Development of a multi-objective artificial tree (MOAT) algorithm and its application in acoustic metamaterials

Qiqi Li[1]*, Zhichen He[2], Eric Li[3]*, Tao Chen[2], Qiuyu Wang[4], Aiguo Cheng[2]

[1]*College of Automotive and Mechanical Engineering, Changsha University of Science and Technology, 410114, Yuhua District, Changsha City, Hunan Province, P. R. China*

[2]*State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body, Hunan University, Changsha, 410082 P. R. China*

[3]*School of Science, Engineering & Design, Teesside University, Middlesbrough, UK*

[4]*Tangsteel Company Technical Center, Tangshan, Hebei, P. R. China*

Abstract

Although there are many algorithms that can solve the multi-objective optimization problems (MOPs) efficiently, each algorithm has its own disadvantages. The emergence of new algorithms is beneficial to make up the deficiencies of existing algorithms. Inspired by the organic matter transport process and the branch update theory of the banyan, this work proposed a new bio-inspired algorithm, named the multi-objective artificial tree (MOAT) algorithm to solve the MOPs. In MOAT, an improved crossover operator and an improved self-evolution operator are introduced to update solutions, a adaptive grid method is applied to manage the non-dominated solutions, and the strategy of variable number of branches in population is adopted to enhance the accuracy of this algorithm. Many typical test functions and seven well-known multi-objective algorithms, including MOEAD, NSGAII, MOPSO, GDE3, εMOEA, IBEA and MPSO/D, are applied to study the accuracy and efficiency of MOAT. Experimental tests show that the results of MOAT are better than those of the seven algorithms, and the performance of MOAT is demonstrated. In addition, this new algorithm is also applied to solve the MOPs of two-dimensional acoustic metamaterials (AMs). The key parameters of AMs are optimized by MOAT to mitigate impact load and reduce structural mass, and the performance of these AMs is significantly improved.

***Key words*:** multi-objective optimization problems; bio-inspired algorithm; multi-objective artificial tree algorithm; acoustic metamaterials;

* Corresponding author.

E-mail address: hdliqiqi@163.com (Qiqi Li); ericsg2012@gmail.com (Eric Li)

1 Introduction

Multi-objective optimization problems (MOPs) [1-4] are common in engineering design, and their objectives always conflict with each other. Regarding MOPs, the improvement of one goal will impede the others. The optimum solutions of these MOPs are a set of solutions called Pareto solution set, and the graph consisting of these Pareto solutions is called Pareto front. Many theories have been proposed to solve the MOPs, and one of the most important branches is the bio-inspired optimization algorithm. In 1985, based on the standard genetic algorithm (GA), Schaffer [5] proposed the first multi-objective evolutionary algorithm. Since this algorithm was proposed, many efficient bio-inspired multi-objective algorithms have been proposed by researchers, such as improved strength Pareto evolutionary algorithm (SPEA2) [6] and non-dominated sorting genetic algorithm II (NSGA-II) [7].

In addition to these typical bio-inspired multi-objective algorithms, many efficient new algorithms have also been proposed to improve their ability to solve the MOPs. Some of the latest studies are listed as follows. Sun et al. [8] presented an adaptive multi-objective evolutionary algorithm (AMEA). Clustering approach and advanced sampling strategy are applied to adaptively learn the manifold structure of Pareto solution set and produce promising offspring. For the uneven distribution problem of individuals in the design space, Qiao et al. [9] proposed an adaptive hybrid evolutionary immune algorithm (AUDHEIA). A hyperplane which is associated with the target space is generated, and these individuals in population are mapped into the hyperplane to increase the diversity of these solutions. In addition, the mapped hyperplane is uniformly divided to improve the distribution of these solutions. Facing multi-objective and many-objective problems, Luo et al. [10] studied an indicator-based multi-objective artificial bee colony optimization method (ε-MOABC). An external archive is built in this algorithm based on Pareto dominance and preference indicators to preserve non-dominated solutions generated by each generation. Ou et al. [11] proposed a Pareto-based evolutionary algorithm to solve the dynamic MOPs. Three new strategies which are environment selection technique, mating selection strategy and dynamic response mechanism are introduced to enhance the diversity and convergence of solutions. Lin et al. [12] studied a dynamic control strategy of population size, and this strategy is introduced to the multi-objective immune algorithm (MOIA-DPS). The state of external archive is used to determine whether the size of

population will increase or decrease. A TDE operator which has two search models is alternately exchanged according to the probability to improve the robustness of MOIA-DPS.

It is obvious that the theories of bio-inspired multi-objective algorithms have been making progress, and these new algorithms show great effectiveness and competitiveness. However, these algorithms still have deficiencies in solving some of the MOPs [13]. For example, multi-objective particle swarm optimization (MOPSO) does not solve some MOPs well (such as ZDT4) [13]. Many multi-objective evolutionary methods, such as SPEA2 [6] and NSGA-II [7], are inspired by Darwin's theory of natural selection, which emphasizes natural selection while neglecting cooperation between species [14]. The ε-domination [10] does not effectively maintain solutions in the areas where the distribution of Pareto front is close to horizontal or vertical, resulting in a large loss of them.

All of these prove the No Free Lunch Theorem [15]. Although an algorithm has excellent effects in solving some problems, it may fail to solve other problems. For this reason, new optimization algorithms are always interested by researchers. Many scholars have done a lot of endeavours on the study of new theories, and the proposal of new algorithms can make up the deficiencies of existing algorithms. In this paper, aiming at enhancing the optimization accuracy and efficiency for MOPs, a new algorithm named the multi-objective artificial tree (MOAT) algorithm is proposed. MOAT is inspired by the growth law of banyan which is a multi-objective version of the artificial tree (AT) algorithm [16].

The main challenges of this work include how to extend the basic AT algorithm to a multi-objective algorithm that can solve the MOPs and how to guarantee the accuracy and efficiency of this new algorithm. In order to transform AT from a single-objective version to a multi-objective algorithm, bio-inspired model of this algorithm is transformed from a general tree to a banyan. In AT, branches stand for solutions, and the thickest tree trunk is the best solution. For the MOPs, many branches and multiple tree trunks are required to represent solutions and non-dominated solutions, respectively. Fig. 1 shows a banyan model, and it is obvious that the banyan model has many branches and tree trunks. Therefore, the banyan model meets the requirements. In AT [16], two operators, namely the crossover operator and the self-evolution operator, are applied to update the branches. However, the basic AT algorithm does not consider the information interaction between branches sufficiently [17, 18]. In this work, an improved crossover operator and an improved self-evolution operator are introduced to consider the information interaction between branches

comprehensively. In addition, an adaptive grid method is introduced to manage the solutions in archive to ensure their diversity and convergence.

The main contributions of this proposed algorithm are summarized below. First, it is well known that AT is an efficient single-objective algorithm, and this work is a pioneer which transforms the basic AT algorithm into a multi-objective algorithm. Second, to improve the performance of MOAT, some efficient theories are introduced. The innovations of the proposed algorithm are listed below: (1) Compared to AT, an improved crossover operator and an improved self-evolution operator are introduced in MOAT to update the branches. (2) An adaptive grid method is proposed to manage the non-dominated solutions and guarantee the diversity and convergence of solutions. (3) A variable branch number strategy is proposed to make full use of the non-dominated solutions to improve the accuracy of MOAT.

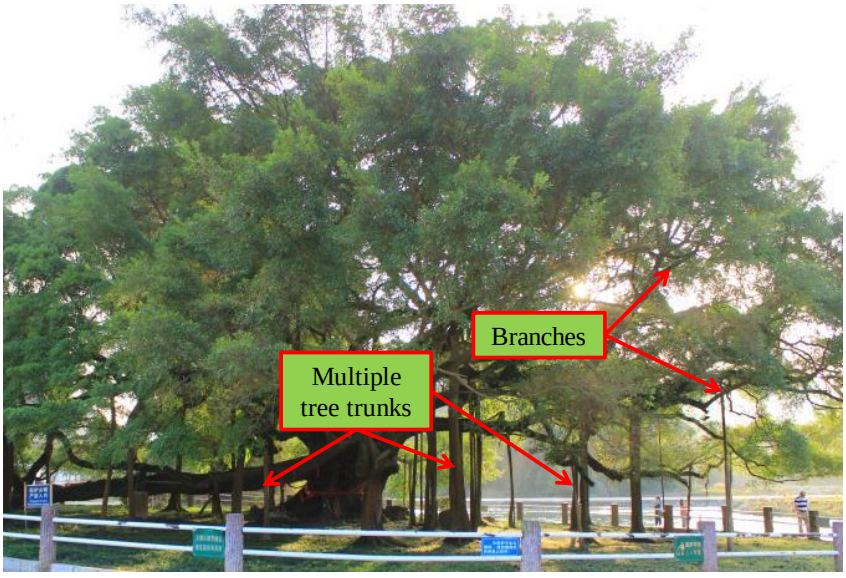


Fig. 1. A banyan with many branches and tree trunks.

In addition to the study of MOAT algorithm, another goal of this work is to apply this new proposed algorithm to design the two-dimensional AMs to maximize its attenuation effects [19-21] and minimize its structural mass. Previous scholars had made some attempts to use the bio-inspired algorithms to design AMs. Li et al. [19] applied AT to optimize the resonant structures inside an sandwich panel to reduce structural deformation under impact load. Some key parameters were considered in the optimization process, and numerical results showed the great improvement of structural responses. Bacigalupo et al. [22] optimized an anti-tetrachiral AM to enlarge its band gaps, and some nonlinear optimization problems and inequality constraints were introduced in their work.

In this work, the bio-inspired model and the flow of MOAT are firstly studied. Some typical test

functions are applied to study the efficiency and accuracy of MOAT. In order to further test the performance of MOAT, numerical results calculated by MOAT are compared with seven well-known multi-objective algorithms, including MOEA/D [23], NSGAII [7], MOPSO [24], GDE3 [25], εMOEA [26], IBEA [27] and MPSO/D [28]. Experimental results show that the performance of MOAT is better than that of the seven algorithms. This proves that MOAT is an effective algorithm, and this new proposed theory provides us a new choice for solving the MOPs. Then, MOAT is applied to optimize the two-dimensional AMs to improve their impact responses. Finally, the optimum responses of these AMs with lightweight structure and small impact load are obtained.

This paper is organized as follows: the theory of MOAT algorithm is presented in Section 2. Section 3 compares the optimization results of MOAT and seven well-known multi-objective algorithms for ten typical test problems. Section 4 shows the optimization process of the two-dimensional AMs with MOAT. Finally, Section 5 makes the conclusions of this paper.

2 The theory of MOAT

In MOAT, the whole optimization process consists of the transportation of organic matters from leaves to multiple tree trunks and the update of branches. The branches themselves stand for the solutions, the positions of branches represent the design variables, and a thicker branch always dominates a thinner branch. The transfer process of organic matters depends on the renewal of tree branches. Therefore, the update theories of branches are the key to determine the whole optimization process of MOAT. In basic AT, there are two branch update theories, namely the crossover operator and the self-evolution operator. These two operators are applied to update the branches and the solutions.

Fig. 2 shows the bio-inspired model of the banyan, and this model consists of many branches and multiple tree trunks. In Fig. 2, the blue arrows represent the propagation of organic matters, the red circles mean the improved crossover operator, and the yellow boxes denote the improved self-evolution operator. Organic matters are first produced in the leaves and then transmitted to the tree branches with the renewal of branches. The improved crossover operator and the improved single-evolution operator are exhibited to update the branches, and the updated branches are always thicker than the original ones. Meanwhile, these multiple tree trunks denote the Pareto solution set. When the organic matters are transmitted to the multiple tree trunks, meaning the branches are updated to the thickest trunks that cannot be dominated by other branches, the optimization process

is over.

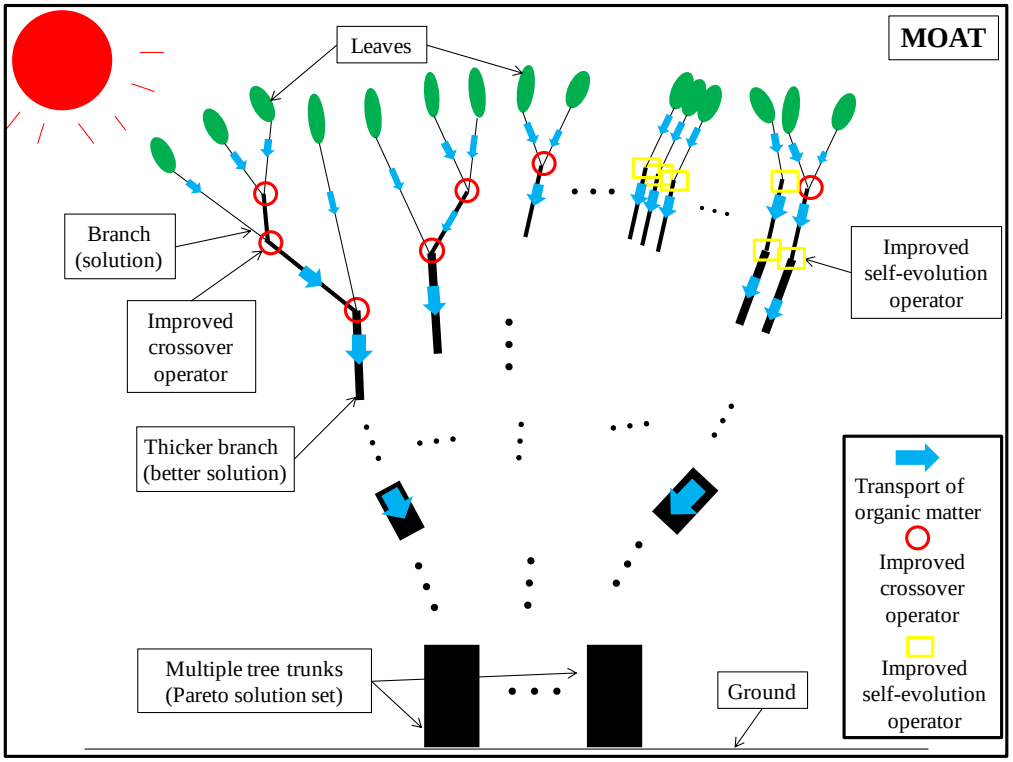


Fig. 2. The branches renewal process and the organic matters transfer process of the banyan model.

The execution process of MOAT algorithm is summarized as follows.

First, a number of branches are randomly generated in the design space as the initial population. These branches are compared with each other, and the relatively thicker branches are selected as the non-dominated solutions. An archive is defined to store these non-dominated solutions.

Second, all these branches in population are renewed by update operators. A pareto dominance analysis is performed with these new branches, and some non-dominated solutions are generated. Then, these newly acquired non-dominated solutions are combined with the non-dominated solutions in archive, and a new pareto dominance analysis is carried out with this combined solution set to obtain the new non-dominated solution set of current cycle.

Third, the new non-dominated solution set replaces the old non-dominated solution set in archive. The adaptive grid method is applied to manage the branches in archive to remove these excessive branches and maintain the diversity of non-dominated solutions.

Fourth, the branch population is updated by the variable branch number strategy.

Fifth, continue to perform the second, third and fourth steps until the Pareto optimum solution set is obtained. The detailed execution process of MOAT is shown as follows.

2.1 Initialize the branch population

Before the branch population is initialized, it is necessary to determine the number of branches in population and the dimension of each branch, that is, the dimension of problems to be solved. In this work, population size and branch dimension are defined as *Bn* and *D*, respectively. Then, some design points which are initial branches are randomly generated in design space. The positions of these branches are $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{Bn})$. The location of *i*-th branch is $\mathbf{x}_i = (x_{i1}, x_{i2}, ..., x_{iD})$. The positions of these initial branches are randomly generated by Eq. (1) below.

$$x_{ij} = x_j^{\min} + rand(0,1) \times (x_j^{\max} - x_j^{\min}) \tag{1}$$

where x_{ij} is the *j*-th variable of the *i*-th branch, *rand*(0,1) is a random number between 0 and 1, $x_j^{\max}$ and $x_j^{\min}$ are the upper and lower bounds for the *j*-th variable of branches. Then, the solutions of initial branch population are calculated, and the initial non-dominated solution set can be acquired through the analysis of Pareto dominance. Then, the initial non-dominated solution set is stored in the archive. The following pseudo code is the process of initializing branch population.

Algorithm 1 Initialize branch population

For *i* = 1 to *Bn*
  For *j*= 1 to *D*
    Calculate $x_{ij} = x_j^{\min} + rand(0,1) \times (x_j^{\max} - x_j^{\min})$
  End For
  Calculate the solution of branch *i*
End For
Perform the pareto dominance analysis
Acquire the initial non-dominated solution set
Acquire the initial archive

2.2 Improved crossover operator and improved self-evolution operator

The renewal of branches depends on the update operators of branches. In AT [16], there are two operators, crossover operator and self-evolution operator. Therefore, the branch update operator should be selected before the branch is renewed. However, the basic crossover and self-evolution operators do not consider the information interaction between branches sufficiently [17]. When a branch is updated by the crossover operator, it has no information exchange with other branches. When a branch is updated by the self-evolution operator, it only considers its own information and the information of the best branch currently found. In this work, these two branch update operators

are improved to fully consider the information interaction between branches to enhance the performance of MOAT. The following is how a operator is selected and how this selected operator is carried out.

A parameter r and a random number $rand(0,1)$ are used to determine which branch update operator is applied. The value of r is between 0 and 1 which is defined in advance. If $rand(0,1) \le r$, the improved crossover operator is applied to renew the branch. Otherwise, the improved self-evolution operator is carried out. Therefore, the value of r significantly affects the execution probability of these two operators. If $r > 0.5$, the execution probability of the improved crossover operator is higher than that of the improved self-evolution operator. The process of the improved crossover operator is summarized as follows.

First, a branch in archive is selected randomly as a leader branch, and its position is the leader branch position $\mathbf{x}_{\text{leader}}$. Then, one branch location $\mathbf{x}_j$ which is different from current branch position $\mathbf{x}_i$ is randomly selected from the branch population. Next, two branch positions are acquired based on the new selected branch position $\mathbf{x}_j$, the current branch position $\mathbf{x}_i$ and the leader branch position $\mathbf{x}_{\text{leader}}$. The mathematical models are shown as Eqs. (2) and (3), where c_1 and c_2 are the coefficients to correct the newly generated branches. If the values of c_1 and c_2 are too large, the newly created branch positions $\mathbf{x}_{\text{L}}$ and $\mathbf{x}_{\text{R}}$ tend to be far from the original branch position $\mathbf{x}_i$, which may cause non-convergence of the optimization process. Conversely, if the values of c_1 and c_2 are too small, the optimization process may be inefficient. Therefore, the values of c_1 and c_2 should be appropriate, and they are recommended to range from 0 to 1 in this work. Finally, based on these two obtained branches, a new branch is produced by Eq. (4).

$$\mathbf{x}_{\text{L}} = \mathbf{x}_i + rand(0,1) \times (\mathbf{x}_{\text{leader}} - \mathbf{x}_j) \times c_1 \tag{2}$$

$$\mathbf{x}_{\text{R}} = \mathbf{x}_i + rand(0,1) \times (\mathbf{x}_{\text{leader}} - \mathbf{x}_i) \times c_2 \tag{3}$$

$$\mathbf{x}_{\text{new}} = rand(0,1) \times \mathbf{x}_{\text{L}} + rand(0,1) \times \mathbf{x}_{\text{R}} \tag{4}$$

where $\mathbf{x}_{\text{L}}$ and $\mathbf{x}_{\text{R}}$ denote two generated branch positions based on $\mathbf{x}_j$, $\mathbf{x}_i$ and $\mathbf{x}_{\text{leader}}$, $\mathbf{x}_{\text{new}}$ denotes the new branch position.

In MOAT, the new branch generated by the improved crossover operator should dominate the original branch. If the new branch is thicker than the original one, the new one replaces the original one. Otherwise, the new branch should be discarded and another new branch is searched by the

improved crossover operator. Then, a second comparison between the new branch and the original branch is carried out. If the new branch is still thinner than the original branch, a third comparison is made. Continue this comparison until a thicker branch is found or the number of comparisons reaches the maximum try number (*Tn*) [16]. *Tn* is a predefined parameter that is used to control the generation of new branches, and the value of *Tn* should be chosen appropriately. Excessive value of *Tn* will cause waste of computing resources, while too small *Tn* will reduce optimization efficiency. If the number of comparisons reaches *Tn*, it means that this branch position cannot produce a thicker branch by the improved crossover operator. Next, another operation is performed to update the branch and branch position. Here, a parameter *h* and a random number *rand*(0,1) are applied to decide the next operation. *h* is a predefined parameter, and its value is between 0 and 1. If *rand*(0,1) ≤ *h*, the current branch, whose position is $\mathbf{x}_{\text{new}}$, will replace the original one for the next round of optimization. Otherwise, the current branch is discarded, and a new branch randomly generated by Eq. (1) in design space replaces the original one. If *h* takes a relatively large value, it tends to retain the currently generated new branch. A relatively high retention probability of $\mathbf{x}_{\text{new}}$ facilitates the convergence of optimization. However, a excessive value of *h* tends to cause the optimization problem to converge to local optimum. Therefore, the value of *h* is recommended between 0.5 and 0.9. The following pseudo code is the process of the improved crossover operator to update the branch position $\mathbf{x}_i$.

Algorithm 2 Improved crossover operator

Set the trial number as 1
While the trial number ≤ *Tn*
Select the leader branch position $\mathbf{x}_{\text{leader}}$
Select the branch position $\mathbf{x}_j$ $(i \neq j)$ in the branch population
Calculate the branch position $\mathbf{x}_{\text{L}} = \mathbf{x}_i + rand(0,1) \times (\mathbf{x}_{\text{leader}} - \mathbf{x}_j) \times c_1$
Calculate the branch position $\mathbf{x}_{\text{R}} = \mathbf{x}_i + rand(0,1) \times (\mathbf{x}_{\text{leader}} - \mathbf{x}_i) \times c_2$
Calculate the new branch position $\mathbf{x}_{\text{new}} = rand(0,1) \times \mathbf{x}_{\text{L}} + rand(0,1) \times \mathbf{x}_{\text{R}}$
Calculate the solution of the new branch whose position is $\mathbf{x}_{\text{new}}$
Perform the Pareto dominance analysis between the new branch ($\mathbf{x}_{\text{new}}$) and the original branch ($\mathbf{x}_i$)
If the new branch ($\mathbf{x}_{\text{new}}$) dominates the original one ($\mathbf{x}_i$)
Replace the original branch ($\mathbf{x}_i$) with the new branch ($\mathbf{x}_{\text{new}}$)
break
else

Increment its trial by 1
End If
End While
If current branch is not updated
If $rand(0,1) \le h$
Replace the original branch ($\mathbf{x}_i$) with the last generated new branch ($\mathbf{x}_{new}$)
else
Replace the original branch ($\mathbf{x}_i$) with the randomly generated branch in the design space
End If
End If

Regarding the improved self-evolution operator, the update of branch position is also based on the new selected branch position $\mathbf{x}_j$, the current branch position $\mathbf{x}_i$ and the leader branch position $\mathbf{x}_{leader}$. The values of c_1 and c_2 in the improved self-evolution operator are the same as their values in the improved crossover operator. The mathematical formula of this operator is written as Eq. (5).

$$\mathbf{x}_{new} = \mathbf{x}_i + rand(0,1) \times (\mathbf{x}_{leader} - \mathbf{x}_j) \times c_1 + rand(0,1) \times (\mathbf{x}_{leader} - \mathbf{x}_i) \times c_2 \quad (5)$$

Similar to the improved crossover operator, the new branch generated by the improved self-evolution operator is also required to dominate the original one. Then, a same branch update process as the improved crossover operator is executed for the improved self-evolution operator. The following pseudo code illustrates the process of the improved self-evolution operator to update the branch position $\mathbf{x}_i$.

Algorithm 3 Improved self-evolution operator

Set the trial number as 1
While the trial number $\le Tn$
Select the leader branch position $\mathbf{x}_{leader}$
Select the branch location $\mathbf{x}_j$ in the branch population
Calculate the new branch position $\mathbf{x}_{new} = \mathbf{x}_i + rand(0,1) \times (\mathbf{x}_{leader} - \mathbf{x}_j) \times c_1 + rand(0,1) \times (\mathbf{x}_{leader} - \mathbf{x}_i) \times c_2$
Calculate the solution of the new branch whose position is $\mathbf{x}_{new}$
Perform the Pareto dominance analysis between the new branch ($\mathbf{x}_{new}$) and the original branch ($\mathbf{x}_i$)
If the new branch ($\mathbf{x}_{new}$) dominates the original one ($\mathbf{x}_i$)
Replace the original branch ($\mathbf{x}_i$) with the new branch ($\mathbf{x}_{new}$)
break
else
Increment its trial by 1
End If
End While
If current branch is not updated
If $rand(0,1) \le h$

Replace the original branch ($\mathbf{x}_i$) with the last generated new branch ($\mathbf{x}_{new}$)

else

Replace the original branch ($\mathbf{x}_i$) with the randomly generated branch in the design space

End If

End If

2.3 Update archive

For each round of cycle, Pareto dominance analysis is applied in branch population to obtain branches that are not dominated by other branches. Then, these obtained non-dominated solutions are merged with previous non-dominated solutions, and a new Pareto dominance analysis is carried out with this merged solution set to acquire a new non-dominated solutions of current cycle. Next, these new non-dominated solutions replace the original non-dominated solutions in archive.

The maximum number of branch stored in archive is defined as *nBn*. In the optimization process, the number of branches in archive needs to be evaluated to avoid excessive branches. After the update of the non-dominated solutions, an assessment is carried out to judge whether the number of branches in archive (NBA) exceeds *nBn*. If NBA > *nBn*, these excess branches should be removed one by one. The concepts of grids, S-values and P-values of these branches in archive are defined to help determine which branch should be removed.

An adaptive grid method is introduced to manage these non-dominated solutions in archive and keep the diversity and convergence of these solutions. This theory draws on the idea of grids and individual management in adaptive grid approach [29], but differs in the method of removing excess branches. This new proposed method is conducive to maintaining the diversity and convergence of the non-dominated solutions. The detailed process of the adaptive grid method is shown as follows. In each round of optimization, when the branches within the archive are updated, the grids corresponding to the branches are also updated. The boundary of solutions in archive is obtained based on the $\boldsymbol{Y}^{\max}=(y_1^{\max},...,y_k^{\max},...,y_n^{\max})$ and $\boldsymbol{Y}^{\min}=(y_1^{\min},...,y_k^{\min},...,y_n^{\min})$, where $y_k^{\max}$ and $y_k^{\min}$ denote the maximum and minimum values of *k*-th objective of these non-dominated solutions, and *n* is the number of objectives. A parameter *N* is defined here to describe the number of grids for each objective, which means that the space of each objective can be divided into *N* parts. The grid size of *k*-th objective is $(y_k^{\max}-y_k^{\min})/N$, and a total of N^n grids is produced for the non-dominated solution set. A reasonable value of *N* is important. On the one hand, a large *N* can manage the branches more

finely. On the other hand, an excessive N value increases computational cost. Then, based on the objective values of these non-dominated solutions as well as the upper and lower boundaries of these grids, it is possible to determine which grids these non-dominated solutions fall into. These grids containing non-dominated solutions can be acquired. In addition, each branch in archive has n grids, which is equal to the number of objectives. The serial number of n grids of each branch can be obtained.

Based on the grid number of these non-dominated solutions, each branch in archive has a S value, and the S value is calculated by the following Eq. (6).

$$S_i = s_{i1} + s_{i2} + s_{ij} + \ldots + s_{in} \tag{6}$$

where s_{ij} is a serial number of grid of j-th objective of i-th branch, S_i is the sum of the serial number of all grids of i-th branch. Therefore, the S values of all branches in archive can be written as $\mathbf{S} = (S_1, S_2, \ldots, S_{Mn})$, where Mn is the number of branches in archive currently. In archive, the probability that the branches are selected is $\mathbf{P} = (P_1, P_2, \ldots, P_{Mn})$, and the mathematical model of the probability can be written as Eq. (7).

$$p_i = S_i \Big/ \sum_{1}^{Mn} S_i \tag{7}$$

where P_i is the probability that the i-th branch is selected, Mn is the number of branches in archive currently, and a smaller S_i corresponds to a smaller P_i.

After the renewal of the archive, a judgment needs to be performed to evaluate whether the number of branches in archive exceeds nBn. If NBA > nBn, the excess branches should be deleted one by one. The process of this operation is summarized as follows.

First, the non-dominated solutions in the same grid should be removed first to ensure the uniformity of the Pareto front. All the grids are checked to find these grids which contain more than one branch, and the branches in these grids are combined as a new set. The values of **S** and **P** of the new set are calculated.

Second, a roulette wheel selection is applied with the probability **P** to select one branch from the new set, and this selected branch will be removed from the archive.

Third, a new evaluation is applied to judge whether all the branches are in different grids or NBA = nBn. If NBA = nBn which means that the number of branches in archive is not excess nBn, and the process of deleting non-dominated solutions ends. If all the branches are in different grids

and NBA > *nBn*, end this operation and proceed to the fourth step. Otherwise, repeat the first, second and third steps.

Fourth, the values of **S** and **P** of all branches in archive are calculated, and the roulette wheel selection is directly carried out with **P** of all solutions to remove one branch from the archive. If NBA = *nBn*, the process of deleting non-dominated solutions ends. Otherwise, continue this fourth step until NBA = *nBn*. The following pseudo code shows how to update the archive.

Algorithm 4 Update archive

Perform the Pareto dominance analysis of current branch population
Obtain the non-dominated solution set
Combine the currently obtained non-dominated solution set with the non-dominated solution set in the archive
Perform the Pareto dominance analysis of the combined non-dominated solution set
Obtain the new non-dominated solution set
Replace the original non-dominated solution set in the archive with the new non-dominated solution set
Define the grids of the archive
While NBA > *nBn*
  If there are grids which contains more than one branch
    Combine the branches in these grids which contains more than one branch as a new set
    Calculate the values of **S** and **P** of the new set
    Perform the roulette wheel selection with the probability **P** to select one branch
    Remove the selected branch from the archive
  else
    Calculate the values of **S** and **P** of all branches in the archive
    Perform the roulette wheel selection with the probability **P** to select one branch
    Remove the selected branch from the archive
  End If
End While

2.4 Update the branch population

In MOAT, the initial branch number in population is *Bn*, and the branch number in population varies with the optimization process. The strategy of variable number of branches is applied to improve the efficiency of MOAT. The update process of the branch population is summarized as follows.

First, an evaluation is carried out to judge whether the branch number in population (BNP) exceeds *Bn*. If BNP > *Bn*, these branches in population which is also in the archive should be removed from the population.

Second, if the branch number in current population is still greater than *Bn*, these excessive branches should also be deleted. Then, the grids of the branch population are produced, and the

values of **S** of these branches are calculated based on Eq. (6). *Bn* branches with less S values are retained and the other branches are removed.

Third, the branch population and the non-dominated solutions in archive are merged together to be the new branch population for the next cycle of optimization. The following pseudo code shows the update process of the branch population.

Algorithm 5 Update the branch population

```
If BNP > Bn
    Remove the duplicate branches both in the population and archive from the population
    If BNP > Bn
        Calculate the values of S of all branches in the population
        Keep Bn branches with smaller S values in the population
    End If
End If
Combine current branch population and the branches in the archive as the new population
```

3 Numerical experiments

3.1 Ten typical test problems and seven compared algorithms

Numerical analyses are conducted to study the performance of MOAT fully, and computational results of MOAT are compared with NSGAII [7], MOEA/D [23], MOPSO [24], GDE3 [25], εMOEA [26], IBEA [27] and MPSO/D [28]. These algorithms are described as follows:

MOEA/D: This is a widely used decomposition-based multi-objective algorithm. A MOP is decomposed into a scale of sub-problems, and these sub-problems can be optimized simultaneously. Computational complexity of MOEA/D per generation is lower than NSGA-II and MOGLS. Experimental results show that MOEA/D is better than or similar with NSGA-II and MOGLS in typical MOPs. It has been demonstrated that MOEA/D can handle the MOPs effectively.

NSGAII: This algorithm was proposed by Deb at al. in 2002 which can overcome the main defects of NSGA. NSGA-II introduced a fast non-dominated sorting method with low computation complexity. In addition, a mating pool which is combined by parent and offspring populations is applied, and a selection operator is applied to choose the best solutions. Computational results on several typical test functions show that NSGA-II can find better solutions than other Pareto based algorithms.

MOPSO: This is an expanded algorithm of particle swarm optimization (PSO), which can be used to solve MOPs. The strategy of Pareto advantage is applied by MOPSO to decide the direction of flight

of particles. Previously discovered non-dominant vectors are kept in a repository, and these vectors will be applied by other particles to direct their own movements later. The efficiency of MOPSO is proved by the calculative results of some typical test functions.

GDE3: GDE3 is an improved version of generalized differential evolution (GDE) that can be used to solve the optimization problems with any number of constraints and objectives. For case without constraint and only one objective, GDE3 degenerates into DE. Compared with the early version of GDE, GDE3 can obtain the better distributed solutions.

εMOEA: εMOEA is an improved multi-objective evolutionary method based on ε-dominance strategy. In εMOEA, the archive is applied to store the non-dominated solutions. The space of the archive of each objective is divided into a series of grids, and the size of one grid of *i*-th objective is ε_i. The ε-dominant strategy does not allow two solutions to be in the same grid, which guarantees the diversity of solutions. Computational results acquired by εMOEA are compared with NSGA-II, C-NSGA-II, PESA and SPEA2 in terms of many typical test problems, and its high performance in solve the MOPs is demonstrated.

IBEA: This is an indicator-based multi-objective evolutionary algorithm proposed by Zitzler and Künzli. This algorithm integrates the preference information into a multi-objective algorithm to solve the MOPs. An optimization goal based on a binary indicator is first defined, and an indicator is then applied into the selection process. Compared to existing algorithms, IBEA can be adjusted according to user's preferences without using any additional diversity retention strategies. In terms of different indicators, IBEA can significantly improve the results calculated by NSGA-II and SPEA2 on several benchmarking functions.

MPSO/D: This is a new decomposition-based multi-objective algorithm. In MPSO/D, a direction vector is used to decompose the objective space of the MOPs into many sub-areas, and each sub-area contains one solution to keep the diversity. Then, a strategy of crowding distance is applied to compute the fitness value of saving solutions to select the better solutions, and the particle position of global best historical is determined by the neighboring particles of current particle. Experimental results show that MPSO/D performs better than NSGAII, MOEA/D and NNIA in terms of some typical test problems.

In MOAT, parameters *Bn*, *nBn*, *N*, *Tn*, c_1, c_2, *r* and *h* are set as 100, 100, 50, 5, 0.382, 0.618, 0.7 and 0.7, and parameters of other multi-objective algorithms are from references [7, 23-28]. Problems of

UF1 - UF4, UF7 [10, 14] and ZDT1 - ZDT6 [9] are applied to test the performances of these algorithms. Maximum number of function evaluation for ZDT and UF problems are set as 30,000 and 100,000, respectively. Thirty independent calculations are carried out on all questions with different random seeds. Means and standard deviations (SDs) of different metrics results are acquired. In order to increase the legibility of the results, the best and second best metric values are marked as dark gray and light gray colors. Furthermore, t-test is also carried out based on these metrics results to reduce familywise errors.

3.2 The performance metric

W is a set of points which are evenly distributed on the real Pareto front, and *A* is a set of points calculated by the multi-objective algorithm. Whether point set A is successful can be assessed by metrics. In this work, metrics SPREAD [14], IGD [14, 28] and HV [14, 28] are applied to evaluate the non-dominated solutions. Furthermore, in order to better assess the results, number of points in *W* set is defined as 100 for the two objective problems.

3.2.1. SPREAD

SPREAD shows the extent of spread of solution set *A*, and a small value of SPREAD indicates the fit of *A* and *W* is great. The SPREAD indicator is written as follows.

$$\mathrm{SPREAD} = \frac{\sum_{i=1}^{m} d(s_i, A) + \sum_{X \in A} | d(X, A) - \bar{d} |}{\sum_{i=1}^{m} d(s_i, A) + | A | \bar{d}} \tag{8}$$

where $s_1, \ldots, s_m$ are m uniformly distributed points in W, m is the number of objectives, $d(s_i, A)$ is the Euclidean distance between the set A *and* s_i, $d(X, A) = \min_{Y \in A, Y \neq X} \left\| X - Y \right\|^2$ and $\bar{d} = \frac{1}{|A|} \sum_{X \in A} d(X, A)$.

3.2.2. Inverted generational distance (IGD)

IGD indicator is a widespread metric to quantitatively evaluate the calculative results of multi-objective algorithms. The value of IGD is the average distance from *A* to *W*, and a smaller IGD indicates that the calculated solutions are closer to the true Pareto front. Therefore, a smaller IGD is better. The IGD indicator is written as follows.

$$\mathrm{IGD}(A, W) = \frac{\sum_{v \in W} d(v, A)}{| W |} \tag{9}$$

where $d(v, A)$ is the minimum Euclidian distance between v and A, and IGD(A, W) can effectively measure the diversity and convergence of solution set A. In order to obtain a small value of IGD(A,

W), the solution set *A* must be very close to *W* and cannot lose any part of the whole Pareto front.

3.2.3. Hypervolume (HV)

HV indicator represents the volume of a hypercube enclosed by the individuals in solution set *A* and the reference points in target space. HV indicator can evaluate the convergence, uniformity and extensive of solutions simultaneously, and give a comprehensive evaluation result. The value of HV can be calculated by Eq. (10).

$$\mathrm{HV} = Volume\left(\bigcup_{X \in A} [f_1(X), r_1] \times [f_2(X), r_2] \times ... \times [f_n(X), r_n] \right) \tag{10}$$

where $[f_1(\mathrm{X}), r_1] \times [f_2(\mathrm{X}), r_2] \times ... \times [f_n(\mathrm{X}), r_n]$ represents a hypercube surrounded by all points that are dominated by *X* and not dominated by reference point *Ref*, the reference point is defined as $Ref = (r_1, r_2, ..., r_n)$ and *n* is the number of objective of MOP. In this paper, the reference point is defined as $r_i = \max(W_i) \times 1.1, \ \ i = 1, 2, ..., n$, for all test problems.

3.3 Numerical results of these MOPs obtained by MOAT and seven comparison algorithms

Numerical results of these test functions are shown as follows. Table 1 is the IGD metric results obtained by MOAT and seven algorithms. Tables 2 and 3 show the SPREAD and HV metrics results, respectively. From Table 1, it is obvious that MOAT obtains the satisfactory IGD results on these benchmark functions compared with these well-known algorithms. MOAT yields better IGD values than the other algorithms for functions ZDT3 and ZDT4, and MPSO/D and NSGAII perform the second best for ZDT3 and ZDT4. MOAT performs the second best for functions ZDT1 and ZDT2, and MPSO/D and NSGAII obtain the best results of ZDT3 and ZDT4 in terms of the mean and SD values, respectively. Regarding function ZDT6, GDE3 and MPSO/D obtain the first and second ranks of IGD values among these algorithms, respectively. For functions UF3 and UF4, MOAT hits the first rank, and GDE3 and NSGAII obtain the second best results of these two problems. Regarding the problems UF1 and UF7, MOAT acquires the second satisfactory IGD values, and GDE3 obtains the best results of IGD metric. Finally, MOAT obtains the third satisfactory IGD values for UF2. In addition, the t-test for the IGD results of these functions at a 0.05 significance is performed, and symbolic results are also shown in Table 1. It is obvious that t-test results of MOAT are significantly better than those of the compared algorithms for questions ZDT3, ZDT4, UF3 and

UF4. Regarding functions ZDT1, ZDT2, UF1 and UF7, MOAT obtains 6 "+" among these 7 compared algorithms, which means the t-test results of MOAT on these four problems are significantly better than those of 6 algorithms. On functions ZDT6 and UF2, MOAT acquires 5 "+". In short, the number of best solutions obtained by MOAT is more than the other seven comparison algorithms obviously in terms of IGD metric. Therefore, the performance of MOAT is superior to other algorithms in current state. It proves the role of MOAT in solving MOPs.

Table 1. The IGD metric values of these typical MOPs obtained by eight algorithms.

		MOEAD	NSGAII	MOPSO	GDE3	εMOEA	IBEA	MPSO/D	MOAT
	Mean	0.011544	0.004642	36.113	10.219	0.042972	0.004538	0.003833	0.004143
ZDT1	SD	0.00864	0.000156	7.97	2.15	0.0211	0.000135	0.000014	0.000842
	t-test	+	+	+	+	+	+	-	
	Mean	0.053961	0.004741	43.405	14.974	0.098908	0.009483	0.007063	0.006672
ZDT2	SD	0.0738	0.000168	11	4.08	0.065	0.00103	0.00861	0.002793
	t-test	+	-	+	+	+	+	+	
	Mean	0.036232	0.050827	38.036	12.65	0.22224	0.096403	0.010273	0.002284
ZDT3	SD	0.032	0.0623	8.61	3.77	0.11	0.0336	0.000439	0.000423
	t-test	+	+	+	+	+	+	+	
	Mean	0.020878	0.0055	18.459	21.613	0.041074	0.098339	22.417	0.004251
ZDT4	SD	0.0142	0.000732	6.83	3.81	0.0324	0.127	4.77	0.001236
	t-test	+	+	+	+	+	+	+	
	Mean	0.007028	0.003739	0.47578	0.003107	0.029163	0.005061	0.003086	0.003126
ZDT6	SD	0.00109	0.000107	2.09	0.000019	0.0018	0.000182	0.000004	0.000914
	t-test	+	+	+	-	+	+	-	
	Mean	0.23881	0.11899	0.56451	0.03826	0.13820	0.12012	0.05932	0.0504024
UF1	SD	0.09040	0.03910	0.12700	0.00472	0.03000	0.03750	0.01410	0.074184
	t-test	+	+	+	-	+	+	+	
	Mean	0.18788	0.04999	0.11617	0.02933	0.09057	0.05375	0.01898	0.0305294
UF2	SD	0.05460	0.02720	0.00823	0.00370	0.01960	0.01850	0.00318	0.005768
	t-test	+	+	+	-	+	+	-	
	Mean	0.31841	0.23476	0.55044	0.07539	0.27498	0.28417	0.12454	0.0151431
UF3	SD	0.03210	0.05270	0.02040	0.02210	0.05170	0.04170	0.01510	0.003502
	t-test	+	+	+	+	+	+	+	
	Mean	0.12608	0.07475	0.11394	0.08174	0.10329	0.07582	0.08425	0.031064
UF4	SD	0.00578	0.00367	0.01520	0.00697	0.00506	0.00380	0.00423	0.009686
	t-test	+	+	+	+	+	+	+	
	Mean	0.33412	0.15200	0.65441	0.01755	0.24000	0.12422	0.03779	0.0371684
UF7	SD	0.20000	0.14600	0.12000	0.00130	0.14200	0.12700	0.06820	0.026581
	t-test	+	+	+	-	+	+	+	

Note: "+" and "-" mean the t-test result of MOAT is significantly better than and worse than that of the corresponding algorithm.

The SPREAD metric results from Table 2 further confirm the high performance of MOAT.

From this table, it can be seen that MOAT performs superior to other algorithms on problems ZDT3 and ZDT4. These indicate that solutions obtained by MOAT are closer to the real Pareto frontier than the compared algorithms. MOAT gets the second best SPREAD values for ZDT1 and ZDT6, and its results are only slightly worse than those of MPSO/D. In addition, MOAT also hits the second ranks compared with the other algorithms for problems UF3 and UF4. On problems ZDT2 and UF7, the third best SPREAD values are achieved by MOAT. Besides these numerical results, seven pairs of t-test at a significant level of 0.05 are carried out for the SPREAD results. Symbolic results are shown in Table 2. From Table 2, MOAT obtains 7 "+" for problems ZDT3 and ZDT4, which means the t-test results of MOAT are significantly better than those of other compared algorithms for these two problems. Regarding functions ZDT1, ZDT6, UF3 and UF4, MOAT obtains the symbol "+" for 6 times among these 7 algorithms. Furthermore, MOAT obtains 5 "+" for problem UF7 and 4 "+" for function UF1. Therefore, based on these numerical and symbolic results, MOAT can find the satisfactory SPREAD results on these problems. In addition, the number of top solutions of SPREAD metric obtained by MOAT is significantly more than the other algorithms. This proves the performance of MOAT.

Table 2. The SPREAD metric results acquired by all the algorithms.

		MOEAD	NSGAII	MOPSO	GDE3	εMOEA	IBEA	MPSO/D	MOAT
	Mean	0.45478	0.43379	NaN	0.94325	0.62634	0.36663	0.30945	0.327734
ZDT1	SD	0.166	0.0437	NaN	0.0308	0.283	0.0389	0.0117	0.0357808
	t-test	+	+	+	+	+	+	-	
	Mean	0.79831	0.44945	NaN	NaN	0.66391	0.38685	0.23368	0.394276
ZDT2	SD	0.418	0.0525	NaN	NaN	0.343	0.0462	0.17	0.022266
	t-test	+	+	+	+	+	-	-	
	Mean	0.71789	0.59851	NaN	0.96087	0.99317	0.76407	0.63676	0.521499
ZDT3	SD	0.079	0.108	NaN	0.0242	0.203	0.0144	0.0294	0.059418
	t-test	+	+	+	+	+	+	+	
	Mean	0.61373	0.42543	NaN	1.0384	1.4622	1.2262	NaN	0.402516
ZDT4	SD	0.289	0.0521	NaN	0.0953	0.65	0.0832	NaN	0.0267498
	t-test	+	+	+	+	+	+	+	
	Mean	0.16215	0.41357	1.208	0.80066	0.89074	0.41003	0.14932	0.1552
ZDT6	SD	0.0272	0.0418	0.321	0.757	0.602	0.128	0.0035	0.019087
	t-test	+	+	+	+	+	+	-	
	Mean	1.0024	0.85976	0.79714	0.23525	1.2657	1.036	1.03602	0.9392696
UF1	SD	0.0413	0.166	0.0651	0.0479	0.307	0.0408	0.0692	0.097094
	t-test	+	-	-	-	+	+	+	
UF2	Mean	0.59899	0.5502	0.78695	0.2044	0.59641	1.0519	0.43134	1.0079474

	SD	0.0696	0.0643	0.0414	0.0336	0.154	0.0552	0.0431	0.133458
	t-test	-	-	-	-	-	+	-	
	Mean	1.0003	1.0465	0.90031	0.2705	NaN	1.012	0.79576	0.3999989
UF3	SD	0.00328	0.0767	0.0541	0.0566	NaN	0.0528	0.0934	0.03556
	t-test	+	+	+	-	+	+	+	
	Mean	0.46722	0.45009	0.82735	0.23608	0.47898	1.2324	0.50114	0.4450333
UF4	SD	0.0702	0.0572	0.0571	0.02	0.0754	0.126	0.0474	0.130324
	t-test	+	+	+	-	+	+	+	
	Mean	0.87755	0.74835	0.82653	0.18972	0.75444	1.1095	0.37331	0.7155088
UF7	SD	0.271	0.204	0.0607	0.0327	0.305	0.124	0.182	0.197991
	t-test	+	+	+	-	+	+	-	

Note: "+" and "-" mean the t-test result of MOAT is significantly better than and worse than that of the corresponding algorithm.

Table 3 shows the numerical results of HV metric. From Table 3, it is clear that MOAT obtains better HV results on functions ZDT3, UF3 and UF4 compared with the other algorithms, and MPSO/D, GDE3 and NSGAII acquire the second best results for these problems, respectively. On problems ZDT2, ZDT4, ZDT6 and UF1, MOAT performs the second best. NSGAII hits the optimum solutions of problems ZDT2 and ZDT4, and the maximum HV values of functions ZDT6 and UF1 are achieved by GDE3. In addition, MOAT also achieves the third best HV values for functions ZDT1 and UF7. Table 3 also shows the t-test results of HV metric at a 0.05 significance level. From Table 3, MOAT acquires 7 "+" among these 7 algorithms for functions ZDT3, UF3 and UF4. Regarding problems ZDT2, ZDT4, ZDT6 and UF1, MOAT acquires 6 "+". Finally, on functions ZDT1 and UF7, MOAT acquires 5 "+". The calculative results of these algorithms show that MOAT can achieve a satisfied HV values on these test problems. Based on the number of top solutions of HV metric acquired by each algorithm, a better performance of MOAT than the other seven algorithms in current state is proved. MOAT is demonstrated to be a much efficient algorithm for solving the MOPs.

Table 3. The HV metric results obtained by MOAT and seven comparison algorithms.

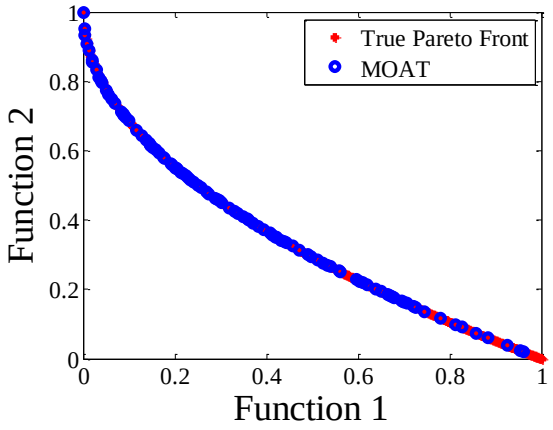
		MOEAD	NSGAII	MOPSO	GDE3	εMOEA	IBEA	MPSO/D	MOAT
	Mean	0.86084	0.86049	0.00000	0.00000	0.81224	0.87111	0.87088	0.8693054
ZDT1	SD	0.00798	0.00020	0.00000	0.00000	0.01930	0.00013	0.00007	0.0029704
	t-test	+	+	+	+	+	-	-	
	Mean	0.46671	0.53737	0.00000	0.00000	0.40947	0.53601	0.53381	0.5364726
ZDT2	SD	0.08120	0.00021	0.00000	0.00000	0.07210	0.00269	0.01060	0.1071984
	t-test	+	-	+	+	+	+	+	
ZDT3	Mean	0.97748	0.96758	0.00000	0.00000	0.77030	0.91544	1.01910	1.0227411
	SD	0.04140	0.07960	0.00000	0.00000	0.13900	0.04750	0.00052	0.0223882

	t-test	+	+	+	+	+	+	+	
	Mean	0.84144	0.86746	0.00000	0.00000	0.80795	0.79440	0.00000	0.8639084
ZDT4	SD	0.01600	0.00189	0.00000	0.00000	0.04430	0.09150	0.00000	0.0019281
	t-test	+	-	+	+	+	+	+	
	Mean	0.42519	0.43226	0.40390	0.43342	0.39886	0.43135	0.43237	0.4325559
ZDT6	SD	0.00186	0.00035	0.09530	0.00005	0.00361	0.00027	0.00001	0.0003616
	t-test	+	+	+	-	+	+	+	
	Mean	0.57232	0.70091	0.17444	0.80983	0.67169	0.71560	0.76212	0.76370
UF1	SD	0.08420	0.05640	0.10300	0.00899	0.03670	0.03540	0.02610	0.02062
	t-test	+	+	+	-	+	+	+	
	Mean	0.73370	0.81686	0.70968	0.83179	0.74481	0.81453	0.84659	0.78240
UF2	SD	0.03710	0.01710	0.01050	0.00424	0.01760	0.01380	0.00387	0.01545
	t-test	+	-	+	-	+	-	-	
	Mean	0.45611	0.54515	0.16164	0.75211	0.49231	0.53222	0.69552	0.85960
UF3	SD	0.05440	-0.05890	-0.01990	0.03370	-0.05640	0.04570	0.01670	0.01118
	t-test	+	+	+	+	+	+	+	
	Mean	0.32694	0.41567	0.34948	0.40046	0.35909	0.41129	0.39624	0.49370
UF4	SD	0.00849	0.00676	0.02280	0.01130	0.00938	0.00509	0.00722	0.00430
	t-test	+	+	+	+	+	+	+	
	Mean	0.38788	0.54326	0.04926	0.67680	0.44990	0.56834	0.65444	0.62180
UF7	SD	0.15700	0.12500	0.04850	0.00270	0.10100	0.10900	0.06340	0.01060
	t-test	+	+	+	-	+	+	-	

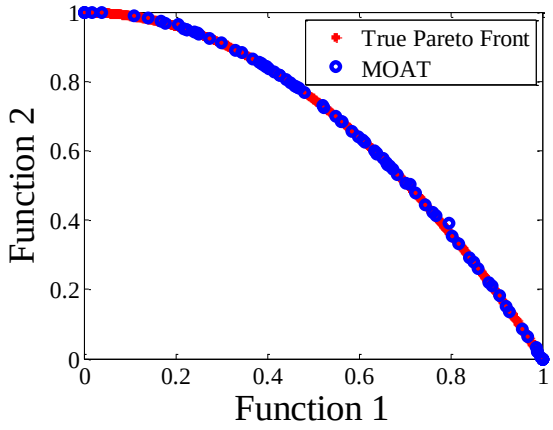
Note: “+” and “-” mean the t-test result of MOAT is significantly better than and worse than that of the corresponding algorithm.

In short, regarding these 10 benchmark functions, MOAT yields better numerical and t-test results of IGD metric than those of algorithms MOEAD, NSGAII, MOPSO, GDE3, εMOEA,IBEA and MPSO/D on problems 10, 9, 10, 6, 10, 10 and 7 respectively. The better rates of MOAT on these functions are 100%, 90%, 100%, 60%, 100%, 100% and 70%, respectively. Regarding SPREAD metric, the numerical and symbolic results of MOAT are better than those of the seven compared algorithms on problems 8, 8, 8, 5, 9, 9 and 5, respectively, and the better ratios of MOAT on these functions are 80%, 80%, 80%, 50%, 90%, 90% and 50%, respectively. In addition, the numbers of better HV metric results obtained by MOAT over algorithms MOEAD, NSGAII, MOPSO, GDE3, εMOEA, IBEA and MPSO/D are 10, 7, 10, 6, 10, 8 and 7, and the better ratios of MOAT on these functions are 100%, 70%, 100%, 60%, 100%, 80% and 70%, respectively. Based on the number of the best solutions (numerical and symbolic results) on IGD, SPREAD and HV metrics, the performance of MOAT is clearly proved. It can be concluded that MOAT exhibits stable and great performance for these test problems.

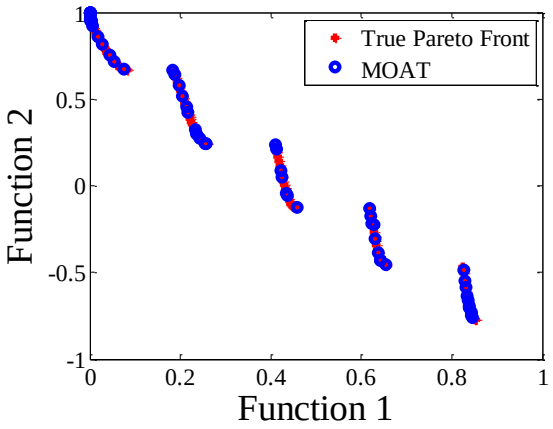
Fig. 3 shows the plots of the non-dominated solutions calculated by MOAT and the real Pareto fronts of these MOPs. It is obvious that these non-dominated solutions approximate the true Pareto fronts very closely. Especially for functions ZDT1 - ZDT6, the non-dominated solutions are almost the same as the ideal Pareto fronts with high density of solutions. On functions UF2, UF3, UF4 and UF7, MOAT obtains a number of continuous Pareto fronts which converge very closely to the ideal Pareto fronts. This shows that the solutions obtained by MOAT have great diversity and convergence. Based on the results of Fig. 3, the high performance of MOAT is proved again.



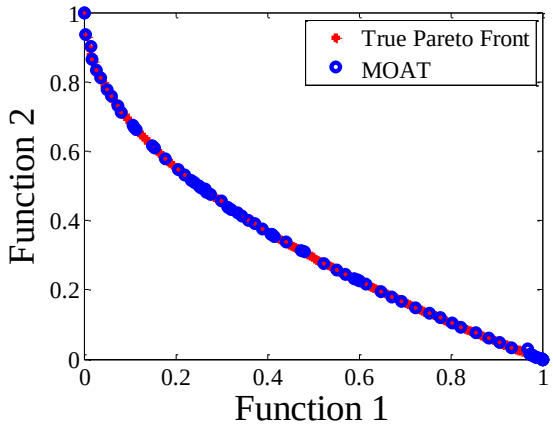
ZDT1



ZDT2



ZDT3



ZDT4

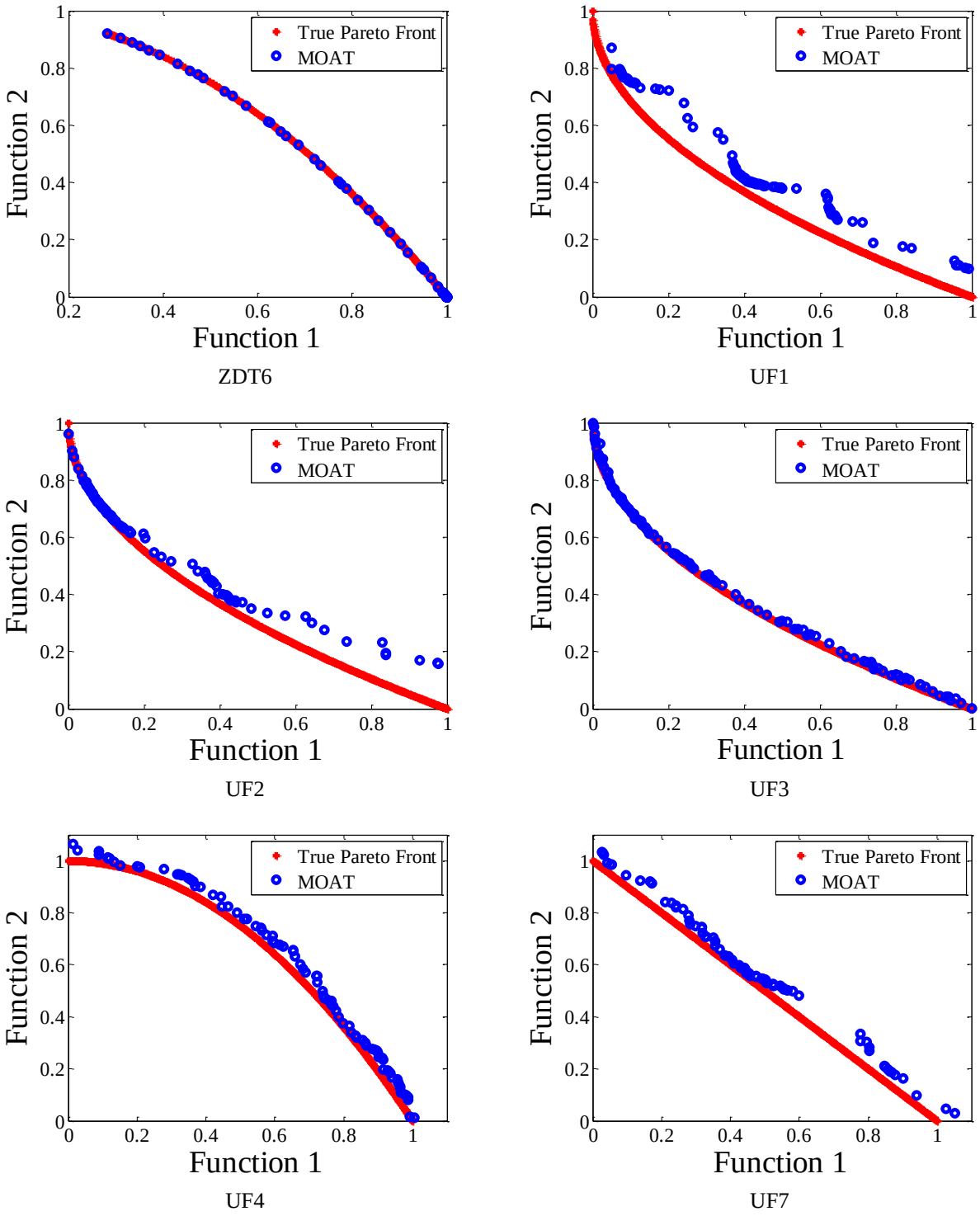


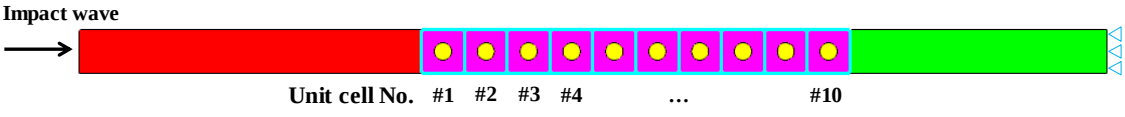
Fig. 3. Plots of the real Pareto fronts of these test functions and the non-dominated solutions obtained by MOAT.

4. Optimize the two-dimensional AMs based on MOAT

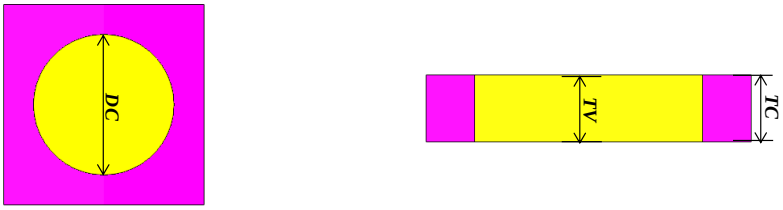
4.1 The description of the two-dimensional AM

In this section, the attenuation effect of AMs on impact wave is studied. Fig. 4 (a) shows one two-dimensional AM model, and the AM is located between the red and green parts. The impact

wave is applied on the front end of the red component, and the end of the green part is constrained by its six degrees of freedom. Entire AM consists of three parts, which are external framework, coating and vibrators. As shown in Fig. 4 (a), the blue structure is the framework, the yellow circular structure is the vibrator and the pink structure is the coating. The detailed size of AM is illustrated in Fig. 4 (b). The thickness of coating (*TC*), the thickness of vibrator (*TV*) and the diameter of vibrator (*DV*) are 4.0mm, 4.0mm and 4.0mm, and the materials of vibrator and coating are copper and rubber. The impact analysis is carried out with commercial software LSDYNA. Constitutive models MAT_PIECEWISE_LINEAR_PLASTICITY and MAT_OGDEN_RUBBER in the material library of LSDYNA are applied to simulate the metal oscillators and rubber coatings. The detailed material information of AM is illustrated in Tables 4 and 5.



(a)



(b)

Fig. 4. (a) The impact model with the two-dimensional AM. (b) The detailed size of one unit cell.

Table 4. Material parameters of the metal oscillators.

Material	*E* (Gpa)	ν	*ρ* (kg/m3)
Lead	40.8	0.37	11600
Steel	207.0	0.30	7784
Copper	110.0	0.33	8900

Table 5. Material parameters of rubber.

Material	*E* (Gpa)	ν	*ρ* (kg/m3)	*N*	*NV*	*SGL*	*SW*	*ST*
Rubber	0.0001175	0.496	1200	5	6	1	1	1

4.2 Impact results of the AM and original models

In order to better study the attenuation effect of AM, two indicators which are the mean impact force (*MF*) and the peak impact force (*PF*), are applied to measure the mechanical behavior of the structure. To calculate *MF*, the impulse (*I*) should be calculated first. The Impulse (*I*) is the integral of impact force with time, which can be written as follows.

$$I(t)=\int_0^t F(x)dx \tag{11}$$

where t stands for the time and F is the impact force. Then, MF can be calculated as follows.

$$MF=I/t \tag{12}$$

The impact results of the original and AM models are shown in Fig. 5 and Table 6. From Fig. 5 and Table 6, the metamaterial mass (MM) is 4.33 g, the values of PF of the original and AM models are 477.5 N and 348.2 N, and the values of MF of the original and AM models are 96.6 N and 90.6N. Therefore, the attenuation effect of impact stress waves achieved by the AM is clearly proved.

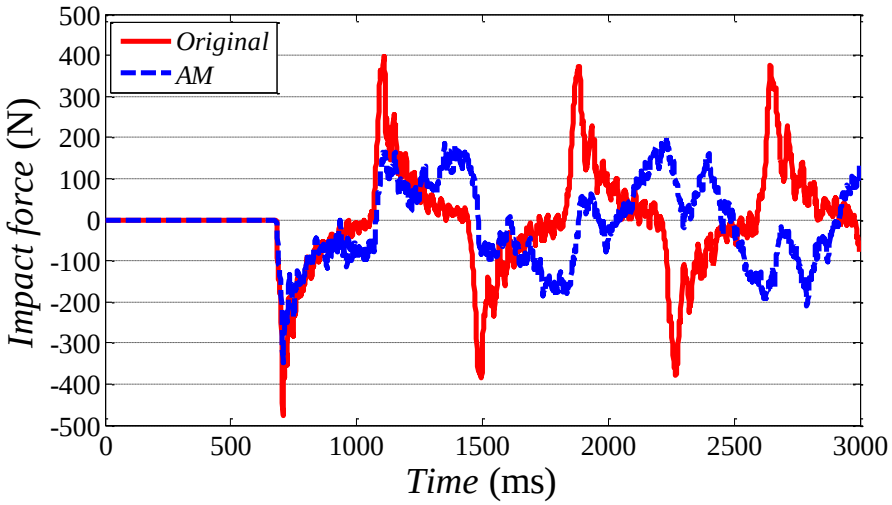


Fig. 5. Impact forces of the original and AM models.

Table 6. Results of the original and AM models.

	PF (N)	*MF* (N)	*MM* (g)
Original	477.5	96.6	——
AM	348.2	90.6	4.33

4.3 Optimize the two-dimensional AMs with MOAT, response surface and Latin hypercube

In order to speed up the optimization process, the third-order response surface methodology and the Latin hypercube method are applied. In addition, some polynomial terms with low significance are removed to improve the reliability of the approximation models and reduce the number of required design points. The total number of terms of the reduced three-order response surface model (RSM) is thirteen, and the function expression of the reduced RSM is written as follows.

$$F(x)= a_0+\sum_{i=1}^{N} b_i x_i+\sum_{i=1(i<j)}^{j\le N} c_{ij}x_i x_j+\sum_{i=1}^{N} d_i x_i^2+\sum_{i=1}^{N} e_i x_i^3 \tag{13}$$

where $F(x)$ is the value of RSM, a_0, b_i, c_{ij}, d_i and e_{ij} are the adjustable coefficients which are used to reduce the error between analysis and surrogate model, and N is the number of design variables.

It is well known that a better attenuation effect requires a heavier structure. However, the excessive mass is also a key factor which restricts the application of AM. In this paper, we want to design the key parameters of AM to enhance its mitigation effect and reduce its structural mass. The geometric parameters *TC*, *TV* and *DV* as well as the vibrator materials of AMs are optimized in order to minimize *PF*, *MF* and *MM*. The candidate vibrator materials are copper, steel and lead. The detailed material properties of these candidate materials are shown in Table 4. The design spaces of *TC*, *TV* and *DV* are shown in Table 7.

The mathematical models of the MOP that aims to minimize *PF* and *MM* as well as *MF* and *MM* are expressed in Eqs. (14) and (15).

$$\begin{cases} \text{Min} \quad PF,\ MM \\ \text{s.t.} \\ \quad 0 < DC < 9 \\ \quad 0 < TV < 10 \\ \quad 1 < TC < 10 \end{cases} \tag{14}$$

$$\begin{cases} \text{Min} \quad MF,\ MM \\ \text{s.t.} \\ \quad 0 < DC < 9 \\ \quad 0 < TV < 10 \\ \quad 1 < TC < 10 \end{cases} \tag{15}$$

Table 7. The design space of the variables.

	Low limit (mm)	Upper limit (mm)
TC	1	10
TV	1	10
DC	1	9

A total of forty samples are obtained, and the RSMs of AMs with copper, steel and lead oscillators are established in this work, respectively. The RSMs of responses *PF*, *MF* and *MM* of copper AM are written as follows.

$$\begin{aligned} PF = {} & 473.970 - 13.350x_1 - 10.699x_2 - 53.627x_3 + 4.267x_1^2 + 1.642x_2^2 + 4.580x_3^2 \\ & - 1.004x_1x_2 - 0.272x_1x_3 - 0.951x_2x_3 - 0.279x_1^3 - 0.011x_2^3 - 0.095x_3^3 \end{aligned} \tag{16}$$

$$\begin{aligned} MF = {} & 123.191 - 3.518x_1 - 0.492x_2 - 9.138x_3 - 0.294x_1^2 - 0.220x_2^2 + 1.176x_3^2 \\ & - 0.213x_1x_2 + 0.372x_1x_3 - 0.128x_2x_3 + 0.034x_1^3 + 0.024x_2^3 - 0.067x_3^3 \end{aligned} \tag{17}$$

$$\begin{aligned} MM = {} & 2.095 - 1.338x_1 + 0.395x_2 + 0.593x_3 - 0.099x_1^2 - 0.213x_2^2 + 0.117x_3^2 \\ & + 0.609x_1x_2 - 0.043x_1x_3 - 0.032x_2x_3 + 0.031x_1^3 + 0.013x_2^3 - 0.006x_3^3 \end{aligned} \tag{18}$$

Regarding the steel AM, its responses of *PF*, *MF* and *MM* are written as Eqs. (19) - (21).

$$\begin{aligned} PF = {} & 471.683 - 12.404x_1 - 5.963x_2 - 57.987x_3 + 3.900x_1^2 + 0.741x_2^2 + 5.763x_3^2 \\ & - 0.901x_1x_2 - 0.269x_1x_3 - 1.011x_2x_3 - 0.254x_1^3 + 0.036x_2^3 - 0.171x_3^3 \end{aligned} \tag{19}$$

$$\begin{aligned} MF = {} & 127.782 - 4.402x_1 + 0.477x_2 - 12.817x_3 + 0.037x_1^2 - 0.447x_2^2 + 1.882x_3^2 \\ & - 0.220x_1x_2 + 0.412x_1x_3 - 0.113x_2x_3 + 0.004x_1^3 + 0.039x_2^3 - 0.107x_3^3 \end{aligned} \tag{20}$$

$$\begin{aligned} MM = {} & 1.741 - 1.108x_1 + 0.371x_2 + 0.651x_3 - 0.094x_1^2 - 0.190x_2^2 + 0.109x_3^2 \\ & + 0.533x_1x_2 - 0.048x_1x_3 - 0.029x_2x_3 + 0.027x_1^3 + 0.011x_2^3 - 0.006x_3^3 \end{aligned} \tag{21}$$

The RSMs of the lead AM are illustrated as Eqs. (22) - (24).

$$\begin{aligned} PF = {} & 475.793 - 20.757x_1 - 13.988x_2 - 42.873x_3 + 5.466x_1^2 + 2.161x_2^2 + 1.889x_3^2 \\ & - 0.869x_1x_2 + 0.020x_1x_3 - 0.825x_2x_3 - 0.359x_1^3 - 0.044x_2^3 + 0.067x_3^3 \end{aligned} \tag{22}$$

$$\begin{aligned} MF = {} & 122.924 - 9.119x_1 + 0.243x_2 - 5.015x_3 + 1.020x_1^2 - 0.478x_2^2 + 0.287x_3^2 \\ & - 0.160x_1x_2 + 0.352x_1x_3 - 0.083x_2x_3 - 0.055x_1^3 + 0.040x_2^3 - 0.015x_3^3 \end{aligned} \tag{23}$$

$$\begin{aligned} MM = {} & 2.953 - 1.900x_1 + 0.463x_2 + 0.451x_3 - 0.111x_1^2 - 0.273x_2^2 + 0.137x_3^2 \\ & + 0.795x_1x_2 - 0.031x_1x_3 - 0.040x_2x_3 + 0.040x_1^3 + 0.016x_2^3 - 0.007x_3^3 \end{aligned} \tag{24}$$

It is worth noting that only the validated RSMs can ensure the accuracy of the metamodel-based optimization solutions. Therefore, a index R-square (R^2) [30] is used to evaluate the fitting accuracy of RSM, and a larger R^2 value means a better fitting between the predicted and true values. The formula of R^2 is written as follows.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y}_i)^2} \tag{25}$$

where y_i is the true value of the *i*th sample point, $\hat{y}_i$ is the predicted value of the *i*th sample point obtained by the approximate model, $\overline{y}_i$ is the mean of y_i, and n is the number of test points.

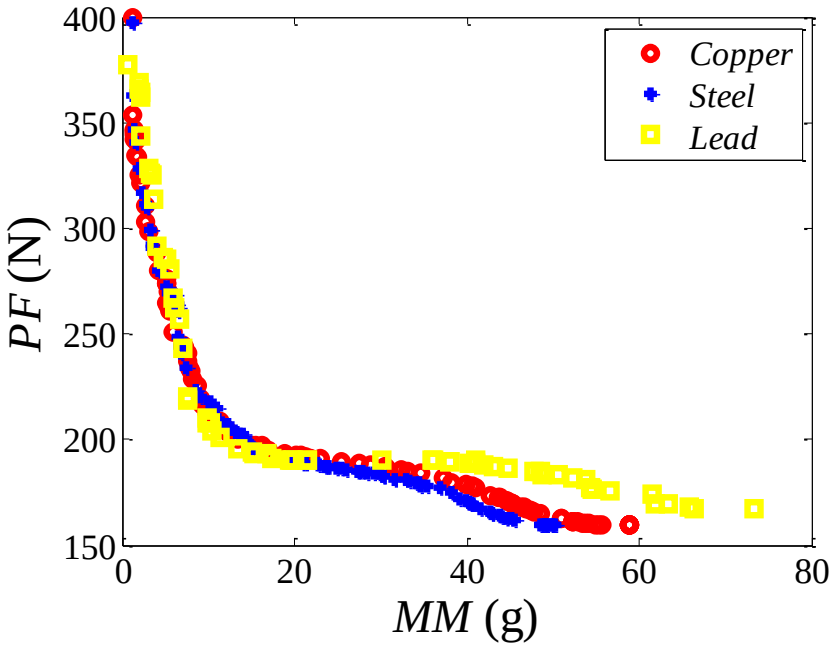
For these three kinds of AMs, the verification of their RSMs is performed with 10 random samples in the design space. The results of R^2 are shown in Table 8. From Table 8, the values of R^2 of *MM*, *PF* and *MF* are 0.978, 0.925 and 0.921 for copper AM, 0.978, 0.930 and 0.914 for steel AM as well as 0.977, 0.942 and 0.911 for lead AM. Obviously, the values of R^2 of these objectives are close to 1. Therefore, the accuracy of these RSMs meets the requirements, and they can be used for the further optimization design.

Table 8. Error analyses of these RSMs.

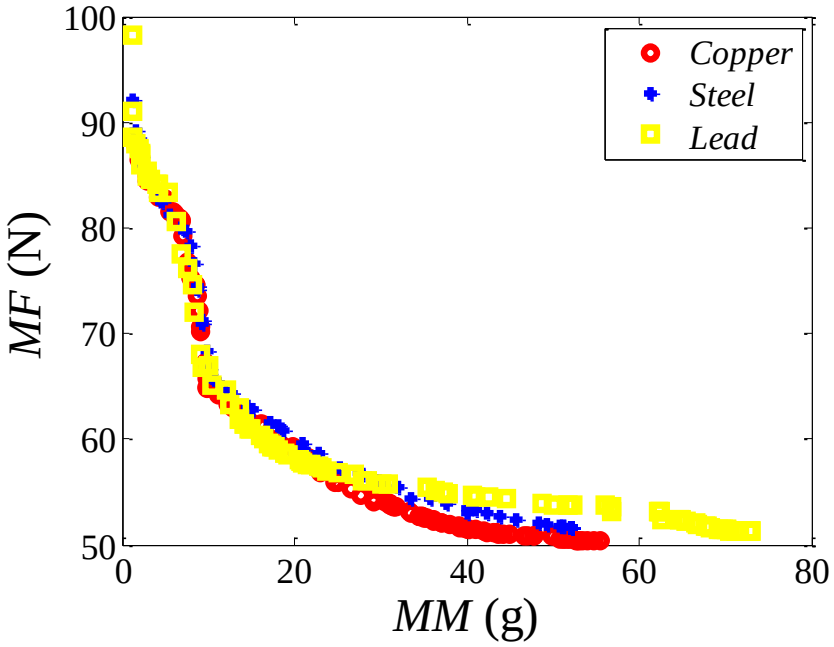
	Copper	Steel	Lead
M	0.978	0.978	0.977
PF	0.925	0.930	0.942
MF	0.921	0.914	0.911

MOAT is employed to gain the optimum solutions to satisfy the maximum requirements of all objectives, which yields a set of Pareto-solutions. Fig. 6 shows the Pareto fronts *PF* - *MM* and *MF* - *MM* of the copper, steel and lead AMs. Regarding Pareto fronts *PF* - *MM* and *MF* - *MM*, the optimum values of *PF* and *MM* as well as *MF* and *MM* strongly conflict with each other. In other words, any improvement in one objective must sacrifice the other one. This suggests that MOAT successfully generates a widely distributed Pareto-optimum solutions, and the Pareto fronts can provide many effective choices and keep the balance between these objectives.

Based on these Pareto fronts of the copper, steel and lead AMs in Fig. 6, some interesting conclusions are obtained. (1) With the increase of *MM*, the values of *PF* and *MF* of these three AMs are reduced quickly first and then their reduction rates decrease. Especially for the lead AM, its values of *PF* and *MF* are reduced more slowly than the copper and steel AMs in the high *MM* region. (2) When *MM* is less than 30 g, the performances of these three AMs are similar to each other regarding the responses of *PF* and *MF*. (3) The densities of oscillators have less influence on the performances of these AMs when *MM* is less than 30 g. (4) When *MM* is larger than 30 g, the attenuation effects of copper and steel AMs are better than those of lead AM.



(a)



(b)

Fig. 6. Pareto fronts of the responses (a) *PF* and *MM*, (b) *MF* and *MM* with these three kinds of AM models.

Since different AMs have different non-dominated solutions, a uniform optimum solution selection criterion is needed. In this work, the minimum distance theory [31], which can balance the objectives to maximize revenue, is applied to obtain the optimum solutions of problems *PF* - *MM* and *MF* - *MM*.

$$D = \left(\sum_{i=1}^{n}\left(\frac{f_{ij} - \min(f_i(x))}{\max(f_i(x)) - \min(f_i(x))}\right)^2\right)^{\frac{1}{2}} \tag{26}$$

where D is the minimum distance between target point and reference point, and D should be as small as possible. n is the number of optimization objectives, f_{ij} is the j-th solution of the i-th objective, and $\min(f_i(x))$ and $\max(f_i(x))$ are the minimum and maximum solutions of the i-th objective. Based on Eq. (26), the best solutions of problems *PF-M* and *MF-M* for the copper, steel and lead AMs are shown in Tables 9 and 10.

Regarding *PF-MM*, the optimum solutions of the copper, steel and lead AMs are 208.78 N - 10.79 g, 209.42 N - 11.10 g and 200.97 N - 11.29 g, respectively. For *MF-MM*, the optimum solutions of these AMs are 60.04 N - 17.20 g, 64.31 N - 12.64 g and 60.90 N - 14.94 g, respectively. It is clear that the optimum solutions of these AMs are different from each other, which provides us a variety of options. Finally, through quantitative analyses of the structural mass and attenuation effects of AMs, the optimum solution of the copper AM is selected to attenuate *PF*, and the optimum

solution of lead AM is selected for the mitigation of *MF*.

Table 9. The optimum solutions of *PF-M* for copper, steel and lead AMs.

	Copper	Steel	Lead
M (g)	10.79	11.10	11.29
PF(N)	208.78	209.42	200.97
DV (mm)	1.972	1.732	1.940
TV (mm)	6.081	7.291	6.773
TC (mm)	7.912	8.769	7.916

Table 10. The optimum solutions of *MF-M* for copper, steel and lead AMs.

	Copper	Steel	Lead
M (g)	17.20	12.64	14.94
MF(N)	60.04	64.31	60.90
DV (mm)	3.079	1.662	2.199
TV (mm)	8.554	10.000	8.671
TC (mm)	10.000	10.000	10.000

5 Conclusion

The No Free Lunch Theorem reveals that no algorithm can solve all MOPs efficiently, and new optimization algorithms are always followed by scholars. In this work, a new algorithm named the multi-objective artificial tree (MOAT) algorithm is proposed. It is an improved version of the basic AT algorithm. In order to extend the basic AT algorithm to a multi-objective algorithm, the bio-inspired model of AT is changed from a general tree to a banyan. Two improved branch update operators, an adaptive grid method and a variable branch number strategy, are introduced to ensure the performance of MOAT.

Many typical test problems are applied to test MOAT. Experimental results of MOAT are compared with MOEA/D, NSGAII, MOPSO, GDE3, εMOEA, IBEA and MPSO/D to prove its performance. The test results of IGD, SPREAD and HV performance metrics show that MOAT exhibits better performance than the other seven comparison algorithms. Therefore, it documents that MOAT is a bio-inspired multi-objective algorithm with strong competition. In addition, MOAT is also applied to optimize the two-dimensional acoustic metamaterials (AMs) to maximize their attenuation effects and minimize their structural masses. Finally, the non-dominated solutions obtained by MOAT provide us many effective choices, which can keep the balance between these optimization objectives, and the optimum solutions we chose improve the performances of AMs significantly.

Although this work presents a new algorithm with high computational efficiency and accuracy,

and based on this new algorithm to solve the MOPs of AMs efficiently, this paper has some limitations in the study of new types of Pareto fronts. In future work, we plan to work on the diversity of non-dominated solutions and the diversity of Pareto fronts with MOAT.

References

1. Hammami M, Bechikh S, Hung CC, Said LB. **A Multi-objective hybrid filter-wrapper evolutionary approach for feature selection**. *Memetic Computing* 2018:1-16.

2. Li H, Lei W, Hei X, Wei L, Jiang Q. **A decomposition-based chemical reaction optimization for multi-objective vehicle routing problem for simultaneous delivery and pickup with time windows**. *Memetic Computing* 2018; 10(1):103-120.

3. Qiu J, Liu M, Zhang L, Li W, Cheng F. **A multi-level knee point based multi-objective evolutionary algorithm for AUC maximization**. *Memetic Computing* 2019; (15):1-12.

4. Wang Z, Jin X, Li Q, Sun G. **On crashworthiness design of hybrid metal-composite structures**. *International Journal of Mechanical Sciences* 2020; 171:105380.

5. Schaffer JD. **Multiple Optimization with Vector Evaluated Genetic Algorithms**. In: *International Conference on Genetic Algorithms*; 1985. pp. 93-100.

6. Zitzler E, Laumanns M, Thiele L. **SPEA2: Improving the strength Pareto evolutionary algorithm**. *TIK-report* 2001; 103.

7. Deb K, Pratap A, Agarwal S, Meyarivan T. **A fast and elitist multiobjective genetic algorithm: NSGA-II**. *IEEE transactions on evolutionary computation* 2002; 6(2):182-197.

8. Sun J, Zhang H, Zhou A, Zhang Q, Zhang K. **A new learning-based adaptive multi-objective evolutionary algorithm**. *Swarm and Evolutionary Computation* 2019; 44:304-319.

9. Qiao J, Li F, Yang S, Yang C, Li W, Gu K. **An adaptive hybrid evolutionary immune multi-objective algorithm based on uniform distribution selection**. *Information Sciences* 2019.

10. Luo J, Liu Q, Yang Y, Li X, Chen M-r, Cao W. **An artificial bee colony algorithm for multi-objective optimisation**. *Applied Soft Computing* 2017; 50:235-251.

11. Ou J, Zheng J, Ruan G, Hu Y, Zou J, Li M, et al. **A pareto-based evolutionary algorithm using decomposition and truncation for dynamic multi-objective optimization**. *Applied Soft Computing* 2019:105673.

12. Lin Q, Zhu Q, Wang N, Huang P, Wang W, Chen J, et al. **A multi-objective immune algorithm with dynamic population strategy**. *Swarm and Evolutionary Computation* 2019; 50:100477.

13. Nebro AJ, Durillo JJ, Garcia-Nieto J, Coello CC, Luna F, Alba E. **Smpso: A new pso-based metaheuristic for multi-objective optimization**. In: *Computational intelligence in miulti-criteria decision-making, 2009 mcdm'09 ieee symposium on*: IEEE; 2009. pp. 66-73.

14. Luo J, Yang Y, Liu Q, Li X, Chen M, Gao K. **A new hybrid memetic multi-objective optimization algorithm for multi-objective optimization**. *Information Sciences* 2018; 448-449:164-186.

15. Wolpert DH, Macready WG. **No free lunch theorems for optimization**. *IEEE transactions on evolutionary computation* 1997; 1(1):67-82.

16. Li QQ, Song K, He ZC, Li E, Cheng AG, Chen T. **The artificial tree (AT) algorithm**. *Engineering Applications of Artificial Intelligence* 2017; 65:99-110.

17. Xu H, Zhang L, Li Q. **A novel inverse procedure for load identification based on improved artificial tree algorithm**. *Engineering with Computers* 2019.

18. Li QQ, He ZC, Li E. **The feedback artificial tree (FAT) algorithm**. *Soft Computing* 2020.
19. Li QQ, He ZC, Li E, Cheng AG. **Improved impact responses of a honeycomb sandwich panel structure with internal resonators**. *Engineering Optimization* 2019:1-22.
20. Li QQ, He ZC, Li E. **Dissipative multi-resonator acoustic metamaterials for impact force mitigation and collision energy absorption**. *Acta Mechanica* 2019; 230(8):2905-2935.
21. Li QQ, He ZC, Li E, Cheng AG. **Design and optimization of three-resonator locally resonant metamaterial for impact force mitigation**. *Smart Materials and Structures* 2018; 27(9):095015.
22. Bacigalupo A, Gnecco G, Lepidi M, Gambarotta L. **Optimal design of low-frequency band gaps in anti-tetrachiral lattice meta-materials**. *Composites Part B* 2016; 115:S135983681632039X.
23. Zhang Q, Li H. **MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition**. *IEEE Transactions on Evolutionary Computation* 2007; 11(6):712-731.
24. Coello CAC, Lechuga MS. **MOPSO: a proposal for multiple objective particle swarm optimization**. In: *wcci*; 2002. pp. 1051-1056.
25. Kukkonen S, Lampinen J. **GDE3: The third evolution step of generalized differential evolution**. In: *Evolutionary Computation, 2005 The 2005 IEEE Congress on*; 2005. pp. 443-450 Vol.441.
26. Deb K, Mohan M, Mishra S. **Evaluating the ϵ-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions**. *Evolutionary computation* 2005; 13(4):501-525.
27. Zitzler E, Künzli S. **Indicator-Based Selection in Multiobjective Search**. In: *Parallel Problem Solving from Nature - PPSN VIII*. Edited by Yao X, Burke EK, Lozano JA, et al. Berlin, Heidelberg: Springer Berlin Heidelberg; 2004. pp. 832-842.
28. Dai C, Wang Y, Ye M. **A new multi-objective particle swarm optimization algorithm based on decomposition**. *Information Sciences* 2015; 325:541-557.
29. Coello CAC, Pulido GT, Lechuga MS. **Handling multiple objectives with particle swarm optimization**. *IEEE Transactions on Evolutionary Computation* 2004; 8(3):256-279.
30. Tammareddi S, Sun G, Li Q. **Multiobjective robust optimization of coronary stents**. *Materials & Design* 2016; 90:682-692.
31. Sun G, Li G, Zhou S, Li H, Hou S, Li Q. **Crashworthiness design of vehicle by using multiobjective robust optimization**. *Structural and Multidisciplinary Optimization* 2011; 44(1):99-110.