DOI: 10.31534/engmod.2023.2.ri.05v
Original scientific paper
Received: 21.09.22.

A Novel Algorithm for Solving Structural Optimization Problems

Dandash Alaa(1,2,*), **Liao HuaLin**(1), **Xiao WenSheng**(2)

(1) *School of Petroleum Engineering, China University of Petroleum (East China), Qingdao, Shandong, 266580, CHINA*

(2) *College of Mechanical and Electronic Engineering, China University of Petroleum (East China), Qingdao, Shandong, 266580, CHINA*

(*) *corresponding author, e-mail: alaa.dandash@live.com*

SUMMARY

In the past few decades, metaheuristic optimization methods have emerged as an effective approach for addressing structural design problems. Structural optimization methods are based on mathematical algorithms that are population-based techniques. Optimization methods use technology development to employ algorithms to search through complex solution space to find the minimum. In this paper, a simple algorithm inspired by hurricane chaos is proposed for solving structural optimization problems. In general, optimization algorithms use equations that employ the global best solution that might cause the algorithm to get trapped in a local minimum. Hence, this methodology is avoided in this work. The algorithm was tested on several common truss examples from the literature and proved efficient in finding lower weights for the test problems.

KEYWORDS: *structural optimization; optimum truss design; stochastic search method; metaheuristic algorithm; size optimization.*

1. INTRODUCTION

Optimization methods or Optimization Algorithms aim to reach the best results for a problem under certain circumstances [1, 2]. In recent decades, various optimization methods have emerged, with the concept behind them based on characteristics and behavior of natural, biological, molecular, physical, swarm of insects, and neurobiological systems [3]. A common approach in metaheuristic optimization is randomly generating an initial population of potential solutions and gradually updating the population in the systematic process [4, 5].

Examples of metaheuristic methods in literature include but not limited to, Genetic Algorithms GA [6-9]; Evolution Strategies ES [10-15]; Particle Swarm Optimization [16-21]; Artificial Immune Algorithm AIA [22]; Simulated Annealing SA [23-25]; Ant Colony Optimization ACO [26-29]; Harmony Search HS [30-33]; Artificial Bee Colony algorithm ABC [34-37]; Gravitational Search Algorithm GSA [38]; Shuffled Frog Leaping SFL [39]; Big Bang-Big Crunch optimization BB-BC [40, 41]; Charged System Search CSS [42]; Teaching-Learning-Based Optimization TLBO [4, 5, 43-46]; Imperialist Competitive Algorithm ICA [47]; Flower

Pollination Algorithm FPA [48]; Swallow Swarm Optimization algorithm SSO [49]; Water Evaporation Optimization WEO [50]; Water Cycle Algorithm WCA [51]; Passing Vehicle Search PVS [52]; Water Wave Optimization WWO [53]; Jaya Algorithm JA [54-56]; Colliding Bodies Optimization CBO [57-59]; Fruit Fly Algorithm FFA [60-62]; Grenade Explosion Method GEM [63]; and many modified, improve and hybrid algorithms [64-69].

GAs rely on the concept of Darwinian theory about evolution, where the fittest solution would survive through the consequent iterations until the end of the process [6-9]. GAs encode the population of solutions as strings of DNAs and cross or mutate them to produce new generations. PSO algorithms use the social behavior of birds while flying to find food sources [6, 8]. SA is a unique algorithm that simulates the thermodynamic change in a metal state based on the metal temperature [3, 24]. HS algorithm tries to find the nice tune while the musician works on his performance [32], CSS makes use of mechanics and physical laws that affect the particles in the system [42], and the ICA tries to mimic countries behavior based on human social or more accurate political behavior [47]. The Jaya algorithm aims to improve the solution in each iteration by a concept of victory, as the algorithm's name indicates, and involves an interaction between the best and worst solutions in the population [54-56].

Examples of using optimization algorithms to solve engineering and optimal design problems are available in the literature. Kaveh and Ghazaan used CBO to solve the sizing optimization problem of truss structures with stress and displacement constraints [59]. Similar works can be found [5, 25, 32, 33, 37]. Farshchin et al. [45] and Pham [71] solved the optimum design problem of truss structures with frequency constraints [45]. Degertekin et al. solved sizing, layout, and topology design optimization of truss structures utilizing the Jaya algorithm [56].

In this work, a new algorithm for solving structural optimization problems is proposed and tested on three common examples from the literature. The common optimization algorithms employ equations that rely on the global best solution as guidance for convergence, which might lead the algorithm to be trapped in a local optimum. This methodology is avoided in this work, and the algorithm randomly moves in the search space, which makes it more diverse and gives a higher probability of finding the actual global minimum.

The remaining of this paper is organized as follows: Section 2 provides a description of the proposed algorithm; Section 3 defines the structural optimization problem; Section 4 shows how to implement the proposed algorithm to solve structural optimization problems; Section 5 presents the test problems, sensitivity analysis, and results; finally the paper is concluded in Section 6.

2. DESCRIPTION OF THE PROPOSED ALGORITHM

The proposed algorithm tries to mimic a hurricane chaos movement that drives the particles in the solution space intending to hit the solution at least once. Imagine a hurricane phenomenon where unbalanced air pressure in a hurricane system creates a vortex with a curving axis. This vortex moves as a hurricane or tornado, carrying many objects or particles. These objects move in the hurricane based on their self-weight, distance from the hurricane axis, and hurricane velocity. The hurricane axis movement drives or leads the whole system. This axis has curving points that change position in the system with each iteration. This will lead the particles to change their position. Figure 1 gives an illustration of such a hurricane system.

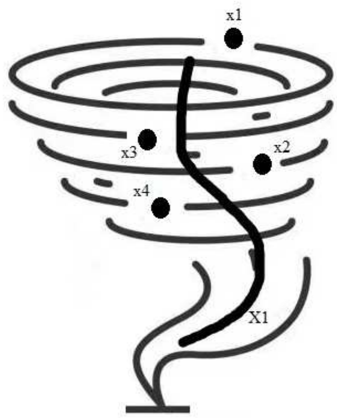


Fig. 1 *An illustration of such a Hurricane system*

A set of points named the "axis of points" is selected to represent the hurricane axis, and the particles in the population are randomly assigned to the axis points. Now, the system is given an initial velocity which is not uniform but rather each particle in the system has its initial velocity as described in Eq. (1) where a random factor is assigned to represent the self-weight or inertia for each particle. The system is moving according to the axis movement that drags the whole system together, and the initial step size is calculated in Eq. (2). At the same time, the hurricane axis would randomly change its shape by Eq. (3), where the axis points play a major role. For this purpose, a random factor is assigned as a curvature factor. Moreover, one additional random factor is employed as to describe a changing speed for the system and is defined as an acceleration factor. In the following, the inertia, curving, and acceleration factors are hypothetical quantities to mimic the hurricane system. The above theory is expressed mathematically as follows:

1. Create the population of particles $X0$;

Consider a solution space with N initial particles distributed randomly. The initial position of the particle i is $X0_i = (x_i^1, x_i^2, x_i^3, ..., x_i^n)$ for $i = 1,2,3.....N$, where x_i^d is the position of i^{th} particle in the d^{th} dimension.

2. Randomly create a number g of curving points referred to as $X1$ that represent the "axis of points", where the curving points are of the same dimension size as the solution vectors $X0$;

3. For the initial hurricane velocity; find the new position $X00$ for each particle as follows:

$$X00_i^d = X0_i^d + a_1 \cdot rand() \cdot X1_m^d \tag{1}$$

$a_1 \cdot rand()$ here represents an inertia factor for the particles relative to the axis points which decides the initial velocity for each particle in the system. The subscript m indicates (refers to) the curving points on the hurricane axis.

4. Calculate the initial step of the system related to the hurricane's initial velocity from step (3):

$$dx = X00 - X0 \tag{2}$$

in steps 3 and 4 the particles are randomly assigned to specific points on the hurricane axis as in Eq. (1) while Eq. (2) provides the first step of the system related to the hurricane axis, and it can be considered as the initial step size of the hurricane system where it does not change with iterations, i.e., $X0$ and $X00$ are fixed for all iterations in each separate run;

5. Update the particles' positions *X0* according to the following expression:

$$X_i^d(ite) = [a_2 \cdot rand() \cdot X1_m^d - X_i^d(ite-1)] + a_3 \cdot rand() \cdot dx_i^d \quad \textbf{(3)}$$

Where the procedure starts with $X_i^d = X0_i^d$. With each iteration *ite* in step 5, the algorithm randomly chooses one point on the axis to update the position of all the particles in the system. The movement of the hurricane axis from step 4 gives a random update for the particles urged by the hurricane axis movement. $a_2 \cdot rand()$ here represents a curving factor of the hurricane axis that is randomly changing with each iteration, $a_3 \cdot rand()$ is a speeding or acceleration factor for the hurricane axis to give a randomly changing velocity with each iteration leading the curving points (axis points) to change their position, hence, all the particles in the system would change their position accordingly. The factors a_1, a_2, and a_3 are selected for each case separately as desired by the researcher. From this expression the effect of curving points is amplified based on the term $[a_2 \cdot rand() \cdot X1_m^d - X_i^d(ite-1)]$. A fly back mechanism is employed to send the particles back to the solution space in case of violation of the upper and lower boundaries. Each particle gets its inertia factor for each run, which means that this factor is fixed throughout all iterations for one run. While, the curving, and acceleration factors are updated with each iteration for each particle. It is worth mentioning that the random factors are not fixed for each particle, but each dimension takes its random factor where it is mentioned.

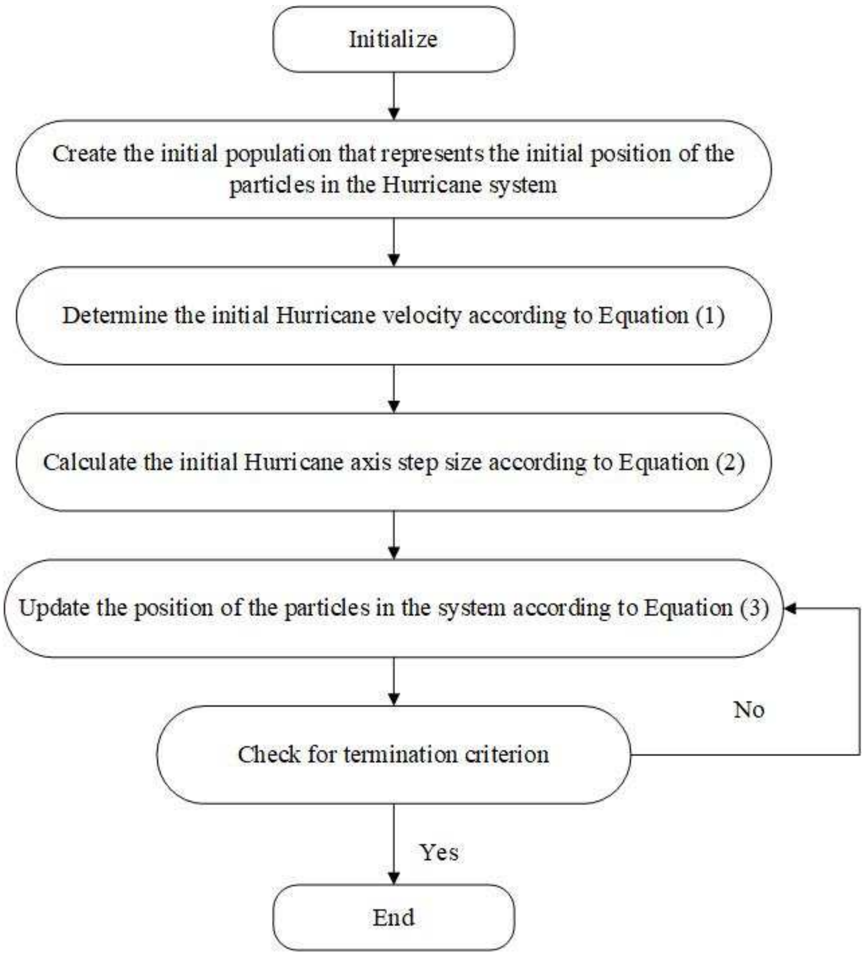


Fig. 2 *Flowchart of the proposed optimization*

3. STRUCTURAL OPTIMIZATION PROBLEM

The goal of employing optimization algorithms in structural design is usually to find the structure's minimal weight under certain constraints. Constraints are usually defined as stress limits, frequency limits, and/or displacement limits on the nodes. These limitations could be applied separately or in combination. In this paper, the optimization problem is defined as follows [59, 69]:

$$\begin{aligned} & W(A)=\sum_{1}^{nm}\gamma_i\ L_i\ A_i \\ & \quad \textit{subjected to} \\ & \textit{stress constraints: } \sigma_i^t<\sigma_i<\sigma_i^c \quad i=1,2,3,......nm \\ & \textit{displacement constraints: } \delta_{min}<\delta_j<\delta_{max} \quad j=1,2,3,......nn \\ & \textit{cross-section constraints: } A_{min}<A_k<A_{max} \quad k=1,2,3,......ng \end{aligned} \tag{4}$$

in which, *W(A)* is the weight of the structure as a function of the cross-section A of the structural elements, γ is the material density of the structural member *i*, L is the length of member *i*, σ_i^c and σ_i^t define the stress limits in tension and compression stresses for member *i*. δ_{min} and δ_{max} define the displacement limits for node *j*. *nm* is the number of elements, *nn* is the number of nodes, *ng* is the number of groups of elements for a specific design (problem), where for each case the structural elements are grouped based on the loading conditions and design specifics.

3.1 PENALTY FUNCTION

Structural optimization problems are unconstrained problems. In order to deal with constraints penalty functions are employed.

The penalty function in this work is defined as follows [56]:

$$P=(1+\phi)^c \tag{5}$$

where c takes a fixed value of 2 in this work, however, it might take an updating value when needed to sharpen the influence of penalty [56], φ is the combined penalties of the stress and displacement constraints, expressed as [56]:

$$\phi=\sum_{1}^{nm}\phi_s+\sum_{1}^{nj}\phi_d \tag{6}$$

Where *nm* is the number of members in the structure, and *nj* is the number of joints, the stress constraint penalty ϕ_s for member i, and the displacement constraint penalty ϕ_d for node *j* are defined as [46]:

$$\begin{cases} \phi_s=0 & \textit{if there is no violations of the constraints} \\ \phi_s=\dfrac{|\sigma_i-\sigma_{allowable}|}{|\sigma_{allowable}|} & \textit{in case the stress in member i violates the stress boundaries} \end{cases} \tag{7}$$

$$\begin{cases} \phi_d = 0 & \textit{if there is no violations of the constraints} \\ \phi_d = \dfrac{\left|\delta_j - \delta_{permissible}\right|}{\left|\delta_{permissible}\right|} & \textit{in case the joint j displacement violates the permissible boundaries} \end{cases} \tag{8}$$

where, $\sigma_{allowable}$ defines the stress limit in member i, $\delta_{permissible}$ defines the displacement limit for joint *j*.

The stress and displacement violations are considered according to the following expression [41]:

$$W_P = W(A) \cdot P \tag{9}$$

Equation (9) is the evaluation function used to choose the best design that has fewer constraints' violations, while the goal function is *W(A)*.

4. IMPLEMENTATION OF THE HURRICANE ALGORITHM FOR TRUSS OPTIMIZATION

1. The proposed algorithm is a population-based algorithm, hence, the first step is to generate a random population that represents possible solutions for the problem. The upper and lower limits for design variables are set for each example separately. The values for each solution vector are decided according to the following Eq. [56]:

$$x0_i^d = x_{min}^d + rand() \cdot (x_{max}^d - x_{min}^d) \tag{10}$$

where $rand()$ is a randomly generated value in the [0,1] interval, x_{min}^d and x_{max}^d are the upper and lower limits for the design vector $x0_v^d$ on the d^{th} dimension.

2. Calculate the structure weight *W(A)*; stress σ_i and displacement δ_j violations; penalty functions *P*; and the penalized weight W_p according to Eqs. (4-9).

3. Compare the results from all particles in the population using the penalized weight W_p from step 2 and save the best weight as the current best weight.

4. Update the population according to Eq. (3).

5. Update the best function value W_p, whereas, at the end of each iteration:

a. The algorithm compares the best obtained weight W_p in the current iteration with the current best weight from previous iterations.

b. If any design in the current iteration has a lower penalized weight W_p than the current best weight it will automatically replace it, otherwise the algorithm keeps current best weight unchanged.

6. Repeat steps (2.-5.) until the maximum number of iterations (structural analysis) is reached.

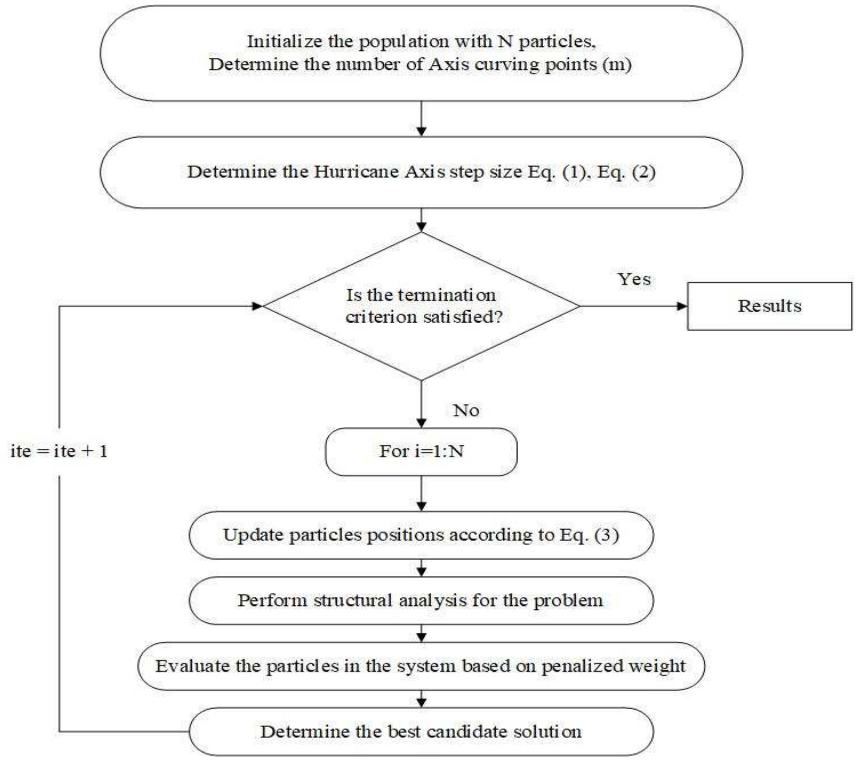


Fig. 3 *Flowchart of structural optimization process with the proposed algorithm*

5. TESTING THE ALGORITHM

To test this algorithm, three examples of structural optimization benchmarks were considered. Namely, the *Ten-bar* planar truss; the Twenty-five bar spatial truss; and the Seventy-two bar spatial truss that are demonstrated hereinafter. For the test, *50* independent runs were executed, and each run completed *2000* iterations with *40,000* structural analyses for each run. Each run is terminated when it reaches the maximum number of iterations. In all the tests, the number of initial populations is set to *20* particles. The factors of inertia, curvature and acceleration are set as $a_1 = a_2 = a_3 = 1$. The number of curving points is chosen for each case to find the minimum value of weight for each example.

5.1 TEN-BAR PLANAR TRUSS

The *10-bar* truss problem is a common example in the field of structural optimization. Figure 4 shows the structure's conditions for this test. The material density is *2767.990 kg/m³* and the elasticity modulus is *68,950 MPa*. The stress limit for each_member is *172.375 MPa* in both tension and compression, while all nodes are subjected to displacement limits of *5.08 cm* in both vertical and horizontal directions. In this example, the algorithm deals with *10* design variables ranging from *0.6452 cm²* to *225.806 cm²*. Two load cases are studied: Case 1, *P1 = 444.8 kN* and *P2 = 0*; and Case 2, *P1 = 667.2 kN* and *P2 = 222.4 kN*. Many researchers dealt with this problem, e.g., Lee and Geem employed the harmony search HS algorithm [33], Sonmez used the ABC algorithm [37], Camp et al. used the TLBO [46] and GA algorithm [70], and Li et al. utilized different variations of the PSO algorithm [20]. The results are presented in Table 1 and Table 2 for load case 1 and load case 2, respectively.

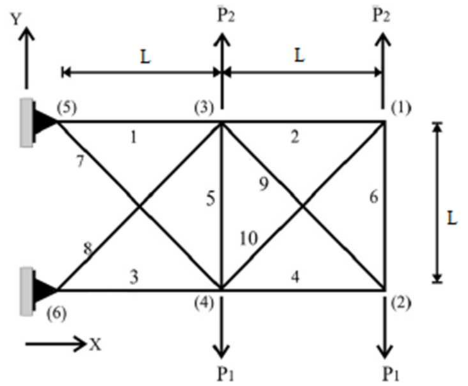


Fig. 4 *Schematic of the structure of the 10-bar truss problem, L=914.4 cm*

Table 1 and Table 2 show that the proposed algorithm found the lightest weight of all other methods, however, it needed double the number of structural analyses compared to the HS algorithm [33] yet much less than the PSO algorithm [20]. In this example the number of curving points on the hurricane axis is set to *10* for load case 1 and 5 for load case 2.

The proposed algorithm found the best design to be *2281.434 kg* and *2086.162 kg* of weight, for case 1 and case 2 respectively, this shows that the proposed algorithm has proven superior to the other methods in finding the lightest design. The proposed algorithm completed *40,000* structural analyses compared to *125,000* and *150,000* structural analyses for HPSP and PSO algorithms respectively. However, the HS and the EHS algorithms needed *20,000* and *11,402* structural analyses to finish the task in case 2.

Table 1 *Results of optimized design for 10-bar truss compared to previous researchers' work (load case 1)*

Design variables [cm²]	*Lee and Geem [33]*	*Sonmez [37]*	*Camp et al. [46]*	*Camp et al. [70]*	*Li et al. [20]*		*This study*
	HS	*ABC*	*TLBO*	*GA*	*PSOPC*	*PSO*	*HCOA*
A_1	*194.516*	*197.083*	*197.857*	*186.580*	*197.219*	*215.929*	*141.986*
A_2	*0.658*	*0.6452*	*0.6452*	*0.6452*	*0.6452*	*0.7097*	*0.6452*
A_3	*146.516*	*149.548*	*149.406*	*155.290*	*148.219*	*149.529*	*164.811*
A_4	*98.516*	*98.18*	*98.210*	*90.064*	*97.729*	*99.839*	*95.797*
A_5	*0.658*	*0.6452*	*0.6452*	*0.6452*	*0.6452*	*23.542*	*0.6452*
A_6	*3.51*	*3.555*	*3.497*	*3.613*	*3.529*	*0.748*	*0.6452*
A_7	*48.652*	*48.148*	*135.648*	*141.613*	*48.342*	*53.729*	*119.541*
A_8	*139.096*	*135.858*	*48.164*	*49.613*	*136.509*	*150.580*	*42.760*
A_9	*138.387*	*138.716*	*0.6452*	*0.6452*	*136.490*	*148.477*	*0.6452*
A_{10}	*0.6452*	*0.6452*	*138.490*	*142.516*	*0.6452*	*1.226*	*170.856*
Number of structure analyses	*20,000*	$500*10^3$	*NA*	*NA*	*150,000*	*150,000*	*40,000*
Weight (kg)	*2294.216*	*2295.576*	*2295.619*	*2302.576*	*2295.631*	*2508.139*	*2281.434*

Table 2 *Results of optimized design for 10-bar truss compared to previous researchers' work (load case 2)*

	Lee and Geem [33]	*Sonmez [37]*	*Li et al. [20]*		*Kaveh and Talatahari [69]*	*Degertekin [32]*	*This study*
Design variables [cm²]	*HS*	*ABC*	*HPSO*	*PSO*	*HPSACO*	*EHS*	
A_1	*149.999*	*151.4692*	*150.664*	*147.967*	*149.638*	*152.187*	*143.998*
A_2	*0.6581*	*0.6484*	*0.6452*	*0.729*	*0.6452*	*0.6452*	*0.6452*
A_3	*165.9997*	*162.834*	*164.529*	*163.580*	*158.613*	*164.013*	*210.351*
A_4	*93.613*	*92.606*	*91.935*	*92.729*	*91.748*	*93.471*	*51.701*
A_5	*0.6452*	*0.6458*	*0.6452*	*0.6452*	*0.6452*	*0.6452*	*0.6452*
A_6	*12.755*	*12.710*	*12.723*	*12.839*	*12.703*	*12.742*	*19.023*
A_7	*78.774*	*80.082*	*79.761*	*79.651*	*80.574*	*79.755*	*102.627*
A_8	*81.355*	*83.177*	*83.768*	*83.374*	*83.387*	*81.819*	*102.126*
A_9	*131.355*	*131.189*	*131.329*	*133.406*	*135.174*	*131.109*	*0.6452*
A_{10}	*0.6452*	*0.6452*	*0.652*	*0.6452*	*0.6516*	*0.6452*	*75.997*
Number of structure analyses	*20,000*	*500*10³*	*125,000*	*150,000*	*10,650*	*11,402*	*40,000*
Weight (kg)	*2117.737*	*2121.487*	*2121.583*	*2122.572*	*2120.898*	*2122.368*	*2086.162*

5.2 TWENTY-FIVE-BAR SPATIAL TRUSS

Figure 5 shows the *25-bar* spatial truss, the material density is $2767.990\ kg/m^3$ and the modulus of elasticity is *68,950 MPa*. For this example, the structural members of the *25-bar* truss are grouped into eight different groups, as given in Table 3, the structure was optimized under two independent loading conditions as presented in Table 4. The stress limits for each group are described in Table 3, while all nodes are subjected to displacement limits of ± 0.889 *cm* in both vertical and horizontal directions. In this example, the algorithm deals with *8* design variables ranging from $0.06452\ cm^2$ to $21.94\ cm^2$. Lamberti solved this problem using an improved SA algorithm [25]. The results and comparison are presented in Table 5.

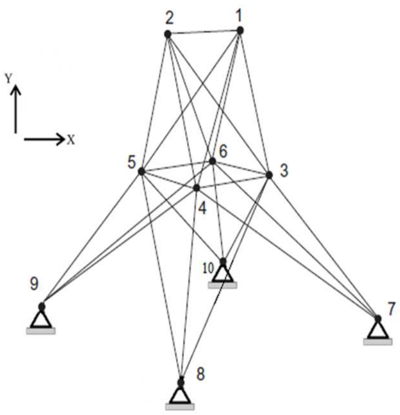


Fig. 5 *The structure of the 25-bar truss*

Table 3 *Characteristics of the 25-bar truss*

Node ID	Nodal coordinats			Group ID	Group members	Stress limitations in tension [MPa]	Stress limitations in compression [MPa]
	X [cm]	Y [cm]	Z [cm]				
1	-95.25	0	508	1	1	257.7903	241.951
2	95.25	0	508	2	2,3,4,5	257.7903	79.910
3	-95.25	95.25	254	3	6,7,8,9	257.7903	119.313
4	95.25	95.25	254	4	10,11	257.7903	241.951
5	95.25	-95.25	254	5	12,13	257.7903	241.951
6	-95.25	-95.25	254	6	14,15,16,17	257.7903	46.602
7	-254	254	0	7	18,19,20,21	257.7903	46.602
8	254	254	0	8	22,23,24,25	257.7903	76.4077
9	254	-254	0				
10	-254	-254	0				

Table 4 *Loading conditions for 25-bar truss*

Node ID	Condition 1			Condition 2		
	P_x [kN]	P_y [kN]	P_z [kN]	P_x [kN]	P_y [kN]	P_z [kN]
1	0.0	88.9644	-22.241	4.4482	44.482	-2.2241
2	0.0	-88.9644	-22.241	0.0	44.482	-2.2241
3	0.0	0.0	0.0	2.2241	0.0	0.0
6	0.0	0.0	0.0	2.2241	0.0	0.0

Table 5 *Results of optimized design for 25-bar truss compared to previous researchers' work*

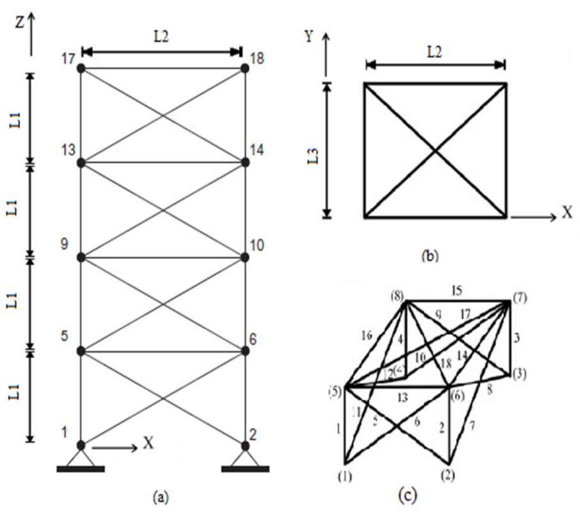
Design variables [cm²]	Lee and Geem [33]	Li et al. [20]	Lamberti [25]	This study
	HS	HPSO	Improved SA	
A_1	0.3032	0.06452	0.06452	4.5652
A_2	13.045	12.7097	12.8193	8.1664
A_3	19.032	19.458	19.3129	20.6683
A_4	0.06452	0.06452	0.06452	0.06452
A_5	0.0903	0.06452	0.06452	0.06452
A_6	4.439	4.4774	4.4477	4.6013
A_7	10.690	10.845	10.8187	14.0077
A_8	17.181	17.0516	17.1748	13.2213
Number of structure analyses	15,000	125,000	1050	40,000
Weight (kg)	246.927	247.294	247.276	239.015

In this case, the algorithm found the best weight of *239.015 kg* compared to *246.927*, *247.294* and *247.276 kg* for HS, HPSO and improved SA algorithms, respectively. However, the number of structural analyses is still an issue, where the improved SA and HS algorithms could solve this problem with *1050* and *15000* structural analyses compared to *40,000* structural analyses for the proposed algorithm, while the proposed algorithm still performed better than HPSO with *125,000* structural analyses. In this example, the number of curving points on the hurricane axis is set to *30*, in this case, the curvature points play the role of pseudo population. However, according to the algorithm concept, this does not affect the number of structural analyses.

5.3 SEVENTY-TWO-BAR SPATIAL TRUSS

The *72-bar* spatial truss is shown in Figure 6, the modulus of elasticity is *68,950 MPa* and the material density is *2767.990 kg/m³*. The displacement limits of *0.635 cm* are applied to the upper four nodes in both vertical and horizontal directions. The stress limit for each member is *172.375 MPa* in both tension and compression. For this case the structure was optimized under two loading conditions as presented in Table 6, design variables for structural members are divided into *16* groups: (1) A_1–A_4, (2) A_5–A_{12}, (3) A_{13}–A_{16}, (4) A_{17}–A_{18}, (5) A_{19}–A_{22}, (6) A_{23}–A_{30}, (7) A_{31}–A_{34}, (8) A_{35}–A_{36}, (9) A_{37}–A_{40}, (10) A_{41}–A_{48}, (11) A_{49}–A_{52}, (12) A_{53}–A_{54}, (13) A_{55}–A_{58}, (14) A_{59}–A_{66}, (15) A_{67}–A_{70}, (16) A_{71}–A_{72}. The results of optimization are given in Table 7 for loading conditions in case 1, while Table 8 provides results for loading conditions in case 2.

In this example, the number of curving points on the hurricane axis is set to *20* for load case 1 and *10* for load case 2. The results for load case 1 are abnormal where the algorithm showed a bad performance compared to all the other cases. There was no clear explanation for this behavior, as the procedure followed in all the cases was the same. The proposed algorithm found the best design for case 2 to be *147.059 kg* of weight compared to *165.498 kg* for the HPSO algorithm. Moreover, the proposed algorithm needed *40,000* structural analyses compared to *125,000* for HPSO algorithms.



L1 = 152.4 cm, L2 = 304.8 cm, L3 = 304.8 cm

Fig. 6 *Schematic of the 72-bar truss: (a) side view; (b) top view; (c) connectivity for one story*

Table 6 *Loading conditions for 72-bar truss*

Node ID	Case 1			Case 2		
	P_x [kN]	P_y [kN]	P_z [kN]	P_x [kN]	P_y [kN]	P_z [kN]
17	22.441	22.441	-22.441	0	0	-22.441
18	0	0	0	0	0	-22.441
19	0	0	0	0	0	-22.441
20	0	0	0	0	0	-22.441

Table 7 *Results of optimized design for 72-bar truss compared to previous researchers' work (load case 1)*

Design variables [cm²]	Lee and Geem [33]	Li et al. [20]	Degertekin [32]	Camp [41]	This study
	HS	HPSO	EHS	BB-BC	
Group 1	11.5484	11.9806	12.6903	11.9851	7.4645
Group 2	3.3613	35.5161	3.2903	3.2639	3.6213
Group 3	0.6452	0.6452	0.6452	0.6452	0.6452
Group 4	0.6452	0.6452	0.6452	0.6452	1.6916
Group 5	7.9290	8.0968	8.3419	8.0490	11.5026
Group 6	3.36774	3.2452	3.2968	3.3993	3.9548
Group 7	0.6452	0.6452	0.6452	0.6452	0.6452
Group 8	0.6452	0.6452	0.6452	0.6529	0.6452
Group 9	3.3355	3.1999	3.2193	3.3606	3.2323
Group 10	3.2516	3.2645	3.2323	3.3368	5.0722
Group 11	0.6452	0.6452	0.6452	0.6477	0.9097
Group 12	0.5616	0.6452	0.6452	0.6484	0.6452
Group 13	1.0064	0.6452	1.0323	1.0097	0.6452
Group 14	3.5290	3.3806	3.3677	3.5529	3.1535
Group 15	2.8516	2.5806	3.0839	2.5303	0.8406
Group 16	3.8064	3.4452	3.8129	3.8206	2.2168
Number of structure analyses	15,000	125,000	15,044	19,621	40,000
Weight (kg)	172.034	167.67	172.8323	172.297	180.458

Table 8 *Results of optimized design for 72-bar truss compared to previous researchers' work (load case 2)*

Design variables [cm²]	*Lee and Geem [33]* *HS*	*Li et al. [20]* *HPSO*	*Lamberti [25]* *Improved SA*	*Erbatur et al. [9]* *GA*	*This study*
Group 1	*12.6645*	*12.3032*	*1.0742*	*0.9999*	*3.9329*
Group 2	*3.1032*	*3.3806*	*3.4599*	*3.4516*	*0.8187*
Group 3	*0.0645*	*0.0645*	*2.8774*	*3.0968*	*5.0813*
Group 4	*0.071*	*0.0645*	*3.7168*	*3.3548*	*8.0516*
Group 5	*7.9548*	*8.3097*	*3.3593*	*2.9677*	*5.9535*
Group 6	*3.2645*	*3.3742*	*3.3419*	*3.4193*	*0.9116*
Group 7	*0.071*	*0.0645*	*0.0645*	*0.7742*	*1.9981*
Group 8	*0.0774*	*0.0645*	*0.7361*	*1.0645*	*1.4419*
Group 9	*3.471*	*3.5097*	*8.3245*	*7.4516*	*3.9123*
Group 10	*3.4387*	*3.4064*	*3.3355*	*3.7742*	*0.7999*
Group 11	*0.0645*	*0.1226*	*0.0645*	*0.6452*	*1.7806*
Group 12	*1.0774*	*0.129*	*0.0645*	*0.6452*	*5.2826*
Group 13	*1.0387*	*1.1355*	*12.1716*	*11.3226*	*9.0839*
Group 14	*3.4968*	*3.4516*	*3.3348*	*3.2581*	*1.1413*
Group 15	*3.0839*	*2.7484*	*0.0645*	*0.6774*	*0.9129*
Group 16	*3.5548*	*3.9484*	*0.0645*	*0.9999*	*5.2406*
Number of structure analyses	*20,000*	*125,000*	*N/A*	*N/A*	*40,000*
Weight (kg)	*165.257*	*165.498*	*165.018*	*174.978*	*147.059*

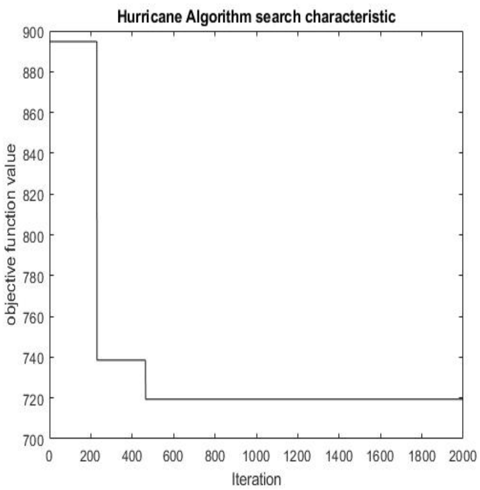
5.4 SENSITIVITY ANALYSIS

The case of *25 bar*s truss and *72 bars* truss were selected for the purpose of sensitivity analysis, where the number of curving points varied between *5-50*. The results did not show a clear correlation between the number of curving points and the results improvement. However, it showed that each case might have a specific number of curve points that could be optimal to find the minimum. The results of the sensitivity analysis are presented in Table 9. For the *25*-truss case, the best results were found to be *239.015 kg* of weight using *30* curving points, while the best weight in the *72*-truss case 2 was found to be *180.458 kg* of weight using *20* curving points.

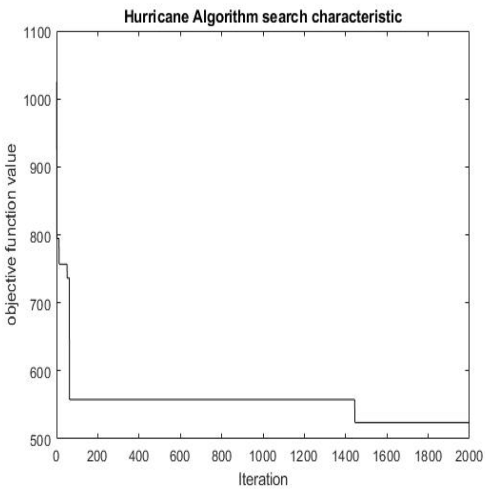
Table 9 *The results of the sensitivity analysis*

Number of curving points	*Best weight for 25 bar truss example / kg*	*Best weight for 72 bar truss example (load case1) / kg*
5	*248.079*	*201.979*
7	*548.286*	*199.959*
10	*255.668*	*201.306*
15	*249.203*	*194.847*
20	*241.037*	*180.458*
25	*257.574*	*203.32*
30	*239.015*	*205.758*
35	*249.497*	*210.082*
40	*243.146*	*205.985*
45	*255.852*	*195.793*
50	*246.943*	*202.667*

Figure 7 shows the search characteristics of the proposed algorithm. The plot shows that the minimum value changes in a stepwise manner, where the plot keeps a specific value for several iterations and then changes to a new minimum value when it is found. This is related to the algorithm concept as it goes throughout the solution space with the aim of hitting the solution at least once. The plot shows a searching characteristic rather than a convergence characteristic as in other algorithms.



a) Minimum for the best run; 25 bars truss



b) Minimum for the best run; 72 bars truss (load case1)

Fig. 7 *Convergence characteristics of the hurricane chaos algorithm*

6. CONCLUSIONS

This paper proposed a simple and efficient algorithm for solving structural design optimization problems. Three common examples from the literature were used to test the algorithm. The algorithm takes inspiration from a natural phenomenon, where it simulates the chaotic nature of a hurricane system. The results showed that the proposed algorithm could achieve good performance overall compared to the referenced algorithms from the literature. However, the algorithm showed abnormal behavior in one case study. Moreover, the algorithm needs a high number of structural analyses to achieve a good performance. The simplicity of the algorithm seems like an advantage, however, it has a disadvantage where it needs to decide the proper number of hurricane axis curving points. This shows a need for a dynamic updating system like PSO. Another way is to use the proposed algorithm in combination with another algorithm to improve the diversity of the algorithms.

DECLARATION

The authors confirm that this article's content has no conflicts of interest.

7. REFERENCES

[1] G. Alonso, E. del Valle, J.R. Ramirez, Optimization methods (chapter 5), *Desalination in Nuclear Power Plants*, Woodhead Publishing Series in Energy, pp. 67-76, 2020.
https://doi.org/10.1016/B978-0-12-820021-6.00005-3

[2] P. Wilson, H.A. Mantooth, Model-Based Optimization Techniques (chapter 10), *Model-Based Engineering for Complex Electronic Systems*, pp. 347-367, 2013.
https://doi.org/10.1016/B978-0-12-385085-0.00010-5

[3] S.S. Rao, *Engineering Optimization Theory and Practice*, fourth edition, New Jersey: John Wiley & SonsS, INC, 2009.

[4] M. Farshchin, M. Maniat, C.V. Camp, S. Pezeshk, School based optimization algorithm for design of steel frames, *Engineering Structures*, Vol. 171, pp. 326–335, 2018.
https://doi.org/10.1016/J.ENGSTRUCT.2018.05.085

[5] S.O. Degertekin, M.S. Hayalioglu, Sizing truss structures using teaching-learning-based optimization, *Computers and Structures*, Vol. 119, pp. 177–188, 2013.
https://doi.org/10.1016/j.compstruc.2012.12.011

[6] J. Holland, *Adaptation in Natural and Artificial Systems*, Ann Arbor: University of Michigan Press, 1975.

[7] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, New York: Addison-Wesley, 1988.

[8] T. Bäck, *Evolutionary Algorithms in Theory and Practice*, New York: Oxford University Press, 1996. https://doi.org/10.1093/oso/9780195099713.003.0007

[9] F. Erbatur, O. Hasançebi, İ. Tütüncü, H. Kılıç, Optimal design of planar and space structures with genetic algorithms, *Computers and Structures*, Vol. 75, No. 2, pp. 209–224, 2000. https://doi.org/10.1016/S0045-7949(99)00084-X

[10] I. Rechenberg, Cybernetic solution path of an experimental problem, in: Royal Aircraft Establishment, Library Translation No. 1122, Farnborough, Hants. U.K., 1965.

[11] M. Papadrakakis, N.D. Lagaros, G. Thierauf, J. Cai, Advanced solution methods in structural optimization based on evolution strategies, *Engineering Computations*, Vol. 15, No. 1, pp. 12-34, 1998. https://doi.org/10.1108/02644409810200668

[12] M.M. Efrén, M.E. Miranda-Varela, R.D.C. Gómez-Ramón, Differential evolution in constrained numerical optimization: an empirical study, *Information Sciences*, Vol. 180, No. 22, pp. 4223–4262, 2010. https://doi.org/10.1016/j.ins.2010.07.023

[13] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization*, Vol. 11, pp. 341–359, 1997. https://doi.org/10.1023/A:1008202821328

[14] E. Mezura-Montes, C.A.C. Coello, A simple multi-membered evolution strategy to solve constrained optimization problems, *IEEE Transactions on Evolutionary Computation*, Vol. 9, No. 1, pp. 1–17, 2005. https://doi.org/10.1109/TEVC.2004.836819

[15] E. Mezura-Montes, C.A.C. Coello, Useful infeasible solutions in engineering optimization with evolutionary algorithms, *Advances in Artificial Intelligence of LNCS*, Vol. 3789, Berlin: Springer-Verlag, 2005. https://doi.org/10.1007/11579427_66

[16] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory. MHS'95, Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pp. 39-43, (Japan), 1995. https://doi.org/10.1109/MHS.1995.494215

[17] J. Kennedy, R. C. Eberhart, Particle swarm optimization, in: Proceedings of *IEEE International Conference on Neural Networks*, Piscataway, NJ, pp.1942–1948, 1995.

http://dx.doi.org/10.1109/ICNN.1995.488968

[18] M. Clerc, *Particle Swarm Optimization*, London: ISTE Publishing Company, 2006.

https://doi.org/10.1002/9780470612163

[19] S. He, E. Prempain, Q.H. Wu, An improved particle swarm optimizer for mechanical design optimization problems, *Engineering Optimization*, Vol. 36, No. 5, pp. 585–605, 2004. https://doi.org/10.1080/03052150410001704854

[20] L.J. Li, Z.B. Huang, F. Liu, Q.H. Wu, A heuristic particle swarm optimizer for optimization of pin connected structures, *Computers and Structures*, Vol. 85, No. 7, pp. 340-349, 2007.

https://doi.org/10.1016/J.COMPSTRUC.2006.11.020

[21] Y. Lu, J. Jan, S. Hung, G. Hung, Enhancing particle swarm optimization algorithm using two new strategies for optimizing design of truss structures, *Engineering Optimization*, Vol. 45, No. 10, pp. 1251-1271 ,2013. https://doi.org/10.1080/0305215X.2012.729054

[22] J.D. Farmer, N. Packard, A. Perelson, The immune system, adaptation, and machine learning, *Physica D: Nonlinear Phenomena*, Vol. 22, No. (1-3), pp. 187–204, 1986.

https://doi.org/10.1016/0167-2789(86)90240-X

[23] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, *Science*, Vol. 220, pp. 671–680, 1983. https://doi.org/10.1126/science.220.4598.671

[24] P.J.M. Van Laarhoven, E.H.L. Aarts, *Simulated Annealing: Theory and Applications*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1987.

https://doi.org/10.1007/978-94-015-7744-1

[25] L. Lamberti, An efficient simulated annealing algorithm for design optimization of truss structures, *Computers and Structures*, Vol. 86, pp. 1936-1953, 2008.

https://doi.org/10.1016/j.compstruc.2008.02.004

[26] M. Dorigo, Optimization, Learning and Natural Algorithms, Ph.D. Thesis, Politecnico di Milano, Italy, 1992.

[27] M. Dorigo, V. Maniezzo, A. Colorni, The ant system: optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 26, No. 1, pp. 29-41, 1996. https://doi.org/10.1109/3477.484436

[28] M. Dorigo, T. Stützle, *Ant Colony Optimization*, MIT Press, Cambridge (MA), USA, 2004.

https://doi.org/10.7551/mitpress/1290.001.0001

[29] C. Blum, Ant colony optimization: introduction and recent trends, *Physics of Life Reviews*, Vol. 2, No. 4, pp. 353-373, 2005. https://doi.org/10.1016/j.plrev.2005.10.001

[30] K.S. Lee, Z.W. Geem, A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice, *Computer Methods in Applied Mechanics and Engineering*, Vol. 194, pp. 3902–3933, 2005.

https://doi.org/10.1016/j.cma.2004.09.007

[31] Z.W. Geem, J.H. Kim, G.V. Loganathan, A new heuristic optimization algorithm: harmony search, *Simulation*, Vol. 76, No. 2, pp. 60–68, 2001.

https://doi.org/10.1177/003754970107600201

[32] S.O. Degertekin, Improved harmony search algorithms for sizing optimization of truss structures, *Computers and Structures*, Vol. 92, pp. 229-241, 2012.

https://doi.org/10.1016/j.compstruc.2011.10.022

[33] K.S. Lee, Z.W. Geem, A new structural optimization method based on the harmony search algorithm, *Computers and Structures*, Vol. 82, pp.781-798, 2004.

https://doi.org/10.1016/j.compstruc.2004.01.002

[34] D. Karaboga, B. Akay, A modified artificial bee colony (ABC) algorithm for constrained optimization problems, *Applied Soft Computing*, Vol. 11, No. 3, pp. 3021-3031, 2011.

https://doi.org/10.1016/j.asoc.2010.12.001

[35] D. Karaboga, B. Basturk, Artificial Bee Colony (ABC) Optimization Algorithm for Solving Constrained Optimization Problems (chapter), *Foundations of Fuzzy Logic and Soft Computing*, eds. P. Melin, O. Castillo, L.T. Aguilar, J. Kacprzyk, W. Pedrycz, Lecture Notes in *Computer Science*, Vol. 4529, Springer, Berlin, Heidelberg, pp. 789–798, 2007.

https://doi.org/10.1007/978-3-540-72950-1_77

[36] D. Karaboga, B. Basturk, On the performance of artificial bee colony (ABC) algorithm, *Applied Soft Computing*, Vol. 8, No. 1, pp. 687-697, 2008.

https://doi.org/10.1016/j.asoc.2007.05.007

[37] M. Sonmez, Artificial Bee Colony algorithm for optimization of truss structures, *Applied Soft Computing*, Vol. 11, No. 2, pp. 2406-2418, 2011.

https://doi.org/10.1016/j.asoc.2010.09.003

[38] E. Rashedi, H.N. Pour, S. Saryazdi, GSA: a gravitational search algorithm, *Information Sciences*, Vol. 179, No. 13, pp. 2232-2248, 2009.

https://doi.org/10.1016/j.ins.2009.03.004

[39] M.M. Eusuff, E. K. Lansey, Optimization of water distribution network design using the shuffled frog leaping algorithm, *Journal of Water Resources Planning and Management*, Vol. 129, No. 3, pp. 210–225, 2003.

https://doi.org/10.1061/(ASCE)0733-9496(2003)129:3(210)

[40] O. K. Erol, I. Eksin, A new optimization method: big bang-big crunch, *Advances in Engineering Software*, Vol. 37, No. 2, pp. 106–111, 2006.

https://doi.org/10.1016/j.advengsoft.2005.04.005

[41] C.V. Camp, Design of space trusses using big bang–big crunch optimization, *Journal of Structural Engineering*, Vol. 133, No. 7, pp. 999-1008, 2007.

https://doi.org/10.1061/(ASCE)0733-9445(2007)133:7(999)

[42] A. Kaveh, S. Talatahari, A novel heuristic optimization method: charged system search, *Acta Mechanica*, Vol. 213, No. 3, pp. 267–289, 2010.

https://doi.org/10.1007/s00707-009-0270-4

[43] R.V. Rao, V.J. Savsani, D.P. Vakharia, Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems, *Computer-Aided Design*, Vol. 43, No. 3, pp. 303-315, 2011. https://doi.org/10.1016/j.cad.2010.12.015

[44] R.V. Rao, V.J. Savsani, D.P. Vakharia, Teaching–Learning-Based Optimization: An optimization method for continuous non-linear large scale problems, *Information Sciences*, Vol. 183, No. 1, pp. 1-15, 2012. https://doi.org/10.1016/j.ins.2011.08.006

[45] M. Farshchin, C.V. Camp, M. Maniat, Multi-class teaching–learning-based optimization for truss design with frequency constraints, *Engineering Structures*, Vol. 106, pp. 355-369, 2016. https://doi.org/10.1016/j.engstruct.2015.10.039

[46] C.V. Camp, M. Farshchin, Design of space trusses using modified teaching–learning based optimization, *Engineering Structures*, Vol. 62-63, pp. 87-97, 2014.

https://doi.org/10.1016/j.engstruct.2014.01.020

[47] A. Kaveh, S. Talatahari, Optimum design of skeletal structures using imperialist competitive algorithm, *Computers and Structures*, Vol. 88, No. (21-22), pp. 1220-1229, 2010. https://doi.org/10.1016/j.compstruc.2010.06.011

[48] X.S. Yang, Flower Pollination Algorithm for Global Optimization, (chapter), *Unconventional Computation and Natural Computation*, eds. J. Durand-Lose, N. Jonoska, Lecture Notes in *Computer Science*, Vol. 7445, Springer, Berlin, Heidelberg, pp. 240–249, 2012. https://doi.org/10.1007/978-3-642-32894-7_27

[49] M. Neshat, G. Sepidnam, M. Sargolzaei, Swallow swarm optimization algorithm: a new method to optimization, *Neural Computing and Applications*, Vol. 23, No. 2, 429-454, 2013. https://doi.org/10.1007/s00521-012-0939-9

[50] A. Kaveh, T. Bakhshpoori, Water evaporation optimization: a novel physically inspired optimization algorithm, *Computers and Structures*, Vol.167, pp. 69–85, 2016.

https://doi.org/10.1016/j.compstruc.2016.01.008

[51] H. Eskandar, A. Sadollah, A. Bahreininejad, M. Hamdi, Water cycle algorithm – A novel metaheuristic optimization method for solving constrained engineering optimization problems, *Computers and Structures*, Vol. 110-111, pp. 151-166, 2012.

https://doi.org/10.1016/j.compstruc.2012.07.010

[52] P. Savsani, V. Savsani, Passing Vehicle Search (PVS): A novel metaheuristic algorithm, *Applied Mathematical Modelling*, Vol. 40, No. (5–6), pp. 3951–3978, 2015.

https://doi.org/10.1016/j.apm.2015.10.040.

[53] Y.J. Zheng, Water wave optimization: A new nature-inspired metaheuristic, *Computers and Operations Research*, Vol. 55, pp. 1-11, 2015.

https://doi.org/10.1016/j.cor.2014.10.008

[54] R.V. Rao, Jaya, A simple and new optimization algorithm for solving constrained and unconstrained optimization problems, *International Journal of Industrial Engineering Computations*, Vol. 7, No. 1, pp. 19-34, 2016.

https://doi.org/10.5267/j.ijiec.2015.8.004

[55] R.V. Rao, G.G. Waghmare, A new optimization algorithm for solving complex constrained design optimization problems, *Engineering Optimization*, Vol. 49, No. 1, pp. 60-83, 2017.

https://doi.org/10.1080/0305215X.2016.1164855

[56] S.O. Degertekin, L. Lamberti, I.B. Ugur, Sizing, layout and topology design optimization of truss structures using the Jaya algorithm, *Applied Soft Computing*, pp. 903-928, 2018.

https://doi.org/10.1016/j.asoc.2017.10.001

[57] A. Kaveh, V.R. Mahdavi, Colliding Bodies Optimization method for optimum discrete design of truss structures, *Computers and Structures*, Vol. 139, pp. 43-53, 2014.

https://doi.org/10.1016/j.compstruc.2014.04.006

[58] A. Kaveh, M. Ilchi Ghazaan, Enhanced colliding bodies optimization for design problems with continuous and discrete variables, *Advances in Engineering Software*, Vol. 77, pp. 66-75, 2014. https://doi.org/10.1016/j.advengsoft.2014.08.003

[59] A. Kaveh, M. Ilchi Ghazaan, A comparative study of CBO and ECBO for optimal design of skeletal structures, *Computers and Structures*, Vol. 153, pp. 137-147, 2015.

https://doi.org/10.1016/j.compstruc.2015.02.028

[60] W.T. Pan, A New Fruit Fly Optimization Algorithm: Taking the financial distress model as an example, *Knowledge-Based Systems*, Vol. 26, pp. 69-74, 2012.

https://doi.org/10.1016/j.knosys.2011.07.001

[61] L. Wang, Y.L. Shi, S. Liu, An improved fruit fly optimization algorithm and its application to joint replenishment problems, *Expert Systems with Applications*, Vol. 42, No. 9, pp. 4310-4323, 2015. https://doi.org/10.1016/j.eswa.2015.01.048

[62] H.D. Dai, A.L. Liu, J.H. Lu, S.W. Dai, X.N. Wu, Y.Y. Sun, Optimization about the layout of IMUs in large ship based on fruit fly optimization algorithm, *Optik*, Vol. 126, No. 4, pp. 490-493, 2015. https://doi.org/10.1016/j.ijleo.2014.08.037

[63] A. Ahrari, A.A. Atai, Grenade explosion method – a novel tool for optimization of multimodal functions, *Applied Soft Computing*, Vol. 10, No. 4, pp. 1132-1140, 2010.

https://doi.org/10.1016/j.asoc.2009.11.032

[64] J. Liu, L. Tang, A modified genetic algorithm for single machine scheduling, *Computers and Industrial Engineering*, Vol. 37, No. (1-2), pp. 43-46, 1999.

https://doi.org/10.1016/S0360-8352(99)00020-0

[65] C.L. Sun, J.C. Zeng, J.S. Pan, An improved vector particle swarm optimization for constrained optimization problems, *Information Sciences*, Vol. 181, No. 6, pp. 1153-1163, 2011. https://doi.org/10.1016/j.ins.2010.11.033

[66] L.L. Liu, S.X. Yang, D.W. Wang, Force-imitated particle swarm optimization using the near-neighbor effect for locating multiple optima, *Information Sciences*, Vol. 182, No. 1, pp. 139-155, 2011. https://doi.org/10.1016/j.ins.2010.11.013

[67] A.R. Yildiz, A novel hybrid immune algorithm for global optimization in design and manufacturing, *Robotics and Computer-Integrated Manufacturing*, Vol. 25, No. 2, pp. 261-270, 2009. https://doi.org/10.1016/j.rcim.2007.08.002

[68] İ. Karen, A.R. Yildiz, N. Kaya, N. Öztürk, F. Öztürk, Hybrid approach for genetic algorithm and Taguchi's method based design optimization in the automotive industry, *International Journal of Production Research*, Vol. 44, No. 22, pp. 4897-4914, 2006.

https://doi.org/10.1080/00207540600619932

[69] A. Kaveh, S. Talatahari, Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures, *Computers and Structures*, Vol. 87, No. (5-6), pp. 267-283, 2009 https://doi.org/10.1016/j.compstruc.2009.01.003

[70] C.V. Camp, S. Pezeshk, G. Cao, Optimized design of two-dimensional structures using a genetic algorithm, *Journal of Structural Engineering*, Vol. 124, No. 5, pp. 551-559, 1998.

https://doi.org/10.1061/(ASCE)0733-9445(1998)124:5(551)

[71] H.A. Pham, Truss optimization with frequency constraints using enhanced differential evolution based on adaptive directional mutation and nearest neighbor comparison, *Advances in Engineering Software*, Vol. 102, pp. 142–154, 2016.

https://doi.org/10.1016/j.advengsoft.2016.10.004