Incorporating Great Deluge with Harmony Search for Global Optimization Problems

Mohammed Azmi Al-Betar, Osama Nasif Ahmad, Ahamad Tajudin Khader, and Mohammed A. Awadallah

Abstract Harmony search (HS) algorithm is relatively a recent metaheuristic optimization method inspired by natural phenomenon of musical improvisation process. despite its success, the main drawback of harmony search are contained in its tendency to converge prematurely due to its greedy selection method. This probably leads the harmony search algorithm to get stuck in local optima and unsought solutions owing to the limited exploration of the search space. The great deluge algorithm is a local search-based approach that has an efficient capability of increasing diversity and avoiding the local optima. This capability comes from its flexible method of accepting the new constructed solution. The aim of this research is to propose and evaluate a new variant of HS. To do so, the acceptance method of the great deluge algorithm is incorporated in the harmony search to enhance its convergence properties by maintaining a higher rate of diversification at the initial stage of the search process. The proposed method is called Harmony Search Great Deluge (HS-GD) algorithm. The performance of HS-GD and the classical harmony search algorithm was evaluated using a set of ten benchmark global optimization functions. In addition, five benchmark functions of the former set were employed to compare the results of the proposed method with three previous harmony search variations including the classical harmony search. The results show that HS-GD often outperforms the other comparative approaches.

Keywords: Harmony Search, Great Deluge, Global Optimization, Diversification, Intensification

---

Mohammed Azmi Al-Betar
Department of Computer Science, Jadara University, PO Box 733, Irbid, Jordan
e-mail: mohbetar@cs.usm.my

Mohammed Azmi Al-Betar · Osama Nasif Ahmad · Ahamad Tajudin Khader · Mohammed A. Awadallah
School of Computer Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia e-mail: {mohbetar, osnasif, tajudin, mohawad}@cs.usm.my

J. C. Bansal et al. (eds.), *Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012)*, Advances in Intelligent Systems and Computing 201, DOI: 10.1007/978-81-322-1038-2_24,

1 Introduction

Evolutionary algorithms (EAs) are a class of optimization methods which normally start with a population of random solutions. These solutions are evolutionary evolved using effective operators that drive the search either randomly or structurally based on the objective function of the current population. These operators explore and exploit the problem search space to come up with an optimal solution. However, this class of algorithms easily gets trapped into a chronic premature convergence problem due to identical population in the last stage of search [10].

Harmony Search (HS) algorithm [14] is a recent EA that imitates the behavior of a group of musicians when improvising a musical harmony. It has several impressive characteristics related to its simplicity, flexibility, adaptability, generality, and scalability [4]. As such, HS algorithm has been successfully adapted to a plethora of optimization problems such as Structural Design, Clustering, Bioinformatics, nurse restoring and timetabling [1, 6, 3, 2, 5, 9, 8, 7] and many others.It has been the subject of various researches that have been conducted to further improve its performance [16, 6]. The Theory of the HS has also undergone improvement as shown in [4].

HS is initiated with a population of random solutions stored in Harmony Memory (HM). Iteratively, it generates a new solution using three operators: i) Memory Consideration, which exploits the current population, ii) pitch adjustment, which locally refines some solutions in the current population, and iii) random consideration, which explores new solutions. The new solution is then evaluated to replace the worst solution in HM, if better. This process is repeated until a termination rule is reached.

As aforementioned, HS greedily accepts the new solution to enter the population, if and only if, it is better than the worst solution in HM. This acceptance criteria is similar to Hill climbing optimizer. The main weakness of Hill climbing is related to its simplicity to get stuck in local optima due to the lack of exploration capability. Therefore, several variations of Hill climbing that inject an explorative strategy with hill climbing were proposed. Examples include Simulated Annealing and Great Deluge [12]. The acceptance criteria of SA and GD algorithms substitutes the current solution with the new solution, if better or if it is accepted by certain threshold though it is worst [12]. This acceptance criteria empowers an explorative capability of SA and GD and eventually avoids the trap of local optima.

According to Geem et al., [15], the majority of researches and studies have been conducted in order to enhance the solution accuracy and speed up the convergence rate of HS. The resulting variations have advantages in terms of the implementation time but they still suffer from deficiencies in terms of avoiding the premature convergence. In this paper, Harmony Search- Great Deluge (HS-GD) algorithm is proposed. In HS-GD algorithm, the acceptance method of GD and its main related concepts are incorporated with the greedy acceptance method of classical HS algorithm to empower its explorative properties and eventually avoid a chronic premature convergence problem. Using standard mathematical functions, the experimental results show that the proposed HS-GD algorithm has improve the performance of the classical HS.

2 Harmony search Great Deluge (HS-GD) algorithm

Harmony Search (HS) is an evolutionary algorithm (EA) inspired by the musical improvisation process [14], where a group of musicians improvise the pitches of their musical instruments, *practice after practice*, seeking for a pleasing harmony as determined by an audio-aesthetic standard. Analogously in optimization, a set of decision variables is assigned with values, *iteration by iteration*, seeking for a '*good enough*' solution as evaluated by an objective function.

The great deluge(GD) algorithm has a special acceptance method. It accepts the new generated solution if its quality is better than or equal to a predefined linearly boundary (called "level" or "ceiling") increasing (in maximization) or decreasing (in minimization) that increases or decreases according to a fixed rate. This method is incorporated in HS. The flowchart of the HS-GD algorithm is shown in Figure 1 where the acceptance method of GD is highlighted by the red diamond. The HS-GD algorithm has five main steps illustrated as follows:

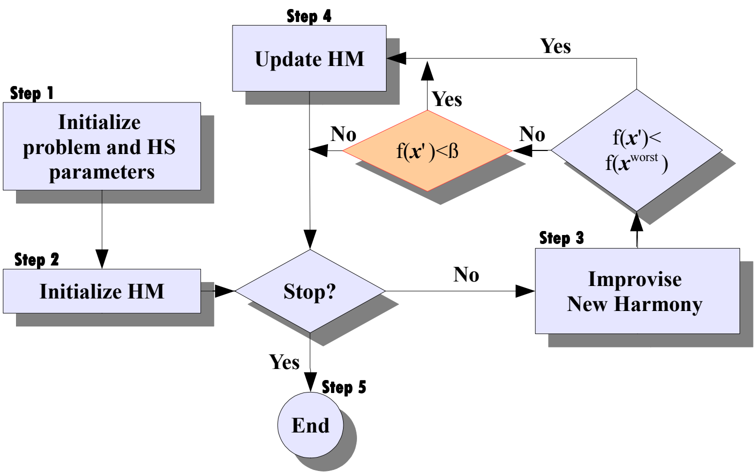

Fig. 1 The flowchart of the HS-GD algorithm

1. **Initialize the problem and HS-GD parameters.** Normally, the optimization problem is initially modeled as: $\min\{f(\boldsymbol{x})\,|\,\boldsymbol{x}\in\mathbf{X}\}$, where $f(\boldsymbol{x})$ is the objective function; $\boldsymbol{x}=\{x_i\,|\,i=1,\ldots,N\}$ is the set of decision variables. $\mathbf{X}=\{\boldsymbol{X}_i\,|\,i=1,\ldots,N\}$ is the possible value range for each decision variable, where $\boldsymbol{X}_i\in[LB_i,UB_i]$, where LB_i and UB_i are the lower and upper bounds for the decision

variable x_i respectively and N is the number of decision variables. The parameters of the HS-GD algorithm required to solve the optimization problem are also specified in this step as described in Table 1. Since this work only attends to the minimization problems, it is always referred to the boundary of accepting a worse solution with 'ceiling' instead of 'level'.

Table 1 Parameters of HS-GD algorithm

Parameter	Description
The Harmony Memory Consideration Rate (HMCR)	It is used in the improvisation process to determine whether the value of a decision variable is to be selected from the solutions stored in the Harmony Memory (HM)
The Harmony Memory Size (HMS)	It is similar to the population size in Genetic Algorithm
The Pitch Adjustment Rate (PAR)	It decides whether the decision variables are to be adjusted to a neighbouring value.
Number of Improvisations (NI)	It corresponds to the number of iterations.
Distance Bandwidth (BW)	It determines the distance of adjustment in the pitch adjustment operator.
Ceiling (β)	The ceiling is the boundary of the acceptability of the quality of a candidate solution. It is called "level" in maximization and "ceiling" in minimization problems
Decay rate ($\Delta\beta$)	The decay rate is the rate that the ceiling changes during the search process (increases in maximization and decreases in minimization)

2. **Initialize the harmony memory.** The harmony memory (HM) is an augmented matrix of size $N \times \text{HMS}$ which contains sets of solution vectors determined by HMS (see (1)). In this step, these vectors are randomly generated as follows: $x_i^j = LB_i + (UB_i - LB_i) \times U(0,1), \forall i = 1,2,\ldots,N$ and $\forall j = 1,2,\ldots,\text{HMS}$, and $U(0,1)$ generate a uniform random number between 0 and 1. The generated solutions are stored in HM in ascending order according to their objective function values.

$$\mathbf{HM} = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_N^1 \\ x_1^2 & x_2^2 & \cdots & x_N^2 \\ \vdots & \vdots & \cdots & \vdots \\ x_1^{\text{HMS}} & x_2^{\text{HMS}} & \cdots & x_N^{\text{HMS}} \end{bmatrix}. \tag{1}$$

After the HM is randomly filled by the solution vectors, the ceiling (β) is initialized to be assigned the fitness value of the worst solution vector in HM.
In an analogous manner to the HS-GD algorithm, the decay rate ($\Delta\beta$) of the ceiling is calculated using the following formula.

$$\Delta\beta = \frac{\beta_0 - f(\boldsymbol{x}^{opt})}{\text{NI}} \tag{2}$$

where β_0 is the initial value of the ceiling that is set to the fitness value of the worst solution in the initial HM, $f(\boldsymbol{x}^{opt})$ is the estimated quality of the final solution (the global optimum) and NI is the maximum number of improvisations.

3. **Improvise a new harmony.** In this step, the HS-GD algorithm generating (improvising) a new harmony vector from scratch, $\boldsymbol{x}' = (x'_1, x'_2, \cdots, x'_N)$, based on three operators: (1) memory consideration, (2) random consideration, and (3) pitch adjustment.

   Memory consideration. In memory consideration, the value of the first decision variable x'_1 is randomly *selected* from the historical values, $\{x_1^1, x_1^2, \ldots, x_1^{\text{HMS}}\}$, stored in HM vectors. Values of the other decision variables, $(x'_2, x'_3, \ldots, x'_N)$, are sequentially selected in the same manner with probability (w.p.) HMCR where HMCR $\in (0,1)$.

   Random consideration. Decision variables that are not assigned with values according to memory consideration are randomly assigned according to their possible range by random consideration with a probability of (1-HMCR) as follows:

$$x'_i \leftarrow \begin{cases} x'_i \in \{x_i^1, x_i^2, \ldots, x_i^{\text{HMS}}\} & \text{w.p.} \quad \text{HMCR} \\ x'_i \in \boldsymbol{X}_i & \text{w.p.} \quad \text{1 - HMCR.} \end{cases}$$

   Pitch adjustment. Each decision variable $x'_i, i \in \{1, 2, \ldots, N\}$ of a new harmony vector, that has been assigned a value by memory considerations is pitch adjusted with the probability of PAR where PAR $\in (0,1)$ as follows:

$$\text{Pitch adjusting decision for } x'_i \leftarrow \begin{cases} \text{Yes} & \text{w.p.} \quad \text{PAR} \\ \text{No} & \text{w.p.} \quad \text{1-PAR.} \end{cases}$$

   If the pitch adjustment decision for x'_i is Yes, the value of x'_i is modified to its neighboring value as follows: $x'_i = x'_i + U(-1,1) \times \text{BW}$

4. **Update the harmony memory.**
   This section describes the main focus of this research. The worst vector in HM is excluded and the new harmony $\boldsymbol{x}'$ is included if its fitness value meets the following two conditions:

   - Better than the fitness value of the worst solution vector in HM (greater in the maximization problems and less in minimization problems).
   - Better than or equal to the current ceiling (β).

   The ceiling (β) is degraded with every improvisation of a new solution vector by subtracting the decay rate $(\Delta\beta)$ value.

5. **Check the stop criterion.** Step 3 and step 4 of HS algorithm are repeated until NI is reached.

The procedure of HS-GD algorithm can be presented as in Algorithm 1:

Algorithm 1 HS-GD algorithm

Set HMCR, PAR, NI, HMS, BW.
$\boldsymbol{x}^{opt}$ = expected optimal solution of the minimization problem.
$x_i^j = LB_i + (UB_i - LB_i) \times U(0,1)$, $\forall i = 1,2,\ldots,N$ and $\forall j = 1,2,\ldots,$HMS {generate HM solutions}
Calculate($f(\boldsymbol{x}^j)$), $\forall j = (1,2,\ldots,\text{HMS})$
$\beta_0 = \beta = f(\boldsymbol{x}^{\text{worst}})$ of the initial HM.
$\Delta\beta = (\beta_0 - f(\boldsymbol{x}^{opt}))/$ NI.
$itr = 0$
while ($itr \leq$ NI) **do**
  $\boldsymbol{x}' = \phi$
  for $i = 1,\cdots,N$ **do**
    if ($U(0,1) \leq$ HMCR) **then**
      $x_i' \in \{x_i^1, x_i^2, \ldots, x_i^{\text{HMS}}\}$ {memory consideration}
      if ($U(0,1) \leq$ PAR) **then**
        $x_i' = x_i' + U(-1,1) \times$ FW { pitch adjustment }
      end if
    else
      $x_i' = LB_i + (UB_i - LB_i) \times U(0,1)$ { random consideration }
    end if
  end for
  if ($f(\boldsymbol{x}') < f(\boldsymbol{x}^{\text{worst}})$) OR ($f(\boldsymbol{x}') < \beta$) **then**
    Include $\boldsymbol{x}'$ to the **HM**.
    Exclude $\boldsymbol{x}^{\text{worst}}$ from **HM**.
  end if
  $\beta = \beta - \Delta\beta$
  $itr = itr + 1$
end while

3 Experimental results

3.1 Benchmark Functions

Table 2 overviews a summary for 10 global minimization benchmark functions used to evaluate HS-GD algorithm most of which previously used in [16, 4]. These benchmark functions provide a trad-off between unimodal and multimodal functions. The benchmark functions were implemented with N=30, with the exception of Six-Hump Camel-Back function which is two-dimensional.

3.2 Sensitivity Analysis of HS-GD Algorithm

The effect of different parameter settings of the HS-GD parameters (β_0 and $\Delta\beta$) is investigated. The parameters (HMCR, HMS and PAR) are set as recommended in the previous work of HS as follows [4]: HMS=50, HMCR=0.98, PAR=0.3,

Table 2 Benchmark functions used to evaluate HS variations

Function Name	Expression	Search Range	Optimum Value	Category [17]
Sphere function	$f_1(x) = \sum_{i=1}^{N} x_i^2$	$x_i \in [-100, 100]$	$\min(f_1) = f(0,\ldots,0) = 0$	unimodal
Schwefel's Problem 2.22 [20]	$f_2(x) = \sum_{i=1}^{N} \mid x_i \mid + \prod_{i=1}^{N} \mid x_i \mid$	$x_i \in [-10, 10]$	$\min(f_2) = f(0,\ldots,0) = 0$	unimodal
Step function	$f_3(x) = \sum_{i=1}^{N} (\lfloor x_i + 0.5 \rfloor)^2$	$x_i \in [-100, 100]$	$\min(f_3) = f(0,\ldots,0) = 0$	unimodal & discontinues
Rosenbrock function	$f_4(x) = \sum_{i=1}^{N-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	$x_i \in [-30, 30]$	$\min(f_4) = f(1,\ldots,1) = 0$	multimodal
Rotated hyper-ellipsoid function	$f_5(x) = \sum_{i=1}^{N} \left(\sum_{j=1}^{i} x_j \right)^2$	$x_i \in [-100, 100]$	$\min(f_5) = f(0,\ldots,0) = 0$	unimodal
Schwefel's problem 2.26 [20]	$f_6(x) = -\sum_{i=1}^{N} \left(x_i \sin(\sqrt{\mid x_i \mid}) \right)$	$x_i \in [-500, 500]$	$\min(f_6) = f(420.9687,\ldots, 420.9687) = -12569.5$	multimodal
Rastrigin function	$f_7(x) = \sum_{i=1}^{N} (x_i^2 - 10\cos(2\pi x_i) + 10)$	$x_i \in [-5.12, 5.12]$	$\min(f_7) = f(0,\ldots,0) = 0$	multimodal
Ackley's function	$f_8(x) = -20\exp\left(-0.2\sqrt{\frac{1}{30}\sum_{i=1}^{N} x_i^2} \right) - \exp\left(\frac{1}{30}\sum_{i=1}^{N} \cos(2\pi x_i) \right) + 20 + e$	$x_i \in [-32, 32]$	$\min(f_8) = f(0,\ldots,0) = 0$	multimodal
Griewank function	$f_9(x) = \frac{1}{4000}\sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	$x_i \in [-600, 600]$	$\min(f_9) = f(0,\ldots,0) = 0$	multimodal
Six-Hump Camel-Back function	$f_{10}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	$x_i \in [-5, 5]$	$\min(f_{10}) = f(-0.08983, 0.7126) = -1.0316285$	multimodal

NI=100,000, BW=0.03. Note that the default value of ceiling β_0 is $f(\boldsymbol{x}^{worst})$ where $\boldsymbol{x}^{worst}$ is the worst vector in the initial HM. Furthermore, $\Delta\beta$ is $\frac{\beta_0 - f(\boldsymbol{x}^{opt})}{NI}$.

3.2.1 The effect of initializing the ceiling (β)

In this section, the performance of the proposed HS-GD using two different ceiling values (β) is investigated. Table 3 summarizes the results of two values for initializing the ceiling parameter (β_0) in HS-GD. Firstly, the ceiling is initialized by assigning it the cost function of the best solution vector of the initial HM. Secondly, the ceiling is initialized by assigning it the cost function of the worst solution vector of the initial HM.

The results show that HS-GD is sensitive to the ceiling value of setting the initial ceiling. The best results are obtained when the ceiling is assigned the cost function of the worst harmony $f(\boldsymbol{x}^{worst})$ of the initial HM. Notably, assigning the best cost function $f(\boldsymbol{x}^{best})$ of the initial HM to (β_0) might hinder accepting the worst moves at

Table 3 The effect of initializing the ceiling in HS-GD for ten benchmark functions

Function	$\beta_0 = f(\boldsymbol{x}^{best})$	$\beta_0 = f(\boldsymbol{x}^{worst})$
f_1: Sphere	7.026E-09	**5.555E-09**
	(3.862E-09)	**(2.602E-09)**
f_2: Rosenbrock	1.020E+00	**8.338E-01**
	(5.344E-01)	**(5.140E-01)**
f_3: Ackley	9.423E-05	**8.722E-05**
	(2.587E-05)	**(2.637E-05)**
f_4: Griewank	5.547E-02	**4.347E-02**
	(3.057E-02)	**(2.755E-02)**
f_5: Rastrigin	8.374E-07	**7.684E-07**
	(4.384E-07)	**(4.986E-07)**
f_6: Schwefel Problem 2.22	**1.245E-04**	1.341E-04
	(3.290E-05)	(3.580E-05)
f_7: Step	7.140E-09	**6.681E-09**
	(4.589E-09)	**(5.605E-09)**
f_8: Rotated hyper-ellipsoid	7.754E+01	**5.664E+01**
	(7.554E+01)	**(4.243E+01)**
f_9: Schwefel Problem 2.26	**-4.190E+03**	**-4.190E+03**
	(7.531E-09)	**(3.571E-09)**
f_{10}: Camel-Back	**-1.032E+00**	**-1.032E+00**
	(0)	**(0)**

the early stage of the search process (especially when the fitness values of both the worst and the best harmonies of the initial HM are relatively distant). This probably allows the greedy replacement to dominate at the initial stage of the run, and thus to speed up the convergence towards, probably, unsought solutions.

3.2.2 Effect of the decay rate ($\Delta\beta$)

The performance of the proposed method using high and low decay rate ($\Delta\beta$) is investigated in this section. The different values of $\Delta\beta$ indicate the speed that the ceiling of accepting the worst moves is degraded.

– The effect of using high decay rates

The results for the ten benchmark functions using varying high ($\Delta\beta$) values (i.e. $\Delta\beta\times 20$, $\Delta\beta\times 2$, $\Delta\beta\times 1.05$ and the standard $\Delta\beta$) are summarized in Table 4.

The fast decay of the ceiling narrows the condition of accepting the worst moves. The value ($\Delta\beta\times 20$) means that the decay rate is proportional to 5% of the entire run. The value ($\Delta\beta\times 2$) means that the decay rate is proportional to 50% of the entire run. The value ($\Delta\beta\times 1.05$) means that the decay rate is proportional to roughly 95% of the entire run.

It is observed that the best results for the majority of the benchmark functions are obtained when the decay rate is proportional to the entire run.

Table 4 The effect of using high decay rates for ten benchmark functions

Function	$\Delta\beta\times 20$	$\Delta\beta\times 2$	$\Delta\beta\times 1.05$	$\Delta\beta\times 1$
f_1: Sphere	6.840E-09	6.084E-09	6.618E-09	**5.555E-09**
	(5.155E-09)	(4.588E-09)	(4.395E-09)	**(2.602E-09)**
f_2: Rosenbrock	9.338E-01	1.127E+00	1.027E+00	**8.338E-01**
	(5.636E-01)	(4.854E-01)	(5.901E-01)	**(5.140E-01)**
f_3: Ackley	**8.639E-05**	8.913E-05	8.765E-05	8.722E-05
	(2.422E-05)	(2.640E-05)	(2.877E-05)	(2.637E-05)
f_4: Griewank	4.547E-02	6.363E-02	5.431E-02	**4.347E-02**
	(2.425E-02)	(3.275E-02)	(3.020E-02)	**(2.755E-02)**
f_5: Rastrigin	9.965E-07	8.325E-07	9.026E-07	**7.684E-07**
	(6.988E-07)	(4.553E-07)	(6.025E-07)	**(4.986E-07)**
f_6: Schwefel Problem 2.22	1.370E-04	**1.276E-04**	1.279E-04	1.341E-04
	(3.544E-05)	**(3.528E-05)**	(3.666E-05)	(3.580E-05)
f_7: Step	7.192E-09	7.212E-09	8.045E-09	**6.681E-09**
	(3.162E-09)	(3.961E-09)	(3.506E-09)	**(5.605E-09)**
f_8: Rotated hyper-ellipsoid	6.822E+01	6.262E+01	6.602E+01	**5.664E+01**
	(5.222E+01)	(5.442E+01)	(7.556E+01)	**(4.243E+01)**
f_9: Schwefel Problem 2.26	**-4.190E+03**	**-4.190E+03**	**-4.190E+03**	**-4.190E+03**
	(5.414E-09)	**(2.011E-09)**	(4.096E-09)	(3.571E-09)
f_{10}: Camel-Back	**-1.032E+00**	**-1.032E+00**	**-1.032E+00**	**-1.032E+00**
	(0)	**(0)**	**(0)**	**(0)**

– The effect of using low decay rates

The results for the same ten benchmark functions using different low $\Delta\beta$ values (i.e. $\Delta\beta\div 20$, $\Delta\beta\div 2$, $\Delta\beta\div 1.05$ and the standard $\Delta\beta$) are summarized in Table5.

The slow decay of the ceiling expands the condition of accepting the worst moves. The value ($\Delta\beta\div 20$) means that the entire run is proportional to 5% of the entire decay of the ceiling. The value ($\Delta\beta\div 2$) means that the entire run is proportional to 50% of the entire decay. The value ($\Delta\beta\div 1.05$) means that the entire run is proportional to roughly 95% of the entire decay.

It is observed that the best results for the majority of the benchmark functions are obtained at the standard value of $\Delta\beta$ where the entire run is proportional to the entire decay of the ceiling β.

3.3 Comparative Analysis

Numerous variations of harmony search are presented in the literature [13]. To have a fair comparison, two improved variations of HS in addition to the basic HS algorithm are selected taking into consideration that all the selected methods are comparable to the proposed method in terms of the number of the basic operations that each method achieves (i.e. the time complexity). Furthermore, comparable recommended setting of the unified parameters among the selected methods is also considered. The

Table 5 The effect of using low decay rates for ten benchmark functions

Function	$\Delta\beta \div 20$	$\Delta\beta \div 2$	$\Delta\beta \div 1.05$	$\Delta\beta \div 1$
f_1: Sphere	7.264E-09	6.908E-09	6.802E-09	**5.555E-09**
	(4.325E-09)	(4.685E-09)	(4.461E-09)	**(2.602E-09)**
f_2: Rosenbrock	9.319E-01	1.134E+00	1.027E+00	**8.338E-01**
	(5.586E-01)	(4.878E-01)	(5.924E-01)	**(5.140E-01)**
f_3: Ackley	**8.302E-05**	8.658E-05	9.585E-05	8.722E-05
	(1.608E-05)	(2.956E-05)	(2.952E-05)	(2.637E-05)
f_4: Griewank	4.699E-02	6.363E-02	5.430E-02	**4.347E-02**
	(2.714E-02)	(3.274E-02)	(3.018E-02)	**(2.755E-02)**
f_5: Rastrigin	8.801E-07	8.748E-07	9.141E-07	**7.684E-07**
	(4.818E-07)	(4.250E-07)	(6.031E-07)	**(4.986E-07)**
f_6: Schwefel Problem 2.22	**1.257E-04**	1.343E-04	1.279E-04	1.341E-04
	(3.733E-05)	(3.976E-05)	(3.666E-05)	(3.580E-05)
f_7: Step	8.274E-09	9.000E-09	8.131E-09	**6.681E-09**
	(6.788E-09)	(4.578E-09)	(4.004E-09)	**(5.605E-09)**
f_8: Rotated hyper-ellipsoid	6.703E+01	6.708E+01	6.503E+01	**5.664E+01**
	(5.417E+01)	(6.124E+01)	(7.482E+01)	**(4.243E+01)**
f_9: Schwefel Problem 2.26	**-4.190E+03**	**-4.190E+03**	**-4.190E+03**	**-4.190E+03**
	5.902E-09	**(1.965E-09)**	(2.939E-09)	(3.571E-09)
f_{10}: Camel-Back	**-1.032E+00**	**-1.032E+00**	**-1.032E+00**	**-1.032E+00**
	(0)	**(0)**	**(0)**	**(0)**

methods that are used to compare the proposed variant of HS are summarized in Table 6.

Table 6 The methods used in the comparative study.

Method	Denotation	Reference
A new heuristic optimization algorithm: harmony search	HS	Geem et al., [14]
Self-adaptive harmony search algorithm	SaHS	Wang and Huang, [19]
An improved harmony search algorithm with differential mutation operator	DHS	Chakraborty et al., [11]

Table 7 summarizes the results of comparing the proposed method with the selected methods for ten-dimensional objective function (n=10). The results of HS, SaHS and DHS are obtained from [18].

As can be seen from the results, the proposed HS-GD algorithm outperforms the other methods for the majority of the benchmark optimization functions. f_1 is a unimodal problem, and it is straightforward and easy to solve. Thus the HS variant that tends to converge prematurely seems to be a more efficient choice for such problems.

f_2 is a unimodal and can be considered as a multimodal problem. It has a narrow valley from its local optima to its global optimum. f_3 has one narrow global optimum valley and many shallow local optima. f_4 is more difficult when the dimensions of the function decrease. f_5 is considered a more complex multimodal problem with multiple local optima that may lead to an ambush into a local optima

Table 7 Mean and standard deviation of five benchmark function's optimization results (N=10)

Function	HS	SaHS	DHS	HS-GD
Sphere	**3.52E-09**	1.90E-02	8.13E-02	5.56E-09
	-6.75E-09	-1.95E-02	-5.21E-02	-2.60E-09
Rosenbrock	1.05E+00	5.66E+00	6.03E+00	**8.34E-01**
	-4.97E-01	-2.58E+00	-1.65E+00	**-5.14E-01**
Ackley	9.56E-05	5.82E-02	1.47E-01	**8.72E-05**
	-2.68E-05	-4.90E-02	-6.70E-02	**-2.64E-05**
Griewanks	5.91E-02	8.42E-02	1.57E-01	**4.35E-02**
	-3.37E-02	-3.67E-02	-5.00E-02	**-2.76E-02**
Rastrigin	8.89E-07	1.39E-02	3.48E-02	**7.68E-07**
	-6.04E-07	-1.39E-02	-2.31E-02	**-4.99E-07**

easily. Therefore, an algorithm that is more efficient in maintaining a higher rate of diversity may be more capable of obtaining better results for these functions. Apparently, the proposed HS-GD algorithm seems to be the best choice for solving such problems.

4 Conclusion and future work

This paper has proposed a Harmony Search Great Deluge (HS-GD) algorithm which is able to avoid the premature convergence situation by means of employing the acceptance method of GD. In this context, the HS-GD algorithm is able to maintain the right balance between diversification (exploration) and intensification (exploitation) during the search. The HS-GD is evaluated using ten benchmark mathematical functions circulated in the literature. The proposed method is able to perform better than the classical HS. Additionally, comparative evaluation shows that the proposed method can also be considered as an efficient technique for global optimization problems. For future work, the following three directions can be recommended:

1. Incorporating other methods, such as local search techniques, in HS-GD is suggested to reinforce the intensification specifically at the advanced stage of the search process.
2. Modifying HS-GD itself in a way that the decay rate changes dynamically throughout the decrease of the ceiling (in minimization) or the increase of the level (in maximization).
3. The performance of the proposed method can be further investigated by utilizing it in solving the combinatorial problems such as the Travelling Salesman Problem (TSP), the Knapsack Problem (KP) and Timetabling Problems. These problems are constrained and the algorithm that attempts to solve them is more likely to fall into local minima easily. Accordingly, an algorithm that maintains a higher rate of diversity, such as HS-GD, seems to be an efficient choice.

References

[1] M.S. Abual-Rub, M.A. Al-Betar, R. Abdullah, and A.T. Khader. A hybrid harmony search algorithm for ab initio protein tertiary structure prediction. *Network Modeling and Analysis in Health Informatics and Bioinformatics*, pages 1–17.

[2] M. A. Al-Betar, A. T. Khader, and F. Nadi. Selection mechanisms in memory consideration for examination timetabling with harmony search. In *GECCO '10: Proceedings of Genetic and Evolutionary Computation Conference*. ACM, Portland, Oregon, USA, July 7–11 2010.

[3] M. A. Al-Betar, A. T. Khader, and J. J. Thomas. A combination of metaheuristic components based on harmony search for the uncapacitated examination timetabling. In *8th International Conference on the Practice and Theory of Automated Timetabling (PATAT 2010)*, Belfast, Northern Ireland, August 10–13 2010.

[4] M.A. Al-Betar, I.A. Doush, A.T. Khader, and M.A. Awadallah. Novel selection schemes for harmony search. *Applied Mathematics and Computation*, 218(10), 2011.

[5] M.A. Al-Betar and A.T. Khader. A harmony search algorithm for university course timetabling. *Annals of Operations Research*, 194:1–29, 2012.

[6] M.A. Al-Betar, A.T. Khader, and M. Zaman. University course timetabling using a hybrid harmony search metaheuristic algorithm. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, (99):1–18.

[7] O. Alia, M. Al-Betar, R. Mandava, and A. Khader. Data clustering using harmony search algorithm. *Swarm, Evolutionary, and Memetic Computing*, pages 79–88, 2011.

[8] M. Awadallah, A. Khader, M. Al-Betar, and A. Bolaji. Nurse rostering using modified harmony search algorithm. *Swarm, Evolutionary, and Memetic Computing*, pages 27–37, 2011.

[9] M.A. Awadallah, A.T. Khader, M.A. Al-Betar, and A.L. Bolaji. Nurse scheduling using harmony search. In *Bio-Inspired Computing: Theories and Applications (BIC-TA), 2011 Sixth International Conference on*, pages 58–63. IEEE, 2011.

[10] Christian Blum and Andrea Roli. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, 35(3):268–308, 2003.

[11] P. Chakraborty, G.G. Roy, S. Das, D. Jain, and A. Abraham. An improved harmony search algorithm with differential mutation operator. *Fundamenta Informaticae*, 95(4):401–426, 2009.

[12] G. Dueck. New optimization heuristics. *Journal of computational physics*, 104(1):86–92, 2005.

[13] Z. Geem. State-of-the-art in the structure of harmony search algorithm. *Recent Advances In Harmony Search Algorithm*, pages 1–10, 2010.

[14] Z. W. Geem, J. H. Kim, and G. V. Loganathan. A New Heuristic Optimization Algorithm: Harmony Search. *Simulation*, 76(2):60–68, 2001.

[15] Z.W. Geem, M. Fesanghary, J. Choi, MP Saka, J.C. Williams, M.T. Ayvaz, L. Li, S. Ryu, and A. Vasebi. Recent advances in harmony search. *Advance in evolutionary algorithms, I-Teach Education and Publishing, Vienna, Austria*, pages 127–142, 2008.

[16] M. G. H. Omran and M. Mahdavi. Global-best harmony search. *Applied Mathematics and Computation*, 198(2):643–656, 2008.

[17] Quan-Ke Pan, P.N. Suganthan, M. Fatih Tasgetiren, and J.J. Liang. A self-adaptive global best harmony search algorithm for continuous optimization problems. *Applied Mathematics and Computation*, 216(3):830 – 848, 2010.

[18] AK Qin and F. Forbes. Dynamic regional harmony search with opposition and local learning. In *Proceedings of the 13th annual conference companion on Genetic and evolutionary computation*, pages 53–54. ACM, 2011.

[19] C.M. Wang and Y.F. Huang. Self-adaptive harmony search algorithm for optimization. *Expert Systems with Applications*, 37(4):2826–2837, 2010.

[20] Xin Yao, Yong Liu, and Guangming Lin. Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation*, 3(2):82 –102, 1999.