Hybrid Algorithm of Harmony Search, Particle Swarm and Ant Colony for Structural Design Optimization

A. Kaveh[1] and S. Talatahari[2]

Abstract. This chapter considers the implementation of the heuristic particle swarm ant colony optimization (HPSACO) methodology to find an optimum design of different types of structures. HPSACO is an efficient hybridized approach based on the harmony search scheme, particle swarm optimizer, and ant colony optimization. HPSACO utilizes a particle swarm optimization with a passive congregation algorithm as a global search, and the idea of ant colony approach worked as a local search. The harmony search-based mechanism is used to handle the variable constraints. In the discrete HPSACO, agents are allowed to select discrete values from the permissible list of cross sections. The efficiency of the HPSACO algorithm is investigated to find an optimum design of truss structures with continuous or discrete search domains and for frame structures with a discrete search domain. The results indicate that the HPSACO is a quite effective algorithm to find the optimum solution of structural optimization problems with continuous or discrete variables.

1 Introduction

Structural design optimization is a critical and challenging activity that has received considerable attention in the last two decades [1]. A high number of design variables, largeness of the search space and controlling a great number of design constraints are major preventive factors in performing optimum design in a reasonable time. Despite these facts, designers and owners have always desired to have optimal structures [2]. Therefore, different methods of structural optimization have been introduced which can be categorized in two general groups: classical methods and heuristic approaches.

Classical optimization methods are often based on mathematical programming. Many of these methods require substantial gradient information, and final results depend on the initially selected points. The number of computational operations increases as the design variables of a structure becomes greater and the solution

[1] Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Narmak, Tehran-16, Iran
Email: alikaveh@iust.ac.ir
[2] Department of Civil Engineering, University of Tabriz, Tabriz, Iran
Email: siamak.talat@gmail.com

Z.W. Geem (Ed.): Harmony Search Algo. for Structural Design Optimization, SCI 239, pp. 159–198.
springerlink.com

does not necessarily correspond to the global optimum or even the neighborhood of it, in some cases.

The computational drawbacks of classical numerical methods have forced researchers to rely on heuristic algorithms such as genetic algorithms (GAs), particle swarm optimizer (PSO), ant colony optimization (ACO) and harmony search (HS). These methods have attracted a great deal of attention, because of their high potential for modeling engineering problems in environments which have been resistant to a solution by classic techniques. They do not require gradient information and possess better global search abilities than the conventional optimization algorithms. Although these are approximate methods (i.e. their solutions are good, but not provably optimal), they do not require the derivatives of the objective function and constraints [3]. Having in common the processes of natural evolution, these algorithms share many similarities: each maintains a population of solutions which are evolved through random alterations and selection. The differences between these procedures lie in the representation technique utilized to encode the candidates, the type of alterations used to create new solutions, and the mechanism employed for selecting new patterns.

The genetic algorithm is one of the heuristic algorithms initially suggested by Holland, and developed and extended by some of his students, Goldberg and De Jong. These algorithms simulate a natural genetics mechanism for synthetic systems based on operators that are duplicates of natural ones. In the last decade, GA has been used in the optimum structural design. One of the first applications was the weight minimization of a 10-bar truss by Goldberg and Samtani [4]. Also, many researchers have used genetic search in the design of various structures in which the search space was non-convex or discrete, Hajela [5], Rajeev and Krishnamoorthy [6,7], Koumousis and Georgious [8], Hajela and Lee [9], Wu and Chow [10], Soh and Yang [11], Camp *et al.* [12], Shrestha and Ghaboussi [13], Pezeshek *et al.* [14] Erbatur *et al.* [15], Coello and Christiansen [16], Greiner *et al.* [17], Kameshki and Saka [18-20], Saka [21, 22], and Kaveh and colleagues [23-28], among many others.

Application of swarm intelligence for optimization was first suggested by Eberhart and Kennedy [29] under the name of particle swarm optimization (PSO). The strength of PSO is underpinned by the fact that decentralized biological creatures can often accomplish complex goals by cooperation. A standard PSO algorithm is initialized with a population (swarm) of random potential solutions (particles). Each particle iteratively moves across the search space and is attracted to the position of the best fitness historically achieved by the particle itself (local best) and by the best among the neighbors of the particle (global best) [30]. Compared to other evolutionary algorithms based on heuristics, the advantages of PSO consist of easy implementation and a smaller number of parameters to be adjusted. Therefore, it has been widely employed for structural optimization problems [31-35]. However, it is known that the PSO algorithm had difficulties in controlling the balance between exploration (global investigation of the search place) and exploitation (the fine search around a local optimum) [36].

Ant colony optimization (ACO) was first proposed by Dorigo [37, 38] as a multi-agent approach to solve difficult combinatorial optimization problems and it has been applied to various engineering problems in recent years [39-44]. ACO was inspired by the observation of real ant colonies. Ants are social insects whose behavior is directed more to the survival of the colony as a whole than to that of a single individual component of the colony. An important behavior of ant colonies is their foraging behavior, and in particular, how the ants can find shortest paths between food sources and their nest. While walking from food sources to the nest and vice versa, ants deposit on the ground a substance called pheromone. Ants can smell pheromone and when choosing their way, they tend to choose, in probability, paths marked by strong pheromone concentrations. When more paths are available from the nest to a food source, a colony of ants will be able to exploit the pheromone trails left by the individual ants to discover the shortest path from the nest to the food source and back. One basic idea of the ACO approach is to employ the counterpart of the pheromone trail used by real ants as an indirect communication and as a form of memory of previously found solutions.

The harmony search method, as discussed in the previous chapters, is another robust heuristic optimization technique that imitates the musical performance process which takes place when a musician searches for a better state of harmony. Jazz improvisation seeks to find musically pleasing harmony similar to the optimum design process which seeks to find the optimum solution. 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. This approach is suggested by Geem *et al.* in 2001 [45] and first applied to a design of water distribution network. Since then, the algorithm has attracted many researchers due to its simplicity and effectiveness [1, 46-51].

Although there are several papers utilizing heuristic methods in the structural optimization field, using an individual heuristic method has often had some drawbacks because usually each method is suitable for solving only a specific group of problems. Preference for a special method will differ depending on the kind of the problem being studied. One technique to overcome these problems is hybridizing various methods to reach a robust approach.

In this chapter, the implementation of an efficient hybrid algorithm based on harmony search, particle swarm and ant colony strategies, namely heuristic particle swarm ant colony optimization (HPSACO), is developed to find an optimum design of truss structures with continuous or discrete domains and to find frame structures with a discrete search domain.

2 Review of PSO, ACO and HS Algorithms

Since HPSACO methodology is based on PSO, ACO and HS, in order to make the chapter self-explanatory, the characteristics of these algorithms are briefly explained in this section.

2.1 Particle Swarm Optimization

Particle swarm optimization (PSO) is a stochastic optimization method capable of handling non differentiable, nonlinear, and multi module objective functions. The

PSO method is motivated from the social behavior of bird flocking and fish schooling [29]. PSO has a population of individuals that move through search space and each individual has a velocity that acts as an operator to obtain a new set of individuals. Individuals, called particles, adjust their movements depending on both their own experience and the population's experience. Effectively, each particle continuously focuses and refocuses on the effort of its search according to both the local and global best. This behavior mimics the cultural adaptation of a biological agent in a swarm: it evaluates its own position based on certain fitness criteria, compares it to others, and imitates the best position in the entire swarm [30].

Through the updating process, each particle moves by adding a change velocity $\mathbf{V}_i^{k+1}$ to the current position $\mathbf{X}_i^k$ as follows

$$\mathbf{X}_i^{k+1} = \mathbf{X}_i^k + \mathbf{V}_i^{k+1} \tag{1}$$

The velocity is a combination of three contributing factors:

1. Previous velocity, $\mathbf{V}_i^k$, considering former attempts;
2. Movement in the direction of the local best, $\mathbf{P}_i^k$, using the autobiographical memory;
3. Movement in the direction of the global best, $\mathbf{P}_g^k$, based on the publicized knowledge.

The mathematical relationship can be expressed as

$$\mathbf{V}_i^{k+1} = \omega \mathbf{V}_i^k + c_1 r_1 (\mathbf{P}_i^k - \mathbf{X}_i^k) + c_2 r_2 (\mathbf{P}_g^k - \mathbf{X}_i^k) \tag{2}$$

where ω is an inertia weight to control the influence of the previous velocity; r_1 and r_2 are two random numbers uniformly distributed in the range of (0, 1); c_1 and c_2 are two acceleration constants. $\mathbf{P}_i^k$ is the best position of the i th particle up to iteration k and $\mathbf{P}_g^k$ is the best position among all particles in the swarm up to iteration k. $\mathbf{P}_i^k$ and $\mathbf{P}_g^k$ are given by the following equations

$$\mathbf{P}_i^k = \begin{cases} \mathbf{P}_i^{k-1} & f(\mathbf{X}_i^k) \geq f(\mathbf{P}_i^{k-1}) \\ \mathbf{X}_i^k & f(\mathbf{X}_i^k) < f(\mathbf{P}_i^{k-1}) \end{cases} \tag{3}$$

$$\mathbf{P}_g^k = \left\{ \mathbf{P}_i^k \mid f(\mathbf{P}_i^k) = \min(f(\mathbf{P}_g^{k-1}) \wedge f(\mathbf{P}_j^k) \, , j = 1,2,.., M) \right\} \tag{4}$$

where $f(\mathbf{X})$ is the objective function, M is the total number of particles.

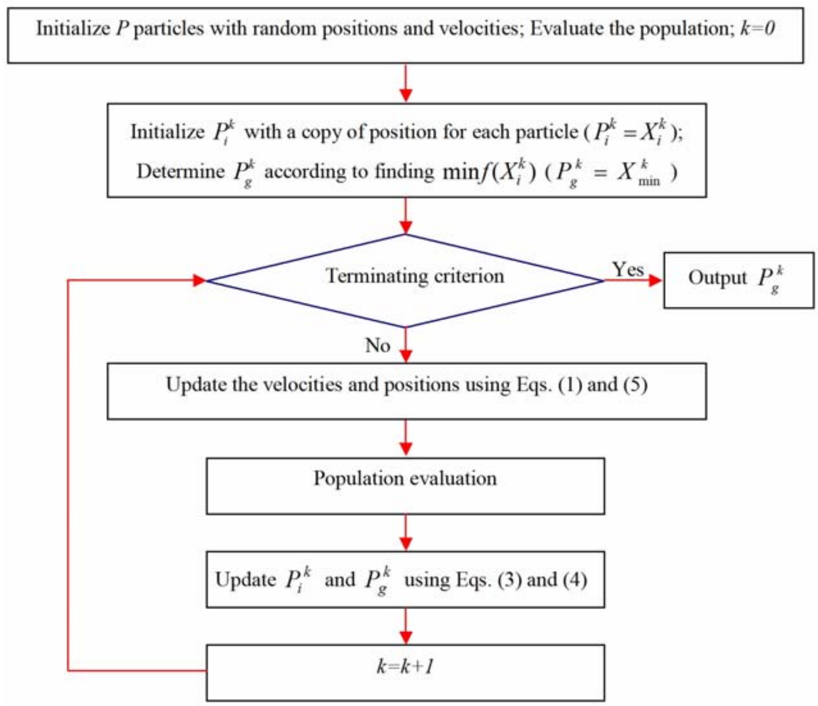


Fig. 1 The flow chart for the PSOPC algorithm

The pseudo-code of the PSO algorithm can be summarized as follows:

Step 1: *Initialization*. Initialize an array of particles with random positions and their associated velocities.

Step 2: *Function evaluation*. Evaluate the fitness function of each particle.

Step 3: *Local best updating*. Compare the current value of the fitness function with the particles' previous best value and update $\mathbf{P}_i^k$ according to Eq. (3).

Step 4: *Global best updating*. Determine the current global minimum fitness value among the current positions and update $\mathbf{P}_g^k$ according to Eq. (4).

Step 5: *Solution construction*. Change the velocities according to Eq. (2) and move each particle to the new position according to Eq. (1).

Step 6: *Terminating criterion controlling*. Repeat Steps 2–5 until a terminating criterion is satisfied. The terminating criteria are usually one of the following:

- *Maximum number of iterations*: the optimization process is terminated after a fixed number of iterations, for example, 1000 iterations.
- *Number of iterations without improvement*: the optimization process is terminated after some fixed number of iterations without any improvement.

- *Minimum objective function error*: the error between the values of the objective function and the best fitness is less than a pre-fixed anticipated threshold.

Adding the passive congregation model to the PSO may increase its performance. He *et al.* [52] proposed a hybrid PSO with passive congregation (PSOPC). In this method, the velocity is defined as

$$\mathbf{V}_i^{k+1} = \omega \mathbf{V}_i^k + c_1 r_1 (\mathbf{P}_i^k - \mathbf{X}_i^k) + c_2 r_2 (\mathbf{P}_g^k - \mathbf{X}_i^k) + c_3 r_3 (\mathbf{R}_i^k - \mathbf{X}_i^k) \tag{5}$$

where $\mathbf{R}_i$ is a particle selected randomly from the swarm; c_3 is the passive congregation coefficient; r_3 is a uniform random sequence in the range (0, 1).

Several benchmark functions have been tested in Ref. [52]. The results show that the PSOPC has a better convergence rate and a higher accuracy than the PSO. Figure 1 shows the flow chart for the PSOPC algorithm.

2.2 Ant Colony Optimization

In 1992, Dorigo developed a paradigm known as ant colony optimization (ACO), a cooperative search technique that mimics the foraging behavior of real live ant colonies [37]. The ant algorithms mimic the techniques employed by real ants to rapidly establish the shortest route from food source to their nest and vice versa. Ants start searching the area surrounding their nest in a random manner. Ethologists observed that ants can construct the shortest path from their colony to the feed source and back using pheromone trails [53, 54], as shown in Figure 2(a). When ants encounter an obstacle (Figure 2(b)), at first, there is an equal probability for all ants to move right or left, but after a while (Figure 2(c)), the number of ants choosing the shorter path increases because of the increase in the amount of the pheromone on that path. With the increase in the number of ants and pheromone on the shorter path, all of the ants will choose and move along the shorter path, Figure 2(d).

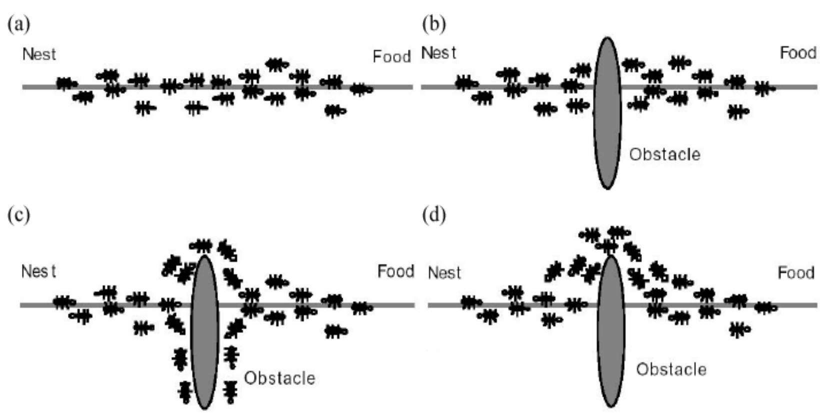


Fig. 2 Ants find the shortest path around an obstacle

In fact, real ants use their pheromone trails as a medium for communication of information among them. When an isolated ant comes across some food source in its random sojourn, it deposits a quantity of pheromone on that location. Other randomly moving ants in the neighborhood can detect this marked pheromone trail. Further, they follow this trail with a very high degree of probability and simultaneously enhance the trail by depositing their own pheromone. More and more ants follow the pheromone rich trail and the probability of the trail being followed by other ants is further enhanced by the increased trail deposition. This is an autocatalytic (positive feedback) process which favors the path along which more ants previously traversed. The ant algorithms are based on the indirect communication capabilities of the ants. In ACO algorithms, virtual ants are deputed to generate rules by using heuristic information or visibility and the principle of indirect pheromone communication capabilities for iterative improvement of rules.

ACO was initially used to solve the traveling salesman problem (TSP). The aim of TSP is finding the shortest Hamiltonian graph, $G=(N,E)$, where N denotes the set of nodes, and E is the set of edges. The general procedure of the ACO algorithm manages the scheduling of four steps [3]:

Step 1: *Initialization*. The initialization of the ACO includes two parts: the first consists mainly of the initialization of the pheromone trail. Second, a number of ants are arbitrarily placed on the nodes chosen randomly. Then each of the distributed ants will perform a tour on the graph by constructing a path according to the node transition rule described below.

Step 2: *Solution construction*. Each ant constructs a complete solution to the problem according to a probabilistic state transition rule. The state transition rule depends mainly on the state of the pheromone and visibility of ants. Visibility is an additional element used to make this method more efficient. For the path between i to j, it is represented as η_{ij} and in TSP, it has a reverse relation with the distance between i to j. The node transition rule is probabilistic. For the kth ant on node i, the selection of the next node j to follow is according to the node transition probability

$$P_{ij}(t) = \frac{[\tau_{il}(t)]^{\alpha} \cdot [\eta_{il}]^{\beta}}{\sum_{l \in N_i^k} [\tau_{il}(t)]^{\alpha} \cdot [\eta_{il}]^{\beta}} \quad \forall j \in N_i^k \tag{6}$$

where $\tau_{ij}(t)$ is the intensity of pheromone laid on edge (i, j); N_i^k is the list of neighboring nodes from node i available to ant k at time t. Parameters α and β represent constants which control the relative contribution between the intensity of pheromone laid on edge (i, j) reflecting the previous experiences of the ants about this edge, and the value of visibility determined by a Greedy heuristic for the original problem.

Step 3: *Pheromone updating rule*. When every ant has constructed a solution, the intensity of pheromone trails on each edge is updated by the pheromone updating rule. The pheromone updating rule is applied in two phases. First, an evaporation phase where a fraction of the pheromone evaporates, and then a reinforcement phase when the elitist ant which has the best solution among others, deposits an amount of pheromone

$$\tau_{ij}(t+n) = (1-\rho)\cdot\tau_{ij}(t) + \rho\cdot\Delta\tau_{ij}^{+} \tag{7}$$

where ρ ($0<\rho<1$) represents the persistence of pheromone trails (($1-\rho$) is the evaporation rate); n is the number of variables or movements an ant must take to complete a tour and $\Delta\tau_{ij}^{+}$ is the amount of pheromone increase for the elitist ant and equals

$$\Delta\tau_{ij}^{+} = \frac{1}{L^{+}} \tag{8}$$

where L^{+} is the length of the solution found by the elitist ant.

Step 4: *Terminating criterion controlling.* Steps 2 and 3 are iterated until a terminating criterion.

The flow chart of the ACO procedure is illustrated in Figure 3.

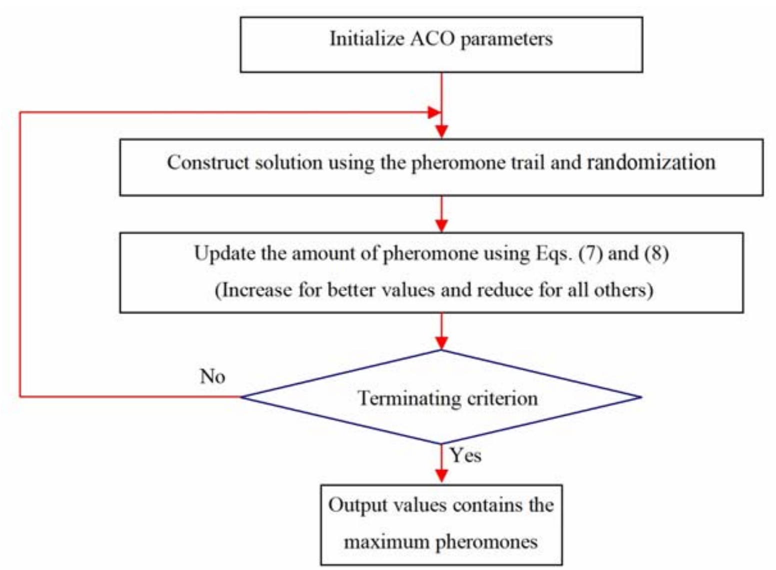


Fig. 3 The flow chart for the ACO algorithm

2.3 Harmony Search Algorithm

Harmony search (HS) algorithm is based on musical performance processes that occur when a musician searches for a better state of harmony, such as during jazz improvisation [45]. The engineers seek for a global solution as determined by an objective function, just like the musicians seek to find musically pleasing harmony as determined by an aesthetic. The HS algorithm was presented in previous chapters, and we briefly explain the steps in the algorithm here. Figure 4 shows the HS optimization procedure including the following steps [1]:

Step 1: *Initialization.* HS algorithm includes a number of optimization operators, such as the harmony memory (**HM**), the harmony memory size (HMS), the harmony memory considering rate (HMCR), and the pitch adjusting rate (PAR). In the HS algorithm, the **HM** stores the feasible vectors, which are all in the feasible space. The harmony memory size determines the number of vectors to be stored.

$$\mathbf{HM} = \begin{bmatrix} \mathbf{X}^1 \\ \vdots \\ \mathbf{X}^{HMS} \end{bmatrix}_{HMS \times ng} \tag{9}$$

Step 2: *Solution construction*. A new harmony vector is generated from the **HM**, based on memory considerations, pitch adjustments, and randomization. The HMCR varying between 0 and 1 sets the rate of choosing a value in the new vector from the historic values stored in the **HM**, and (1−HMCR) sets the rate of randomly choosing one value from the possible range of values.

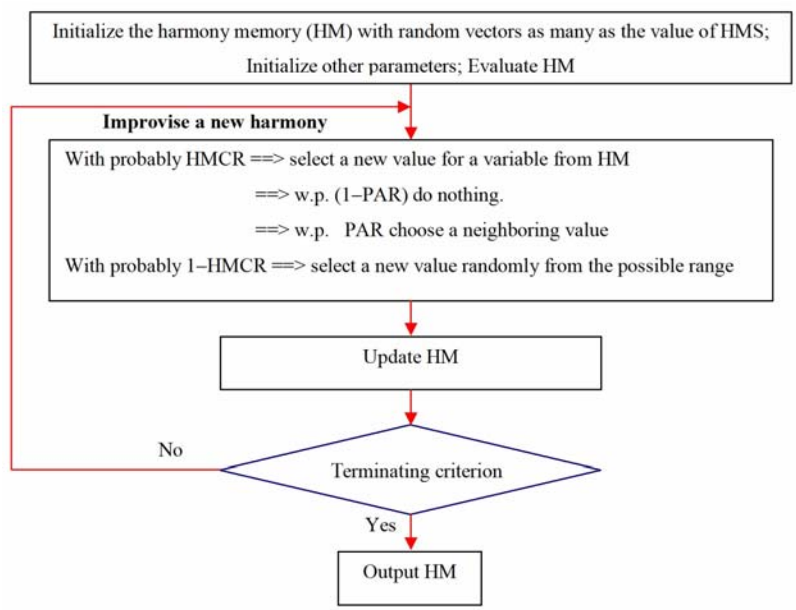


Fig. 4 The flow chart for the HS algorithm

$$x_i^k = \begin{cases} \text{Select from } \{x_i^1, x_i^2, \ldots, x_i^{HMS}\} & \text{w.p. HMCR} \\ \\ \text{Select from the possible range} & \text{w.p. } (1-\text{HMCR}) \end{cases} \quad (10)$$

where x_i^k is the ith design variable in the iteration k, and w.p. is abbreviation for "with probability". The pitch adjusting process is performed only after a value is chosen from the **HM**. The value (1−PAR) sets the rate of doing nothing. A PAR of 0.1 indicates that the algorithm will choose a neighboring value with 10% ×HMCR probability.

Step 3: *Harmony memory updating.* In Step 3, if a new harmony vector is better than the worst harmony in the **HM**, judged in terms of the objective function value, the new harmony is included in the **HM** and the existing worst harmony is excluded from the **HM**.

Step 4: *Terminating criterion controlling.* Repeat Steps 2 and 3 until the terminating criterion is satisfied. The computations are terminated when the terminating criterion is satisfied. Otherwise, Steps 2 and 3 are repeated.

3 Statement of the Optimization Design Problem

Selection of the objective function in optimal design problems is highly significant. Usually finding a mathematical formula for the objective function is not an easy task, especially when the optimization problem is very detailed. In most cases, the objective function shows one important feature of a design, but it can also contain a combination of different features [2]. Objective functions that can be used to measure the quality of design may include minimum construction cost, minimum life cycle cost, minimum weight, and maximum stiffness, as well as other objectives [1]. However, for structural optimization problems, minimization of the weight is often used as the objective function. Structural design is often limited by problem-specified constraints (e.g., feasible strength, displacements,

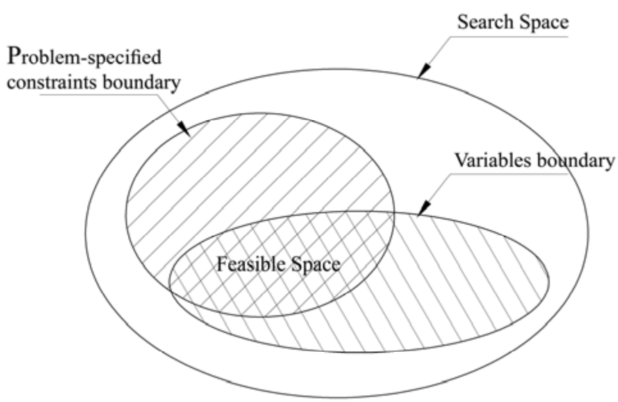


Fig. 5 Search space division

eigen-frequencies) and design variable constraints (e.g., type and size of the available structural members and cross-sections). The optimum design of structures involves a set of design variables that has the minimum weight located in the feasible space which does not violate either problem-specified constraints or design variable constraints, as illustrated in Figure 5.

3.1 Optimum Design of Truss Structures

Optimum design of truss structures involves arriving at optimum values for member cross-sectional areas x_i that minimize the structural weight W.

$$\begin{aligned} &\text{Find} \quad \mathbf{X} = [x_1, x_2, ..., x_{ng}], \\ &\qquad\qquad x_i \in D_i \\ &\text{to minimize} \quad W(\mathbf{X}) = \sum_{i=1}^{nm} \gamma_i \cdot x_i \cdot L_i \end{aligned} \tag{11}$$

where $\mathbf{X}$ is the vector containing the design variables; ng is the number of design variables or the number of groups; $W(\mathbf{X})$ is the cost function which is taken as the weight of the structure; nm is the number of members making up the structure; γ_i is the material density of member i; L_i is the length of member i; D_i is an allowable set of values for the design variable x_i which can be considered as a continuous set or a discrete one. In the continuous problems, the design variables can vary continuously in the optimization

$$D_i = \{x_i \mid x_i \in [x_{i,\min}, x_{i,\max}]\} \tag{12}$$

where $x_{i,\min}$ and $x_{i,\max}$ are minimum and maximum allowable values for the design variable i, respectively. If the design variables represent a selection from a set of parts, the problem is considered as discrete

$$D_i = \{d_{i,1}, d_{i,2}, ..., d_{i,r(i)}\} \tag{13}$$

where $r(i)$ is the number of available discrete values for the ith design variable.

This minimum design also has to satisfy the problem-specified constraints that limit structural responses, as follows

$$\begin{aligned} &\text{subject to} \quad \delta_{\min} \le \delta_{\mathrm{i}} \le \delta_{\max} \qquad i = 1,2,..., m \\ &\qquad\qquad\quad \sigma_{\min} \le \sigma_{\mathrm{i}} \le \sigma_{\max} \qquad i = 1,2,..., nm \end{aligned} \tag{14}$$

where σ_i and δ_i are the stress and nodal deflection, respectively; m is the number of nodes; and min and max mean the lower and upper boundaries, respectively.

3.2 Optimum Design of Steel Frames

Similar to truss structures, the aim of the optimum design of steel frames is to find a design with minimum weight as described in Equation (11) which must satisfy the following constraints:

Stress constraints

$$\sigma_{\min} \le \sigma_{\mathrm{i}} \le \sigma_{\max} \qquad i = 1,2,..., nm \tag{15}$$

Maximum lateral displacement

$$\frac{\Delta_T}{H} \le R \tag{16}$$

Inter-story displacement constraints

$$\frac{\Delta_j}{h_j} \le R_I \qquad j = 1,2,..., ns \tag{17}$$

where D_i is considered a set of 267 W-sections from the AISC database [55] for the design variable x_i; Δ_T is the maximum lateral displacement; H is the height of the frame structure; R is the maximum drift index; Δ_j is the inter-story drift; h_j is the story height of the jth floor; ns is the total number of stories; and R_I is the inter-story drift index permitted by the code of the practice.

For the code of practice AISC [55], the allowed inter-story drift index is given as 1/300, and the LRFD interaction formula constraints (AISC, Equation H1-1a,b) are expressed as

$$\frac{P_u}{2\phi_c P_n} + \left(\frac{M_{ux}}{\phi_b M_{nx}} + \frac{M_{uy}}{\phi_b M_{ny}} \right) \le 1 \quad \text{For} \qquad \frac{P_u}{\phi_c P_n} < 0.2 \tag{18}$$

$$\frac{P_u}{\phi_c P_n} + \frac{8}{9}\left(\frac{M_{ux}}{\phi_b M_{nx}} + \frac{M_{uy}}{\phi_b M_{ny}} \right) \le 1 \quad \text{For} \qquad \frac{P_u}{\phi_c P_n} \ge 0.2 \tag{19}$$

where P_u is the required strength (tension or compression); P_n is the nominal axial strength (tension or compression); ϕ_c is the resistance factor ($\phi_c = 0.9$ for tension, $\phi_c = 0.85$ for compression); M_{ux} and M_{uy} are the required flexural strengths in the x and y directions, respectively; M_{nx} and M_{ny} are the nominal flexural strengths in the x and y directions (for two-dimensional structures, M_{ny}=0); and ϕ_b is the flexural resistance reduction factor ($\phi_b = 0.90$).

4 A Heuristic Particle Swarm Ant Colony Optimization

The heuristic particle swarm ant colony optimization (HPSACO), a hybridized approach based on HS, PSO and ACO, is described in this section. HPSACO utilizes a particle swarm optimization with passive congregation (PSOPC) algorithm as a global search, and the ant colony approach worked as a local search. In the HPSACO algorithm, fly-back mechanism and the harmony search are used to handle the constraints. Fly-back mechanism handles the problem-specific constraints, and the HS deals with the variable constraints. HPSACO utilizes an efficient terminating criterion considering exactitude of the solutions. This terminating criterion is defined in a way that after decreasing the movements of particles, the search process stops. In the discrete method of HPSACO, agents are not allowed to select any value except discrete cross sections from the permissible list.

4.1 Combining PSO with ACO

Compared to other evolutionary algorithms based on heuristics, the advantages of PSO include an easy implementation and its smaller number of parameters to be adjusted. However, it is known that the original PSO had difficulties in controlling the balance between exploration (global investigation of the search place) and exploitation (the fine search around a local optimum) [36]. In order to improve upon this character of PSO, one method is to hybridize PSO with other approaches such as ACO. The resulted method, called particle swarm ant colony optimization (PSACO), was initially introduced by Shelokar *et al.* [56] for solving the continuous unconstrained problems and recently utilized for the design of structures by authors [57, 58]. We have applied PSOPC instead of the PSO to improve the performance of the new method. The relation of the standard deviation in ACO stage is different with Ref. [56] and the inertia weight is changed in PSOPC stage.

The implementation of PSACO algorithm consists of two stages [57]. In the first stage, it applies PSOPC, while ACO is implemented in the second stage. ACO works as a local search, wherein, ants apply pheromone-guided mechanism to refine the positions found by particles in the PSOPC stage. In the PSACO, a simple pheromone-guided mechanism of the ACO is proposed to be applied for the local search. The proposed ACO algorithm handles *M* ants equal to the number of particles in PSOPC.

In ACO stage, each ant generates a solution around $\mathbf{P}_g^k$ which can be expressed as

$$\mathbf{Z}_i^k = N(\mathbf{P}_g^k, \sigma) \tag{20}$$

In the above equation, $N(P_g^k, \sigma)$ denotes a random number normally distributed with mean value P_g^k and variance σ, where

$$\sigma = (x_{\max} - x_{\min}) \times \eta \tag{21}$$

η is used to control the step size. The normal distribution with mean $\mathbf{P}_g^k$ can be considered as a continuous pheromone which has the maximum value in $\mathbf{P}_g^k$ and which decreases going away from it. In ACO algorithms, the probability of selecting a path with more pheromone is greater than other paths. Similarly, in the normal distribution, the probability of selecting a solution in the neighborhood of $\mathbf{P}_g^k$ is greater than the others. This principle is used in the PSACO algorithm as a helping factor to guide the exploration and to increase the controlling in exploitation.

In the present method, the objective function value $f(\mathbf{Z}_i^k)$ is computed and the current position of ant *i*, $\mathbf{Z}_i^k$, is replaced by the current position of particle *i* in the swarm, $\mathbf{X}_i^k$, if $f(\mathbf{X}_i^k) > f(\mathbf{Z}_i^k)$ and the current ant is in the feasible space.

4.2 HS Added to PSACO as a Variable Constraint Handling Approach

The heuristic particle swarm ant colony optimization algorithm (HPSACO) is resulted from combining PSACO and HS [59]. The framework of the HPSACO algorithm is illustrated in Figure 6. A hybrid particle swarm optimizer and harmony search scheme (HPSO) was proposed by Li *et al.* [32] for truss design. A particle in the search space may violate either the problem-specific constraints or the limits of the variables as illustrated in Figure 5. If a particle flies out of the variable boundaries, the solution cannot be used even if the problem-specific constraints are satisfied. Using the harmony search-based handling approach, this problem is dealt with. In this mechanism, any component of the solution vector (particle) violating the variable boundaries can be generated randomly from $\mathbf{P}_i^k$ as

$$x_{i,j} = \begin{cases} \text{w.p. HMCR} \Longrightarrow \text{select a new value for a variable from } \mathbf{P}_i^k \\ \qquad\qquad \Longrightarrow \text{w.p. } (1-\text{PAR}) \text{ do nothing} \\ \qquad\qquad \Longrightarrow \text{w.p. PAR choose a neighboring value} \\ \text{w.p. } (1-\text{HMCR}) \Longrightarrow \text{select a new value randomly} \end{cases} \tag{22}$$

where $x_{i,j}$ is the *j*th component of the particle i The HMCR varying between 0 and 1 sets the rate of choosing a value in the new vector from the historic values stored in the $\mathbf{P}_i^k$, and (1–HMCR) sets the rate of randomly choosing one value from the possible list of values. The pitch adjusting process is performed only after a value is chosen from $\mathbf{P}_i^k$. The value (1–PAR) sets the rate of doing nothing. A PAR (Pitch Adjusting Rate) of 0.1 indicates that the algorithm will choose a neighboring value with 10% ×HMCR probability. Therefore, the harmony search concept is used to check whether the particles violate the variables' boundaries.

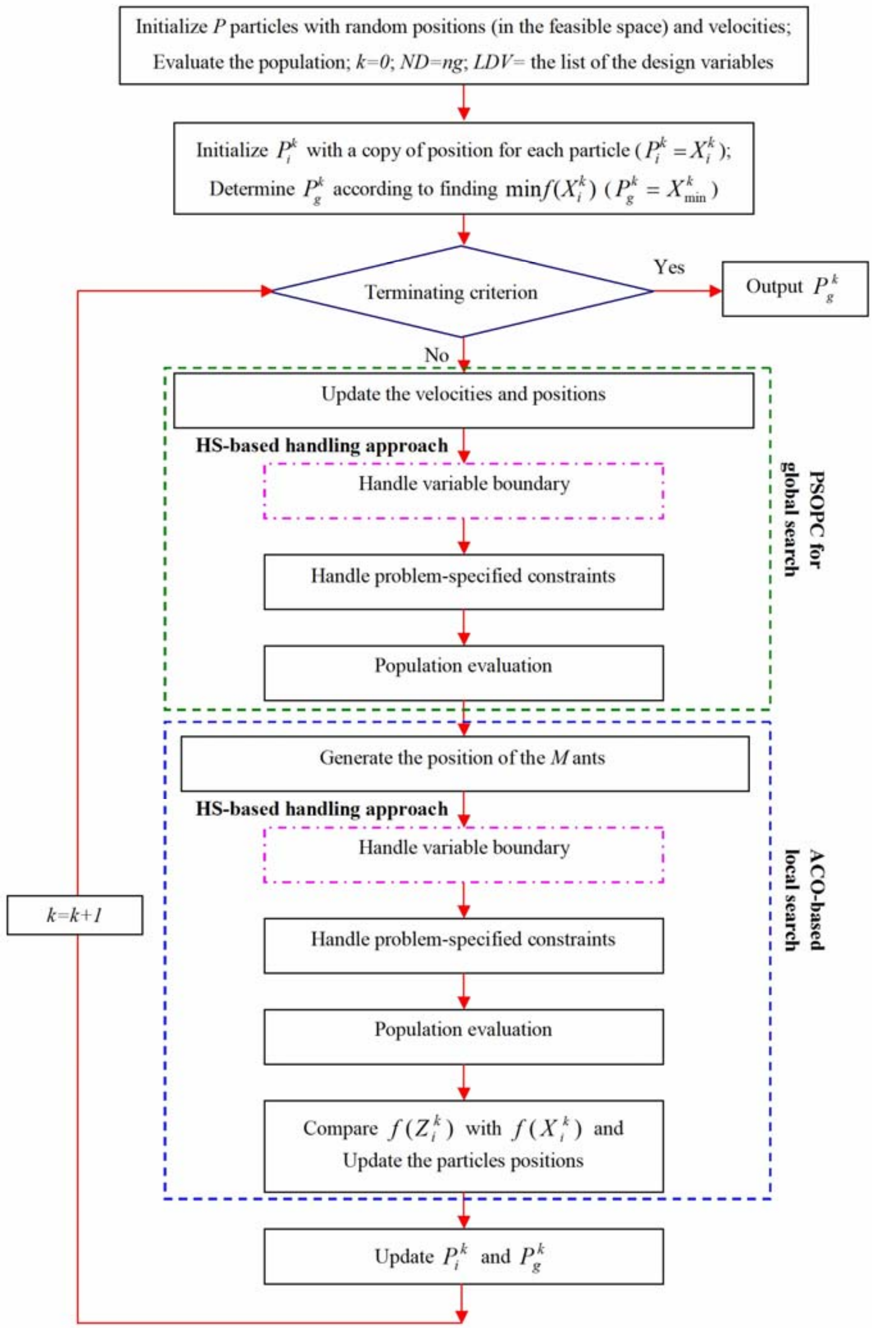


Fig. 6 The flow chart for the HPSACO

4.3 Problem-Specified Constraint Handling Approach

As described in the previous section, there are some problem-specified constraints in structural optimization problems that should be carefully handled. So far, a number of approaches have been proposed by incorporating constraint-handling techniques to solve constrained optimization problems. The most common approach adopted to deal with constrained search spaces is the use of penalty functions. When using a penalty function, the amount of constraint violation is used to punish or penalize an infeasible solution so that feasible solutions are favored by the selection process. Despite the popularity of penalty functions, they have several drawbacks. The main one is that they require a careful fine tuning of the penalty factors that accurately estimates the degree of penalization to be applied in order to approach efficiently the feasible region [60].

Several approaches have been proposed to avoid this dependency on the values of the penalty factors, like special encodings, whose aim is to generate only feasible solutions, and the use of special operators to preserve their feasibility during all the evolutionary process [61, 62]. An alternative approach is the use of repair algorithms whose goal is to change an infeasible solution into a feasible one [63]. The separation of constraints and objectives is another approach to deal with constrained search spaces, where the idea is to avoid the combination of the value of the objective function and the constraints of a problem to assign fitness, like when using a penalty function [60, 64].

Fly-back mechanism is one of the methods for separating constraints and objective functions, introduced by He *et al.* [64]. Compared to other constraint-handling techniques, this method is relatively simple and easy to implement. For most of the structural optimization problems, the global minimum locates on or close to the boundary of a feasible design space. According to the fly-back mechanism, the particles are initialized in the feasible region. When the particles fly in the feasible space to search the solution, if any one of them flies into the infeasible region, it will be forced to fly back to the previous position to guarantee a feasible solution. The particle which flies back to the previous position may be closer to the boundary at the next iteration. This makes the particles fly to the global minimum with a great probability. Although some experimental results have shown that it can find a better solution with a fewer number of iterations than the other techniques [64], the fly-back mechanism has the difficulty of finding the first valid solutions for the swarm. However, if the first selections are limited to a neighborhood of the maximum value of permitted cross sectional areas, it can be expected, after a few iterations, the feasible swarm will be obtained. This neighborhood can be defined as [59]

$$\left[x_{\max} - \frac{x_{\max} - x_{\min}}{4}, x_{\max} \right] \tag{23}$$

4.4 Terminating Criterion

The maximum number of the iterations is the most usual terminating criterion in PSO literature. If it is selected as a big number, the number of analyses and as a

Table 1 The pseudo-code for the HPSACO

Set $k=0$; ND= the number of design variables (ng);
Set LDV= the list of design variables;
Randomly initialize positions and velocities of all particles (from the range of $[A_{min}, A_{max}]$)
FOR(each particle i in the initial population)
    WHILE(the constraints are violated)
        Randomly re-generate the current particle X_i
    END WHILE
    Generate local best: Set $P_i^k = X_i^k$
    Generate global best: Find $\min f(X_i^k)$, P_g^k is set to the position of $X_{\min}^k$
END FOR
 	
WHILE($ND \neq 0$)
    FOR(each particle (ant) i in the swarm(colony))
        Generate the velocity and update the position of the current particle (vector) X_i^k
        Variable boundary handling: Check whether each component of the current vector violates its corresponding boundary or not. If it does, select the corresponding component of the vector from P_i^k based on memory considerations, pitch adjustments, and randomization
        Problem-specified constraints handling: Check whether the current particle violates the problem constraints or not. If it does, reset it to the previous position X_i^{k-1}
        Calculate the fitness value $f(X_i^k)$ of the current particle
        Generate the position of the current ant $Z_i^k = N(P_g^k, \sigma)$
        Variable boundary handling: Check whether each component of the current vector violates its corresponding boundary or not. If it does, select the corresponding component of the vector from P_i^k based on memory considerations, pitch adjustments, and randomization.
        Problem-specified constraints handling: Check whether the current ant violates the problem constraints or not. If it does, reset it to the current particle X_i^k
        Calculate the fitness value $f(Z_i^k)$ of the current ant
        Update current particle position: Compare the fitness value of current ant with current particle. If the $f(Z_i^k)$ is better than the fitness value of $f(X_i^k)$, set $f(X_i^k) = f(Z_i^k)$ and $X_i^k = Z_i^k$
        Update local best: Compare the fitness value of $f(P_i^k)$ with $f(X_i^k)$. If the $f(X_i^k)$ is better than the fitness value of $f(P_i^k)$, set P_i^k to the current position X_i^k
    END FOR
    Update global best: Find the global best position in the swarm. If the $f(X_i^k)$ is better than the fitness value of $f(P_g^k)$, P_g^k is set to the position of the current particle X_i^k
 	
    FOR(each $j \in LDV$)
        IF($\lvert V_{ij}^k \rvert < A^*/2, \quad \forall i \in swarm$)
            Set $ND=ND-1$ and Delete variable j from LDV
        END IF
    END FOR
 	
    Set $k=k+1$
END WHILE

result, the time of optimization will increase; vice versa, if it is selected small, the probability of finding a desirable solution will decrease. Thus, the necessity for an exact definition of the terminating criterion is vital. The following terminating criterion is considered to fulfill this goal.

This terminating criterion is defined by using a pre-fixed value denoted by A^*. For the discrete problems, A^* is equal to the minimum value of the difference between cross-sectional areas of two successive discrete sections, and for continuous problems, A^* is considered as the required exactitude of the solutions with a reverse relation. According to this criterion, as A^* increases, exactitude of the solutions decreases and the searching process must be stopped earlier, and if the amount of A^* decreases, then the searching process must be continued until an exact result is attained. Therefore, if in an iteration of search process, the absolute value of the component i in all of the particles' velocity vectors is less than $A^*/2$, continuation of the search process cannot change the amount of variable i; then the variable i reaches an optimum value and can be deleted from the virtual list of design variables. As a result, the terminating criterion is defined as continuing the search process until all variables are deleted. In the other words, when the variation of a variable is less than $A^*/2$, this criterion omits it from the virtual list of variables. When this list is emptied, the search process stops. With these alterations, the number of iterations decreases.

The pseudo-code for the HPSACO algorithm using this terminating criterion is listed in Table 1.

4.5 A Discrete HPSACO

In the discrete HPSACO, a new position of each agent is defined as the following:

For particles

$$\mathbf{X}_i^{k+1} = Fix(\mathbf{X}_i^k + \mathbf{V}_i^{k+1}) \tag{24}$$

For ants

$$\mathbf{Z}_i^k = Fix\left(N(\mathbf{P}_g^k, \sigma)\right) \tag{25}$$

where $Fix(\mathbf{X})$ is a function which rounds each element of $\mathbf{X}$ to the nearest permissible discrete value. Using this position updating formula, the agents will be permitted to select discrete values. Although this change is simple and efficient, it may reduce the exploration in the algorithm. Therefore, in order to increase the exploration, the velocity of particles is redefined [58] as

$$\begin{aligned}\mathbf{V}_i^{k+1} = \omega\mathbf{V}_i^k + c_1 r_1(\mathbf{P}_i^k - \mathbf{X}_i^k) + c_2 r_2(\mathbf{P}_g^k - \mathbf{X}_i^k) + \\ c_3 r_3(\mathbf{R}_i^k - \mathbf{X}_i^k) + c_4 r_4(\mathbf{Rd}_i^k - \mathbf{X}_i^k)\end{aligned} \tag{26}$$

where c_4 is the exploration coefficient; r_4 is a uniformly distributed random number in the range of (0, 1); and $\mathbf{Rd}^k$ is a vector generated randomly from the search domain.

4.6 Parameter Setting

For the proposed algorithm, a population of 50 individuals is used for both particles and ants (M=50); the value of constants c_1 and c_2 are set 0.8 and the passive congregation coefficient c_3 is taken as 0.6. The value of inertia weight ($\omega(k)$) decreases linearly from 0.9 to 0.4 as follows

$$\omega(k) = 0.9 - 0.0015 \times k \geq 0.4 \tag{27}$$

where k = the iteration number. In this way, the balance of $\omega(k)$ with fast rate of convergence in the HPSACO method is maintained.

The amount of step size (η) in ACO stage is recommended as 0.01 [57]. If η is too small, the velocity of particles will decrease rapidly and the search process will stop in early iterations; thus the obtained results stay far away from an optimum; on the contrary, if it is selected too big, the HPSACO algorithm will perform similar to the PSOPC algorithm and the effect of the ACO stage will be eliminated, and a desirable solution cannot be obtained in smaller number of iterations.

The parameters of the HS part (HMCR and PAR) similar to the effect of η, can be investigated. With small values for HMCR (large values for PAR), the effect of the HS part will be deleted. We have selected these values close to the amounts employed in the original HS algorithm [1]. If HMCR is selected from the range of [0.8, 0.98] and PAR is taken from [0.05,0.25], we expect a good performance for the HPSACO. In this study, HMCR is set to 0.95 and PAR is taken as 0.10.

5 Discussion on the Efficiency of the HPSACO

In order to verify the effectiveness of the HPSACO algorithm, a benchmark problem (10-bar truss) chosen from the literature is employed. In the next section, four design examples consisting of a 120-bar dome shaped truss with continuous design variables, a 582-bar space truss tower with 32 discrete design variables and a 3-bay 15-story steel frame structure are used to evaluate the numerical performance of the HPSACO algorithm in optimum design of different types of structures.

5.1 Benchmark Problem

The 10-bar truss design has become a common problem in the field of structural design for testing and verifying the efficiency of many different optimization methods. Figure 7 shows the geometry and support conditions for this 2-dimensional, cantilevered truss with the corresponding loading condition. The material density is 0.1 lb/in^3 (2767.990 kg/m^3) and the modulus of elasticity is 10,000 ksi (68,950 MPa). The members are subjected to the stress limits of ±25 ksi (172.375 MPa) and all nodes in both vertical and horizontal directions are subjected to the displacement limits of ±2.0 in (5.08 cm). There are 10 design variables in this example and a set of pseudo variables ranging from 0.1 to 35.0 in^2 (from 0.6452 cm^2 to 225.806 cm^2). A^* is considered as 0.001 for this example.

The PSO and PSOPC algorithms achieve the best solutions after 3,000 iterations (150,000 analyses) [32] and the HS algorithm reaches a solution after 20,000 analyses [1]. However, the HPSACO algorithm finds the best solution after about 426 iterations (10,650 analyses). The best weight of HPSACO is 5056.56 lb while the best results of PSO and PSOPC are 5061.00 lb, 5529.50 lb, respectively. The results of this method are compared with other methods in Table 2.

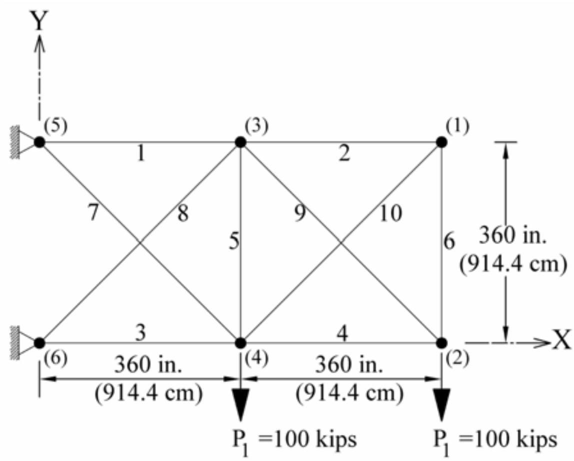


Fig. 7 A 10-bar planar truss

Table 2 Optimal design comparison for the 10-bar planner truss

Element group		Optimal cross-sectional areas (in.^{2})						
		GA [12]	HS [1]	PSO [32]	PSOPC [32]	HPSO [32]	PSACO [57]	HPSACO
1	A_1	28.92	30.15	33.469	30.569	30.704	30.068	30.307
2	A_2	0.10	0.102	0.110	0.100	0.100	0.100	0.100
3	A_3	24.07	22.71	23.177	22.974	23.167	23.207	23.434
4	A_4	13.96	15.27	15.475	15.148	15.183	15.168	15.505
5	A_5	0.10	0.102	3.649	0.100	0.100	0.100	0.100
6	A_6	0.56	0.544	0.116	0.547	0.551	0.536	0.5241
7	A_7	7.69	7.541	8.328	7.493	7.460	7.462	7.4365
8	A_8	21.95	21.56	23.340	21.159	20.978	21.228	21.079
9	A_9	22.09	21.45	23.014	21.556	21.508	21.630	21.229
10	A_{10}	0.10	0.100	0.190	0.100	0.100	0.100	0.100
Weight (lb)		5076.31	5057.88	5529.50	5061.00	5060.92	5057.36	5056.56

5.2 Discussion

The main reasons for the improvements obtained by the HPSACO method can be summarized as the following [59]:

1. *Increasing the exploitation*: In structural optimization, usually there are some local optimums in the neighborhood of a desirable solution. Thus, the probability of finding a desirable optimum increases with additional searches around the local optimums. HPSACO does extra search (exploitation) around the local optimums, and therefore obtains the desirable solution with higher probability in a smaller number of iterations.

The difference between the best and the worst results of the 10-bar truss for PSOPC in 50 tests is 365.2lb (7.21%), the average weight is 5173.45lb, and the standard deviation is 81.17lb (see Table 3). With adding the ACO principles to the PSOPC (PSACO [57]), these values are reduced to 3.2lb (0.06%), 5058.23lb, and 1.46lb, respectively. In addition, although PSO is a weak approach, applying ACO principles in PSO results in a improvement of its performance. The average weight of PSO+ACO in 50 runs is 5079.19lb, and the standard deviation is 4.76lb, which are better than PSOPC. Therefore, increasing the exploitation by applied pheromone-guided mechanism for updating the positions of the particles, not only improves the results, but also reduces the standard deviation drastically.

2. *Guiding the exploration*: Heuristic methods utilize two factors: the random search factor and the information collected from the search space during the optimization process. In early iterations, the random search factor has more power than the collective information factor, but the increase in the number of iterations gradually abates the power of the random search factor and increases the power of the collective information factor. In HPSACO, ACO stage plays an auxiliary role in rapidly increasing the collective information factor; consequently, the convergence rate increases faster.

   Although minimizing the maximum value of the velocity can make fewer particles violate the variable boundaries, it may also prevent the particles from crossing the problem-specific constraints and can cause the reduction in exploration. The harmony search-based handling approach deals with this problem.

PSOPC requires 3000 iterations to reach a solution for 10-bar truss. However, the number of required iterations to reach a solution for PSOPC+ACO (PSACO) in 50 runs on average is 635.2 iterations. Also, PSO+ACO on average needs 567 iterations to reach the optimum solution, while PSO cannot reach an appropriate solution until the maximum number of iterations is achieved (3000 iterations).

In order to investigate the advantages of the HS-based handling approach, the comparison of the performance of PSOPC with HPSO (PSOPC+HS), or PSO+ACO with PSO+ACO+HS, or PSOPC+ACO (PSACO) with PSOPC+ACO+HS (HPSACO) can be helpful. Table 3 summarizes the performances of all the above mentioned PSO-based approaches for the 10-bar truss on 50 runs for each algorithm. Although the results and standard deviations of HPSACO and PSACO do not differ much, the convergence rate of HPSACO is higher than that of PSACO. In average, HPSACO needs 420.3 iterations to reach a solution, while for PSACO this average number is 635.2.

3. *Efficient terminating criterion:* In optimization problems, the terminating criterion is a part of the search process which can be used to eliminate additional unnecessary iterations. To fulfill this goal, an efficient terminating criterion is defined as continuing the search process until the variation of a variable is less than a pre-defined value.

Figure 8 shows the average and a typical maximum absolute value of velocity for the first design variable in 50 tests for the 10-bar truss without considering the proposed terminating criterion. As shown in the figure, generally $\max(|V_{i1}^k|)$ is a decreasing function with a slight disorder. When it gets less than A^*, there is a probability (even slight) that the values of velocities in the next iterations become more than A^*. Instead, if the upper bound of the maximum absolute value of velocities is selected as $A^*/2$, there is a small probability that particle velocities in the next iterations become more than A^* and as a result, continuing the search process cannot help to improve the results.

Table 3 Investigation on the performance of various PSO-based algorithms for the 10-bar truss in 50 runs

Algorithm	Minimum iterations	Maximum iterations	Average iterations	Best weight (lb)	Worst weight (lb)	Average weight (lb)	Standard deviation (lb)
PSOPC	3000	3000	3000	5061.00	5406.26	5173.45	81.17
PSOPC+HS	3000	3000	3000	5060.92	5103.63	5078.69	13.05
PSO+ACO	373	567	439.6	5065.23	5092.71	5079.19	4.76
PSO+ACO+HS	226	414	296.3	5065.61	5078.26	5070.86	2.87
PSOPC+ACO	619	655	635.2	5057.36	5060.61	5058.23	1.46
PSOPC+ACO+HS (HPSACO)	405	436	420.3	5056.56	5061.12	5057.66	1.42

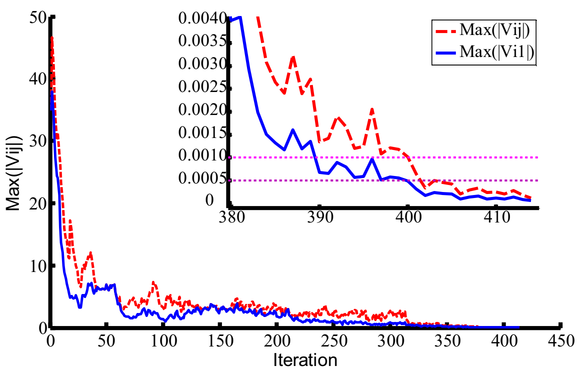


Fig. 8 The history of $\max(|V_{i1}^k|)$ in 50 tests for the 10-bar truss

6 Design Examples

6.1 A Truss Structure with Continuous Design Variables

Figure 9 shows the topology and group numbers of 120-bar dome shaped truss. The modulus of elasticity is 30,450 ksi (210,000 MPa), and the material density is 0.288 lb/in.^{3} (7971.810 kg/m^3). The yield stress of steel is taken as 58.0 ksi (400 MPa). The dome is considered to be subjected to vertical loading at all the unsupported joints. These loads are taken as –13.49 kips (–60 kN) at node 1, –6.744 kips (–30 kN) at nodes 2 through 14 and –2.248 kips (–10 kN) at the rest of the nodes. The minimum cross-sectional area of all members is 0.775 in.^{2} (2 cm^2). The allowable tensile and compressive stresses are used according to the AISC ASD [55] code, as follows

$$\begin{cases} \sigma_i^+ = 0.6F_y & \text{For} \quad \sigma_i \geq 0 \\ \sigma_i^- & \text{For} \quad \sigma_i < 0 \end{cases} \tag{28}$$

where σ_i^- is calculated according to the slenderness ratio:

$$\sigma_i^- = \begin{cases} \left[\left(1 - \dfrac{\lambda_i^2}{2C_C^2}\right) Fy\right] \Big/ \left(\dfrac{5}{3} + \dfrac{3\lambda_i}{8C_C} - \dfrac{\lambda_i^3}{8C_C^3}\right) & \text{For} \quad \lambda_i < C_C \\ \dfrac{12\pi^2 E}{23\lambda_i^2} & \text{For} \quad \lambda_i \geq C_C \end{cases} \tag{29}$$

where E is the modulus of elasticity; F_y is the yield stress of steel; C_c is the slenderness ratio (λ_i) dividing the elastic and inelastic buckling regions ($C_C = \sqrt{2\pi^2 E / F_y}$); λ_i is the slenderness ratio ($\lambda_i = kL_i / r_i$); k is the effective length factor; L_i is the member length; and r_i is the radius of gyration which can be expressed in terms of cross-sectional areas, i.e., $r_i = aA_i^b$ [29]. Here, a and b are the constants depending on the types of sections adopted for the members. Here, pipe sections (a = 0.4993 and b = 0.6777) were used for the bars.

In this example, four cases of constraints are considered: with stress constraints and no displacement constraints (Case 1), with stress constraints and displacement limitations of ±0.1969 in (±5 mm) imposed on all nodes in x- and y-directions (Case 2), no stress constraints but displacement limitations of ±0.1969 in. (±5 mm) imposed on all nodes in z-directions (Case 3), and all constraints explained above (Case 4). For Case 1 and Case 2, the maximum cross-sectional area it is 5.0 in.^{2} (32.26 cm^2) and for Case 3 and Case 4 is 20.0 in.^{2} (129.03 cm^2).

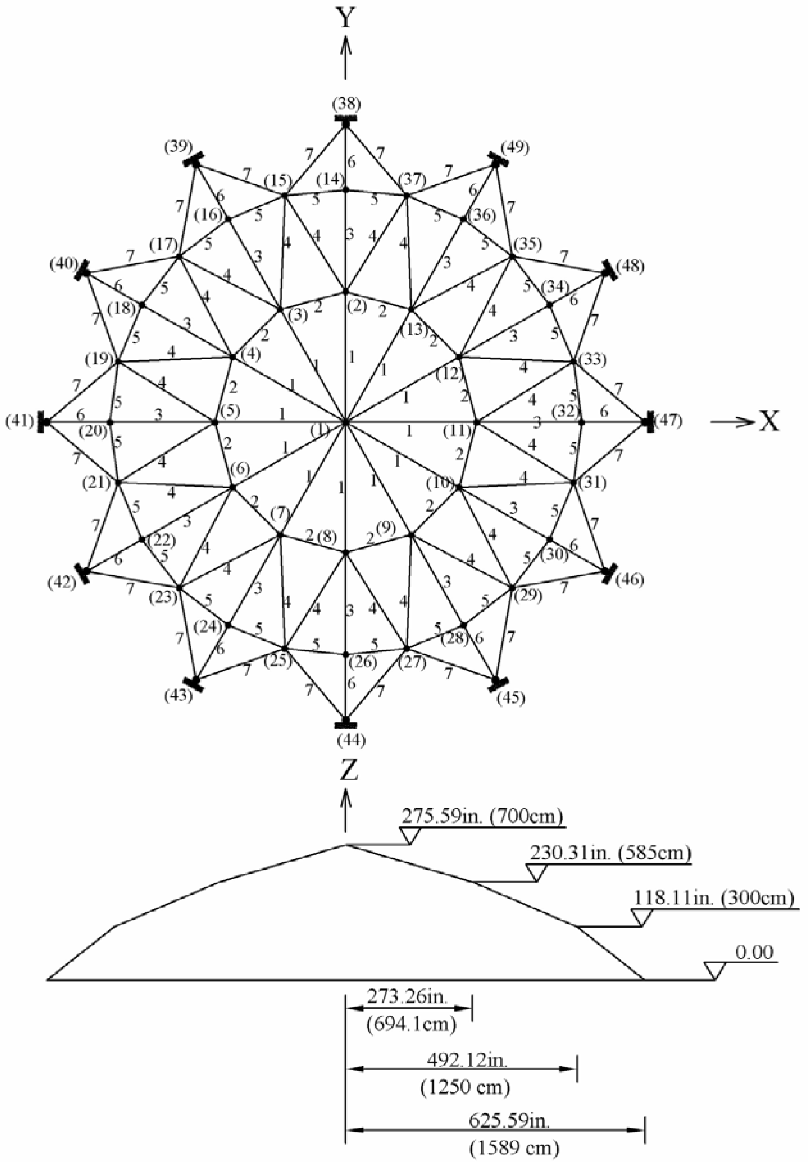


Fig. 9 A 120-bar dome shaped truss

The best solution vectors and the corresponding weights for all cases are provided in Table 4. Figure 10 shows the convergence for different cases. In all cases, HPSACO needs nearly 10,000 analyses (400 iterations) to reach a solution which is less than 125,000 (2,500 iterations) and 35,000 analyses for PSOPC and HS [1], respectively. Figures 11-14 compare the allowable and existing stress and displacement constraint values of the HPSACO results for four cases. In Case 1, the stress constraints of some elements in the 4th and 7th groups are active as shown in Figure 11(a). According to Figures 12(a), 13(a) and 14(a), the maximum values of displacements in the *x*, *y* and *z* directions are 0.3817in., 0.4144in. and 0.988in., respectively. In Case 2, the stress constraints in the 2nd, 4th and 7th groups and the displacement of node 26 in *y* direction are active. The maximum value for displacement in the *x* direction is 0.1817in. (Figure 11(b)).. The displacement constraints in the *x* and *y* directions do not affect the results of Case 3 and Case 4. The active constraints for Case 3 are the displacements of the 1st to 13th nodes in the *z* direction (Figure 14(c)). In Case 4, the stresses in the elements of the 7th group and the displacements of the 1st to 13th nodes in *z* directions affect the results.

Table 4 Optimal design comparison for the 120-bar dome truss (four cases)

Element group	Optimal cross-sectional areas (in.^{2})					
	Case 1			Case 2		
	PSO	PSOPC	HPSACO	PSO	PSOPC	HPSACO
1	3.147	3.235	3.311	15.978	3.083	3.779
2	6.376	3.370	3.438	9.599	3.639	3.377
3	5.957	4.116	4.147	7.467	4.095	4.125
4	4.806	2.784	2.831	2.790	2.765	2.734
5	0.775	0.777	0.775	4.324	1.776	1.609
6	13.798	3.343	3.474	3.294	3.779	3.533
7	2.452	2.454	2.551	2.479	2.438	2.539
Weight (lb)	32432.9	19618.7	19491.3	41052.7	20681.7	20078.0
	Case 3			Case 4		
	PSO	PSOPC	HPSACO	PSO	PSOPC	HPSACO
1	1.773	2.098	2.034	12.802	3.040	3.095
2	17.635	16.444	15.151	11.765	13.149	14.405
3	7.406	5.613	5.901	5.654	5.646	5.020
4	2.153	2.312	2.254	6.333	3.143	3.352
5	15.232	8.793	9.369	6.963	8.759	8.631
6	19.544	3.629	3.744	6.492	3.758	3.432
7	0.800	1.954	2.104	4.988	2.502	2.499
Weight (lb)	46893.5	31776.2	31670.0	51986.2	33481.2	33248.9

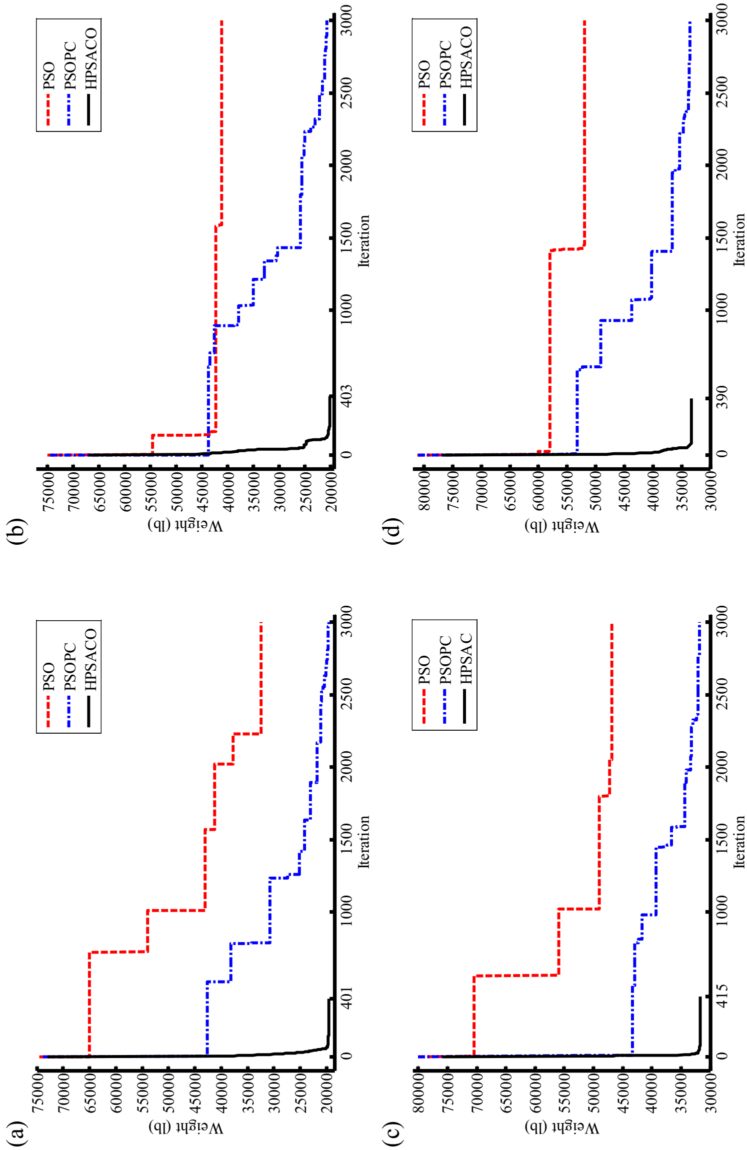


Fig. 10 Comparison of the convergence rates of the three algorithms for the 120-bar truss (a) Case 1 (b) Case 2 (c) Case 3 (d) Case 4

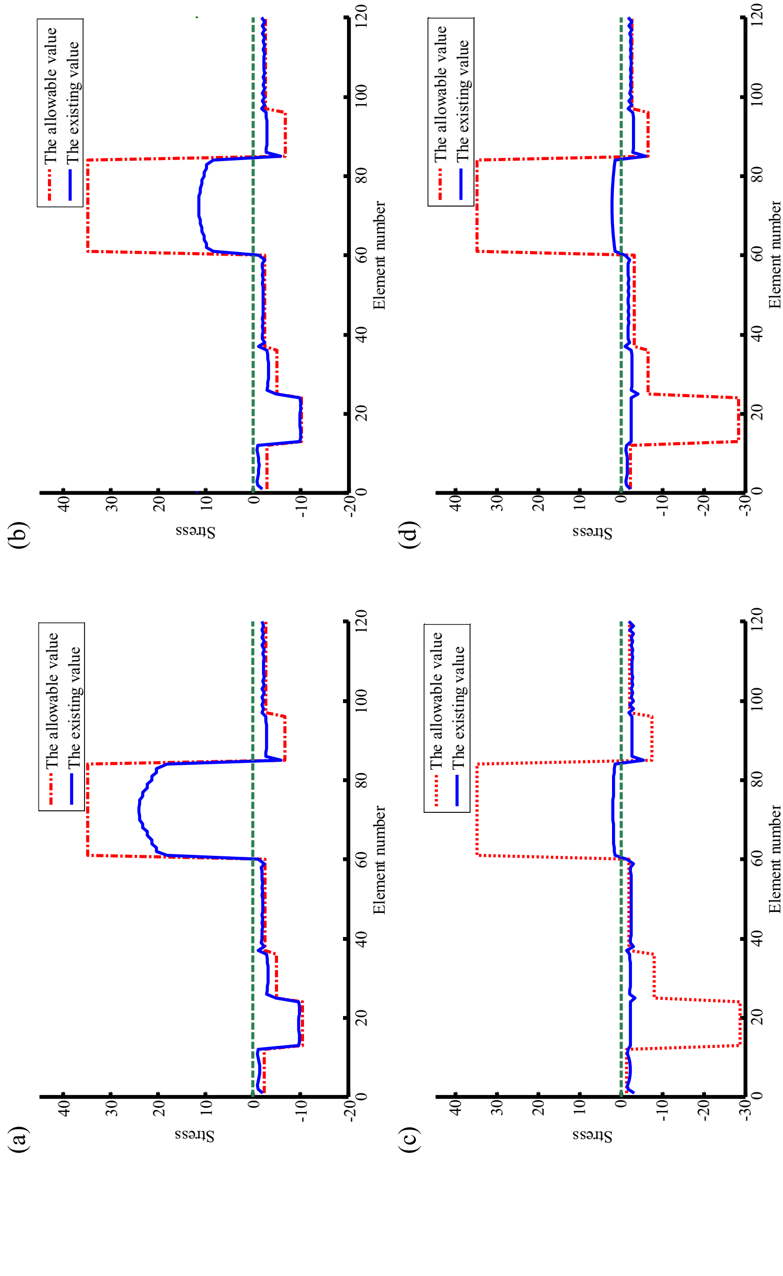


Fig. 11 Comparison of the allowable and existing stresses in the elements of the 120-bar truss (a) Case 1 (b) Case 2 (c) Case 3 (d) Case 4

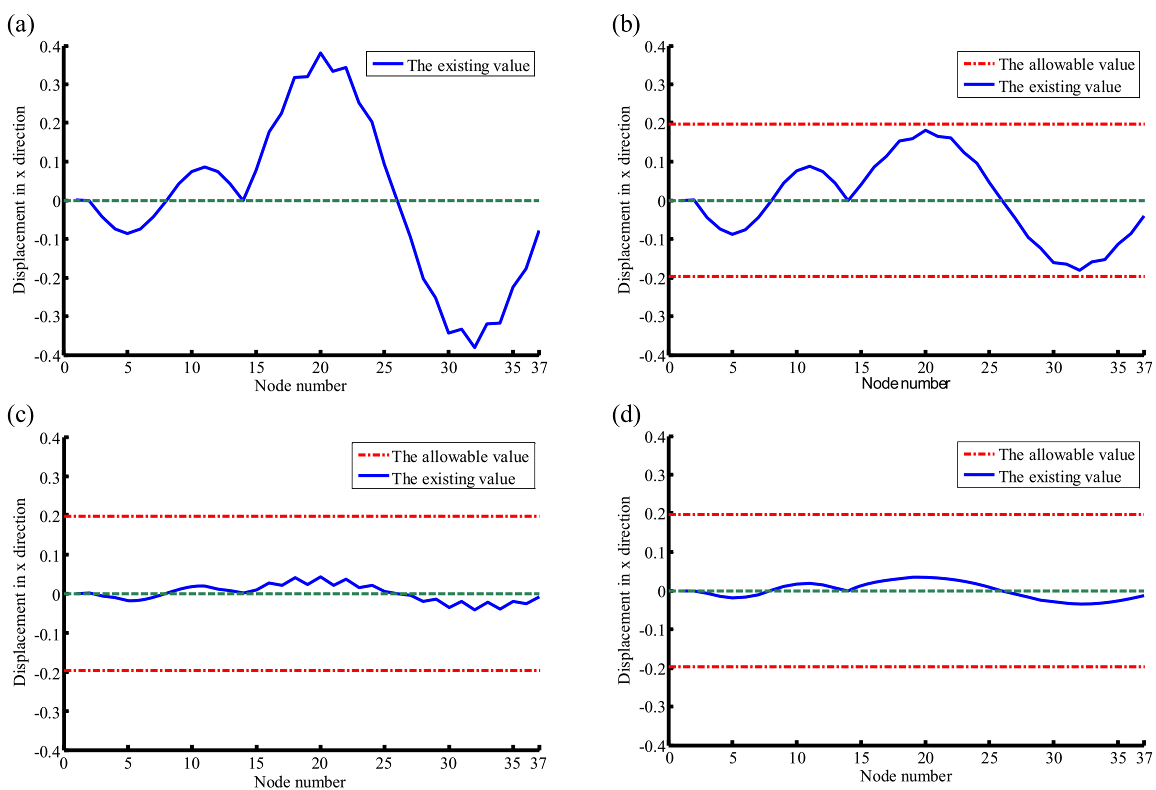


Fig. 12 Comparison of the allowable and existing nodal displacements in the x direction of the 120-bar truss (a) Case 1 (b) Case 2 (c) Case 3 (d) Case 4

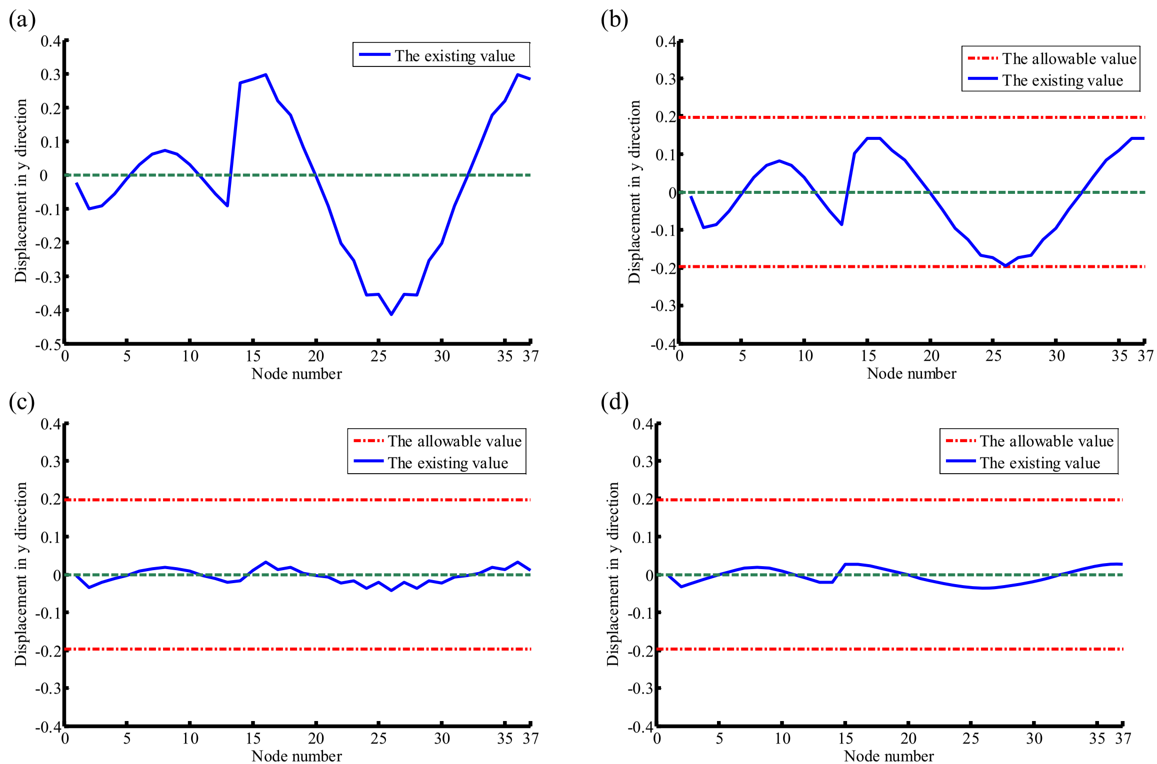


Fig. 13 Comparison of the allowable and existing nodal displacements in the y direction of the 120-bar truss (a) Case 1 (b) Case 2 (c) Case 3 (d) Case 4

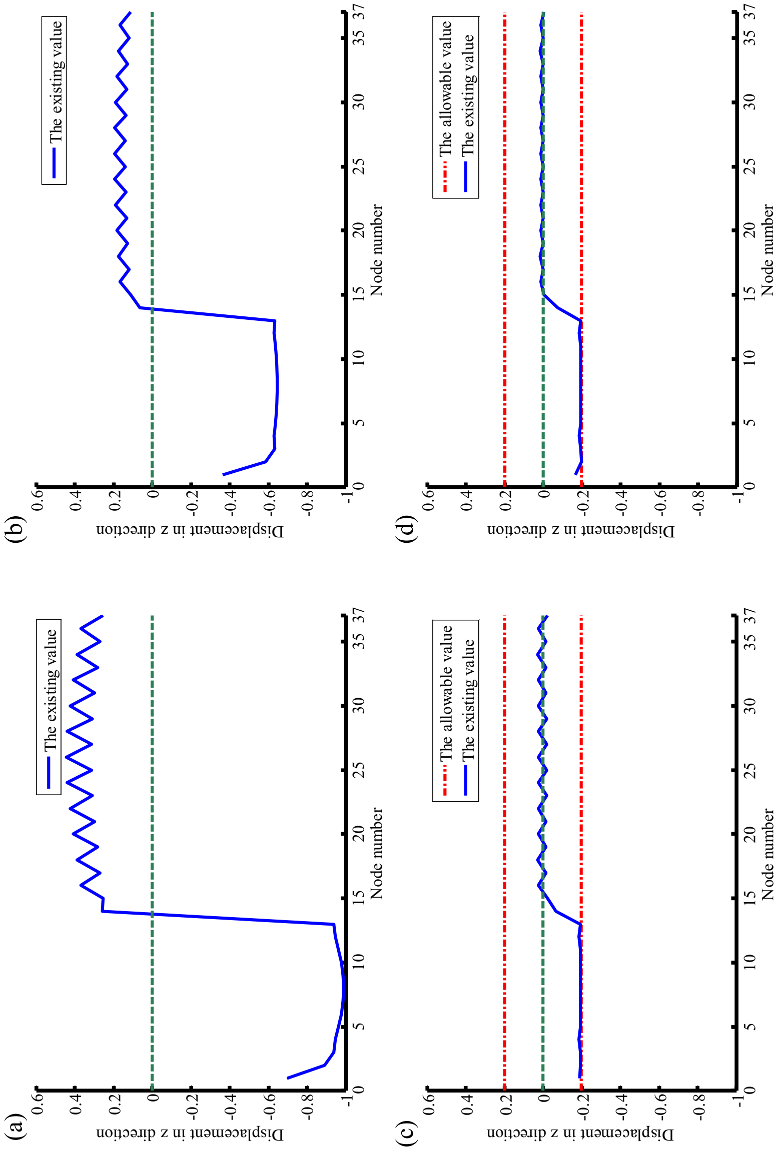


Fig. 14 Comparison of the allowable and existing nodal displacements in the z direction of the 120-bar truss (a) Case 1 (b) Case 2 (c) Case 3 (d) Case 4

6.2 A Truss Structure with Discrete Design Variables

A 582-bar tower truss shown in Figure 15 with an 80 m height is chosen from [65] as an example of truss structure with discrete design variables. The symmetry of the tower around *x*- and *y*-axes is considered to group the 582 members into 32 independent size variables. A single load case is considered consisting of lateral loads of 5.0 kN (1.12 kips) applied in both *x*- and *y*-directions and a vertical load of −30 kN (−6.74 kips) applied in the *z*-direction at all nodes of the tower. A discrete set of 137 economical standard steel sections selected from the W-shape profile list based on area and radii of gyration properties is used to size the variables [65]. The lower and upper bounds on size variables are taken as 6.16 in.^{2} (39.74 cm^2) and 215.0 in.^{2} (1387.09 cm^2), respectively. The stress limitations of the members are imposed according to the provisions of ASD-AISC, as in the previous example. The other constraint is the limitation of node displacements (no more than 8.0 cm or 3.15 in. in any direction). In addition, the maximum slenderness ratio is limited to 300 and 200 for tension and compression members, respectively.

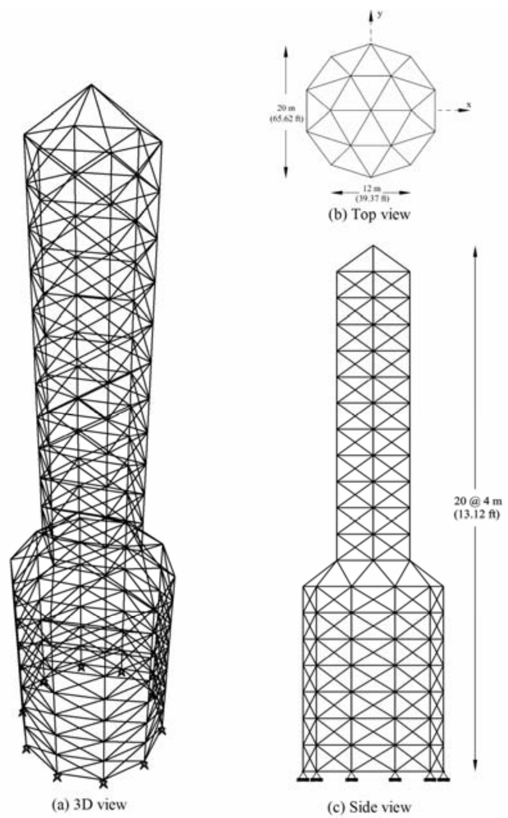


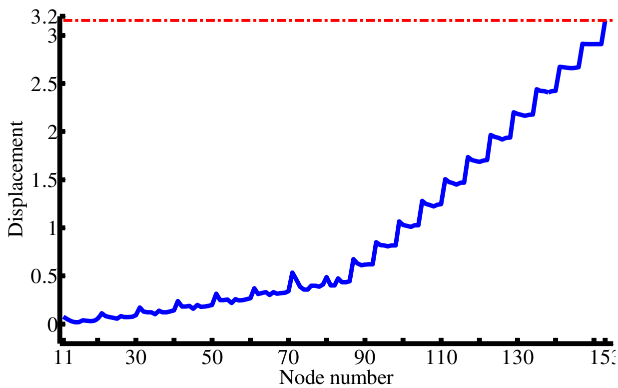
Fig. 15 A 582-bar tower truss (a) 3D view (b) Top view (c) Side view

Table 5 Optimal design comparison for the 582-bar truss tower

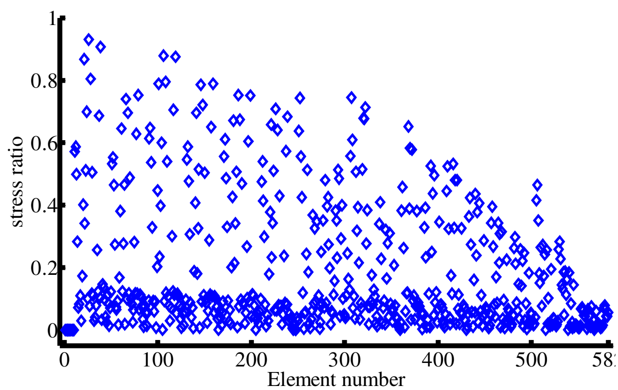
Element group	Optimal W-shaped sections			
	PSO [65]		HPSACO	
	Ready section	Area, cm^2 ($in.^2$)	Ready section	Area, cm^2 ($in.^2$)
1	W8×21	39.74 (6.16)	W8×24	45.68 (7.08)
2	W12×79	149.68 (23.2)	W12×72	136.13 (21.1)
3	W8×24	45.68 (7.08)	W8×28	53.16 (8.24)
4	W10×60	113.55 (17.08)	W12×58	109.68 (17)
5	W8×24	45.68 (7.08)	W8×24	45.68 (7.08)
6	W8×21	39.74 (6.16)	W8×24	45.68 (7.08)
7	W8×48	90.97 (14.1)	W10×49	92.90 (14.4)
8	W8×24	45.68 (7.08)	W8×24	45.68 (7.08)
9	W8×21	39.74 (6.16)	W8×24	45.68 (7.08)
10	W10×45	85.81 (13.3)	W12×40	75.48 (11.7)
11	W8×24	45.68 (7.08)	W12×30	56.71 (8.79)
12	W10×68	129.03 (20)	W12×72	136.129 (21.1)
13	W14×74	140.65 (21.8)	W18×76	143.87 (23.3)
14	W8×48	90.97 (14.1)	W10×49	92.90 (14.4)
15	W18×76	143.87 (22.3)	W14×82	154.84 (24)
16	W8×31	55.90 (9.13)	W8×31	58.84 (9.12)
17	W8×21	39.74 (6.16)	W14×61	115.48 (17.9)
18	W16×67	127.10 (19.7)	W8×24	45.68 (7.08)
19	W8×24	45.68 (7.08)	W8×21	39.74 (6.16)
20	W8×21	39.74 (6.16)	W12×40	75.48 (11.7)
21	W8×40	75.48 (11.7)	W8×24	45.68 (7.08)
22	W8×24	45.68 (7.08)	W14×22	41.87 (6.49)
23	W8×21	39.74 (6.16)	W8×31	58.84 (9.12)
24	W10×22	41.87 (6.49)	W8×28	53.16 (8.24)
25	W8×24	45.68 (7.08)	W8×21	39.74 (6.16)
26	W8×21	39.74 (6.16)	W8×21	39.74 (6.16)
27	W8×21	39.74 (6.16)	W8×24	45.68 (7.08)
28	W8×24	45.68 (7.08)	W8×28	53.16 (8.24)
29	W8×21	39.74 (6.16)	W16×36	68.39 (10.6)
30	W8×21	39.74 (6.16)	W8×24	45.68 (7.08)
31	W8×24	45.68 (7.08)	W8×21	39.74 (6.16)
32	W8×24	45.68 (7.08)	W8×24	45.68 (7.08)
Volume	22.3958 m^3 (1366674.89 in^3)		22.0607 m^3 (1346227.65 in^3)	

PSO has obtained the lightest design when compared to some other meta-heuristic algorithms such as evolution strategies, simulated annealing, tabu search, ant colony optimization, harmony search and genetic algorithms reported by Hasançebi *et al.* [65]. Table 5 gives the best solution vectors of the PSO and HPSACO algorithms [66]. The optimum result of the HPSACO approach is 22.06 m^3 while it is 22.39 m^3 for the PSO algorithm. HPSACO needs nearly 8,500 analyses to reach a solution which is significantly less than 50,000 analyses for PSO.

Figure 16 compares the allowable and existing stress ratio and displacement value in the x direction of the HPSACO result. The maximum values of displacements in the *x*, *y* and *z* directions are 3.1498 in., 2.9881 in. and 0.9258 in., respectively. The maximum stress ratio is 93.06% as show in the figure.



(a)



(b)

Fig. 16 Comparison of the allowable and the existing constraints for the 582-bar truss using the HPSACO (a) displacement in the x direction (b) stress ratio

6.3 A Steel Frame Structure

Figure 17 shows the configuration and applied loads of 3-bay 15-story frame structure. The displacement and AISC combined strength constraints are the performance constraint of the frame. The sway of the top story is limited to 9.25 in. (23.5 cm). The material has a modulus of elasticity E=29,000 ksi (200,000 MPa) and a yield stress of f_y=36 ksi (248.2 MPa). The effective length factors of the members are calculated as $K_x \geq 0$ for a sway-permitted frame and the out-of-plane effective length factor is specified as K_y=1.0. Each column is considered as unbraced along its length, and the unbraced length for each beam member is specified as one-fifth of the span length.

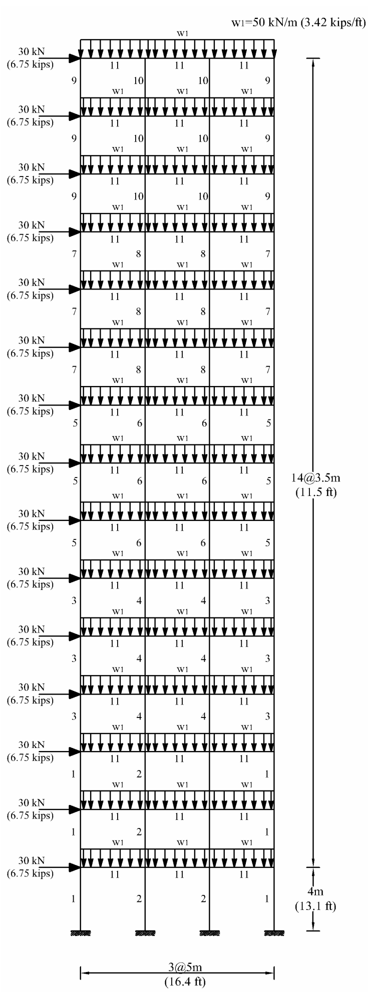
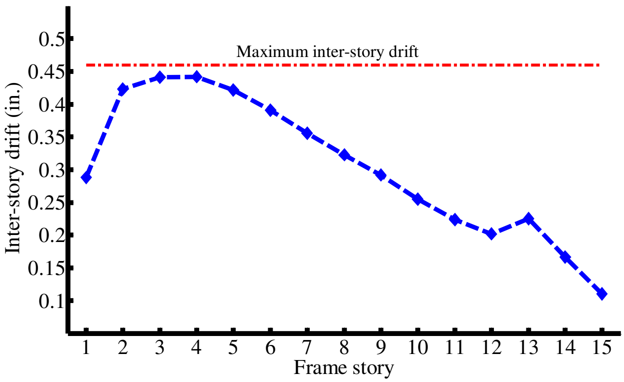


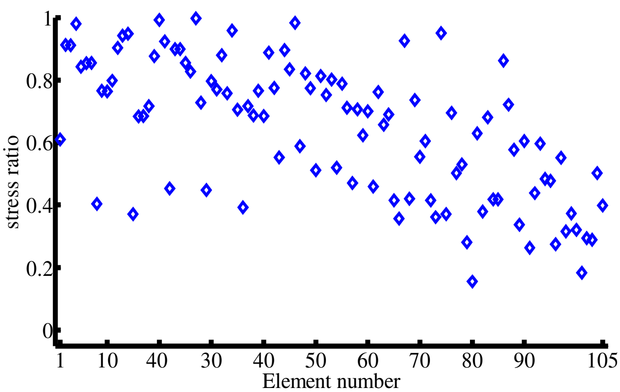
Fig. 17 Topology of the 3-bay 15-story frame

The optimum design of the frame is obtained after 6,800 analyses by using HPSACO, having the minimum weight of 426.36 kN (95.85 kips). The optimum designs for PSOPC and PSO had the weight of 452.34 kN (101.69 kips) and 496.68 kN (111.66 kips), respectively. Table 6 summarizes the optimal designs for these algorithms.

The global sway at the top story is 11.57 cm (4.56 in.), which is less than the maximum sway. Figure 18 shows the inter-story drift for each story and the stress ratio of elements for the design of the HPSACO algorithm.



(a)



(b)

Fig. 18 Comparison of the allowable and the existing constraints for the 3-bay 15-story frame using the HPSACO (a) inter-story drift (b) stress ratio

Table 6 Optimal design comparison for the 3-bay 15-story frame

Element group	Optimal W-shaped sections		
	PSO	PSOPC	HPSACO
1	W33×118	W26×129	W21×111
2	W33×263	W24×131	W18×158
3	W24×76	W24×103	W10×88
4	W36×256	W33×141	W30×116
5	W21×73	W24×104	W21×83
6	W18×86	W10×88	W24×103
7	W18×65	W14×74	W21×55
8	W21×68	W26×94	W26×114
9	W18×60	W21×57	W10×33
10	W18×65	W18×71	W18×46
11	W21×44	W21×44	W21×44
Weight kN (kips)	496.68 (111.66)	452.34 (101.69)	426.36 (95.85)
The global sway (cm)	10.42	11.36	11.57
Max. stress ratio	99.54%	99.57%	99.72%

7 Summary and Conclusions

Structural design optimization is a critical and challenging activity that has received considerable attention in the last decades. Despite the existing factors that prevent performing optimum design, designers and owners have always desired to have optimal structures. To fulfill this aim, several classical methods and heuristic approaches have been developed. The drawbacks of the classical optimization methods consist of complex derivatives, sensitivity to initial values, and the large amount of their enumeration memory required. Thus the advantages of heuristic algorithms have caused a considerable increase in applying heuristic methods such as genetic algorithms (GAs), particle swarm optimizer (PSO), ant colony optimization (ACO) and harmony search (HS). Heuristic methods are quite suitable and powerful for obtaining the solution of optimization problems. These methods have attracted a great deal of attention because of their high potential for modeling engineering problems in environments which have been resistant to solutions by classic techniques.

There are several papers utilizing heuristic methods in the field of structural optimization, but using an individual heuristic method has often had some drawbacks because usually each method is suitable for solving only a specific group of problems and preference for a special method will differ depending on the kind of the problem being studied. One technique overcome to these problems is to hybridize various methods to reach a single robust approach. In this chapter, a new hybridized approach based on HS, PSO and ACO is presented which is called the heuristic particle swarm ant colony optimization (HPSACO).

HPSACO utilizes a particle swarm optimization with a passive congregation (PSOPC) algorithm as a global search, and the idea of an ant colony approach worked as a local search which updates the positions of the particles by applied pheromone-guided mechanism. This principle is used in the HPSACO as a helping factor to guide the exploration and to increase the control of exploitation. In the HPSACO algorithm, fly-back mechanism and the harmony search are used to handle the constraints. Fly-back mechanism handles the problem-specific constraints, and the HS deals with the variable constraints. A particle in the search space may violate either the problem-specific constraints or the limits of the variables. Since the fly-back mechanism is used to handle the problem-specific constraints, the particle will be forced to fly back to its previous position regardless whether it violates the problem-specific constraints. If it flies out of the variable boundaries, the solution cannot be used even if the problem-specific constraints are satisfied. Although minimizing the maximum value of the velocity can make fewer particles violate the variable boundaries, it may also prevent the particles from crossing the problem-specific constraints and can cause the reduction in exploration. Using a harmony search based handling approach, this problem is dealt with. According to this approach, any component of the solution vector violating the variable boundaries can be regenerated from harmony memory.

In optimization problems, and particularly in structural optimization, the number of iterations is highly important. The terminating criterion is a part of the search process which can be used to eliminate additional unnecessary iterations. HPSACO utilizes an efficient terminating criterion considering exactitude of the solutions. This terminating criterion is defined in a way that after decreasing the movements of particles, the search process stops. When the variation of a variable is less than a determined exactitude, this criterion deletes it from the virtual list of variables. When this list becomes empty, the search process stops. Using this terminating criterion, the number of required iterations decreases.

Some changes are made in order to reach a discrete version of HPSACO. In the discrete method, agents are allowed to select discrete values from the permissible list of cross sections, and if any one of agents selects another value for a design variable, the discrete HPSACO changes the amount of it with the value of the nearest discrete cross section. Although this change is simple and efficient, its effect may be to reduce exploration of the algorithm. Therefore, the formula of particles' velocity is improved by adding an exploration term.

In order to find an optimum design for different types of structures, the implementation of the HPSACO methodology is investigated. We start with truss structures considering a continuous domain as the search space. The second problem contains a large-scale truss structure with a discrete search space. Then, the efficiency of the HPSACO algorithm is investigated to find optimum design of frame structures. The results confirm that the HPSACO algorithm is quite effective in finding the optimum design of structures and can be successfully applied to structural optimization problems with continuous or discrete variables.

References

1. Lee, K.S., Geem, Z.W.: A new structural optimization method based on the harmony search algorithm. Computers and Structures 82, 781–798 (2004)
2. Kaveh, A., Farahmand Azar, B., Talatahari, S.: Ant colony optimization for design of space trusses. International Journal of Space Structures 23(3), 167–181 (2008)
3. Kaveh, A., Talatahari, S.: An improved ant colony optimization for constrained engineering design problems. Engineering Computations (accepted for publication, 2009)
4. Goldberg, D.E., Samtani, M.P.: Engineering optimization via genetic algorithm, in Will KM (Eds). In: Proceeding of the 9th Conference on Electronic Computation, ASCE, February 1986, pp. 471–482 (1986)
5. Hajela, P.: Genetic search - an approach to the non-convex optimization problem. AIAA Journal 28(71), 1205–1210 (1990)
6. Rajeev, S., Krishnamoorthy, C.S.: Discrete optimization of structures using genetic algorithms. Journal of Structural Engineering, ASCE 118(5), 1233–1250 (1992)
7. Rajeev, S., Krishnamoorthy, C.S.: Genetic algorithm-based methodologies for design optimization of trusses. Journal of Structural Engineering, ASCE 123(3), 350–358 (1997)
8. Koumousis, V.K., Georgious, P.G.: Genetic algorithms in discrete optimization of steel truss roofs. Journal of Computing in Civil Engineering, ASCE 8(3), 309–325 (1994)
9. Hajela, P., Lee, E.: Genetic algorithms in truss topological optimization. International Journal of Solids and Structures 32(22), 3341–3357 (1995)
10. Wu, S.-J., Chow, P.-T.: Integrated discrete and configuration optimization of trusses using genetic algorithms. Computers and Structures 55(4), 695–702 (1995)
11. Soh, C.K., Yang, J.: Fuzzy controlled genetic algorithm search for shape optimization. Journal of Computing in Civil Engineering, ASCE 10(2), 143–150 (1996)
12. Camp, C., Pezeshk, S., Cao, G.: Optimized design of two dimensional structures using a genetic algorithm. Journal of Structural Engineering, ASCE 124(5), 551–559 (1998)
13. Shrestha, S.M., Ghaboussi, J.: Evolution of optimization structural shapes using genetic algorithm. Journal of Structural Engineering, ASCE 124(11), 1331–1338 (1998)
14. Pezeshk, S., Camp, C.V., Chen, D.: Design of nonlinear framed structures using genetic optimization. Journal of Structural Engineering, ASCE 126(3), 382–388 (2000)
15. Erbatur, F., Hasancebi, O., Tutuncil, I., Kihc, H.: Optimal design of planar and structures with genetic algorithms. Computers and Structures 75, 209–224 (2000)
16. Coello, C.A., Christiansen, A.D.: Multiobjective optimization of trusses using genetic algorithms. Computers and Structures 75(6), 647–660 (2000)
17. Greiner, D., Winter, G., Emperador, J.M.: Optimising frame structures by different strategies of genetic algorithms. Finite Elements in Analysis and Design 37(5), 381–402 (2001)
18. Kameshki, E.S., Saka, M.P.: Optimum design of nonlinear steel frames with semi-rigid connections using a genetic algorithm. Computers and Structures 79, 1593–1604 (2001)
19. Kameshki, E.S., Saka, M.P.: Genetic algorithm based optimum bracing design of non-swaying tall frames. Journal of Constructional Steel Research 57, 1081–1097 (2001)
20. Kameshki, E.S., Saka, M.P.: Optimum geometry design of nonlinear braced domes using genetic algorithm. Computers and Structures 85(1-2), 71–79 (2007)
21. Saka, M.P.: Optimum design of pitched roof steel frames with haunched rafters by genetic algorithm. Computers and Structures 81(18-19), 1967–1978 (2003)
22. Saka, M.P.: Optimum topological design of geometrically nonlinear single layer latticed domes using coupled genetic algorithm. Computers and Structures 85(21-22), 1635–1646 (2007)

23. Kaveh, A., Kalatjari, V.: Genetic algorithm for discrete sizing optimal design of trusses using the force method. International Journal for Numerical Methods in Engineering 55, 55–72 (2002)
24. Kaveh, A., Kalatjari, V.: Topology optimization of trusses using genetic algorithm, force method, and graph theory. International Journal for Numerical Methods in Engineering 58(5), 771–791 (2003)
25. Kaveh, A., Abditehrani, A.: Design of frames using genetic algorithm, force method and graph theory. International Journal for Numerical Methods in Engineering 61, 2555–2565 (2004)
26. Kaveh, A., Khanlari, K.: Collapse load factor of planar frames using a modified Genetic algorithm. Communications in Numerical Methods in Engineering 20, 911–925 (2004)
27. Kaveh, A., Rahami, H.: Analysis, design and optimization of structures using force method and genetic algorithm. International Journal for Numerical Methods in Engineering 65(10), 1570–1584 (2006)
28. Kaveh, A., Shahrouzi, M.: Simultaneous topology and size optimization of structures by genetic algorithm using minimal length chromosome. Engineering Computations 23(6), 664–674 (2006)
29. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan (1995)
30. Kennedy, J., Eberhart, R.C., Shi, Y.: Swarm intelligence. Morgan Kaufman Publishers, San Francisco (2001)
31. Schutte, J.J., Groenwold, A.A.: Sizing design of truss structures using particle swarms. Structural and Multidisciplinary Optimization 25, 261–269 (2003)
32. Li, L.J., Huang, Z.B., Liu, F., Wu, Q.H.: A heuristic particle swarm optimizer for optimization of pin connected structures. Computers and Structures 85, 340–349 (2007)
33. Perez, R.E., Behdinan, K.: Particle swarm approach for structural design optimization. Computers and Structures 85, 1579–1588 (2007)
34. Kathiravan, R., Ganguli, R.: Strength design of composite beam using gradient and particle swarm optimization. Composite Structures 81(4), 471–479 (2007)
35. Suresh, S., Sujit, P.B., Rao, A.K.: Particle swarm optimization approach for multiobjective composite box-beam design. Composite Structures 81(4), 598–605 (2007)
36. Angeline, P.: Evolutionary optimization versus particle swarm optimization: philosophy and performance difference. In: Proceeding of the Evolutionary Programming Conference, San Diego, USA (1998)
37. Dorigo, M.: Optimization, learning and natural algorithms. PhD thesis, Dip. Elettronica e Informazione, Politecnico di Milano, Italy (1992)
38. Dorigo, M., Maniezzo, V., Colorni, A.: The Ant System: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B 26(1), 1–13 (1996)
39. Camp, C.V., Bichon, J.: Design of space trusses using ant colony optimization. Journal of Structural Engineering, ASCE 130(5), 741–751 (2004)
40. Serra, M., Venini, P.: On some applications of ant colony optimization metaheuristic to plane truss optimization. Structural Multidisciplinary Optimization 32(6), 499–506 (2006)
41. Camp, C.V., Bichon Stovall, J.P.: Design of steel frames using ant colony optimization. Journal of Structural Engineering, ASCE 131, 369–379 (2005)
42. Kaveh, A., Shojaee, S.: Optimal design of skeletal structures using ant colony optimisation. International Journal for Numerical Methods in Engineering 70(5), 563–581 (2007)
43. Kaveh, A., Hassani, B., Shojaee, S., Tavakkoli, S.M.: Structural topology optimization using ant colony methodology. Engineering Structures 30(9), 2259–2265 (2008)
44. Kaveh, A., Shahrouzi, M.: Dynamic selective pressure using hybrid evolutionary and ant system strategies for structural optimization. International Journal for Numerical Methods in Engineering 73(4), 544–563 (2008)

45. Geem, Z.W., Kim, J.H., Loganathan, G.V.: A new heuristic optimization algorithm; harmony search. Simulation 76, 60–68 (2001)
46. Lee, K.S., Geem, Z.W.: A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Computer methods in applied mechanics and engineering 194, 3902–3933 (2005)
47. Lee, K.S., Choi, C.S.: Discrete-continuous configuration optimization methods for structures using the harmony search algorithm. Key Engineering Materials 324-245, 1293–1296 (2006)
48. Saka, M.P.: Optimum geometry design of geodesic domes using harmony search algorithm. Advances in Structural Engineering 10(6), 595–606 (2007)
49. Degertekin, S.O.: Optimum design of steel frames using harmony search algorithm. Structural and Multidisciplinary Optimization 36, 393–401 (2008)
50. Degertekin, S.O.: Harmony search algorithm for optimum design of steel frame structures: a comparative study with other optimization methods. Structural Engineering and Mechanics 29, 391–410 (2008)
51. Saka, M.P.: Optimum design of steel sway frames to BS5950 using harmony search algorithm. Journal of Constructional Steel Research 65(1), 36–43 (2009)
52. He, S., Wu, Q.H., Wen, J.Y., Saunders, J.R., Paton, R.C.: A particle swarm optimizer with passive congregation. Biosystem 78, 135–147 (2004)
53. Deneubourg, J.L., Goss, S.: Collective patterns and decision-making. Ethnology Ecology and Evolution 1, 295–311 (1989)
54. Goss, S., Beckers, R., Deneubourg, J.L., Aron, S., Pasteels, J.M.: How trail laying and trail following can solve foraging problems for ant colonies. In: Hughes, R.N. (ed.) Behavioural mechanisms in food selection, NATO-ASI Series, Berlin, vol. G 20 (1990)
55. American Institute of Steel Construction, AISC, Manual of steel construction–Load resistance factor design, 3rd edn. AISC, Chicago (1991)
56. Shelokar, P.S., Siarry, P., Jayaraman, V.K., Kulkarni, B.D.: Particle swarm and ant colony algorithms hybridized for improved continuous optimization. Applied Mathematics and Computation 188, 129–142 (2007)
57. Kaveh, A., Talatahari, S.: A hybrid particle swarm and ant colony optimization for design of truss structures. Asian Journal of Civil Engineering 9(4), 329–348 (2008)
58. Kaveh, A., Talatahari, S.: A discrete particle swarm ant colony optimization for design of steel frames. Asian Journal of Civil Engineering 9(6), 563–575 (2008)
59. Kaveh, A., Talatahari, S.: Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures. Computers and Structures 87(5–6), 267–283 (2009)
60. Coello, C.A.C.: Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. Computer Methods in Applied Mechanics and Engineering 191, 1245–1287 (2002)
61. Michalewicz, Z.: Genetic algorithms + data structures = evolution programs, 3rd edn. Springer, Heidelberg (1996)
62. Koziel, S., Michalewicz, Z.: Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. Evolutionary Computation 7(1), 19–44 (1999)
63. Liepins, G.E., Vose, M.D.: Representational issues in genetic optimization. Journal of Experimental and Theoretical Computer Science 2(2), 4–30 (1990)
64. He, S., Prempain, E., Wu, Q.H.: An improved particle swarm optimizer for mechanical design optimization problems. Engineering Optimization 36(5), 585–605 (2004)
65. Hasançebi, O., Çarbas, S., Dogan, E., Erdal, F., Saka, M.P.: Performance evaluation of metaheuristic search techniques in the optimum design of real size pin jointed structures. Computers and Structures 87(5–6), 284–302 (2009)
66. Kaveh, A., Talatahari, S.: A particle swarm ant colony optimization for truss structures with discrete variables. Journal of Constructional Steel Research (2009)