Knowledge-Based Systems 161 (2018) 398–414

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Multiobjective adaptive symbiotic organisms search for truss optimization problems

Ghanshyam G. Tejani[a,*], Nantiwat Pholdee[b], Sujin Bureerat[b], Doddy Prayogo[c]

[a] Department of Mechanical Engineering, GSFC University, Vadodara, Gujarat, India
[b] Department of Mechanical Engineering, Faculty of Engineering, Khon Kaen University, Thailand
[c] Department of Civil Engineering, Petra Christian University, Jalan Siwalankerto 121-131, Surabaya 60236, Indonesia

ARTICLE INFO

Article history:
Received 11 February 2018
Revised 5 June 2018
Accepted 5 August 2018
Available online 11 August 2018

Keywords:
Meta-heuristic
Structural optimization
Archive technique
Discrete design
Constraint problem

ABSTRACT

This paper presents a multiobjective adaptive symbiotic organisms search (MOASOS) and its two-archive technique for solving truss optimization problems. The SOS algorithm considers the symbiotic relationship among various species, such as mutualism, commensalism, and parasitism, to live in nature. The heuristic characteristics of the mutualism phase permits the search to jump into not visited sections (named an exploration) and allows a local search of visited sections (named an exploitation) of the search region. As search progresses, a good balance between an exploration and exploitation has a greater impact on the solutions. Thus, adaptive control is now incorporated to propose MOASOS. In addition, two-archive approach is applied in MOASOS to maintain population diversity which is a major issue in multiobjective meta-heuristics.

For the design problems, minimization of the truss' mass and maximization of nodal displacement are objectives whereas elemental stress and discrete cross-sectional areas are assumed to be behaviour and side constraints respectively. The usefulness of these methods to solve complex problems is validated by five truss problems (i.e. 10-bar truss, 25-bar truss, 60-bar truss, 72-bar truss, and 942-bar truss) with discrete design variables. The results of the proposed algorithms have demonstrated that adaptive control is able to provide a better and competitive solutions when compared against the previous studies.

1. Introduction

Over the last few decades, investigation on truss design optimization has been one of the main issues in structural design. A truss is a special kind of engineering structure constructed by using a number of 2-node members interconnected with revolute joints. This implies that the structural elements will ideally experience only tension or compression. There have been many research aspects for truss optimization since it was first studied. Truss optimization problems can be categorized as topology, shape, and sizing optimization depending on the design variables assigned. Sizing design variables are usually assigned to find optimal cross-sectional areas of truss members whereas shape design variables will alter the nodal positions from their original places. Topological design variables, on the other hand, are set to find an initial structural layout or configuration given that a designer may not know it a-priory. The combination of two or even three types of design variables for one optimization runs can be achieved. The combination of shape and sizing variables leads to a shape and sizing optimization problem which has been proven to give better design results than merely performing optimal truss sizing. The simultaneous operation of topology, shape, and sizing design for one run, sometimes called automated design, is arguably the best design strategy. The design problems can also be labelled as small- medium- and large-scale optimization problems depending on the total number of design variables being used.

Based on the number of objective functions, truss design problem can be single-objective or multiobjective optimization. A typical truss design problem is posed to minimize its mass or weight subject to stress, displacement, and buckling constraints. Nevertheless, some recent work has been devoted to mass minimization with natural frequency constraints. For multiobjective cases, the first objective is usually structural cost or weight while other objectives may be used to measure structural performance. As already known that mass minimization subject to structural safety constraints always results in a design solution that is in the boarder-line of safety and failure, it is therefore not practical without further modification. The use of factors of safety is one of the ways to alleviate such a problem. On the other hand, designers

* Corresponding author.
E-mail addresses: p.shyam23@gmail.com (G.G. Tejani), nantiwat@kku.ac.th (N. Pholdee), sujbur@kku.ac.th (S. Bureerat), prayogo@petra.ac.id (D. Prayogo).

https://doi.org/10.1016/j.knosys.2018.08.005
0950-7051/© 2018 Elsevier B.V. All rights reserved.

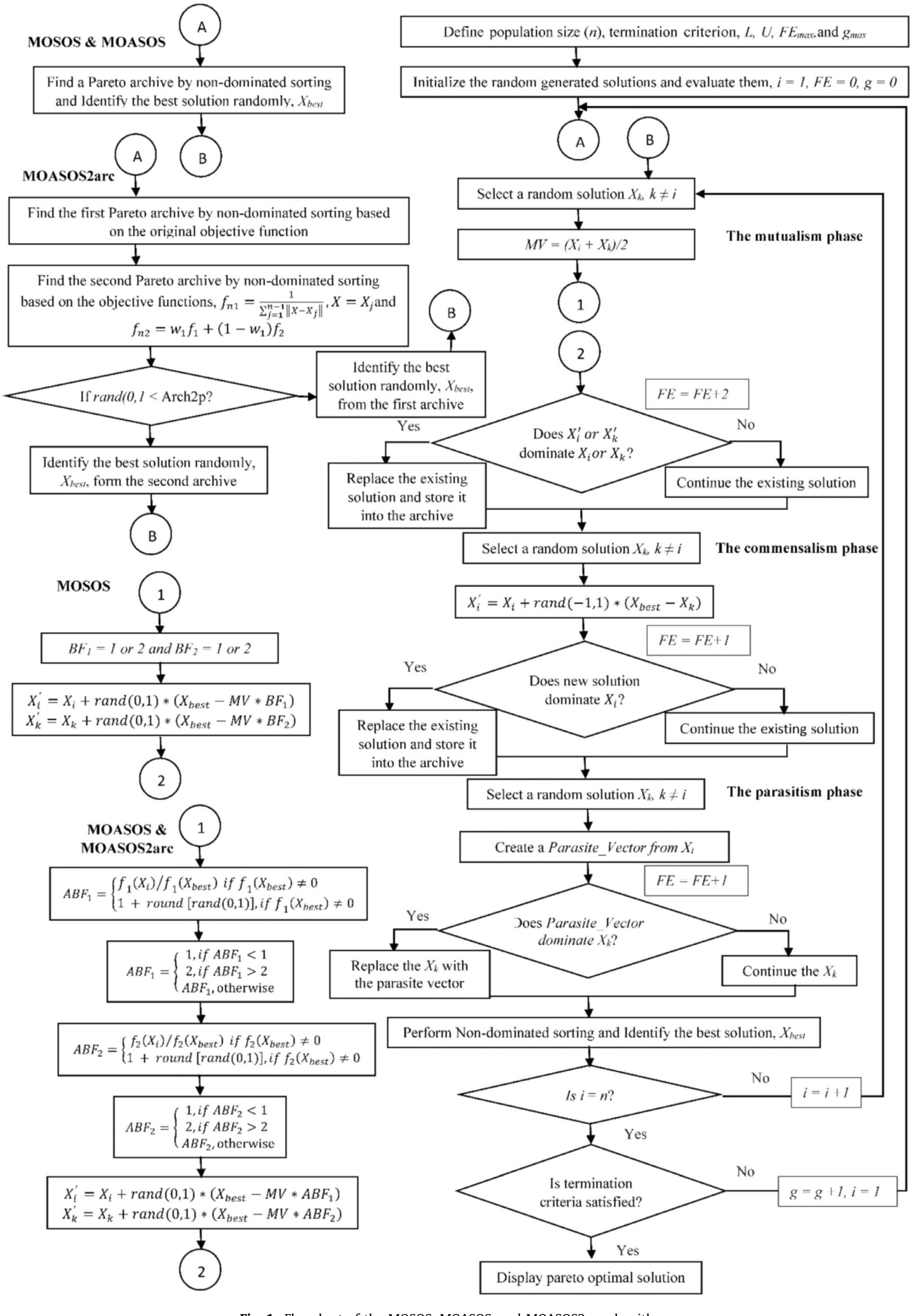
MOSOS & MOASOS
A
Find a Pareto archive by non-dominated sorting and Identify the best solution randomly, X_{best}
B
MOASOS2arc
A
Find the first Pareto archive by non-dominated sorting based on the original objective function
Find the second Pareto archive by non-dominated sorting based on the objective functions, $f_{n1} = \frac{1}{\sum_{j=1}^{n-1}\|X-X_j\|}$, $X = X_j$ and $f_{n2} = w_1 f_1 + (1-w_1) f_2$
If $rand(0,1 <$ Arch2p?
Identify the best solution randomly, X_{best}, from the first archive
Identify the best solution randomly, X_{best}, form the second archive
MOSOS
1
$BF_1 = 1$ or 2 and $BF_2 = 1$ or 2
$X_i' = X_i + rand(0,1) * (X_{best} - MV * BF_1)$
$X_k' = X_k + rand(0,1) * (X_{best} - MV * BF_2)$
2
MOASOS & MOASOS2arc
$ABF_1 = \begin{cases} f_1(X_i)/f_1(X_{best}) \; if \; f_1(X_{best}) \neq 0 \\ 1 + round\,[rand(0,1)], if \; f_1(X_{best}) \neq 0 \end{cases}$
$ABF_1 = \begin{cases} 1, if \; ABF_1 < 1 \\ 2, if \; ABF_1 > 2 \\ ABF_1, \text{otherwise} \end{cases}$
$ABF_2 = \begin{cases} f_2(X_i)/f_2(X_{best}) \; if \; f_2(X_{best}) \neq 0 \\ 1 + round\,[rand(0,1)], if \; f_2(X_{best}) \neq 0 \end{cases}$
$ABF_2 = \begin{cases} 1, if \; ABF_2 < 1 \\ 2, if \; ABF_2 > 2 \\ ABF_2, \text{otherwise} \end{cases}$
$X_i' = X_i + rand(0,1) * (X_{best} - MV * ABF_1)$
$X_k' = X_k + rand(0,1) * (X_{best} - MV * ABF_2)$
Define population size (n), termination criterion, L, U, FE_{max}, and g_{max}
Initialize the random generated solutions and evaluate them, $i = 1$, $FE = 0$, $g = 0$
Select a random solution X_k, $k \neq i$
$MV = (X_i + X_k)/2$
The mutualism phase
$FE = FE+2$
Does X_i' or X_k' dominate X_i or X_k?
Yes
No
Replace the existing solution and store it into the archive
Continue the existing solution
Select a random solution X_k, $k \neq i$
The commensalism phase
$X_i' = X_i + rand(-1,1) * (X_{best} - X_k)$
$FE = FE+1$
Does new solution dominate X_i?
Replace the existing solution and store it into the archive
Continue the existing solution
Select a random solution X_k, $k \neq i$
The parasitism phase
Create a $Parasite\_Vector$ from X_i
$FE = FE+1$
Does $Parasite\_Vector$ dominate X_k?
Replace the X_k with the parasite vector
Continue the X_k
Perform Non-dominated sorting and Identify the best solution, X_{best}
Is $i = n$?
$i = i+1$
Is termination criteria satisfied?
$g = g+1$, $i = 1$
Display pareto optimal solution

Fig. 1. Flowchart of the MOSOS, MOASOS, and MOASOS2arc algorithms.

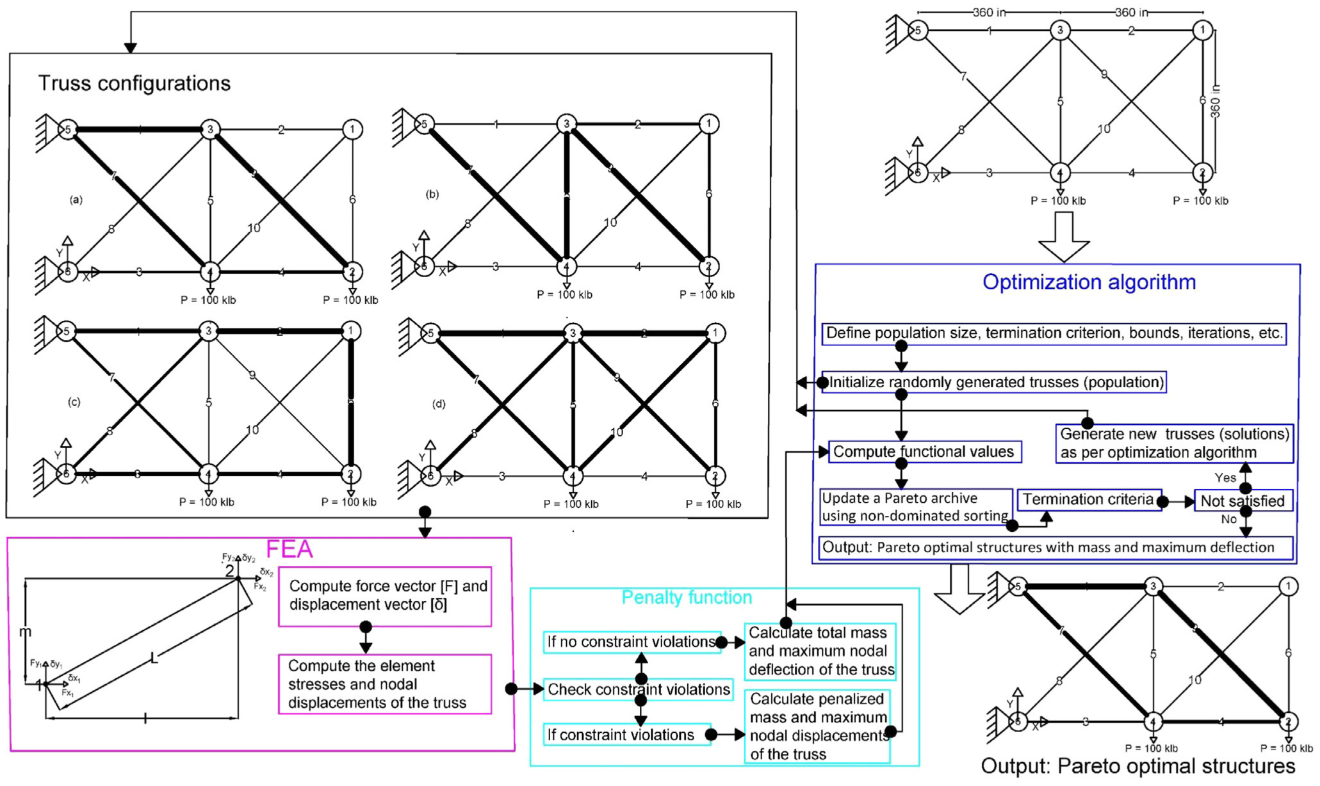
Truss configurations
(a)
(b)
(c)
(d)
P = 100 klb
FEA
Compute force vector [F] and displacement vector [δ]
Compute the element stresses and nodal displacements of the truss
Penalty function
If no constraint violations
Check constraint violations
If constraint violations
Calculate total mass and maximum nodal deflection of the truss
Calculate penalized mass and maximum nodal displacements of the truss
Optimization algorithm
Define population size, termination criterion, bounds, iterations, etc.
Initialize randomly generated trusses (population)
Compute functional values
Generate new trusses (solutions) as per optimization algorithm
Update a Pareto archive using non-dominated sorting
Termination criteria
Not satisfied
Yes
No
Output: Pareto optimal structures with mass and maximum deflection
Output: Pareto optimal structures
360 in

Fig. 2. Schematic diagram of proposed methodology.

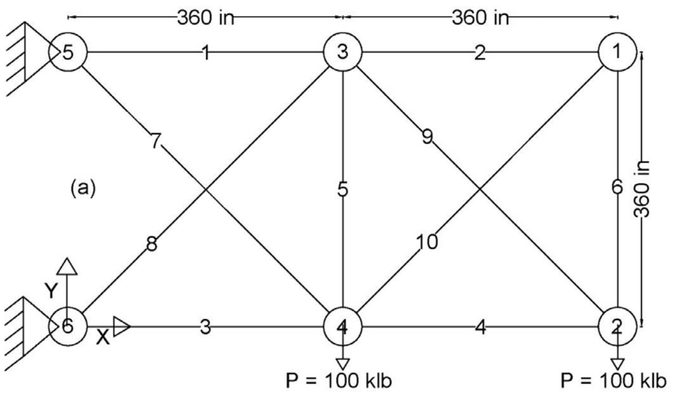

Fig. 3. 10-bar truss.

may add another objective function to a design problem where such an objective function should be an indicator for structural reliability e.g. displacement minimization, compliance minimization, natural frequency maximization, frequency response function minimization, etc. With two or more objectives, the design problems become more difficult and has a set of countless optimal results.

Another aspect of truss optimization is development of an optimizer or optimization method. Normally, there are two types of optimisers used for truss optimization, gradient-based methods and meta-heuristics (MHs). The former has been proven effective but, with its complication, derivative dependence, and other limitations, it is less popular. The latter is probably the most popularly used algorithm for solving truss optimization. MHs are advantageous in that they are simple to code, create, and understand. The methods are more flexible, and designers can modify them or even introduce a new algorithm or concept. Since they are derivative-free, almost any kind of design problems can be solved by using such optimizers. Most of them are efficient for solving global optimization. In cases of truss design, a feasible region may be highly non-convex such as some cases of natural frequency constraints, therefore, using MHs can be a better choice. Nevertheless, it is always recognized that the convergence rate of MHs is poor and one run requires a huge number of function evaluations. Thus, improving the performance of MHs is always an issue.

For single-objective MHs, some recently developed methods are, for example, Vortex Search Algorithm [12], Search Group Algorithm (Gonçalves, [13]), and sine-cosine optimization [16]. Other types of recent development for MHs are self-adaptive algorithms such as LSHADE [26]. For multiobjective optimization which is often called multiobjective evolutionary algorithms (MOEAs), there have been numerous MH optimisers proposed in literature. Non-dominated sorting genetic algorithm which now have had three versions is arguably the best-known algorithm [8,9,20]. The others well known algorithms include multi-objective immune algorithm [15], Hybrid Multi-Objective Particle Swarm Optimization [14], MOEA based on Decomposition [28], Multi-Objective Particle Swarm Optimization (Reyes-Sierra and Coello Coello, [6]), and SPEA (Zitzler, [30]). One of the recently proposed MOEAs is multi-objective symbiotic organisms search (MSOS) [5,18,24,25], which is found to be an efficient optimizer.

SOS has superior performance over a number of algorithms available since it was first published and has been in solving many optimization problems [5]. The SOS algorithm has been examined for constrained and unconstrained benchmark engineering problems and proved to be better performer with other MHs [4,5]. Cheng et al. [5] proposed a discrete SOS algorithm to optimize multiple-resources levelling problems. Abdullahi et al. [1] used a discrete SOS in efficient task scheduling in cloud computing. Tran et al. [24,25] used MOSOS to do optimization of time-cost-labour utilization problems in construction projects. Panda and Pani [18] used an adaptive penalty function in MOSOS to handle

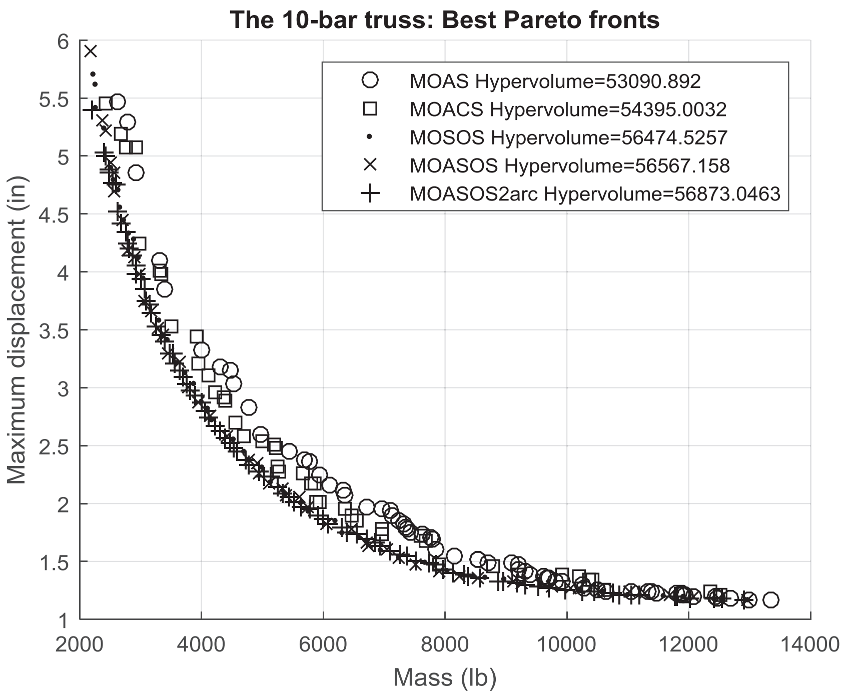

Fig. 4. Best Pareto fronts of the 10-bar truss.

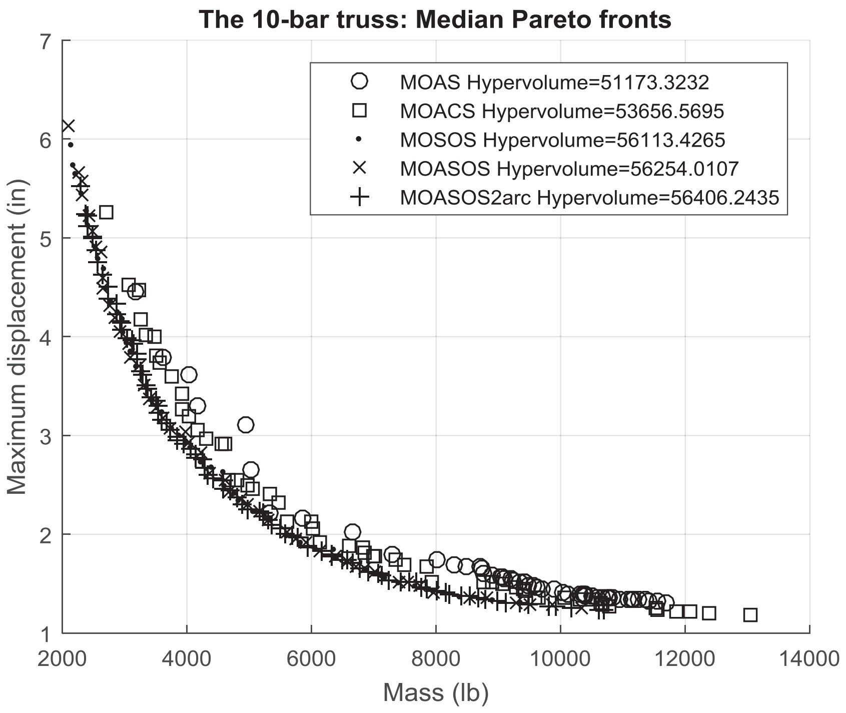

Fig. 5. Median Pareto fronts of the 10-bar truss.

equality and inequality constrains. Do and Lee [11] used a modified SOS in pin-jointed structures with discrete design variables. Yu et al. [27] used SOS for optimization the capacitated vehicle routing problem as a discrete optimization problem.

Capability of SOS in the field of structural optimization is still under research; however, Tejani et al. [22,23], Yu et al. [27], and Cheng and Prayogo [4] have investigated SOS for some structural optimization problems in single objective optimization. SOS works on three phases viz. the mutualism phase, commensalism phase, and parasitism phase. In the basic SOS algorithm, the benefit factor is decided through heuristic step and it can be either one or two, which means organism gets partially or fully benefits from the interaction. However, in real practice organism may get benefit in any proportion. Moreover, Tejani et al. [22] proposed an adaptive SOS algorithm with the use of adaptive control mechanism (viz. adaptive benefit factor). Automatically driven teaching factor has improved the performance of the SOS algorithm in order to set a good balance between exploration and exploitation of the search space and to enhance the diversity of the population. Adaptive controlling is proposed in few studies of MOEAs: Zhu

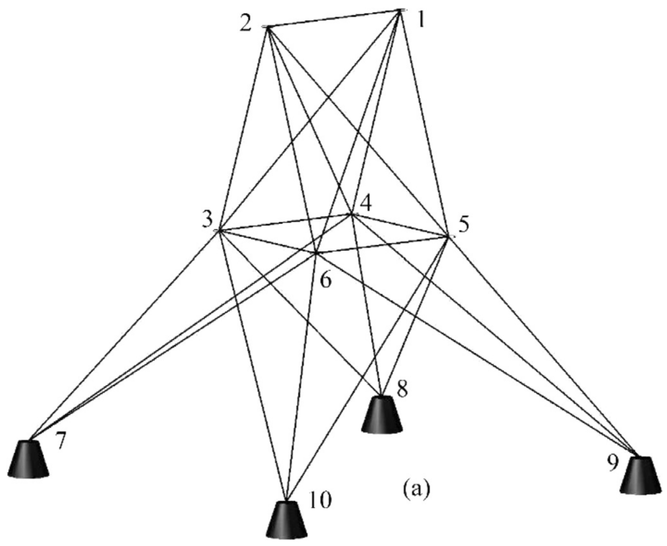

Fig. 6. The 25-bar truss.

et al. [29] presented an adaptive hybrid crossover operator for multiobjective evolutionary algorithm to enhance the search capability. Daryani et al. [10] applied an adaptive group search optimization multiobjective optimization problem to precise algorithms' convergence characteristic. This paper intends to investigate a good balance between exploration and exploitation of the search space. Therefore, we proposed two new versions of the basic SOS algorithm by considering adaptive benefit factor and two-archive technique in the basic SOS algorithm. It is also observed from the literature that MOSOS has not been investigated for structural optimization so far. Moreover, modification of MOSOS is still under research. These motives encouraged us to propose three variants of the SOS algorithm and to investigate its effect on structural optimization problems.

The SOS algorithm has even shown unique characteristics including: (1) no need for parameter adjustments since the algorithm is completely parameter-free; (2) Excellent capabilities for exploration with both mutualism and commensalism; (3) exploitation capabilities gained by cloning and mutating within the parasitism phase; (4) inferior solutions can be eliminated completely during the parasitism phase. With these four benefits, it is clear to see how the SOS algorithm excels. Compared with other meta-heuristic algorithms, there are very few that possess all four of the properties above and this leads to more accurate results and reliable processes.

Since the method has just been developed, there is room for further development, therefore, in this work we propose to improve the performance of the algorithm by integrating a self-adaptive strategy and a two-archive technique. The new MOASOS and MOASOS2arc algorithms are then used to solve a number of multiobjective truss optimization test problems while the objective functions include structural mass and maximum nodal displacement. The results obtained from using several optimisers are compared and discussed.

2. The symbiotic organisms search (SOS) algorithm

SOS was introduced by Cheng and Prayogo [4] as a continuous-based meta-heuristic algorithm that utilizes a population-based search strategy by maintaining a population of potential solutions when finding global optimum solutions to a given problem. The algorithm is motivated by the relationship among numerous organisms surviving together within an ecosystem. Generally, organisms have biological interdependence with others to grow or survive together within natural ecosystem. This phenomenon is called as 'symbiosis'.

In the beginning, the SOS algorithm starts with the initialization of the ecosystem population. After the initialization process, then the algorithm generates and evaluates each organism positions by calculating their respective objective function values, such that the organism with the best objective value is selected as X_{best}. The process is repeated iteratively by updating the current solution until the global best solution is discovered. In this situation, the SOS algorithm applied the principle of three fundamental symbiosis in living organisms; mutualism, commensalism, and parasitism; to update the new organism's position. The algorithm loop is terminated when the maximum number of fitness evaluation is met. The details of these natural symbiosis encoded in the SOS algorithm based on the three fundamental relationships structures are presented.

2.1. Mutualism phase

In this phase, both organisms experience advantages from the symbiotic relationship. For example, mutualism is found in flower and pollinator. After collecting food from the flower, the pollinator helps the flowers to become fruit. In this way, this symbiotic connotation benefits both individuals from the exchange. Therefore, this association is called a mutually advantageous symbiotic.

In the mutualism phase, the design variables (X_i) or solution 'i' sets relationship with a randomly selected design variables (X_k) or solution 'k' ($k \neq i$). The relationship between these populations outcomes to individual benefits. New solutions are governed by a Mutual Vector (MV) and Benefit Factors (BF_1 and BF_2). The MV implies the mutual relationship between solutions 'i' and 'k', shown in Eq. (3). The benefit factors are governed by a random integer number with an equal chance as either 1 or 2, shown in Eqs. (4) and (5). Therefore, the BF_1 and BF_2 indicate two situations where solutions get an advantage partly or completely from the relationship respectively. Populations are also affected by the best solution (X_{best}). The best solution is randomly selected from the first non-dominated sorting. The fitter solutions are selected as per greedy selection. The mathematical formulation of this phase is specified as below:

$$X_i' = X_i + rand(0,1) * (X_{best} - MV * BF_1) \tag{1}$$

$$X_k' = X_k + rand(0,1) * (X_{best} - MV * BF_2) \tag{2}$$

$$MV = \frac{X_i + X_k}{2} \tag{3}$$

$$BF_1 = round[rand(0,1)] + 1 \tag{4}$$

$$BF_2 = round[rand(0,1)] + 1 \tag{5}$$

where, $i,\ k \in (1, 2, \ldots, n);\ k \neq i$

2.2. Commensalism phase

In this phase, only one organism experiences advantage while the other does not gain or lose anything. The relationship between shark and remora fish is a key example with remora fish attaching themselves behind the shark's body. The fish manages to consume leftover food while the shark does not gain or lose anything from the behaviour.

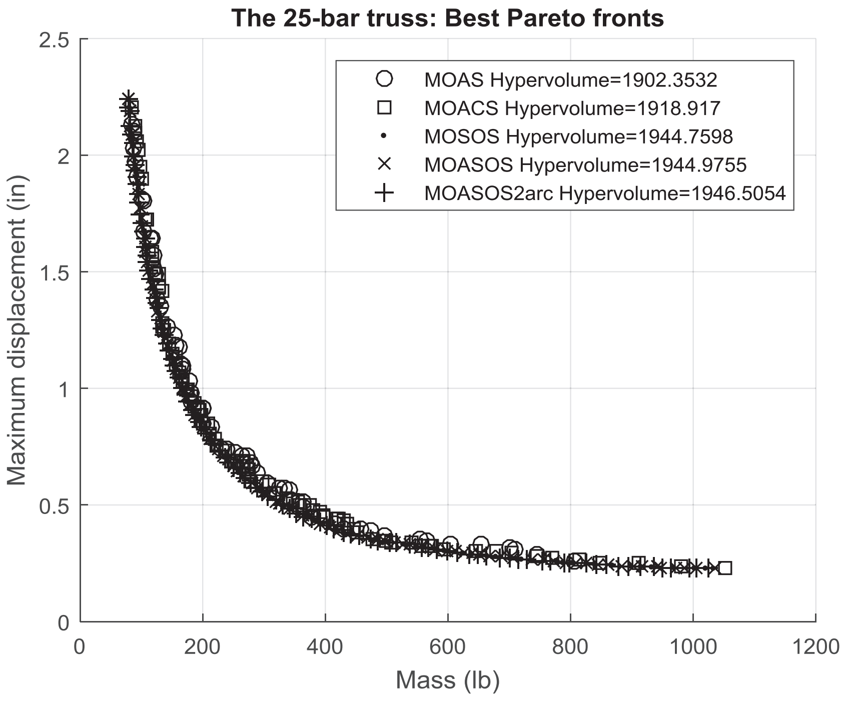

Fig. 7. Best Pareto fronts of the 25-bar truss.

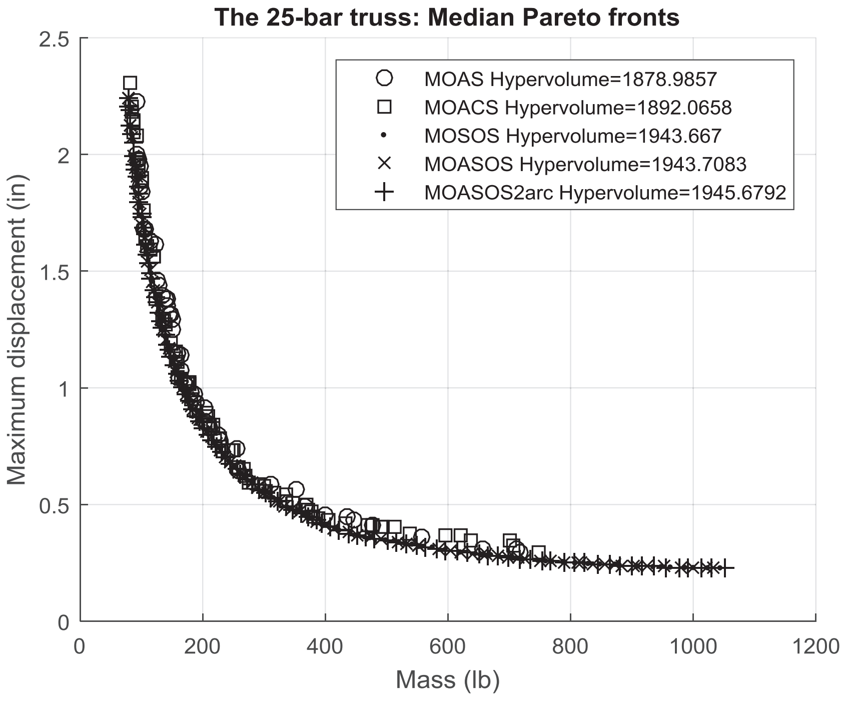

Fig. 8. Median Pareto fronts of the 25-bar truss.

This phase simulates the commensalism between two living organisms with one benefitting and the other seeing no change at all. The design variables (X_i) sets relationship with another design variables (X_k) where $k \neq i$. The relationship between these solutions leads to the individual advantage of solution '*i*' but does not affect solution '*k*'. Solutions are also affected by the best solution (X_{best}), randomly selected for the current set of non-dominated solutions. The fitter solutions are selected as per greedy selection. The mathematical formulation of this phase is specified as below:

$$X_i' = X_i + rand(-1,\ 1) * (X_{best} - X_k) \tag{6}$$

$$where,\ \ i,\ k \in (1, 2, \ldots, n);\ k \neq i$$

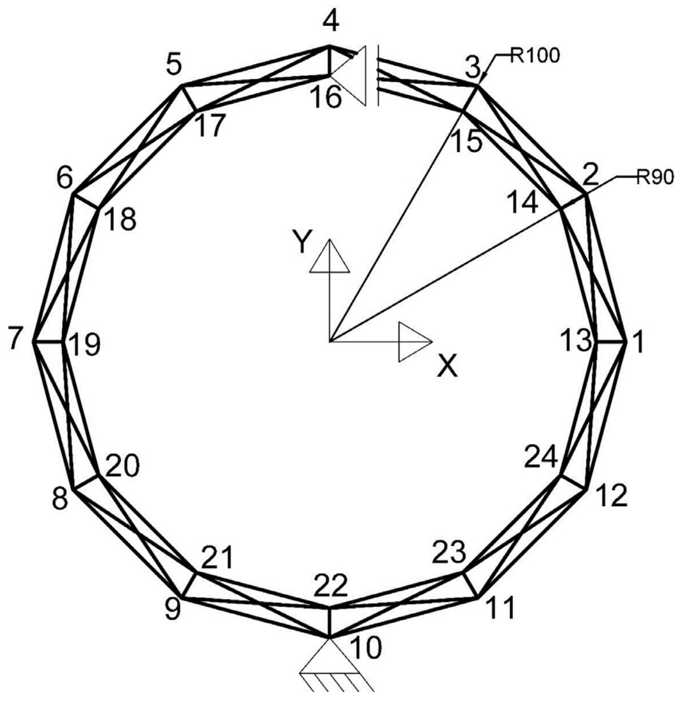

Fig. 9. 60-bar truss.

2.3. Parasitism phase

In this phase, one organism is eliminated from the ecosystem completely while the other gain an advantage. The most common example would be humans and mosquitos. After biting the human, the mosquito creates a parasite within the body. As the germs reproduce with bacteria, this leads to disease and even death when untreated or if the immune system is not strong enough to eliminate the parasite. This phase simulates parasitism between two living organisms is simulated with one benefitting and the other being harmed.

Like the anopheles mosquito, the design variable (X_i) creates an artificial parasite known as Parasite Vector (PV). PV is created by regenerating some randomly selected elements of solution '*i*' within their bounds, therefore, PV is a being a clone of the original elements of solution '*i*'. A randomly selected solution X_k for $k \neq i$ is assumed to be a human. If PV has better functional value compared to solution '*k*', the parasite PV will kill and replace solution '*k*'.

3. The multiobjective adaptive symbiotic organisms search (MOASOS) algorithm and a two-archive technique in MOASOS

In the mutualism phase of SOS, the benefit factors are key considerations to decide the influence of MV. Benefit factors are decided as per random choice as 1 or 2 in the basic SOS. This practice corresponds to the situation where populations get benefit partially or fully from MV. Thus, during the course of optimization, the organisms update only with these two possibilities. In the optimization algorithm, lower value of benefit factor allows the fine search in small steps but causes slow convergence and larger value of benefit factor speeds up the search. Moreover, in an actual mutualism phenomenon, these benefit factors may not always at its end state but varies in between also. By this motivation, the benefit factors (BF_1 and BF_2) are changed to adaptive benefit factors (ABF_1 and ABF_2) aiming for search performance enhancement, defined by the following equations:

$$ABF_1 = \begin{cases} f_1(X_i)/f_1(X_{best})\ if\ f_1(X_{best}) \neq 0 \\ 1 + round\,[rand(0,1)],\ if\ f_1(X_{best}) \neq 0 \end{cases} \quad (7)$$

$$ABF_1 = \begin{cases} 1,\ if\ ABF_1 < 1 \\ 2,\ if\ ABF_1 > 2 \\ ABF_1,\ otherwise \end{cases} \quad (8)$$

$$ABF_2 = \begin{cases} f_2(X_i)/f_2(X_{best})\ if\ f_2(X_{best}) \neq 0 \\ 1 + round\,[rand(0,1)],\ if\ f_2(X_{best}) \neq 0 \end{cases} \quad (9)$$

$$ABF_2 = \begin{cases} 1,\ if\ ABF_2 < 1 \\ 2,\ if\ ABF_2 > 2 \\ ABF_2,\ otherwise \end{cases} \quad (10)$$

It should be noted that the ABF_1 and ABF_2 are used for minimization of objective functions. In this mutualism phase of SOS, the design variables may have small and large move its locations as it works on various factors. The small and large movements of the design variables respectively impact the exploration and exploration of a search space. The ABF_1 and ABF_2 let strong exploration competence when a population ('*i*' *or* '*k*') is away from the best population. The ABF_1 and ABF_2 set good exploitation when a population is nearby the best population. Multiobjective adaptive symbiotic organisms search (MOASOS) aims to effectively add the robust and global search features of the adaptive benefits factors.

Populations are evolving to a fitter version only if their new fitness dominates their pre-interaction fitness. In this case, the old X_i and X_j will be replaced immediately by X_i' (X_i new) and X_j' (X_j new$_)$, respectively. The X_i and X_j will be moved into advanced population. Otherwise, the X_i' and X_j' will be added into advanced population for selecting the next generation ecosystem. In this way, the proposed algorithm can converge faster while maintaining good diversity. Since algorithm may gain some important information from dominated the solution in latter sorting.

Furthermore, a two-archive technique is embedded to MOASOS leading to a new algorithm termed MOASOS2arc. The main concept of using the two-archive approach is that another type of external archive will be generated based on a sharing function. The new archive is proposed to handle population diversity which is a common issue in multiobjective meta-heuristic. In the population reproduction of MOASOS2arc, $\boldsymbol{X}_{best}$ can be selected from both archives at random. To find the second archive, new objective functions are calculated and non-dominated solutions according to the new objective functions are evaluated. For an individual, the first new objective function is the reciprocal of the sum of distances between it and other solutions in a population while the second objective is the weighted sum of the original objective functions. For an individual $\mathbf{X}$ which having real objective function $\mathbf{f} = \{f_1, f_2\}$, the new functions for the second archive denoted as, f_{n1} and f_{n2}, can be expressed as:

$$f_{n1} = \frac{1}{\sum_{j=1}^{n-1} \left\| X - X_j \right\|},\ X = X_j \quad (11)$$

And

$$f_{n2} = w_1 f_1 + (1 - w_1) f_2 \quad (12)$$

where n is a population size while w_1 is a uniform random number in the range [0,1].

$\mathbf{X}_{\rm best}$ will be randomly selected from the second archive if a random number in [0,1] lower than a threshold values (*rand(0,1)* < *Arch2*$_{\rm p}$) where the threshold values (*Arch2*$_{\rm p}$) is iteratively adapted. Otherwise, $\mathbf{X}_{\rm best}$ is randomly selected from the first non-dominated archive. The term of *Arch2*$_{\rm p}$ can be iteratively adaptive based on the following function:

$$Arch2_p(t) = R \times e^{St} \quad (13)$$

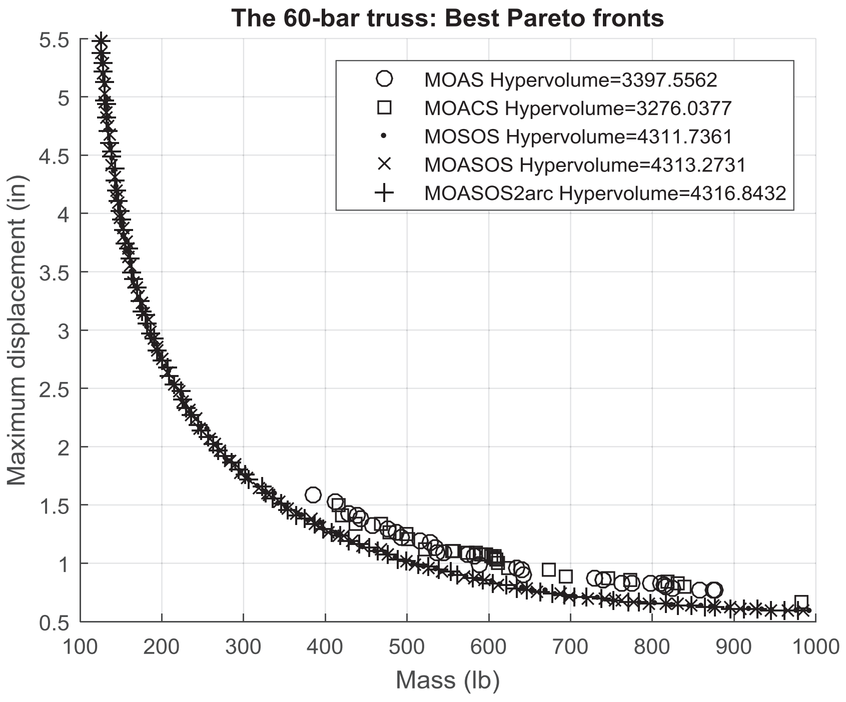

Fig. 10. Best Pareto fronts of the 60-bar truss.

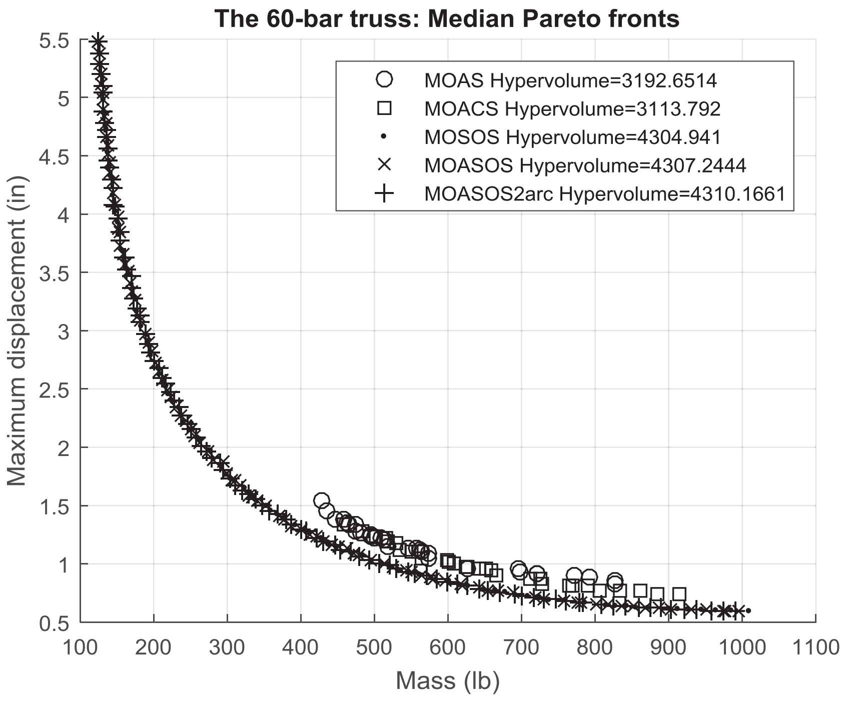

Fig. 11. Median Pareto fronts of the 60-bar truss.

$$S = \frac{\ln\left(Arch2_{pf}\right) - \ln\left(Arch2_{ps}\right)}{t_{max} - 1} \tag{14}$$

$$R = \frac{Arch2_{ps}}{e^{S}} \tag{15}$$

$Arch2_{\mathrm{ps}}$ and $Arch2_{\mathrm{pf}}$ are the starting and ending values of $Arch2_{\mathrm{p}}$ which set to be 0.1 and 0.5, respectively. The concept of using the second archive is to provide more diversity in the final stage of an optimisation run, however, based on Eq. (13), the second archive will be less used in the early stage of the run. This idea was successfully used in [17].

MOSOS, MOASOS, and MOASOS2arch simulate three phases such as ‘mutualism phase’, ‘commensalism phase’, and ‘parasitism phase’. It presents various steps of these algorithms like initialization, mutualism phase, commensalism phase, parasitism phase, and termination criteria. Detailed steps of MOSOS, MOASOS, and MOASOS2arch are explained as below:

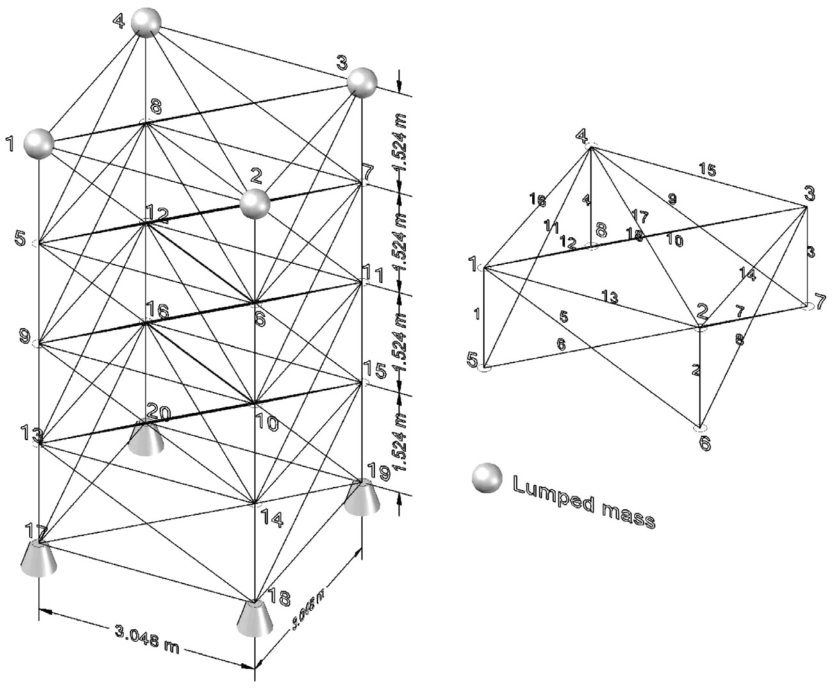

Fig. 12. the 72-bar truss.

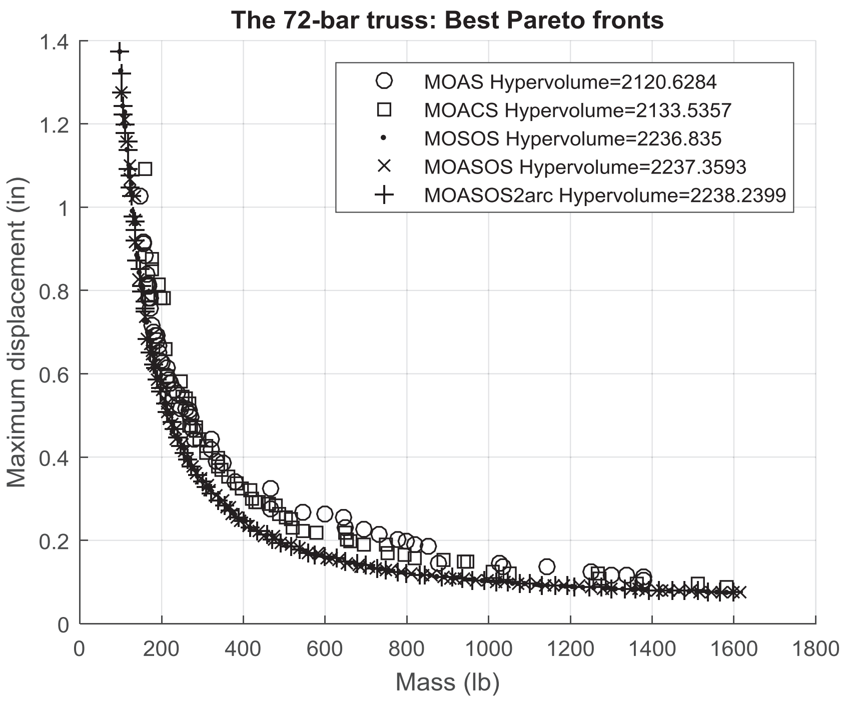

Fig. 13. Best Pareto fronts of the 72-bar truss.

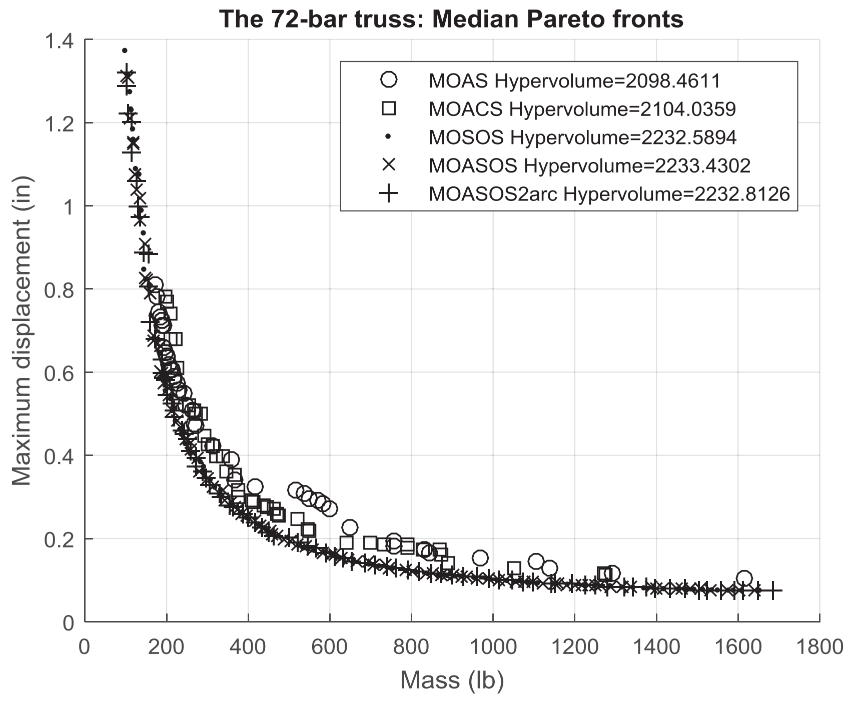

Fig. 14. Median Pareto fronts of the 72-bar truss.

```
Initialize population size ($n$), Number of design variables ($m$), limits on design variables ($L$, $U$), stopping criteria (‘$FE_{max}$’ or ‘$g_{max}$’)    /* Initialization /*
$X_{i,j} = L_{i,j} + R_{i,j}*(U_{i,j} - L_{i,j}),\ for \forall\ i \in [1, n],\ for \forall\ j \in [1, m]$   /* $R \in [0,\ 1]$ /*   /* Initialize population /*
Evaluate the population and arrange the population in ascending order $\rightarrow FE = n$
Identify the best population of the ecosystem.
while ($g < g_{max}$ and $FE < FE_{max}$) do        /* Initialize optimization loop /*
    for i = 1 to n do
        $MV = \frac{X_i+X_k}{2}$        /* ‘$k$’ is a randomly selected population, $k \neq i$ /*
%% if the MOSOS algorithm then
        $BF_1 = 1 + round\ [rand(0,1)]$        /* The mutualism phase /*
        $BF_2 = 1 + round\ [rand(0,1)]$
        $X_i' = X_i + rand(0,1)*(X_{best} - MV*BF_1)$ /* $X_{best}$ is randomly selected from the first non-dominated rank /*
        $X_k' = X_k + rand(0,1)*(X_{best} - MV*BF_2)$
%% if the MOASOS algorithm then
        if $f_1(X_{best}) \neq 0 \rightarrow ABF_1 = f_1\ (X_i)/\ f_1\ (X_{best})$
        if $ABF_1 < 1 \rightarrow ABF_1 = 1$
        if $ABF_1 > 2 \rightarrow ABF_1 = 2$
        $X_i' = X_i + rand(0,1)*(X_{best} - MV*ABF_1)$ /* $X_{best}$ is randomly selected from either the first
                  or second archives in cases of MOASOS2arc/*
        if $f_2(X_{best}) \neq 0 \rightarrow ABF_2 = f_2\ (X_i)/\ f_2\ (X_{best})$
        if $ABF_2 < 1 \rightarrow ABF_2 = 1$
        if $ABF_2 > 2 \rightarrow ABF_2 = 2$
        $X_k' = X_k + rand(0,1)*(X_{best} - MV*ABF_2)$ %%
        Evaluate $f_1(X_i');\ f_2(X_i')\ \&\ f_1(X_k');\ f_2(X_k') \rightarrow FE = FE + 2$
        $F(X_i') < F(X_i) \leftrightarrow X_i = X_i'$        /* Greedy selection/*
        $F(X_k') < F(X_k) \leftrightarrow X_k = X_k'$        /* Greedy selection/*
        $X_i' = X_i + rand(-1,1)*(X_{best} - X_k)$        /* The commensalism phase /*
        Evaluate $F(X_i') \rightarrow FE = FE + 1$        /* ‘k’ is a randomly selected population, k ≠ i /*
        $(X_i') < F(X_i) \leftrightarrow X_i = X_i'$        /* Greedy selection/*
        Parasite_Vector /* Parasite vector is a fusion of design variables of the population ‘i’ and randomly generated design variables within its bound /* /*
The parasitism phase /*
        $(Parasite\_Vector) < F(X_k) \leftrightarrow X_k = Parasite\_Vector$        /* Greedy selection/*
end for        /* Population loop ends /*
end while      /* Optimization loop ends /*
```

The flowchart of the MOSOS, MOASOS, and MOASOS2arc algorithms is shown in Fig. 1.

4. Problem definition

A typical multiobjective truss optimization problem is modeled to find element sizes which minimize truss mass and maximum nodal deflection subject to stress constraints. The optimization problem can be written as:

$$Find,\ A = \{A_1,\ A_2, .., A_m\} \tag{16}$$

to minimize, mass of truss and maximum elemental deflection

$$f_1(A) = \sum_{i=1}^{m} A_i \rho_i\ L_i\ and\ f_2(A) = max\left(\left|\delta_j\right|\right)$$

$$Subjected\ to:$$

Behavior constraints:

$g(A)$: *Stress constraints*, $|\sigma_i| - \sigma_i^{max} \le 0$

Side constraints:

Cross − sectional area constraints, $A_i^{min} \le A_i \le A_i^{max}$

where, $i = 1, 2, \ldots, m;\ j = 1, 2, \ldots, n$

Where, $A_{i,}\, \rho_{i,} L_i, E_i, and\, \sigma_i$ stand for design variable, mass density, element length, Young modules, and element stress, on the element *'i'* respectively. δ_j is displacement node *'j'* respectively. The superscripts 'max' and 'min' denote maximum allowable limit, minimum allowable limit respectively.

4.1. Penalty function

Assuming each objective function distinctly and its minimization subject to q constraints, the penalty function of a given solution can be written as:

$$f(X) * (1 + \varepsilon_1 * \mathrm{C})^{\varepsilon_2}, \mathrm{C} = \sum\nolimits_{i=1}^{q} \mathrm{C}_i,\ \mathrm{C}_i = \left|1 - \frac{p_i}{p_i^*}\right| \quad (17)$$

Where, p_i is the level of constraint violation having the bound as p_i^*. The parameter q is a number of active constraints. The variables ε_1 and ε_2 are pre-determined by the user. In this study, the values of both ε_1 and ε_2 are set as 3, which were obtained from experimenting their effect on the balance of the exploitation-exploration balance [21–23].

4.2. The proposed methodology

The brief stepwise discussion of the proposed methodology is as below:

Step 1. Define the basic configuration of the truss structure.
Step 2. Assign material properties, loadings, and boundary conditions.
Step 3. Go to the multiobjective optimization algorithm: define objective functions, population size, design variables, bounds, and a termination criterion.
Step 4. Initialize a randomly generated set of trusses (i.e., population) within its upper and lower bounds.
Step 5. Go to truss configurations: generate trusses as per the basic truss structure.
Step 6. Perform finite element analysis using the matrix method of structural analysis of a truss.
Step 7. Compute the global stiffness matrix.
Step 8. Compute a force vector, solve the boundary conditions, and then solve for nodal displacement.
Step 9. Compute element stresses.
Step 10. Go to a penalty function and check for constraint violations. If there exist constraint violations, assign penalty values as per Eq. (17); otherwise, compute the total mass and maximum nodal displacement of the truss.
Step 11. Go to multiobjective optimization algorithm: assign functional values.
Step 12. Update a Pareto archive using a non-dominated sorting algorithm.
Step 13. Check the termination criterion. If it is not fulfilled, generate new trusses (i.e., solutions) as per reproduction of the employed algorithm. Go to step 5. If the criterion is met, go to step 14.
Step 14. Output: Pareto optimal truss structures with the total mass and maximum nodal displacement.

Graphical illustration of the proposed methodology is presented in Fig. 2.

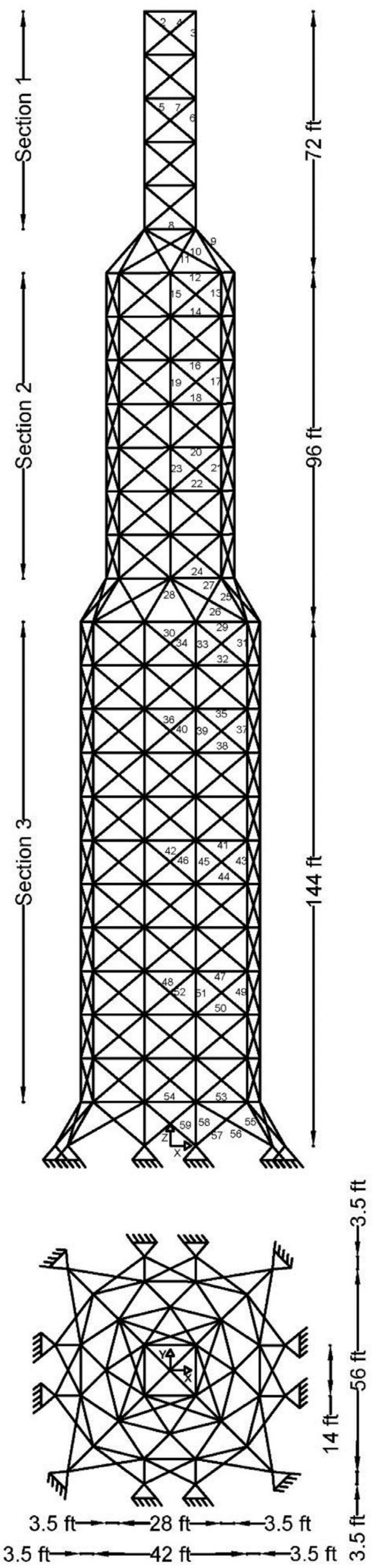

Fig. 15. the 942-bar truss.

5. Design problems, results, and discussions

Five benchmark trusses from Angelo et al. [2,3] are used to investigate the performance of the proposed algorithms. The MOSOS, MOASOS, and MOASOS2arc algorithms are performed for 100 independent runs. The front hypervolume and spacing-to-extent indi-

Table 1
The hypervolume values of results obtained for the 10-bar truss using MOAS, MOACS, MOSOS, MOASOS, and MOASOS2arc.

Algorithms	Min	Max	Avg.	SD	Friedman test	Friedman rank	Nemenyi average rank	Nemenyi rank
MOAS	47302.93	53090.89	50902.21	1294.12	100	5	5	5
MOACS	52662.60	54395.00	53639.77	307.79	200	4	4	4
MOSOS	55466.26	56474.53	56092.89	212.59	352	3	2.48	3
MOASOS	55850.54	56567.16	56236.08	156.33	397	2	2.03	2
MOASOS2arc	55890.60	56873.05	56389.83	196.10	451	1	1.49	1

cators are used for performance investigation. The average value of the hypervolume from 100 independent runs (Avg.) of each method is used to measure the search convergence of the algorithm while the standard deviation (SD) of hypervolume is used to measure the search consistency. In addition, a front spacing (S) matric [19] is used to measure relative distance between consecutive solutions in the obtained non-dominated set. Given that a set of non-dominated front P having M objective functions is obtained from a particular optimization method, the spacing of such a front can be computed as:

$$Spacing = \frac{1}{|P|-1}\sum_{i=1}^{|P|}\left(d_i - \bar{d}\right)^2 \tag{18}$$

where $|P|$ is the number of members in the set P. d_i is the Euclidian distance of the vector of objective functions i to its nearest neighbor. $\bar{d}$ is the mean value of d_i.

The measure of front extension can be determined as:

$$Extent = \sum_{i=1}^{M}\left|f_i^{max} - f_i^{min}\right| \tag{19}$$

where $f_i^{\min}$ and $f_i^{\max}$ are the minimum and maximum values for the ith objective function of all members in P. The lower *Spacing* the better front while the higher *Extent* the better. The combination of both indicators leads to a new performance metric which measures both front spacing and front extent, which is defined as the ratio of spacing to extent,

$$Spacing - to - Extent = Spacing/Extent \tag{20}$$

where the lower *Spacing-to-Extent* the better non-dominated front.

Also, the two-step statistical tests called Friedman test and Nemenyi test [7] are used to rank the algorithms based on the experimental results. The five truss optimization problems can be detailed in the subsequent sections.

5.1. A 10-bar truss

The fist problem, the 10-bar truss is presented in Fig. 3. The material properties are as density is 0.1 lb/in^3 and modulus of elasticity is 10^4 ksi. Elements 2 and 4 are subjected to vertical download forces as 100 kips. The tensile and compressive stresses are limited to 25 ksi. The elemental cross-sections are selected from 42 discrete values as 1.62, 1.80, 1.99, 2.13, 2.38, 2.62, 2.63, 2.88, 2.93, 3.09, 3.13, 3.38, 3.47, 3.55, 3.63, 3.84, 3.87, 3.88, 4.18, 4.22, 4.49, 4.59, 4.80, 4.97, 5.12, 5.74, 7.22, 7.97, 11.50, 13.50, 13.90, 14.20, 15.50, 16.00, 16.90, 18.80, 19.90, 22.00, 22.90, 26.50, 30.00, and 33.50 in^2.

This problem was run with population size of 50 and number of generations 100 thus it consumes 15000 FEs. After performing 100 optimisation runs for all optimisers, the hypervolume values are presented in Table 1. The best method based on the Avg. value is MOASOS2arc while the second best is MOASOS. For the measure of search consistency, the best performer is MOASOS while the second best is MOASOS2arc. The maximum and hypervolume values is obtained by MOASOS2arc. For the comparison based on the Friedman test and Nemenyi test are at 95% significant level, the MOASOS2arc and MOASOS algorithms are the best and second best performer similar to the measurement based on Avg. of hypervolume values.

The front Spacing-to-Extent performance metrics are calculated for this problem and the results are presented in Table 2. As per the Friedman test and Nemenyi test at 95% significant level, the MOASOS2arc outperforms other algorithms and similar results expressed as per Avg. and SD of front Spacing-to-Extent values. Also, the MOASOS2arc and MOASOS algorithms performs better compare to its basic version. Friedman test and Nemenyi test are shown nearly similar results.

Figs. 4 and 5 show the best obtained Pareto fronts of all the proposed algorithms and the Pareto fronts at median run. It is observed that MOASOS2arc is slightly better than other algorithms. It is also observed from the Pareto fronts of MOASOS and MOASOS2arc that a wide range of distinct solutions is obtained, and the solutions are well distributed along the obtained non-dominated fronts. Overall, these results demonstrate that MOASOS2arc is the best performer and proposed modifications advance effectiveness of the basic MOSOS algorithm.

5.2. A 25-bar truss

The second problem, the 25-bar truss is presented in Fig. 6. The material properties are as density is 0.1 lb/in^3 and modulus of elasticity is 10^4 ksi. Loading is considered as $P_{x1} = 1\,Klb$, $P_{y1} = P_{z1} = P_{y2} = P_{z2} = -10\,Klb$, $P_{x3} = 0.5\,Klb$, $P_{x6} = 0.6\,Klb$. The tensile and compressive stresses are limited to 40 ksi. The elemental cross-sections are selected from 42 discrete values as 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.8, 3.0, 3.2, and 3.4 in^2.

For the 25-bar truss, the problem was run with the population size of 100 and the number of generations 167, thus, it takes 50,000 FEs. After performing 100 optimisation runs for all optimisers, the hypervolume values are presented in Table 3. The best for convergence and consistency is MOASOS2arc while the second best is MOASOS. The maximum hypervolume is obtained by MOASOS2arc. Based on the Friedman test and Nemenyi test, MOASOS2arc and MOASOS are still the best and second best performers.

The front Spacing-to-Extent performance metrics are examined for this problem and the results are shown in Table 4. As per the Friedman test and Nemenyi test at 95% significant level, the MOASOS2arc, MOASOS, and MOSOS rank first, second, and third respectively and similar results are obtained by Avg. and SD of front Spacing-to-Extent values. In addition, the MOASOS2arc and MOASOS algorithms performs better compare to its basic version. Friedman test and Nemenyi test are shown nearly similar results.

The Pareto fronts obtained from the best runs of all algorithms and at median position are shown in Figs. 7 and 8. It is also noticed from the Pareto fronts of MOASOS and MOASOS2arc that a wide range of distinct solutions is obtained, and the solutions are well distributed along the obtained non-dominated fronts. Overall, these results reveal that MOASOS2arc is slightly better than the other algorithms similarly to the case of the10-bar truss and proposed modifications upgrade effectiveness of the basic MOSOS algorithm.

Table 2
The front spacing-to-extent values of results obtained for the 10-bar truss using MOAS, MOACS, MOSOS, MOASOS, and MOASOS2arc.

Algorithms	Min	Max	Avg.	SD	Friedman test	Friedman rank	Nemenyi average rank	Nemenyi rank
MOAS	0.005387	0.024711	0.010590	0.003781	292	2	2.92	2
MOACS	0.007219	0.029625	0.014219	0.004558	415	5	4.15	5
MOSOS	0.007441	0.022216	0.011524	0.002013	350	4	3.5	4
MOASOS	0.007705	0.021266	0.011380	0.002189	340	3	3.4	3
MOASOS2arc	0.004698	0.007001	0.005661	0.000461	103	1	1.03	1

Table 3
The hypervolume values of results obtained for the 25-bar truss using MOAS, MOACS, MOSOS, and MOASOS.

Algorithms	Min	Max	Avg.	SD	Friedman test	Friedman rank	Nemenyi average rank	Nemenyi rank
MOAS	1848.04	1902.35	1878.74	9.77	123	5	4.77	5
MOACS	1850.64	1918.92	1890.61	14.39	177	4	4.23	4
MOSOS	1942.37	1944.76	1943.60	0.50	344	3	2.56	3
MOASOS	1942.65	1944.98	1943.76	0.48	356	2	2.44	2
MOASOS2arc	1944.13	1946.51	1945.61	0.45	500	1	1	1

Table 4
The front spacing-to-extent values of results obtained for the 25-bar truss using MOAS, MOACS, MOSOS, and MOASOS.

Algorithms	Min	Max	Avg.	SD	Friedman test	Friedman rank	Nemenyi average rank	Nemenyi rank
MOAS	0.007937	0.058983	0.022595	0.008424	465	5	4.65	5
MOACS	0.005254	0.044937	0.017026	0.008361	430	4	4.3	4
MOSOS	0.005219	0.007943	0.006651	0.000453	237	3	2.37	3
MOASOS	0.005655	0.007759	0.006601	0.000440	214	2	2.14	2
MOASOS2arc	0.005733	0.006779	0.006362	0.000208	154	1	1.54	1

Table 5
The hypervolume values of results obtained for the 60-bar truss using MOAS, MOACS, MOSOS, MOASOS, and MOASOS2arc.

Algorithms	Min	Max	Avg.	SD	Friedman test	Friedman rank	Nemenyi average rank	Nemenyi rank
MOAS	2465.08	3397.56	3179.88	166.65	173	4	4.27	4
MOACS	2905.27	3276.04	3106.68	74.18	127	5	4.73	5
MOSOS	4297.11	4311.74	4304.74	3.06	344	3	2.56	3
MOASOS	4298.92	4313.27	4307.08	3.42	399	2	2.01	2
MOASOS2arc	4299.15	4316.84	4309.75	3.44	457	1	1.43	1

Table 6
The front spacing-to-extent values of results obtained for the 60-bar truss using MOAS, MOACS, MOSOS, MOASOS, and MOASOS2arc.

Algorithms	Min	Max	Avg.	SD	Friedman test	Friedman rank	Nemenyi average rank	Nemenyi rank
MOAS	0.009977	0.133920	0.034915	0.019500	460	5	4.6	5
MOACS	0.007890	0.074504	0.029912	0.013732	440	4	4.4	4
MOSOS	0.005487	0.008685	0.006526	0.000587	247	3	2.47	3
MOASOS	0.005589	0.007388	0.006346	0.000409	230	2	2.3	2
MOASOS2arc	0.005371	0.006465	0.005849	0.000202	123	1	1.23	1

Table 7
The hypervolume values of results obtained for the 72-bar truss using MOAS, MOACS, MOSOS, MOASOS, and MOASOS2arc.

Algorithms	Min	Max	Avg.	SD	Friedman test	Friedman rank	Nemenyi average rank	Nemenyi rank
MOAS	2070.95	2120.63	2098.37	10.02	142	5	4.58	5
MOACS	2043.73	2133.54	2101.00	18.78	158	4	4.42	4
MOSOS	2223.50	2236.83	2232.36	2.19	380	3	2.20	3
MOASOS	2227.37	2237.36	2233.18	2.21	423	1	1.77	1
MOASOS2arc	2225.95	2238.24	2232.63	2.40	397	2	2.03	2

Table 8
The front spacing-to-extent values of results obtained for the 72-bar truss using MOAS, MOACS, MOSOS, MOASOS, and MOASOS2arc.

Algorithms	Min	Max	Avg.	SD	Friedman test	Friedman rank	Nemenyi average rank	Nemenyi rank
MOAS	0.010919	0.043568	0.022728	0.007183	434	4	4.34	4
MOACS	0.007918	0.076088	0.026837	0.013808	442	5	4.42	5
MOSOS	0.013350	0.015643	0.014393	0.000435	324	3	3.24	3
MOASOS	0.004268	0.008528	0.005835	0.000774	108	1	1.08	1
MOASOS2arc	0.006503	0.007543	0.007126	0.000229	192	2	1.92	2

Table 9
The hypervolume values of results obtained for the 942-bar truss using MOAS, MOACS, MOSOS, MOASOS, and MOASOS2arc.

Algorithms	Min	Max	Avg.	SD	Friedman test	Friedman rank	Nemenyi average rank	Nemenyi rank
MOAS	44268539.95	55932657.12	50768236.63	3289808.43	150	4	4.5	4
MOACS	48328371.14	53517016.79	51442233.77	917155.44	150	4	4.5	4
MOSOS	59247759.94	79812406.08	61635925.25	2844333.17	359	3	2.41	3
MOASOS	60216408.45	64558874.01	61770355.74	617413.33	443	1	1.57	1
MOASOS2arc	59720107.25	62630388.16	61452029.45	573717.84	398	2	2.02	2

Table 10
The front spacing-to-extent values of results obtained for the 942-bar truss using MOAS, MOACS, MOSOS, MOASOS, and MOASOS2arc.

Algorithms	Min	Max	Avg.	SD	Friedman test	Friedman rank	Nemenyi average rank	Nemenyi rank
MOAS	0.014273	0.120749	0.042659	0.020981	473	5	4.73	5
MOACS	0.010403	0.079907	0.029013	0.014678	427	4	4.27	4
MOSOS	0.003170	0.010096	0.004965	0.001050	245	3	2.45	3
MOASOS	0.003285	0.007260	0.004315	0.000515	185	2	1.85	2
MOASOS2arc	0.003226	0.006074	0.004202	0.000475	170	1	1.7	1

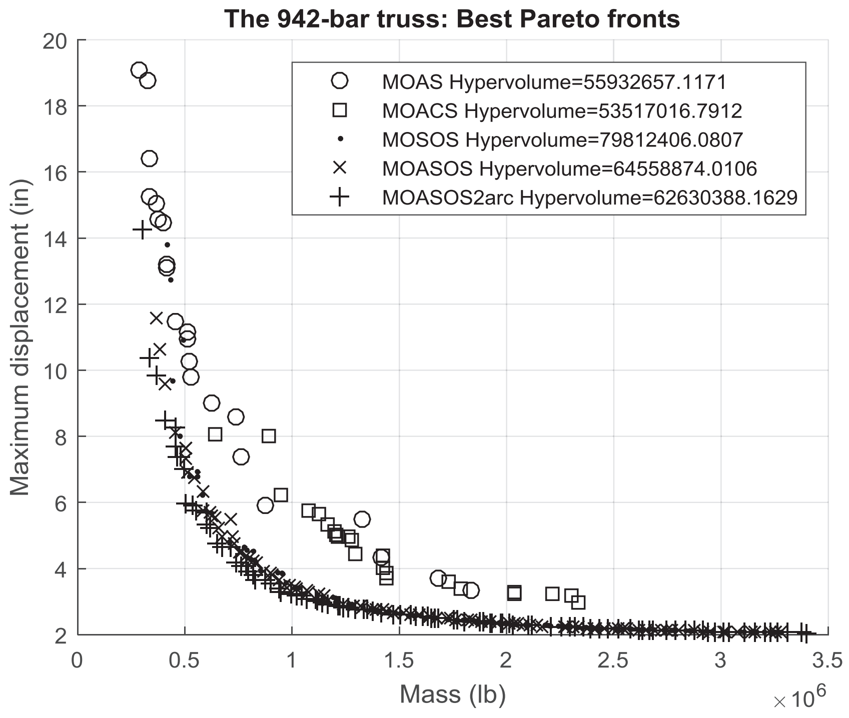

Fig. 16. Best Pareto fronts of the 942-bar truss.

5.3. A 60-bar truss

The third problem, the 60-bar trussed ring is presented in Fig. 9. The material properties are as density is 0.1 lb/in^3 and modulus of elasticity is 10^4 ksi. Loading is assumed as load case 1: $P_{x1} = -10\,Klb$ and $P_{x7} = 9\,Klb$, load case 2: $P_{x15} = P_{x18} = -8\,Klb$, and $P_{y15} = P_{y18} = 3\,Klb$, and load case 3: $P_{x22} = -20\,Klb$ and $P_{y22} = 10\,Klb$. The tensile and compressive stresses are limited to 40 ksi. The elemental cross-sections are selected from 45 discrete values as [0.5,0.6,0.7,…,4.9] in^2.

For the 60-bar truss, the problem was run with the population size of 100 and the number of generations 167, thus, it takes 50,000 FEs. Having performed 100 optimisation runs for all optimisers, the hypervolume values are presented in Table 5. The best for convergence rate is MOASOS2arc while the second best is MOASOS similar to the cases of 10-bar truss and 25-bar truss. For the measure of search consistency, the best performer is MOSOS while the second best is MOASOS. The maximum hypervolume is still obtained by MOASOS2arc for this case. The conclusion based on the Friedman test and Nemenyi test is that MOASOS2arc and MOASOS are still the best and the second best performers.

The front Spacing-to-Extent performance metrics are analysed for this problem and the results are illustrated in Table 6. It is observed as per the Friedman test and Nemenyi test at 95% significant level that the MOASOS2arc, MOASOS, and MOSOS rank first, second, and third respectively and similar results are also obtained by Avg. and SD of front Spacing-to-Extent values. In addition, the MOASOS2arc and MOASOS algorithms performs better compare to its basic version. Friedman test and Nemenyi test are shown nearly similar results.

For this case, MOASOS2arc is said to be the overall best performer. Figs. 10 and 11 show the Pareto fronts obtained from the best runs of all the algorithms and the Pareto fronts at median runs. It is also observed from the Pareto fronts of MOASOS and MOASOS2arc that a wide range of distinct solutions is obtained, and the solutions are well distributed along the obtained non-dominated fronts. The Pareto fronts obtained from MOASOS2arc, MOASOS, and MOSOS are stable compare to that obtained by using

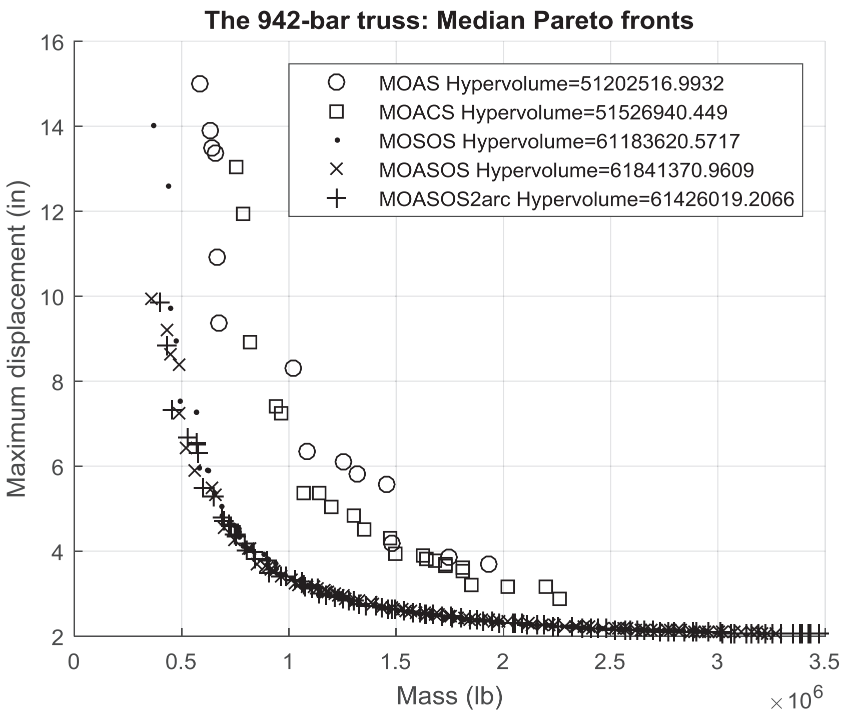

Fig. 17. Median Pareto fronts of the 942-bar truss.

MOAS and MOACS. It can be concluded that MOASOS2arc front is slightly better than the other algorithms and proposed modifications advance effectiveness of the basic MOSOS algorithm.

5.4. A 72-bar truss

The fourth problem, the 72-bar truss is presented in Fig. 12. The material properties are as density is 0.1 lb/in^3 and modulus of elasticity is 10^4 ksi. Loading is assumed as load case 1: $F_{1x} = F_{1y} = 5\ kips\ and\ F_{1z} = -5\ kips$ and load case 2: $F_{1z} = F_{2z} = F_{3z} = F_{4z} = -5\ kips$. The tensile and compressive stresses are limited to 25 ksi. The elemental cross-sections are selected from 25 discrete values as [0.1,0.2,0.3,…, 2.5] in^2. The bar elementals are grouped into 16 groups to consider structural similarity as per previous studies.

For the 72-bar truss, the problem was run with population size of 100 and number of generations 167, therefore, it consumes 50,000 FEs. After performing 100 optimisation runs for all optimisers, the hypervolume values are presented in Table 7. The best algorithm based on the Avg. value is MOASOS while the second best is MOASOS2arc. For the measure of search consistency, the best performer is MOSOS while the second best is MOASOS similar to the case of 60-bar truss. The maximum hypervolume is still obtained by MOASOS2arc for this case. The conclusion based on the Friedman test and Nemenyi test is similar to the measurement based on Avg. of the hypervolume values i.e. MOASOS is the best while MOASOS2arc is the second best.

The front Spacing-to-Extent performance metrics are calculated for this case and the results are illustrated in Table 8. It is observed as per the Friedman test and Nemenyi test at 95% significant level that the MOASOS, MOASOS2arc, and MOSOS rank first, second, and third respectively and similar results are also obtained by Avg. of front Spacing-to-Extent values. In addition, the MOASOS2arc and MOASOS algorithms performs better compare to its basic version. Friedman test and Nemenyi test are shown nearly similar results.

Figs. 13 and 14 shown the best Pareto fronts obtained from the best runs and the median results of all algorithms. It is observed that MOASOS is slightly better than the other algorithms. It is also observed from the Pareto fronts of MOASOS and MOASOS2arc that a wide range of distinct solutions is obtained, and the solutions are well distributed along the obtained non-dominated fronts. The Pareto fronts obtained from MOASOS2arc, MOASOS, and MOSOS are stable compare to that obtained by using MOAS and MOACS. For this case, the results reveal that MOASOS is the best algorithm and proposed modifications lead to greater effectiveness of the basic MOSOS algorithm.

5.5. A 942-bar truss

The fifth problem, the 942-bar truss is presented in Fig. 15. The material properties are as density is 0.1 lb/in^3 and modulus of elasticity is 10^4 ksi. Loading is assumed as the vertical loads along z axis are – 3 kips, – 6 kips, and −9 kips at each of the nodes in the first, second, and third sections, respectively; the lateral loads along y axis are 1 kips at each node of the truss; and the lateral loads along x axis are 1.5 kips and 1.0 kips at each node on the left and right sides of the truss, respectively. The tensile and compressive stresses are limited to 25 ksi. The elemental cross-sections are selected from 200 discrete values as [1,2,3,…, 200] in^2. The bar elementals are grouped into 59 groups to consider structural similarity as per previous studies [2,3].

For the 942-bar truss, the problem was run with the population size of 100 and the number of generations 167. It consequently uses 50,000 FEs. After performing 100 optimisation runs for all optimisers, the hypervolume values are presented in Table 9. The best algorithm based on the Avg. value is MOASOS while the second best is MOSOS. For the measure of search consistency, the best performer is MOASOS2arc while the second best is MOASOS. The maximum hypervolume value is obtained from MOSOS for this case. The comparison based on the Friedman test and Nemenyi test shows that the best performer is MOASOS while the second best is MOASOS2arc. For this case, the results demonstrate that MOASOS is better optimiser compare to other algorithms.

The front Spacing-to-Extent performance metrics are examined for this problem and the results are illustrated in Table 10. It is observed as per the Friedman test and Nemenyi test at 95% significant level that the MOASOS2arc, MOASOS, and MOSOS rank first, second, and third respectively and similar results are also obtained by Avg. and SD of front Spacing-to-Extent values. In addition, the MOASOS2arc and MOASOS algorithms performs better compare to its basic version. Friedman test and Nemenyi test are shown nearly similar results.

Figs. 16 and 17 shown the Pareto fronts obtained from the best and median runs of all algorithms. It is observed that MOASOS2arc and MOASOS are better performer than the other algorithms. It is also observed from the Pareto fronts of MOASOS and MOASOS2arc that a wide range of distinct solutions is obtained, and the solutions are well distributed along the obtained non-dominated fronts. The Pareto fronts obtained from using MOASOS2arc, MOASOS, and MOSOS are superior to that obtained by using MOAS and MOACS. For this case, the results reveal that MOASOS2arc and MOASOS are nearly similar and better compared to other algorithms. It is also observed that the proposed modifications improved effectiveness of the basic MOSOS algorithm.

6. Conclusions

In this study, the MOSOS, MOASOS, and MOASOS2arc algorithms are proposed to optimize five constrained benchmark truss problems. The basic MOSOS algorithm works on the symbiotic relationship called as mutualism, commensalism, and parasitism. As the main contribution, this work proposes an adaptive benefit factor to handle trade-off between exploration and exploitation of search process during the mutualism phase of the basic MOSOS algorithm whereas two-archive approach is combined with MOASOS to handle population diversity.

The effectiveness of the proposed algorithms is investigated to design planar trusses (i.e. 10-bar truss and 60-bar truss) and space trusses (i.e. 25-bar truss, 60-bar truss, and 942-bar truss) subjected to elemental stress and discrete cross-sectional areas as behaviour and side constraints respectively. Objective functions are minimization of the truss' mass and maximum of nodal displacement. The MOSOS, MOASOS, and MOASOS2arc algorithms are firstly employed in multiple objective structural optimization problem s and thus it simulates engineering applications.

This study compared performance of the MOASOS and MOASOS2arc algorithms with the original MOSOS and other MHs such as MOAS and MOACS. The front hypervolume and Spacing-to-Extent matrices were employed to measure effectiveness of the algorithms. Also, the two-step statistical tests called Friedman test and Nemenyi test were used to rank the algorithms. It was observed that in all the problems, the MOASOS and MOASOS2arc algorithms has a better capability for obtaining results as compared to the results of MOSOS, MOAS, and MOACS. MOASOS2arc performs better compare to MOASOS to optimize small trusses such as 10-bar truss, 25-bar truss, and 60-bar truss; whereas MOASOS performs better compare to MOASOS2arc to optimize large trusses such as 72-bar truss and 942-bar truss with discrete design variables. Overall, the presented improvements upsurge a good balance between exploitation and exploration, and able to maintain population diversity in the original MOSOS algorithm.

References

[1] Mohammed Abdullahi, Md Asri Ngadi, Shafi'i Muhammad Abdulhamid, Symbiotic organism search optimization based task scheduling in cloud computing environment, Futur. Gener. Comput. Syst. 56 (2016) 640–650, doi:10.1016/j.future.2015.08.006.

[2] Jaqueline S. Angelo, Helio JC Barbosa, Heder S Bernardino, Multi-objective ant colony approaches for structural optimization problems, in: Proceedings of the Eleventh International Conference on Computational Structures Technology, 2012 , doi:10.4203/ccp.99.66.

[3] Jaqueline S. Angelo, Heder S. Bernardino, Helio JC Barbosa, Ant colony approaches for multiobjective structural optimization problems with a cardinality constraint, Adv. Eng. Softw. 80 (C) (2015) 101–115, doi:10.1016/j.advengsoft.2014.09.015.

[4] Min-yuan Yuan Cheng, Doddy Prayogo, Symbiotic organisms search: a new metaheuristic optimization algorithm, in: Computers and Structures, 139, Elsevier Ltd, 2014, pp. 98–112, doi:10.1016/j.compstruc.2014.03.007.

[5] Min-yuan Cheng, Doddy Prayogo, Duc-hoc Tran, Optimizing multiple-resources leveling in multiple projects using discrete symbiotic organisms search, J. Comput. Civ. Eng. (2015), doi:10.1061/(ASCE)CP.1943-5487.0000512.

[6] Margarita Reyes-Sierra, Carlos A Coello Coello, Multi-objective particle swarm optimizers: a survey of the state-of-the-art, Int. J. Comput. Intell. Res. 2 (3) (2006) 287–308, doi:10.5019/j.ijcir.2006.68.

[7] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30, doi:10.1016/j.jecp.2010.03.005.

[8] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evolut. Comput. 6 (2) (2002) 182–197, doi:10.1109/4235.996017.

[9] Kalyanmoy Deb, Himanshu Jain, An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, part I: solving problems with box constraints, IEEE Trans. Evolut. Comput. 18 (4) (2013) 577–601, doi:10.1109/TEVC.2013.2281534.

[10] Narges Daryani, Mehrdad Tarafdar Hagh, Saeed Teimourzadeh, Adaptive group search optimization algorithm for multi-objective optimal power flow problem, in: Applied Soft Computing, 38, Elsevier B.V., 2016, pp. 1012–1024, doi:10.1016/j.asoc.2015.10.057.

[11] D.T.T. Do, J. Lee, A modified symbiotic organisms search (MSOS) algorithm for optimization of pin-jointed structures, Applied Soft Computing, 61, Elsevier B.V., 2017, doi:10.1016/j.asoc.2017.08.002.

[12] Berat Doğan, Tamer Ölmez, A New Metaheuristic for numerical function optimization: vortex search algorithm, Inf. Sci. 293 (2015) 125–145, doi:10.1016/j.ins.2014.08.053.

[13] Matheus Silva Gonçalves, Rafael Holdorf Lopez, Leandro Fleck Fadel Miguel, Search group algorithm: a new metaheuristic method for the optimization of truss structures, Comput. Struct. 153 (2015) 165–184, doi:10.1016/j.compstruc.2015.03.003.

[14] A. Kaveh, K. Laknejadi, A hybrid multi-objective particle swarm optimization and decision making procedure for optimal design of truss structures, Iran. J. Sci. Technol. 35 (C2) (2011) 137–154.

[15] Guan-Chun Luh, Chung-Huei Chueh, Multi-objective optimal design of truss structure with immune algorithm, Comput. Struct. 82 (11) (2004) 829–844, doi:10.1016/j.compstruc.2004.03.003.

[16] Seyedali Mirjalili, SCA: a sine cosine algorithm for solving optimization problems, Knowl. Based Syst. (2016), doi:10.1016/j.knosys.2015.12.022.

[17] K. Nuaekaew, P. Artrit, N. Pholdee, S. Bureerat, Optimal reactive power dispatch problem using a two-archive multi-objective grey wolf optimizer, Expert Syst. Appl. 87 (2017) 79–89, doi:10.1016/j.eswa.2017.06.009.

[18] Arnapurna Panda, Sabyasachi Pani, A symbiotic organisms search algorithm with adaptive penalty function to solve multi-objective constrained optimization problems, Appl. Soft Comput. 46 (2016) 344–360, doi:10.1016/j.asoc.2016.04.030.

[19] Jason R. Schott, Master's Thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1995.

[20] N Srinivas, Kalyanmoy Deb, Multiobjective optimization using nondominated sorting in genetic algorithms, Evolut. Comput. 2 (3) (1995) 221–248, doi:10.1162/evco.1994.2.3.221.

[21] Ghanshyam G Tejani, Vimal J Savsani, Sujin Bureerat, Vivek K Patel, Topology and size optimization of trusses with static and dynamic bounds by modified symbiotic organisms search, J. Comput. Civ. Eng. 32 (2) (2018) 1–11, doi:10.1061/(ASCE)CP.1943-5487.0000741.

[22] Ghanshyam G. Tejani, Vimal J. Savsani, Vivek K. Patel, Adaptive symbiotic organisms search (SOS) algorithm for structural design optimization, J. Comput. Des. Eng. 3 (3) (2016) 226–249, doi:10.1016/j.jcde.2016.02.003.

[23] Ghanshyam G. Tejani, Vimal J. Savsani, Vivek K. Patel, Seyedali Mirjalili, Truss optimization with natural frequency bounds using improved symbiotic organisms search, Knowledge-Based Systems, Elsevier B.V., 2017, doi:10.1016/j.knosys.2017.12.012.

[24] Duc-Hoc Tran, Duc-Long Luong, Minh-Tin Duong, Trong-Nhan Le, Anh-Duc Pham, Opposition multiple objective symbiotic organisms search (OMOSOS) for time, cost, quality and work continuity tradeoff in repetitive projects, J. Comput. Des. Eng. (2017), doi:10.1016/j.jcde.2017.11.008.

[25] Duc-Hoc Tran, Min-Yuan Cheng, Doddy Prayogo, A novel multiple objective symbiotic organisms search (MOSOS) for time–cost–labor utilization tradeoff problem, Knowl. Based Syst. 94 (2016) 132–145, doi:10.1016/j.knosys.2015.11.016.

[26] Ryoji Tanabe, Alex S Fukunaga, Improving the search performance of SHADE using linear population size reduction, in: Proceedings of the IEEE Congress on Evolutionary Computation, CEC, 2014, pp. 1658–1665, doi:10.1109/CEC.2014.6900380.

[27] Vincent F. Yu, A.A.N. Perwira Redi, Chao Lung Yang, Eki Ruskartina, Budi Santosa, Symbiotic organisms search and two solution representations for solving the capacitated vehicle routing problem, in: Applied Soft Computing, 52, Elsevier B.V., 2017, pp. 657–672, doi:10.1016/j.asoc.2016.10.006.
[28] Q. Zhang, H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evolut. Comput. 11 (6) (2007) 712–731, doi:10.1109/TEVC.2007.892759.
[29] Qingling Zhu, Qiuzhen Lin, Zhihua Du, Zhengping Liang, Wenjun Wang, Zexuan Zhu, Jianyong Chen, Peizhi Huang, Zhong Ming, A novel adaptive hybrid crossover operator for multiobjective evolutionary algorithm, in: Information Sciences, 345, Elsevier Inc, 2016, pp. 177–198, doi:10.1016/j.ins.2016.01.046.
[30] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: improving the strength pareto evolutionary algorithm for multiobjective optimization, Evolut. Methods Des. Optim. Control 1 (2002) 1–6.