008) 643–656.

[9] Chia-Ming Wang, Yin-Fu Huang. Self-adaptive harmony search algorithm for optimization[J], Expert System with Applications, 2010,37(4):2826-2837.