



Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems

M. Fesanghary^{a,*}, M. Mahdavi^b, M. Minary-Jolandan^c, Y. Alizadeh^a

^a *Department of Mechanical Engineering, Amirkabir University of Technology, 424 Hafez Avenue, 15875-4413 Tehran, Iran*

^b *Department of Computer Engineering, Sharif University of Technology, Azadi Street, 11365-8639 Tehran, Iran*

^c *Department of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, 1206 West Green Street, Urbana, IL 61801, United States*

Received 4 April 2007; received in revised form 2 January 2008; accepted 11 February 2008

Abstract

This study presents a hybrid harmony search algorithm (HNSA) to solve engineering optimization problems with continuous design variables. Although the harmony search algorithm (HSA) has proven its ability of finding near global regions within a reasonable time, it is comparatively inefficient in performing local search. In this study sequential quadratic programming (SQP) is employed to speed up local search and improve precision of the HSA solutions. Moreover, an empirical study is performed in order to determine the impact of various parameters of the HSA on convergence behavior. Various benchmark engineering optimization problems are used to illustrate the effectiveness and robustness of the proposed algorithm. Numerical results reveal that the proposed hybrid algorithm, in most cases is more effective than the HSA and other meta-heuristic or deterministic methods.

© 2008 Elsevier B.V. All rights reserved.

Keywords: Heuristic; Harmony search algorithm; Hybridization; Sequential quadratic programming; Engineering optimization

1. Introduction

Evolutionary algorithms (EAs) are efficient at exploring entire search space; however, they are relatively poor at finding the precise optimum solution in the region to which the algorithm converges. Many researchers [1–3] have shown that EAs perform well for global searching due to their capability of quickly exploring and finding promising regions in the search space, but they take a relatively long time to converge to a local optimum.

On the other hand, gradient based algorithms are very effective deterministic methods in finding a stationary point near the initial starting point [1]. There exist many efficient gradient descent methods for finding local minima of a function, e.g. the steepest descent method, the Newton

method and the quasi Newton methods. In general, gradient based algorithms converge faster and they can obtain solutions with higher accuracy compared to stochastic approaches in fulfilling the local search task. They have been used widely in a large class of problems, especially as a vital component of sophisticated algorithms. However, these approaches often rely heavily on the initial starting point, the topology of the feasible region and the surface associated with the objective functions. A good starting point is vital for these methods to be executed successfully.

To obtain a more robust optimization technique, it is common to combine different search strategies trying to compensate deficiencies of the individual algorithms. During the last few years, new techniques have been developed in order to improve the lack of accuracy of the EAs, using local optimization algorithms. These techniques are based on combination of local optimization procedures, which are good at finding local optima (local exploiter), and global search methods (global explorer). These are commonly known as hybrid algorithms and have been successfully

* Corresponding author. Tel.: +98 (21)66405844; fax: +98 (21)66419736.

E-mail addresses: fesanghary@gmail.com (M. Fesanghary), mahdavi@ce.sharif.edu (M. Mahdavi), mminary2@uiuc.edu (M. Minary-Jolandan), alizadeh@aut.ac.ir (Y. Alizadeh).

used to solve a wide variety of problems [1,4] and experimental studies. The results show that hybrid methods search more efficiently and often find better solutions [5–8]. A more detailed and comparative study of these hybrid methods is given at references [9,10].

To improve the efficiency of the HSA, in our previous study [11], we used dynamic parameter adjusting in improvisation step (defined in Section 2.1.3) and discussed its impacts. The results of our work showed great improvement in convergence rate and quality of the HSA solutions. In this study we improve the efficiency of the HSA by incorporation of local search methods. This can be considered as an innovative type of hybridization of the HSA that has not yet been explored in the literature. In particular, we investigate the possibility of the use of SQP as a local optimizer. Moreover, different combination strategies in hybridization which are important in computational efficiency of the algorithm are discussed.

In this study, first, we present a brief overview of the HSA. Since there are not any precise recommendations for tuning the HSA parameters in the literature, an empirical study to determine the impact of different parameters of the algorithm on the solution quality and convergence behavior is performed. Finally the proposed hybrid method is described for engineering optimization problems with continuous design variables. Various standard benchmark engineering optimization examples including function minimization problems and structural optimization problems from the literature are also presented to show the effectiveness of the HHSA.

2. Hybrid harmony search algorithm

This section describes the proposed hybrid harmony search algorithm. First, a brief overview of the HSA is provided, and then an empirical study to determine the impact of parameters of the algorithm on evolution of the solution is performed. At the end of the section the hybrid strategy of the proposed HHSA is presented.

2.1. Harmony search algorithm

The harmony search algorithm which is a nature-inspired algorithm, mimicking the improvisation process of music players has been developed by Geem et al. [12]. The HSA is simple in concept, few in parameters, and easy in implementation. It has been successfully applied to various benchmark and real-world problems including traveling salesman problem [13], parameter optimization of river flood model [14], design of pipeline network [15,16], and design of truss structures [17]. The steps in the procedure of harmony search are shown in Fig. 1. They are as follows [18]:

- Step 1: Initialize the problem and algorithm parameters.
 Step 2: Initialize the harmony memory.
 Step 3: Improve a new harmony.

- Step 4: Update the harmony memory.
 Step 5: Check the stopping criterion.

These steps are described in the next five subsections.

2.1.1. Initialize the problem and algorithm parameters

In Step 1, the optimization problem is specified as follows:

$$\begin{aligned} & \text{Minimize} && f(\vec{x}) \\ & \text{subject to} && g_i(\vec{x}) \geq 0 \quad i = 1, 2, \dots, M, \\ & && h_j(\vec{x}) = 0 \quad j = 1, 2, \dots, P, \\ & && Lx_k \leq x_k \leq Ux_k \quad k = 1, 2, \dots, N, \end{aligned} \quad (1)$$

where $f(\vec{x})$ is the objective function, M is the number of inequality constraints and P is the number of equality constraints. x is the set of each decision variable x_i ; N is the number of decision variables. The lower and upper bounds for each decision variable are Lx_i and Ux_i respectively. The HSA parameters are also specified in this step. These 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 (NI), or stopping criterion. The harmony memory (**HM**) is a memory location where all the solution vectors (sets of decision variables) are stored. The **HM** is similar to the genetic pool in the genetic algorithms (GAs) [17]. Here, HMCR and PAR are parameters that are used to improve the solution vector. Both are defined in Step 3.

2.1.2. Initialize the harmony memory

In Step 2, the **HM** matrix is filled with as many randomly generated solution vectors as the HMS:

$$\mathbf{HM} = \begin{bmatrix} x_1^1 & x_2^1 & \dots & x_{N-1}^1 & x_N^1 \\ x_1^2 & x_2^2 & \dots & x_{N-1}^2 & x_N^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{\text{HMS}-1} & x_2^{\text{HMS}-1} & \dots & x_{N-1}^{\text{HMS}-1} & x_N^{\text{HMS}-1} \\ x_1^{\text{HMS}} & x_2^{\text{HMS}} & \dots & x_{N-1}^{\text{HMS}} & x_N^{\text{HMS}} \end{bmatrix}. \quad (2)$$

Infeasible solutions that violate the constraints have a chance to be included in the **HM** with hope of forcing the search towards the feasible solution area. Static penalty functions are used to calculate the penalty cost for an infeasible solution. The total cost for each solution vector is evaluated using:

$$\begin{aligned} \text{fitness}(\vec{x}) = & f(\vec{x}) + \sum_{i=1}^M \alpha_i \times \min[0, g_i(\vec{x})]^2 + \sum_{j=1}^P \beta_j \\ & \times \min[0, h_j(\vec{x})]^2, \end{aligned} \quad (3)$$

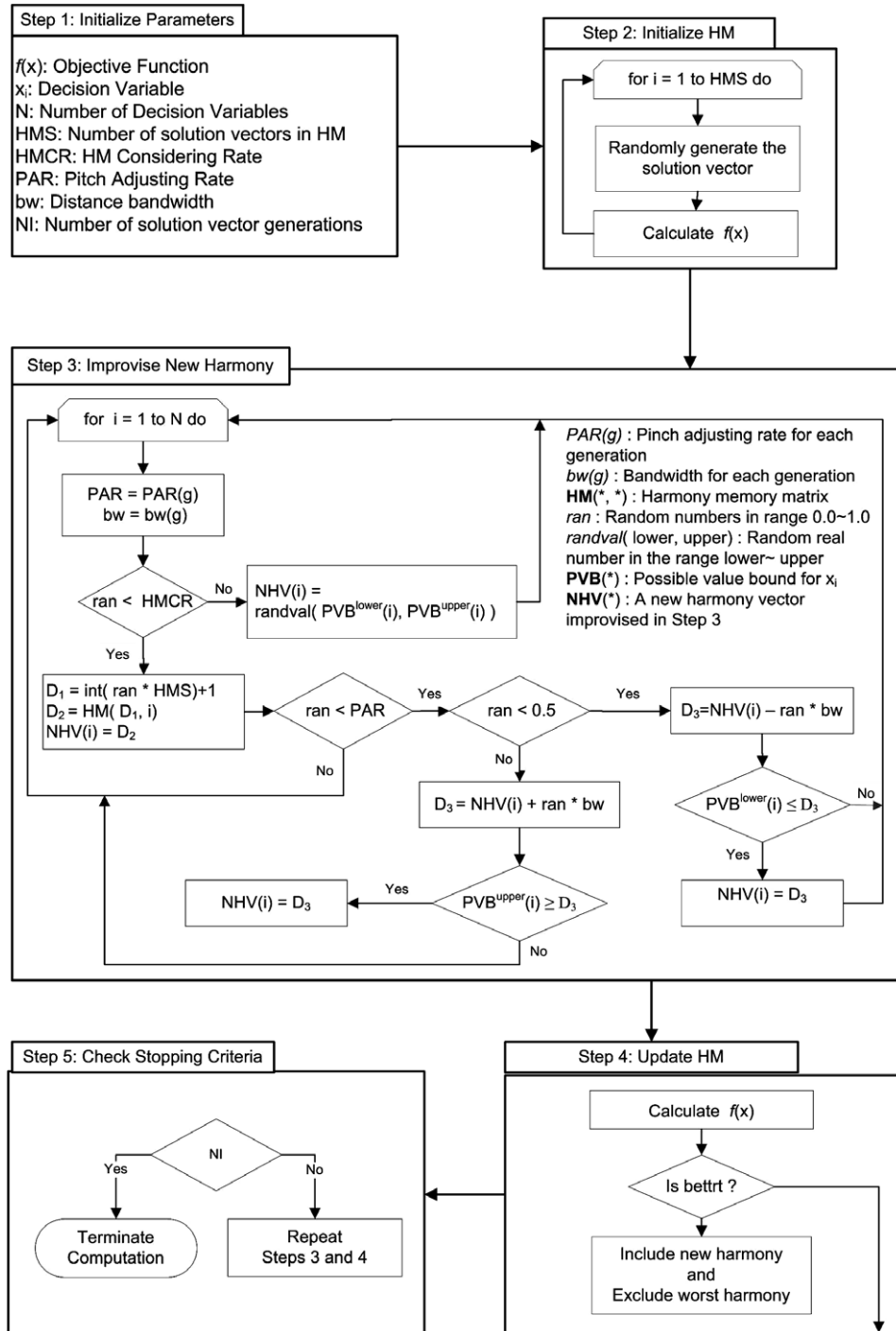


Fig. 1. Optimization procedure of the harmony search algorithm.

where α_i and β_j are the penalty coefficients. Generally, it is difficult to find a specific rule to determine the values of the penalty coefficients and normally these parameters remain problem-dependent.

2.1.3. Improvise a new harmony

A new harmony vector, $\vec{x}' = (x'_1, x'_2, \dots, x'_N)$, is generated based on three rules: (1) memory consideration, (2) pitch adjustment and (3) random selection. Generating a new

harmony is called ‘improvisation’ [18]. In the memory consideration, the value of the first decision variable (x'_1) for the new vector is chosen from any of the values in the specified **HM** range ($x_1^1 - x_1^{HMS}$). Values of the other decision variables (x'_2, x'_3, \dots, x'_N) are chosen in the same manner. The **HMC**R, which varies between 0 and 1, is the rate of choosing one value from the historical values stored in the **HM**, while (1-**HMC**R) is the rate of randomly selecting one value from the possible range of values, as shown in Eq. (4).

if($rand()$ < HMCR)

$$x'_i \leftarrow x'_i \in \{x_i^1, x_i^2, \dots, x_i^{HMS}\}$$

else

$$x'_i \leftarrow x'_i \in \mathbf{X}_i$$

end.

where $rand()$: is a uniform random number between 0 and 1 and \mathbf{X}_i is the set of the possible range of values for each decision variable, that is $Lx_i \leq \mathbf{X}_i \leq Ux_i$.

For example, a HMCR of 0.85 indicates that the HSA will choose the decision variable value from historically stored values in the **HM** with an 85% probability or from the entire possible range with a (100–85)% probability. Every component obtained by the memory consideration is examined to determine whether it should be pitch-adjusted. This operation uses the PAR parameter, which is the rate of pitch adjustment as follows:

if($rand()$ < PAR)

$$x'_i = x'_i \pm rand() * bw$$

else

$$x'_i = x'_i$$

end.

where bw, is an arbitrary distance bandwidth.

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

$$PAR(gn) = PAR_{\min} + \frac{(PAR_{\max} - PAR_{\min})}{NI} \times gn, \quad (6)$$

where $PAR(gn)$ is the pitch adjusting rate for each generation, PAR_{\min} is the minimum pitch adjusting rate, PAR_{\max} is the maximum pitch adjusting rate and gn is the generation number.

$$bw(gn) = bw_{\max} \exp(c \cdot gn),$$

$$c = \frac{\ln\left(\frac{bw_{\min}}{bw_{\max}}\right)}{NI}, \quad (7)$$

where $bw(gn)$ is the bandwidth for each generation, bw_{\min} is the minimum bandwidth and bw_{\max} is the maximum bandwidth.

Recently other variants of harmony search have been proposed. Omran and Mahdavi [19] proposed a new variant of harmony search, called the global best harmony search (GHS), in which concepts from swarm intelligence are borrowed to enhance the performance of HSA such that the new harmony can mimic the best harmony in the **HM**. Also, Geem [20] proposed a new stochastic derivative for discrete variables based on a harmony search algorithm to optimize problems with discrete variables and problems

in which the mathematical derivative of the function cannot be analytically obtained.

2.1.4. Update harmony memory

If the new harmony vector, $\vec{x}' = (x'_1, x'_2, \dots, x'_N)$, has better fitness function than the worst harmony in the **HM**, the new harmony is included in the **HM** and the existing worst harmony is excluded from the **HM**.

2.1.5. Check stopping criterion

The HSA is terminated when the stopping criterion (e.g. maximum number of improvisations) has been met. Otherwise, Steps 3 and 4 are repeated.

2.2. Empirical study of the impact of different HSA parameters on convergence behavior

The aim of this section is to study the evolution of the algorithm solution over generations under different settings of three important parameters: the pitch adjusting rate (PAR), harmony memory size (HMS), and harmony memory considering rate (HMCR).

Keeping that in mind, we will now show the effect of changes in single parameter. Particularly, we tested the following seven different scenarios as shown in Table 1. Each scenario was tested over 30 runs and maximum number of iterations is fixed to 100,000 for all runs. Values of bw_{\max} and bw_{\min} are set to 4 and 0.01 respectively. In Figs. 2–4 the average cost for welded beam design problem (defined in Section 3.3.1) is shown over generation number.

In Fig. 2, the effect of variation of the HMCR is shown. As mentioned earlier, the HMCR determines the rate of choosing one value from the historical values stored in the **HM**. The larger the HMCR is the less exploration is achieved; and the algorithm further relies on stored values in **HM** and this potentially leads to the algorithm getting stuck in a local optimum. On the other hand, choosing too small HMCR will decrease the algorithm efficiency and the HSA behaves like a pure random search, with less assistance from the historical memory. As seen in Fig. 2 large and small HMCR values lead to a decrease in the solution quality.

In Fig. 3 the solution evolution for different HMS is shown. We can see that decreasing the HMS leads to premature convergence and increasing the HMS leads to significant improvements in the initial phase of a run. Note

Table 1
Scenarios for ceteris paribus analysis of convergence behavior

| Scenario | HMCR | HMS | PAR_{\min} | PAR_{\max} |
|----------|------|-----|--------------|--------------|
| Standard | 0.6 | 5 | 0.45 | 0.90 |
| 2 | 0.9 | 5 | 0.45 | 0.90 |
| 3 | 0.1 | 5 | 0.45 | 0.90 |
| 4 | 0.6 | 1 | 0.45 | 0.90 |
| 5 | 0.6 | 10 | 0.45 | 0.90 |
| 6 | 0.6 | 5 | 0.05 | 0.45 |
| 7 | 0.6 | 5 | 0.05 | 0.90 |

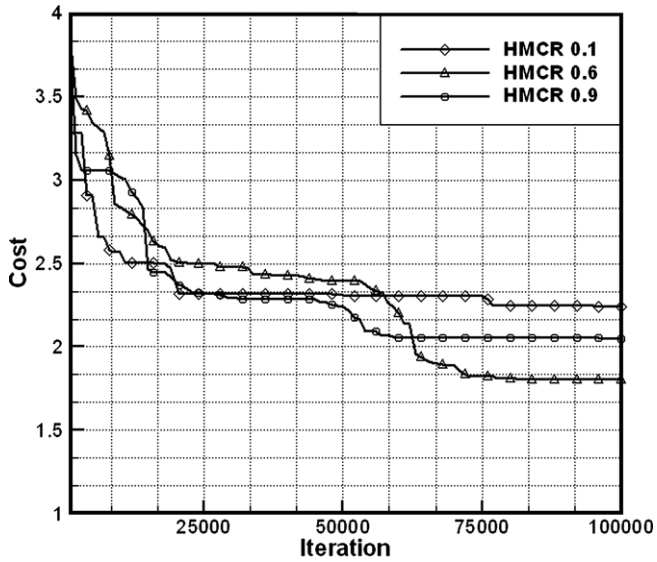


Fig. 2. Effects of HMCR on the convergence behavior.

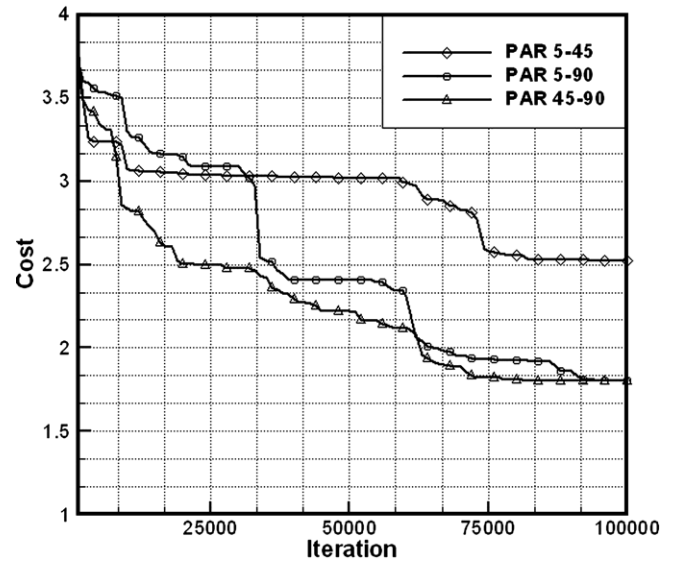


Fig. 4. Effects of PARs on the convergence behavior.

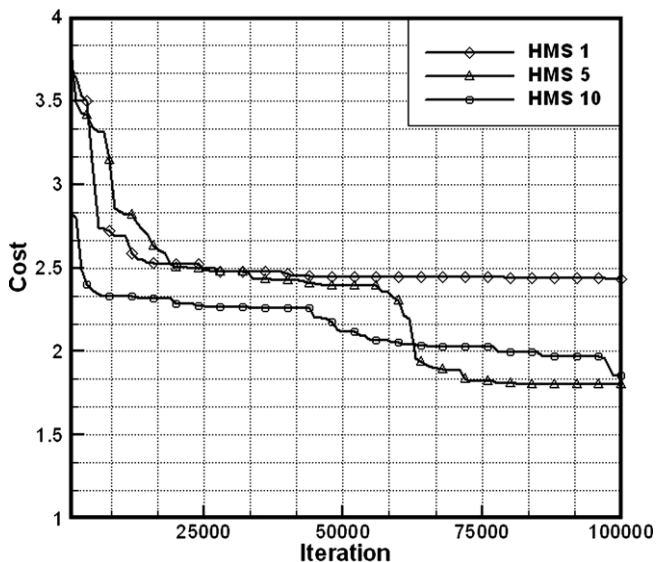


Fig. 3. Effects of HMS on the convergence behavior.

that when the time or the number of iterations is finite, increasing the HMS may deteriorate the quality of the solution. In general, the larger the HMS the more time (or iterations) is needed for the algorithm to find the optimal solution; but a higher quality is usually achieved.

Based on this study and the frequently used HMS values in other HSA applications available in the literature [11–16], it seems that for most moderate sized problems, a typical value for HMS is in the order of 4 to 10.

Finally, Fig. 4 shows the evolution of the solution quality over generations for different PARs. In final generations, which algorithm converged to the optimal solution vector, large PAR values with small bw values usually cause the improvement of the solutions [11]. As seen in the standard scenario and scenario 7 which have large

PAR_{max}, the result obtained by algorithm is better than those obtained by scenario 6 which have a small PAR_{max}. Although the standard scenario and scenario 7 produce the same results, the standard scenario due to smoother convergence is preferable.

2.3. Hybridization

As the HSA only uses the fitness function in the update harmony step (step 4.), it is a rather blind optimizer which does not use any auxiliary information such as derivatives or other specific knowledge about the special structure of the objective function. If there is such knowledge, however, it is unwise and inefficient not make use of it. Plenty of synergism lies in the combination of the HSA and local search methods.

The particular combination is important in terms of possible solution quality and computational efficiency. We need to find the right balance between local exploitation and global exploration. There are different combination strategies; for example in the first strategy the HSA performs a coarse search in the first stage. When the HSA is completed or shows a negligible trend of improvement after many iterations, the local optimizer begins its task and uses the best vectors from the harmony memory (HM) as the starting point. In the second strategy, we can run both methods simultaneously: All new improvised vectors are continuously used as initial values for the local method. If the locally optimized vectors have a better fitness value than those in the HM, they will be re-implanted into the HM.

In order to obtain a high quality solution with minimum resource requirement we combine two above mentioned strategies with a probability of P_c . In this way the local method is integrated into the HSA as illustrated in Fig. 5. For instance, in each iteration, with probability

```

Procedure HHS algorithm {
  Initiate parameters;
  Repeat {
    Construct a new vector; (see 2.1.3)
    for each new vector{
      Calculate the fitness;
      if ( rand() < Pc ) then Improve the vector by applying the local search;
      Update harmony memory (if applicable);
    }
  } until a pre-specified stopping criteria is met;
  for each vector in harmony memory{
    Improve the vector by applying the local search;
    Update the best found solution (if applicable);
  }
}

```

Fig. 5. The hybrid harmony search algorithm.

equal to P_c , the local optimizer which uses the new improvised vector as its starting point is applied. The **HM** is updated if the locally optimized vector has better fitness value than those in the **HM**. Finally, after the specified termination criteria for the HSA are reached, the local optimizer is then applied using the final best solutions obtained by the HSA (vectors stored in the **HM**); and optimizes them. The use of the **HM** allows the selection of the best vectors that may represent different regions in the search space. In this way, the optimized vectors are more likely to converge towards different local optima.

In order to improve the performance of the hybrid algorithm, the computation time spent by the global search and local search should be adjusted. In many problems, it is not efficient to carry out a local optimization algorithm for every solution vector due to the dimensions of the search space and limitations of computational resources. This is especially important when the local search algorithm have a high computational cost. If we apply the optimization algorithm to every new improvised vector, we will obtain many similar solutions with a considerable waste of time. Therefore, in our approach we apply the local search method to only a few solution vectors.

In this strategy P_c will vary in an attempt to control the allocation of resources between the global search and the local search. As P_c increases, the local search consumes more time, but in general, hybrid method is expected to produce better results. When $P_c = 1$, the strategy is similar to the second hybrid strategy; and when P_c is zero, the strategy is similar to the first strategy. The P_c parameter must be fixed by the user before the optimization process begins. Based on our experience we recommend a fairly small value for P_c (e.g. $P_c = 0.1$). One of the main advantages of this strategy is that the computational cost of applying the local optimization algorithm to only a few solution vectors, hardly affects the total time spent by the algorithm.

There are many efficient nonlinear programming methods such as Davidon-Fletcher-Powell (DFP), SQP and BFGS that can be used as a local optimizer. In this work the SQP, is used as local optimizer, due to its better performance over other nonlinear programming methods in terms of efficiency, accuracy, and percentage of successful

solutions over a vast number of test problems [21]. In this study, DONLP2 [22] a general nonlinear constraint-handling package, which is an implementation of the SQP method, was used.

3. Examples

Several examples taken from the optimization literature will be used to demonstrate the function and capability of the proposed method. These examples have been previously solved using a variety of other techniques (both evolutionary and traditional mathematical programming methods), which is useful to determine the quality of the solutions produced by the proposed approach. The simulations were carried out on a Pentium III 850-MHz processor. For all examples presented in this paper, the HHS parameters were set the same as the standard scenario with $P_c = 0.1$.

3.1. Unconstrained function minimization examples

3.1.1. Goldstein and Price function I (with four local minima)

The problem can be stated as follows:

$$\begin{aligned} \text{Minimize } f(\vec{x}) = & \{1 + (x_1 + x_2 + 1)^2(19 - 14x_1 \\ & + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\} \\ & \times \{30 + (2x_1 - 3x_2)^2(18 - 32x_1 \\ & + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\}. \quad (8) \end{aligned}$$

This function is an eighth-order polynomial in two variables. As shown in Fig. 6, however, the function has four local minima, one of which is global, as follows [18]:

$f(1.2, 0.8) = 840.0$, $f(1.8, 0.2) = 84.0$, $f(-0.6, -0.4) = 30.0$ and $f^*(0, -1.0) = 3.0$ (global minimum). In this example, the bounds for two design variables (x_1 and x_2) were set

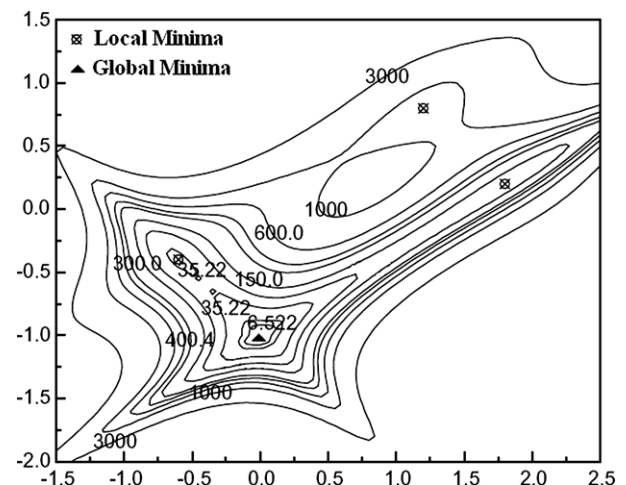


Fig. 6. Goldstein and Price function I.

between -50 and 50 . After 2400 iterations, the best solution vector obtained by the HHSA was $x = (0.0000000872, -1.0000000011)$ with a corresponding function value of $f(x) = 3.000000000$. The HHSA solution is very close to the global optimum value.

3.1.2. Goldstein and Price function II (with many local minima)

Minimize

$$f(\vec{x}) = \exp \left\{ \frac{1}{2} (x_1^2 + x_2^2 - 25)^2 \right\} + \sin^4(4x_1 - 3x_2) + \frac{1}{2} (2x_1 + x_2 - 10)^2. \quad (9)$$

The objective function $f(\vec{x})$ has a minimum solution at $x = (3, 4)$ with a corresponding function value of $f(\vec{x}) = 1.0$, as shown in Fig. 7. The two design variables x_1, x_2 were initially bounded between -50 and 50 . The algorithm found the best solution at $x^* = (3.00000, 4.00000)$ with the corresponding objective function value of $f^*(x) = 1.00000$.

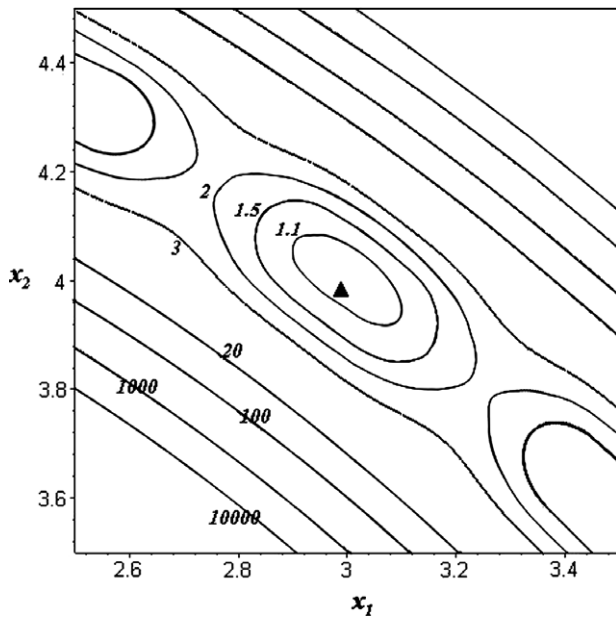


Fig. 7. Goldstein and Price function II.

3.2. Constrained function minimization examples

3.2.1. Constrained function I: (Himmelblau's nonlinear optimization problem)

This problem was originally proposed by Himmelblau [23], and it has been used before as a benchmark for several GA-based techniques [24–26]. In this problem there are five design variables (x_1, x_2, x_3, x_4, x_5), six nonlinear inequality constraints and 10 boundary conditions. The problem can be stated as follows:

Minimize $f(\vec{x}) = 5.3578547x_3^2 + 0.835689x_1x_5 + 37.293239x_1 - 40792.141$ (10)

Subject to $g_1(\vec{x}) = 85.334407 + 0.0056858x_2x_5 + 0.00026x_1x_4 - 0.0022053x_3x_5$, (11)

$g_2(\vec{x}) = 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 - 0.0021813x_3^2$, (12)

$g_3(\vec{x}) = 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 - 0.0019085x_3x_4$, (13)

$0 \leq g_1(\vec{x}) \leq 92$, (14)

$90 \leq g_2(\vec{x}) \leq 110$, (15)

$20 \leq g_3(\vec{x}) \leq 25$, (16)

$78 \leq x_1 \leq 102$, (17)

$33 \leq x_2 \leq 45$, (18)

$27 \leq x_i \leq 45 \quad (i = 3, \dots, 5)$. (19)

The optimal solution is obtained at $x^* = (78, 33, 27.085149, 45, 44.925329)$ with corresponding function value equal to $f^*(x) = -31024.3166$ after approximately 28000 function evaluations. This vector was obtained after 1.306 s. Table 2 compares the best solution of example 3.2.1 obtained using the HHSA with previous best solutions reported by Homaifar et al. [26], Shi and Eberhart [27], Coello [25], Deb [28], and Lee and Geem [18]. Homaifar et al., Coello and Deb using the GA-based methods obtained a best solution function value of $f(x) = -30005.7$, $f(x) = -31020.859$ and $f(x) = -30665.500$, respectively. Shi and Eberhart [27] solved the problem using a modified particle swarm optimization algorithm, and obtained a best solution of $f(x) = -31025.561$, (the constraint g_1 and g_3 was active on that solution). Lee and Geem [18] solved the problem using the HSA and obtained a best solution of $f(x) = -30665.500$ after 65,000 iterations.

Table 2
Optimal results for Himmelblau's nonlinear optimization problem (N/A not available)

| Methods | Optimal design variables (x) | | | | | Optimal solution |
|-----------------------|------------------------------|--------|----------|-------|----------|------------------|
| | x_1 | x_2 | x_3 | x_4 | x_5 | |
| Homaifar et al. [26] | 80.39 | 35.07 | 32.05 | 40.33 | 33.34 | -30005.700 |
| Coello [25] | 78.0495 | 33.007 | 27.081 | 45.00 | 44.94 | -31020.859 |
| Shi and Eberhart [27] | 78.0 | 33.0 | 27.07099 | 45.0 | 44.969 | -31025.561 |
| Deb [28] | N/A | N/A | N/A | N/A | N/A | -30665.500 |
| Lee and Geem [18] | 78.0 | 33.0 | 29.995 | 45.0 | 36.776 | -30665.500 |
| Present study | 78.0 | 33.0 | 27.08515 | 45.0 | 44.92533 | -31024.316 |

It is obvious from the Table 2 that the result obtained using HHSA is better than the best feasible solution previously reported.

3.2.2. Constrained function II

This problem has seven design variables, four nonlinear inequality constraints and fourteen boundary conditions. The problem can be stated as follows:

$$\begin{aligned} \text{Minimize } f(\vec{x}) &= (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 \\ &+ 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 \\ &+ x_7^4 - 4x_6x_7 - 10x_6 - 8x_7 \end{aligned} \quad (20)$$

$$\text{Subject to } g_1(\vec{x}) = 127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 \geq 0, \quad (21)$$

$$g_2(\vec{x}) = 282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 \geq 0, \quad (22)$$

$$g_3(\vec{x}) = 196 - 23x_1 - x_2^2 - 6x_6^2 + 8x_7 \geq 0, \quad (23)$$

$$g_4(\vec{x}) = -4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 \geq 0, \quad (24)$$

$$-10 \leq x_i \leq 10 \quad (i = 1, \dots, 7). \quad (25)$$

After approximately 42,000 function evaluations which took 1.873 s the optimal solution was obtained at $x^* = (2.33047, 1.95137, -0.47772, 4.36574, -0.62448, 1.03794, 1.59414)$ with the corresponding function value equal to $f^*(x) = 680.6300577$. No constraints are active for this solution. Table 3 compares the best solution of example 3.2.2 obtained using the HHSA with the previous best solutions reported by Lee and Geem [18], Deb [28] and Michalewicz [29]. It can be seen that the results obtained using the HHSA, is more optimized than any other earlier solutions reported in the literature.

3.3. Structural engineering optimization examples

3.3.1. Welded beam design

The welded beam structure, shown in Fig. 8, is a practical design problem that has been often used as a benchmark for testing different optimization methods [30–34].

Table 3
Optimal results for constrained function II (N/A not available)

| Optimal design variables (x) and f(x) | Michalewicz [29] | Deb [28] | Lee and Geem [18] | Present study |
|---------------------------------------|------------------|----------|-------------------|---------------|
| x_1 | N/A | N/A | 2.32345617 | 2.33047 |
| x_2 | N/A | N/A | 1.951242 | 1.95137 |
| x_3 | N/A | N/A | 0.448467 | 0.47772 |
| x_4 | N/A | N/A | 4.3619199 | 4.36574 |
| x_5 | N/A | N/A | 0.630075 | 0.62448 |
| x_6 | N/A | N/A | 1.03866 | 1.03794 |
| x_7 | N/A | N/A | 1.605384 | 1.59414 |
| f(x) | 680.642 | 680.6344 | 680.641357 | 680.630057 |

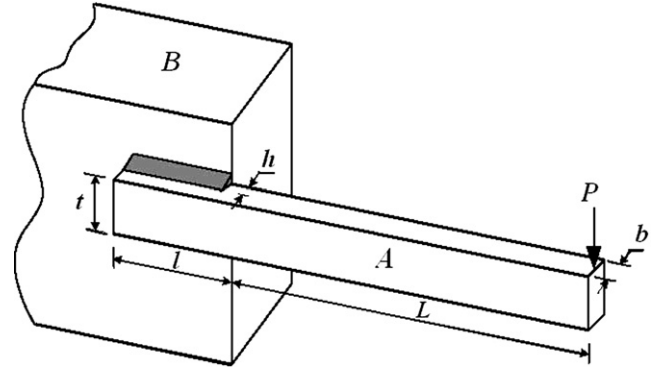


Fig. 8. Welded beam structure.

The objective is to find the minimum fabricating cost of the welded beam subject to constraints on shear stress (τ), bending stress (σ), buckling load (P_C), end deflection (δ), and side constraint. There are four design variables: h ($=x_1$), l ($=x_2$), t ($=x_3$) and b ($=x_4$). The mathematical formulation of the objective function $f(\vec{x})$, which is the total fabricating cost mainly comprised of the set-up, welding labor, and material costs, is as follows:

$$\text{Minimize } f(\vec{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2) \quad (26)$$

$$\text{Subject to } g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \leq 0, \quad (27)$$

$$g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \leq 0, \quad (28)$$

$$g_3(\vec{x}) = x_1 - x_4 \leq 0, \quad (29)$$

$$g_4(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \leq 0, \quad (30)$$

$$g_5(\vec{x}) = P - P_C(\vec{x}) \leq 0, \quad (31)$$

$$0.125 \leq x_1 \leq 0, \quad 0.1 \leq x_2, \quad x_3 \leq 10, \quad 0.1 \leq x_4 \leq 5, \quad (32)$$

where

$$\begin{aligned} \tau(\vec{x}) &= \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}, \\ \tau' &= \frac{P}{\sqrt{2}x_1x_2}, \quad \tau'' = \frac{MR}{J}, \quad M = P\left(L + \frac{x_2}{2}\right), \\ R &= \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \\ J &= 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}, \end{aligned} \quad (33)$$

$$\sigma(\vec{x}) = \frac{6PL}{x_4x_3^2}, \quad \delta(\vec{x}) = \frac{6PL^3}{E x_3^3 x_4}$$

$$P_C(\vec{x}) = \frac{4.013E\sqrt{x_3^2x_4^6/36}}{L^2} \left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right).$$

$$P = 6000 \text{ lb}, \quad L = 14 \text{ in.}, \quad \delta_{\max} = 0.25 \text{ in.},$$

$$E = 30 \times 10^6 \text{ psi}, \quad G = 12 \times 10^6 \text{ psi}$$

$$\tau_{\max} = 13,600 \text{ psi}, \quad \sigma_{\max} = 30,000 \text{ psi}.$$

Deb [28,33] and Coello [25,34] solved this problem using GA-based methods. Radgsdell and Phillips [32] compared optimal results of different optimization methods that were

mainly based on mathematical optimization algorithms. These methods, are APPROX (Griffith and Stewart’s successive linear approximation), DFP, SIMPLEX (Simplex method with a penalty function), and RANDOM (Richardson’s random method) algorithms. Lee and Geem [18] solved this problem using the HSA and Mahdavi et al. [11] solved it using IHS method. The comparison of results, are shown in Table 4.

The HNSA result, which was obtained after approximately 90,000 function evaluations (4.138 s), was the same as those reported by Mahdavi et al. [11]. Note that the approach proposed by Lee and Geem [18] required 110,000 evaluations of the fitness function to produce the result shown in Table 4. The approach of Mahdavi et al. [11] required 200,000 evaluations of the fitness function. In contrast, the HNSA requires only 90,000 evaluations of the fitness function.

3.3.2. Design of a 10-bar plane truss

Consider the ten-bar plane truss shown in Fig. 9, taken from Lee and Geem [18]. The problem is to find the cross-sectional area of each member of this truss, such that its weight ($f(\vec{x})$) is minimized subject to stress and displacement constraints. The weight of the truss is given by

$$f(\vec{x}) = \sum_{j=1}^{10} \rho A_j L_j, \tag{34}$$

where A_j is the cross-sectional area of the j th member, L_j is the length of the j th member, and ρ is the weight density of the material.

This problem was previously analyzed by Lee and Geem [18], Schmit and Miura [35], Venkayya [36], Rizzi [37], and Khan and Willmert [38]. The assumed data are modulus of elasticity, $E = 1.0 \times 10^4$ ksi (68965.5 MPa), density, $\rho = 0.10$ lb/in.³ (2768.096 kg/m³), and a load of 100 kips (45351.47 kg) in the negative y -direction applied at nodes 2 and 4. The maximum allowable stress of each member is σ_a , and assumed to be ± 25 ksi (172.41 MPa). The maximum allowable displacement of each node (horizontal and vertical) is represented by u_a , and assumed to be 2 in

Table 4
Optimal results for welded beam design (N/A not available)

| Methods | Optimal design variables (x) | | | | Cost |
|----------------------------|----------------------------------|---------|---------|---------|--------|
| | h | l | t | b | |
| Coello [34] | N/A | N/A | N/A | N/A | 1.8245 |
| Coello [25] | 0.2088 | 3.4205 | 8.9975 | 0.2100 | 1.7483 |
| Lee and Geem [18] | 0.2442 | 6.2231 | 8.2915 | 0.2443 | 2.3807 |
| Ragsdell and Phillips [32] | | | | | |
| APPROX | 0.2444 | 6.2189 | 8.2915 | 0.2444 | 2.3815 |
| DFP | 0.2434 | 6.2552 | 8.2915 | 0.2444 | 2.3841 |
| SIMPLEX | 0.2792 | 5.6256 | 7.7512 | 0.2796 | 2.5307 |
| RANDOM | 0.4575 | 4.7313 | 5.0853 | 0.6600 | 4.1185 |
| Deb [33] | N/A | N/A | N/A | N/A | 2.38 |
| Deb [28] | 0.2489 | 6.1730 | 8.1789 | 0.2533 | 2.4328 |
| Mahdavi et al. [11] | 0.20573 | 3.47049 | 9.03662 | 0.20573 | 1.7248 |
| Present study | 0.20572 | 3.47060 | 9.03682 | 0.20572 | 1.7248 |

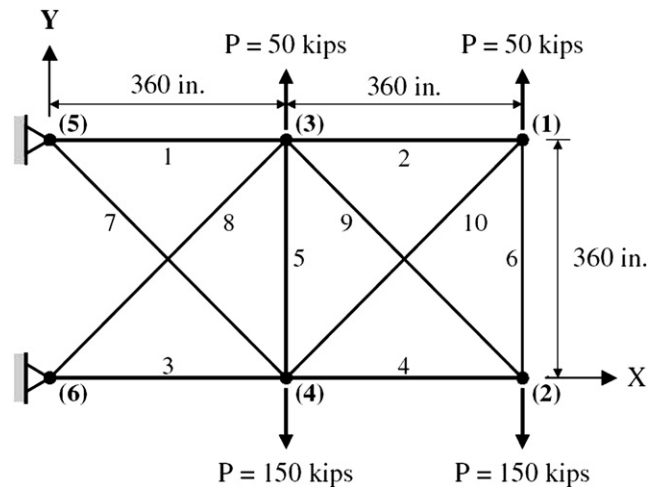


Fig. 9. Ten-bar plane truss.

(5.08 cm). A minimum cross-sectional area of 0.1 in.² was enforced. There are 10 stress constraints, and 12 displacement constraints. The cross-sectional area of each element can be different, thus the problem has 10 design variables.

Table 5 shows the best solution vector and also provides a comparison between the optimal design results reported in the literature and those obtained in the present study. After approximately 22,000 function evaluations which took 8.814 s, the best obtained solution was $f^*(x) = 4668.72$, which is better than any of the solutions produced by the other techniques. It should be noted that HSA[18] requires 50,000 fitness function evaluations to produce the results presented in Table 5.

3.3.3. Design of a four-storey, two-bay steel frame

As the final example, a four-storey, two-bay frame is considered. The frame has 15 nodes and 20 elements. The frame is modeled using I-beam elements and the elements are grouped into five different dimensional sets. The frame dimensions, configuration, loading, and grouping of the members are shown in Fig. 10. The objective is to minimize the weight of the frame through finding the optimum cross-section dimensions b , h , t_w and t_f for each group of elements. Since four design variables are considered for each group, 20 sizing variables are considered. The material density is 7850 kg/m³ and the Young’s modulus is 210 kN/mm². Maximum allowable stress is 1600 MPa and maximum top storey sway is limited to 0.02 m. The geometric constraints are as follows:

$$\begin{aligned}
 0.07 \text{ m} &\leq h \leq 0.6 \text{ m} \\
 0.06 \text{ m} &\leq b \leq 0.3 \text{ m} \\
 0.007 \text{ m} &\leq t_f \leq 0.025 \text{ m} \\
 0.004 \text{ m} &\leq t_w \leq 0.025 \text{ m} \\
 0.3 &\leq b/h \leq 1 \\
 0.03 &\leq t_w/b \leq 0.1 \\
 0.03 &\leq t_f/h \leq 0.1.
 \end{aligned} \tag{35}$$

Table 5
Optimal results of the ten-bar plane truss

| Methods | Optimal cross-sectional areas A (in. ²) | | | | | | | | | | Weight (lb) |
|------------------------|---|-------|-------|-------|-------|-------|-------|-------|-------|----------|-------------|
| | A_1 | A_2 | A_3 | A_4 | A_5 | A_6 | A_7 | A_8 | A_9 | A_{10} | |
| Lee and Geem [18] | 23.25 | 0.102 | 25.73 | 14.51 | 0.100 | 1.977 | 12.21 | 12.61 | 20.36 | 0.100 | 4668.81 |
| Schmit and Miura [35] | | | | | | | | | | | |
| NEW SUMT | 23.55 | 0.100 | 25.29 | 14.36 | 0.100 | 1.970 | 12.39 | 12.81 | 20.34 | 0.100 | 4676.96 |
| CONMIN | 23.55 | 0.176 | 25.20 | 14.39 | 0.100 | 1.967 | 12.40 | 12.86 | 20.41 | 0.100 | 4684.11 |
| Venkayya [36] | 25.19 | 0.363 | 25.42 | 14.33 | 0.417 | 3.144 | 12.08 | 14.61 | 20.26 | 0.513 | 4895.60 |
| Rizzi [37] | 23.53 | 0.100 | 25.29 | 14.37 | 0.100 | 1.970 | 12.39 | 12.83 | 20.33 | 0.100 | 4676.92 |
| Khan and Willmert [38] | 24.72 | 0.100 | 26.54 | 13.22 | 0.108 | 4.835 | 12.66 | 13.78 | 18.44 | 0.100 | 4792.52 |
| Present Study | 23.31 | 0.100 | 24.63 | 14.59 | 0.100 | 1.967 | 12.49 | 12.94 | 20.51 | 0.100 | 4668.72 |

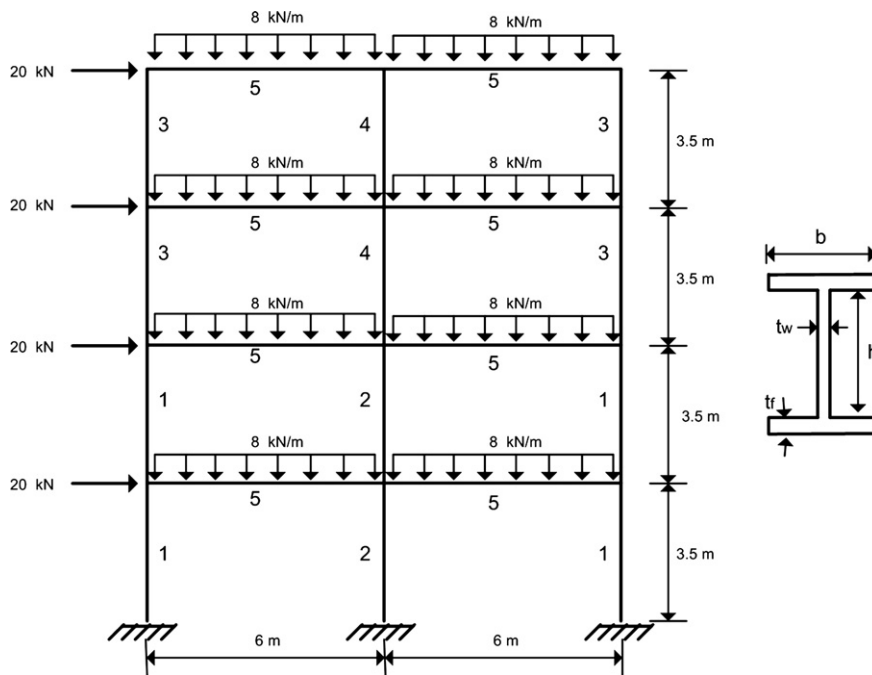


Fig. 10. Four-storey, two-bay steel frame.

Table 6
Optimal results for the four-storey, two-bay steel frame using HHSA

| Group no. | Optimal design variables | | | |
|-----------|--------------------------|---------|---------|---------|
| | h | t_r | t_w | b |
| 1 | 0.36172 | 0.00700 | 0.00654 | 0.16133 |
| 2 | 0.41582 | 0.01194 | 0.00889 | 0.19892 |
| 3 | 0.24644 | 0.00936 | 0.00740 | 0.14667 |
| 4 | 0.33538 | 0.01141 | 0.00911 | 0.18888 |
| 5 | 0.49117 | 0.00791 | 0.00484 | 0.16102 |

Total weight: 3733.9 kg
Maximum stress: 56.6 MPa
Maximum top storey sway: 0.0199 m
Run-time: 28.249 s

Table 7
Optimal results for the four-storey, two-bay steel frame using HSA

| Group no. | Optimal design variables | | | |
|-----------|--------------------------|---------|---------|---------|
| | h | t_r | t_w | b |
| 1 | 0.36164 | 0.01011 | 0.00654 | 0.16133 |
| 2 | 0.41575 | 0.01194 | 0.00889 | 0.19892 |
| 3 | 0.24635 | 0.00936 | 0.00740 | 0.14668 |
| 4 | 0.33532 | 0.01140 | 0.00911 | 0.18888 |
| 5 | 0.49095 | 0.00791 | 0.00484 | 0.16102 |

Total weight: 3845.2 kg
Maximum stress: 49.8 MPa
Maximum top storey sway: 0.0191 m
Run-time: 36.110 s

To show the efficiency of the hybrid method this example has been solved using both HSA and HHSA methods. Table 6 gives the best solution vectors and the corresponding

weight using HHSA. An optimal structural weight of 3733.9 kg was achieved. The optimum results obtained after approximately 41,000 fitness function evaluations.

The maximum stress and maximum top storey sway are 56.6 MPa and 0.0199 m, respectively. The HSA results, which are obtained after 52,000 fitness function evaluations, are also shown in Table 7. The results indicate that the hybrid approach produces better result in comparison with HSA.

4. Conclusions

In the present study, a new simple but effective and efficient method that combines the two powerful search algorithms namely the sequential quadratic programming and the harmony search algorithm was proposed. The proposed hybrid algorithm employs the HSA to provide a near global search region and simultaneously with a small probability uses SQP for local search. When the specified termination criteria of the HSA are reached, the local search (SQP) is applied to vectors stored in the HM as a fine tuning to determine the optimal solution at the final step. An empirical study to determine the impact of different parameters of the HSA on the solution quality and convergence behavior was performed and the results were discussed. The ability of the algorithm was demonstrated using several test problems and its performance was compared with other conventional methods. The results reveal that the hybrid approach is able to obtain good results not only in terms of the quality of the obtained solutions but also in terms of the required number of fitness function evaluations.

Acknowledgements

The authors gratefully acknowledge Mohsen Rezapour, Ahmad Moradi, Mohammadreza Abdolvand and Morteza Damanafshan for their valuable suggestions and discussions.

References

- [1] C.R. Houck, J.A. Joines, M.G. Kay, Comparison of genetic algorithms, random start, and two-opt switching for solving large location–allocation problems, *Comput. Operat. Res.* 23 (6) (1996) 587–596.
- [2] C.R. Houck, J.A. Joines, J.R. Wilson, Empirical investigation of the benefits of partial Lamarckianism, *Evol. Comput.* 5 (1) (1997) 31–60.
- [3] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolutionary Programming*, Springer-Verlag, New York, 1994.
- [4] J.A. Joines, M.G. Kay, Utilizing hybrid genetic algorithms, in: R. Sarker, M. Mahamurdian, X. Yao (Eds.), *Evolutionary Optimization*, Kluwer, Norwell, MA, 2002.
- [5] P. Moscato, On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms, California Institute of Technology, Pasadena, Technical Report 826, 1989.
- [6] P. Moscato, C. Cotta, A gentle introduction to memetic algorithms, in: F. Glover, G. Kochenberger (Eds.), *Handbook of Metaheuristics*, Kluwer, Norwell, MA, 1999.
- [7] H. Bersini, B. Renders, Hybridizing genetic algorithms with hill-climbing methods for global optimization: two possible ways, in: *Proceedings of the IEEE International Symposium Evolutionary Computation*, Orlando, FL, 1994.
- [8] C.J. Merz, A principal components approach to combining regression estimates, *Mach. Learn.* 36 (1) (1999) 9–32.
- [9] J. He, J. Xu, X. Yao, Solving equations by hybrid evolutionary computation techniques, *IEEE Trans. Evol. Comput.* 4 (3) (2000) 295–304.
- [10] R. Sarker, M. Mohammadian, X. Yao (Eds.), *Evolutionary Optimization*, Kluwer, Norwell, MA, 2002.
- [11] M. Mahdavi, M. Fesanghary, E. Damangir, An improved harmony search algorithm for solving optimization problems, *Appl. Math. Comput.* 188 (2007) 1567–1579.
- [12] Z.W. Geem, J.H. Kim, G.V. Loganathan, A new heuristic optimization algorithm: harmony search, *Simulation* 76 (2) (2001) 60–68.
- [13] Z.W. Geem, C. Tseng, Y. Park, Harmony search for generalized orienteering problem: best touring in China, *Springer Lecture Notes Comput. Sci.* 3412 (2005) 741–750.
- [14] J.H. Kim, Z.W. Geem, E.S. Kim, Parameter estimation of the nonlinear muskingum model using harmony search, *J. Am. Water Resour. Assoc.* 37 (5) (2001) 1131–1138.
- [15] Z.W. Geem, J.H. Kim, G.V. Loganathan, Harmony search optimization: application to pipe network design, *Int. J. Model. Simulat.* 22 (2) (2002) 125–133.
- [16] Z.W. Geem, Optimal cost design of water distribution networks using harmony search, *Engrg. Optim.* 38 (3) (2006) 259–280.
- [17] K.S. Lee, Z.W. Geem, A new structural optimization method based on the harmony search algorithm, *Comput. Struct.* 82 (9–10) (2004) 781–798.
- [18] K.S. Lee, Z.W. Geem, A new meta-heuristic algorithm for continues engineering optimization: harmony search theory and practice, *Comput. Methods Appl. Mech. Engrg.* 194 (2004) 3902–3933.
- [19] M.G.H. Omran, M. Mahdavi, Global-best harmony search, *Appl. Math. Comput.* (2007), doi:10.1016/j.amc.2007.09.004.
- [20] Z.W. Geem, Novel derivative of harmony search algorithm for discrete design variables, *Appl. Math. Comput.*, doi:10.1016/j.amc.2007.09.049.
- [21] P.T. Boggs, J.W. Tolle, Sequential quadratic programming, *Acta Numer.* 4 (1995) 1–52.
- [22] P. Spellucci, An SQP method for general nonlinear programs using only equality constrained subproblems, *Math. Program.* 82 (1998) 413–448.
- [23] D.M. Himmelblau, *Applied Nonlinear Programming*, McGraw-Hill, New York, 1972.
- [24] M. Gen, R. Cheng, *Genetic Algorithms & Engineering Design*, Wiley, New York, 1997.
- [25] C.A.C. Coello, Use of a self-adaptive penalty approach for engineering optimization problems, *Comput. Ind.* 41 (2) (2000) 113–127.
- [26] A. Homaiifar, S.H.-V. Lai, X. Qi, Constrained optimization via genetic algorithms, *Simulation* 62 (4) (1994) 242–254.
- [27] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: *Proceedings of the International Congress on Evolutionary Computation 1998*, IEEE Service Center, Piscataway, NJ, 1998.
- [28] K. Deb, An efficient constraint handling method for genetic algorithms, *Comput. Methods Appl. Mech. Engrg.* 186 (2000) 311–338.
- [29] Z. Michalewicz, Genetic algorithms, numerical optimization, and constraints, in: L. Esheman (Ed.), *Proceedings of the Sixth International Conference on Genetic Algorithms*, Morgan Kaufman, San Mateo, 1995.
- [30] G.V. Reklaitis, A. Ravindran, K.M. Ragsdell, *Engineering Optimization Methods and Applications*, Wiley, New York, 1983.
- [31] J.N. Siddall, *Analytical Decision-Making in Engineering Design*, Prentice-Hall, New Jersey, 1972.
- [32] K.M. Ragsdell, D.T. Phillips, Optimal design of a class of welded structures using geometric programming, *ASME J. Engrg. Ind. Ser. B* 98 (3) (1976) 1021–1025.
- [33] K. Deb, Optimal design of a welded beam via genetic algorithms, *AIAA J.* 29 (11) (1991) 2013–2015.
- [34] C.A.C. Coello, Constraint-handling using an evolutionary multiobjective optimization technique, *Civil Engrg. Environ. Syst.* 17 (2000) 319–346.
- [35] L.A. Schmit, J.H. Miura, *Approximation Concepts for Efficient Structural Synthesis*, NASA CR-2552, NASA, Washington, 1976.

- [36] V.B. Venkayya, Design of optimum structures, *Comput. Struct.* 1 (1–2) (1971) 265–309.
- [37] P. Rizzi, Optimization of multi-constrained structures based on optimality criteria, in: *Conference on AIAA/ASME/SAE 17th Structures, Structural Dynamics, and Materials*, King of Drussia, Pennsylvania, 1976.
- [38] M.R. Khan, K.D. Willmert, W.A. Thornton, An optimality criterion method for large-scale structures, *AIAA J.* 17 (7) (1979) 753–761.