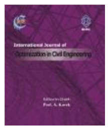
INTERNATIONAL JOURNAL OF OPTIMIZATION IN CIVIL ENGINEERING
Int. J. Optim. Civil Eng., 2020; 10(2):231-260



OPTIMAL SIZE AND GEOMETRY DESIGN OF TRUSS STRUCTURES UTILIZING SEVEN META-HEURISTIC ALGORITHMS: A COMPARATIVE STUDY

A. Kaveh[*, †,1], K. Biabani Hamedani[1] and F. Barzinpour[2]

[1]*School of Civil Engineering, Iran University of Science and Technology, Tehran, Iran*
[2]*School of Industrial Engineering, Iran University of Science and Technology, Tehran, Iran*

ABSTRACT

Meta-heuristic algorithms are applied in optimization problems in a variety of fields, including engineering, economics, and computer science. In this paper, seven population-based meta-heuristic algorithms are employed for size and geometry optimization of truss structures. These algorithms consist of the Artificial Bee Colony algorithm, Cyclical Parthenogenesis Algorithm, Cuckoo Search algorithm, Teaching-Learning-Based Optimization algorithm, Vibrating Particles System algorithm, Water Evaporation Optimization, and a hybridized ABC-TLBO algorithm. The Taguchi method is employed to tune the parameters of the meta-heuristics. Optimization aims to minimize the weight of truss structures while satisfying some constraints on their natural frequencies. The capability and robustness of the algorithms is investigated through four well-known benchmark truss structure examples.

Keywords: structural optimization; optimal design; meta-heuristic algorithms; truss structures; natural frequency; Taguchi method.

Received: 5 December 2019; Accepted: 10 February 2020

1. INTRODUCTION

Optimization approaches can be classified into two main categories, including deterministic approaches, and meta-heuristics. In deterministic approaches, also known by mathematical programming, utilizing analytical properties of the problem, a sequence of points converging to a global optimum is generated. Due to difficulties of the

---

[*] Corresponding author: School of Civil Engineering, Iran University of Science and Technology, Narmak, Tehran, P.O. Box 16846-13114, Iran
[†]E-mail address: alikaveh@iust.ac.ir (A. Kaveh)

deterministic approaches, meta-heuristics are getting more attention. In this regard, numerous meta-heuristic algorithms have been developed as efficient tools for solving complicated optimization problems. Meta-heuristics have found many applications in various disciplines such as engineering, economics, and medicine. Meta-heuristics can be classified based on several criteria. The most common classification of meta-heuristics is population-based optimization versus single-solution-based optimization [1]. In the population-based meta-heuristics, a set of solutions is generated and spread over the design space. Next, an iterative search procedure continues until a termination criterion is fulfilled.

One of the most challenging issues for structural designers is time and resource management. For this purpose, structural designers can use meta-heuristics as appropriate answer to overcome these problems, especially for complicated structures. Truss structures are among the most common structures in the structural engineering and have numerous applications in construction industry, including bridges, roofs, and industrial buildings. Therefore, economical and safe design of these structures is very significant. In general, truss structure optimization problems can be classified into three different categories of size optimization, geometry optimization, and topology optimization. In most structural optimization problems, either a dual combination of these three areas is used or all three areas are considered. Truss optimization under frequency constraints gives the ability to a designer to control the selected frequencies in a desire fashion in order to improve the dynamic characteristics of the structure. So truss optimization with frequency constraints has been receiving considerable attention in the past decades. Extensive efforts have been put into this field and remarkable achievements have been made. For instance, Lingyun et al. [2] employed Genetic Algorithm (GA) for optimal sizing and shape design of truss structures with frequency constraints. Kaveh and Talatahari [3] performed optimal design of truss structures with discrete variables using an efficient hybrid algorithm so-called Discrete Heuristic Particle Swarm Ant Colony Optimization (DHPSACO). In 2009, Kaveh and Talatahari [4] employed Big Bang-Big Crunch (BB-BC) algorithm in order to optimize space truss structures. Kaveh and Zolghadr [5] examined the performance of a hybridized CSS-BBBC algorithm on optimum design of truss structures with natural frequency constraints. Kaveh and Khayatazad [6] utilized Ray Optimization (RO) algorithm in order to optimized size and geometry of truss structures. Kaveh and Zolghadr [7] performed optimal size and layout of truss structures with frequency constraints using Democratic Particle Swarm Optimization (DPSO). In 2014, Kaveh et al. [8] introduced a new method namely Chaotic Swarming of Particles (CSP) and employed it for size optimization of truss structures. Kaveh and Mahdavi [9] used Colliding Bodies Optimization (CBO) algorithm to optimize truss structures with continuous variables. Kaveh et al. [10] applied Dolphin Echolocation Optimization (DEO) algorithm for optimal design of truss structures with natural frequencies. In 2015, Kaveh and Ilchi Ghazaan [11] utilized Improved Ray Optimization (IRO) algorithm to solve truss layout and sizing optimization with multiple natural frequency constraints. Kaveh and Ilchi Ghazaan [12] employed two hybridized optimization algorithm for finding the optimal mass of truss structures with natural frequency constraints. In 2016, Kaveh and Ilchi Ghazaan [13] performed optimal design of truss structures with multiple natural frequency constraints utilizing Vibrating Particles System (VPS) algorithm. Degertekin et al. [14] evaluated the suitability of the Jaya Algorithm (JA) for weight minimization of truss structures.

In this study, the performance of seven different population-based meta-heuristic algorithms is studied in the optimum design of truss structures with multiple natural frequency constraints. These algorithms consist of Artificial Bee Colony (ABC), Cyclical Parthenogenesis Algorithm (CPA), Cuckoo Search (CS), Teaching-Learning-Based Optimization (TLBO), Vibrating Particles System (VPS), Water Evaporation Optimization (WEO), and a hybridized ABC-TLBO. To evaluate the performance of the utilized meta-heuristics, they are applied to optimal design of four well-known benchmark trusses.

The rest of this paper is structured as follows: In Section 2, the utilized meta-heuristic algorithms are presented, and the optimization problem is defined. Section 3 includes four benchmark truss examples. In addition, the results of the Taguchi method are presented in this section. Eventually, the last section concludes the paper.

2. MATERIALS AND METHODS

2.1 Meta-heuristic algorithms

In this paper, seven population-based meta-heuristic optimization algorithms are employed to minimize the weight of truss structures. These algorithms are as follows: Artificial Bee Colony (ABC) algorithm, Big Bang-Big Crunch (BB-BC) algorithm, Cyclical Parthenogenesis Algorithm (CPA), Cuckoo Search (CS) algorithm, Thermal Exchange Optimization (TEO) algorithm, Teaching-Learning-Based Optimization (TLBO) algorithm, Water Evaporation Optimization (WEO), and a hybridized ABC-TLBO algorithm. Kaveh and Bakhshpoori [1] coded the original version of the first six algorithms and characterized their properties. The meta-heuristics are presented briefly in the following sections.

2.1.1 Artificial bee colony algorithm (ABC)

The Artificial Bee Colony (ABC) algorithm, has been introduced by Karaboga in 2005 [15], uses the foraging behaviour of the honey bees. In this algorithm, each candidate solution is represented by a food source, and its nectar quality corresponds to the objective function of that solution. These food sources are modified by honey bees in a repeated manner aiming to reach food sources with better nectar. In ABC, there are three types of honey bees: employed or recruited, onlooker, and scout bees, with each having different responsibilities. Bees perform modification with different techniques according to their duties. In each iteration, the ABC algorithm searches in three sequential phases. Employed bees modify the food sources and share their information with onlooker bees. Onlooker bees select a food source based on the information from employed bees and try to modify it. Scout bees perform random searches in the vicinity of the hive.

After randomly generating initial bees, iterative process of the algorithm starts until stopping criterion is achieved. Each iteration is composed of three sequential phases. In the first phase which is known as employed or recruited phase, bees search for new food sources based on the information of the individual understandings. In the second phase or onlooker phase, all employed bees share their information of food sources (position and nectar quality) with onlookers in the dance area. The most promising food source is selected by the onlookers based on a selection probability scheme such as the fitness proportionate selection scheme. More onlookers get attracted toward superlative food sources. It should be noted that the number of onlookers is the same as the employed bees and both are the same as the number of food sources around the hive. In other words, every bee whether employee or onlooker corresponds to one food source. The third phase (the scout bee phase) starts if a food source cannot be further improved for a predefined number of trials. In this phase, the food source had to be deserted, and its coupled employed bee transformed into a scout bee. The abandoned food sources are replaced with the randomly generated new ones by the scout bees in the search space. In the following, these phases are formulated as follows:

1. Generation of new honey bees ($newHB$) based on the recruited or employed bees strategy. Each employed bee attempts to find a new better food source by searching around its corresponding food source with a random permutation-based step size toward a randomly selected other food source except for herself. This phase can be stated mathematically as:

$$\begin{gathered} stepsize = rand_{(i)(j)} \times (HB - HB[permute(i)(j)]) \\ newHB = HB + stepsize \end{gathered} \tag{1}$$

where $rand_{(i)(j)}$ is a random number chosen from the continuous uniform distribution on the [-1, 1] interval, $permute$ is different rows permutation functions, i is the number of honey bees, and j is the number of dimensions of the problem. This phase enables the ABC in the aspect of diversification so that each bee attempts to search its own neighborhood. It should be noted that this search takes place over large steps at the beginning of the algorithm and gradually it gets smaller as the population approaches each other with the completion of the algorithm process.

2. Generate new honey bees ($newHB$) based on the onlooker bees strategy. After completing search process of all the employed bees, share their information (nectar quality and position) of corresponding food sources with onlooker bees. The numbers of employed and onlooker bees are the same. Each onlooker bee is attracted by an employed bee with the probability P_i, and she selects a food source associated with that employed bee to generate new food source for possible modification. It seems that onlooker bees are more attracted with food sources with better nectar quality. A selection probability scheme such as the fitness proportionate selection or roulette wheel selection scheme is used in the ABC calculated by the following expression:

$$P_i = PFit_i / \sum_{i=1}^{nHB} PFit_i \tag{2}$$

in which $PFit_i$ is the penalized objective function of the i-th food source. After choosing a food source (HB_{rws}) based on the roulette wheel selection scheme by the i-th onlooker bee, a neighborhood source is determined by adding a permutation-based random step wise toward a randomly selected food source except herself:

$$stepsize = rand_{(i)(j)} \times (HB_{rws} - HB[permute(i)(j)])$$
$$newHB = \begin{cases} HB_{rws} + stepsize, & \text{if } rand < mr \\ HB_{rws}, & \text{otherwise} \end{cases} \tag{3}$$

where $rand_{(i)(j)}$ is a random number chosen from the continuous uniform distribution on the [-1, 1] interval, $permute$ is different rows permutation functions, i is the number of honey bees, and j is the number of dimensions of the problem. Another parameter, modification rate (mr), is defined in the version of the ABC algorithm for constrained optimization as a control parameter that controls whether the selected food source by onlooker bee will be modified or not. $Rand$ is a randomly chosen real number in the range [0, 1]. This phase ensures the intensification capability of the algorithm so that onlooker bees prefer further to explore the neighborhood of the superlative food sources.

3. In scout bee phase, employed bees who cannot modify their food sources after a specified number of trials (A) become scouts. The corresponding food source will be abandoned, and a random-based new food source will be generated in the vicinity of the hive. This phase merely produces diversification and allows to have new and probability infeasible candidate solutions. It sounds that this phase will be active in the near to end cyclic process of the algorithm.

2.1.2 Cyclical Parthenogenesis Algorithm (CPA)

Cyclical Parthenogenesis Algorithm (CPA) is developed by Kaveh and Zolghadr [16]. This algorithm is inspired by the social behaviour and reproduction of zoological species like aphids. Each candidate solution in this algorithm is considered as an aphid and the candidates are grouped into several colonies with equal numbers of aphids each inhabiting a host plant. Each colony iteratively tries to improve the quality of its aphids by reproduction mechanisms with and without mating with a chance to get merit from other colonies using an information exchange mechanism. The role (female or male) of each aphid in each colony is determined depending on their quality. Each colony reproduces independently to improve the position of its aphids in the search space. In order to prevent the reproduction of colonies independently to benefit the winged aphids, colonies can exchange a level of information between themselves. Colony improvements exchange between them and information are repeated in the cyclic body of the algorithm to fulfil the stopping criterions in order to direct each colony toward a better position in the search space. The rules of CPA are stated at the following:

Rule 1 (Initialization): CPA starts from a set of candidate solutions or aphids randomly generated within the search space. The number of aphids is considered as nA. These aphids are grouped into nC number of colonies with the same number of members or aphids (nM). The concept of multiple colonies allows CPA to search different portions of the search space more or less independently and prevents the unwanted premature convergence phenomenon. For coding CPA in a simple manner and to be easy for tracing, the colonized aphids are determined by a cell array (CA). Therefore, CA is an array of nC colonies with nM aphids.

After evaluation of the initial population or the colonized aphids, the corresponding objective function (Fit) and penalized objective function ($PFit$) cells are produced. According to this rule, nM is not considered as a population parameter of the algorithm, so that can be calculated using two population parameters of the algorithm: $nM = nA/nC$. It should be noted that CPA considers nM unchanged in the optimization procedure.

Rule 2 (Reproduction or Parthenogenesis of Aphids): In each iteration, nM new candidate solutions or offspring are generated in each of the colonies. These new solutions can be reproduced either with or without mating. A ratio Fr of the best of the new solutions of any colony are considered as female aphids; the rest are considered as male aphids. Therefore in each colony, $Fr \times nM$ number of offspring will be reproduced without mating, and $(1 - Fr) \times nM$ number of offspring will be reproduced with mating. Altogether nM number of offspring will be reproduced. For reproducing $Fr \times nM$ number of offspring without mating, a female parent (F) is selected randomly from the female aphids of the colony for $Fr \times nM$ times. Then, this randomly selected female parent reproduces a new offspring without mating by the following expression:

$$newCA = F + \alpha_1 \times \frac{randn}{NITs} \times (Ub - Lb) \tag{4}$$

where $randn$ is a random number drawn from a normal distribution, $NITs$ is the current number of algorithm iteration, and α_1 is a scaling parameter for controlling step size of searching. In order to reproduce $(1 - Fr) \times nM$ number of offspring, each of the male aphids (M) selects a female aphid (F) randomly in order to produce an offspring through mating:

$$newCA = M + \alpha_2 \times rand \times (F - M) \tag{5}$$

where $rand$ is a random number uniformly distributed within (0, 1) interval and α_2 is a scaling parameter for controlling searching step size. It can be seen that in this type of reproduction, two different solutions share information, while when reproduction occurs without mating, the new solution is generated using merely the information of one single parent solution.

Rule 3 (Death and Flight): When all of the new solutions or offspring of all colonies are generated and the objective function values are evaluated, flying occurs with a probability of Pf where two of the colonies are selected randomly and named as $colony1$ and $colony2$. A winged aphid is reproduced by and identical to the best female of $colony1$ and then flies to $colony2$. In order to keep the number of members of each colony constant, it is assumed that the worst member of $colony2$ dies. Parameter Pf is responsible for defining the level of information exchange among the colonies. With no possible flights ($Pf = 0$), the colonies would be performing their search in a completely independent manner, i.e., an optimization runs with nA aphids divided into nC colonies would be similar to nC-independent runs each with $nM = nA/nC$ aphids in one colony. It is obvious that this would not be particularly favorable since it is, in fact, changing the population of aphids without actually utilizing the abovementioned benefits of the multiple colonies. On the other hand, permitting too many flights ($Pf = 1$) results in the same effect by merging the information sources of different colonies. It is important to note that at the early stages of the optimization process, it is more favorable to give the colonies a higher level of independence so that they can search the problem space without being affected by the other colonies. However, as the optimization process proceeds, it is desirable to let the colonies share more information so as to provide the opportunity for the more promising regions of the search space to be searched thoroughly. Considering Pf linearly increasing from 0 to 1 results in the best performance of the algorithm, since it conforms to the abovementioned discussion on information circulation:

$$Pf = (NITs - 1)/(maxNITs - 1) \tag{6}$$

Rule 4 (Updating the Colonies or the Replacement Strategy): Considering the fact that the aphids of each colony are capable of reproducing a genetically identical offspring without mating, CPA compares the newly generated set of offspring based on Rule 2 for each colony with the current position of the colony and transmits the nM best ones for the next iteration.

Rule 5 (Termination Criteria): A maximum number of objective function evaluations ($maxNFEs$) or a maximum number of algorithm iterations ($maxNITs$) is considered as the stopping criterion.

2.1.3 Teaching-learning-based optimization algorithm (TLBO)

Rao et al. [18] developed the Teaching-Learning-Based Optimization (TLBO) algorithm in 2011 which is based on the classical school learning process. TLBO consists of two stages: the effect of a teacher on learners and the influence of learners on each other. In this algorithm, the initial population comprising of students or learners is selected randomly. In each iteration, the smartest student with the highest objective function is assigned as the teacher. Students are updated iteratively to search the optimum within two phases: based on the knowledge transfer from the teacher (teacher phase) and interaction with other students (learner phase). In TLBO the performance of the class in learning or the performance of the teacher in teaching is considered as a normal distribution of marks obtained by the students. TLBO improves other students in the teacher phase by employing the difference between the teacher's knowledge and the average knowledge of all the students. The knowledge of each student is obtained based on the position taken place by that student in the search space. In a class, students also improve themselves via interacting with each other after the teaching is completed. In the learner phase, the TLBO algorithm improves the quality of each student by the knowledge interaction between that student and another randomly selected one. In the following, these two phases are presented and formulated:

1. Generation or education of the new learners ($newL$) based on the teacher phase. The class performance as a normal distribution of grades obtained by students can be characterized with the mean value of the distribution. In this phase TLBO aims to improve the class performance by shifting the mean position of the class individuals toward the best learner which is considered as the teacher. This phase is the elitism or global search or intensification ability of the algorithm. In this regard, TLBO updates the learners by a step size toward the teacher obtained based on the difference between the teacher's position and the mean position of all students combining with randomization. Considering the mean

position of students in the search space as $MeanL$, this phase can be formulated as follows:

$$\begin{gathered} stepsize_i = T - TF_i \times MeanL \\ newL = L + rand_{(i)(j)} \times stepsize \\ i = 1,2,\dots,nL \text{ and } j = 1,2,\dots,nV \end{gathered} \tag{7}$$

in which $rand_{(i)(j)}$ is a random number chosen from the continuous uniform distribution on the [0, 1] interval and TF is a teaching factor considered for controlling how much the teacher will change the mean knowledge of the class which can be either 1 or 2.

2. Generating new learners ($newL$) or updating the knowledge of students by interacting with each other in the learner phase. In this phase, each student interacts with a randomly selected one (L_{rp}) except him or her for possible improvement of knowledge. After comparison, the student will be moved toward the randomly selected one if it is smarter ($PFit_i < PFit_{rp}$) and shifted away otherwise. The learner phase can be stated mathematically in the following equation:

$$\begin{gathered} stepsize_i = \begin{cases} L_i - L_{rp}, & PFit_i < PFit_{rp} \\ L_{rp} - L_i, & PFit_i \geq PFit_{rp} \end{cases} \\ newL = L + rand_{(i)(j)} \times stepsize \\ i = 1,2,\dots,nL \text{ and } j = 1,2,\dots,nV \end{gathered} \tag{8}$$

in which $rand_{(i)(j)}$ is a random number chosen from the continuous uniform distribution on the [0, 1] interval. The learner phase is the diversification capability of the algorithm by which each individual tries to improve by searching its neighborhood and sharing information with one randomly selected individual. The step size of search will be decreased gradually as the students approach each other with the progress of the algorithm.

2.1.4 Cuckoo search algorithm (CS)

Yang and Deb [17] developed Cuckoo Search (CS) as a population-based meta-heuristic algorithm inspired by the behaviour of some cuckoo species. Cuckoos are fascinating birds due to their aggressive reproduction strategy. These species lay their eggs in the nests of other host birds. The host takes care of the eggs presuming that the eggs are of its own. However, some of the host birds can combat this parasitic behaviour of cuckoos and throw out the discovered alien eggs or build their new nests in new locations. In the search space, all the nests or eggs whether they belong to the cuckoos or host birds, represent the candidate solutions. Cuckoos and host birds try to breed their generation. In the cyclic body of the algorithm, cuckoos and host birds perform two sequential search phases. First, the cuckoos produce the eggs. In this phase, eggs are produced by guiding the current solutions toward the best possible solution. Then these new eggs are intruded to the nests of host birds based on the replacement strategy. After cuckoo breeding, it turns to the host birds. If a cuckoo's egg is very similar to a host's egg, then this cuckoo's egg is less likely to be discovered. In this phase host birds discover a fraction of alien eggs and update them by addition of a random permutation-based step size. Based on the replacement strategy, the host bird replaces the produced egg with the current one. These two search phases are repeated in the cyclic body of the algorithm until it reaches a stopping criterion. In the following after introducing the Levy flight, CS is formulated in two phases:

Levy Flights as Random Walks: The randomization plays an important role in both exploration and exploitation in meta-heuristic algorithms. The essence of such randomization is random walks. A random walk is a random process which consists of taking a series of consecutive random steps. Let S_N denote the sum of each consecutive random step X_i; then S_N forms a random walk:

$$S_N = \sum_{i=1}^{N} X_i = X_1 + X_2 + \cdots + X_N = S_{N-1} + X_N \tag{9}$$

where X_i is a random step drawn from a random distribution, which means the next state will only depend on the current existing state and the motion or transition X_N from the existing state to the next state. If each step is carried out in the n-dimensional space, the random walk becomes in higher dimensions. There is no reason why each step length should be fixed. In fact, the step size can also vary according to a known distribution. For example, if the step length obeys the Gaussian distribution, the random walk becomes the Brownian motion. A very special case is when the step length obeys the Levy distribution; such a random walk is called a Levy flight or Levy walk. From the implementation point of view, the generation of random numbers with Levy flights consists of two steps: choice of a random direction and the generation of steps which obey the chosen Levy distribution, while the generation of steps is quite tricky. There are a few ways for achieving this, but one of the most efficient and yet straightforward ways is to use the so-called Mantegna algorithm. In Mantegna's algorithm, the step length S can be calculated by

$$S = \frac{u}{|v|^{1/\beta}} \tag{10}$$

where β is a parameter between [1, 2] interval; u and v are drawn from normal distribution. That is

$$u \sim N(0, \sigma_u^2), v \sim N(0, \sigma_v^2) \tag{11}$$

where

$$\sigma_u = \left\{ \frac{\Gamma(1+\beta) \times \sin(\pi\beta/2)}{\Gamma\big((1+\beta)/2\big) \times \beta \times 2^{(\beta-1)/2}} \right\}, \sigma_v = 1 \tag{12}$$

First phase (Cuckoo Breeding): In this step, all the nests except the best one ($bestNest$) are replaced based on their quality by new cuckoo eggs ($newNest$) produced by guiding the current solutions ($Nest$) toward the $bestNest$ in combination with the Levy flight as:

$$\begin{gathered} stepsize = rand_{(i)(j)} \times \alpha \times S \times (Nest - bestNest) \\ newNest = Nest + stepsize \end{gathered} \tag{13}$$

where α is the step size parameter and should be considered more than zero and should be related to the scales of the problem; $rand_{(i)(j)}$ is a random number chosen from the continuous uniform distribution on the [-1, 1] interval, and S is a random walk based on the Levy flights. This phase guarantees the elitism and intensification ability of the algorithm. The $bestNest$ is kept unchanged and other solutions updated toward it.

Second phase (Alien Eggs Discovery by the Host Birds): The alien eggs discovery is performed for each component of each solution in terms of the discovering probability matrix (P) such as:

$$P_{(i)(j)} = \begin{cases} 1, & \text{if} \quad rand < pa \\ 0, & \text{if} \quad rand \geq pa \end{cases} \tag{14}$$

where $rand$ is a random number in [0, 1] interval and pa is the discovering probability. It should be noted that the P matrix has the same size as the $Nest$ matrix. Existing eggs are replaced considering their quality by the newly generated ones from their current positions through random walks with a random permutation-based step size such as:

$$\begin{gathered} stepsize = rand_{(i)(j)} \times (Nest[randp1(i)(j)] - Nest[randp2(i)(j)]) \\ newNest = Nest + stepsize \times P \end{gathered} \tag{15}$$

where $randp1$ and $randp2$ are random permutation functions used for different rows permutation applied on $Nest$ matrix and P is the discovery probability matrix. This phase guarantees the diversification ability of the algorithm.

2.1.5 Vibrating particles system algorithm (VPS)

Vibrating Particles System (VPS) algorithm is a meta-heuristic search algorithm suggested by Kaveh and Ilchi Ghazaan [19]. This algorithm is motivated by the free vibration of systems with single degree of freedom having a viscous damper. Similar to other population-based meta-heuristics, VPS starts with a random set of initial solutions and considers them as the free vibrated single degree of freedom systems with viscous damper. For under-damped conditions, each free vibrated system or vibrating particle oscillates and returns to its equilibrium state. As the optimization process proceeds, using the combination of randomness and exploitation of the obtained results, VPS iteratively improves the quality of the particles by oscillating them toward the equilibrium position. The equilibrium position of each particle is considered as three parts, the best position achieved so far across the whole population (HP), a good particle (GP), and a bad particle (BP). In this way the main features of the VPS consists of three essential concepts, self-adaptation (particle moves toward HB), cooperation (the GP and BP, that are selected from particles themselves, can influence the new position of the particles), and competition (the influence of GP being higher than that of BP). The number of vibrating particles is considered as nVP. These particles form the matrix of Vibrating Particles (VP). After evaluating the objects, the corresponding objective function (Fit) and the penalized objective function ($PFit$) are produced. VPS updates the particles in a way that considers for each particle, three equilibrium positions with different weights (ω_1, ω_2, and ω_3) that the particle tends to approach: (1) the best position achieved so far across the entire population (HP), (2) a good particle (GP), and (3) a bad particle (BP). In order to select GP and BP for each particle, the current population is sorted according to their penalized objective function values in an increasing order, and then GP and BP are chosen randomly from the first and second halves except itself, respectively. Damping level plays an important role in the vibration. Much more damping level higher rate at which the amplitude of a free damped vibration decrease. In order to model this phenomenon in the VPS, a descending function (D) proportional to the number of iterations is proposed as follows:

$$D = (NITs/maxNITs)^{-\alpha} \tag{16}$$

where $NITs$ is the current iteration number of the algorithm, $maxNITs$ is the maximum number of algorithm iterations considered as the stopping criteria, and α is a constant. According to the mentioned concepts, the particles are updated by the following formula which will be read as free vibration formula hereafter:

$$\begin{gathered} newVP_i = \omega_1(D \times A \times rand + HP) + \omega_2(D \times A \times rand + GP_i) \\ + \omega_3(D \times A \times rand + BP_i) \\ A = \omega_1 \times (HP - VP_i) + \omega_2 \times (GP_i - VP_i) + \omega_3 \times (BP_i - VP_i) \\ \omega_1 + \omega_2 + \omega_3 = 1 \end{gathered} \tag{17}$$

in which VP_i and $newVP_i$ are the current and updated positions of the i-th particle, respectively; ω_1, ω_2, and ω_3 are three weights to measure the relative importance of the best-so-far particle found by the algorithm (HP), the good particle (GP), and bad particle (BP) of the i-th particle, respectively; and $rand$s are random numbers uniformly generated between zero and one. A parameter like p within (0, 1) is defined, and it is specified whether the effect of BP must be considered in updating position or not. For each particle, p is compared with $rand$ (a random number uniformly distributed in the range of [0, 1]); if $p < rand$, then $\omega_3 = 0$ and $\omega_2 = 1 - \omega_1$. Three essential concepts, consisting of self-adaptation, cooperation, and competition, are considered in VPS. A particle moves toward HP, so the self-adaptation is provided. Any particle has the chance to have an influence on the new position of the other one, so the cooperation between the particles is supplied. Due to the p parameter, the influence of GP (good particle) is more than that of BP (bad particle); therefore, the competition is provided. As it was mentioned in the introduction section, VPS uses harmony search-based handling approach to deal with a particle violating the limits of the variables. In this approach, a vibrating particles memory ($VP - M$) is utilized to save the nVP number of the best vibrating particles and their related objective function ($Fit - M$) and penalized objective function ($PFit - M$) values. To fulfill this aim, vibrating particles memory is utilized to save the same number with the number of the particles (nVP). Considering memory and benefitting it in the form of different strategies can improve the meta-heuristics performance, without increasing the computational cost. It should be noted again that VPS used it just for regenerating the particles exited from the search space. According to this mechanism, any component of the solution vector violating the variable boundaries can be regenerated from the $VP - M$ as:

$$VP(i,j) = \begin{cases} \text{w.p. } vpmcr & \Rightarrow \quad \text{select a new value from } VP - M, \\ \text{w.p. } (1 - par) & \Rightarrow \quad \text{do nothing,} \\ \text{w.p. } par & \Rightarrow \quad \text{choose a neighboring value,} \\ \text{w.p. } (1 - vpmcr) & \Rightarrow \quad \text{select a new value randomly} \end{cases} \tag{18}$$

where "w.p." is the abbreviation for "with the probability," $VP(i,j)$ is the j-th component of the i-th vibrating particle, $vpmcr$ is the vibrating particle memory considering rate varying between 0 and 1 and sets the probability of choosing a value in the new vector from the historic values stored in $VP - M$, and $(1 - vpmcr)$ sets the probability of choosing a random value from the possible range of values. The pitch-adjusting process is performed only after a value is chosen from $VP - M$. The value $(1 - par)$ sets the rate of doing nothing, and par sets the rate of choosing a value from neighboring the best vibrating particle or the particles saved in memory. For choosing a value from neighboring the best vibrating particle or the particles saved in memory, for continuous search space, a randomly generated step size can be used ($\pm bw \times rand$).

2.1.6 Water evaporation optimization algorithm (WEO)

Kaveh and Bakhshpoori [20] developed the Water Evaporation Optimization (WEO) algorithm. This algorithm is inspired by evaporation of a tiny amount of water molecules on the solid surface with different wettability. The algorithm considers water molecules as individuals. The solid surface or substrate with variable wettability is reflected as the search space. Decreasing the surface wettability reforms the water aggregation from a monolayer to a sessile droplet. Such behaviour is consistent with how the layout of individuals changes to each other as the algorithm progresses. Decreasing the wettability of the surface can decrease the objective function for a minimizing optimization problem. The evaporation flux rate of the water molecules is considered as the most suitable measure for updating the individuals. Their change pattern is in good agreement with the local and global search ability of the algorithm and can help WEO to have well-converged behaviour and simple algorithmic structure. The pseudo code of the WEO algorithm for solving constrained optimization problems is as follows:

Define the algorithm parameters: nWM and $maxNFEs$.
Generate random initial water molecules (WM).
Evaluate the initial molecules and form its corresponding vectors of the objective function (Fit) and penalized objective function ($PFit$).
While $NFEs \leq maxNFEs$
  Update $NITs$.
  if $NITs \leq maxNITs/2$
    Generate new water molecules based on the monolayer evaporation strategy.
    Evaluate the newly generated water molecules, and replace the current molecules with the evaporated ones if the newest ones are better.
    Update $NFEs$.
  else
    Generate new water molecules based on the droplet evaporation strategy.
    Evaluate the newly generated water molecules, and replace the current molecules

with the evaporated ones if the newest ones are better.
Update $NFEs$.
end if
Determine and monitor the best water molecule ($bestWM$).
end While

2.1.7 Hybridized ABC-TLBO algorithm (ABC-TLBO)

The ABC-TLBO is a high-level relay hybridized algorithm. The maximum number of objective function evaluations is defined as the stopping criteria of the algorithm. The ABC algorithm works at the first half of the algorithm (the first half of objective function evaluations), whereas the TLBO algorithm works at the second half of the algorithm (the second half of objective function evaluations). In other word, when the number of objective function evaluations reaches to the half of the maximum number of objective function evaluations, the hybridized ABC-TLBO algorithm switches from the ABC algorithm to the TLBO algorithm.

2.2 Definition of the optimization problem

In truss size and geometry simultaneous optimization problems, the the goal is to achieve a minimum-weight truss structure satisfying certain constraints on natural frequencies. In other words, the optimization goal is to find a set of design variables to minimize the weight function while all constraints are satisfied. Design variables, which include nodal coordinates and element cross-sectional areas, are assumed to change continuously. In addition, each variable may be restricted within an acceptable region. The candidate solutions are encoded by a vector of real values. The structural topology is kept fixed in the design process. A truss structure is made up of structural elements called bars. The weight of a truss structure depends on the total amount of material used for the bars. Thus, the optimization problem can be described mathematically as follows [3]:

Find

$$\{X\} = [x_1, x_2, \ldots, x_{nDV}], \quad x_i \in S_i \tag{19}$$

to minimize

$$PFit(\{X\}) = W(\{X\}) + f_{penalty}(X) \tag{20}$$

where

$$W(\{X\}) = \sum_{i=1}^{ne} \rho_i \times A_i \times L_i \tag{21}$$

subjected to:

$$\begin{cases} x_i^L \le x_i \le x_i^U \\ \omega_k \le \omega_k^* \\ \omega_j \ge \omega_j^* \end{cases} \tag{22}$$

where $\{X\}$ is the vector of design variables, including both nodal coordinates and cross-sectional areas; nDV is the number of design variables; x_i represents the design variable i; S_i is the available set of values for the design variable x_i; $W(\{X\})$ denotes the objective function (the weight of the truss structure); $f_{penalty}(\{X\})$ denotes the penalty function; $PFit(\{X\})$ denotes the penalized objective function; ne is the number of members of the truss; ρ_i, A_i, and L_i represent the material density, length and the cross-sectional area of member i, respectively. ω_k is the k-th natural frequency of the structure and ω_k^* is its upper bound; ω_j is the th natural frequency of the structure and ω_j^* is its lower bound. x_i^l and x_i^u are the lower and upper bounds of the design variable x_i, respectively.

The problem constraints are handled by a simple penalizing strategy. In penalizing strategies, infeasible solutions are considered during the search process. The unconstrained objective function is extended by a penalty function that will penalize infeasible solutions. The penalty function is defined as:

$$f_{penalty}(X) = \varepsilon . v, \quad v = \sum_{i=1}^{q} v_i \tag{23}$$

where q is the number of frequency constraints; ε is the aggregation weight. In this research, the value of aggregation weight is considered to be equal to 10^9. If the i-th constraint is satisfied, then v_i will be taken as zero, otherwise it will be taken as:

$$v_i = \left| 1 - \left(\frac{\omega_i}{\omega_i^*} \right) \right| \tag{24}$$

3. RESULTS AND DISCUSSION

3.1 Parameter tuning results

In this study, the Taguchi method is employed to tune the parameters of the meta-heuristics. In the Taguchi method, orthogonal arrays are used to study a large number of decision variables with a small number of experiments. In this method, the objective functions are categorized into the three groups of "Smaller is better", "Larger is better", and "Nominal is best", with each having a particular formula for the signal-to-noise (S/N) ratio. In this study, the aim is to minimize the weight of the structures; hence, "Smaller is better" is selected. First, the parameters along with their levels are introduced. Then the proper scheme of the Taguchi method is selected. Next, the results are analyzed through the analysis of variance. Finally, the best combination of the parameters is selected for the tuned meta-heuristics. For each parameter, the level with higher value of S/N ratio and lower value of Mean is the best level of the parameter. The parameters of algorithms are tuned for the problem of size optimization of a 10-bar planar truss (shown in Fig. 1). The TLBO algorithm has only one independent parameter (number of learners (nL)). Therefore, the Taguchi method is not needed for the TLBO algorithm. In addition, the parameters of the hybridized ABC-TLBO algorithm are not tuned. In the following sections, the results of parameter tuning for each algorithm are presented.

3.1.1 Parameter tuning of artificial bee colony

The ABC algorithm has three independent parameters consisting of number of honey bees (nHB), specified number of trials (A) by which an employed bee becomes a scout bee, and the modification rate (mr) as a control parameter that controls whether the selected food source by onlooker bee will be modified or not. Five levels are considered for each parameter; hence, in Taguchi design, "Number of factors" and "Type of Design" are chosen three, and "4-Level Design", respectively. Table 1 lists the parameters of the ABC algorithm and their levels. According to the results obtained from the MINITAB software, best values for the parameters may be written as follows:
$nHB = 50;\ mr = 0.6;\ A = 200$

Table 1: Parameters and their levels for the ABC algorithm

Parameters	Levels				
	1	2	3	4	5
nHB	10	20	30	50	100
mr	0.2	0.4	0.6	0.8	1
A	40	100	200	300	400

3.1.2 Parameter tuning of cyclical partenogenesis algoritm

The CPA algorithm has five independent parameters: number of aphids (nA), number of colonies (nC), parameters to control the searching step sizes (α_1 and α_2), and parameter to determine the ratio of aphids of each colony to be considered as female (Fr). Four levels are considered for each parameter; hence, in Taguchi design, "Number of factors" and "Type of Design" are chosen five, and "4-Level Design", respectively. Table 2 lists the parameters of the CPA algorithm and their levels. According to the results obtained from the MINITAB software, the best values for the parameters may be written as follows:
$nA = 75;\ nC = 7;\ \alpha_1 = 1;\ \alpha_2 = 2;\ Fr = 0.4$

Table 2: Parameters and their levels for the CPA algorithm

Parameters	Levels			
	1	2	3	4
nA	30	45	60	75
nC	5	7	9	11
α_1	0.25	0.5	0.75	1
α_2	1.25	1.5	1.75	2
Fr	0.2	0.4	0.6	0.8

3.1.3 Parameter tuning of cuckoo search

The CS algorithm has three independent parameters: number of nests (nN), step size controlling parameter (α), and discovering probability (pa). Five levels are considered for each parameter; hence, in Taguchi design, "Number of factors" and "Type of Design" are chosen three, and "5-Level Design", respectively. Table 3 lists the parameters of the CS algorithm and their levels. According to the results obtained from the MINITAB software, the best values for the parameters may be written as follows:

$nN = 20;\ pa = 0.15;\ \alpha = 1$

Table 3: Parameters and their levels for the CS algorithm

Parameters	Levels				
	1	2	3	4	5
nN	10	20	30	50	100
pa	0.05	0.1	0.15	0.25	0.5
α	0.25	0.5	1	1.5	2

3.1.4 Parameter tuning of water evaporation optimization

The WEO algorithm has five independent parameters: number of water molecules (nWM), the maximum and minimum values of the substrate interaction energy (E_{max} and E_{min}), and the maximum and minimum values of the contact angle (θ_{max} and θ_{min}). Four levels are considered for each parameter; hence, in Taguchi design, “Number of factors” and “Type of Design” are chosen five, and “4-Level Design”, respectively. Table 4 lists the parameters of the WEO algorithm and their levels. According to the results obtained from the MINITAB software, the best values for the parameters may be written as follows:

$nWM = 8;\ E_{max} = -1;\ E_{min} = -3;\ \theta_{max} = -20;\ \theta_{min} = -45$

Table 4: Parameters and their levels for the WEO algorithm

Parameters	Levels			
	1	2	3	4
nWM	8	16	24	32
E_{max}	-0.25	-0.5	-1	-2
E_{min}	-3	-3.5	-4	-4.5
θ_{max}	-10	-20	-25	-30
θ_{min}	-35	-45	-55	-65

3.1.5 Parameter tuning of vibrating particles system

The VPS algorithm has eight independent parameters: number of vibrating particle (nVP), parameters of harmony search-based handling approach ($vpmcr$, par, and bw), weights to measure the relative importance of the good particle and the best-so-far particle found by the algorithm (ω_2 and ω_1), the probability of considering the effect of bad particle in updating position (p), and α. Four levels are considered for the parameter except bw. For this parameter, two levels are considered. Therefore, in Taguchi design, “Number of factors” and “Type of Design” are chosen eight, and “Mixed Level Design”, respectively. Table 5 lists the parameters of the VPS algorithm and their levels. According to the results obtained from the MINITAB software, the best values for the parameters may be written as follows:

$nVP = 13\ \ \omega_1 = \omega_2 = 0.3;\ bw = 0.1;\ \alpha = 0.05;\ p = 0.1;\ vpmcr = 0.95;\ par = 0.1$

Table 5: Parameters and their levels for the VPS algorithm

Parameters	Levels			
	1	2	3	4
bw	0.1	0.2		
$\alpha, p,$ and par	0.05	0.1	0.15	0.2
nVP	10	13	16	20
ω_1, ω_2	0.1	0.2	0.3	0.4
$vpmcr$	0.95	0.85	0.75	0.65

3.2 Optimization results

In this section, four benchmark truss structure problems are optimized utilizing the employed algorithms, and the performance of the meta-heuristics are compared to each other. The structures are as follows: a 10-bar planar truss, a 200-bar planar truss, a 72-bar space truss, and a 52-bar dome-like truss. This study aims to compare the susceptibility and robustness of the algorithms in both aspects of the accuracy and convergence speed. The initial populations are generated randomly. For each Metaheuristic, the best design of 10 independently runs is reported. Convergence curves of all algorithms are provided. The maximum number of analyses is considered as the stopping criterion of the algorithms.

3.2.1 Example 1: the 10-bar planar truss

Size optimization of the 10-bar planar truss shown in Fig. 1 is considered. This is a well-known problem in the field of weight optimization of the structures with frequency constraints. The cross-sectional area of each of the members is considered to be an independent variable. A non-structural mass of 453.6 kg is attached to the free nodes. Table 6 shows the material properties, variable bounds, and frequency constraints for this example. This problem has been investigated by several researchers, including Lingyun et al. using GA [2], Kaveh and Zolghadr [5] using a hybridized CSS-BBBC algorithm, Kaveh and Zolghadr [7] using DPSO, Kaveh et al. [10] using DEO, Kaveh and Ilchi Ghazaan [11] using IRO, Kaveh and Ilchi Ghazaan [13] using VPS, etc.

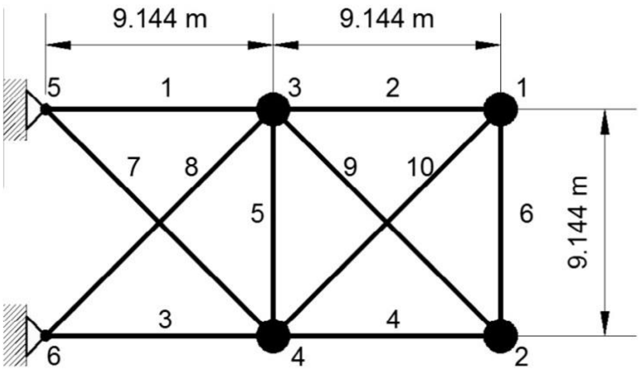


Figure 1. The 10-bar planar truss

Table 6: Material properties, variable bounds and frequency constraints of the 10-bar planar truss

Property / Unit	Value
E (Modulus of elasticity) / GPa	68.95
ρ (Material density) / kg/m^3	2767.99
Lower bound of design variables / cm^2	0.645
Upper bound of design variables / cm^2	50
Frequency constraints / Hz	$\omega_1 \geq 7$, $\omega_2 \geq 15$, $\omega_3 \geq 20$

Table 7 lists the optimal results for the 10-bar truss reported by other researchers. Table 8 lists the optimized designs obtained by the utilized algorithms for the 10-bar truss. This table illustrates the best optimized weights, average optimized weights and standard deviations on optimized weights obtained by the algorithms. The maximum number of objective function evaluations is defined as the stopping criteria of the algorithms, which is considered equal to 20000 for all algorithms. The optimization results show that the algorithms have acceptable performances. A careful examination of Table 8 reveals that hybridized ABC-TLBO, TLBO, and VPS have better results in terms of the best optimized weight, average optimized weight, and standard deviation on optimized weights. Comparing the results obtained by the utilized algorithms (Table 8) with those of other researchers (Table 7) indicated that all of the utilized meta-heuristics converge to solutions very close to the best results of other researchers, which demonstrates the high performance of these algorithms. Table 9 represents the natural frequencies of the optimized structures obtained by the algorithms. It can be seen that all of the constraints are satisfied. As Table 9 suggests, the values obtained by different algorithms for the first and third natural frequencies are very close to their lower bounds, while the values obtained by examined algorithms for the second natural frequency are not close its lower bound. This means that the first and third natural frequencies of the structure control the design process. Fig. 2 shows the convergence histories of the algorithms for the 10-bar truss. As Fig. 2 suggests, the convergence rates of hybridized ABC-TLBO, CPA, and TLBO are higher than those of other considered meta-heuristics.

Table 7: Optimal designs found for the 10-bar plane truss problem by other researchers

Design variable	Areas (cm^2)					
	Ref [2]	Ref [5]	Ref [7]	Ref [10]	Ref [11]	Ref [13]
1	42.234	39.569	35.944	35.3	35.047	35.147
2	18.555	16.740	15.530	15.1	15.138	14.669
3	38.851	34.361	35.285	36.5	35.813	35.689
4	11.222	12.994	15.385	15.4	15.071	15.093
5	4.783	0.645	0.648	0.645	0.645	0.645
6	4.451	4.802	4.583	4.6	4.630	4.622
7	21.049	26.182	23.610	23.7	23.940	23.555
8	20.949	21.260	23.599	24	23.823	24.468
9	10.257	11.766	13.135	11.5	12.530	12.720
10	14.342	11.392	12.357	13.5	12.927	12.685
Besst weight (kg)	542.75	529.25	532.39	532.814	531.24	530.77

Table 8: Comparison of optimal designs found for the 10-bar plane truss (present work)

Design variable	Areas (cm^2)						
	ABC	TLBO	CS	WEO	VPS	CPA	ABC-TLBO
1	35.0453	35.359	35.697	35.599	35.800	35.901	36.195
2	14.9950	14.915	14.905	14.908	15.107	14.702	15.044
3	35.5583	35.900	35.362	35.536	35.701	35.660	34.661
4	14.7868	15.140	13.985	15.078	14.913	15.023	14.979
5	0.6450	0.645	0.645	0.650	0.646	0.645	0.645
6	4.6239	4.603	4.710	4.637	4.599	4.621	4.662
7	24.2375	23.614	25.792	23.893	24.131	23.760	24.034
8	23.7482	24.209	23.469	24.034	23.751	24.402	23.847
9	12.5907	12.740	11.585	12.625	12.501	12.439	12.915
10	13.0012	12.371	13.275	12.543	12.399	12.350	12.521
Best weight (kg)	530.78	530.77	531.84	530.97	530.75	530.82	530.72
Average weight (kg)	552.54	538.80	557.09	550.58	541.20	542.79	537.69
Std. Dev. (kg)	51.61	30.11	60.55	43.87	20.91	30.84	20.38
No. of analyses	20000	20000	20000	20000	20000	20000	20000

Table 9: Natural frequencies (Hz) evaluated at the optimum designs of the 10-bar truss

Frequency number	Natural frequencies (Hz)						
	ABC	TLBO	CS	WEO	VPS	CPA	ABC-TLBO
1	7.0000	7.0000	7.0003	7.0003	7.0000	7.0002	7.0003
2	16.1515	16.1853	16.1664	16.1920	16.1987	16.1981	16.1756
3	20.0001	20.0003	20.0343	20.0042	20.0002	20.0004	20.0105
4	20.0003	20.0036	20.0672	20.0432	20.0009	20.0107	20.0684

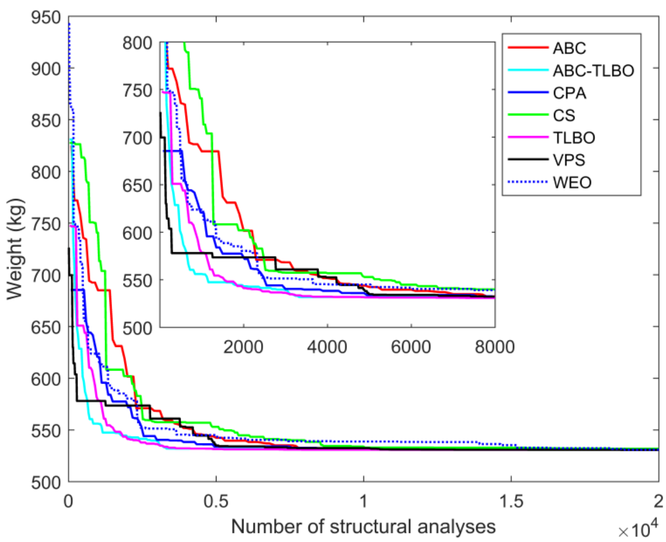


Figure 2: Convergence histories of the algorithms for the 10-bar planar truss

3.2.2 Example 2: the 200-bar planar truss

Fig. 3 shows the topology and the pattern for node and element numbering of a 200-bar planar truss. This is a benchmark problem in the field of weight minimization of truss structures with multiple frequency constraints. The cross-sectional area of each of the members is considered to be an independent variable. Table 10 summarizes the material properties and frequency constraints for this example. Non-structural masses of 100 kg are attached to the upper nodes. A lower bound of 0.1 cm^2 is assumed for the cross-sectional areas. The elements are divided into 29 groups. This problem has been investigated by several researchers, including Kaveh and Zolghadr [5] using a hybridized CSS-BBBC algorithm, and Kaveh and Ilchi Ghazaan [12] using two hybridized optimization algorithm.

Table 10: Material properties, variable bounds and frequency constraints of the 200-bar truss

Property / Unit	Value
E (Modulus of elasticity) / GPa	210
ρ (Material density) / kg/m^3	7860
Added mass / kg	100
Lower bound of design variables / cm^2	0.1
Frequency constraints / Hz	$\omega_1 \geq 5$, $\omega_2 \geq 10$, $\omega_3 \geq 15$

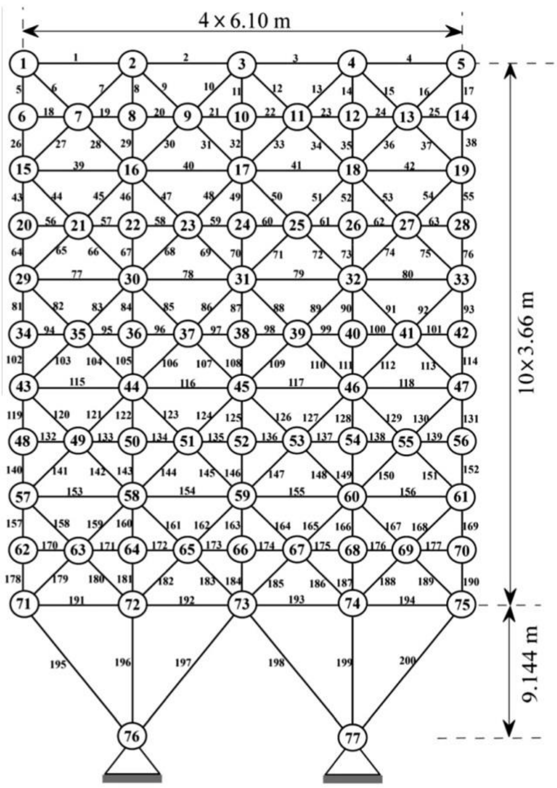


Figure 3: Schematic of the 200-bar planar truss

Table 11 lists the optimized designs obtained by the utilized algorithms for the 200-bar planar truss. This table illustrates the best optimized weights, average optimized weights and standard deviations on optimized weights obtained by the algorithms. The maximum number of objective function evaluations is defined as the stopping criteria of the algorithms, which is considered equal to 30000 for all algorithms. The optimization results show that the algorithms have acceptable performances. A careful examination of Table 11 reveals that hybridized ABC-TLBO, WEO, ABC, and TLBO have better results in terms of the best optimized weight, whereas hybridized ABC-TLBO, VPS, WEO, and CS have better results in terms of the average optimized weight. The best optimal results reported by Kaveh and Zolghadr [5], and Kaveh and Ilchi Ghazaan [12] are 2298.61 and 2156.73 kg, respectively. Comparing the results obtained by the utilized algorithms (Table 11) with those reported by Kaveh and Zolghadr [5] and Kaveh and Ilchi Ghazaan [12] indicated that all of the utilized meta-heuristics converge to solutions very close to the best results of other researchers, which demonstrates the high performance of these algorithms. Table 12 represents the natural frequencies of the optimized structures obtained by the algorithms. It can be seen that all of the constraints are satisfied. As Table 12 suggests, the values obtained by different algorithms for the first and third natural frequencies of the structure are very close to their lower bounds, while the values obtained by examined algorithms for the second natural frequency are not close its lower bound. This means that the first and third natural frequencies of the structure control the design process. Fig. 4 shows the convergence histories of the algorithms for the 200-bar planar truss. As Fig. 4 suggests, the convergence rates of hybridized ABC-TLBO, VPS, and CS are considerably higher than those of other considered meta-heuristics.

Table 11: Comparison of optimal designs found for the 200-bar plane truss (present work)

Design variable	Areas (cm^2)						
	ABC	TLBO	CS	WEO	VPS	CPA	ABC-TLBO
1	0.3030	0.3096	0.3355	0.3046	0.2989	0.2719	0.3543
2	0.4599	0.4485	0.4290	0.4725	0.4645	0.5019	0.4761
3	0.1000	0.1001	0.1000	0.1004	0.1000	0.1000	0.1056
4	0.1003	0.1000	0.1010	0.1023	0.1000	0.1000	0.1069
5	0.5386	0.5407	0.4899	0.4940	0.5081	0.5299	0.5668
6	0.8127	0.8320	0.8337	0.8251	0.8241	0.8245	0.8148
7	0.1000	0.1000	0.1035	0.1034	0.1012	0.1219	0.1095
8	1.4111	1.4044	1.4258	1.4575	1.4646	1.3727	1.4216
9	0.1000	0.1000	0.1000	0.1002	0.1004	0.1000	0.1050
10	1.6580	1.6083	1.5634	1.5724	1.6046	1.5974	1.7580
11	1.1720	1.1590	1.1590	1.1597	1.1584	1.1745	1.1603
12	0.1035	0.1039	0.1511	0.1305	0.1126	0.1667	0.1535
13	3.0494	2.8872	2.8110	2.8892	3.0006	2.9261	2.8319
14	0.1030	0.1059	0.1000	0.1144	0.1004	0.1028	0.2705
15	3.2694	3.1250	3.2470	3.2485	3.2955	3.1685	3.0893
16	1.5201	1.5775	1.5277	1.5794	1.5629	1.6226	1.5483
17	0.2808	0.2266	0.3530	0.2415	0.2880	0.2982	0.2832

18	5.0808	5.1565	5.2126	5.0382	4.8897	5.0649	5.1530
19	0.1000	0.1032	0.1168	0.1131	0.1164	0.1022	0.1327
20	5.4214	5.4625	5.4139	5.4587	5.4388	5.3174	5.1608
21	2.0794	2.0568	2.0340	2.1107	2.1315	2.1205	1.9811
22	0.7290	0.7871	0.6283	0.6219	0.8631	0.8556	0.4080
23	7.6408	7.6738	7.3767	7.4145	7.6361	7.5153	7.9123
24	0.2149	0.1286	0.1048	0.1117	0.1625	0.1097	0.2117
25	7.8842	7.8370	8.0048	7.7565	7.6715	7.8672	7.3745
26	2.7381	2.8117	2.7501	2.7339	2.9449	2.7310	2.6607
27	10.3701	10.7253	10.5691	10.7550	10.2654	10.6839	10.8293
28	21.3897	21.5427	21.7397	21.6967	20.7173	20.8942	22.0511
29	10.9775	10.0024	10.4538	10.2041	11.8240	11.6154	10.7496
Best weight (kg)	2157.68	2157.79	2158.36	2157.66	2158.48	2159.13	2157.58
Average weight (kg)	3212.41	3036.06	2983.59	2956.19	2726.60	3291.73	2323.51
Std. Dev. (kg)	2614.68	2326.90	2970.54	1936.00	2676.39	2109.42	237.17
No. of analyses	30000	30000	30000	30000	30000	30000	30000

Table 12: Natural frequencies (Hz) evaluated at the optimum designs of the 200-bar truss

Frequency number	Natural frequencies (Hz)						
	ABC	TLBO	CS	WEO	VPS	CPA	ABC-TLBO
1	7.0000	7.0000	7.0003	7.0003	7.0000	7.0002	7.0003
2	16.1515	16.1853	16.1664	16.1920	16.1987	16.1981	16.1756
3	20.0001	20.0003	20.0343	20.0042	20.0002	20.0004	20.0105
4	20.0003	20.0036	20.0672	20.0432	20.0009	20.0107	20.0684
5	28.6121	28.4839	28.0827	28.5563	28.4811	28.4803	28.6063

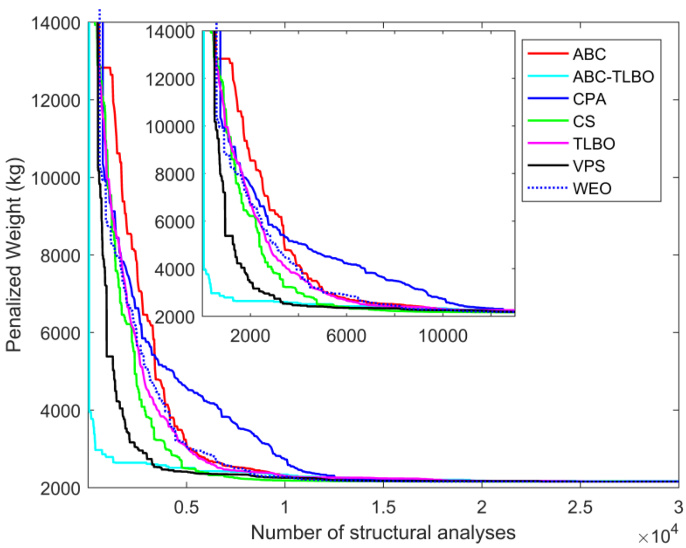


Figure 4: Convergence histories of the algorithms for the 200-bar planar truss

3.2.3 Example 3: the 72-bar space truss

Fig. 5 shows the topology and the pattern for node and element numbering of a 72-bar space truss. This is a benchmark problem in the field of weight minimization of truss structures with multiple frequency constraints. The cross-sectional area of each of the members is considered to be an independent variable. Table 13 summarizes the material properties and frequency constraints for this example. Non-structural masses of 2268 kg are attached to the nodes 1-4. The elements are divided into 16 groups. This problem has been investigated by several researchers, including Kaveh and Zolghadr [5] using a hybridized CSS-BBBC algorithm, Kaveh et al. [10] using DEO, Kaveh and Ilchi Ghazaan [11] using IRO, Kaveh and Ilchi Ghazaan [12] using two hybridized optimization algorithm, Kaveh and Ilchi Ghazaan [13] using VPS, etc.

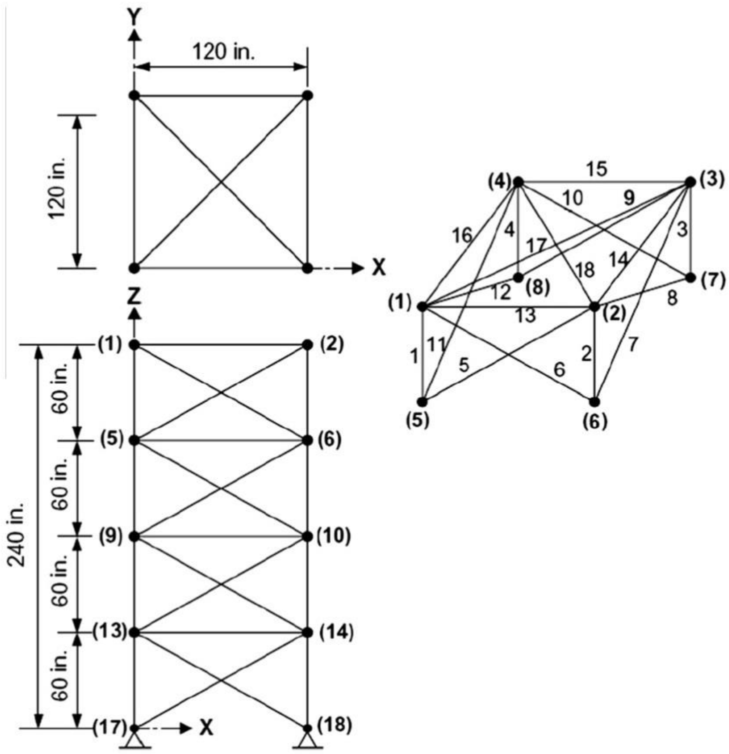


Figure 5: Schematic of a spatial 72-bar truss

Table 13: Material properties, variable bounds and frequency constraints of the 72-bar truss

Property / Unit	Value
E (Modulus of elasticity) / GPa	68.95
ρ (Material density) / kg/m^3	2767.99
Added mass / kg	2268
Lower bound of design variables / cm^2	0.645
Upper bound of design variables / cm^2	20
Frequency constraints / Hz	$\omega_1 = 4$, $\omega_3 \geq 6$

Table 14 lists the optimized designs obtained by the utilized algorithms for the 72-bar truss. The table illustrates the best optimized weights, average optimized weights and standard deviations on optimized weights obtained by the algorithms. The maximum number of objective function evaluations is defined as the stopping criteria of the algorithms, which is considered equal to 20000 for all algorithms. A careful examination of Table 14 reveals that the algorithms have very close performances in terms of the best optimized weight, whereas hybridized ABC-TLBO, CPA, VPS, and CS have better results in terms of the average optimized weight and standard deviation on optimized weights. The best optimal design results reported by Kaveh and Zolghadr [5], Kaveh et al. [10], Kaveh and Ilchi Ghazaan [11], and Kaveh and Ilchi Ghazaan [12], are 327.507, 329.422, 327.597, and 327.65 kg, respectively. Comparing the results obtained by the utilized algorithms (Table 14) with the above-mentioned results indicated that all of the utilized algorithms converge to solutions very close to the best results of other researchers, which demonstrates the high performance of the algorithms. Table 15 represents the natural frequencies of the optimized structures obtained by the algorithms. It can be seen that all of the constraints are satisfied. As Table 15 suggests, the values obtained by different algorithms for the first and third natural frequencies are very close to their lower bounds. This means that the first and third natural frequencies control the design process. Fig. 6 shows the convergence histories for the 72-bar truss. As Fig. 6 suggests, the convergence rates of hybridized ABC-TLBO, VPS, and CPA are higher than those of other considered meta-heuristics.

Table 14: Comparison of optimal designs found for the 72-bar truss (present work)

Design variable	Areas (cm^2)						
	ABC	TLBO	CS	WEO	VPS	CPA	ABC-TLBO
1	3.4515	3.5750	3.2273	3.4301	3.4626	3.7438	3.4894
2	7.8096	7.9231	7.7472	7.8474	7.9238	7.8904	7.8887
3	0.6450	0.6450	0.6450	0.6508	0.6450	0.6450	0.6450
4	0.6453	0.6451	0.6450	0.6527	0.6450	0.6450	0.6450
5	8.1568	7.9839	8.4921	8.0996	8.0001	8.6836	7.9435
6	8.0141	8.0377	8.0895	7.9830	8.0558	8.0373	8.0880
7	0.6450	0.6450	0.6450	0.6543	0.6459	0.6450	0.6450
8	0.6451	0.6450	0.6749	0.6502	0.6450	0.6457	0.6450
9	13.1548	12.8563	12.9831	12.6499	12.8181	12.2459	13.0660
10	8.0896	7.9735	7.9272	8.0932	8.0162	8.1989	8.1608
11	0.6453	0.6469	0.6450	0.6587	0.6451	0.6450	0.6450
12	0.6450	0.6450	0.6476	0.6555	0.6457	0.6450	0.6450
13	16.9350	17.2664	17.0308	17.5009	17.4008	17.0873	17.2086
14	8.2091	8.1859	8.3732	8.1990	8.1219	7.9965	7.9835
15	0.6450	0.6456	0.6450	0.6489	0.6450	0.6450	0.6450
16	0.6450	0.6453	0.6450	0.6878	0.6450	0.6450	0.6450
Best weight (kg)	327.63	327.59	327.87	327.86	327.56	327.74	327.53
Average weight (kg)	348.65	345.95	342.98	352.80	341.43	338.93	339.37
Std. Dev. (kg)	55.54	56.10	44.31	65.25	42.52	34.60	35.15
No. of analyses	20000	20000	20000	20000	20000	20000	20000

Table 15: Natural frequencies (Hz) evaluated at the optimum designs of the 72-bar truss

Frequency number	Natural frequencies (Hz)						
	ABC	TLBO	CS	WEO	VPS	CPA	ABC-TLBO
1	4.0002	4.0001	4.0003	4.0000	4.0000	4.0000	4.0006
2	4.0002	4.0001	4.0003	4.0000	4.0000	4.0000	4.0006
3	6.0000	6.0000	6.0001	6.0002	6.0000	6.0000	6.0000
4	6.2453	6.2479	6.2502	6.2614	6.2407	6.2696	6.2425
5	9.0530	9.0804	9.0143	9.0780	9.0668	9.0981	9.0686

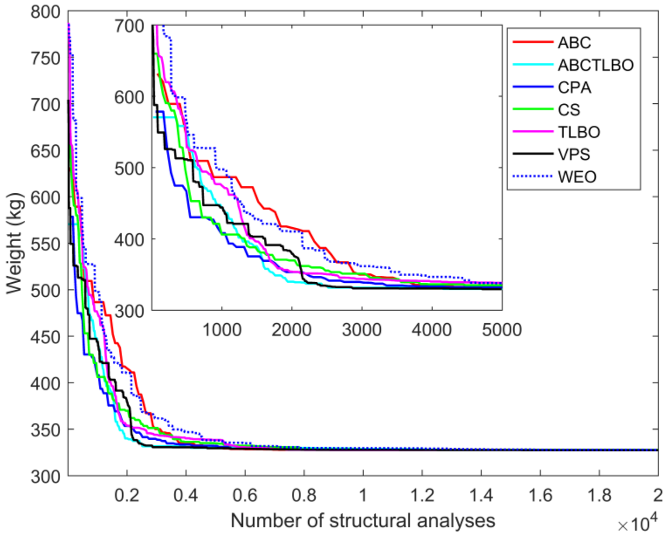


Figure 6: Convergence histories of the algorithms for the spatial 72-bar truss

3.2.4 Example 4: the 52-bar dome-like truss

The forth example is 52-bar dome-like space truss, as depicted in Fig. 7. This is a simultaneous shape and size optimization problem, where both the cross-sectional area of the members and the nodal coordinates are considered as variables. Non-structural masses of 50 kg are attached to all free nodes. Material properties, frequency constraints and variable bounds for this example are summarized in Table 16. All of the elements of the structure are categorized in eight groups according to Table 17. All free nodes are permitted to move in a symmetrical manner, they can move $\pm 2\ m$ in each allowable direction from their initial position. Constraints are imposed on the first two natural frequencies. Therefore this is an optimization on shape and size with thirteen variables (eight sizing variables and five shape variables) and two frequency constraints. This problem has been investigated by several researchers, including Lingyun et al. using GA [2], Kaveh and Zolghadr [5] using a hybridized CSS-BBBC algorithm, Kaveh and Zolghadr [7] using DPSO, Kaveh and Mahdavi [9] using CBO, Kaveh et al. [10] using DEO, Kaveh and Ilchi Ghazaan [11] using IRO, Kaveh and Ilchi Ghazaan [12] using two hybridized optimization algorithm, etc.

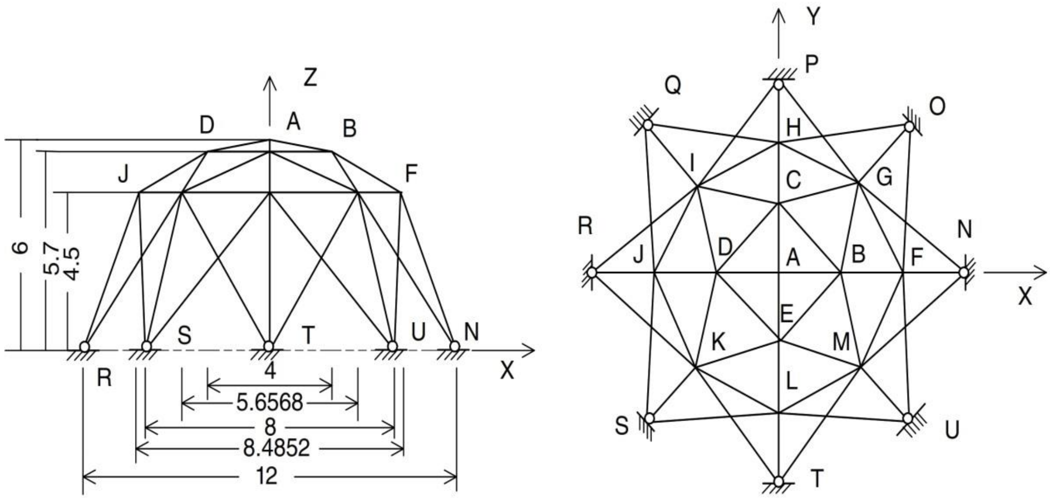


Figure 7: Schematic of the 52-bar dome-like truss

Table 16: Material properties, variable bounds and frequency constraints of the 52-bar truss

Property / Unit	Value
E (Modulus of elasticity) / GPa	210
ρ (Material density) / kg/m^3	7800
Added mass / kg	50
Lower bound of design variables / cm^2	1
Upper bound of design variables / cm^2	10
Frequency constraints / Hz	$\omega_1 \leq 15.916$, $\omega_2 \geq 24.648$

Table 17: Members grouping of the 52-bar dome-like truss

Group member	Members in the group
1	1-4
2	5-8
3	9-16
4	17-20
5	21-28
6	29-36
7	37-44
8	45-52

Table 18 lists the optimized designs obtained by the utilized algorithms for the 52-bar truss. This table illustrates the best optimized weights, average optimized weights and standard deviations on average optimized weights obtained by the algorithms. The maximum number of objective function evaluations is defined as the stopping criteria of the algorithms, which is considered equal to 20000 for all algorithms. A careful examination of

Table 18 reveals that the algorithms have very close performances in terms of the best optimized weight, whereas hybridized ABC-TLBO, CPA, VPS, and TLBO have better results in terms of the best optimized weight, average optimized weight and standard deviation on optimized weights. The best optimal design results reported by Lingyun et al. [2], Kaveh and Zolghadr [5], Kaveh and Zolghadr [7], Kaveh and Mahdavi [9], Kaveh et al. [10], Kaveh and Ilchi Ghazaan [11], and Kaveh and Ilchi Ghazaan [12] are 236.0458, 197.309, 195.351, 197.962, 195.852, 195.38, and 194.85 kg, respectively. It can be seen that the best optimized results obtained by the hybridized ABC-TLBO, ABC, TLBO, WEO, VPS, and CPA are better than the above-mentioned results, which demonstrates the high performance of these algorithms. Table 19 represents the natural frequencies of the optimized structures obtained by the algorithms. It can be seen that all of the constraints are satisfied. As Table 19 suggests, the values obtained by different algorithms for the second natural frequency of the structure is very close to its lower bound, whereas those obtained for the first natural frequency of the structure is far from its upper bound. This means that the second natural frequency of the structure controls the design process. Fig. 8 shows the convergence histories of the algorithms for the 52-bar truss. As Fig. 8 suggests, the convergence rates of hybridized ABC-TLBO and TLBO are considerably higher than those of other considered meta-heuristics.

Table 18: Comparison of optimal designs found for the 52-bar dome-like truss (present work)

Design variable	Areas (cm^2)						
	ABC	TLBO	CS	WEO	VPS	CPA	ABC-TLBO
1	6.1291	5.9763	5.6633	5.9073	5.9480	5.7209	5.9870
2	3.8552	3.7000	3.7000	3.7344	3.7235	3.7068	3.8484
3	2.5000	2.5000	2.5000	2.5044	2.5002	2.5000	2.5000
4	2.1942	2.3740	2.4461	2.3300	2.2540	2.2475	2.0663
5	4.0371	4.0191	4.0356	4.0055	3.9709	3.9430	3.9781
6	1.0000	1.0023	1.0000	1.0109	1.001	1.0000	1.0005
7	1.1690	1.0771	1.0000	1.0696	1.1683	1.1551	1.2601
8	1.2406	1.1975	1.1680	1.1748	1.2336	1.2093	1.2622
9	1.2895	1.4409	1.6634	1.4967	1.4579	1.4560	1.4604
10	1.4088	1.4147	1.4235	1.3966	1.4104	1.3472	1.4418
11	1.0000	1.0000	1.0004	1.0062	1.0001	1.0000	1.0000
12	1.4886	1.4779	1.4493	1.5056	1.4697	1.5240	1.4646
13	1.4833	1.4794	1.5184	1.4784	1.4748	1.5183	1.4542
Best weight (kg)	194.08	193.56	195.27	193.97	193.50	193.87	193.34
Average weight (kg)	266.60	212.14	244.34	281.85	229.48	238.43	206.92
Std. Dev. (kg)	160.23	73.76	118.64	145.22	93.06	118.33	44.86
No. of analyses	20000	20000	20000	20000	20000	20000	20000

Table 19: Natural frequencies (Hz) evaluated at the optimum designs of the 52-bar truss

Frequency number	Natural frequencies (Hz)						
	ABC	TLBO	CS	WEO	VPS	CPA	ABC-TLBO
1	11.1726	12.0940	13.9785	12.1631	11.5153	12.1183	10.5787
2	28.6480	28.6480	28.6525	28.6508	28.6480	28.6480	28.6486
3	28.6484	28.6481	28.6591	28.6526	28.6484	28.6488	28.6490
4	28.6484	28.6487	28.7887	28.6688	28.6506	28.6507	28.6581

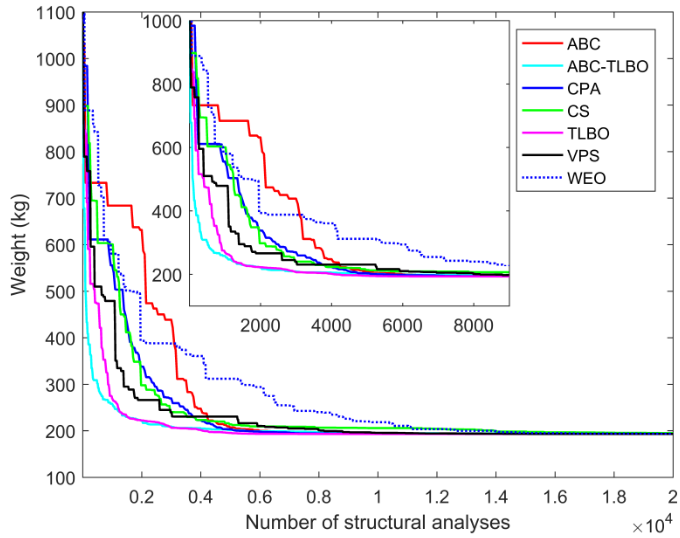


Figure 8: Convergence histories of the algorithms for the 52-bar dome-like truss

4. CONCLUSION

This study examined seven population-based meta-heuristics in the context of simultaneously size and geometry optimization of truss structures and investigated their performances. The objective of optimization was to minimize the weight of truss structures subject to multiple natural frequency constraints. The candidate solutions were encoded by a vector of real values. A simple penalizing strategy was utilized to handle constraints of the problem. The investigated meta-heuristics were Artificial Bee Colony, Cyclical Parthenogenesis Algorithm, Cuckoo Search, Teaching-Learning-Based Optimization, Vibrating Particles System, Water Evaporation Optimization algorithms and a hybridized ABC-TLBO algorithm. The algorithms were tuned with Taguchi method. In order to illustrate the capability and efficiency of the utilized meta-heuristics, the algorithms were applied to continuous size and geometry optimization of four benchmark truss structures. The optimization results indicate that the utilized algorithms have efficient performances for the simultaneously size and geometry optimization of truss structures with continuous design variables. The optimization results indicates that the hybridized ABC-TLBO, TLBO, VPS, and CPA algorithms have better performances in terms of the best optimized weights, average optimized weights, and standard deviation on optimized weights.

REFERENCES

1. Kaveh A, Bakhshpoori T. *Metaheuristics: Outlines, MATLAB Codes and Examples*, Springer, 1st edition, Cham, Switzerland, 2019.
2. Lingyun W, Mei Z, Guangming W, Guang M. Truss optimization on shape and sizing with frequency constraints based on genetic algorithm, *Comput Mech* 2005; **35**(5): 361-8.
3. Kaveh A, Talatahari S. A particle swarm ant colony optimization for truss structures with discrete variables, *J Construct Steel Res* 2009; **65**(8-9):1558-68.
4. Kaveh A, Talatahari S. Size Optimization of Space Trusses Using Big Bang-Big Crunch Algorithm, *Comput Struct* 2009; **87**(17-18): 1129-40.
5. Kaveh A, Zolghadr A. truss optimization with natural frequency constraints using a hybridized CSS-BBBC algorithm with trap recognition capability, *Comput Struct* 2012; **102-103**: 14-27.
6. Kaveh A, Khayatazad M. Ray optimization for size and shape optimization of truss structures, *Comput Struct* 2013; **117**: 82-94.
7. Kaveh A, Zolghadr A. Democratic PSO for Truss Layout and Size Optimization with Frequency Constraints, *Comput Struct* 2014; **130**: 10-21.
8. Kaveh A, Sheikholeslami R, Talatahari S, Keshvari-Ilkhichi M. Chaotic swarming of particles: a new method for size optimization of truss structures, *Adv Eng Softw* 2014; **67**: 136-47.
9. Kaveh A, Mahdavi VR. Colliding bodies optimization method for optimum design of truss optimization with continuous variables, *Adv Eng Softw* 2014; **70**(1): 1-12.
10. Kaveh A, Jafari L, Farhoudi N. Truss Optimization with natural frequency constraints using a dolphin echolocation algorithm, *Asian J Civ Eng* 2015; **16**(1): 29-46.
11. Kaveh A, Ilchi Ghazaan M. Layout and size optimization of trusses with natural frequency constraints using improved ray optimization algorithm, *Iran J Sci Technol-Tran Civ Eng* 2015; **39**(C2+): 395-408.
12. Kaveh A, Ilchi Ghazaan M. Hybridized optimization algorithms for design of trusses with multiple natural frequency constraints, *Adv Eng Softw* 2015; **79**: 137-47.
13. Kaveh A, Ilchi Ghazaan M. Vibrating particles system algorithm for truss optimization with multiple natural frequency constraints, *Acta Mech* 2017; **228**(1): 307-22.
14. Degertekin SO, Lamberti L, Ugur I.B. Sizing, layout and topology design optimization of truss structures using the Jaya algorithm, *Appl Soft Comput* 2018; **70**: 903-28.
15. Karaboga D. An idea based on honey bee swarm for numerical optimization, Technical Report, Department of Computer Engineering, Faculty of Engineering, Erciyes University, Erciyes, Turkey, 2005.
16. Kaveh A, Zolghadr A. Cyclical Parthenogenesis Algorithm: A New Meta-heuristic Algorithm, *Asian J Civ Eng* 2017; **18**(5): 671-701.
17. Yang XS, Deb S. Engineering Optimisation by Cuckoo Search, *Int J Math Modell Numer Optim* 2010; **1**(4): 330-43.

18. Rao RV, Savsani VJ, Vakharia DP. Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems, *Comput Aided Des* 2011; **43**(3): 303-15.
19. Kaveh A, Ilchi Ghazaan M. A new meta-heuristic algorithm: vibrating particles system, *Iran J Sci Technol-Tran Civ Eng* 2017; **24**(2): 551-66.
20. Kaveh A, Bakhshpoori T. Water Evaporation Optimization: A Novel Physically Inspired Optimization Algorithm, *Comput Struct* 2016; **167**(20): 69-85.