Structural Engineering and Mechanics, *Vol. 52, No. 2 (2014) 405-426*
DOI: http://dx.doi.org/10.12989/sem.2014.52.2.405

A new hybrid meta-heuristic for structural design: ranked particles optimization

A. Kaveh[*] and A. Nasrollahi

Centre of Excellence for Fundamental Studies in Structural Engineering, School of Civil Engineering, Iran University of Science and Technology, Narmak, Tehran-16, Iran

(Received March 25, 2013, Revised May 27, 2014, Accepted July 20, 2014)

Abstract. In this paper, a new meta-heuristic algorithm named *Ranked Particles Optimization* (*RPO*), is presented. This algorithm is not inspired from natural or physical phenomena. However, it is based on numerous researches in the field of meta-heuristic optimization algorithms. In this algorithm, like other meta-heuristic algorithms, optimization process starts with by producing a population of random solutions, *Particles*, located in the feasible search space. In the next step, cost functions corresponding to all random particles are evaluated and some of those having minimum cost functions are stored. These particles are ranked and their weighted average is calculated and named *Ranked Center*. New solutions are produced by moving each particle along its previous motion, the ranked center, and the best particle found thus far. The robustness of this algorithm is verified by solving some mathematical and structural optimization problems. Simplicity of implementation and reaching to desired solution are two main characteristics of this algorithm.

Keywords: meta-heuristic optimization algorithm; Ranked Particles Optimization; RPO; particle; ranked center

1. Introduction

There are two general approaches for optimization, namely, mathematical programming and meta-heuristic algorithms. Some of mathematical programming approaches are linear programming, homogenous linear programming, integer programming, dynamic programming, and nonlinear programming which have been applied in many optimization problems. Most of these methods use gradient information to search the solution space near an initial starting point. In general, gradient-based methods converge faster and can obtain solutions with higher accuracy compared to stochastic approaches in fulfilling the local search task. However, for effective implementation of these methods, the variables and cost function of the generators should be continuous. Also, a good starting point is vital for these methods to be executed successfully. In many optimization problems, prohibited zones, side limits, and non-smooth or non-convex cost functions need to be considered. As a result, non-convex optimization problems cannot be solved by the traditional mathematical programming methods. Although dynamic programming or mixed integer nonlinear programming and their modifications offer some facility in solving non-convex

∗Corresponding author, Professor, E-mail: alikaveh@iust.ac.ir

problems, these methods, in general, require considerable computational effort.

Nature and physics have always been two major sources of inspiration and most of the meta-heuristic algorithms are inspired by solutions that nature herself seems to have chosen for hard problems. The Evolutionary Algorithm (EA) proposed by Fogel *et al*. (1966), De Jong (1975) and Koza (1990), and the Genetic Algorithm (GA) proposed by Holland (1975) and Goldberg (1989) are inspired from the biological evolutionary process. Studies on animal behavior led to Ant Colony Optimization (ACO) proposed by Dorigo *et al*. (1995) and Particle Swarm Optimizer (PSO) formulated by Eberhart and Kennedy (1995). Also, Simulated Annealing (SA) proposed by Kirkpatrick *et al*. (1983), Charged System Search (CSS) proposed by Kaveh and Talatahari (2010a), the Magnetic Charged System Search (MCSS) presented by Kaveh *et al*. (2013), and Colliding Bodies Optimization (CBO) developed by Kaveh and Mahdavi (2014) are introduced utilizing physical and mechanical phenomena.

Each of the above mentioned algorithms has some benefits and drawbacks. For example, the implementation the PSO is easy and can search a continuous search space; however a lack of balance between exploration and exploitation reduces its robustness. Instead of devising a new meta-heuristic from zero, we can utilize the existing features in a new meta-heuristic to create a new robust algorithm. For instance, we can choose the PSO principals and use ranked selection method in GA to make the required balanced search in PSO. Also, we can use single agent meta-heuristic algorithms, which do not provide capability of parallel processing, to deal with the violated particles from the feasible search space.

The objective of this paper is to present a new optimization algorithm based on principles from existing researches in the field of meta-heuristic optimization algorithms, which will be called Ranked Particles Optimization (RPO). The main sources of inspiration of this new algorithm are Particle Swarm algorithm, Big Bang-Big Crunch by Erol and Eksin (2006), Genetic Algorithm, and Harmony search by Geem *et al.* (2001).

The remainder of this paper is organized as follows: Section 2 presents the basic aspects and the characteristics of the RPO. Numerical examples are presented in Section 3 to verify the efficiency of the new algorithm, and some concluding remarks are provided in Section 4.

2 Ranked particles optimization

2.1 Background

There has been great effort to create and improve various meta-heuristic optimization algorithms and this has led to various useful tools in this field. Many biological and social interactions between natural systems and many physical laws are utilized in optimization algorithms. On the other hand, many special concepts such as how to deal with violated variables from feasible search space, balancing global and local searches, and how to handle the constraints are developed to enhance the quality of the obtained solution. Each algorithm has its own characteristics and so none of them is perfect. Utilizing these useful tools in one algorithm can leads to a better optimization algorithm. Elements of an optimization problem are: (a) Cost Function, (b) Design variable (Solution), (c) Constraints, and (d) Search Space

And the relations between the abovementioned elements are as follows

$$Minimize:\ f(X) \tag{1a}$$

$$X = \lfloor x_1, x_2, \ldots, x_p \rfloor \tag{1b}$$

$$\textit{Subjected to: } \begin{cases} g_1(X) \le 0 \\ g_2(X) \le 0 \\ \ldots \\ g_m(X) \le 0 \end{cases}, \quad \textit{and} \quad \textit{Subjected to: } \begin{cases} h_1(X) = 0 \\ h_2(X) = 0 \\ \ldots \\ h_n(X) = 0 \end{cases} \tag{1c}$$

$$\textit{In which: } X_{\min} \le X \le X_{\max} \tag{1d}$$

Where, *f*(*X*) is the cost function; *X* is the design vector (solution) consisting of *p* independent variables x_1, x_2, …, and x_p; $g_1(X)$, $g_2(X)$,…, and $g_m(X)$ are *m* unequal constraints; $h_1(X)$, $h_2(X)$,…, and $h_n(X)$ are *n* equal constraints; and $X_{\min}$ and $X_{\max}$ are minimum and maximum feasible design vectors, therefore Eq. (1d) denotes the feasible search space of the problem. Note that there is not any relation between *p*, *n*, and *m*.

Steps of most of the meta-heuristic algorithms are as follows:

2.1.1 Initialization

Most of the meta-heuristic algorithms need a population of initial solutions. Usually, these initial solutions are produced randomly in the search space.

2.1.2 Searching for a better solution in the feasible search space

Having random initial solutions, the existing solutions should be updated using a logical manner. This process is called searching. In this step, solutions are updated iteratively using a search engine which is inspired from nature or physics.

To reach a good solution, search engine of each algorithm should provide two main phases which are (a) *Exploration* (diversification or global search), and (b) *Exploitation* (intensification or local search). At initial iterations, the algorithm should perform a global search and cover the whole search space. In this stage, some points which are expected to be near the global minimum of the cost function are found. Then at the latest iterations, the algorithm should perform a local search using the solution vectors found so far to increase the precision of the solution. In every meta-heuristic algorithm, there should be a balance between exploration and exploitation. Further exploration diversifies the optimization process and brings down the precision of the solution. On the contrary, further exploitation intensifies the optimization process the risk of finding a local optimum instead of a global optimum increases.

2.1.3 Stopping criteria

There are some criteria to finish the iterative process. Some of them are:

• Maximum number of iterations: the optimization process is terminated after a fixed number of iterations, for example, 1,000 iterations.

• Number of iterations without improvement: the optimization process is terminated after some fixed number of iterations without any improvement.

• Minimum objective function error: the difference between the values of the best objective function and the global optimum is less than a pre-fixed anticipated threshold.

• Difference between the best and the worst solutions: the optimization process is stopped if the difference between the objective values of the best and the worst solutions becomes less than a specified accuracy.

2.2 Presentation of ranked particles optimization

In this section the new meta-heuristic algorithm, named *Ranked Particles Optimization, RPO* is presented. First, items of the algorithm and terminology of parameters existing in the RPO are defined here. These definitions are as follows:

(a) Particle, *P*: Each solution in this algorithm is named a *Particle*. Number of particles is the population of the algorithm.

(b) Particles' Memory, *PM*: In RPO there is a memory in which some of the best particles are stored. Size of Particles' Memory is named *PMS* and it depends on the problem. A large value of PMS results in high diversification and low values provides intensification for the algorithm.

(c) Rank, *R*: Each particle in the PM is ranked such that the best particle has the rank of PMS and this rank decreases for each particle by unity, thus the rank of the worst particle in the PM is 1.

(d) Ranked Center, *C*: ranked center of particles existed in the PM is calculated using Eq. (2)

$$C = \frac{\sum_{i=1}^{PMS} R_i \times X_i}{\sum_{i=1}^{PMS} R_i} \tag{2}$$

This definition of *C* prevents the algorithm to be greedy but *C* is nearer to the best particle; this is inspired from selection process in the Genetic Algorithm using Rank method instead of roulette wheel. For example, if in this definition, ranked center is obtained using the inverse of the cost function, *C* will again be nearer to the best particle but this approach will intensify the algorithm and the risk of being trapped in the local minimum increases.

(e) Velocity, *V*: velocity of each particle is the subtraction of new and previous location of the particle. Thus

$$X_i^{k+1} = X_i^{k} + V_i^{k+1} \tag{3}$$

In the RPO, the velocity of each particle is updated using three movements: (i) moving in the direction of the previous velocity, (ii) moving in the direction of Ranked Center, (iii) Moving in the direction of the best particle and is formulated as

$$V_i^{k+1} = \alpha\left(1-\frac{k}{\max iteration}\right)^{\beta} V_i^k + rand_1 \times r \times \left(C^k - X_i^k\right) + rand_2 \times \left(X_{best}^k - X_i^k\right) \tag{4}$$

Where, α and β are two constants which control convergence of the algorithm. Larger α leads to more diversification and larger β leads to faster convergence and more intensification of the algorithm. V_i^k and V_i^{k+1} are velocity of *i*th particle in the *k*th and *k*+1th iterations. $rand_1$ and $rand_2$ are two random numbers. C^k is the center at iteration *k*. X_i^k is the *i*th solution particle in the iteration *k* and X_{best}^k is the best solution at *k*th iteration. And *r* is a value for preventing the algorithm to be trapped in a local minimum and is defined as follows

$$r = \begin{cases} +1 & rand \le P \\ -1 & rand > P \end{cases} \tag{5}$$

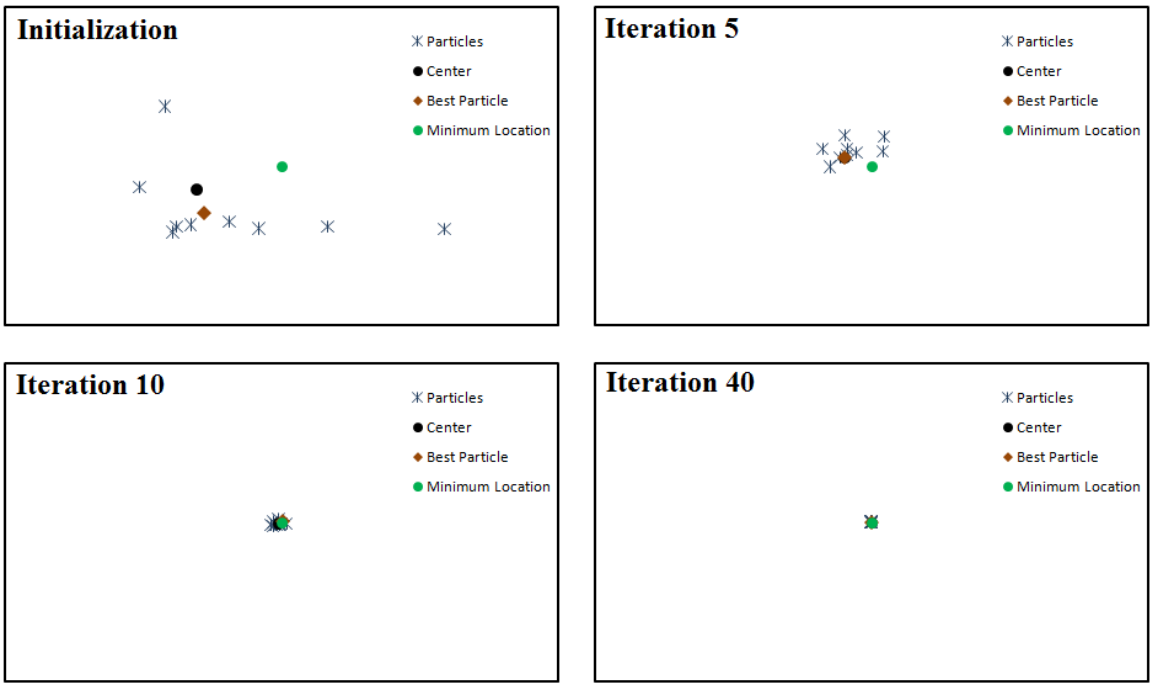

Fig. 1 Location of particles in different iterations of RPO

Where P is a predefined value in the [0,1] and should be near to 1. If r=−1 some particles recede from the ranked center and other parts of the search space are explored and this ensures that the particles are not trapped in a local minimum.

Eq. (4) is like velocity definition of the PSO but it uses the center of ranked particles instead of local best. This modification causes global search at the initial searches when particles are far apart, because center is far from the best particle; and in the latest searches when particles are near each other, it provides a local search around the best particle. Therefore, it makes a balance between exploration and exploitation. Fig. 1 shows the movement of particles in the RPO. In this figure it can be seen that the distance between ranked center and the best particle is high at initial stages and movement along center and the best particle provides global search and in the latest stages of algorithm progression, this distances get smaller and a local search is performed about the best particle and center.

There is a problem in relation to many meta-heuristic algorithms: how to deal with an agent violating the limits of the variables. In order to solve this problem, one of the simplest approaches is utilizing the nearest limit values for the violated variable. Alternatively, one can force the violating particle to return to its previous position, or one can reduce the maximum value of the velocity to allow fewer particles to violate the variable boundaries. Although these approaches are simple, they are not sufficiently efficient and may lead to reduce the exploration of the search space. This problem has previously been addressed and solved using the Harmony Search-based handling approach (Kaveh and Talatahari 2009a). According to this mechanism, every component of the solution vector violating the variable boundaries can be regenerated from the PM as:

A new harmony vector is improvised from the PM based on PMCR and PAR. With the probability of PMCR the new vector is generated from PM and with the probability of (1-PMCR) the new vector is generated randomly from possible ranges of values. The pitch adjusting process is performed only after a value is selected from PM. The value (1-PAR) sets the rate of doing nothing. A PAR of 0.25 indicates that the algorithm will select a neighboring value with 0.25×PMCR. It is recommended not to set PMCR as 1.0 because it is probable that the global minimum does not exist in PM. With regard to the foregoing statements, the search of PM is summarized in Eq. (6). In which the term "w.p." represents "with the probability.

$$x_{i,j} = \begin{cases} w.p.\, PMCR \rightarrow & \begin{cases} select\, a\, value\, for\, the\, variable\, from\, PM \\ w.p.\,(1-PAR)\; do\, nothing \\ w.p.\, PAR\; select\, a\, neighboring\; value \end{cases} \\ w.p.\,(1-PMCR) \rightarrow & select\, the\, new\, variable\, randomly \end{cases} \tag{6}$$

Regarding the above definitions, steps of the RPO can be expressed as follows:

2.2.1 Initialization of the optimization algorithm

Eq. (7) is utilized to randomly generate particles so to cover the entire design space and initial velocities are considered to be zero.

$$X = X_{\min} + rand \times \left(X_{\max} - X_{\min}\right) \tag{7}$$

2.2.2 Evaluate particles

The cost function is computed for each particle and PMS of particles are stored in the PM. Assign ranking to the particles stored in the PMS such that the best particle has a rank equal to PMS and the worst one has a rank equal to 1 and others have rank between 1 and PMS by unit steps. Also, the ranked center of stored particles is calculated using Eq. (2).

2.2.3 Update particles' positions

The position of each particle is updated with Eq. (3) based on the velocity and previous position of the particle. If new position of each particle is outside feasible search space, its position should be corrected using harmony search as presented in Eq. (6).

2.2.4 Update particles' velocities

The velocity of each particle is updated according to Eq. (4) based on the velocity and position of ranked center and the best particle in the PM.

2.2.5 Update memory

If Step 2.2.3 results in a better solution than particles stored in the PM, the new particle is included in the PM and the worst particle in the PM is excluded.

2.2.6 Stopping criterion

Repeat steps 2.2.2 to 2.2.4 until one of considered stopping criteria is met.

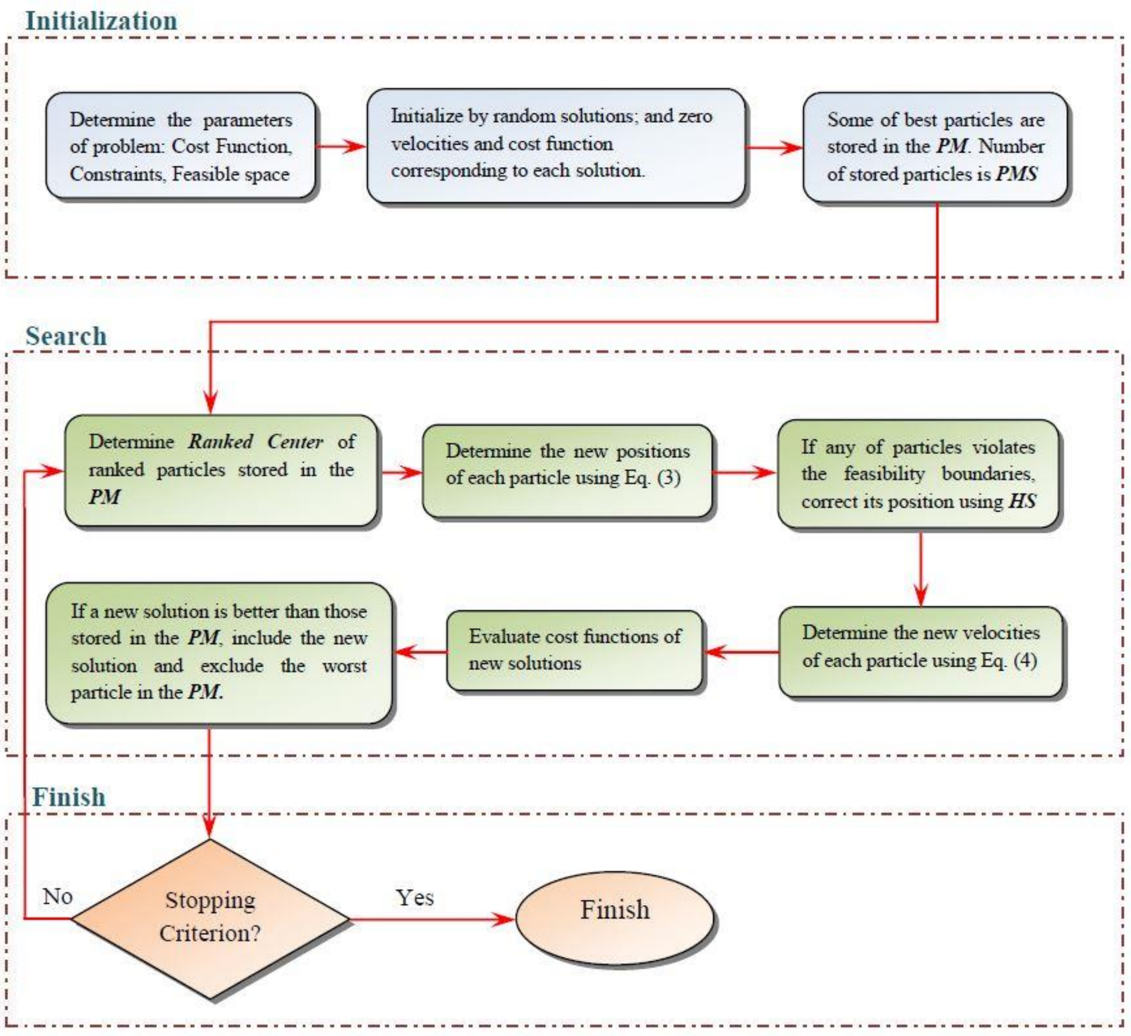
Initialization
Determine the parameters of problem: Cost Function, Constraints, Feasible space
Initialize by random solutions; and zero velocities and cost function corresponding to each solution.
Some of best particles are stored in the *PM*. Number of stored particles is *PMS*
Search
Determine *Ranked Center* of ranked particles stored in the *PM*
Determine the new positions of each particle using Eq. (3)
If any of particles violates the feasibility boundaries, correct its position using *HS*
Determine the new velocities of each particle using Eq. (4)
Evaluate cost functions of new solutions
If a new solution is better than those stored in the *PM*, include the new solution and exclude the worst particle in the *PM*.
Finish
No
Stopping Criterion?
Yes
Finish

Fig. 2 Flowchart of RPO

The abovementioned steps are shown in the flowchart of Fig. 2.

3. Verification of the ranked particles optimization

In order to verify the efficiency of the new algorithm, some numerical examples are considered from literature. The examples contain 14 uni-modal and multi-modal functions. These numerical examples are presented in Section 3.1. The performance of the RPO to optimize these functions is investigated in Section 3.2. In Section 3.3, some well-studied engineering design problems taken from the optimization literature are used to illustrate the way in which the proposed method works. Also, in this section, a new optimization problem is presented and optimal design of this problem is implemented using some well-known optimization techniques for comparison.

Table 1 Specifications of the benchmark problems

Function name	Interval	Function	Global minimum
Aluffi Pentiny	$X \in [-10,10]^2$	$f(X)=\frac{1}{4}x_1^4-\frac{1}{2}x_1^2+\frac{1}{10}x_1+\frac{1}{4}x_2^2$	-0.352386
Bohachevsky1	$X \in [-100,100]^2$	$f(X)=x_1^2+2x_2^2-\frac{3}{10}\cos(3\pi x_1)-\frac{4}{10}\cos(4\pi x_2)+\frac{7}{10}$	0.0
Bohachevsky2	$X \in [-50,50]^2$	$f(X)=x_1^2+2x_2^2-\frac{3}{10}\cos(3\pi x_1)\cos(4\pi x_2)+\frac{3}{10}$	0.0
Becker and Lago	$X \in [-10,10]^2$	$f(X)=(\|x_1\|-5)^2+(\|x_2\|-5)^2$	0.0
Branin	$x_1 \in [-5,10],\ x_2 \in [0,15]$	$f(X)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos(x_1)+10$	0.397887
Camel	$X \in [-5,5]^2$	$f(X)=4x_1^2-2x_1^4+\frac{1}{3}x_1^6+x_1x_2+x_2^2$	-1.0316
Cb3	$X \in [-5,5]^2$	$f(X)=2x_1^4-1.05x_1^4+\frac{1}{6}x_1+x_1x_2+x_2^2$	0.0
Cosine mixture	$n=4,\ X \in [-1,1]^n$	$f(X)=\sum_{i=1}^n x_i^2-\frac{1}{10}\sum_{i=1}^n \cos(5\pi x_i)$	-0.4
De Jong	$X \in [-5.12,5.12]^2$	$f(X)=x_1^2+x_2^2+x_3^2$	-1.0
Exponential	$n=2,4,8,\ X \in [-1,1]^n$	$f(X)=-\exp\left(-0.5\sum_{i=1}^n x_i^2\right)$	-1.0
Griewank	$X \in [-100,100]^2$	$f(X)=1+\frac{1}{200}\sum_{i=1}^n x_i^2-\prod_{i=1}^2 \cos\left(\frac{x_i}{\sqrt{i}}\right)$	0.0
Rastrigin	$X \in [-1,1]^2$	$f(X)=\sum_{i=1}^n \left(x_i^2-\cos(18x_i)\right)$	-2.0
Goldstein and Price	$X \in [-2,2]^2$	$f(X)=\left[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)\right] \times\left[30+(2x_1-3x_2)^2(18-32x_1-12x_1^2+48x_2-36x_1x_2+27x_2^2)\right]$	3.0

Similar to the other meta-heuristics, for the RPO, a large value for the number of particles increases the search strength of the algorithm as well as the computational cost, and vice versa a small number causes a quick convergence without performing a complete search.

Here, the number of Particles is set to 20 and the maximum number of the permitted iterations is considered as 100. For this problems, PMS and P are 5 and 0.95, respectively. α in mathematical problems is 0.5 and in structural problems is 1. β in both mathematical and structural problems, except for the fourth structural problem is 1. In the fourth structural problem, influence of different values of β on performance of the RPO is investigated. These values seem to be suitable for finding the optimum results. Also, the value of PMCR is set to 0.95 and that of PAR is taken as 0.10. Each problem is solved 50 times using RPO and the best result, mean of results, worst result, and standard deviation of obtained results from different runs are calculated.

3.1 Description of the mathematical examples

In this section, a number of benchmark functions chosen from Tsoulos (2008) are optimized using RPO and compared to GA, some of its variations, CSS and RO by Kaveh and Khayatazad (2012) to verify the efficiency of RPO. The description of these test problems is provided in Table 1. When the dimension is selected as 2, a perspective view and the related contour lines for some of these functions are illustrated in Fig. 3.

3.1.1 Results

The results obtained by RPO are listed in Table 2 along with those obtained by GA, some of its

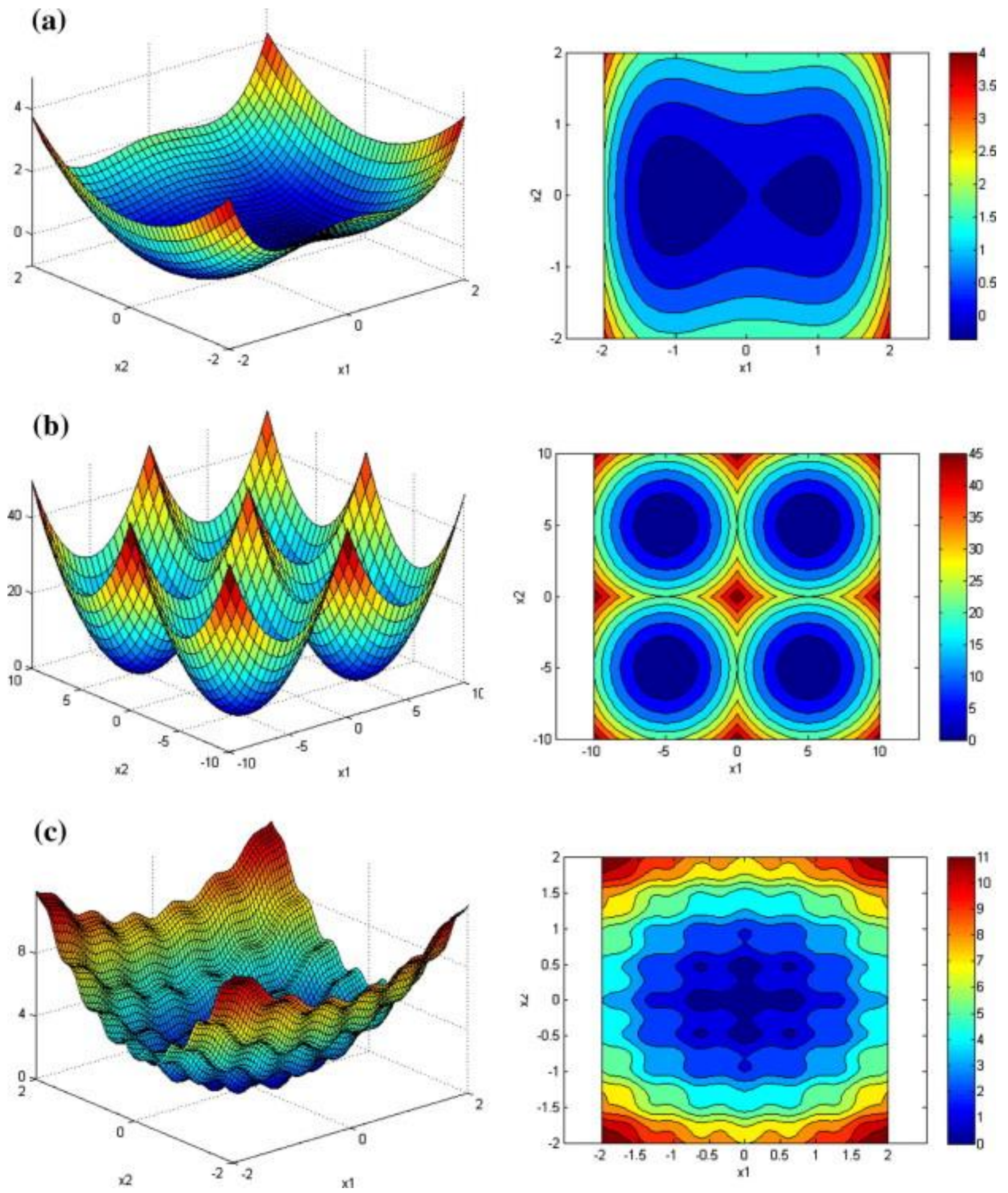

Fig. 3(a) A perspective view and the related contour lines for some of function when n=2 (a) Aluffi-Pentiny, (b) Becker and lago, (c) Bohachevsky 1

variations, CSS, and RO. The numbers denote the average number of function evaluations from 50 independent runs for every objective function described in Section 3.1. The numbers in parentheses represent the fraction of successful runs in which the algorithm has located the global minimum with predefined accuracy, which is taken as $\varepsilon=f_{\min}-f_{final}=10^{-4}$. The absence of the parentheses denotes that the algorithm has been successful in all independent runs. To sum up, comparison of the results demonstrates that RPO has a faster convergence than other considered algorithms.

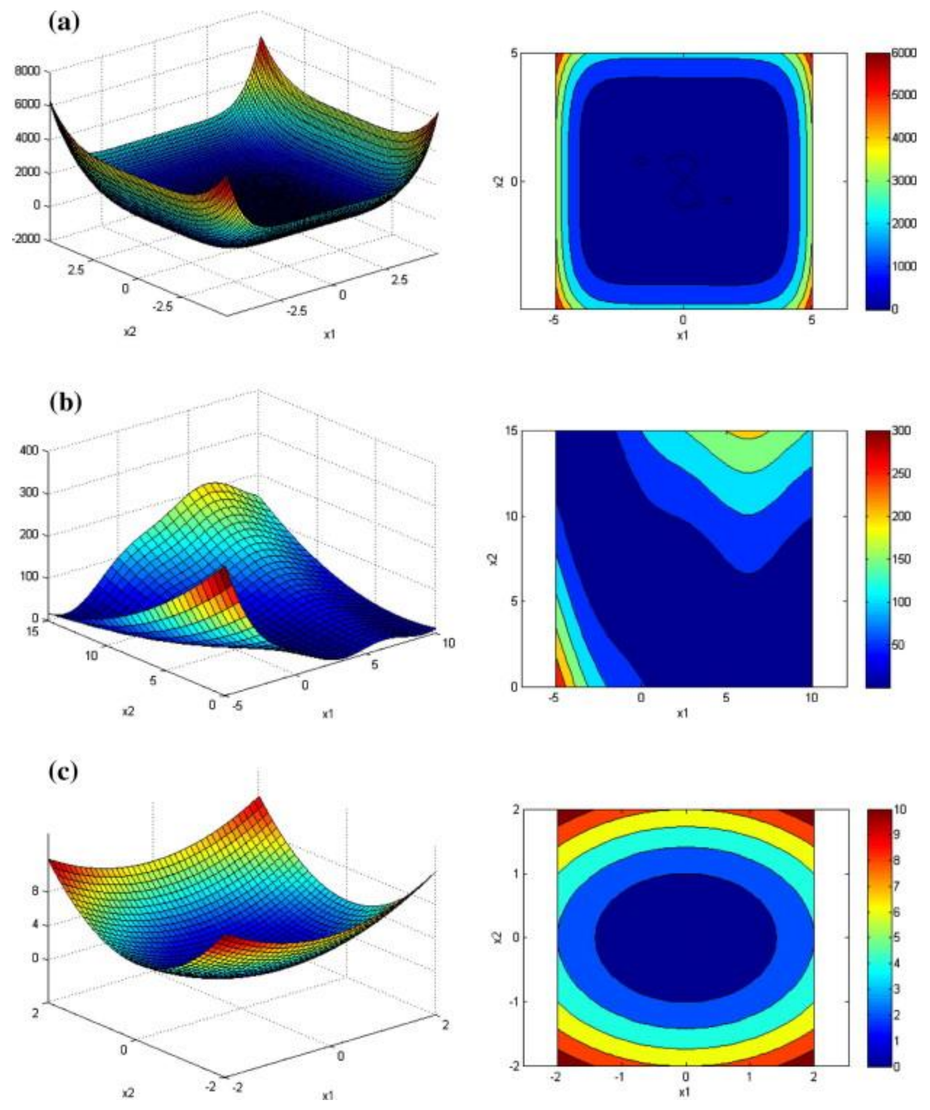

Fig. 3(b) A perspective view and the related contour lines for some of function when *n*=2, (a) Camel, (b) Branin, (c) Bohachevsky 2

Table 2 Performance comparison for the benchmark problems

Function	GEN	GEN_S	GEN_S_M_LS	CSS	RO	RPO
Aluffi Pentiny	1360 (0.99)	1360	1253	804	331	221
Bohachevsky 1	3992	3356	1615	1187	677	461
Bohachevsky 2	20234	3373	1636	742	582	468
Becker and Lago	19596	2412	1436	423	303	237
Branin	1442	1418	1257	852	463	221
Camel	1358	1358	1300	575	332	241
Cb3	9771	2045	1118	436	262	224

Table 2 Continued

Cosine mixture	2105	2105	1539	1563	802	249
De Jong	9900	3040	1281	630	452	283
Exponential n=2	938	936	807	132	136	110
Exponential n=4	3237	3237	1496	867	382	225
Exponential n=8	3237	3237	1496	1426	1287	451
Griewank	18838 (0.91)	3111 (0.91)	1652 (0.99)	1551	1091(0.98)	468
Rastrigin	1533 (0.97)	1523 (0.97)	1381	1402	1013(0.98)	323
Goldstein and Price	1478	1487	1325	682	451	283

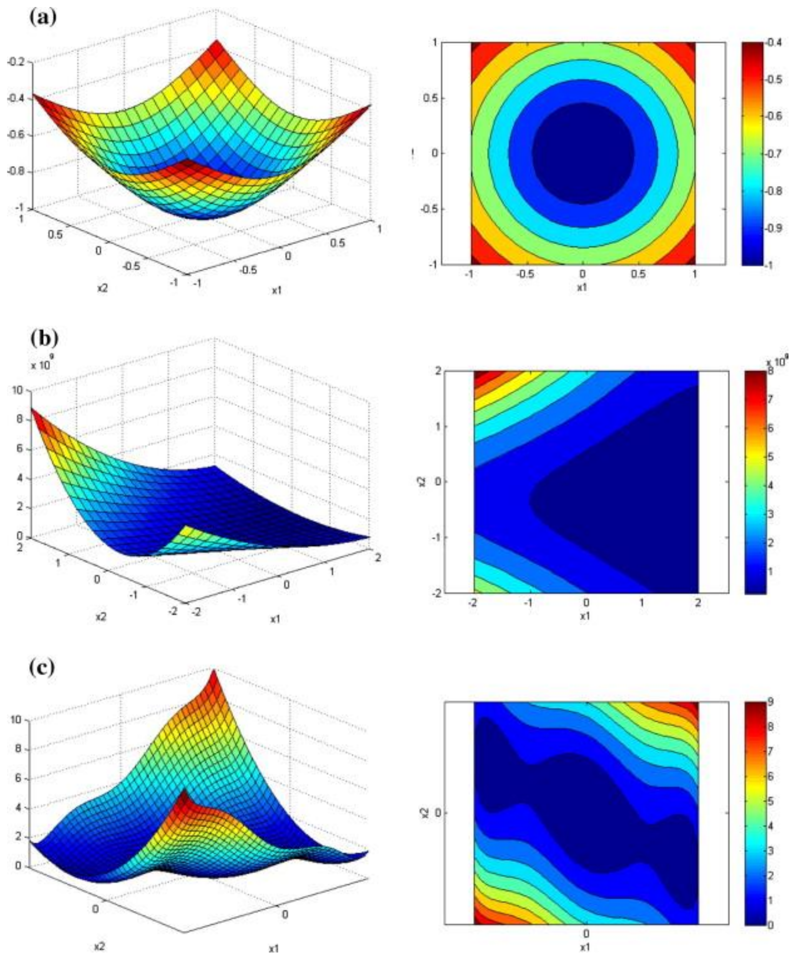

Fig. 3(C) A perspective view and the related contour lines for some of function when *n*=2 (a) Exponential, (b) Goldstein and price, (c) Cb3

3.2 Engineering design problems

Three engineering design problems which have been previously solved using a variety of other techniques are considered to show the validity and effectiveness of the proposed algorithm. Also, a

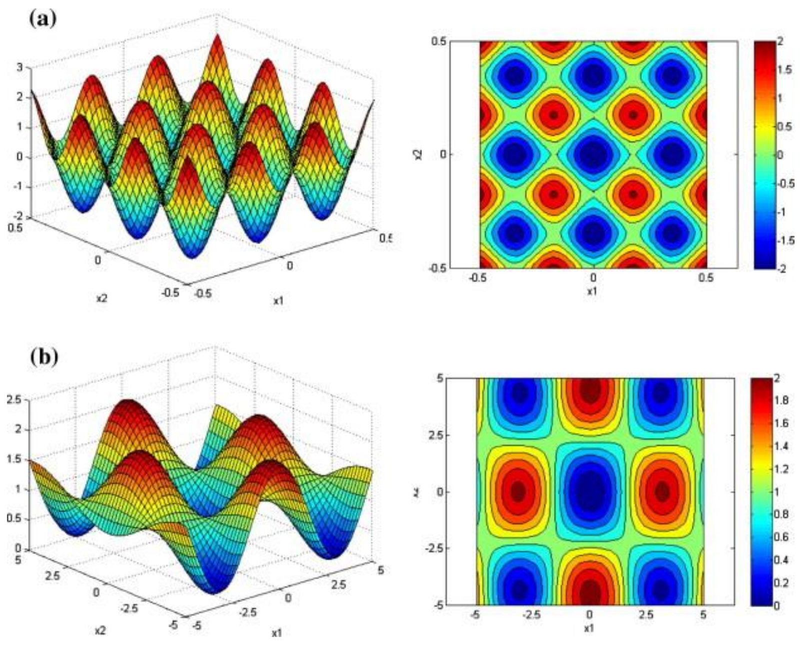

Fig. 3(D) A perspective view and the related contour lines for some of function when *n*=2 (a) Griewank, (b) Rastrigin

new structural optimization problem is introduced here and solved using some existing meta-heuristics and RPO to compare them with the new algorithm. In utilizing the penalty functions, if the constraints are between the allowable limits, the penalty will be zero; otherwise, the amount of penalty is equal to the normalized violation Kaveh and Talatahari (2009b).

The objective function considered for the examples is defined as follows

$$W(X)=\sum_{i=1}^{N}\rho.L_i.X_i \tag{8}$$

Where X is the vector of design variables, and in the examples of this study, it is the vector of cross sectional area of the members; ρ is the material density, and L_i is members length; and N is the number of elements.

To handle the constraints, a simple penalty approach is employed. The penalty function is defined as

$$P=\left(1+\lambda\sum_{i=1}^{M}C_i\right) \tag{9}$$

Where M is the number of constraints; λ is the Lagrange coefficients, and in this example it is considered to be 10; and C_i is *i*th constraint violation ratio, and it is defined as follows:

To control that the element stresses to be less than the allowable value, C_i is defined as

$$C_i=\begin{cases}\dfrac{\sigma_i}{\sigma_{all}}-1 & if \quad \sigma_i>\sigma_{all}\\ 0 & else\end{cases} \tag{10}$$

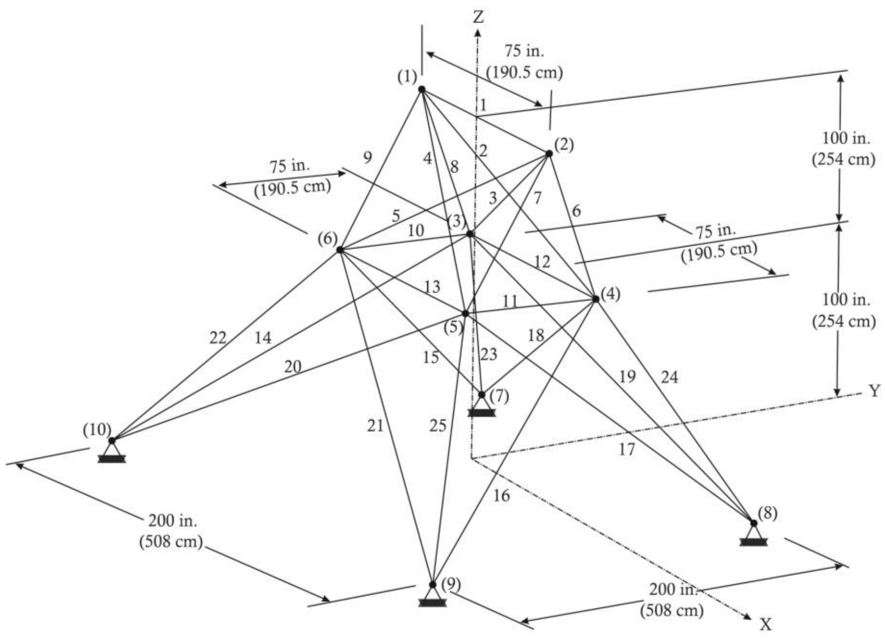

Fig. 4 Geometry and element numbering of the 25-bar element space truss

Table 3 Loading conditions for the 25-bar spatial truss

Node	Case 1			Case 2		
	P_x kips (kN)	P_y kips (kN)	P_z kips (kN)	P_x kips (kN)	P_y kips (kN)	P_z kips (kN)
1	0.0	20 (89)	-5.0 (-22.25)	1.0 (4.45)	10.0 (44.5)	-5.0 (-22.25)
2	0.0	-20 (-89)	-5.0 (-22.25)	0.0	10.0 (44.5)	-5.0 (-22.25)
3	0.0	0.0	0.0	0.5 (2.22)	0.0	0.0
4	0.0	0.0	0.0	0.5 (2.22)	0.0	0.0

Where *i* denotes the *i*th element; σ_i is the stress of the *i*th element; and σ_{all} is the allowable stress.

And to control the nodal displacements to be less than the allowable value, C_i is defined as

$$C_i = \begin{cases} \dfrac{\delta_i}{\delta_{all}} - 1 & if \quad \delta_i > \delta_{all} \\ 0 & else \end{cases} \tag{11}$$

Where *i* denotes the *i*th node; δ_i is the displacement of the *i*th node; and δ_{all} is the allowable displacement.

Finally, to minimize the weight of structure, and to ensure that the structure will provides stress and displacement requirements, the following function should be minimized:

$$\Phi(X) = W(X) \times P \tag{12}$$

Where $\Phi(X)$ is known as the penalized cost function.

3.2.1 A 25-bar spatial truss structure

A 25-bar spatial truss structure is considered with the topology and nodal numbering shown in Fig. 4. Table 3 shows two load cases for which the design is performed. In this example, the material density is considered as 0.1 lb/in^3 (2767.990 kg/m^3) and the modulus of elasticity is taken as 10,000 *ksi* (68,950 *MPa*). Twenty five bars are classified into eight groups, as follows:

(1) A1; (2) A2~A5; (3) A6~A9; (4) A10~A11; (5) A12~A13; (6) A14~A17; (7) A18~A21; and (8) A22~A25.

Maximum displacement limitations of ±0.35 in. (±8.89 mm) were imposed on every node in every direction and the axial stress constraints vary for each group shown in Table 4. The range of cross-sectional areas varies from 0.01 to 3.4 in^2 (from 0.06452 cm^2 to 21.94 cm^2).

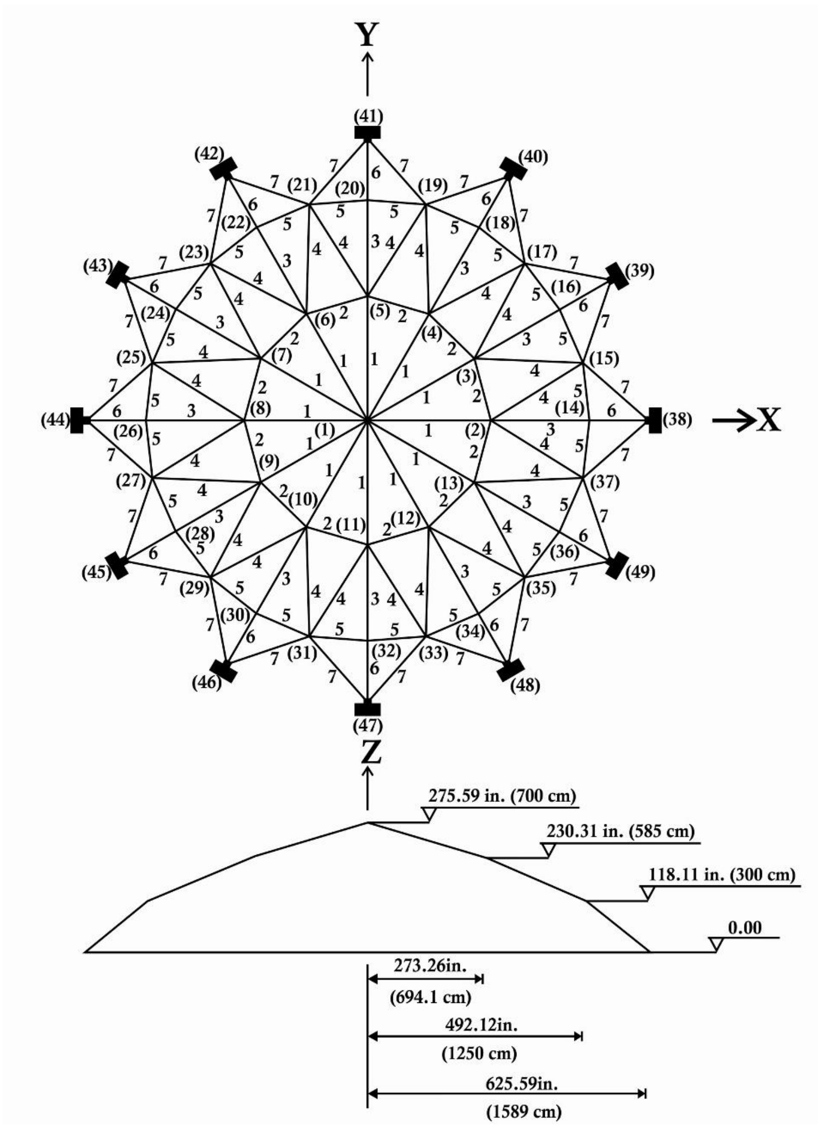

Fig. 5 Geometry and element grouping of the 120-bar element space truss dome

Table 5 Performance comparison for the 25-bar spatial truss

Author	Schutte and Groenwold (2003)	Kaveh and Talatahari (2009b)	Lee and Geem (2004)	Present Work
Group	PSO	HBB–BC	HS	RPO
1	0.01	0.01	0.01	0.010
2	2.121	1.993	2.121	2.024
3	2.893	3.008	2.893	2.917
4	0.01	0.01	0.01	0.010
5	0.01	0.01	0.01	0.010
6	0.671	0.679	0.671	0.677
7	1.611	1.611	1.611	1.684
8	2.717	2.678	2.171	2.691
Best Weight (lb)	545.21	545.16	545.21	545.132
Number of analyses	9,596	12,500	15,000	2,460

Table 5 provides some of solutions of this problem. From this table, it can be concluded that RPO has a solution slightly better than PSO, HBB-BC, and HS. Also, number of analyses, and consequently time and computational effort, in RPO implementation to reach the optimum solution is significantly less than other meta-heuristics presented in the table.

3.2.2 A 120-bar spatial truss dome

Design of a 120-bar spatial dome truss, shown in Fig. 5, is considered as the second example to compare the practical capability of the proposed algorithm. This dome is utilized in literature to find size optimum design. The modulus of elasticity is 30,450 ksi (210,000 MPa), and the material density is 0.288 lb/in^3 (7971.810 kg/m^3). The yield stress of steel is taken as 58.0 ksi (400 MPa). This dome is considered to be subjected to vertical loading at all the unsupported joints. These loads are taken as −13.49 kips (−60 kN) at node 1, −6.744 kips (−30 kN) at nodes 2 through 14, and −2.248 kips (−10 kN) at the remaining nodes. The minimum cross sectional area of all members is 0.775 in^2 (2 cm^2) and the maximum cross-sectional area is taken as 20.0 in^2 (129.03 cm^2). The stress constraints of the structural members are calculated as per AISC (1989) specifications as illustrated in Eq. (13). The 120 bar spatial truss members are categorized into 7 groups as shown in Fig. 5. For further optimal design of domes the reader may refer to Gholizadeh and Barati (2014)

$$\begin{cases} \sigma_i^+ = 0.6F_y & \text{for} \quad \sigma_i \geq 0 \\ \sigma_i^- & \text{for} \quad \sigma_i < 0 \end{cases} \tag{13}$$

Where, σ_i^- is calculated according to the slenderness ratio using

$$\sigma_i^- = \begin{cases} \left[\left(1-\frac{\lambda_i^2}{2C_c^2}\right)F_y\right] \Big/ \left(\frac{5}{3}+\frac{3\lambda_i}{8C_c}-\frac{\lambda_i^3}{8C_c^3}\right) & \text{for} \quad \lambda_i < C_c \\ \frac{12\pi^2 E}{23\lambda_i^2} & \text{for} \quad \lambda_i \geq C_c \end{cases} \tag{14}$$

Table 6 Performance comparison for the 120-bar spatial truss dome

Element group	Keleşoğlu and Ü lker (2005)	Kaveh and Talatahari (2009a)	Kaveh and Talatahari (2010b)	Kaveh and Nasrollahi (2013)	RPO (Present work)
1	5.606	3.095	3.027	3.037	3.01372
2	7.75	14.405	14.606	3.867	3.938936
3	4.311	5.020	5.044	3.241	3.237504
4	5.424	3.352	3.139	2.246	2.236409
5	4.402	8.631	8.543	1.637	1.606373
6	6.223	3.432	3.367	2.492	2.481978
7	5.405	2.499	2.497	2.301	2.301893
Best Weight (lb)	38237.83	33248.9	33251.9	18292.8	18251.58
Mean weight (lb)	N/A	N/A	N/A	18377.6	18324.88
Worst weight (lb)	N/A	N/A	N/A	18489.5	18490.51
Standard deviation	N/A	N/A	N/A	176.525	70.22976

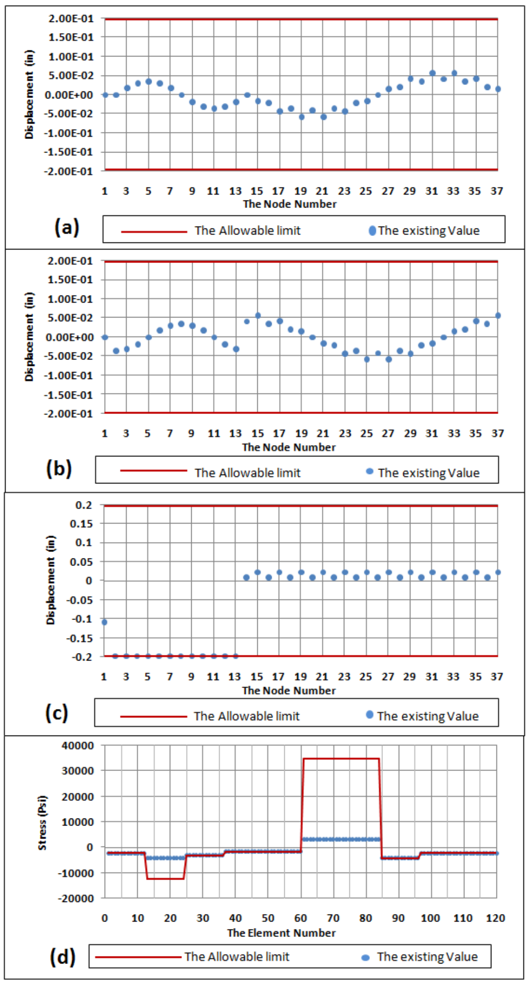

Fig. 6 Nodal displacement and stress of elements of designed 120-bar truss dome

Where, E is the modulus of elasticity; F_y is the yield strength of steel; C_c is the slenderness ratio which divides the elastic and inelastic buckling regions $\left(C_c = \sqrt{2\pi^2 E / F_y}\right)$; and λ_i is the slenderness ratio. The relation between cross sectional area and radius of gyration for Pipe sections is as follows

$$r = c_1 A^{c_2} \tag{15}$$

Where, r is the radius of gyration and A is the cross sectional area; c_1 and c_2 are constants which for pipe sections are 0.4993 and 0.6777, respectively. The displacement constraint for this example is 0.1969 *in* in every direction.

The optimization results reported in literature and RPO are presented in Table 6. From this table it can be concluded that RPO which results is better in value and standard deviation, performs efficiently. To prove that none of constraints is violated, Fig. 6 is presented. From this figure, it is seen that displacement of each node and stress of each element is in the allowable bounds.

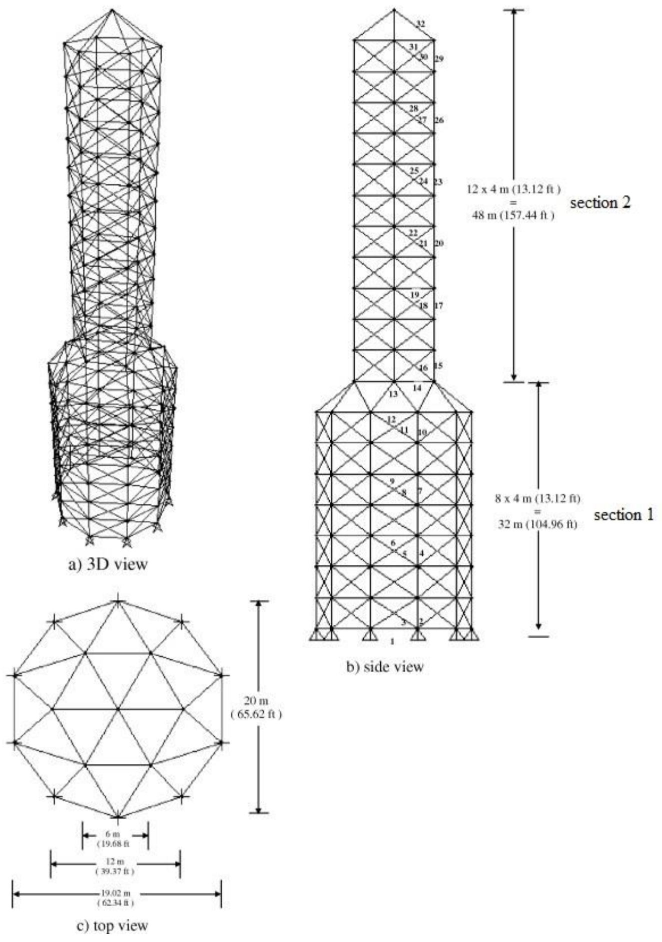

Fig. 7 Geometry and element grouping of the 582-bar spatial truss

3.2.3 A 582-bar spatial truss tower

Geometry and 32 element grouping of a 582-bar spatial truss tower shown in Fig. 7 is the same as works of Hasancebi *et al*. (2009). However, since, in this study, application of RPO is considered only in problems with continuous search space, it is changed and solved using different meta-heuristic. Also, influence of parameter β in the Eq. (4) is investigated in this section. This parameter controls the convergence and distinguishes exploration and exploitation of the algorithm. A large value of β leads to a fast convergence and more exploitation and may result in an immature solution. On the other hand, small values of β cause more exploration and heighten the probability of finding a better solution. However, a small value of β reduces the convergence of the algorithm and more analyses are needed for solving the problem and large value of β may lead to an immature solution. Therefore, this problem is solved using different values of β to verify the mentioned claim.

In this problem, modulus of elasticity and yield stress are E=29,000 ksi (203893.6 MPa) and F_y=36 ksi (253.1 MPa), respectively. The material density is 0.288 lb/in^3 (7971.810 kg/m^3). The stress constraints of the structural members are calculated in according with AISC (1989) specifications as illustrated in Eq. (8). The minimum and maximum cross sectional area are 0.775 in^2 and 28.5 in^2.

The imposed loads on the structure are as follows: A single load case is considered such that it consists of lateral loads of 1.0 kips (4.448 kN) applied in both x- and y-directions and a vertical load of -7.5 kips (33.362 kN) applied in the z-direction at nodes of section 1. Lateral loads of 1.0 kips (4.448 kN) applied in both x- and y-directions and a vertical load of -5 kips (22.241 kN) applied in the z-direction at nodes of section 2 of the tower. The maximum allowable displacement of each node is considered 12.5984 in (32 cm) which is 1/250 times the height of the tower.

Table 7 presents the solution of this problem using different meta-heuristics and RPO with different values of β. It is seen that the best result is obtained when it is solved using RPO with β=1/4.

Fig. 8 shows the convergence history of RPO with different values of β. As seen in this figure convergence rate of algorithm increases when the value of β is larger. Therefore, parameter β can be used to control the convergence rate of the algorithm to obtain a better solution. However, there is not a certain method for determining this parameter and for a new problem, it should be set by a try-and-error and can be investigated the influence of this parameter in the following studies.

Table 7 Performance comparison for the 582-bar spatial truss

Group				RPO			
	CSS	PSO	HS	β=1	β=1/2	β=1/3	β=1/4
1	2.23	3.57	1.79	1.66	1.70	1.62	1.93
2	11.82	22.50	15.32	17.78	17.04	17.79	13.16
3	6.38	7.99	6.03	6.56	5.92	5.89	6.25
4	11.17	15.91	10.15	10.67	11.89	10.96	11.64
5	6.21	12.55	7.02	5.44	5.76	5.50	5.39
6	2.48	18.34	3.03	1.53	1.43	2.88	1.44
7	8.75	17.17	8.55	8.73	8.54	8.67	8.56
8	5.26	8.83	5.50	5.21	5.04	5.09	5.00
9	1.72	26.19	2.79	1.96	1.79	2.07	1.55

Table 7 Continued

10	8.93	23.96	10.72	8.89	9.73	8.49	7.61
11	4.38	15.57	4.52	9.99	4.04	4.43	4.20
12	12.56	6.16	10.63	9.50	9.01	12.32	9.00
13	11.39	20.16	15.36	12.53	11.01	11.17	11.26
14	10.27	22.89	11.71	9.07	10.66	9.42	9.65
15	15.68	27.57	15.98	8.28	10.62	11.37	18.65
16	7.01	6.40	7.62	8.31	7.62	7.17	6.84
17	16.99	8.06	8.13	11.59	7.96	22.21	7.92
18	5.30	14.83	5.85	5.34	4.95	4.72	5.13
19	1.28	2.18	2.16	1.78	1.52	7.12	1.57
20	6.04	11.03	10.51	24.40	11.19	19.33	8.52
21	5.17	6.33	4.69	4.27	4.61	4.40	4.62
22	5.36	1.81	1.50	0.78	0.78	0.78	1.47
23	14.58	6.98	10.86	4.37	25.96	4.39	5.26
24	3.82	5.35	4.58	5.06	3.96	3.73	4.89
25	2.93	1.33	0.80	0.78	0.78	0.78	0.78
26	7.59	27.80	5.14	5.67	3.35	17.31	5.80
27	3.39	6.27	3.48	3.44	3.08	3.08	3.12
28	1.94	5.00	4.08	1.74	0.78	5.85	0.78
29	2.35	27.98	18.67	7.54	6.62	8.60	14.06
30	2.46	11.08	2.26	4.35	1.85	2.55	1.81
31	1.14	11.94	1.25	16.42	1.95	1.33	0.78
32	10.57	25.74	4.59	2.02	1.98	12.49	2.73
Best Weight (lb)	245215.19	477650.13	251511.86	252780.24	229974.06	253569.51	220704.11
No. of analyses	5520	7140	7969	8920	9660	8480	6680

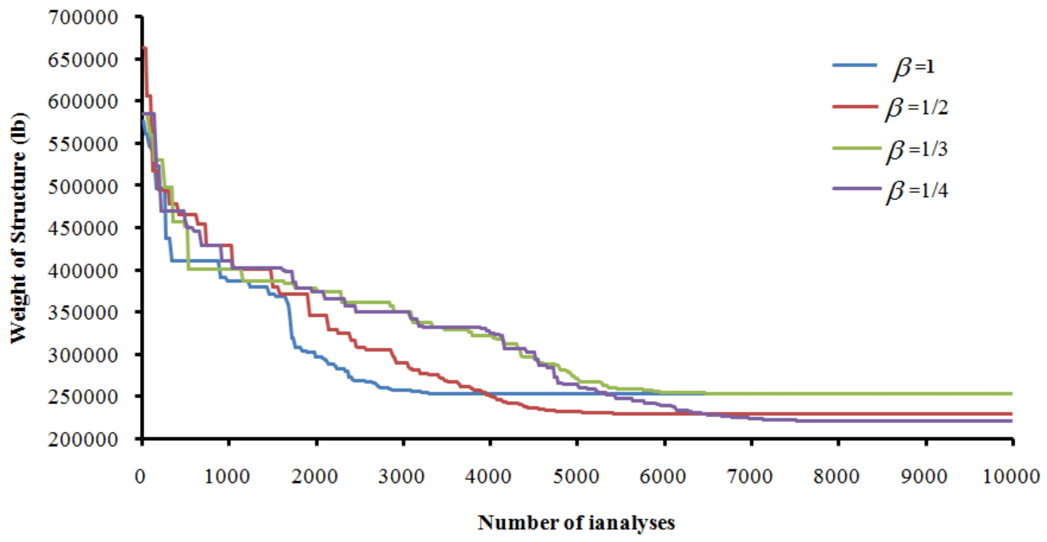

Fig. 8 Convergence history of the 582-bar spatial truss problem using different values for β

4. Conclusions

In this paper, a new meta-heuristic optimization algorithm namely *Ranked Particle Optimization, (RPO)* is presented. This algorithm is inspired from numerous works in the field of meta-heuristic algorithms. Similar to other meta-heuristics, RPO initializes by random solutions named *Particles* and evaluating the cost function for each random solution. Some of the best particles (*PMS*) are *Ranked* and stored in a memory (*Particle Memory, PM*). *Ranked Center* of stored particles is calculated and new solutions are determined by moving existing particles in the direction of previous *Velocity*, ranked center, and the *Best Particle*.

Moving towards previous velocity of particles provides exploration and moving in the direction of the ranked center at initial stages is a global search an in the latest iterations is a local search, and moving towards the best particle provides exploitation. In one hand, ranked particles intensifies the influence of better particles, on the other hand prevents the algorithm to be trapped in a local minima. To make a balance between exploration and exploitation, most of the meta-heuristics use a linear varying formula for velocity of displacement of particles to convert from global search phase to local search. In this study, a multiplier α and a power β is introduced to the velocity of the particles as well as linear decreasing inertia weight of PSO which is introduced here, too. Larger α and β leads to faster convergence and higher probability of trapping in a local minima. On the contrary, smaller value of these parameters increases the precision of the solutions but reduces its convergence rate. One example was solved using different values of β and it is concluded that we can reach to better solutions when a more suitable β is introduced to the algorithm.

The main source of inspiration of RPO is PSO and BB-BC, but there are some substantial differences between RPO and these algorithms. First, movement in the direction of particles' local best is replaced by a movement in the direction of ranked center. This idea is based in the BB-BC, and since center of best particles is always nearer to the global optimum than particles' local best, it enhances the robustness of the algorithm.

Second, in the RPO, a ranked center is used based on the rank selection process in the GA. But in BB-BC the mass center is calculated using inverse of the cost functions which intensifies the algorithm if a few particles have cost functions, and thus the particles may be trapped in local minima.

Third, there is a robust method for handling the violated particles from feasible search space in RPO, and it is HS. This approach keeps progression of searching when some particles violates boundaries; because way of correction of positions of violated particles is also a meta-heuristic while in the PSO and BB-BC, the lack of such approach, cause a halt in the search of the algorithm.

This new algorithm is applied to various mathematical and structural optimization problems and a good performance was observed in finding global optima with analyses. Finally, ease of implementation is another advantage of RPO.

Acknowledgements

The first author is grateful to the Iran National Science Foundation for the support.

References

American Institute of Steel Construction (AISC) (1989), *Manual of steel construction-allowable stress design*, 9th Edition, American Institute of Steel Construction, Chicago

De Jong, K. (1975), "Analysis of the behavior of a class of genetic adaptive systems", Ph.D. Thesis, University of Michigan, Ann Arbor, MI.

Dorigo, M., Maniezzo, V. and Colorni, A. (1996), "The ant system: optimization by a colony of cooperating agents", *IEEE Tran. Syst. Man Cyber.* **26**(1), 29-41.

Eberhart, R.C. and Kennedy, J. (1995), "A new optimizer using particle swarm theory", *Proceedings of the sixth international symposium on micro machine and human science*, Nagoya, Japan.

Erol, O.K. and Eksin, I. (2006), "New optimization method: big bang-big crunch" *Adv. Eng. Softw*., **37**, 106-111.

Fogel, L.J., Owens, A.J. and Walsh, M.J. (1966), *Artificial Intelligence Through Simulated Evolution*, Wiley, Chichester.

Geem, Z.W., Kim, J.H. and Loganathan, G.V. (2001), "A new heuristic optimization algorithm: harmony search" *Simulation* **76**(2), 60-68.

Goldberg, D.E. (1989), *Genetic Algorithms in Search Optimization and Machine Learning*, Addison-Wesley, Boston.

Gholizadeh, S. and Barati, H. (2014) "Topology optimization of nonlinear single layer domes by a new metaheuristic", *Steel Compos. Struct*, **16**(6), 681-701.

Hasancebi, O., Carbas, S., Dogan, E., Erdal, F. and Saka, M.P. (2009), "Performance evaluation of metaheuristic search techniques in the optimum design of real size pin jointed structures", *Comput. Struct.*, **87**, 284-302.

Holland, J.H. (1975), *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor.

Kaveh, A. and Khayatazad, M. (2012), "A new meta-heuristic method: Ray Optimization", *Comput. Struct.*, **112-113**, 283-294.

Kaveh, A. and Mahdavai, VR. (2014), "Colliding bodies optimization: a novel meta-heuristic method", *Comput. Struct.*, **139**, 18-27.

Kaveh, A. and Nasrollahi, A. (2013), "Engineering design optimization using a hybrid PSO and HS algorithm", *Asian J. Civil Eng*., **14**(2), 201-223.

Kaveh, A. and Talatahari, S. (2009a), "Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures", *Comput. Struct.*, **87**(5-6), 267-283.

Kaveh, A. and Talatahari, S. (2009b), "Size optimization of space trusses using big bang-big crunch algorithm", *Comput. Struct*., **87**(17-18), 1129-1140.

Kaveh, A. and Talatahari, S. (2010a), "A novel heuristic optimization method: charged system search", *Acta Mech*., **213**(3-4), 267-289.

Kaveh, A. and Talatahari, S. (2010b), "Optimal design of skeletal structures via the charged system search algorithm", *Struct. Multidiscip. Optim.*, **41**(6), 893-911.

Kaveh, A., Motie Share, M.A. and Moslehi, M. (2013), "A new meta-heuristic algorithm for optimization: magnetic charged system search", *Acta Mech.*, **224**(1), 85-107

Keleşoğlu, O. and Ülker, M. (2005), "Fuzzy optimization geometrical nonlinear space truss design", *Turk. J. Eng. Environ. Sci*., **29**, 321-329.

Kirkpatrick, S., Gelatt, C. and Vecchi, M. (1983), "Optimization by simulated annealing", *Sci*., **220**(4598), 671-680.

Koza, J.R. (1990), "Genetic programming: a paradigm for genetically breeding populations of computer programs to solve problems", Report No. STAN-CS-90-1314, Stanford University, Stanford, CA.

Lee, K.S. and Geem, Z.W. (2004), "A new structural optimization method based on the harmony search algorithm", *Comput. Struct*., **82**, 781-798.

Rashedi, E., Nezamabadi-pour, H. and Saryazdi, S. (2009), "GSA: a gravitational search algorithm", *Inf.*

Sci., **179**, 2232-2248.

Schutte, J.J. and Groenwold, A.A. (2003), “Sizing design of truss structures using particle swarms”, *Struct. Multidiscip. Optim*., **25**, 261-269.

Tsoulos, I.G. (2008), “Modifications of real code genetic algorithm for global optimization”, *Appl. Math. Comput*., **203**, 598-607.