Journal of Computational Design and Engineering 6 (2019) 49–59

Contents lists available at ScienceDirect

Journal of Computational Design and Engineering

journal homepage: www.elsevier.com/locate/jcde

Multi-objective colliding bodies optimization algorithm for design of trusses

Ali Kaveh [a,*], Vahid Reza Mahdavi [b]

[a] Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Tehran, P.O. Box 16846-13114, Iran
[b] Department of Civil Engineering, Iran University of Science and Technology, Narmak, Tehran, P.O. Box 16846-13114, Iran

ARTICLE INFO

Article history:
Received 6 September 2017
Received in revised form 29 March 2018
Accepted 8 April 2018
Available online 11 April 2018

Keywords:
Design optimization
Multiobjective algorithm
Truss structural optimization
Colliding bodies optimization algorithm
Maximin method

ABSTRACT

This article presents a new population-based optimization algorithm to solve the multi-objective optimization problems of truss structures. This method is based on the recently developed single-solution algorithm proposed by the present authors, so called colliding bodies optimization (CBO), with each agent solution being considered as an object or body with mass. In the proposed multi-objective colliding bodies optimization (MOCBO) algorithm, the collision theory strategy as the search process is utilized and the Maximin fitness procedure is incorporated to the CBO for sorting the agents. A series of well-known test functions with different characteristics and number of objective functions are studied. In order to measure the accuracy and efficiency of the proposed algorithm, its results are compared to those of the previous methods available in the literature, such as SPEA2, NSGA-II and MOPSO algorithms. Thereafter, two truss structural examples considering bi-objective functions are optimized. The performance of the proposed algorithm is more accurate and requires a lower computational cost than the other considered algorithms. In addition, the present methodology uses simple formulation and does not require internal parameter tuning.

1. Introduction

In the real-world structural design, engineers face with the problem of selecting their design from a set of alternatives that fits better to their goals. In order to model these problems, engineers usually need to select their preference over a set of alternatives and construct preference relation judgment matrices using their expressed pair wise comparison information. Multi-objective optimization (MOO) is one such a technique which aims at extracting a set of multiple and conflicting solutions, called the Pareto-optimal set, in place of a single optimal solution (Coello & Lechuga, 2002; Coello, Van Veldhuizen, & Lamont, 2002).

This class of optimization problems has a rather different perspective in comparison to single-objective optimization (SOO) problems. In the past years, a vast number of methods have been developed to solve various SOO problems (Deb, 1999; Kaveh, 2017; Kaveh, Shojaei, Golipour, & Rahami, 2013; Guo, Wang, & Wu, 2016; Hassanzadeh & Rouhani, 2010; Nigdeli, Bekdaş, Kim, & Geem, 2015; Bilela, Mohameda, Zouhaiera, & Lotfib, 2016; Srinivas & Deb, 1994; Yi, Li, & Zhang, 2015 and Yi, Wen, & Li, 2016). These methods are generally divided into two groups: mathematical programming methods and population-based algorithms (Miettinen, 1999). The first group consists of methods like weighted sum and the ε-constraint methods (Haimes, Lasdon, & Wismer, 1971; Kim & de Weck, 2006; Mousa, El-Shorbagy, & Abd-El-Wahed, 2012; Zadeh, 1963), which are based on converting the MOO problem to an SOO problem. On the other hand, the population-based MOO algorithms are suitable for global search because of their capability of exploring and finding Pareto-optimal sets in the search space in a single simulation run (Hosseini & Al Khaled, 2014). Over the recent decades, many nature-inspired MOO algorithms have been proposed for solving numerical optimization problems (Chao, Shengqiang, Xinyu, & Liang, 2016; Deb, Pratap, Agarwal, & Meyarivan, 2002; Hu, Rong, Liang-Lin, & Li-xian, 2011; Ko & Wang, 2011). In these algorithms, some strategies are applied to the formulation of single-objective algorithms for searching the approximate Pareto-optimal sets. The unique purposes of these algorithms are maintaining good spread among the solutions and convergence to the true Pareto-front. For example, in the NSGA-II algorithm (Deb et al., 2002), the non-dominated sorting approach and a selection operator are incorporated to the Genetic algorithm. Many PSO based algorithms have been utilized for the solution of the MOO problems in which

Peer review under responsibility of Society for Computational Design and Engineering.

* Corresponding author.
E-mail address: alikaveh@iust.ac.ir (A. Kaveh).

https://doi.org/10.1016/j.jcde.2018.04.001
2288-4300/

different strategies are chosen for selecting the leader populations in an external archive of non-dominated solutions (Clarke & McLeskey, 2015; Hu et al., 2011).

Colliding bodies optimization (CBO) is a population-based evolutionary algorithm which uses an analogy of the laws of collision between objects (Kaveh & Mahdavi, 2014). CBO has obtained good results for many constrained and unconstrained benchmark functions and engineering single-objective problems (Kaveh & Mahdavi, 2015). The formulation of this algorithm is simple and uses no memory requiring no parameter tuning. Kaveh and Ardalani (2016) recently proposed a simplified multi-objective CBO technique based on non-dominated sorting method for optimizing the construction material costs of reinforced concrete structural elements and carbon dioxide.

The application of multi-objective optimization algorithms to structural optimization problems has engrossed the interest of many researchers (Angelo, Bernardino, & Barbosa, 2015; Fragiadakis, Lagaros, & Papadrakakis, 2006; Ho-Huu, Duong-Gia, Vo-Duy, Le-Duc, & Nguyen-Thoi, 2018; Ho-Huu, Hartjes, Visser, & Curran, 2018; Kaveh & Laknejadi, 2011, 2013; Kaveh, Laknejadi, & Alinejad, 2012; Kaveh & Massoudi, 2014; Liu, Burns, & Wen, 2005; Liu, Paulino, & Gardoni, 2016; Luh & Chueh, 2004; Mathakari, Gardoni, Agarwal, & Raich, 2007). In a vast majority of truss structural design applications, the fitness function was based on a single evaluation criterion. The goal of these problems is minimizing a given criterion, such as weight, subjected to some constraints, such as stresses, displacements and dynamic constraints. A practical optimization problem often contains more than one objective to be optimized, such as minimum weight or cost, maximum stiffness, minimum displacement at specific structural points, maximum structural strain energy, and maximum natural frequency of free vibration (Kaveh & Laknejadi, 2013). These encourage the researchers to formulate a multi-objective optimization problem to look for a set of compromise solutions in the search space.

In this article, we introduce a new multi-objective evolutionary optimization algorithm, MOCBO, based on the maximin fitness function to achieve the diversity and convergence of the solutions in the objective function space. The maximin strategy is used for ranking the populations, and then mating mechanism of CBO algorithm is performed to generate the new populations. The MOCBO is first validated using standard test functions taken from the specialized literature, having two and three objective functions. The MOCBO is compared with respect to the well-known optimization methods that are based on Genetic algorithm and particle swarm optimization. Then, MOCBO and MOPSO are utilized for the simultaneous weight and displacement optimization of truss structures.

This paper is presented in five sections: Section 2 describes the concepts of MOO problems, Maximum fitness function method, and the CBO algorithm. Then the implementation of the proposed algorithm is outlined in Section 3, followed by a section discussing the results of the developed method. The final section concludes the study.

2. Preliminaries

2.1. Basic definitions for MOO

As mentioned before, a MOO problem deals with the task of optimization with a set of conflicting objectives simultaneously (Mirjalili, Saremi, Mirjalili, & Coelho, 2016). The brief fundamentals of these formulations are as follows.

Definition 1 (*General Multi-Objective Optimization Problem : From M objectives, a set of tradeoff solutions can formally be shown as follows:*).

$$\begin{aligned} &\text{Find } X = [x_1, x_2, x_3, .., x_n] \\ &\text{to minimize } F(X) = \{f_1(X), f_2(X), \ldots, f_M(X)\} \\ &\text{subjected to } g_j(X) \leqslant 0, j = 1, 2, \ldots, m \\ &x_{imin} \leqslant x_i \leqslant x_{imax} \end{aligned} \tag{1}$$

where X denotes the vector of variables with n unknowns and g_j denotes the jth constraint from m inequality constraints; $f_i(X)$ represents the ith objective function. Also, x_{imin} and x_{imax} are the lower and upper bounds of design variable vector, respectively. The MOO problems involve optimizing some objectives where decreasing one objective may leads to an increase of the other objective. Thus, the main goal of the multi-objective optimization algorithms is to find acceptable vectors such that all the objective values become as small as possible.

Definition 2 (*Pareto Dominance*). (Deb, 1999): Vector $\mathbf{u} = (u_1, u_2, \ldots, u_M)$ dominates vector $\mathbf{v} = (v_1, v_2, \ldots, v_M)$, denoted by $\mathbf{u} \prec \mathbf{v}$, if and only if $\mathbf{u}$ is partially less than $\mathbf{v}$, i.e., $\forall i \in \{1, 2, \ldots, M\}, u_i \leqslant v_i \wedge \exists i \in \{1, 2, \ldots, M\}, u_i < v_i$.

Definition 3 (*Pareto Optimal*). (Deb, 1999): Solution $x \in \Omega$ is Pareto Optimal with respect to Ω if there is no $x\prime \in \Omega$ for which $\mathbf{v} = \{f_1(x'), f_2(x'), \ldots, f_M(x')\}$ dominates $\mathbf{u} = \{f_1(x), f_2(x), \ldots, f_M(x)\}$.

Also, a solution is called *Pareto Optimal* (non-dominated solution) if there does not exist any solution in the search space that dominates it. The set of all Pareto-optimal solutions in the decision variable space is collectively known as the Pareto optimal set (PS), and these corresponding set in the objective space is the Pareto-optimal Front (PF) (Coello & Lechuga, 2002; Srinivas & Deb, 1994).

2.2. The standard CBO algorithm

The CBO algorithm is a metaheuristic optimization algorithm inspired by the momentum and energy conservation laws of one dimensional collision (Kaveh & Mahdavi, 2014). In this algorithm, a number of Colliding Bodies (CBs) are considered, where each body is treated as an object with specified mass and velocity. The collisions occur among pairs of objects, and the new positions of the colliding bodies are updated based on the collision laws. CBO starts with a set of agents which are determined with random initialization of a population of individuals in the search space. Then, CBs are sorted in an ascending order based on the value of their cost function. The sorted CBs are divided equally into two groups. The first group consists of stationary CBs and the velocity of these agents before collision is considered as zero. The second group contains the moving agents that move toward the CBs of the first group. A brief description of the original CBO algorithm is as follows:

The changes of the body position represent the velocity of the CBs before collision as:

$$v_i = \begin{cases} 0, & i = 1, \ldots, n \\ x_i - x_{i-n}, & i = n+1, \ldots, 2n \end{cases} \tag{2}$$

where v_i and x_i are the velocity and position vectors of the ith CB, respectively. $2n$ is the number of population size. The velocity of the ith pair of CBs after the collision becomes:

$$v_i' = \begin{cases} \frac{(m_{i+n} + \varepsilon m_{i+n}) v_{i+n}}{m_i + m_{i+n}}, & i = 1, \ldots, n \\ \frac{(m_i - \varepsilon m_{i-n}) v_i}{m_i + m_{i-n}} & i = n+1, \ldots, 2n \end{cases} \tag{3}$$

where v_i and v'_i are the velocities of the ith CB before and after the collision, respectively; m_i is the mass of the ith CB defined as:

$$m_k = \frac{\frac{1}{fit(k)}}{\sum_{i=1}^{n} \frac{1}{fit(i)}} \quad k = 1, 2, \ldots, 2n \tag{4}$$

Here, *fit*(*i*) represents the value of the objective function for the *i*th agent. Naturally, a CB with good value exerts a larger mass and fewer moves than the bad ones. Here, the ε is the coefficient of restitution (COR) which is defined as the ratio of the separation velocity of two agents after collision to approach velocity of two agents before collision. In this algorithm, this index is defined to control of the exploration and exploitation rates. In order to achieve this, the COR is decreased linearly from unit value to zero. Thus, ε is defined as:

$$\varepsilon = 1 - \frac{iter}{iter_{\max}} \tag{5}$$

where *iter* is the actual iteration number and $iter_{max}$ is the maximum number of iterations. Here, COR with unit value and zero value represent the global search and local search, respectively. The new positions of the CBs are evaluated using the generated velocities after the collision in the position of the stationary CBs:

$$x_i^{new} = \begin{cases} x_i + rand \circ v_i', & i = 1, \ldots, n \\ x_{i-n} + rand \circ v_i', & i = n+1, \ldots, 2n \end{cases} \tag{6}$$

where x_i^{new} and v'_i are the new position and the velocity after the collision for the *i*th CB, respectively.

2.3. The maximin strategy

The solution ranking procedure of the proposed method is based on the maximin sorting scheme. The maximin strategy was first introduced by Balling (2003) in the game theory. Recently in the literature multi-objective optimization algorithms, the maximin method is used instead of the well-known crowding distance technique, because more uniformly distributed solutions than those considered by the original crowding distance sorting rule (Menchaca-Mendez & Coello, 2016; Xiaodong, 2004) can be achieved.

Let us consider a MOO problem with *M* objective functions and an optimization algorithm with *n* solutions. The maximin value for the *k*th solution can be expressed as follows:

$$f_{\max}^{\mathrm{i}\min k} = \max_{j=1,2,\ldots,n; j\neq k} \{ \min_{i=1,2,\ldots,M} \{ f_i(x_k) - f_i(x_j) \} \} \tag{7}$$

In this equation, the min operator is first used to calculate the minimum value over all the objective functions. Then, the max operator is taken over all agents except for the *k*th solution. From this equation it can be concluded that: (i) any solution for which the maximin value > 0 is a dominated solution, (ii) any solution with maximin value as < 0 is a non-dominated solution, and (iii) any solution having the maximin value as = 0 is a weakly-dominated solution (Xiaodong, 2004).

Another characteristic that makes the maximin fitness function applicable to multi-objective optimization is that its value can be used to show the diversity of the non-dominated solutions. Therefore no additional diversity maintaining mechanism such as a niching technique is required. In fact, a solution with a smaller maximin value is located in sparsely populated areas.

3. The proposed multi-objective algorithm

This section introduces a MOO algorithm, which is a single solution search method. The proposed algorithm is named as "multi-objective colliding bodies optimization (MOCBO)" because of utilizing the formulation of the CBO as the search engine of the algorithm. As previously discussed, the multi-objective algorithms aim to achieve two main goals: (1) extracting a non-dominated front that is close to the true Pareto front (convergence), and (2) maintaining the diversity of the solutions along the resulting Pareto front (sparsity) (Kaveh & Laknejadi, 2011). In the proposed method, two operations different from the standard single solution CBO are performed: i) ranking the solutions based on maximin value to push the agents to low-crowded region of Pareto front; ii) incorporating an archive to the algorithm for saving the non-dominated solutions forming the Pareto front.

3.1. Methodology

The following steps summarize the main procedure for the implementation of the MOCBO.

Step 1. Similar to the other meta-heuristic algorithms, the initial positions of the population are calculated with random initialization in the search space as:

$$X_i^0 = X_{\min} + R(X_{\max} - X_{\min}) \quad , i = 1, 2, \ldots, N \tag{8}$$

where X_i^0 determines the initial value vector of the *i*th population; X_{min} and X_{max} are respectively the minimum and maximum allowable values vectors of variables; *R* is a random vector in the interval [0, 1] from a uniform distribution; and *N* is the number of populations.

Step 2. For all populations $X_{i=1, 2, \ldots, N}$, the objective functions $\{f_1(X_i), f_2(X_i), \ldots, f_M(X_i)\}$ are calculated.

Step 3. An empty external archive is first defined and then it is updated in each iteration. In this update all the currently non-dominated solutions are inserted into the archive and dominated solutions from the archive are deleted. Since the size of the archive is limited, one should apply a secondary mechanism for keeping this limit: first, the objective function space is divided into grids (hypercubes), *ngrid*. Then, the non-dominated solutions to their corresponding hypercube are located according to their objective function values (Coello, Pulido, & Lechuga, 2004). Afterwards, hypercubes that contain more than one population, the nearest population to grids is kept and the remaining populations are eliminated.

Step 4. The maximin value $f_{\max \mathrm{i} \min}^{\mathrm{k}=1,2,\ldots,N}$ of each solution is computed as given by Eq. (7).

Step 5. The arrangement of the populations is performed in ascending order based on the maximin values. Hence, if a population has the smallest maximin value, it is assigned as the best rank. This ranking approach is beneficial to the convergence and sparsity of algorithm.

Step 6. In this step, first the sorted populations are equally divided into two groups: (1) The lower half of sorted populations (stationary group), and (2) The upper half of sorted populations (moving group). With regard to ranking the populations based on maximin value, the stationary group is close to the true Pareto front as well as located in sparse regions of the resulting Pareto front than the moving group. Then, the populations of moving group are moved toward the stationary group and mating and collision process takes place. Hence, the velocities of the populations are evaluated as described in the CBO algorithm by Eq. (2).

Step 7. After the collision, the velocities of populations in each group are calculated as explained in the CBO algorithm with different definition of the population mass. By reshaping Eq. (3), the velocity of populations after the collision will be obtained as:

$$v_i' = \begin{cases} \frac{(1+\varepsilon)v_{i+n}}{1+mr_i}, & i = 1, \ldots, n \\ \frac{(1-\varepsilon mr_{i-n})v_i}{1+mr_{i-n}} & i = n+1, \ldots, 2n \end{cases} \tag{9}$$

where mr_i is the ratio of masses of the *i*th populations pair (Fig. 1). In other words, mr_i is the ratio of masses of the stationary and moving populations with its value being bigger than one. In the proposed MOCBO algorithm, this is calculated as:

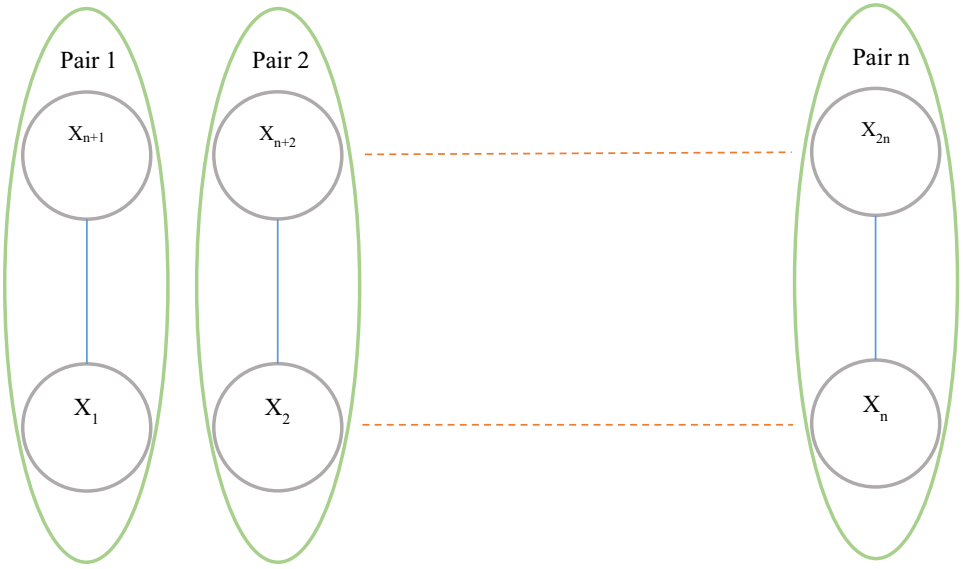

Fig. 1. The mating process in MOCBO.

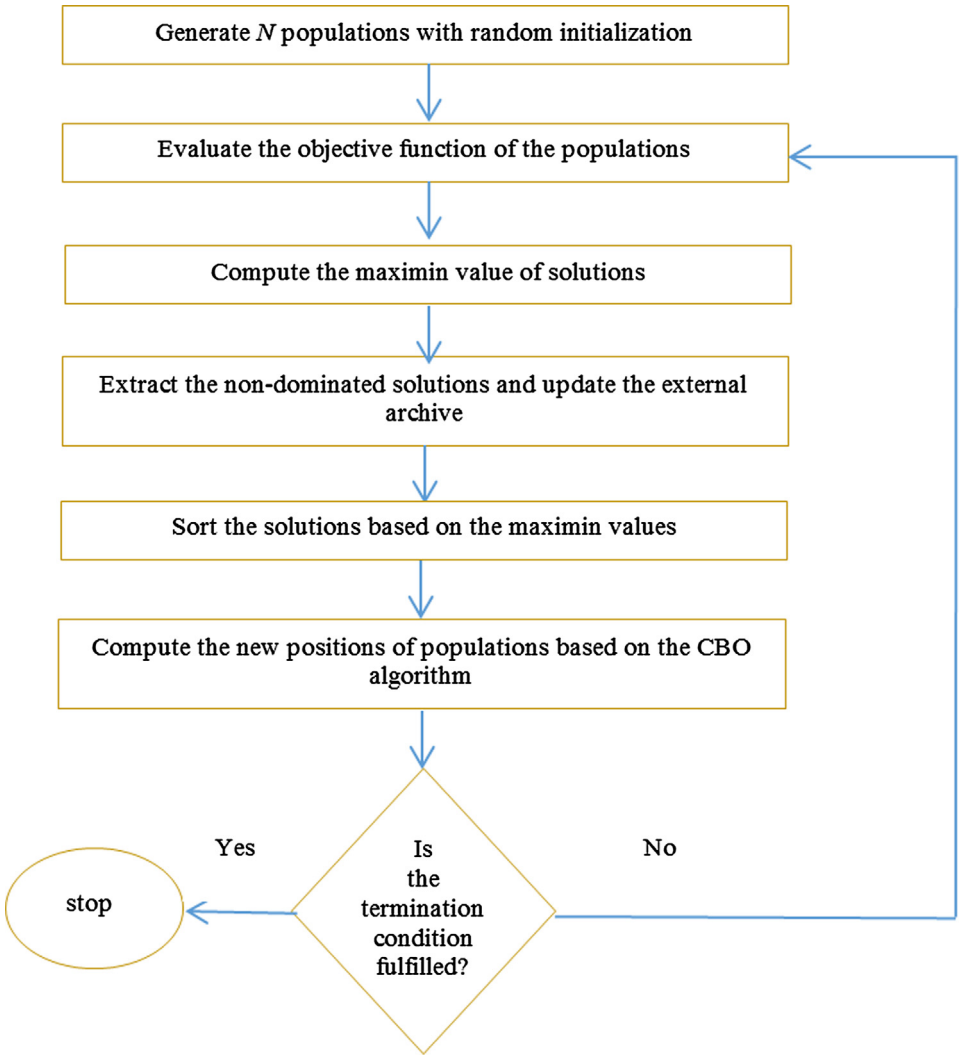

Fig. 2. Flowchart of the MOCBO algorithm.

Table 1
Mathematical test functions.

Functions	d	Bounds	Objective functions
ZDT1	30	[0,1]	$F_1 = x_1, F_2 = g(1 - \sqrt{F_1/g}), g = 1 + (9\sum_{i=2}^{d} x_i/(d-1))$
ZDT3	30	[0,1]	$F_1 = x_1, F_2 = g[1 - \sqrt{F_1/g} - (F_1/g)\sin(10\pi F_1)], g = 1 + (9\sum_{i=2}^{d} x_i/(d-1))$
ZDT4	10	$x_1 \in [0,1]$ $x_i \in [-5,5]$ $i = 2,\ldots,d$	$F_1 = x_1, F_2 = g[1 - \sqrt{F_1/g}], g = 1 + 10(d-1) + \sum_{i=2}^{d}[x_i^2 - 10\cos(4\pi x_i)]$
ZDT6	10	[0,1]	$F_1 = 1 - \exp(-4x_1)\sin^6(6\pi x_1), F_2 = g[1 - (F_1/g)^2], g = 1 + 9[\sum_{i=2}^{d} x_i/(d-1)]^{0.25}$
DTLZ1	12	[0,1]	$F_1 = (1/2)x_1x_2(1+g), F_2 = (1/2)x_1(1-x_2)(1+g), F_3 = (1/2)(1-x_1)(1+g) g = 100[10 + \sum_{i=3}^{d}(x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5))]$
DTLZ2	12	[0,1]	$F_1 = (1+g)\cos(x_1(\pi/2))\cos(x_2(\pi/2)), F_2 = (1+g)\cos(x_1(\pi/2))\sin(x_2(\pi/2)) F_3 = (1+g)\sin(x_1(\pi/2)), g = \sum_{i=3}^{d}(x_i - 0.5)^2$
DTLZ4	12	[0,1]	$F_1 = (1+g)\cos(x_1^{\pi}(\pi/2))\cos(x_2^{\pi}(\pi/2)), F_2 = (1+g)\cos(x_1^{\pi}(\pi/2))\sin(x_2^{\pi}(\pi/2)) F_3 = (1+g)\sin(x_1^{\pi}(\pi/2)), g = \sum_{i=3}^{d}(x_i - 0.5)^2$

Table 2
Parameter settings for all algorithms.

Parameters	Algorithms			
	SPEA2	NSGA-II	MOPSO	MOCBO
Crossover probability (p_c)	0.9	0.9	–	–
Mutation probability (p_w)	1/d	1/d	0.5	–
Population size (N)	100	100	100	50
External archive size ($nrep$)	100	100	100	–
Number of adaptive grid ($ngrid$)	–	–	30	30
Inertia weight (ω)	–	–	0.4	–

Table 3
Mean (and variance) of results from different optimization methods for test functions.

Problems	Metrics	Algorithms			
		SPEA2	NSGA-II	MOPSO	MOCBO
ZDT1	GD	0.09726(0.02689)	0.05625(0.00696)	0.00857(0.00298)	**0.00490(0.00112)**
	S	0.01379(0.01302)	0.00538(0.00450)	0.00588(0.00068)	**0.00289(0.00017)**
	SP	0.93882(0.01562)	0.71375(0.11560)	0.99529(0.00350)	**0.99770(0.00061)**
ZDT3	GD	0.05068(0.01149)	0.04583(0.01206)	0.01974(0.01510)	**0.00737(0.00158)**
	S	0.01274(0.01028)	**0.00951(0.00500)**	0.01406(0.00753)	0.02064(0.00818)
	SP	0.87266(0.00969)	0.65134(0.11338)	**0.90142(0.01886)**	0.87301(0.06852)
ZDT4	GD	1.80204(0.60866)	1.37588(0.05149)	1.36851(1.19047)	**0.00904(0.00202)**
	S	2.76859(3.78815)	0.00918(0.00212)	**0.01227(0.00436)**	0.01777(0.01754)
	SP	0.93378(0.35639)	0.76206(0.01898)	1.00523(0.48230)	**0.99691(0.00278)**
ZDT6	GD	0.04333(0.03131)	0.10794(0.09963)	0.47975(1.18272)	**0.00513(0.01317)**
	S	0.08039(0.11257)	0.19886(0.19108)	0.14881(0.13069)	**0.00485(0.00041)**
	SP	0.96264(0.02055)	0.95268(0.18296)	0.94769(0.50052)	**0.99650(0.01259)**
DTLZ1	GD	21.62238(9.20021)	7.61469(2.62614)	**0.00782(0.00137)**	0.09859(0.15295)
	S	40.56597(15.3632)	0.34408(0.13343)	**0.01382(0.00098)**	0.14898(0.28486)
	SP	0.58251(0.30189)	0.98467(0.19702)	0.99180(0.01315)	**0.99989(0.00013)**
DTLZ2	GD	0.05407(0.01143)	0.01693(0.00374)	0.08344(0.00658)	**0.00697(0.00035)**
	S	0.04520(0.01610)	0.03553(0.00288)	0.04442(0.00544)	**0.01054(0.00562)**
	SP	0.99957(0.00113)	0.97136(0.02561)	0.95127(0.01891)	**0.99999(0.00005)**
DTLZ4	GD	0.06809(0.01545)	0.01506(0.00242)	0.08520(0.00813)	**0.00999(0.00013)**
	S	0.05347(0.01112)	0.03466(0.00310)	0.04837(0.00825)	**0.02021(0.00144)**
	SP	0.98105(0.01971)	0.99254(0.03651)	0.96254(0.02590)	**1(0)**

Table 4
The average required time (sec) per run for test functions.

Problems	Algorithm			
	SPEA2	NSGA-II	MOPSO	MOCBO
ZDT1	367.5	514.1	40.3	10.1
ZDT3	367.2	559.4	37.1	8.1
ZDT4	365.7	487.9	42.4	9.4
ZDT6	353.8	529.3	36.9	7.8
DTLZ1	727.4	1131.4	264.9	41.5
DTLZ2	726.7	948.9	274.7	38.3
DTLZ4	719.7	961.8	274.5	25.8

$$mr_i = n + 1 - i, \quad i = 1, \ldots, n \tag{10}$$

Then, the masses ratios are decreased from n to 1 with increasing the rank of the pairs. This equation indicates that the low ranked populations are considerably changed after collision compared to the high ranked populations.

Step 8. After calculating the new velocities of populations, the new positions of the populations are calculated as described in the standard CBO algorithm (Eq. (6)).

Step 9. The optimization is repeated from Step 2 until a termination criterion is fulfilled. Many stopping criteria, such as maximum iteration number or no improvement of the external

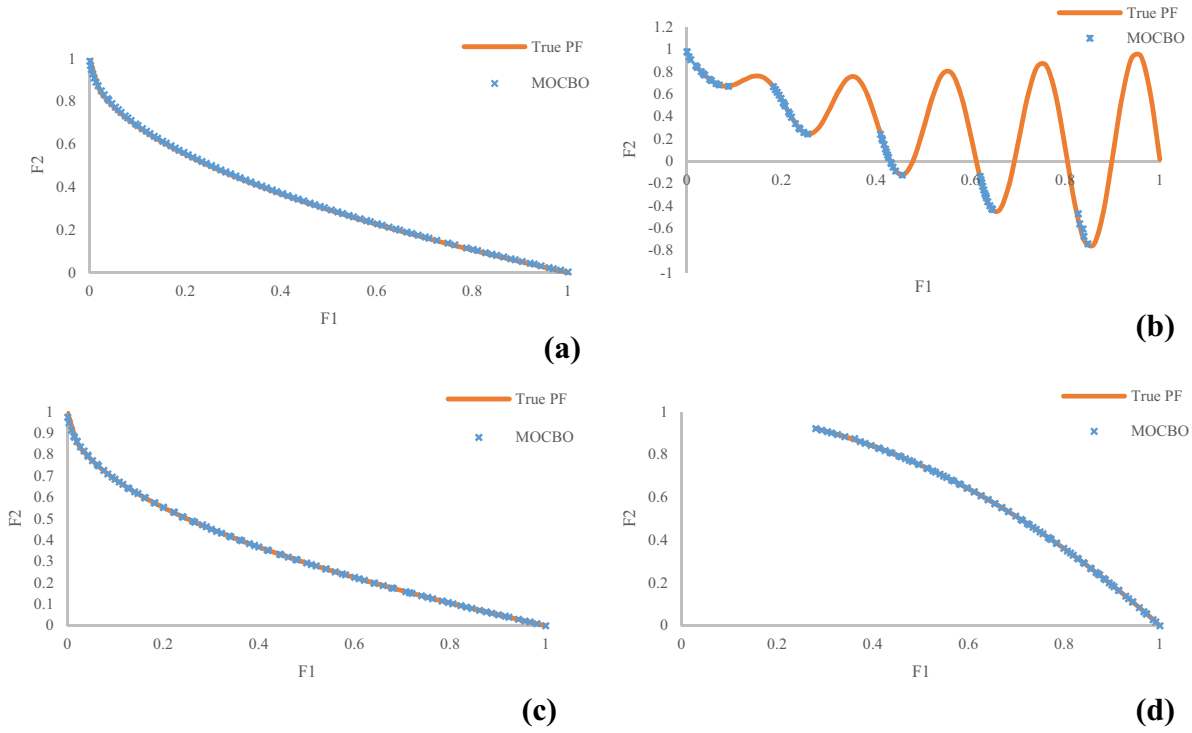

Fig. 3. True and resulted Pareto front by MOCBO on bi-objective test functions: (a) ZDT1, (b) ZDT3, (c) ZDT4, (d) ZDT6.

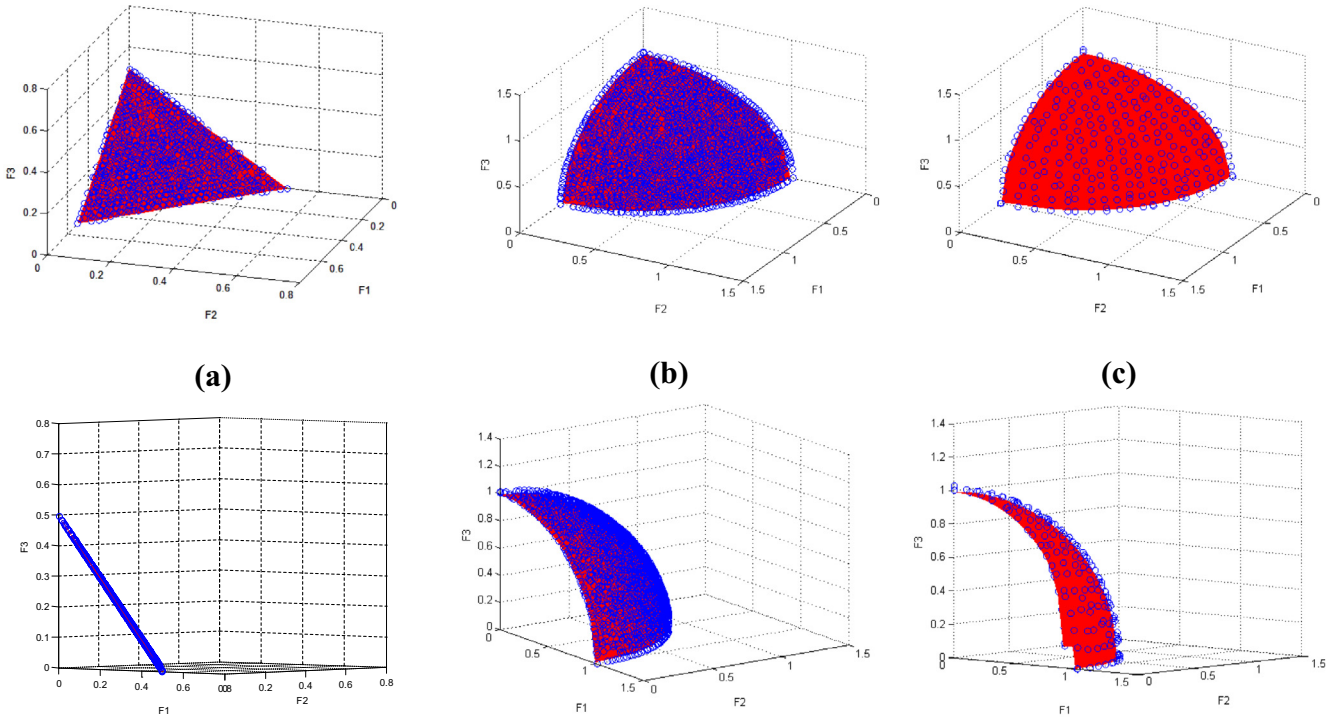

Fig. 4. True and resulted Pareto front by MOCBO on tri-objective test functions: (a) DTLZ1, (b) DTLZ2, (c) DTLZ4.

archives, can be utilized for the proposed method (Abdul Kadhar & Baskar, 2018; Martí, García, Berlanga, & Molina, 2016; Rudenko & Schoenauer, 2004). In the MOCBO algorithm, for ensuring a fair comparison among the proposed algorithms, a maximum number of function evaluations is considered as the stopping criterion.

The flowchart of the proposed MCBO algorithm is shown in Fig. 2.

4. Numerical examples

In this section, seven benchmark test function examples and two well-studied truss structures taken from the previous optimization literature are used to study the efficiency of the proposed approach. The test function examples have been previously solved using a variety of other multi-objective algorithms, which is useful to show the validity and robustness of the presented method. The truss structure optimization problems are also previously solved using different single-objective optimization algorithms. The algorithms are coded in MATLAB. Structural analysis is performed with the direct stiffness method. The optimization process is performed by a Core™ 2 Duo 2.53 GHz computer and the time for all the computations are evaluated in clock time.

4.1. Performance metrics

In this paper, we utilized three most important metrics to evaluate the performance of the proposed MOO algorithm as follow (Deb et al., 2002; Zitzler, Deb, & Thiele, 2000):

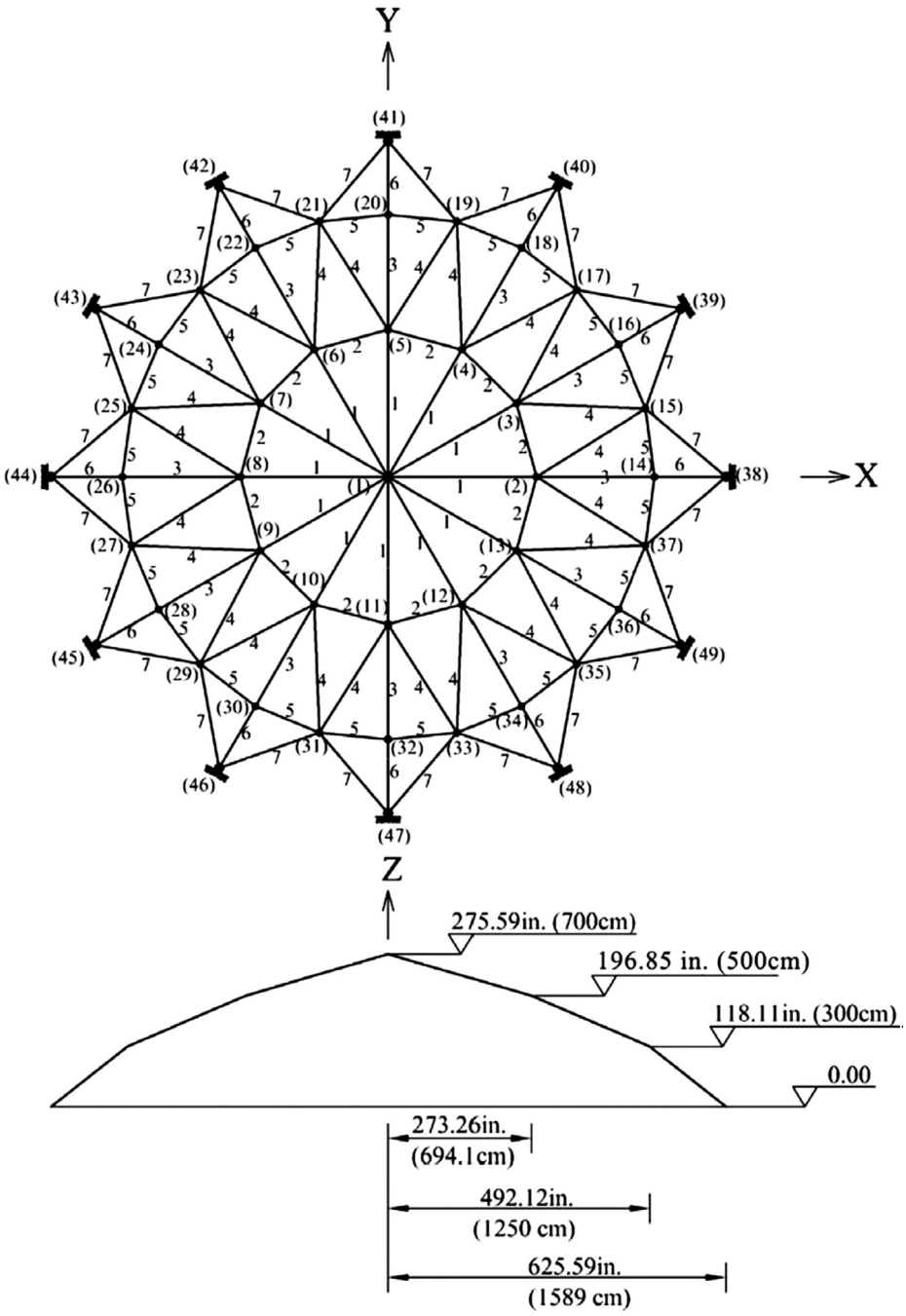

Fig. 5. Schematic of the spatial 120-bar dome truss with indication of design variables.

- Generational distance (GD): This measures the distance between the true (PF_{true}) and resulted Pareto front (PF_{known}). The metric is represented as follows:

$$GD = \left(\frac{1}{n_{pf}}\sum_{i=1}^{n_{pf}} dis_i^2\right)^{\frac{1}{2}} \tag{11}$$

where n_{pf} is the number of non-dominated solutions in PF_{known} and d_i is the Euclidean distance between the solution i in PF_{known} and its nearest solution in PF_{true}. A smaller value of GD represents better convergence.

- Spacing (S): This metric provides an indication of the evenness of the solutions distributed along the set of PF_{known}:

$$S = \left(\frac{1}{n_{pf}}\sum_{i=1}^{n_{pf}} (d_i - \bar{d})^2\right)^{\frac{1}{2}} \quad where \quad \bar{d} = \frac{1}{n_{pf}}\sum_{i=1}^{n_{pf}} d_i \tag{12}$$

where d_i is the Euclidean distance between the solution i and its nearest consecutive solution in the PF_{known}. A smaller value of S represents a more uniform distribution of the obtained non-dominated solutions.

- Maximum spread (MS): This metric measures the distance between the boundary solutions based on the extreme function values observed in the PF_{true} and PF_{known}, and it is defined as follows:

$$MS = \left[\frac{1}{m}\sum_{i=1}^{m}\left[\frac{\min(f_i^{\max}, F_i^{\max}) - \max(f_i^{\min}, F_i^{\min})}{F_i^{\max} - F_i^{\min}}\right]^2\right]^{\frac{1}{2}} \tag{13}$$

where m is the number of objectives, $f_i^{\max}$ and $f_i^{\min}$ are the extreme values of the ith objective in PF_{known}. $F_i^{\max}$ and $F_i^{\min}$ are the extreme values of the ith objective in PF_{true}. A larger value of MS represents a better spread of the solutions is achieved.

4.2. Mathematical test functions

Several non-constraint benchmark test functions with known Pareto fronts have been selected from the literature (Deb et al., 2002; Zitzler et al., 2000). These examples consist of four bi-objective and three tri-objective optimization problems with different characteristics of PF. The formulations of the seven problems are shown in Table 1. In this section, four bi-objective examples: ZDT1, ZDT3, ZDT4 and ZDT6 are selected respectively as the convex, disconnected convex, nonconvex and nonuniformly disconnected nonconvex MOO problems. Also, three bi-objective examples: DTLZ1, DTLZ2 and DTLZ4 are selected respectively as the linear separable multimodal, concave scalable multimodal and concave scalable unimodel MOO problems. The variable bounds and number of variables (d) for each problem are also described in Table 1.

In order to perform a comparative study, these tests are also solved using Strength Pareto Evolutionary Algorithm 2 (SPEA2) (Zitzler et al., 2000), Non-dominated Sorting Genetic Algorithm II (NSGA-II) (Deb et al., 2002; Nestorović, Trajkov, & Garmabi, 2015) and Multi-Objective Particle Swarm Optimization (MPSO) (Coello et al., 2004) algorithms. For these algorithms, we used the same parameter settings as in the original papers. The parameter settings for these algorithms and the proposed algorithm are summarized in Table 2. For all the algorithms, the number of iterations is set as 500 and these examples are independently optimized 20 times.

Table 3 presents the average and standard division results of three performance metrics obtained using the compared algorithms and the proposed MOCBO algorithms. The best average results of each metric and problem are shown in boldface. It can be seen, the best results for all metrics of the ZDT1, DTLZ2 and DTLZ4 problems are obtained by the MOCBO. For remaining examples, the results obtained by MOPSO and NSGA-II are better than MOCBO in some metrics. It should be noted that the number of fitness function evaluations for the proposed and selected algorithms are set to 50,000 and 25,000, respectively.

The time elapsed per run for all the algorithms and examples are also recorded in Table 4. Based on the results shown in Table 3, it can be concluded that the consumed time for MOCBO is much less than the required time for the selected algorithms. This is because of achieving the simple non-memory formulation and maximin function in the MOCBO.

Fig. 3 and Fig. 4 show the true Pareto front curves and best Pareto front sets obtained using MOCBO algorithm for two and three objectives examples, respectively. These figures prove that MOCBO is able to find solutions near the global Pareto front with better solution diversities.

4.3. Truss structure examples

In the following, the efficiency of the MOCBO will be tested in optimizing two structural problems considering two conflicting objective functions. Based on the results in Section 4.2, it can be concluded that MOCBO, MOPSO and NSGA-II produce competitive results from GD, SP and MS points of view and the consumed time for these algorithms are much less than the required time for SPEA2. Thus, these problems are solved using MOCBO, MOPSO and NSGA-II algorithms. The number of populations is considered

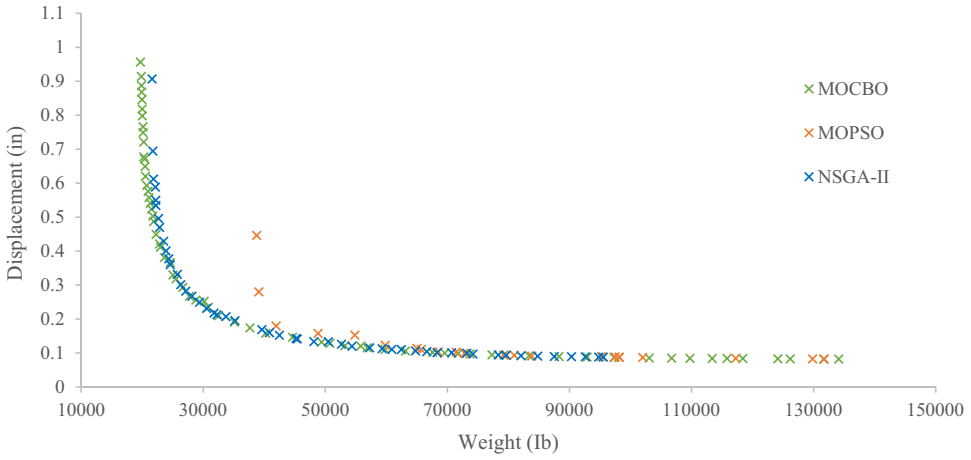

Fig. 6. Pareto front produced of MOPSO, MOCBO and NSGA-II algorithms on 120-bar dome truss example.

as 50 for both MOCBO and MOPSO. The number of iterations is also set to 300.

In structural MOO problem, the main objectives are to minimize the weight of the structures and displacement under some design constraints (Kaveh & Laknejadi, 2013). The MOO problem for a truss structure can be expressed as follows:

$$\begin{aligned} &\text{Find } X=[x_1,x_2,x_3,\ldots,x_n]\\ &\text{to minimize } \{W(X),U(X)\}\\ &\text{subjected to } g_j(X)\\ &\leqslant 0, j=1,2,\ldots,m\\ &x_{lmin}\leqslant x_l\leqslant x_{lmax} \end{aligned} \tag{14}$$

where all design variables are collected in the vector X having n unknowns; $W()$ and $U()$ are respectively the weight and displacements of the truss structure; g_j is the jth constraint from m inequality constraints. Also, x_{lmin} and x_{lmax} are the lower and upper bounds of design variable vector, respectively.

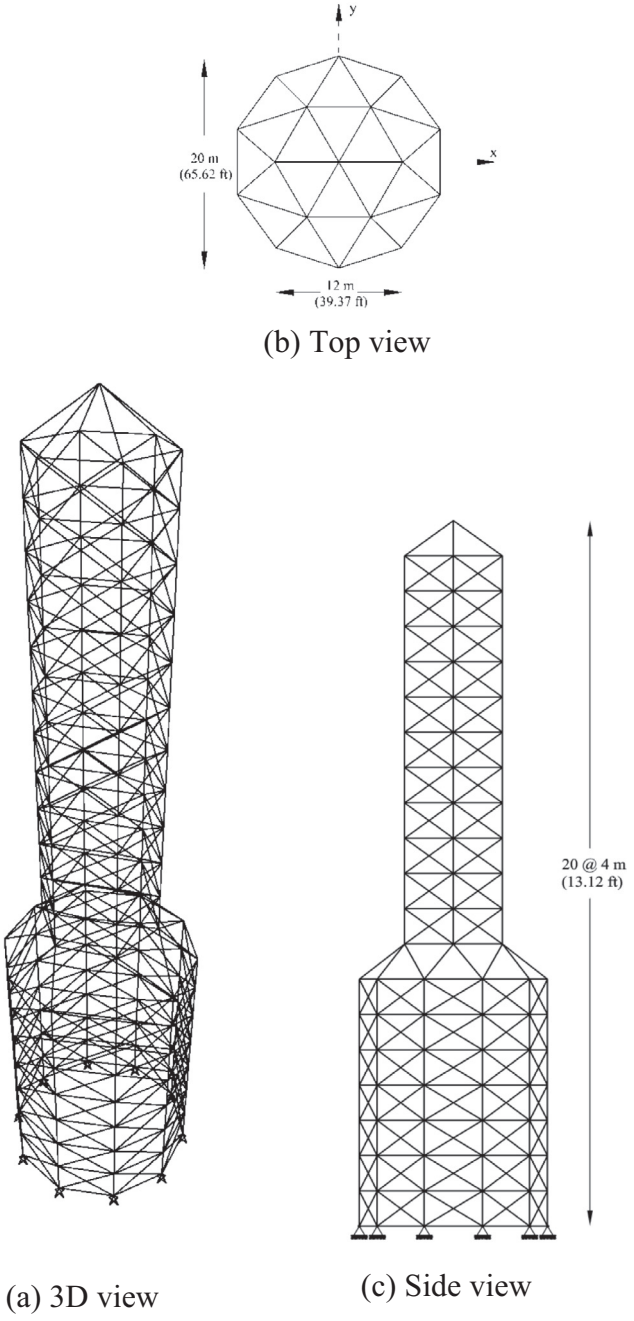

Fig. 7. Schematic of the spatial 582-bar tower.

In this study, we utilized the exterior penalty function method for considering the constrained structural optimization problem as an unconstrained problem (Deb, 2000). When some constraints are violated in a particular solution, the penalty function magnifies all objective functions by taking values bigger than unity.

4.3.1. Design constraints for truss structures

Based on the AISC ASD (1989) code, the allowable tensile and compressive stresses are calculated as follows:

$$\begin{cases} \sigma_i^+ = 0.6F_y & for \quad \sigma_i \geqslant 0\\ \sigma_i^- & for \quad \sigma_i \leqslant 0 \end{cases} \tag{15}$$

With σ_i^- being evaluated in terms of the slenderness ratio as:

$$\sigma_i^- = \begin{cases} \left[\left(1-\frac{\lambda_i^2}{2C_c^2}\right)F_y\right] \Big/ \left(\frac{5}{3}+\frac{3\lambda_i}{8C_c}-\frac{\lambda_i^3}{8C_c^3}\right) & for \quad \lambda_i < C_c\\ \frac{12\pi^2 E}{23\lambda_i^2} & for \quad \lambda_i \geqslant C_c \end{cases} \tag{16}$$

where E and F_y are respectively the modulus of elasticity and yield stress of steel, C_c is the slenderness ratio dividing the elastic and inelastic buckling regions ($C_c=\sqrt{2\pi^2E/F_y}$), λ_i is the slenderness ratio ($\lambda_i=\frac{KL_i}{r_i}$), K is the effective length factor of members, L_i is the length of the ith member and r_i is the radius of gyration of the ith member.

4.3.2. A 120-bar truss dome

The first test case study is the weight and displacement minimization problem of the 120-bar truss dome shown in Fig. 5. This test case was investigated as a single objective (weight optimization) size optimization problem by different methods (Kaveh & Mahdavi, 2014, 2015; Lee & Geem, 2004).

In this paper, the objectives are considered as the total structural weight and the maximum displacement on all the nodes in all directions. The modulus of elasticity, yield stress and material

Table 5
List of sections used for the 582-bar tower truss example.

No.		No.	
1	W8X21	28	W14X61
2	W10X22	29	W21X62
3	W8X24	30	W12X65
4	W6X25	31	W16X67
5	W12X26	32	W10X68
6	W8X28	33	W12X72
7	W12X30	34	W14X74
8	W14X30	35	W18X76
9	W8X31	36	W10X77
10	W10X33	37	W12X79
11	W14X34	38	W14X82
12	W8X35	39	W27X84
13	W16X36	40	W18X86
14	W14X38	41	W12X87
15	W10X39	42	W10X88
16	W8X40	43	W16X89
17	W14X43	44	W14X90
18	W12X45	45	W21X93
19	W10X45	46	W27X94
20	W14X48	47	W12X96
21	W10X49	48	W18X97
22	W12X50	49	W14X99
23	W12X53	50	W10X100
24	W10X54	51	W16X100
25	W12X58	52	W21X101
26	W10X60	53	W24X104
27	W8X21	54	W12X106

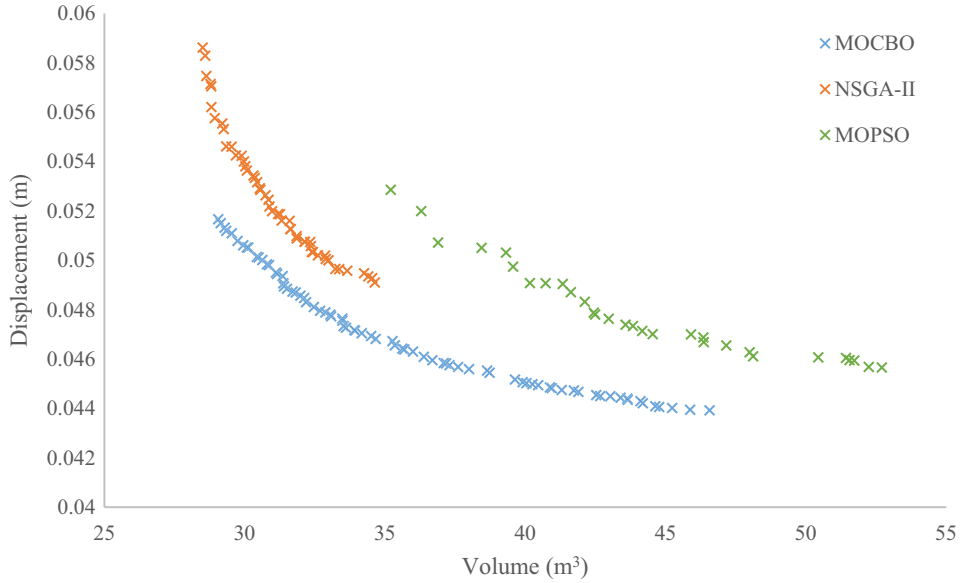

Fig. 8. Pareto front produced of MOPSO, MOCBO and NSGA-II algorithms on 582-bar tower example.

density are respectively considered $E = 30.450$ ksi, $F_y = 58.0$ ksi and $\rho = 0.288$ lb/in^3. The radius of the gyration (r_i) is expressed in terms of cross-sectional areas as $r_i = aA_i^b$ (Saka, 1990), where a and b are constants depending on the types of sections used for the members. For this example, pipe sections ($a = 0.4993$ and $b = 0.6777$) are used for the bars. The members of the dome are categorized into seven groups, as illustrated in Fig. 5. The truss is subjected to vertical loads at all the non-support joints. These 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 upper and lower boundaries of each truss element are 0.775 in^2 and 20.0 in^2, respectively.

As mentioned before, this problem is solved using MOCBO, MOPSO and NSGA-II algorithms. Each algorithm is run 20 times and the best one is selected to present graphically. It should be noted that, since the true Pareto front is not known, the considered performance metrics utilized in the previous section are not applicable here. The resulted Pareto fronts from three multi-objective optimization methods are presented in Fig. 6. It can be seen, the *PF* set obtained using MOCBO is dominated that ones obtained using MOPSO and NSGA-II. Additionally, MOCBO has acceptable convergence performance and is able to cover all parts of PF and the obtained set of solutions is distributed uniformly. The average elapsed time per run of the MOCBO, MOPSO and NSGA-II were 270.2, 330.8 and 510.7 s, respectively.

4.3.3. A 582-bar tower truss

The 582-bar spatial truss structure, shown in Fig. 7, was studied with discrete variables by other researchers as single objective optimization problem (Hasançebi, Çarbas, Dogan, Erdal, & Saka, 2009; Kaveh & Talatahari, 2009). The 582 structural members of this spatial truss are categorized into 32 independent groups.

All nodes of the tower are subjected to lateral loads of 5.0 kN (1.12 kips) in both x- and y-directions and a vertical load of − 30 kN (−6.74 kips) in the z-direction. Design variables are selected from a discrete set of 54 standard steel W-shaped sections, as listed in Table 5, based on the area and radii of gyration of the sections. In this example, the members are also subjected to the maximum slenderness ratio limits of 300 for tension members, and it is recommended to be limited to 200 for compression members based on ASD-American Institute of Steel Construction (AISC) (1989). The modulus of elasticity is $E = 29{,}000$ ksi (203893.6 MPa) and the yield stress of steel is taken as $F_y = 36$ ksi (253.1 MPa).

In this paper, two objectives are considered as explained in the previous example. The results of this problem, MOCBO, MOPSO and NSGA-II Pareto front sets are given in Fig. 8. Similar to the previous example, one can observe that the Pareto fronts obtained by the proposed MOCBO, dominate the MOPSO and NSGA-II's Pareto fronts. These results also prove the convergence and sparsity efficiency of the MOCBO in optimizing multiple conflicting functions. Also, the average required time per run of the MOCBO, MOPSO and NSGA-II were 4290.6, 5970.4 and 6150.1 s, respectively.

5. Conclusion remarks

In this paper, a new multi-objective optimizer based on the CBO algorithm strategy, called MOCBO, is developed. This algorithm is compared to other MOO algorithms, the MOCBO has several distinct features. Firstly, the movement of population/agents in the MOCBO is achieved without considering the old positions in the optimization search at each iteration. Hence, one can consider the proposed algorithm as a memory-less MOO algorithm category. Secondly, the domination situations and ranks of the solutions are evaluated based on the maximin function. Finally, like the standard CBO algorithms, where optimization process is achieved without considering parameter tuning, formulations of this algorithm is parameter independent.

The MOCBO algorithm is first tested over seven benchmark problems consisting of bi-objective and tri-objective unconstraint optimization problems with different characteristics. The results are compared to those of some population based MOO metaheuristics algorithms. This comparison reveals that besides its simplicity, the proposed MOCBO algorithm is also competitive, from computational time point of view compared to the selected MOO algorithms because of its memory-less formulation as well as using the maximin function, when compared to the performance of some other algorithms. Additionally, the MOPSO, MOCBO and NSGA-II are selected for optimization of two truss structural considering two conflicting objective functions containing the structural weight and nodal displacement. Despite the complexity of this type of problem, the MOCBO resulted in better results, high convergence rate and more uniform distribution, than those found in MOPSO and NSGA-II, even for high-dimensional problems.

Conflict of interest

No conflict of interest.

References

Abdul Kadhar, K. M., & Baskar, S. A. (2018). Stopping criterion for decomposition-based multi-objective evolutionary algorithms. *Soft Computing, 22*(1), 253–272.

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

Angelo, J. S., Bernardino, H. S., & Barbosa, H. J. C. (2015). Ant colony approaches for multiobjective structural optimization problems with a cardinality constraint. *Advances in Engineering Software, 80*, 101–115.

Balling, R. (2003). The maximin fitness function; multi-objective city and regional planning. *Proceedings of EMO*, 1–15.

Bilela, N., Mohameda, N., Zouhaiera, A., & Lotfib, R. (2016). An improved imperialist competitive algorithm for multi-objective optimization. *Engineering Optimization, 48*(11), 1823–1844.

Chao, L., Shengqiang, X., Xinyu, L., & Liang, G. (2016). An effective multi-objective discrete grey wolf optimizer for a real-world scheduling problem in welding production. *Advances in Engineering Software, 99*, 161–176.

Clarke, J., & McLeskey, J. T. (2015). Multi-objective particle swarm optimization of binary geothermal power plants. *Applied Energy, 138*, 302–314.

Coello, C.A., Lechuga, M.S. (2002). MOPSO: A proposal for multiple objective particle swarm optimization. In *Proceedings of the 2002 congress on evolutionary computation, 2002*. CEC'02, 2, IEEE, 2002, pp. 1051–1056.

Coello, C. A., Pulido, G. T., & Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation, 8*(3), 256–279.

Coello, C. A. Coello, Van Veldhuizen, D. A., & Lamont, G. B. (2002). *Evolutionary algorithms for solving multi objective problems*. New York: Kluwer Academic Publishers.

Deb, K. (1999). Multi objective genetic algorithms: Problem difficulties and construction of test problem. *Evolutionary Computation, 7*(3), 205–230.

Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering, 186*, 311–338.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGAII. *IEEE Transactions on Evolutionary Computation, 6*(2), 182–197.

Fragiadakis, M., Lagaros, N., & Papadrakakis, M. (2006). Performance-based multiobjective optimum design of steel structures considering life-cycle cost. *Structural and Multidisciplinary Optimization, 32*, 1–11.

Guo, W., Wang, L., & Wu, Q. (2016). Numerical comparisons of migration models for multi-objective biogeography-based optimization. *Information Sciences, 328*, 302–320.

Haimes, Y. Y., Lasdon, L. S., & Wismer, D. A. (1971). On a bicriterion formulation of the problems of integrated system identification and system optimization. *IEEE Transactions on Systems, Man, and Cybernetics, 1*(3), 296–297.

Hasançebi, O., Çarbas, S., Dogan, E., Erdal, F., & Saka, M. P. (2009). Performance evaluation of metaheuristic search techniques in the optimum design of real size pin jointed structures. *Computers & Structures, 87*, 284–302.

Hassanzadeh, H.R., Rouhani, M. (2010). A multi-objective gravitational search algorithm. In *Proceedings of the second international conference on computational intelligence, communication systems and networks (CICSyN)*, IEEE, pp. 7–12.

Ho-Huu, V., Duong-Gia, D., Vo-Duy, T., Le-Duc, T., & Nguyen-Thoi, T. (2018). An efficient combination of multi-objective evolutionary optimization and reliability analysis for reliability-based design optimization of truss structures. *Expert Systems with Applications, 102*, 262–272.

Ho-Huu, V., Hartjes, S., Visser, H. G., & Curran, R. (2018). An improved MOEA/D algorithm for bi-objective optimization problems with complex Pareto fronts and its application to structural optimization. *Expert Systems with Applications, 92*, 430–446.

Hosseini, S., & Al Khaled, A. (2014). A survey on the imperialist competitive algorithm metaheuristic: Implementation in engineering domain and directions for future research. *Applied Soft Computing, 24*, 1078–1094.

Hu, P., Rong, L., Liang-Lin, C., & Li-xian, L. (2011). Multiple swarms multi-objective particle swarm optimization based on decomposition. *Procedia Engineering, 15*, 3371–3375.

Kaveh, A. (2017). *Advances in metaheuristic algorithms for optimal design of structures* (2nd ed.). Switzerland: Springer Verlag.

Kaveh, A., & Ardalani, S. (2016). Cost and CO_2 emission optimization of reinforced concrete frames using enhanced colliding bodies algorithm. *Asian Journal of Civil Engineering, 17*(6), 831–858.

Kaveh, A., & Laknejadi, K. (2011). A novel hybrid charge system search and particle swarm optimization method for multi-objective optimization. *Expert Systems with Applications, 38*, 15475–15488.

Kaveh, A., & Laknejadi, K. (2013). A new multi-swarm multi-objective optimization method for structural design. *Advances in Engineering Software, 58*, 54–69.

Kaveh, A., Laknejadi, K., & Alinejad, B. (2012). Performance based multi-objective optimization of large steel structures. *Acta Mechanica, 223*(2), 355–369.

Kaveh, A., & Mahdavi, V. R. (2014). Colliding bodies optimization: A novel meta-heuristic method. *Computers & Structures, 139*, 18–27.

Kaveh, A., & Mahdavi, V. R. (2015). *Colliding bodies optimization; extensions and applications*. Switzerland: Springer International Publishing.

Kaveh, A., & Massoudi, M. S. (2014). Multi-objective optimization using charged system search. *Scientia Iranica, 21*(6), 1845–1860.

Kaveh, A., Shojaei, I., Golipour, Y., & Rahami, H. (2013). Seismic design of eccentric braced frames using multi-objective optimization. *Structural Engineering and Mechanics; An International Journal, 45*(2), 211–232 (Techno Press).

Kaveh, A., & Talatahari, S. (2009). Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures. *Computers & Structures, 87*(5), 267–283.

Kim, I. Y., & de Weck, O. L. (2006). Adaptive weighted sum method for multiobjective optimization: A new method for Pareto front generation. *Structural and Multidisciplinary Optimization, 31*(2), 105–116.

Ko, C. H., & Wang, S. F. (2011). Precast production scheduling using multi-objective genetic algorithms. *Expert System with Applications, 38*(7), 8293–8302.

Lee, K. S., & Geem, Z. W. (2004). A new structural optimization method based on the harmony search algorithm. *Computers & Structures, 82*, 781–798.

Liu, M., Burns, S. A., & Wen, Y. K. (2005). Multi-objective optimization for performance-based seismic design of steel moment frame structures. *Earthquake Engineering & Structural Dynamics, 34*, 289–306.

Liu, K., Paulino, G., & Gardoni, P. (2016). Reliability-based topology optimization using a new method for sensitivity approximation - application to ground structures. *Journal of Structural and Multidisciplinary Optimization, 54*(3), 553–571.

Luh, G. C., & Chueh, C. H. (2004). Multi-objective optimal design of truss structure with immune algorithm. *Computers & Structures, 82*(11–12), 829–844.

Martí, L., García, J., Berlanga, A., & Molina, J. M. (2016). A stopping criterion for multi-objective optimization evolutionary algorithms. *Information Sciences, 367–368*, 700–718.

Mathakari, S., Gardoni, P., Agarwal, P., & Raich, A. (2007). Reliability-based optimal design of electrical transmission towers using multi-objective genetic algorithms. *Computer-Aided Civil and Infrastructure Engineering, 22*, 282–292.

Menchaca-Mendez, A., & Coello, C. A. Coello (2016). Selection mechanisms based on the maximin fitness function to solve multi-objective optimization problems. *Information Sciences, 332*, 131–152.

Miettinen, K. (1999). *Nonlinear multiobjective optimization*. Boston: Kluwer Academic Publishers.

Mirjalili, S., Saremi, S., Mirjalili, S. M., & Coelho, L. S. (2016). Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. *Expert Systems with Applications, 47*, 106–119.

Mousa, A., El-Shorbagy, M., & Abd-El-Wahed, W. (2012). Local search based hybrid particle swarm optimization algorithm for multiobjective optimization. *Swarm and Evolutionary Computation, 3*, 1–14.

Nestorović, T., Trajkov, M., & Garmabi, S. (2015). Optimal placement of piezoelectric actuators and sensors on a smart beam and a smart plate using multi-objective genetic algorithm. *Smart Structures and Systems, 15*(4), 1041–1062.

Nigdeli, S. M., Bekdaş, G., Kim, S., & Geem, Z. W. (2015). A novel harmony search based optimization of reinforced concrete biaxially loaded columns. *Structural Engineering and Mechanics, 54*(6), 1097–1109.

Rudenko, O., Schoenauer, M. (2004). A steady performance stopping criterion for pareto-based evolutionary algorithms. In *The proceedings of the 6th international multi-objective programming and goal programming conference*. Hammamet, Tunisia.

Saka, M. P. (1990). Optimum design of pin-jointed steel structures with practical applications. *Journal of Structural Engineering, ASCE, 116*, 2599–2620.

Srinivas, N., & Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation, 2*(3), 221–248.

Xiaodong, Li. (2004). Better spread and convergence: Particle swarm multiobjective optimization using the maximin fitness function. genetic and evolutionary computation – GECCO 2004 Volume 3102 of the series. *Lecture Notes in Computer Science*, 117–128.

Yi, T. H., Li, H. N., & Zhang, X. D. (2015). Health monitoring sensor placement optimization for Canton Tower using virus monkey algorithm. *Smart Structures and Systems, 15*(5), 1373–1392.

Yi, T. H., Wen, K. F., & Li, H. N. (2016). A new swarm intelligent optimization algorithm: Pigeon Colony Algorithm (PCA). *Smart Structures and Systems, 18*(3), 425–448.

Zadeh, L. (1963). OptiMality and non-scalar-valued performance criteria. *IEEE Transactions on Automatic Control, 8*(1), 59–60.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation, 8*(2), 173–195.