Expert Systems with Applications 38 (2011) 4642–4648

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Short communication

An effective global harmony search algorithm for reliability problems

Dexuan Zou [a,*], Liqun Gao [a], Steven Li [b], Jianhua Wu [a]

[a] School of Information Science and Engineering, Northeastern University, Shenyang, Liaoning 110004, PR China
[b] Division of Business University of South Australia, GPO Box 2471, Adelaide SA 5001, Australia

ARTICLE INFO

Keywords:
Effective global harmony search algorithm
Reliability problems
Harmony search algorithm
Particle swarm optimization algorithm
Convergence

ABSTRACT

This paper proposes an effective global harmony search algorithm (EGHS) to solve two kinds of reliability problems: the complex (bridge) system optimization problem and the reliability–redundancy optimization problem of the overspeed protection system for a gas turbine. In general, the two problems are formulated as mixed-integer nonlinear programming problems with several constraints. The EGHS combines harmony search algorithm (HS) with concepts from the swarm intelligence of particle swarm optimization algorithm (PSO) to solve optimization problems. The proposed algorithm has been applied to two typical problems with results better than previously reported. The results have demonstrated that the EGHS has strong convergence and capacity of space exploration on solving reliability optimization problems.

1. Introduction

Reliability problem is an important issue in system design, and it often tries to achieve the highest reliability for a system which subject to several constraints such as cost, weight, and volume, for a reliability design of high quality enables a system to work more safely and efficiently. At the stage of designing a highly reliable system, an important problem is how to get the balance between reliability and other resources. That is, there is a difficulty in finding a balance between reliability and other resource constraints. In the past few decades, many approaches have been successfully proposed to solve this troublesome problem.

Chern and Jan (1986) developed a two-phase solution method for solving a class of reliability optimization problems with multiple-choice constraints. They presented a generalized model that can be stated as the problem of finding the optimum number of redundancies which maximize the system reliability subject to resource constraints, or the minimization of the system cost subject to constraints while maintaining an acceptable level of reliability. They assumed that at least one design alternative can be chosen for each subsystem. Li, Sun, and Kinnon (2005) proposed a convexification method to solve a class of continuous relaxation problems. Combined with a branch-and-bound method, their solution scheme provided an efficient way to find an exact optimal solution to integer reliability optimization in complex systems. The purpose of this paper is to develop a new efficient exact method for solving both pure and mixed-integer nonlinear programming problems arising from reliability optimization in complex systems using a convexification scheme. Caserta and Nodar (2009) proposed a Cross Entropy-based algorithm for reliability optimization of complex systems, where one wants to maximize the reliability of a system through optimal allocation of redundant components while respecting a set of budget constraints. Moreover, they showed how a Cross Entropy-based algorithm can be fine-tuned by using a training scheme based upon the Response Surface Methodology. Chen (2006) proposed a penalty guided artificial immune algorithm to solve mixed-integer reliability design problems. It can search over promising feasible and infeasible regions to find the feasible optimal/near optimal solution effectively and efficiently. Prasad and Kuo (2000) presented a search method (P and K-Algorithm) based on lexicographic order and an upper bound on the objective function for solving redundancy allocation problems in coherent systems. The main advantages of the P and K-Algorithm are its simplicity and its applicability to a wide range of complex optimization problems arising in system reliability design. Gen and Yun (2006) employed a soft computing approach for solving various reliability optimization problems. This method combined rough search (RS) technique and local search (LS) technique, which can prevent the premature convergence situation of its solution. In addition, many optimization algorithms based on swarm intelligence are also used to solve reliability problems, such as particle swarm optimization algorithm (Coelho, 2009; Elegbede, 2005; Yin, Yu, Wang, & Wang, 2007), genetic algorithm (Coit & Smith, 1996; Gen & Kim, 1999; Marseguerra, Zio, & Podofillini, 2004; Painton & Campbell, 1995), evolutionary algorithm (Aponte & Sanseverino, 2007; Ramirez-Marquez, 2008; Salazar, Rocco, & Galvn, 2006), ant colony algorithm (Meziane, Massim, Zeblah, Ghoraf, & Rahli,

* Corresponding author.
E-mail address: zoudexuan@163.com (D. Zou).

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.09.120

2005), and so on. Particle swarm optimization algorithm (PSO), which was originally proposed by Kennedy and Eberhart (1995), has been heavily researched and applied to many engineering optimization problems including the reliability problems. The PSO is a population-based heuristic global optimization technology, and it belongs to the category of swarm intelligence. Harmony search algorithm (Geem, Kim, & Loganathan, 2001) is a recently developed algorithm which is inspired by the phenomenon of musician attuning. The HS has been applied to many engineering optimization problems, but it has never been used to solve reliability problems.

This paper proposes a new version of HS – an effective global harmony search algorithm (EGHS) which is inspired by the swarm intelligence of particle swarm optimization algorithm. The EGHS algorithm is used to enhance the performance of HS so as to solve complex reliability problems. A novel location updating equation is designed to make the worst harmony of harmony memory move to the global best harmony in each iteration. Location updating can accelerate the convergence rate of the EGHS; however, it also accelerates the premature convergence of the EGHS. To overcome this disadvantage, random selection with a small probability is introduced into the EGHS. Several experiments have demonstrated that the EGHS algorithm has better performance than the HS algorithm and its improved algorithm (Mahdavi, Fesanghary, & Damangir, 2007) on solving reliability problems.

The paper is organized as follows. In Section 2, the procedure of the HS is briefly presented, and one improved version of HS is also summarized. In Section 3, an effective global harmony search algorithm (EGHS) is proposed, and the procedure of the EGHS is adequately described. In Section 4, some preparation work is considered for using the EGHS to solve reliability problems. In Section 5, a large number of experiments are carried out to test the performance of the EGHS for reliability optimization problems. We end this paper with some conclusions and comments for further research in Section 6.

2. The HS algorithm

The HS algorithm is based on natural musical performance processes that occur when a musician searches for a better state of harmony, such as during jazz improvisation. Jazz improvisation seeks to find musically pleasing harmony as determined by an aesthetic standard, just as the optimization process seeks to find a global solution as determined by an objective function. The pitch of each musical instrument determines the aesthetic quality, just as the objective function value is determined by the set of values assigned to each decision variable. In general, the HS algorithm works as follows:

Step 1. Initialize the problem and algorithm parameters.

The optimization problem is defined as Minimize $f(x)$ subject to $x_{iL} \leqslant x_i \leqslant x_{iU} (i = 1,2,\ldots,N)$. x_{iL} and x_{iU} are the lower and upper bounds for decision variables. The HS algorithm parameters are also specified in this step. They are the harmony memory size (HMS) or the number of solution vectors in the harmony memory; harmony memory considering rate (HMCR); pitch adjusting rate (PAR); and the number of improvisations (K) or stopping criterion.

Step 2. Initialize the harmony memory.

The initial harmony memory is generated from an uniform distribution in the ranges $[x_{iL}, x_{iU}]$ $(i = 1,2,\ldots,N)$, as shown in Eq. (1):

$$HM = \begin{pmatrix} x_1^1 & x_2^1 & \cdots & x_N^1 \\ x_1^2 & x_2^2 & \cdots & x_N^2 \\ \vdots & \vdots & & \vdots \\ x_1^{HMS-1} & x_2^{HMS-1} & \cdots & x_N^{HMS-1} \\ x_1^{HMS} & x_2^{HMS} & \cdots & x_N^{HMS} \end{pmatrix}. \quad (1)$$

Step 3. Improvise a new harmony.

Generating a new harmony is called improvisation. The new harmony vector $x' = (x'_1, x'_2, \ldots, x'_N)$ is determined by three rules: memory consideration, pitch adjustment, and random selection. The procedure works as follows:

for each $i \in [1,N]$ **do**
  if rand() $\leqslant$ HMCR **then**
    $x'_i = x_i^j (j = 1,2,\ldots,HMS)$ %memory consideration
    if rand () $\leqslant$ PAR **then**
      $x'_i = x'_i \pm r \times bw$ %pitch adjustment
      if $x'_i > x_{iU}$
        $x'_i = x_{iU}$
      else if $x'_i < x_{iL}$
        $x'_i = x_{iL}$
      end
    end
  else
    $x'_i = x_{iL} + rand() \times (x_{iU} - _{iL})$ %random selection
  end
end

Both r and $rand()$ are uniformly generated random number in the region of [0,1], and bw is an arbitrary distance bandwidth.

Step 4. Update harmony memory.

If the fitness of the improvised harmony vector $x' = (x'_1, x'_2, \ldots, x'_N)$ is better than that of the worst harmony, replace the worst harmony in the HM with x'.

Step 5. Check the stopping criterion: If the stopping criterion (maximum number of iterations K) is satisfied, computation is terminated. Otherwise, step 3 is repeated.

The most important step of the HS algorithm is Step 3, and it includes memory consideration, pitch adjustment, and random selection. PAR and bw have a profound effect on the performance of the HS. (Mahdavi et al., 2007) proposed a new variant of the HS, called the improved harmony search (IHS). The IHS dynamically updates PAR and bw according to the following equations:

$$\mathrm{PAR(k)}j = \mathrm{PAR_{min}} + \frac{\mathrm{PAR_{max}} - \mathrm{PAR_{min}}}{\mathrm{K}}\mathrm{k}, \quad (2)$$

$$\mathrm{bw(k)} = \mathrm{bw_{max}} \exp\left(\frac{\ln\left(\frac{\mathrm{bw_{min}}}{\mathrm{bw_{max}}}\right)}{\mathrm{K}}\mathrm{k}\right), \quad (3)$$

where K is the maximum number of iterations, and k is the current number of iterations; $\mathrm{PAR_{min}}$ and $\mathrm{PAR_{max}}$ are the minimum adjusting rate and the maximum adjusting rate, respectively; $\mathrm{bw_{min}}$ and $\mathrm{bw_{max}}$ are the minimum bandwidth and the maximum bandwidth, respectively. A large number of experiments and studies show that the IHS based on improved PAR and bw has better optimization performance than the HS in most cases.

3. An effective global harmony search algorithm

Inspired by the swarm intelligence of particle swarm optimization algorithm, a new variation of HS is proposed in this paper. The new approach, called EGHS, modifies the improvisation step of the HS such that the new harmony can mimic the global best harmony in the HM. The EGHS and the HS are different in three aspects as follows:

(1) In Step 1, Harmony memory considering rate (HMCR) and pitch adjusting rate (PAR) are excluded from the EGHS, and location updating probability (LUP) is included in the EGHS;
(2) In Step 3, the EGHS modifies improvisation step of the HS, and it works as follows:

for each $i \in [1, N]$ **do**
  if rand () $\leqslant$ LUP **then**
    $x'_i = x_i^{best} - C(k) \times rand() \times (x_i^{best} - x_i^{worst})$
    %location updating
  else
    $x'_i = x_{iL} + rand() \times (x_{iU} - x_{iL})$ %random selection
  end
end

Here, LUP is defined as location updating probability. x' is a new harmony vector improvised, x'_i is the ith component of x'. "best" and "worst" are the indexes of the global best harmony and the worst harmony in HM, respectively. x_i^{worst} represents the ith component of the worst solution in harmony memory (HM), and x_i^{best} represents the ith component of the global best solution. $C(k)$ is defined as coefficient of optimization scale, and it decreases linearly from 1 to 0 throughout the iteration. $RQ = C(k) \times |x_i^{best} - x_i^{worst}|$ is defined as optimization scale which is the searching region of x'. To analyze the rationality of location updating equation, the schematic diagram of location updating equation is shown as Fig. 1. In Fig. 1, the location of x_i^{worst} is at P, and the location of x_i^{best} is at Q. x_i^{worst} at P should move to x_i^{best} at Q according to "The individual is inclined to imitate its successful companion". The terminal of x'_i locates at a random location between R and Q. A rational explanation is as follows: in the early stage of optimization, if $\frac{RQ}{PQ} = C(k) \to 1$, then R is close to P, which is beneficial to global search; while in the late stage of optimization, if $\frac{RQ}{PQ} = C(k) \to 0$, then R is close to Q, which is beneficial to local search. Based on above consideration, $C(k)$ is designed to decrease linearly throughout the whole iteration. The expression of $C(k)$ is as Eq. (4).

$$C(k) = C_{max} - \frac{C_{max} - C_{min}}{K} k. \tag{4}$$

K is the maximum number of iterations, and k is the number of current iterations. The values of C_{max} and C_{min} are 1 and 0, respectively. The reasonable setting of $C(k)$ keeps a balance between the global search and the local search. Random selection with a small probability is carried out for each component of any non-best solution vector, for it can enhance the capacity of escaping from the local optimum for the proposed algorithm.

(3) In Step 4, the EGHS replaces the worst harmony x^{worst} in HM with the new harmony x' even if x' is worse than x^{worst}.

4. Some preparation work for using the EGHS to solve reliability problems

4.1. Constrained optimization

The general mathematical model of reliability optimization problems can be formulated as follows:

$$\begin{aligned} &\text{Max } f(\mathbf{x}) \\ &\text{s.t.} \quad g_j(\mathbf{x}) \leqslant 0, \quad j = 1, 2, \ldots, n_g, \end{aligned} \tag{5}$$

where $f(\mathbf{x})$ is reliability function, $g_j(\mathbf{x})$ is the jth resource constraint, and n_g is the number of constraints. There is a big difference between unconstrained optimization problems and constrained optimization problems. For unconstrained optimization problem, its global best solution vector is the one who has the minimum or maximum objective function value. In the mean time, the global best solution vector of constrained optimization problem is hard to determine and measure, for it is difficult to tradeoff the balance between the constraints and the objective function value.

The goal of the EGHS algorithm is to adapt the unfeasible solutions to the feasible solutions, so as to reduce the constraint violations of the search for obtaining the highest reliability. For constrained problems, most methods were based on penalty function methods that transform $f(\mathbf{x})$ into an unconstrained function $F(\mathbf{x})$, consisting of a sum of the objective and the constraints weighted by penalties. By using penalty function methods, the objectives are inclined to guide the search toward the feasible solutions. A penalty function method has been also used for handling constrained reliability problems in this paper, and it exerts the penalty on infeasible solutions based on the distance away from the feasible region. It is well known that the maximization of $f(x)$ can be transformed into the minimization of $-f(x)$, thus, according to the Eq. (5) in the reliability problem formulation, the corresponding penalty function has been defined and described as: $\text{Min}F(\mathbf{x}) = -f(\mathbf{x}) + \lambda \sum_{j=1}^{ng} \max(0, g_j)$, where, λ represents penalty coefficient, and it is set to 10^5 in this paper. The above reliability value is to be minimized when the penalty is minimized.

4.2. Process for discrete variables

Many engineering optimization problems involve discrete variables, and for reliability problems, these discrete variables are denoted with n_i which represents the number of components in subsystem i. Any n_i adjusted by Step 3 is a real number, and the most direct processing method is transforming it into a nearest integer.

5. Experimental results and analysis

To study and analyze the optimization performance of the EGHS algorithm for reliability problems, two test problems ($P1$ and $P2$) are considered. The two examples include a complex (bridge) system and an overspeed protection system used by Coelho (2009), and they are formulated below (See Figs. 2 and 3):

5.1. P1. Complex (bridge) system

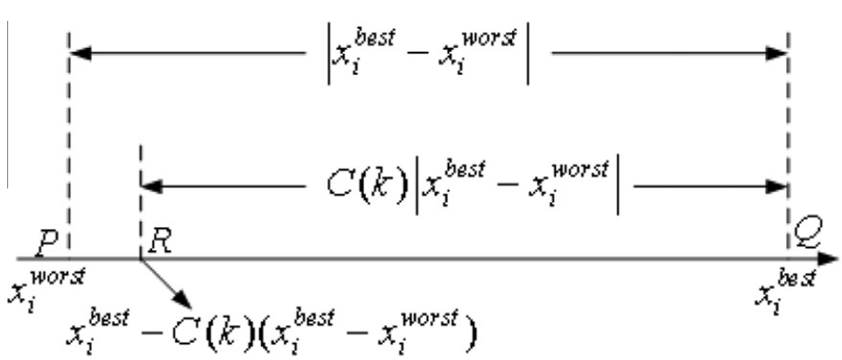

Fig. 1. The schematic diagram of location updating.

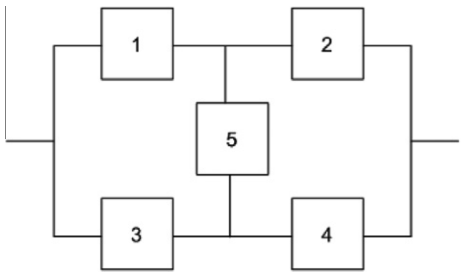

Fig. 2. The schematic diagram of complex (bridge) system.

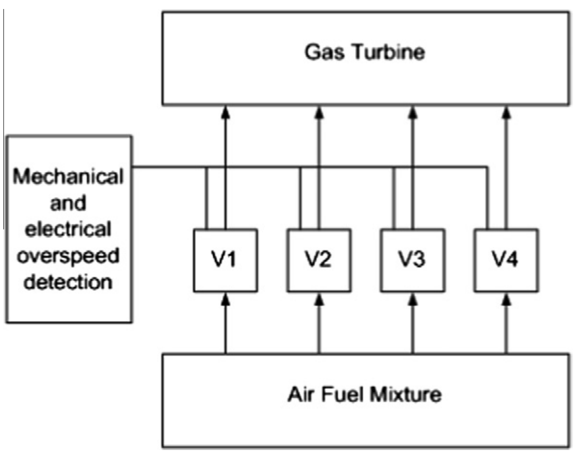

Fig. 3. Schematic diagram for the overspeed protection system of a gas turbine.

*P*1 is a nonlinear mixed-integer programming problem for a complex (bridge) system with five subsystems, and this example problem is used to demonstrate the efficiency of EGHS algorithm. The complex (bridge) system optimization problem is as follows:

$$\begin{aligned}
\text{Max } f(\mathbf{r},\mathbf{n}) = {} & R_1R_2 + R_3R_4 + R_1R_4R_5 + R_2R_3R_5 \\
& - R_1R_2R_3R_4 - R_1R_2R_3R_5 - R_1R_2R_4R_5 \\
& - R_1R_3R_4R_5 - R_2R_3R_4R_5 + 2R_1R_2R_3R_4R_5 \\
\text{s.t.} \quad & g_1(\mathbf{r},\mathbf{n}) = \sum_{i=1}^{m} w_i v_i^2 n_i^2 - V \leqslant 0, \\
& g_2(\mathbf{r},\mathbf{n}) = \sum_{i=1}^{m} \alpha_i \left(-\frac{1000}{\ln(r_i)}\right)^{\beta_i} [n_i + \exp(0.25n_i)] - C \leqslant 0, \\
& g_3(\mathbf{r},\mathbf{n}) = \sum_{i=1}^{m} w_i n_i \exp(0.25n_i) - W \leqslant 0, \\
& 0 \leqslant r_i \leqslant 1, \quad n_i \in Z^+, \quad 1 \leqslant i \leqslant m,
\end{aligned}$$

m is the number of subsystems in the system, and n_i is the number of components in subsystem i, $(1 \leqslant i \leqslant m)$. r_i is the reliability of each component in subsystem i, and $q_i = 1 - r_i$ is the failure probability of each component in subsystem i. $R_i(n_i) = 1 - q_i^{n_i}$ is the reliability of subsystem i, $f(\mathbf{r},\mathbf{n})$ is the system reliability. w_i is the weight of each component in subsystem i, v_i is the volume of each component in subsystem i, and c_i is the cost of each component in subsystem i. V is the upper limit on the sum of the subsystems' products of volume and weight; C is the upper limit on the cost of the system; W is the upper limit on the weight of the system. The parameters β_i and α_i are physical features of system components. Constraint $g_1(\mathbf{r},\mathbf{n})$ is a combination of weight, redundancy allocation, and volume. $g_2(\mathbf{r},\mathbf{n})$ is a cost constraint, while $g_3(\mathbf{r},\mathbf{n})$ is a weight constraint. The input parameters of the complex (bridge) system are shown in Table 1.

Table 1
Data used in complex (bridge) system (*P*1).

i	$10^5\alpha_i$	β_i	$w_i v_i^2$	w_i	V	C	W
1	2.330	1.5	1	7	110	175	200
2	1.450	1.5	2	8			
3	0.541	1.5	3	8			
4	8.050	1.5	4	6			
5	1.950	1.5	2	9			

5.2. P2. Overspeed protection system for a gas turbine

To study and analyze the performance of the NGHS algorithm for the mixed-integer nonlinear reliability design problem, the reliability–redundancy optimization problem of the overspeed protection system for a gas turbine is also considered. Overspeed detection is continuously provided by the electrical and mechanical systems. When an overspeed occurs, it is necessary to cut off the fuel supply. For this purpose, 4 control valves (V1–V4) must close. The control system is modeled as a 4-stage series system. The objective is to determine an optimal level of r_i and n_i at each stage i such that the system reliability is maximized. This reliability problem is formulated as follows:

$$\begin{aligned}
& \text{Max } f(\mathbf{r},\mathbf{n}) = \prod_{i=1}^{m} [1 - (1 - r_i)^{n_i}] \\
& \text{s.t.} \quad g_1(\mathbf{r},\mathbf{n}) = \sum_{i=1}^{m} v_i n_i^2 - V \leqslant 0, \\
& \qquad g_2(\mathbf{r},\mathbf{n}) = \sum_{i=1}^{m} C(r_i)[n_i + \exp(0.25n_i)] - C \leqslant 0, \\
& \qquad g_3(\mathbf{r},\mathbf{n}) = \sum_{i=1}^{m} w_i n_i \exp(0.25n_i) - W \leqslant 0, \\
& \qquad 0.5 \leqslant r_i \leqslant 1 - 10^{-6}, \quad r_i \in R^+, \quad 1 \leqslant n_i \leqslant 10, \quad n_i \in Z^+,
\end{aligned}$$

r_i is reliability of component in stage i, and n_i is the number of redundant components in stage i. v_i is the product of weight and volume per element at stage i. w_i is the weight of each components at the stage i. The exp $(n_i/4)$ accounts for the interconnecting hardware. $C(r_i) = \alpha_i\left(-\frac{T}{\ln(r_i)}\right)^{\beta_i}$ is the cost of each component with reliability r_i at subsystem i. α_i and β_i are constants representing the physical characteristics of each component at stage i. T is the operating time during which the component must not fail. The input parameters defining the overspeed protection system for a gas turbine are shown in Table 2.

In this paper, three harmony search algorithms including HS, IHS, and EGHS are used to solve above two examples (Coelho, 2009). The parameters of the three algorithms are as follows:

For the HS algorithm, harmony memory size HMS = 5, harmony memory consideration rate HMCR = 0.9, pitch adjusting rate PAR = 0.3, bandwidth $\text{bw} = 0.5 \times 10^{-4} \times (x_U - x_L)$. For the IHS algorithm, HMS = 5 and HMCR = 0.9, the minimum adjusting rate $\text{PAR}_{min} = 0.01$ and the maximum adjusting rate $\text{PAR}_{max} = 0.99$, the minimum bandwidth $\text{bw}_{min} = 0.5 \times 10^{-6} \times (x_U - x_L)$ and the maximum bandwidth $\text{bw}_{max} = 0.5 \times 10^{-2} \times (x_U - x_L)$. For the EGHS

Table 2
Data used in overspeed protection system (*P*2).

i	$10^5\alpha_i$	β_i	v_i	w_i	V	C	W	T
1	1.0	1.5	1	6	250	400	500	1000 h
2	2.3	1.5	2	6				
3	0.3	1.5	3	8				
4	2.3	1.5	2	7				

algorithm, HMS = 5, and location updating probability LUP = 0.9. The maximum number of iterations is set to 15,000 and 20,000 for $P1$ and $P2$, respectively. 50 independent runs were made for these three algorithms, and the comparison between the optimization results of the EGHS and those of the two other harmony search algorithms are recorded in Tables 3 and 4.

SD represents standard deviation. For $P1$ and $P2$, the best, worst, mean results obtained by the EGHS are all very close to each other in each case, and the standard deviations are 1.0699e−05 and 2.8874e−05, respectively. These results have demonstrated that the EGHS has strong convergence and stability than two other harmony search algorithms (As Figs. 4 and 5). In addition, the best results for $P1$ and $P2$ using EGHS algorithm are 0.99988960 and 0.99995463, respectively, and both results are better than those obtained by two other harmony search algorithms.

Tables 5 and 6 compare the best results obtained by the EGHS for two examples with those of two other harmony search algorithms (the HS and the IHS) in this paper. For measuring the improvement, MPI (maximum possible improvement) can be used to measure the amount improvement of the solutions found by the EGHS to those found by two other harmony search algorithms, and it is described as: $MPI(\%) = (f_{EGHS} - f_{other})/(1 - f_{other})$, where f_{EGHS} represents the best system reliability obtained by the EGHS and f_{other} represents the best system reliability obtained by the HS or the IHS.

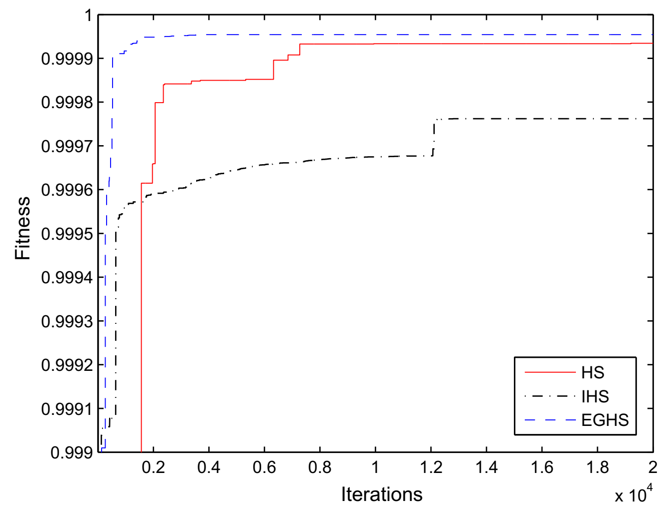

Fig. 5. The result of $P2$ using three algorithms.

Here, Slack represents the unused resources. In Table 5, for the best results found by the HS and the IHS, the corresponding improvements made by the EGHS are 6.3851% and 0.2530%, respectively. In Table 6, for the best results found by the HS and the IHS, the corresponding improvements made by the proposed approach are 9.3869% and 8.1579%, respectively. In short, the pro-

Table 3
Results of the complex (bridge) system using three algorithms.

Alg.	Best	Worst	Mean	SD
HS	0.99988207	0.99778878	0.99970556	3.2e−04
IHS	0.99988932	0.99917988	0.99977004	1.5e−04
EGHS	0.99988960	0.99982887	0.99988263	1.6e−05

Table 4
Results of the overspeed protection system using three algorithms.

Alg.	Best	Worst	Mean	SD
HS	0.99994993	0.99840124	0.99976502	2.9e−04
IHS	0.99995060	0.99932384	0.99981880	1.6e−04
EGHS	0.99995463	0.99985315	0.99993588	2.2e−05

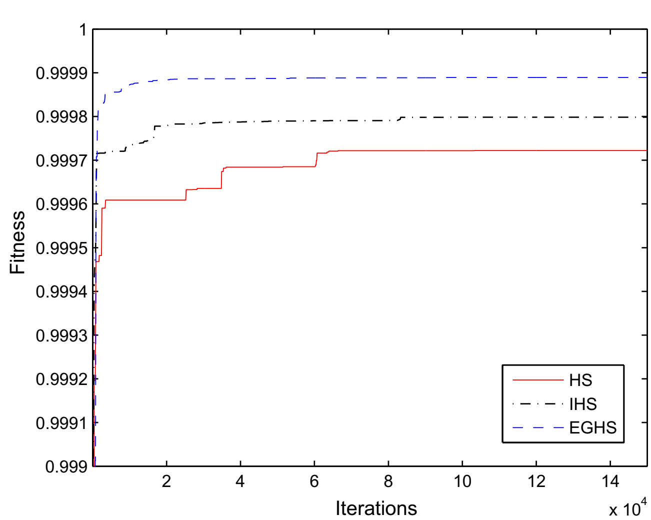

Fig. 4. The result of $P1$ using three algorithms.

Table 5
Comparison of best result for the complex (bridge) system with other results presented in literature.

Parameter	HS	IHS	EGHS
$f(\mathbf{r},\mathbf{n})$	0.99988207	0.99988932	0.99988960
n_1	3	3	3
n_2	3	3	3
n_3	2	2	2
n_4	4	4	4
n_5	1	1	1
r_1	0.80061199	0.83193782	0.82983999
r_2	0.84414296	0.85725105	0.85798911
r_3	0.92568172	0.91017625	0.91333926
r_4	0.67316720	0.64880195	0.64674479
r_5	0.71894936	0.70207710	0.70310972
MPI (%)	6.3851	0.2530	–
Slack (g_1)	5	5	5
Slack (g_2)	0.19544999	0.00001583	0.00000594
Slack (g_3)	1.56046629	1.56046629	1.56046629

Table 6
Comparison of best result for the complex (bridge) system with other results presented in literature.

Parameter	HS	IHS	EGHS
$f(\mathbf{r},\mathbf{n})$	0.99994993	0.99995060	0.99995463
n_1	5	5	5
n_2	6	5	6
n_3	4	4	4
n_4	5	6	5
r_1	0.89230939	0.90406843	0.900925066
r_2	0.84704621	0.88799742	0.851636929
r_3	0.94171595	0.95523072	0.948079849
r_4	0.89763285	0.83575543	0.887654500
MPI (%)	9.3869	8.1579	–
Slack (g_1)	55	55	55
Slack (g_2)	0.16246750	0.00009134	0.00000105
Slack (g_3)	24.80188272	15.36346309	24.80188272

Table 7
Comparison of best result for the complex (bridge) system with other results presented in literature.

Parameter	Hikita et al. (1992)	Hsieh et al. (1998)	Chen (2006)	Coelho (2009)	Zou
$f(\mathbf{r},\mathbf{n})$	0.9997894	0.99987916	0.99988921	0.99988957	0.99988960
n_1	3	3	3	3	3
n_2	3	3	3	3	3
n_3	2	3	3	2	2
n_4	3	3	3	4	4
n_5	2	1	1	1	1
r_1	0.814483	0.814090	0.812485	0.826678	0.82983999
r_2	0.821383	0.864614	0.867661	0.857172	0.85798911
r_3	0.896151	0.890291	0.861221	0.914629	0.91333926
r_4	0.713091	0.701190	0.713852	0.648918	0.64674479
r_5	0.814091	0.734731	0.756699	0.715290	0.70310972
MPI (%)	47.5783	8.6395	0.3520	0.0272	–
Slack (g_1)	18	18	19	5	5
Slack (g_2)	1.854075	0.376347	0.001494	0.000339	0.00000594
Slack (g_3)	4.264770	4.264770	4.264770	1.560466	1.56046629

Table 8
Comparison of best result for the overspeed protection system for a gas turbine with other results presented in literature.

Parameter	Yokota et al. (1996)	Dhingra (1992)	Chen (2006)	Coelho (2009)	Zou
$f(\mathbf{r},\mathbf{n})$	0.999468	0.99961	0.999942	0.999953	0.999955
n_1	3	6	5	5	5
n_2	6	6	5	6	6
n_3	3	3	5	4	4
n_4	5	5	5	5	5
r_1	0.965593	0.81604	0.903800	0.902231	0.900925066
r_2	0.760592	0.80309	0.874992	0.856325	0.851636929
r_3	0.972646	0.98364	0.919898	0.948145	0.948079849
r_4	0.804660	0.80373	0.890609	0.883156	0.887654500
MPI (%)	91.5413	88.4615	22.4138	4.2553	–
Slack (g_1)	92	65	50	55	55
Slack (g_2)	70.733576	0.064	0.002152	0.975465	0.00000584
Slack (g_3)	127.583189	4.348	28.803701	24.801882	24.80188272

posed approach has demonstrated stronger capacity of space exploration than two other harmony search algorithms.

Tables 7 and 8 compare the best results obtained in this paper for two examples with those of other studies reported in the literature. It is obvious that the two best results reported here using the EGHS algorithm are both higher than recent studies presented in the literature. For measuring the improvement, MPI (maximum possible improvement) can be used to measure the amount improvement of the solutions found by the proposed approach to the previous best know solutions, and it is described as: $MPI(\%) = (f_{Zou} - f_{other})/(1 - f_{other})$, where f_{Zou} represents the best system reliability obtained by the proposed algorithm and f_{other} represents the best system reliability obtained by any other method in literature.

Slack is the unused resources. By using MPI, it shows the proposed approach made improvements in $P1$ and $P2$. It can be seen from Table 7 that, for Hikita, Nakagawa, and Harihisa (1992), Hsieh, Chen, and Bricker (1998), Chen (2006), Coelho (2009), the corresponding improvements made by the proposed approach are 47.5878%, 8.6561%, 0.3701%, and 0.0453%, respectively. It should be noticed that even very small improvements in reliability are critical and beneficial to system security and system efficiency. From Table 8, it can be seen that, for Yokota, Gen, and Li, 1996; Dhingra, 1992; Chen, 2006; Coelho, 2009, the corresponding improvements made by the proposed approach are 91.5413%, 88.4615%, 22.4138%, and 4.2553%, respectively. The solution found by the proposed approach is much superior to those by Yokota et al. (1996), Dhingra (1992). In short, the proposed approach has demonstrated stronger capability than other approaches in finding a best solution for reliability optimization problems.

6. Conclusion

In this paper, the optimization performance of the EGHS algorithm on solving reliability problems has been extensively investigated by several experimental studies. The experimental results illustrate that the EGHS has stronger convergence than the other approaches in recent literature. The results also illustrate that the EGHS has high exploration capability of solution space throughout the whole iteration due to the utilization of random selection. In short, it is a promising optimization algorithm, and it may find the required optima in cases when the problem to be solved is too complicated and complex.

References

Aponte, D. E. S., & Sanseverino, C. M. R. (2007). Solving advanced multi-objective robust designs by means of multiple objective evolutionary algorithms (MOEA): A reliability application. *Reliability Engineering and System Safety, 92*(6), 697–706.

Caserta, M., & Nodar, M. C. (2009). A cross entropy based algorithm for reliability problems. *Journal of Heuristics, 15*(5), 479–501.

Chen, T. C. (2006). IAs based approach for reliability redundancy alposition problems. *Applied Mathematics and Computation, 182*(2), 1556–1567.

Chern, M. S., & Jan, R. H. (1986). Reliability optimization problems with multiple constraints. *IEEE Transactions on Reliability, 35*(4), 431–436.

Coelho, L. S. (2009). An efficient particle swarm approach for mixed-integer programming in reliability–redundancy optimization applications. *Reliability Engineering and System Safety, 94*(4), 830–837.

Coit, D. W., & Smith, A. E. (1996). Reliability optimization of series-parallel systems using a genetic algorithm. *IEEE Transactions on Reliability, 45*(2), 254–260.

Dhingra, A. K. (1992). Optimal apportionment of reliability and redundancy in series systems under multiple objectives. *IEEE Transaction on Reliability, 41*(4), 576–582.

Elegbede, C. (2005). Structural reliability assessment based on particles swarm optimization. *Structural Safety, 27*(2), 171–186.

Geem, Z. W., Kim, J. H., & Loganathan, G. V. (2001). A new heuristic optimization algorithm: Harmony search. *Simulation, 76*(2), 60–68.

Gen, M., & Kim, J. R. (1999). GA-based reliability design: State-of-the-art survey. *Computers and Industrial Engineering, 37*(1–2), 151–155.

Gen, M., & Yun, Y. S. (2006). Soft computing approach for reliability optimization: State-of-the-art survey. *Reliability Engineering and System Safety, 91*(9), 1008–1026.

Hikita, M., Nakagawa, H., & Harihisa, H. (1992). Reliability optimization of systems by a surrogate constraints algorithm. *IEEE Transaction on Reliability, 41*(3), 473–480.

Hsieh, Y. C., Chen, T. C., & Bricker, D. L. (1998). Genetic algorithm for reliability design problems. *Microelectron Reliability, 38*, 1599–1605.

Kennedy, J., & Eberhart, R. C. (1995). Particle swarm optimization. *Proceedings of IEEE International Conference on Neural Networks*, 1942–1948.

Li, D., Sun, X. L., & Kinnon, K. M. (2005). An exact solution method for reliability optimization in complex systems. *Annals of Operations Research, 133*(1), 129–148.

Mahdavi, M., Fesanghary, M., & Damangir, E. (2007). An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation, 188*(2), 1567–1579.

Marseguerra, M., Zio, E., & Podofillini, L. (2004). Optimal reliability/availability of uncertain systems via multi-objective genetic algorithms. *IEEE Transactions on Reliability, 53*(3), 424–434.

Meziane, R., Massim, Y., Zeblah, A., Ghoraf, A., & Rahli, R. (2005). Reliability optimization using ant colony algorithm under performance and cost constraints. *Electric Power Systems Research, 76*(1–3), 1–8.

Painton, L., & Campbell, J. (1995). Genetic algorithms in optimization of system reliability. *IEEE Transactions on Reliability, 44*(2), 172–178.

Prasad, V. R., & Kuo, W. (2000). Reliability optimization of coherent systems. *IEEE Transactions on Reliability, 49*(3), 323–330.

Ramirez-Marquez, J. E. (2008). Port-of-entry safety via the reliability optimization of container inspection strategy through an evolutionary approach. *Reliability Engineering and System Safety, 93*(11), 1698–1709.

Salazar, D., Rocco, C. M., & Galvn, B. J. (2006). Optimization of constrained multiple-objective reliability problems using evolutionary algorithms. *Reliability Engineering and System Safety, 91*(9), 1057–1070.

Yin, P. Y., Yu, S. S., Wang, P. P., & Wang, Y. T. (2007). Task alposition for maximizing reliability of a distributed system using hybrid particle swarm optimization. *Journal of Systems and Software, 80*(5), 724–735.

Yokota, T., Gen, M., & Li, H. H. (1996). Genetic algorithm for nonlinear mixed-integer programming problems and its application. *Computers and Industrial Engineering, 30*(4), 905–917.