Modified harmony search

Modified harmony search

Najihah Mohamed[1], Ahmad Lutfi Amri Ramli[2], Ahmad Abd Majid[3] and Abd Rahni Mt Piah[4]

[1] Faculty of Industrial Sciences & Technology, Universiti Malaysia Pahang, Malaysia
[1,2,3] School of Mathematical Sciences, Universiti Sains Malaysia, Malaysia
[4] DRB-HICOM University of Automotive Malaysia, Malaysia

E-mail: najihah@ump.edu.my

Abstract. A metaheuristic algorithm, called Harmony Search is quite highly applied in optimizing parameters in many areas. HS is a derivative-free real parameter optimization algorithm, and draws an inspiration from the musical improvisation process of searching for a perfect state of harmony. Propose in this paper Modified Harmony Search for solving optimization problems, which employs a concept from genetic algorithm method and particle swarm optimization for generating new solution vectors that enhances the performance of HS algorithm. The performances of MHS and HS are investigated on ten benchmark optimization problems in order to make a comparison to reflect the efficiency of the MHS in terms of final accuracy, convergence speed and robustness.

1. Introduction

Nowadays, researchers all over the world are attracted toward the nature-inspired metaheuristics in order to meet demands solution of the complex and real-world problems since it can reduce the computational cost dramatically. It is not about reducing the cost only, but sometimes the traditional methods are, in some cases impossible to apply. Among all the kind, we named a few of quite well-known methods such as genetic algorithm (GA), particle swarm optimization (PSO), artificial bee colony algorithm (ABC) and harmony search (HS). GA which was invented and developed by John Holland in the 1960s at the University of Michigan [1] presented the GA in [2]. Other well-known methods such as PSO invented by [3] and [4] invented ABC.

A music-inspired metaheuristics method, HS [5] have been successfully applied to wide range of optimization problems, for example, wireless sensor network [6], load frequency control [7], power system stabilizers [8], vehicle routing problem [9], knapsack problem [10] and scheduling algorithm [11]. Meanwhile, a few researchers take initiative to improved HS in order to improve optimization methods in their areas such as [12] and [13] which hybridizing HS and PSO, [14] hybridizing genetic programming and HS, [15] proposed global-best harmony while, [16] make a modification to the HS by adapting new parameter.

This paper proposes a useful modification to the classical harmony search which use a concept of GA and PSO as one of the methods in searching the best parameters in order to optimize the objective function. These two employments contribute a significant vectors to enhance the performance of the proposed method. The new version is called modified harmony search (MHS). The results of the experiments conducted are shown and compared with the versions of HS proposed by [5] and global-best harmony search (GHS) by [15].

The rest of the paper is organized in the following way. Section 2 provides an overview of HS algorithm. Global-best HS is summarized in section 3. The proposed approach is presented in section

4. Section 5 presents and discusses the experimental results of the comparative study and section 6 concludes the paper.

2. Harmony Search

Harmony Search (HS) is a metaheuristic algorithm which was originally inspired by the improvisation process of Jazz musicians which mimicking the improvisation of music players [5]. The details process of these steps is illustrated in figure 1.

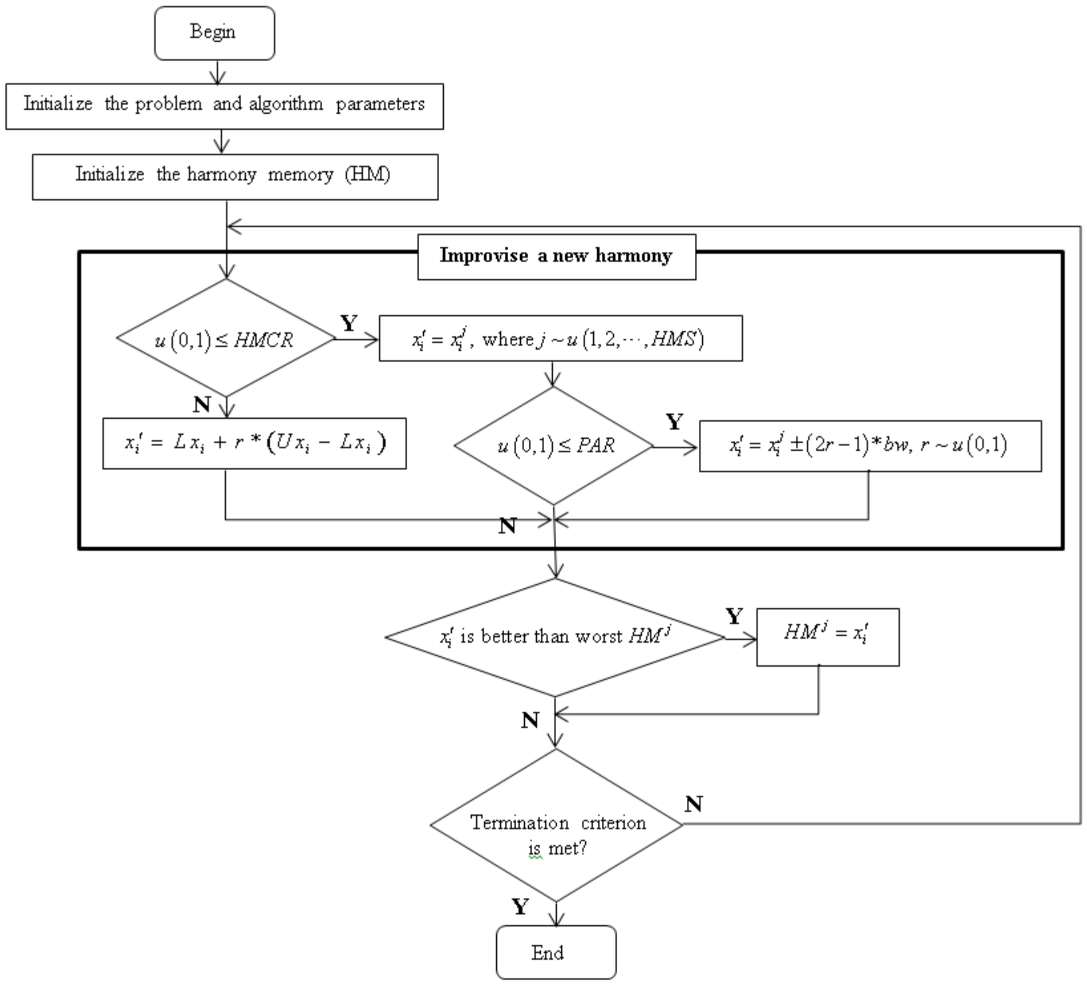
Begin
Initialize the problem and algorithm parameters
Initialize the harmony memory (HM)
Improvise a new harmony
$u(0,1) \le HMCR$
Y
$x_i' = x_i^j$, where $j \sim u(1,2,\cdots,HMS)$
N
$x_i' = Lx_i + r * (Ux_i - Lx_i)$
$u(0,1) \le PAR$
Y
$x_i' = x_i^j \pm (2r-1)*bw,\ r \sim u(0,1)$
N
x_i' is better than worst HM^j
Y
$HM^j = x_i'$
N
Termination criterion is met?
N
Y
End

Figure 1. Procedure of harmony search algorithm [17].

The most important part is step 3, which is *improvise a new harmony*. This is because all the reforms will be put under this step, a mechanism for creating a better new harmony. The details of this step is :

Step 3 : Improvise a new harmony

```
for $j = 1$ to $N$,    if $U(0,1) \le$ HMCR then /* memory consideration*/
                       begin
                         $x_i' = x_j'$, where $j \sim U(1,\cdots,HMS)$.
                         if $U(0,1) \le$ PAR then /*pitch adjustment*/
                           begin
                             $x_i' = x_i' \pm r \times bw$, where $r \sim U(0,1)$ and $bw$ is an arbitrary distance bandwidth.
                           endif
                         else /*random selection*/
                         $x_i' = \mathrm{LB}_i + r \times (\mathrm{UB}_i - \mathrm{LB}_i)$
                       endif
done
```

3. Global-best Harmony Search

[15] proposed a new variant of the HS called, global-best harmony search (GHS), modifies the pitch-adjustment step which new harmony can mimic the best harmony in the HM. Thus, replacing the bw parameter gether and adding a social dimension to the HS.

Step 3 : Improvise a new harmony

```
for $j = 1$ to $N$,    if $U(0,1) \le$ HMCR then /* memory consideration*/
                       begin
                         $x_i' = x_j'$, where $j \sim U(1,\cdots,HMS)$.
                         if $U(0,1) \le \mathrm{PAR}(t)$ then /*pitch adjustment*/
                           begin
                             $x_i' = x_k^{best}$, where best is the best index in the HM and $k \sim U(1,N)$.
                           endif
                         else /*random selection*/
                         $x_i' = \mathrm{LB}_i + r \times (\mathrm{UB}_i - \mathrm{LB}_i)$
                       endif
done
```

4. Modified Harmony Search

New technique called modified harmony search (MHS) employs a novel method for generating new solution vectors that enhances accuracy and convergence rate of HS algorithm. Inspired by the concept from PSO, the position of a particle is influenced by the best position visited itself and the position of the best particle in swarm. MHS added one more step in memory consideration phase of the HS such that the new harmony can mimic the best harmony in the HM. Besides, a concept from GA which is a crossover was being borrowed, where MHS uses a modification of the selected harmony in order to generate a new harmony.

Step 3 : Improvise a new harmony

```
for $j = 1$ to $N$
    if $U(0,1) \le \mathrm{HMCR}$
        $D_{1j} = \mathrm{HM}(rand(\mathrm{HMS}), j)$,    $D_{2j} = \mathrm{HM}(best, j)$ /*best position*/
        if $U(0,1) \le \mathrm{PAR}$ ,  $D_{lj} = D_{lj} \pm \mathrm{BW}(2 \times r - 1), l = 1, 2$ ,    endif
    else
        $D_{lj} = \mathrm{LB}_j + r(\mathrm{UB}_j - \mathrm{LB}_j)$,    $l = 1, 2$
    endif
done
$\alpha = U(0,1)$,    $D_3 = \alpha D_1 + (1-\alpha) D_2$,    $D_4 = \alpha D_2 + (1-\alpha) D_1$ /*cross-over*/
```

4.1 . Example

This section will discussed details about algorithm behavior in consecutive generations of HS, GHS and MHS by using the *Griewank* function which defined in section 5. The number of decision variables is set to 3 with possible values bounds between -600 and 600. Other parameters were set as in next section. The state of HM in different iterations for the algorithms MHS and HS are shown in tables 1 – 3 respectively. Each simulation targeted to have 500 iterations only and all of the algorithms improvised a near optimal, MHS is better than HS and GHS.

Table 1. HM state in different iterations for the *Griewank* function using the HS algorithm. [15].

Rank		x_1	x_2	x_3	$f(x)$
Initial HM	1	206.909180	241.845703	-102.795410	29.141686
	2	-99.938965	332.336426	-397.961426	70.088302
	3	-381.262207	-392.358398	-87.634277	77.971790
	4	-381.262207	-472.924805	-87.634277	95.243494
	5	-470.910645	-54.345703	-430.700684	104.179736
HM after 10 iterations	1	206.909180	-54.332356	-87.634277	13.718178
	2	206.909180	-54.332356	-102.795321	15.721512
	3	206.909180	241.845703	-102.795410	29.141686
	4	206.909180	316.369629	-87.634277	39.325780
	5	-438.061523	-54.345703	-102.795436	52.221392
HM after 100 iterations	1	-15.234375	-54.332356	56.637901	2.787318
	2	109.016680	-54.332356	56.643102	5.635628
	3	109.020996	-54.338577	56.643102	5.635628
	4	109.020996	-54.332356	56.643102	5.635628
	5	109.020996	-54.341623	56.637901	5.635628
HM after 500 iterations	1	-15.296501	-18.134526	-27.026367	0.467774
	2	-15.296501	-18.134526	50.484959	1.206920
	3	-15.296501	-18.134526	50.484959	1.207263
	4	-15.296501	-18.134526	50.484959	1.208392
	5	-15.296501	-18.134526	50.484959	1.208400

Table 2. HM state in different iterations for the *Griewank* function using the GHS algorithm [15].

Rank		x_1	x_2	x_3	$f(x)$
Initial HM	1	206.909180	241.845703	-102.795410	29.141686
	2	-99.938965	332.336426	-397.961426	70.088302
	3	-381.262207	-392.358398	-87.634277	77.971790
	4	-381.262207	-472.924805	-87.634277	95.243494
	5	-470.910645	-54.345703	-430.700684	104.179736
HM after 10 iterations	1	206.909180	-54.345703	-87.634277	13.721652
	2	206.909180	-54.345703	-87.634277	13.721652
	3	206.909180	136.926270	-102.795410	18.311658
	4	206.909180	136.926270	-87.634277	19.032912
	5	206.909180	136.926270	-87.634277	19.032912
HM after 100 iterations	1	-18.859863	24.719238	-45.629883	1.692257
	2	-18.859863	45.593262	-45.629883	1.890251
	3	-18.859863	45.593262	-45.629883	1.890251
	4	-18.859863	45.593262	-45.629883	1.890251
	5	-18.859863	45.593262	-45.629883	1.890251
HM after 500 iterations	1	-18.859863	1.171875	22.082520	0.546616
	2	-18.859863	1.171875	22.082520	0.546616
	3	-18.859863	1.171875	22.082520	0.546616
	4	-18.859863	1.171875	22.082520	0.546616
	5	-18.859863	1.171875	22.082520	0.546616

Table 3. HM state in different iterations for the *Griewank* function using the MHS algorithm.

Rank		x_1	x_2	x_3	$f(x)$
Initial HM	1	-443.204366	208.270871	-60.580594	62.691691
	2	-119.486776	-152.607740	-332.219628	38.432213
	3	189.459541	-174.790596	-442.564732	66.441219
	4	-500.641212	-341.119459	421.814952	137.212745
	5	-144.995047	-294.434745	431.536697	74.818646
HM after 10 iterations	1	-119.489181	-128.606243	-332.218452	35.330456
	2	-119.489453	-128.606243	-332.218908	35.330620
	3	-119.488961	-128.606243	-332.218081	35.330323
	4	-119.489232	-128.606243	-332.218538	35.330487
	5	-119.489453	-128.606243	-332.218908	35.330620
HM after 100 iterations	1	-0.208369	-33.852081	-55.948645	1.842361
	2	-0.208369	-33.852081	-55.950488	1.842708
	3	-0.208369	-33.852081	-55.948645	1.842361
	4	-0.208369	-33.852081	-55.949556	1.842533
	5	-0.208369	-33.852081	-55.949623	1.842545
HM after 500 iterations	1	-0.006091	8.876288	-21.263782	0.174450
	2	-0.006864	8.876288	-21.263782	0.174455
	3	-0.008878	8.876288	-21.263782	0.174470
	4	-0.008878	8.876288	-21.263782	0.174470
	5	-0.008878	8.876288	-21.263782	0.174470

5. Experimental results

In this section, the following functions have been used to compare the performance of the methods. Generally all algorithms are set HMS = 5 and HMCR = 0.9. But for HS and MHS PAR = 0.3 and bw = 0.01, while GHS sets PAR_{min} = 0.01 and PAR_{max} = 0.99 unless another parameter in MHS, which is α will be a random number between 0 and 1, α will be used to do the cross-over process. For each of these functions, the goal is to find the global minimizer.

Table 4. Benchmark Functions**.**

Function Name	**Expression**	**Search Range**	**Optimum Value**
Sphere Function	$f_1(\vec{x}) = \sum_{i=1}^{N} x_i^2$	$-100 \le x_i \le 100$	$\min(f_1) = f_1(0,\cdots,0) = 0$
Schwefel's Problem	$f_2(\vec{x}) = \sum_{i=1}^{N} \lvert x_i \rvert + \prod_{i=1}^{N} \lvert x_i \rvert$	$-10 \le x_i \le 10$	$\min(f_2) = f_2(0,\cdots,0) = 0$
Step Function	$f_3(\vec{x}) = \sum_{i=1}^{N} (\lvert x_i + 0.5 \rvert)^2$	$-100 \le x_i \le 100$	$\min(f_3)$ $f_3(-0.5,\cdots,-0.5) = 0$
Rosenbrock's Function	$f_4(\vec{x}) = \sum_{i=1}^{N-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2 \right]$	$-30 \le x_i \le 30$	$\min(f_4) = f_4(1,\cdots,1) = 0$
Rotated Hyper-Ellipsoid Function	$f_5(\vec{x}) = \sum_{i=1}^{N} \left(\sum_{j=1}^{i} x_j \right)^2$	$-100 \le x_i \le 100$	$\min(f_5) = f_5(0,\cdots,0) = 0$
Generalized Schwefel's Problem	$f_6(\vec{x}) = -\sum_{i=1}^{N} \left[x_i \sin\left(\sqrt{\lvert x_i \rvert}\right) \right]$	$-500 \le x_i \le 500$	$\min(f_6)$ $f_6(420.9687,\cdots,420.9687)$ $= -N \times 418.9829$
Rastrigin Function	$f_7(\vec{x}) = \sum_{i=1}^{N} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right]$	$-5.12 \le x_i \le 5.12$	$\min(f_7) = f_7(0,\cdots,0) = 0$
Ackley's Function	$f_8(\vec{x}) = -20e^{\left(-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}\right)} - e^{\left(\frac{1}{N}\sum_{i=1}^{N}\cos(2\pi x_i)\right)} + 20 + e$	$-32 \le x_i \le 32$	$\min(f_8) = f_8(0,\cdots,0) = 0$
Griewank Function	$f_9(\vec{x}) = \frac{1}{4000}\sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$-600 \le x_i \le 600$	$\min(f_9) = f_9(0,\cdots,0) = 0$
Six-Hump Camel-Back Function	$f_{10}(x_1, x_2) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	$-5 \le x_i \le 5$	$\min(f_{10}) = f_{10}(-0.08983, 0.7126)$ $= -1.0316285$

The results reported in this section are means and standard deviations over 30 simulations. Each simulation runs for 50,000 evaluations of the objective function. Table 5 summarizes the results obtained by these three methods on the benchmark functions. The results show that the MHS outperformed HS and GHS in all functions except for *Rosenbrock* function.

Table 5. Means and standard deviation of the benchmark function optimization results.

	HS	GHS	MHS
Sphere Function	0.000187(0.000032)	0.000010(0.000022)	**0.000000(0.000000)**
Schwefel's Problem	0.171524(0.072851)	0.072815(0.114464)	**0.000002(0.000000)**
Step Function	340.297100(266.691353)	49.669203(59.161192)	**0.000000(0.000000)**
Rosenbrock's Function	4.233333(3.029668)	**0(0)**	2.18492(0.63435)
Rotated Hyper-Ellipsoid F.	4297.816457(1362.148438)	5146.176259(6348.792556)	**0.000000(0.000000)**
Generalized Swefel's Problem	-12539.237786(11.960017)	-12569.458343(0.050361)	**-1256.948662 (0.000000)**
Rastrigin Function	1.390625(0.824244)	0.008629(0.015277)	**0.000000(0.000000)**
Ackley's Function	1.130004(0.407044)	0.020909(0.021686)	**0.000001(0.000000)**
Griewank Function	1.119266(0.041207)	0.102407(0.175640)	**0.00674(0.00082)**
Six-Hump Camel-Back F.	-1.031628(0.00000)	-1.031600(0.00018)	**-1.03163(0.000000)**

6. Conclusion

This paper has presented a new version of harmony search which called modified harmony search. The approach adopted a concept from particle swarm optimization, which allows to keep the best position visited itself and a phase in genetic algorithm that calls cross-over in order to generate a new harmony beside generating harmony as in classic HS. This approach was tested on ten benchmark functions where it shows better performances compared to other approaches.

References

[1] Mitchell Melanie 1996 *Cambridge, MA: MIT Press*
[2] Holland J H 1975 *University of Michigan Press. (Second edition: MIT Press, 1992)*
[3] Kennedy J and Eberhart R 1995 *Proceedings of IEEE International Conference on Neural Networks***IV** 1942-194
[4] Karaboga D 2005 *Technical Report-Tr06,Erciyes University, Engineering Faculty, Computer Engineering Department*
[5] Geem Z W, Kim J H and Loganathan G V 2001 *Simulation* **76.2** 60-68
[6] Alia O M 2017 *Information Sciences* **385-386** 76-95
[7] Khooban M H, Niknam T, Blaabjerg F and Dragicevic T 2017 *Electric Power System Researchs* **143** 585-598
[8] Naresh G, Ramalinga R M and Narasimham S V L 2016 *Swarm and Evolutionary Computation* **27** 169-179
[9] Yassen E T, Ayob M, Nazri M Z A and Sabar N R 2015 *Information Sciences* **325** 140-158
[10] Moosavian N 2015 *Swarm and Evolutionary Computation* **20** 14-22
[11] Dong B, Jiao L and Wu J 2015 *Knowledge-Based Systems* **88** 244-252
[12] Shankar T, Shanmugavel S and Rajesh A 2016 *Swarm and Evolutionary Computation* **30** 1-10
[13] Ouyang H-B, Gao L-Q, Kong X-Y, Li S and Zou D-X 2016 *Information Sciences* **346-347** 318-337
[14] Elola A, Ser J D, Bilbao M N, Perfecto C, Alexandre E and Salcedo-Sanz S 2017 *Applied Soft Computing* **52** 760-770
[15] Omran M G H and Mahdavi M 2008 *Applied Mathematics and Computation* **198** 643-656
[16] Das S, Mukhopadhyay A, Roy A, Abraham A and Panigrahi B K 2011 *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions*
[17] Mohamed N, Majid A A and Piah A R M 2015 *Egyptian Informatics Journal* **16** 175-185