IJST, Transactions of Civil Engineering, Vol. 35, No. C2, pp 137-154
Printed in The Islamic Republic of Iran, 2011

A HYBRID MULTI-OBJECTIVE OPTIMIZATION AND DECISION MAKING PROCEDURE FOR OPTIMAL DESIGN OF TRUSS STRUCTURES*

A. KAVEH** AND K. LAKNEJADI

School of Civil Engineering, Iran University of Science and Technology, Tehran-16, I. R. of Iran
Email: alikaveh@iust.ac.ir

Abstract– In this paper, a new hybrid method is developed for optimal design of truss structures. This method is based on a modified multi-objective particle swarm optimization, tournament decision making process, and a local search algorithm. In structural design practice, different objectives are usually considered in the selection of the final design in which most of these objectives contradict each other. The use of a multi-objective optimization method guides the decision makers to find the most suitable design. Incorporating a decision making process with this optimization, it becomes possible to find a solution which covers most of the requirements. The developed hybrid algorithm is applied to three truss structures to illustrate its ability in finding the optimal solution.

Keywords– Multi-objective optimization, particle swarm optimization (PSO), decision making (DM) process, Pareto-optimal

1. INTRODUCTION

In modern-day design and decision-making processes, optimization plays a major role. In the past, most studies concentrated on finding the optimum corresponding to a single goal such as minimum cost or maximum performance [1-5]. For this purpose, the optimization algorithm searches through possible feasible solutions, and at the end identifies the best result. Such a solution often lacks the effect of other equally important goals. In recent years applied optimization has gone through a face-lift, particularly with the availability of efficient multi-objective optimization algorithms, which enables a designer or a decision maker to consider more than one conflicting goal simultaneously. In practice a good design requires a fine balance among different objectives. Such a task consists of two main sub-tasks: (i) an optimization procedure to discover high-performing solutions trading-off different conflicting goals of the design and (ii) a decision-making task to choose a single preferred solution. Therefore, the final solution should satisfy two conditions: it should be a Pareto-optimal point and it should also be the most suitable solution according to the preferences of a decision-maker.

In order to incorporate decision making process in an optimization procedure, three possible approaches are available in the literature, namely the *priori*, the *progressive* (interactive) and the *posteriori* [6]. Due to the advantages of the latter, in this paper the posteriori approach is used. In this approach first, a multi-objective optimization algorithm meets a discrete representation of the Pareto front, and then the decision making process chooses the final solution.

Many real world engineering design problems can be formulated as multi-objective optimization problems. One of the most important examples is to minimize the total weight of a truss while minimizing

*Received by the editors February 6, 2011; Accepted May 16, 2011.
**Corresponding author

its maximum deflection. In order to solve this problem, different approaches have been utilized [7-9], but the selection of the best solution from the constructed Pareto-front is task that is rarely covered in these studies. In structural design problems many different criteria, such as practical, theoretical or economical limitations may affect the process of selecting the final solution or the decision. Accordingly, incorporating a decision-making process in a suitable multi-objective algorithm may help engineers to make the most satisfactory decision in their design.

Over the past decade, a number of multi-objective evolutionary algorithms (MOEAs) have been developed, such as Non-dominated Sorting Genetic Algorithm (NSGA)-II, Ref. [10], Strength Pareto Evolutionary Algorithm, SPEA2 [11], Pareto Archive Evolution Strategy (PAES) [12], and Multi-objective particle swarm optimization [13]. Recently, researchers have been paying more and more attention to PSO to solve multi-objective problems [14-20]. Changing a PSO to a multi-objective PSO (MOPSO) requires a redefinition of what a guide is in order to obtain a front of optimal solutions. In MOPSO, the Pareto-optimal solutions should be used to determine the guide for each particle. However, selecting a guide from the set of Pareto-optimal solutions for each particle of the population is a very difficult, yet important problem for attaining convergence and diversity of the solutions. In order to solve this problem, in this study MOPSO is combined with the charged system search (CSS) algorithm [1], in such a way that the problem of guide selection of MOPSO is relieved adequately.

In this paper a new multi-objective optimization approach, mainly based on the particle swarm optimization method is incorporated with a simple multi-criteria decision making process called multi-criteria tournament decision making (MTDM), to provide a framework for the optimal design of structures. Since this algorithm is an evolutionary optimization algorithm, the solutions found in Pareto front can be slightly improved by a local search algorithm. In this study after the decision making process, a local search algorithm is used to improve the final selected solution.

2. PRELIMINARIES

For better understanding of the MOPs and DM processes, some concepts are important [6, 21], and have been summarized in the following:

Definition 1 (*General Multi-objective Optimization Problem*). In general, a multi-objective optimization for minimization problems can be described as follows:
Find a vector $x = (x_1, x_2, ..., x_n)$ which satisfies *k* inequality constraints as

$$q_i(x) \le 0 \quad (i = 1,2,...,k)$$

and *l* equality constraints as

$$h_j(x) = 0 \quad (j = 1,2,...,l)$$

and minimizes the vector function

$$\text{Min}_{x\in\Omega} \quad F(x) = \{f_1(x), f_2(x), ..., f_m(x)\} \tag{1}$$

Where Ω is a set of decision vectors, and *m* is the number of objectives. In fact, the aim is to find vectors subjected to some constraints which make all the objective values as small as possible.

Definition 2 (*Pareto Dominance*). A vector $\mathbf{u} = (u_1, u_2, ..., u_n)$ is said to *dominate* another vector $\mathbf{v} = (v_1, v_2, ..., v_n)$ (denoted by $\mathbf{u} \prec \mathbf{v}$) if and only if **u** is partially less than **v**, i.e., $\forall i \in \{1,2,...,n\}$, $u_i \le v_i \wedge \exists i \in \{1,2,...,n\} : u_i < v_i$.

Definition 3 (*Pareto Optimal*). A solution $x \in \Omega$ is said to be Pareto Optimal with respect to Ω if and only if there is no $x' \in \Omega$ for which $\mathbf{v} = (f_1(x'), f_2(x'), \ldots, f_n(x'))$ dominates $\mathbf{u} = (f_1(x), f_2(x), \ldots, f_n(x))$.

The phrase *Pareto Optimal* is taken to mean with respect to the entire decision variable space unless otherwise specified.

Definition 4 (*Pareto Optimal Set*). For a given MOP, $F(x)$, the *Pareto Optimal Set P* is defined as $P = \{x \in \Omega \mid \neg\exists x' \in \Omega \ F(x') \prec F(x)\}$.

Definition 5 (*Pareto Optimal Front*). For a given MOP, $F(x)$, and Pareto Optimal Set P, the *Pareto Front PF* is defined as $PF = \{\mathbf{u} = F(x) \mid x \in P\}$. A solution is said to be Pareto Optimal if it is not dominated by any other solutions in the search space. This is also termed as non-dominated solution.

Definition 6 (*The set A of alternatives*). This set consists of all Pareto-optimal points. It is often infinite, limited only by mathematical constraints. But, after the execution of a multi-objective search, it is reduced to a discrete approximation of the Pareto front, i.e., a discrete set of non-dominated solutions.

Definition 7 (*The set C of criteria*). Each considered criterion illustrates a viewpoint, according to which the alternatives solutions, based on their respective attributes, are evaluated and compared. This criterion is mathematically modeled by a function $c_i(\vec{x}) : A \to \Re$ that assigns a numerical value to each alternative, reflecting the decision-maker's preferences. In the continuous problems, each criterion is often derived from an objective function. However, it should be mentioned that additional criteria that do not correspond to any objective function can also be considered. In this paper, it is assumed that each criterion is derived from one objective function. Hence, the numbers of criteria and objective functions are considered to be identical.

3. MODIFIED MULTI-OBJECTIVE OPTIMIZATION ALGORITHM CSS-MOPSO

The introduced algorithm is based on two optimization methods, namely particle swarm optimization (PSO) and charged system search (CSS). The charged system search is a newly developed algorithm which is based on electrostatic and Newtonian mechanics laws. More details of these algorithms can be found in [2, 22, 23].

The important part in multi-objective particle swarm optimization is to determine the best global particle p_{gd} for each particle i of the population. In single objective PSO the global best particle is determined easily by selecting the particle which has the best position. Since in multi-objective optimization problems there is a set of Pareto optimal solutions as the optimum solutions, usually each particle of the population should select one of the Pareto-optimals as its global best particle, which is called the *global best guide.*

Selection of the global best leader for guiding a particle in a multi-objective particle swarm optimization problem is a challenging problem and different methods are introduced in literature. In the proposed method this problem is solved by usage of the CSS methodology, i.e., the best particles which are stored in an external archive create electrical field around themselves, and particles which are located in the domain of each leader are attracted toward them. Thus each particle flies in the search space by its personal experiment and the resultant force which is acted to this particle by global leaders (in such a case where there is no archive member worse particles, i.e. particles with lower fitness value, will be attracted by better particles). This concept is summarized in the following algorithm.

a) Main algorithm

The algorithm of CSS-MOPSO is as follows:

1) Initialize the particles:
   For i=0 to N (number of particles) initialize randomly *pos(i).*

2) Initialize the speed of each particle.
   For i=0 to N set $v(i)$=0
3) Evaluate each of the particles.
4) Initialize the personal memory of each particle.
   For i=0 to N set $Pbest(i)=pos(i)$.
5) Find the non-dominated solutions in this initial population and store them in the archive.
6) Since the maximum number of cycles has not been reached:
   a) Determine the Euclidean distance between all the particles in the population and also particles in the archive (in search space). Then normalize all distances to R_{allow}. This parameter is considered to overcome the effect of the range of search variables in the optimization process. This parameter is taken as 30 in this paper.
   b) Determine the charge magnitude for all the particles in the population and also the particles in the archive (see section 3b).
   c) Divide objective space into z parts and determine the location of all the particles in the population and archive according to this definition (see section 3c).
   d) Redistribute particles of the population every R cycle in different parts of the objective space (see section 3d).
   e) Initialize the **F** vector (resultant force vector acted on each particle).
   f) In each part, determine the resultant force exerted to each particle using the following expression:
   Case 1: There is at least one archive member in the part: In this case the archive member or members guide all the particles which are located in that part.

$$\mathbf{F}_j = q_j \sum_{i=1}^{k} \left(\frac{Q_i}{a^3} r_{ij} . i_1 + \frac{Q_i}{r_{ij}^2} . i_2 \right) \frac{\mathrm{x}_i - \mathrm{x}_j}{\left\| \mathrm{x}_i - \mathrm{x}_j \right\|} \qquad \begin{cases} i_1 = 1, i_2 = 0 & if \quad r_{ij} < a \\ i_1 = 0, i_2 = 1 & if \quad r_{ij} \geq a \end{cases} \tag{2}$$

   Where k is the number of archive members located in the same part as the particle j, and Q is the charge magnitude (fitness value) of archive particles, and q is the charge magnitude of an ordinary patricle in the population. In this paper, a is set to 1. This case is illustrated in Fig. 1a, in which, for example, the particles P_1 and P_2 are attracted by the archive members P'_1 and P'_2.
   Case 2: There is no archive member in the part: In this case, better particles in the part guide the worse particles, i.e. a particle i is attracted by particle j if and only if the charge magnitude of the particle j is higher than that of the particle i. In this case, the exerting force on each particle is equal to

$$\mathbf{F}_j = q_j \sum_{i=1}^{k} \left(\frac{q_i}{a^3} r_{ij} . i_1 + \frac{q_i}{r_{ij}^2} . i_2 \right) \frac{\mathrm{x}_i - \mathrm{x}_j}{\left\| \mathrm{x}_i - \mathrm{x}_j \right\|} \qquad \begin{cases} i_1 = 1, i_2 = 0 & if \quad r_{ij} < a \\ i_1 = 0, i_2 = 1 & if \quad r_{ij} \geq a \end{cases} \tag{3}$$

   q is the charge magnitude of an ordinary patricle in the population which is located in the same part. This pattern of particle absorbtion is illustrated in Fig. 1b, where, for example, the particle P_4 is attracted by P_1, P_2, P_3 and the particle P_2 is attracted by P_1 and P_3.
   g) Compute the new position and velocity of each particle using the following expressions:

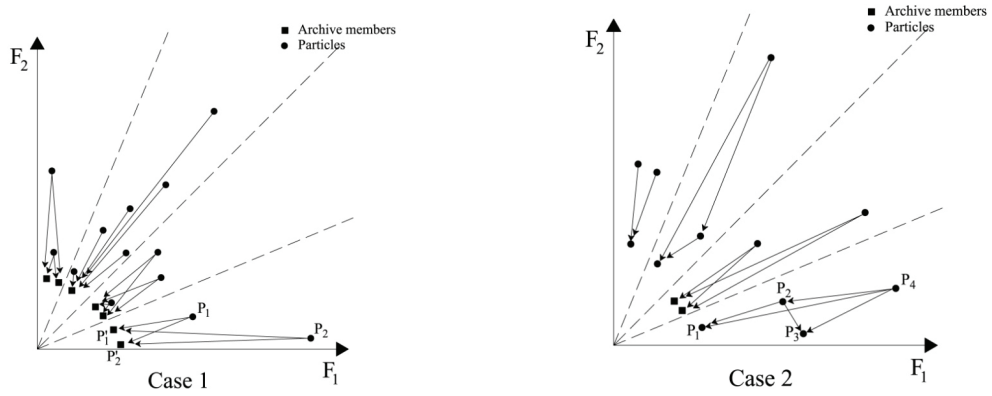

Fig. 1. Attraction strategies in CSS-MOPSO. (a) Case 1 (b) Case 2

$$\mathbf{x}_{j,new} = c_1.r_1.\frac{\mathbf{F}_j}{m_j} + c_2.r_2.(\mathbf{p}_j - \mathbf{x}_{j,old}(t)) + \omega.\mathbf{v}_{j,old} + \mathbf{x}_{j,old} \tag{4}$$

$$\mathbf{v}_{j,new} = \mathbf{x}_{j,new} - \mathbf{x}_{j,old} \tag{5}$$

Where c_1 and c_2 (similar to classical PSO) are user defined coefficients and r_1 and r_2 are two uniformly distributed random numbers in the range [0,1]. m_j is the mass of the jth particle which is equated to q_j in this paper, and $\mathbf{p}_j$ is the personal best of the particle j.

h) Control the velocity of the particles and limit them as recommended in [24], by considering $v_{\max} = x_{\max}$.

i) Apply the mutation operator to the new particles with predefined probability (see section 3.5).

j) Maintain the particles within the search space in case they go beyond their boundaries [14]. When a decision variable goes beyond its boundaries, two things happen: 1) the decision variable takes the value of its corresponding boundary (either the lower or the upper boundary) and 2) its velocity is multiplied by (-1) so that it searches in the opposite direction.

k) Evaluate each of the particles in the population.

l) Update the contents of the archive. This update consists of inserting all the currently non-dominated solutions into the archive and any dominated solutions from the archive are eliminated. Since the size of the archive is limited, we apply a secondary mechanism for keeping this limit: We adopt the concept of crowding distance [10] in order to fix the size of the external archive. First, when non-dominated solutions are inserted into the archive, the size of the archive is considered free, After updating the archive we proceed to update the crowding values of the set of leaders and sort them in descending order and eliminate as many leaders as necessary (from the end of the list) in order to avoid exceeding the allowable size of the leaders set (similar to the method which is proposed in [16]).

m) When the current position of the particle is better than the position contained in its personal memory, the particle's position is updated using *Pbest(i)=pos(i)*. The criterion to decide what position from memory should be retained is simply to apply Pareto dominance (i.e., a new particle replaces its value if such value is dominated by the new particle or if both are non-dominated with respect to each other.)

7) Increment the loop counter.

b) Charge magnitude of particles.

The charge magnitude of particles is related to their fitness value. The scheme which is employed in this paper for the particles in the population is similar to the fitness assignment algorithm introduced in

SPEA2 [11]. Here, each particle in the population is assigned a strength value, representing the number of solutions it dominates (P_t is the collection of members of population and $\overline{P}_t$ is the collection of members of the archive):

$$S(i) = \left| \left\{ j \,\middle|\, j \in P_t + \overline{P}_t \wedge i \succ j \right\} \right| \tag{6}$$

Where the sign $|.|$ denotes the cardinality of a set, + stands for multi-set union, and the symbol $\succ$ corresponds to the Pareto dominance relation. On the basis of the *S* values, the fitness *R*(*i*) of a typical individual *i* is calculated as

$$R(i) = \sum_{j \in P_t + \overline{P}_t,\, j \succ i} S(j) \tag{7}$$

According to this methodology, a particle with lower fitness is a better solution than the other solutions with higher fitness values. In order to reverse this pattern, the charge in the magnitude of each particle is determined as follows:

$$q_i = \frac{R_{\max} - R_i + \varepsilon}{R_{\max}} \tag{8}$$

$$R_{\max} = \max(R(i)) \quad \text{for all } i = 1,2,...,N \tag{9}$$

Where ε, in Eq. (8), is a small positive number to keep $q_i > 0$. According to this definition all members of the archive have zero charges and hence this method is not verified for classifying these members. In order to provide a measure for qualifying the diversity of the archive members, the following charge magnitude is introduced: First the *crowding distances* [10] for the archive members are calculated and then members are sorted in descending order according to their crowding distance, and the charge magnitude is determined as

$$Q_i = 1 + \frac{1}{\text{rank}(\text{archive}(i))} \tag{10}$$

where rank(archive(*i*)) is the rank of archive(i) in the sorted list.

c) Division of the space

In the proposed algorithm, there should be a criterion for determining the domain of each of the archive members, i.e., which particles can be attracted by each archive member. If all archive members are allowed to attract all particles in the population, then the particles will be spread in space and not converge to the Pareto front. Consequently, the attraction domain of each archive member should be limited by a space division strategy. The space division method employed here is the same as the formulation which is introduced in [15]. Using this method, each archive member will be able to attract those particles which are located in the same part (Fig. 2). According to this method, to each point with coordinates $(f_{1,i}, f_{2,i})$ a value σ_i is assigned so that all the points which are on the line $f_2 = \alpha . f_1$ have the same value of σ. Thus σ can be defined as

$$\sigma = \frac{f_1^2 - f_2^2}{f_1^2 + f_2^2} \tag{11}$$

In case the objectives are not in the same range, σ may be calculated as below for a two-objective optimization problem:

$$\sigma = \frac{m_1^2 - m_2^2}{m_1^2 + m_2^2} \quad , \quad m_1 = \frac{f_1 - f_{\min 1}}{f_{\max 1} - f_{\min 1}} \quad , \quad m_2 = \frac{f_2 - f_{\min 2}}{f_{\max 2} - f_{\min 2}} \tag{12}$$

Where $f_{\max 1}$ ($f_{\min 1}$), $f_{\max 2}$ ($f_{\min 2}$) are the maximum (minimum) values of the first and second objective of the particles in the population or archive, respectively.

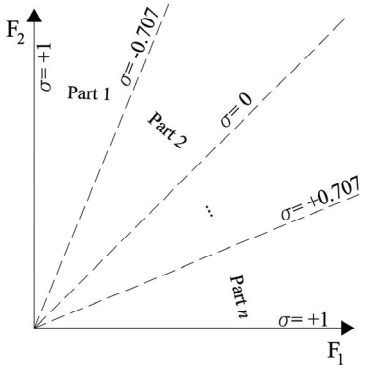

Fig. 2. Division of the objective space by assigning parameter σ to each particle

According to Eq. (11), all the points on the line $f_2 = \alpha . f_1$ have the same σ values as $\sigma_i = (1-\alpha^2)/(1-\alpha^2)$**.** Fig. 2 shows the values of σ for different lines. For more details, the reader can refer to [15]. In the proposed method, the objective space (according to this formulation) is divided into $\lfloor \sqrt{N} \rfloor$ different parts as shown in Fig. 2 (where $\lfloor . \rfloor$ represents the floor operator).

d) Particle redistribution

As mentioned in section 3.3, in the proposed algorithm the objective space is divided into some parts, and particles in each part are guided by the archive members of the same part. Consequently particles usually move in the same part as they are located. Using this strategy, it is possible that we face a part in the objective space with no sufficient particles in it, and consequently the search process in that part proceeds unacceptably. In order to avoid this problem, it is better to utilize a redistribution strategy that helps the main algorithm to cover all parts of the search space and Pareto front. The employed redistribution algorithm omits some particles of the most crowded part of the objective space and creates some new particles in the least crowded part. But this strategy should be employed in a way that does not disorder the search process of the main algorithm. Therefore, it is utilized once in every *R* cycles of the main algorithm. Reducing the parameter *R* has two effects on the search process. Firstly, it increases the required computational effort and, secondly, it may disturb the convergence process of the particles.

In the utilized redistribution algorithm, first the number of the particles and archive members located in each part are determined. Subsequently some particles with the lowest charge magnitude from the most crowded parts are eliminated and then some new particles are generated from the archive members or the best particles in the population which are located in sparse regions of the objective space. In our algorithm this process is continued until in each part there is at least $\lfloor N/(z) \rfloor$ particles (either from population or archive). Here, *z* is the number of parts and *N* is the number of particles in the population. This process is illustrated in Figs. 3a,b. In the first case, in which there are archive members in all parts, generating new particles is performed by using archive members (for example, particles P_1 and P_2 are eliminated and two new particles P_3 and P_4 are generated by using archive members P'_1). If there is more than one archive member in a part, for each particle generation one of them is selected randomly. In the second case, in which there is a part containing no archive members, new particle generation is performed by using the best particles of that part (particle with highest charge magnitude) and if there are several particles with the highest charge magnitude for each particle generation, one of them is selected randomly. As an

example, particles P_1 and P_2 are eliminated and two new particles P_3 and P_4 are generated by using particle P_5. It should be mentioned that the personal best ($\mathbf{p}_j$) of a newly generated particle is equated to itself, this means in the subsequent iteration only the first term of the Eq. (4) guides the particle. Creation of new particles is performed according to the formulation introduced in [18]. Obviously it is necessary to generate a higher number of new particles within the neighborhood of a particle than outside of the neighborhood. In order to achieve this goal, a set of equations is utilized as follows:

$$r_2 = \text{abs}(\text{Gaussian}(0, \frac{1}{9})) \tag{13}$$

$$ld = rld \times x_j^L \quad , \quad ud = rud \times x_j^U \tag{14}$$

$$\Delta d(r_2) = \begin{cases} (ud - ld) \times (2 \times (r_2)^2 + \dfrac{ld}{ud - ld}) & 0 \le r_2 \le 0.5 \\ (ud - ld) \times (1 - (2 \times (r_2 - 1)^2) + \dfrac{ld}{ud - ld} & 0.5 < r_2 \le 1 \end{cases} \tag{15}$$

$$x_{i,j} = x_{i,j} + \Delta d(r_2) \tag{16}$$

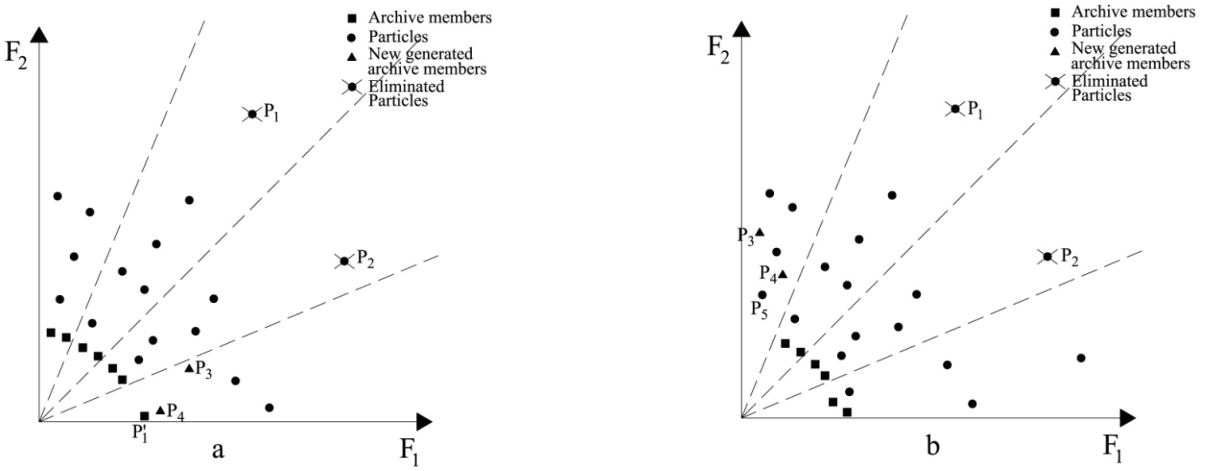

Fig. 3. Redistribution strategy (a) There is at least one archive member in each part (b) In a part with no archive members

Equation (16) is used to determine the additional distance from the selected particle corresponding to the decision space. The Δd is defined as a function of r_2. Several parameters are needed to compute Eq. (16). These parameters are r_2, *ld*, and *ud* which can be computed via Eq. (13) to Eq. (15), respectively. In Eq. (14), *ld* denotes the minimum additional distance. It is computed by multiplying the parameters *rld* and x_j^L, where *rld* is the user-defined lower bound ratio and x_j^L is the lower bound of the decision variable *x* in dimension *j*. Parameter *ud* denotes the maximum additional distance. Eq. (14) shows how the parameter *ud* is calculated, where *rud* is the user-defined upper bound ratio and x_j^U is the upper bound of the decision variable *x* in dimension *j*.

As presented in Eq. (13), the parameter r_2 is the absolute value of a random number in which the random number is drawn from the Gaussian distribution with zero mean and a variance of $1/9$. With the mean 0 and variance (σ^2), $1/9$, more random numbers will be generated near the lower end of the range, i.e., $[0,3\sigma/2]$, whereas fewer random numbers will be generated near the upper range, i.e., $(3\sigma/2,3\sigma]$. Once the Δd is computed, it is added to the decision variable of the selected particle *i* in dimension *j*, i.e., $x_{i,j}$.

e) Mutation operator

CSS and PSO are known to have a very high convergence speed [1, 2, 25, 26]. However, such convergence speed may be harmful in the context of multi-objective optimization, because a CSS-MOPSO based algorithm may converge to a false Pareto front (i.e., the equivalent of a local optimum in global optimization). This drawback of these optimization methods motivated the development of a mutation operator that tries to explore all of the search space. In our proposed algorithm, the mutation (turbulence) operator introduced by [15] is utilized. Mutation operator is applied to each particle with a predefined probability using the following formula:

$$x_j = x_j + R_T . x_j \tag{17}$$

Where R_T is a random value in [0,1].

f) Constraint handling

In order to handle the given constraints, a relatively simple scheme is implemented. Whenever two individuals are compared, first they are checked for constraint violation. If both are feasible, non-dominance is directly applied to decide which one is the winner. If one is feasible and the other is infeasible, the feasible dominates. If both are infeasible, then the one with the lowest amount of constraint violation dominates the other. This approach is identical to the one utilized in [10, 14] to handle the constraints.

4. MULTIPLE CRITERIA DECISION MAKING

Multiple criteria decision making refers to making preference decisions (e.g., evaluation, selection) on available alternatives in terms of multiple, usually conflicting, criteria, the case that always happens in the design of structures such as trusses. Many different approaches can be used for the decision making process [27]. Here, the newly developed method, called multi-criteria tournament decision making [28], is utilized. This algorithm is simple and has a small number of input parameters. The following two sections are devoted to this algorithm and its input parameters.

a) Multi-criteria tournament decision making

A simple method for multi-criteria decision making problem, so-called multi-criteria tournament decision making method (MTDM), is due to [28]. This method provides the ranking of alternatives from best to worst, according to the preferences of a human decision-maker (DM). It has another positive aspect that involves few input parameters, just the importance weight of each criterion. This method introduces a function R, capable of reflecting the DM global interests. In order to find this function, first each possible solution is compared with the others, considering only the *i*th-criterion. The pair-wise comparisons are implemented through the tournament function $T_i(a, A)$, which counts the ratio of times the alternative *a* wins the tournament against each other *b* solution from *A*. Hence, considering that *a* is a non-dominated point in the objective space, $T_i(a, A)$ can be stated as:

$$T_i(a, A) = \sum_{\forall b \in A, a \neq b} \frac{t_i(a,b)}{(|A|-1)} \tag{18}$$

Where in Eq. (18), $t_i(a,b)$ is defined as:

$$t_i(a,b) = \begin{cases} 1 & if \quad f_i(b) - f_i(a) > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{19}$$

The tournament function $T_i(a, A)$ assigns a score to each solution in Pareto front. The assigned score works as a performance measure, which provides a distinct ordering of the elements of A for each criterion. In order to generate the global ranking, taking into account all criteria and their respective weights w_i (priority factors), the scores are aggregated into the global ranking function R. The weighted geometric mean, given by Eq. (20) which is utilized by many different researchers, is considered as the aggregation function in this study. The advantage of this lies in the fact that it provides more harmonious solutions.

$$R(a) = \left(\prod_{i=1}^{m} T_i(a, A)^{w_i} \right)^{1/m} \tag{20}$$

The priority weights must be specified by the DM in accordance with the following conditions: $w_i > 0$ for i=1,…,m and $\sum_{i=1}^{m} w_i = 1$. The ranking index $R(a)$ gives an idea of how much each alternative is preferred to the others. In other words : if $R(a) > R(b)$, then a is preferred to b and when $R(a) = R(b)$, then a is indifferent to b. In view of that, MTD provides a complete preorder [6] of the available alternatives and the best ranked alternative can be considered as the favorite final solution.

b) Specifying the priority weights

As Internet technologies are adopted to facilitate communications among a wide variety of engineers with different cultural and educational backgrounds, different preference formats have been developed in literature which make it possible for DMs to express their opinion. Seven different formats are introduced in [26], all of which can be converted into a specific format, called *multiplicative preference*. After the use of preference aggregation and exploitation process, the weights of the criteria are obtained. By the determination of the weights, the best solution which satisfies most of the DMs is selected.

To make this clear, just the transformation function for the preference ordering is presented here. The process of weight determination can be summarized as follows:

(1) Each DM expresses his preferences using one of the seven possible formats: preference ordering, utility value, linguistic term, selected subset, fuzzy selected subset, normal preference relation, and multiplicative preference relation.

(2) The provided information is converted into multiplicative preference relations. A multiplicative preference relation p_{ij} on the criteria c_i and c_j reflects the ratio between the preference level of c_i over c_j, being understood as c_i is p_{ij}-times as important as c_j. The transformation functions proposed in [26] convert the original information into multiplicative preference relations that reflect the preference intensity using a ratio scale from "1" to "9". Here, "1" indicates indifference between two subjects while "9" indicates that one subject is absolutely preferred to the other. Given that o_i and o_j indicate, respectively, the position of the ith and the jth criteria in the ordering proposed by the DM, p_{ij} is given by:

$$p_{ij} = 9^{u_i - u_j} \quad i, j = 1, \ldots, m \tag{21}$$

Where $u_i = \dfrac{m - o_i}{m - 1}$ and $u_j = \dfrac{m - o_j}{m - 1}$

(3) The multiplicative preference relations are aggregated through the geometric mean:

$$w_i = \left(\prod_{j=1}^{m} p_{ij} \right)^{1/m}$$

(4) The weights are normalized in such way that $\sum_{i=1}^{m} w_i = 1$.

5. FRAMEWORK OF THE OPTIMAL DESIGN

In designing a new structure different tasks may constitute in selecting the final design, and these matters usually contradict each other. Because of this, different engineers who are engaged in a project may have completely different aspects and ideas. In order to solve this problem, in this study an optimal design framework is proposed which may relieve this problem. In the proposed procedure, first the given structural design problem is defined as a multi-objective design optimization, but selecting appropriate objectives is a crucial task which needs considerable investigation. The defined problem should be solved by a proper multi-objective optimization algorithm, and the point that should be considered is the number of fitness function evaluations needed to solve the MOP problem, because in structural design problems each fitness function evaluation involves analyzing a large-scale structure which needs considerable computational effort and correspondingly more required time. The proposed algorithm is aimed to solve this problem by reducing the required fitness function evaluation required for solving MOP. By the end of MOP solution Pareto front, which provides large amounts of information which help the designer to find the final design, is given to engineers (decision makers) to notify them regarding the preferences about different objectives defined in the problem. Then the final solution is selected by a proper algorithm in such a way that most of the DMs will be satisfied by the selected design. This framework is illustrated in Fig. 4.

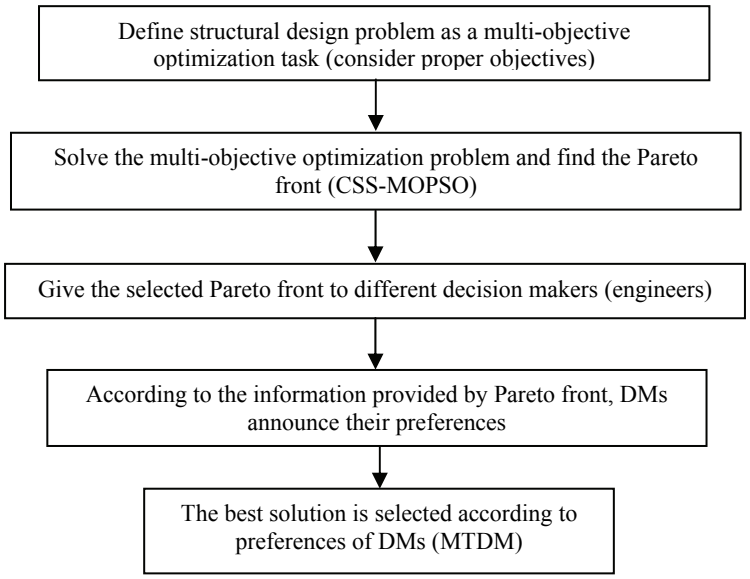

Fig. 4. Flowchart of the proposed optimal design framework

6. EXPERIMENTS RESULTS

In this section the experimental results are presented in order to clarify the performance of the proposed algorithm. For carrying out the necessary computations and evaluating the efficiency of the present algorithm, a computer program is developed using MATLAB language. In this part, two different truss structures, all having continuous design variables, previously defined by other researchers are considered. In order to show the ability of the proposed hybrid multi-objective optimization, the Pareto-front obtained by the CSS-MOPSO is compared to the original MOPSO [14], s-MOPSO [15], and NSGA-II [10].

In this paper, a real coded NSGA-II is run using a population size of 100, a crossover probability of 0.9 $(p_c = 0.9)$, tournament selection, a mutation rate of $1/u$ (where u is the number of decision

variables), and distribution indexes for crossover and mutation operators are taken as $\eta_c = 20$ and $\eta_m = 20$, respectively (as recommended in [10]). MOPSO used a population of 100 particles, an archive size of 100 particles, a mutation rate of 0.5, and 30 divisions for the adaptive grid. Also, s-MOPSO is run with a population of 100 particles, an archive size of 100 particles, a mutation probability of 0.05. And the parameters which are considered for CSS-MOPSO consist of C_1=1, C_2=2, R=15, *rld*=0.01, *rud*=0.05 and mutation probability =0.1. This algorithm employed an archive size of 100 and a population of 50 particles. For all examples presented in this paper, the number of fitness function evaluation (structural analysis) in multi-objective optimization phase is restricted to 30,000. Ten independent runs are performed for each example to obtain the statistical information. In order to compare the results for different MOP problems, usually different performance metrics are utilized in the literature [21]. In this study three performance metrics are used as follows:

Generational distance: GD is a measure of the distance between the true (PF_{true}) and generated Pareto front (PF_{known}). This metric of individual distance representing the distance is given by

$$\mathrm{GD} = \frac{1}{n_{pf}} \left(\sum_{i=1}^{n_{pf}} d_i^2 \right)^{1/2} \tag{22}$$

Where n_{pf} is the number of members in PF_{known} and d_i is the Euclidean distance between the member i in PF_{known} and its nearest member in PF_{true}. A smaller value of GD implies better convergence. As we do not know the PF_{true} of the test instances, we use an approximation of the PF_{true}. The approximation of PF_{true} is obtained from all non-dominated solutions found in all the runs of the four considered algorithms.

Spacing: The metric of spacing (S) gives an indication of how evenly the solutions are distributed along the discovered Pareto-front:

$$\mathrm{S} = \left[\frac{1}{n_{pf}} \sum_{i=1}^{n_{pf}} (d_i - \overline{d})^2 \right]^{1/2} \quad \text{where} \quad \overline{d} = \frac{1}{n_{pf}} \sum_{i=1}^{n_{pf}} d_i \tag{23}$$

Where n_{pf} is the number of members in PF_{known} and d_i is the Euclidean distance (in the objective space) between the member i in PF_{known} and its nearest member in PF_{known}. A smaller value of S implies a more uniform distribution of the solutions in PF_{known}.

Maximum Spread: The metric of maximum spread (MS) measures how "well" the PF_{true} is covered by the PF_{known} through hyper-boxes formed by the extreme function values observed in the PF_{true} and PF_{known}. It is defined as

$$\mathrm{MS} = \left[\frac{1}{m} \sum_{i=1}^{m} \left[\frac{\min(f_i^{\max}, F_i^{\max}) - \max(f_i^{\min} - F_i^{\min})}{F_i^{\max} - F_i^{\min}} \right]^2 \right]^{1/2} \tag{24}$$

Where m is the number of objectives, $f_i^{\max}$ and $f_i^{\min}$ are the maximum and minimum of the ith objective in PF_{known}, respectively, and $F_i^{\max}$ and $F_i^{\min}$ are the maximum and minimum of the ith objective in PF_{true}, respectively. A larger value of MS implies a better spread of solutions. In this study $F_i^{\max}$ and $F_i^{\min}$ are considered as the maximum and minimum of the ith objective in all the Pareto fronts obtained by various algorithms.

a) A 25-bar truss structure

The first structure considered is another famous 25-bar truss as shown in Fig. 5. Again, the problem is to find the cross-sectional area of members such that the total structural weight and the displacement in Y-

direction at node 1 are minimized concurrently. The structure includes 25 members, which are divided into eight groups, as follows: (1) A_1, (2) A_2–A_5, (3) A_6–A_9, (4) A_{10}–A_{11}, (5) A_{12}–A_{13}, (6) A_{14}–A_{17}, (7) A_{18}–A_{21} and (8) A_{22}–A_{25}. The applied load to this structure is: $F_{x(1)}$=4.45(kN), $F_{y(1)}$=–44.5(kN), $F_{z(1)}$=–44.5(kN), $F_{y(2)}$=–44.5(kN), $F_{z(2)}$=–44.5(kN), $F_{x(3)}$=2.25(kN) and $F_{x(6)}$=2.67 (kN).

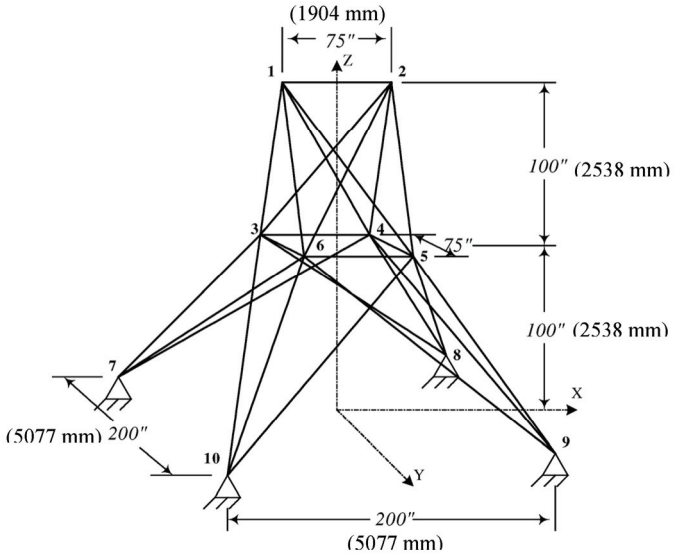

Fig. 5. A 25-bar space truss structure

The upper and lower bounds for the cross sections of each truss element are 64.45 mm^2 (0.1 in^2) and 2191.47 mm^2 (3.4 in^2), respectively. Material properties are taken as modulus of elasticity $E = 68.97\,\text{kN/mm}^2$ $(1\times10^4$ ksi) and weight density $\rho = 2.714\text{E-}8\ \text{kN/mm}^2$ (0.1lb/in^2). Constraints on the truss limit the principal stress σ_j in each element to the maximum allowable stress, $\sigma_a = \pm 0.27584\,\text{kN/mm}^2$ $(\pm40$ ksi).

The performance metrics are calculated for this problem and the results are presented in Table 1. It can be seen that the CSS-MOPSO outperforms or competes with other methods with respect to the three performance metrics. The computational time for the four algorithms illustrates that the better performance of CSS-MOPSO is obtained at the expense of some additional computational time, however this time is much less than the time required for NSGA-II. Also, the two extreme objective values obtained in 10 runs of algorithms are compared in Table 2.

Table 1. Performance metrics for 25 bar truss structure

Optimization method	NSGA-II	MOPSO	s-MOPSO	CSS-MOPSO
GD (Mean)	0.2396	0.0912	0.2021	0.0893
GD(SD)	0.1741	0.0057	0.1828	0.0117
S(Mean)	6.7197	4.9832	6.5673	5.7409
S(SD)	0.4744	1.4514	0.2594	0.2814
MS(Mean)	0.9977	0.8425	0.9490	0.9525
MS(SD)	0.0020	0.0354	0.0098	0.0241
Time (min)	3.41	0.77	1.05	1.98

Table 2. Comparison of the results for 25 bar truss structure

Optimization method	CSS-MOPSO	s-MOPSO	MOPSO	NSGA-II
Extreme obtained values (mm, kN)	[5.8437,4.8111] [62.9807, 0.3440]	[5.8437, 4.8917] [62.7832, 0.3239]	[5.8791,4.4836] [60.3942, 0.3642]	[5.8437,4.8297] [64.5579, 0.3141]
Number of fitness function evaluation	30,000	30,000	30,000	30,000

It should be stated that for the optimal design of a structure, e.g., a truss, sometimes the objectives are dictated by practical matters. An example which can be mentioned for such a case is truss structure where the objectives are the weight of truss and displacement of a specific node.

On the other hand, for structures in which objectives are not clear and different choices can be considered, the selection of the best objectives is a crucial task which requires experience or engineering sense. This problem can be the subject of further researches. According to the presented framework, after finding the Pareto-front the next step is to ask DMs to notify their preferences by considering all information that is integrated in the Pareto-front.

Then Eq. (21) is utilized to calculate the priority weights for all the criteria (objectives). Many different scenarios are possible for a considered problem. For example, these scenarios can be as follow:

Scenario A: The first criterion (objective) is more important: e.g. $(w_1, w_2) = (0.6, 0.4)$

Scenario B: The first criterion (objective) is as important as the second criterion: $(w_1, w_2) = (0.5, 0.5)$

Scenario C: The second criterion (objective) is more important: e.g. $(w_1, w_2) = (0.4, 0.6)$

The selected solutions corresponding to each considered scenario are indicated in Fig. 6, and in Table 3 the best solutions for different scenarios are presented.

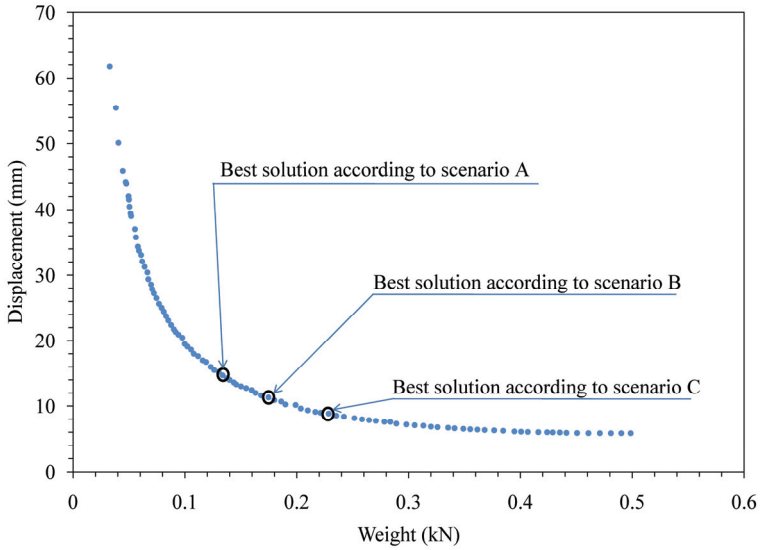

Fig. 6. Best solution according to three different scenarios for the 25-bar truss

Table 3. Best selected solutions

Scenario A	Scenario B	Scenario C
1.189 (kN)	1.548 (kN)	2.036 (kN)
16.7307 (mm)	12.7422 (mm)	9.6144 (mm)

b) A 56-bar truss structure

The second example is a 56-bar space truss [9] with members categorized in three groups as shown in Fig. 7. Joint 1 is loaded with 4 kN (899.24 lb) in the Y-direction and 30 kN (6744.267 lb) in the Z-direction, while the remaining free nodes are loaded with 4 kN (899.24 lb) in the Y-direction and 10 kN (2248.09 lb) in the Z-direction.

The vertical displacements of joints 4, 5, 6, 12, 13 and 14 are restricted to 4 mm (0.158 in), while the displacement of joint 8 in the Y-direction is limited to 2 mm (0.079 in). The modulus of elasticity and the minimum and maximum member cross-sectional areas are taken as 210 kN/mm^2 (3.05×10^4 ksi), 200 mm^2 (0.31 in^2) and 2000 mm^2 (3.1 in^2), respectively. The total structural volume $F_1(x)$, and the displacement at node 1, $F_2(x)$, have to be minimized simultaneously. Objective functions are

$$\min\begin{cases} F_1(x) = \sum_{i=1}^{56} A_i l_i \\ F_2(x) = \sqrt{\delta_{1x}^2 + \delta_{1y}^2 + \delta_{1z}^2} \end{cases} \tag{25}$$

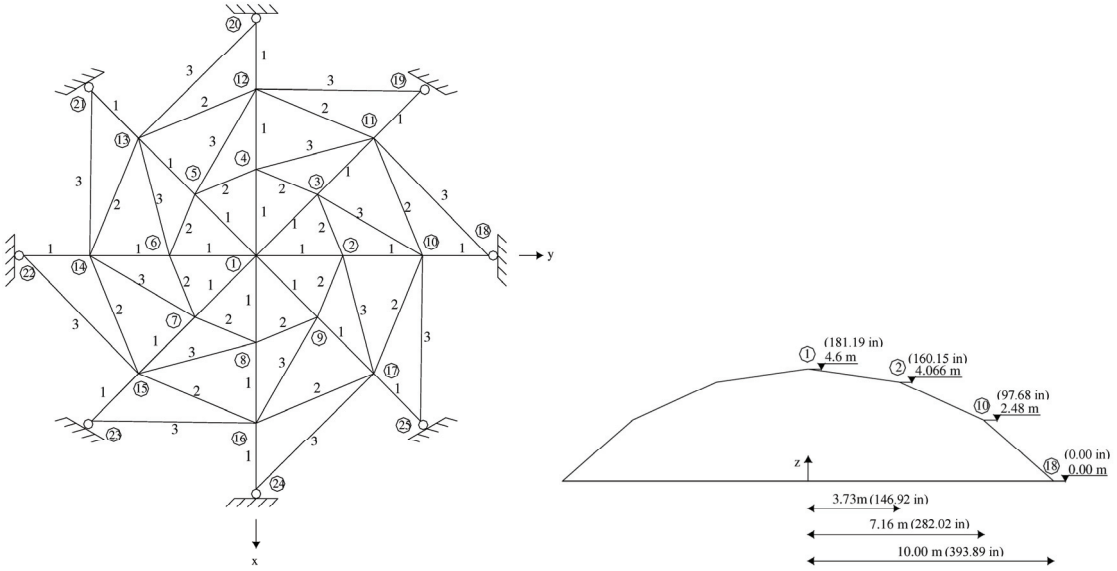

Fig. 7. A 56-bar space truss

This example is solved by the four algorithms and then the performance metrics are computed, as illustrated in Table 4. Considering the three metrics and the required computational time, it can be concluded that the CSS-MOPSO performs best among all the utilized algorithms. In addition, the two extreme objective values, obtained in 10 runs of algorithms are compared in Table 5. It can be observed that the modification performed on MOPSO, in contrast to the improvement in its performance, has increased its computational time. The time required by NSGA-II is much more than that of the other methods.

Table 4. The performance metrics using four algorithms

Optimization method	NSGA-II	MOPSO	s-MOPSO	CSS-MOPSO
GD (Mean)	9625.2015	9310.8786	7206.1220	8903.716037
GD(SD)	1502.9349	1589.7693	1144.5928	891.4767969
S(Mean)	1721882.31	1407913.94	1471693.91	1362344.87
S(SD)	194696.30	89408.13	41408.88	25255.80
MS(Mean)	0.9879	0.9335	0.9820	0.994168136
Time (min)	3.74	1.28	1.48	1.36

Table 5. Comparison of the results for 56 bar truss structure

Optimization method	CSS-MOPSO	s-MOPSO	MOPSO	NSGA-II
Extreme obtained values (mm, mm3)	[2.2148, 402923368.6] [7.5495, 120812690.1]	[2.2137, 402417631.6] [7.4721, 120151168.8]	[2.2154,403070300.4] [7.0825, 123191518.1]	[2.2137, 402403612.4] [7.4883,119960278.7]
Number of fitness function evaluation	30,000	30,000	30,000	30,000

The process of decision making and finding the best solution is performed completely similar to the previous example. In this example, in order to show the wide range of possible solutions, five different scenarios are considered. The results are aggregated in Table 6. The selected solutions corresponding to each considered scenario are provided in Fig. 8.

Table 6. Different possible scenarios for the 56-bar truss with corresponding solutions

Scenario	Importance of criteria	Possible priority weights	Selected solution by MTDM (mm, mm^3)
A	$c_1>>c_2$	[0.9,0.1]	[6.1144,134900811.2]
B	$c_1>c_2$	[0.7,0.3]	[4.4740,166100766.2]
C	$c_1\sim c_2$	[0.5,0.5]	[3.2959,208951138.7]
D	$c_1<c_2$	[0.3,0.7]	[2.5679,267415709.3]
E	$c_1<<c_2$	[0.1,0.9]	[2.2709,353607163.4]

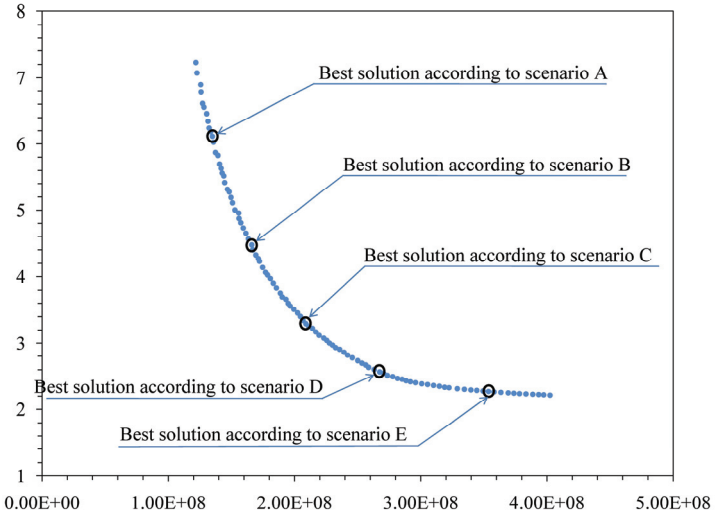

Fig. 8. Best solution according to three different scenarios for the 56-bar truss

7. CONCLUDING REMARKS

In traditional structural optimal design algorithms, usually one objective, e.g. weight, is selected to be minimized, however, in practice many different criteria are involved in designing a structure. In this paper, a new hybrid algorithm for optimal design of structures is proposed and applied to truss structures. In this method the problem of optimization is defined as a multi-objective optimization task, and by applying a modified multi-objective particle swarm optimization, the Pareto-front is obtained. Then all engineers who should make a decision express their preferences about different criteria (objectives or other independent criteria). By aggregating different ideas, the final solution is selected by an algorithm called MTDM.

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

REFERENCES

1. Kaveh, A. & Talatahari, S. (2010). Charged system search for optimum grillage systems design using the LRFD-AISC code. *Journal of Constructional Steel Research*, Vol. 66, pp. 767-771.
2. Kaveh, A. & Talatahari, S. (2010). Optimal design of skeletal structures via the charged system search algorithm. *Structural Multidisciplinary Optimization*, Vol. 37: pp. 893-911.
3. Kaveh, A. & Shakouri, A. (2010). Harmony search algorithm for optimum design of slab formwork, *Iranian Journal of Science and Technology, Transaction B: Engineering*, Vol. 34, pp. 335-351.
4. Kaveh, A. & Malakouti Rad, S. (2010). Hybrid genetic algorithm and particle swarm optimization for the force method-based simultaneous analysis and design, *Iranian Journal of Science and Technology*, *Transaction B*: *Engineering*, Vol. 34, pp. 15-34.

5. Kaveh, A. & Masoudi, M. S. (2011). Cost optimization of a composite floor system using ant colony system, *Iranian Journal of Science and Technology*, *Transaction B*: *Engineering*, to appear.
6. Deb, K. (2001). *Multi objective optimization using evolutionary algorithms*. Chichester, Wiley, U.K.
7. Coello Coello, C. A. & Christiansen, A. D. (2000). Multi objective optimization of trusses using genetic algorithms. *Computers and Structures*, Vol. 75, pp. 647–660.
8. Luh, G. C. & Chueh, C. H. (2004). Multi-objective optimal design of truss structure with immune algorithm. *Computers and Structures*, Vol. 82, pp. 829–844.
9. Kelesoglu, O. (2007). Fuzzy multi objective optimization of truss-structures using genetic algorithm. *Advances in Engineering Software*, Vol. 38, pp. 717–721.
10. Deb, K., Pratap, A., Agarwal, S. & Meyarivan, T. (2002). A fast and elitist multi objective genetic algorithm: NSGA-II. *IEEE Transaction of Evolutionary Computation*, Vol. 6, pp. 182–197.
11. Zitzler, E., Laumanns, M. & Thiele, L. (2001). SPEA2: improving the strength Pareto evolutionary algorithm. Swiss Federal Institute Technology: Zurich, Switzerland.
12. Knowles, J. D. & Corne, D. W. (2000). Approximating the nondominated front using the Pareto archived evolution strategy. *Evolutionary Computation*, Vol. 8, pp. 149–172.
13. Coello Coello, C. A. & Lechuga, M. S. (2002). MOPSO: A proposal for multiple objective particle swarm optimization. *Proceeding Congress on Evolutionary Computation (CEC'2002)*, Vol. 1, pp. 1051–1056.
14. Coello Coello, C. A., Pulido, G. T. & Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization. *IEEE Transaction on Evolutionary Computation*, Vol. 8, No. 3, pp. 256–279.
15. Mostaghim, S. & Teich, J. (2003). Strategies for finding good local guides in multi objective particle swarm optimization (MOPSO). *In Proceedings of the IEEE Swarm Intelligence Symposium*, pp. 26–33.
16. Sierra, M. R. & Coello Coello, C. A. (2005). Improving PSO-based multi objective optimization using crowding, mutation and ε-dominance. *Proceeding of Evolutionary Multi-Criterion Optimization Conference*, Guanajuato, Mexico: pp. 505–519.
17. Fieldsend, J. E. & Singh, S. (2002). A Multi-objective algorithm based upon particle swarm optimization, an Efficient Data Structure and Turbulence. *Proceedings of the 2002 U.K. Workshop on Computational Intelligence*, Birmingham, UK, pp. 37–44.
18. Gong, D. W., Zhang, Y. & Zhang, J. H. (2005). Multi-objective particle swarm optimization based on minimal particle angle. *ICIC 2005*, Part I, LNCS 3644, pp. 571 – 580.
19. Hu, X. & Eberhart, R. (2002) Multi objective optimization using dynamic neighborhood particle swarm optimization. *In Congress on Evolutionary Computation (CEC'2002). Volume 2, Piscataway*, New Jersey, IEEE Service Center: pp. 1677–1681.
20. Sierra, M. R. & Coello Coello, C. A. (2006). Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art. *International Journal of Computational Intelligence Research*, Vol. 2, pp. 287-308.
21. Coello Coello, C. A., Van Veldhuizen, D. A. & Lamont, G. B. (2002). Evolutionary Algorithms for Solving Multi-objective Problems. Kluwer Academic Publishers, New York, USA.
22. Kennedy, J. & Eberhart, R. C. (1995). Particle swarm optimization, *Proceedings of the IEEE International Conference on Neural Networks*. Piscataway: IEEE: pp. 1942-1948.
23. Kaveh, A. & Talatahari, S. (2010). A novel heuristic optimization method: Charged system search. *Acta Mechanica,* Vol. 213, pp. 267-286.
24. Qi, K., Lei, W. & Qidi, W. (2007). A novel self-organizing particle swarm optimization based on gravitation field model. *Proceeding of the American Control Conference*, New York: pp. 528–533.

25. Leong, W. F. & Yen, G. G. (2008). PSO-based multi objective optimization with dynamic population size and adaptive local archives. *IEEE Transactions on Systems, Man & Cybernetics* Vol. 38, pp. 1270-1293.
26. Zhang, Q., Chen, J. C. H. & Chong, P. P. (2004). Decision consolidation: Criteria weight determination using multiple preference formats. *Decision Support Systems* Vol. 38, pp. 247-258.
27. Fishburn, P. C. (1970). *Utility Theory for Decision Making*. Wiley, New York, USA.
28. Parreiras, R. O., Maciel, J. H. R. D. & Vasconcelos, J. A. (2005) Decision Making in Multiobjective Optimization Problems. ISE Book Series on Real Word Multi-Objective System Engineering: pp. 1-20.