Advanced Steel Construction – Vol. 22 No. 1 (2026) 64–77
DOI:10.18057/IJASC.2026.22.1.6

APPLICATION OF IMPROVED DUAL-POPULATION OPTIMIZATION FRAMEWORK IN DESIGN OF GRID STRUCTURE

Xu-Chen Xu [2, *], Hong-Bo Liu [1, 3], Zhi-Hua Chen [1, 2] and Ting Zhou [4]

[1] *State Key Laboratory of Hydraulic Engineering Simulation and Safety, Tianjin University, Tianjin 300072, China*
[2] *Department of Civil Engineering, Tianjin University, Tianjin 300072, China*
[3] *Department of Civil Engineering, Hebei University of Engineering, Handan 056000, China*
[4] *School of Architecture, Tianjin University, Tianjin 300072, China*
** (Corresponding author: E-mail: xcxu_2015@163.com)*

ABSTRACT

With the advantages of simple form, beautiful appearance, uniform force and strong span ability, grid structure has been widely used in long-span space structure. The traditional design often relies on the experience of engineers with low efficiency and using heuristic algorithms to drive design is a very good way. Simple genetic algorithm is a typical heuristic algorithm, the concept is clear, but easy to "precocious" or even not convergence. Based on the idea of "dual-population evolution", this paper introduces a series of strategies, and constructs an improved dual-population genetic algorithm (IDPGA). Two subpopulations evolve independently and exchange some individuals to prevent falling into local optima and expand searching capabilities. Then, combined with ABAQUS script, two acceleration strategies are adopted to form an intelligent optimization framework for solving the grid structure design problems, including static and dynamic optimization problems. The results show that the algorithm is effective, reliable, robust and accurate. In addition, in practical application, a satisfactory engineering solution can be found without fully exerting the optimization ability of the algorithm.

ARTICLE HISTORY

Received: 21 February 2025
Revised: 3 June 2025
Accepted: 4 June 2025

KEYWORDS

Grid structure;
Dual-population;
Genetic algorithm;
Optimization;
Design

1. Introduction

Grid structure generally including trusses, reticulated shells and other structural forms has been widely used in railway stations, exhibition centers, auditoriums, stadiums, terminals and other large public facilities. The traditional structural design often relies on the experience of engineers, who initially draw up the structural design parameters for trial calculation, and adjust them repeatedly according to the results until a satisfactory design scheme is obtained. The whole design process is tedious, the workload is large, the labor cost is high, and the obtained design scheme is not necessarily the optimal scheme in theory. Therefore, how to reduce labor costs and effectively improve design efficiency is of great significance. The working principle of heuristic algorithms is that people simulate biological phenomena in nature to solve practical problems. The requirements of the algorithms for problems are relatively "mild", and many cases prove that the algorithms are effective and reliable. Using heuristic algorithms to design grid structure can undoubtedly significantly improve efficiency.

With the improvement of computer's computing power, heuristic algorithms have gradually been applied to various fields. Doğan et al.[1] used hunting search algorithm to carry out the effect of beam-to-column connections on the minimum weight design of steel plane frames. Talaslioglu [2, 3] utilized pareto archived genetic algorithm to optimize the design of tubular lattice girders in a way of minimizing its entire weight and joint displacement and maximizing its load-carrying capacity and utilized a multi-objective design optimization approach – ImpNSGAII to concern with the design optimization of geometrically nonlinear lattice girders. Kunz et al.[4] implemented the TLBO method for weight-based optimization of space trusses.

Genetic algorithm is a typical heuristic algorithm, whose structure is clear and easy to understand, but it is easy to "precocious" and the solution is not accurate or even convergent. In order to improve the search performance, scholars have made many attempts in the past to solve their own problems. At the level of algorithm, Srinivasa et al.[5] proposed an adaptive migration model, Li et al.[6] proposed a multi-population agent genetic algorithm (MPAGAFS), Park et al.[7] proposed a dual-population genetic algorithm, Umbarkar et al.[8] proposed a multi-threaded parallel dual-population genetic algorithm, and Pourvaziri et al.[9] proposed a mixed multi-population genetic algorithm. These algorithms have achieved good results in solving individual mathematical optimization problems, but a considerable part of the algorithms are limited to solving individual unconstrained optimization problems, while the vast majority of practical engineering problems are multi-constrained optimization problems. Obviously, it is difficult to apply these algorithms to the grid structure in dealing with constrained optimization problems.

Regarding the optimization of grid structure, Mu et al. [10, 11] directly used genetic algorithm to solve an optimal design problem of a 3-bar planar truss. Then the "niche" technology was introduced into the genetic algorithm, the fuzzy control idea was used to dynamically adjust the recombination and mutation probability, and the finite element software ANSYS was used to design the roof of a parrot pavilion with three different initial decision variables. Li et al.[12] considered symmetry and directly used genetic algorithm to carry out topology optimization design for a 72-bar grid, by judging whether the cross-sectional area of the bar is a small number to determine whether the bar is available. Some solutions have been obtained basically of own. However, they are all individual optimization cases, and there is a lack of systematic research on the optimization design of grid structures. The application of genetic algorithms in grid structures is even less. From careful observation of these few cases, it can be discovered that in some problems, the algorithm still oscillates in the late stage of evolution, does not fully converge, and even develops in a worse direction. There is still a large optimization space and the algorithm is not complete.

For the detailed design stage of the grid structure, this paper improves the genetic algorithm to overcome its shortcomings, uses it to solve the problem of determining the cross-section of the members, and conducts a systematic study on its performance.

2. Optimization framework

2.1. Construction of IDPGA

Dual population evolution means that two populations evolve separately and interfere with each other, which not only ensures population diversity but also prevents population from falling into local optimal. Based on this idea, a series of strategies are introduced to improve the simple genetic algorithm to form an improved dual-population genetic algorithm (IDPGA), in which one subpopulation is responsible for detecting, the other for developing. The detailed structure of the algorithm has been explained in reference[13]. Here are some main strategies.

Elite remaining strategy: For detecting subpopulation and developing subpopulation, the best individuals of the parent are taken out, the remaining individuals evolve into offspring, the worst individuals in the offspring are removed, and the best individuals in the parent are added to form a new generation of population. It can ensure that the best individuals in the evolution process will not be lost due to recombination and mutation, and improve the accuracy of the algorithm.

Dynamic migration operator: After each generation, a certain number of individuals are exchanged between the detecting subpopulation and developing subpopulation. Considering that the number is too small to interfere with, and

too large number may completely break the evolution rhythm, resulting in difficult convergence. When the algorithm was used to solve complex mathematical problems in reference[13], considering that the population size of most problems is 100, the size of the dynamic migration operator is set to a random integer between 5 and 1/10 of the population size. This parameter value is adopted in this paper.

Separation of objective and constraints: The selection operator adopts the tournament selection, thus avoids the combination of constraints and the objective function and the introduction of penalty function. When evaluating individuals, there is no longer to calculate the fitness, but to directly determine the superiority or inferiority based on the objective value and constraint violation of the individual.

Normalization of constraints: The influence of constraints on individuals is considered as evenly as possible to eliminate the difference of magnitudes among different constraints and avoid premature evolution of individuals in one direction.

The complete process of IDPGA is shown in Fig. 1.

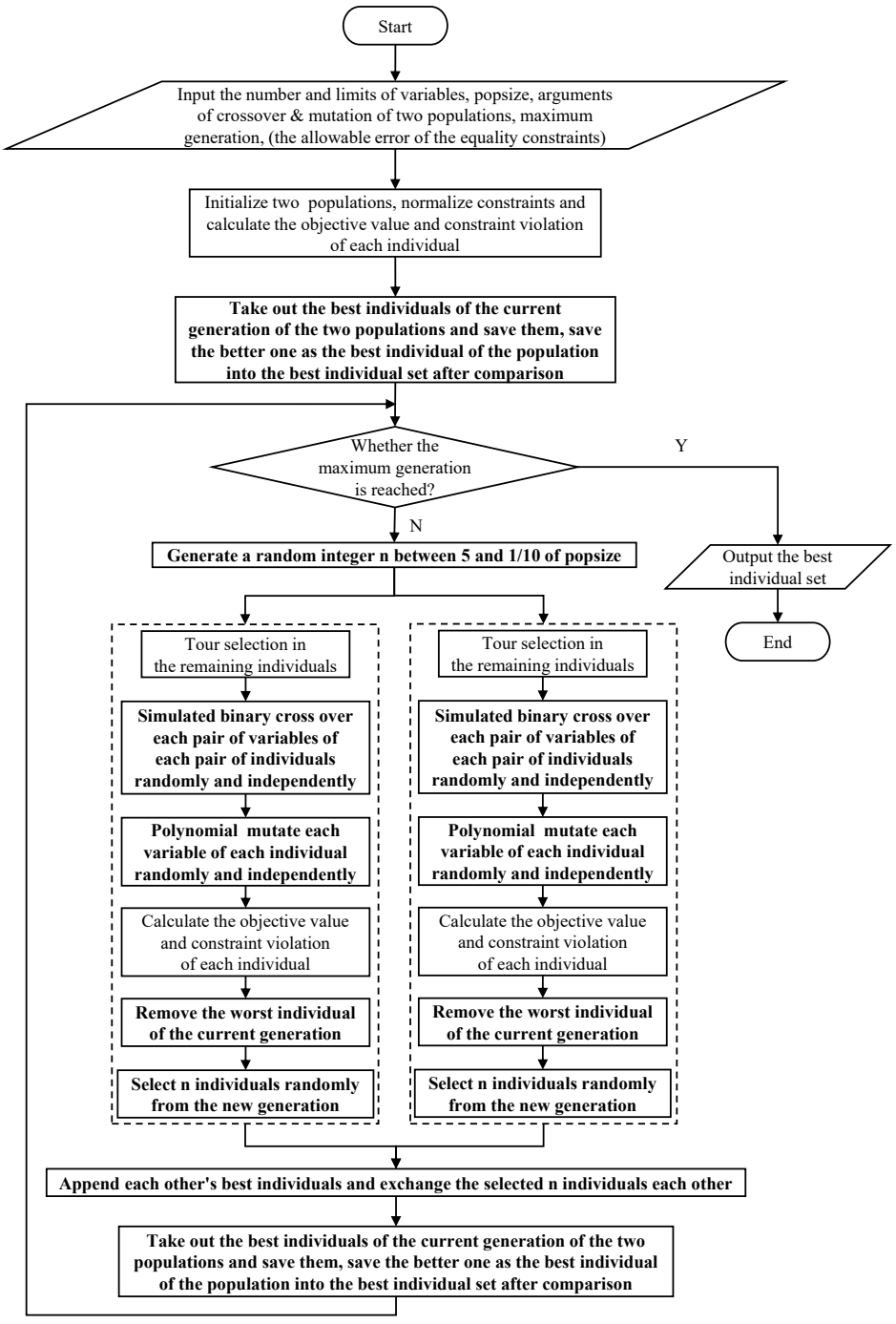


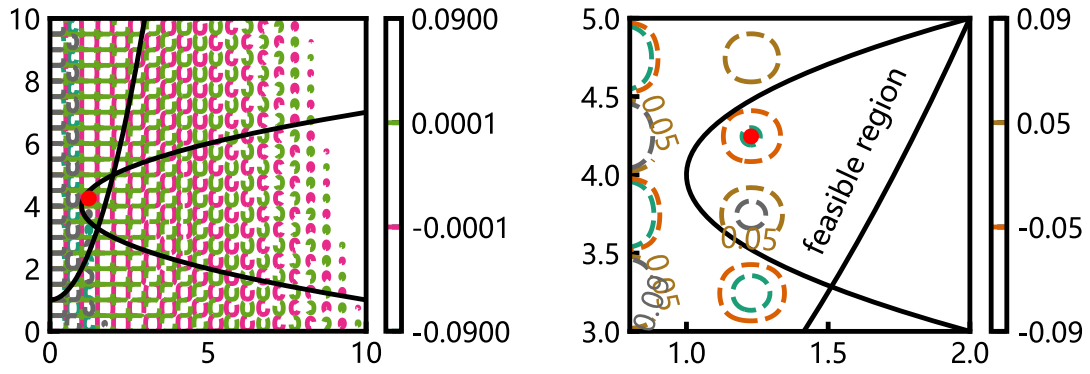
Fig. 1 The flowchart of IDPGA

2.2. Benchmarks and results

In reference[13], 14 benchmarks has been selected to test the performance of IDPGA. It is found that the algorithm is reliable with good robustness and high accuracy. Here is a brief explanation of the results of f6. The mathematical description and optimum are shown as Eq. (1).

The contour lines of objective and feasible region of decision variables are plotted in Fig. 2. The optimum is marked with a red point. The feasible region is small and the contour lines are dispersive, which means it is difficult to find the optimum.

$$\begin{aligned} &\min f(\mathbf{x}) = -\frac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1+x_2)} \\ &s.t.\begin{cases} g_1(\mathbf{x}) = x_1^2 - x_2 + 1 \le 0 \\ g_2(\mathbf{x}) = -x_1 + (x_2-4)^2 + 1 \le 0 \\ 0 \le x_i \le 10 \quad (i=1,2) \end{cases} \\ &f_{opt}(1.2280, 4.2454) = -0.0958 \end{aligned} \tag{1}$$



(a) f6 (b) f6_local

Fig. 2 The contour lines and feasible region

IDPGA was run 30 times to solve f6 and the results are shown in Table 1. The constraint violation here is the sum of constraints after normalization. Then the decision variables corresponding to the best and worst results of 30 runs are presented as shown in Table 2.

In general, the final solution found is a feasible solution with constraint violation of 0.000000, which illustrates IDPGA is effective. Meanwhile, the best solution obtained by IDPGA is equal to the theoretical optimum, and the decision variables obtained are very close to the theoretical point, which illustrates IDPGA is accurate. The standard deviation of objective is 0.0000, which illustrates IDPGA performs relatively good robustness. So IDPGA is reliable.

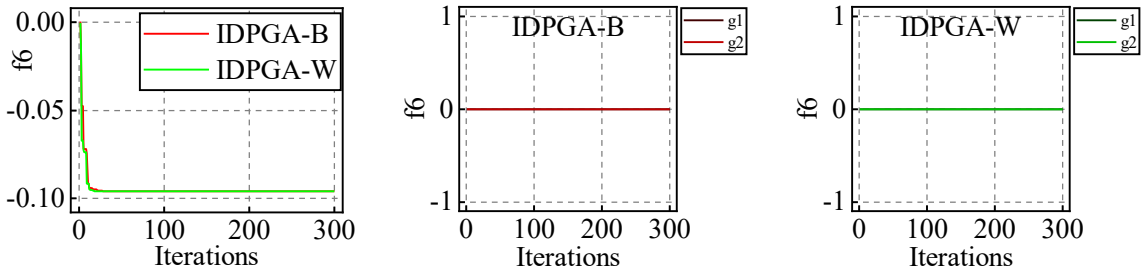
The whole evolution processes of the best and worst results of 30 runs are plotted in Fig. 3. Thereinto, IDPGA-B means the best result and IDPGA-W means the worst result.

Table 1
The objective and constraint violation of 30 runs

Objective				Constraint violation		
Best	Worst	Mean	SD	Min.	Max.	Mean
-0.0958	-0.0958	-0.0958	0.0000	0.000000	0.000000	0.000000

Table 2
Decision variables corresponding to the best and worst results

Theoretical	Best	Worst
(1.2280, 4.2454)	(1.2280, 4.2454)	(1.2280, 4.2454)



(a) objective (b) constraint violation

Fig. 3 The evolution figures

Constraint violation approaches zero at the beginning of evolution, similarly, the objective decreases very quickly at the beginning, which means IDPGA has searched for a feasible solution very quickly. Moreover, it is obviously discovered that IDPGA has converged to an acceptable solution before the stop criterion.

The errors of the simulated value relative to the theoretical value are calculated and shown in Fig. 4. The red columns mean feasible solutions and the green columns mean infeasible solutions.

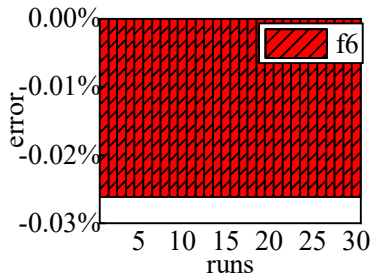


Fig. 4 The errors

Combined with the conclusions above, it can be seen IDPGA is able to obtain acceptable solutions in the face of complex mathematical problems, even better solutions in some cases.

For the benchmarks mentioned in reference[13], this paper supplemented the calculation of the Euclidean distance between the solutions obtained of 30 times and the theoretical optimal point, and plotted these distances as Nightingale rose diagrams, as shown in Fig. 5. Each sector in each diagram represents the decision variable of the solution once, and the radius of the sector represents the distance between the solution and the theoretical optimal point. Nightingale Rose diagram is a bar chart in polar coordinate system, all the data are distributed in a sector, and the sector radius represents the size of the data. Since the area of the sector is squared with the radius, the diagram scales the data proportionally and can intuitively show the difference among similar data.

From Fig. 5, the sector distribution in each diagram is relatively uniform, and the radius of most sectors is close to each other, which indicates that the distance between the solution and the theoretical optimum is close, and IDPGA is relatively stable. Except for problems f3 and f12, the distance between the results of other problems and the theoretical optimum is small, which shows that IDPGA solves most problems more accurately. By observing the decision variables of problems f3 and f12, it is found that there are differences of orders of magnitude between each component. Meanwhile, the algorithm adopts constraint normalization processing to avoid the influence of such differences as much as possible, and considers the influence of each component evenly as much as possible in the search process. Such results show that this constraint normalization processing is not universal.

Considering that the actual engineering optimization problems are simpler than these mathematical optimization problems, IDPGA is still reliable.

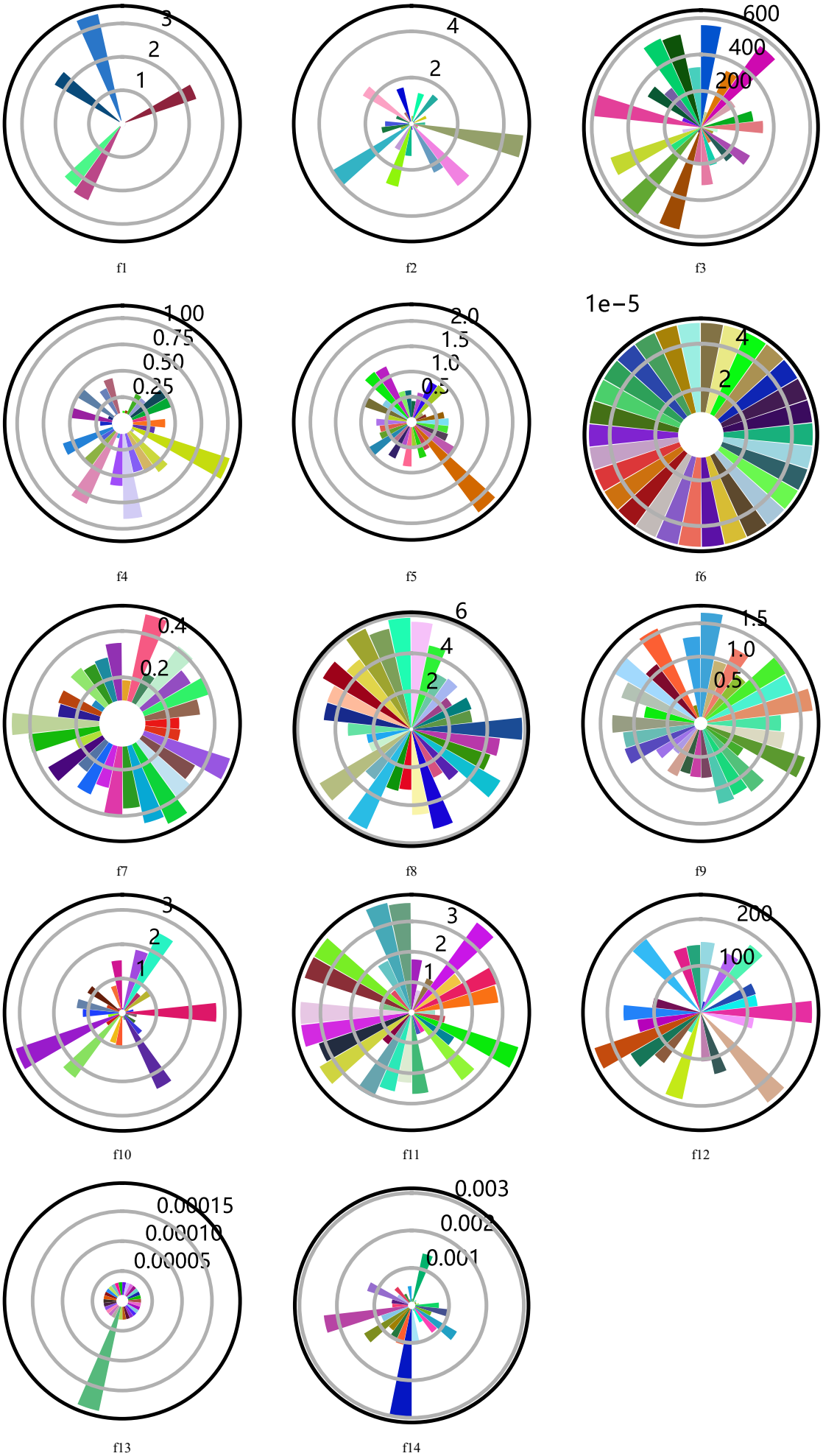


Fig. 5 The distance between the solution and the theoretical optimum

2.3. Interaction between IDPGA and ABAQUS

There is no universal algorithm, and the application of any algorithm has certain restrictions. In other words, if an algorithm is suitable for solving a certain problem or a certain class of problems, there must be other problems that are "powerless", which is the famous "No Free Lunch" theory[14]. Therefore, this paper will not look for the so-called "universal" algorithm, nor will it look for complex algorithms, but as far as possible to apply IDPGA to grid structure optimization problems. Considering that the algorithm in this paper is completed in the current popular Python language, while the secondary development script of the general finite element analysis software ABAQUS supports Python language, and considering that there are few scripts for the co-simulation optimization of Python and ABAQUS in the existing literature, this paper uses ABAQUS as a solver for simulation. So only one programming language, Python is required to achieve all optimization functions, which facilitates the subsequent development of specialized ABAQUS plugins.

In fact, corresponding to the various functional modules in the GUI of ABAQUS, there are special program blocks in the model script. The GUI model corresponds to the object of the Python language, each function module of the GUI corresponds to each class of Python, and the parameters under a function module of the GUI correspond to various functions under a class of Python. In this paper, the interactive optimization script for the mechanical performance of the grid structure is compiled by referring to 'Python 3.10.9 Help Documentation'[15], 'Scripting User's Guide', 'Scripting Reference Guide' and 'Getting Started With Abaqus: Interactive Edition' in the ABAQUS help document[16].

In the optimization framework, IDPGA generates the initial model parameters and creates the initial analysis script, ABAQUS runs the parameterized script for analysis and writes the result data to the txt file, IDPGA reads the data from the txt file, updates the model and continues to drive the iteration, repeating these processes until the iteration is completed, as shown in Fig. 6.

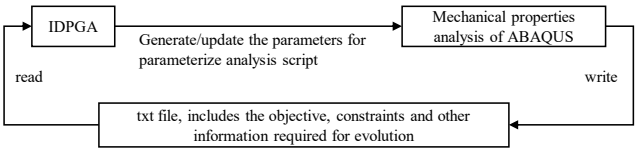


Fig. 6 The diagram of interaction between IDPGA and ABAQUS

When the IDPGA solves mathematical problems, it first initializes the population, then calculates the corresponding objective and constraint violation and other information of different individuals. Since the objective and constraint function are both explicit, the solving speed is very fast, so the information of individuals is solved one by one. When solving the grid structure optimization problem, ABAQUS needs to be repeatedly called for multiple analyses. ABAQUS takes a certain amount of time to create the model and perform the analysis. For each generation, the optimization framework needs to calculate the information such as the objective and constraint violation of each individual. In other words, ABAQUS is repeatedly called for analysis and writing many times (population size). If the model is still analyzed one by one, obviously, the time cost will be quite large. The optimization framework adopts two acceleration strategies for analysis, which gives full play to the computer hardware, saves solving time and improves solving efficiency.

Specific operations of grouping and parallel computing strategy: Firstly, the maximum number of models that can be simultaneously solved by ABAQUS is set according to the computer hardware. Then, all models that need to be submitted for analysis are grouped, that is, the population size is divided by this number, and the number of groups is obtained by adding one after the round down. Finally, according to these groups, ABAQUS is called at the same time to analyze all models in a certain group. Until the models of all groups have been analyzed.

When running the optimization framework in practice, the Python subprocess module 'subprocess' is used to call ABAQUS through the shell layer. In addition, multiple processes are created through the class 'Popen' to run multiple ABAQUS models "simultaneously", and the function 'wait' is set to block to realize the writing after completing. It is worth noting that the creation of multiple processes through the 'Popen' is sequential, so multiple ABAQUS models are not strictly running at the same time, but the time difference caused by the creation of processes is very small, almost instantaneous, and can be considered at the same time.

Comparison strategy between parent and offspring: Careful observation of the structure of IDPGA shows that each iteration needs to calculate the objective and constraint violation of all individuals. For mathematical problems, this time is very short, but for grid structure optimization problems, the time of calling ABAQUS is very large. In IDPGA, the developing subpopulation has a small recombination probability and mutation probability, and the individuals are not easy to lose in the evolution, while the detecting subpopulation has a large recombination probability and mutation probability, and the individuals are easy to lose in the evolution. In other words, when the parent individuals of a certain generation evolve into a new generation of offspring individuals, some individuals may not participate in recombination and mutation, but are directly retained, so the algorithm does not need to calculate the objective and constraint violation of these individuals again, thereby further saving the solving time.

3. Static performance optimization problem of grid structure

3.1. Description of problems

The optimization design problem of static performance of grid structure with constraints of stress and displacement can be described as follows: the objective is the mass of grid structure, the constraints are the allowable stress of each member and the allowable displacement of each node, and the decision variable is the cross-sectional area of each member, and the problem is to find the minimum mass of grid structure. Correspondingly, the mathematical expression of the problem is shown as Eq. (2), where $f(\mathbf{x})$ is the mass of grid structure, ρ is the density of material, A_i and L_i are the cross-sectional area and length of each member, n is the number of members, $\sigma_j(\mathbf{x})$ and $[\sigma]$ are stress and allowable stress of each member, the allowable stresses of the tension member and compression member are often different in practice, $\delta_k(\mathbf{x})$ and $[\delta]$ are displacement and allowable displacement of each node, the allowable displacements of nodes in each direction are often different in practice, $A_{\max}$ and $A_{\min}$ are the upper and lower bounds of the cross-sectional area of each member respectively.

$$\begin{aligned} &\min f(\mathbf{x}) = \sum_{i=1}^{n} \rho A_i L_i \\ &s.t. \begin{cases} \sigma_j(\mathbf{x}) \le [\sigma] & (j = 1,2,...,p) \\ \delta_k(\mathbf{x}) \le [\delta] & (k = p+1, p+2,...,q) \\ A_{\min} \le A_i \le A_{\max} & (i = 1,2,...,n) \end{cases} \end{aligned} \tag{2}$$

The normalization of stress and displacement constraints adopts the form as Eq. (3).

$$\begin{cases} \dfrac{\sigma_j(\mathbf{x}) - [\sigma]}{[\sigma]} \le 0, & j = 1,2,...,p \\ \dfrac{\delta_k(\mathbf{x}) - [\delta]}{[\delta]} \le 0, & k = p+1, p+2,...,q \end{cases} \tag{3}$$

3.2. Simulation settings

The optimization framework was utilized to solve design problems of 10-bar plane truss, 52-bar plane truss, 200-bar plane truss and 120-bar single-layer lattice shell successively. Considering the scale of the problem, computers with different hardware were used for different problems, as shown in Table 3. Hardware 1: Intel Core i7-7700 3.60GHz, 16 GB RAM, 8CPUs, single thread, Win10 platform. Hardware 2: AMD EPYC 7H12 2.60GHz, 32 GB RAM, 32CPUs, single thread, Win10 platform.

Table 3
Computer properties of different problems

Grid structure	Hardware	Whether to adopt the two acceleration strategies
10-bar plane truss	1	No
52-bar plane truss	1	Yes
200-bar plane truss	2	Yes
120-bar single-layer lattice shell	2	Yes

Parameters of IDPGA: The population size was taken as 100. The probabilities of recombination and mutation of developing subpopulation were set as 0.3 and 0.001, the probabilities of recombination and mutation of detecting subpopulation were set as 0.9 and 0.05. The arguments η_c and η_m of

developing subpopulation were set as 1 and 5, the arguments η_c and η_m of detecting subpopulation were set as 1 and 100. The maximum number of generations was different for different problems. The stop criterion was that the maximum number of generations had been reached.

All results were automatically written to Excel files after the optimization framework finished running.

3.3. Problems solving

3.3.1. Optimization design of 10-bar plane truss

The grid structure in this problem is a plane truss composed of 10 elements. As shown in Fig. 7, vertical and downward concentrated force are applied to nodes 2 and 4 respectively. The objective is the mass of the structure, and the 18 constraints include stress constraints of 10 elements and horizontal and vertical displacement constraints of nodes 1, 2, 3 and 4. The decision variable is the cross-sectional area of each element, and 10 in total.

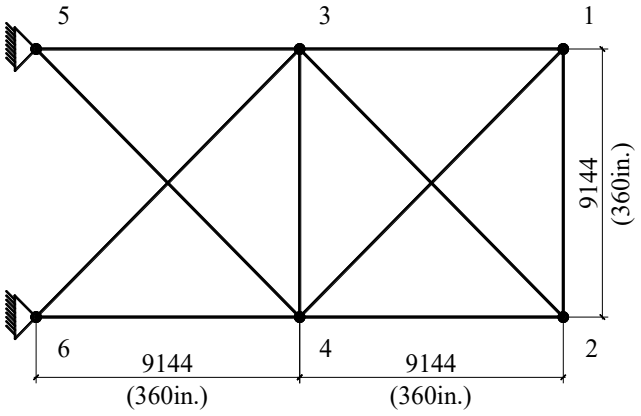


Fig. 7 Diagram of the 10-bar plane truss

The optimization framework was applied to solve this problem for 13 times consecutively, and the results were statistically analyzed, including the objective, constraint violation and time consumption, as shown in Table 4.

Table 4
Optimization results of 13 runs

Run	Objective (t)	Constraint violation	Time (h)
1	2.36287	0	18.0740
2	2.37119	0	17.8861
3	2.37320	0	17.6013
4	2.38420	0	17.5708
5	2.36838	0	17.4421
6	2.36346	0	17.7583
7	2.36565	0	17.6355
8	2.35837	0	17.6093
9	2.37954	0	17.5962
10	2.37536	0	17.6283
11	2.35842	0	17.8011
12	2.36553	0	17.7377
13	2.36240	0	17.7004
Best	2.35837	0	17.4421
Worst	2.38420	0	18.0740
Mean	2.36835	0	17.6955
SD	0.00765	-	-

From the constraint violation, the constraint violation of any solution is 0, that is, all constraints are satisfied, indicating that all solutions are feasible solutions, and IDPGA is reliable. From the objective, 13 solutions are close, the best result is 2.35837t, and the worst result is 2.38420t, both of which are close, indicating that IDPGA is robust. From the time consumption, IDPGA takes less than 18h to solve this problem on average. Then, the evolution process diagram of the objective of the 13 results was uniformly drawn, as shown in Fig. 8.

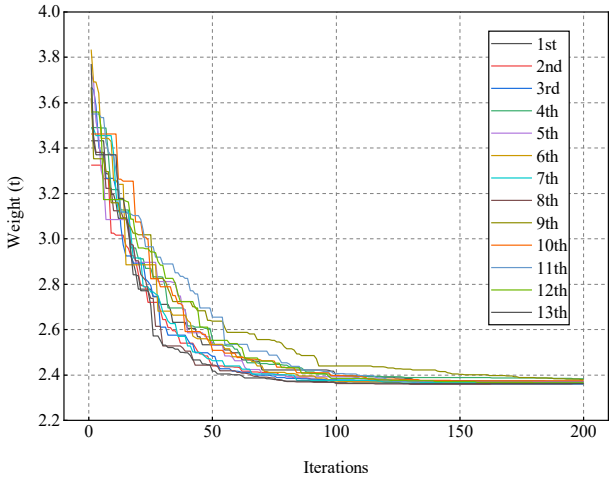


Fig. 8 The evolution process of the objective of 13 results

It can be seen that the objective decreases rapidly before 100 iterations, and then slowly decreases and gradually converges. In most cases, after about 150 iterations, the algorithm has obtained a satisfactory solution. Next, the objectives of the 30 results were plotted in a bar chart, as shown in Fig. 9.

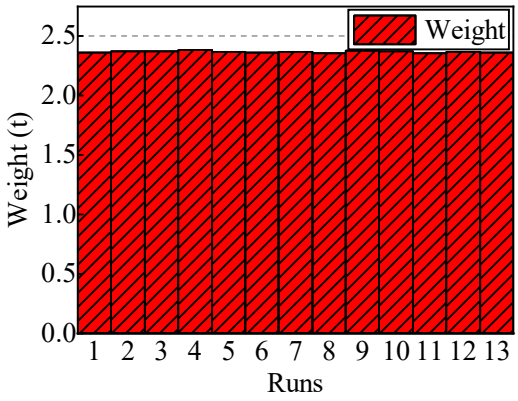


Fig. 9 The bar chart of the objectives of 13 runs

It can be intuitively seen that the height of each bar is very close to others, which shows that the algorithm has good robustness again. Finally, some of the best results reported by other existing algorithms were selected and compared with the results of this algorithm. All the results were listed in Table 5.

Table 5
The best results reported by other algorithms

Algorithm	PSO[17]	PSOPC[17]	HPSO[17]
Objective (t)	2.53184	2.53714	2.50926
Algorithm	GA[18]	FEAPGEN[19]	GSS[20]
Objective (t)	2.48566	2.52057	2.51003

It can be seen that the best value of 2.35837t obtained by IDPGA is better than the results reported above, indicating that the algorithm has good accuracy. Accordingly, the optimized model is shown in Fig. 10, where the maximum stress of the tension element is 144.21MPa, the maximum stress of the compression element is -58.13MPa, the maximum horizontal displacement of the node is -14.57mm, and the maximum vertical displacement is -50.80mm.

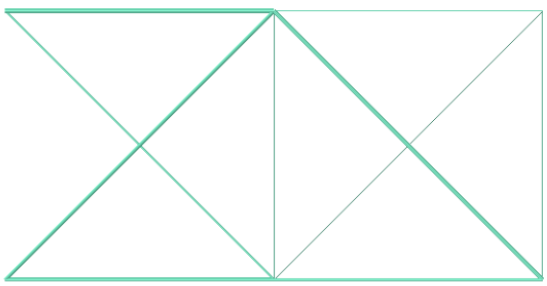


Fig. 10 The diagram of optimized model

3.3.2. Optimization design of 52-bar plane truss

The grid structure in this problem is a plane truss composed of 52 elements. As shown in Fig. 11, the elements are divided into 12 groups according to the cross-sectional area, the group number is identified by the number without brackets, and the horizontal right and the vertical upward concentrated force are applied to the nodes (1), (2), (3) and (4), respectively. The objective is the mass of the structure, because of asymmetrical force, and the 52 constraints include stress constraints of 52 elements and no displacement constraints of nodes. The decision variable is the cross-sectional area of each element, and 12 in total.

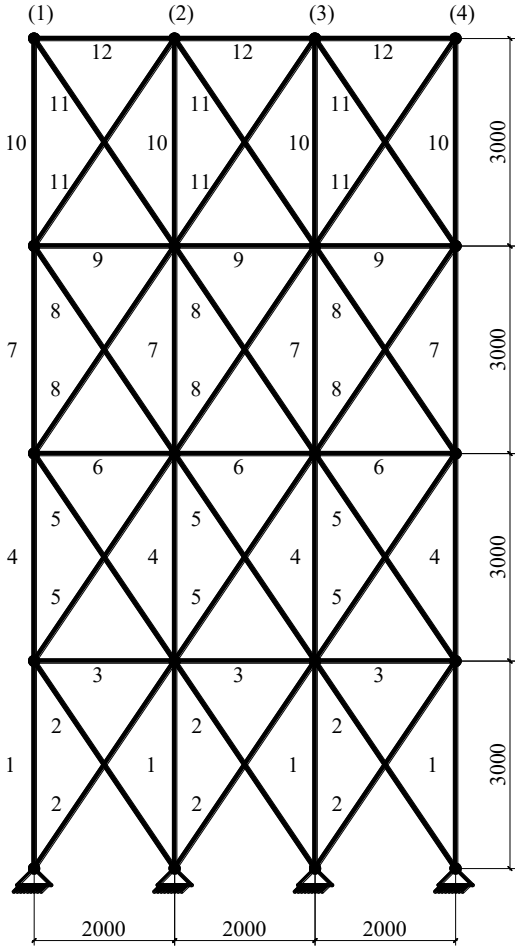


Fig. 11 Diagram of the 52-bar plane truss

From the results of 10-bar plane truss, it can be seen that IDPGA is very robust. Therefore, the optimization framework was only run once to solve the next problems. Similarly, the results were statistically analyzed as shown in Table 6.

Table 6
Optimization results

Objective (t)	Constraint violation	Time (h)
1.89667	0	46.9925

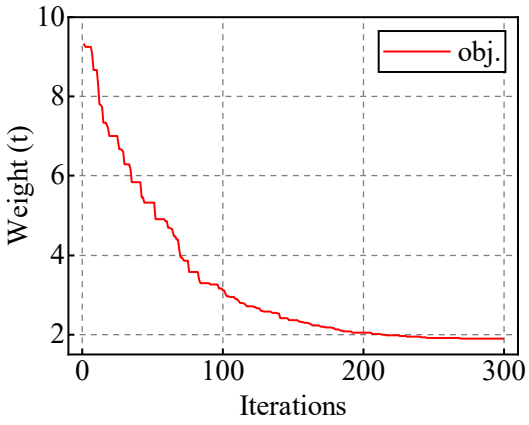


Fig. 12 The evolution process of the objective

Similarly, the constraint violation of the solution is 0, indicating that the solution is a feasible solution. IDPGA takes about 47h to solve this problem. The evolution process diagram of the objective was drawn, as shown in Fig. 12.

It can be seen that the objective decreases rapidly before 200 iterations, and then slowly decreases and gradually converges. After about 250 iterations, the algorithm has obtained a satisfactory solution. Finally, some of the best results reported by other existing algorithms were selected and compared with the results of this algorithm. All the results were listed in Table 7.

Table 7
The best results reported by other algorithms

Algorithm	WCA[21]	IMBA[21]	DE[22]
Objective (t)	1.90261	1.90261	1.90261
Algorithm	AEDE[22]	FWA[23]	IFWA[24]
Objective (t)	1.90261	1.90261	1.90261

It can be seen that the best value of 1.89667t obtained by IDPGA is better than the results reported above, indicating that the algorithm has good accuracy. Accordingly, the optimized model is shown in Fig. 13, where the maximum stress of the tension element is 179.99MPa, the maximum stress of the compression element is -144.00MPa.

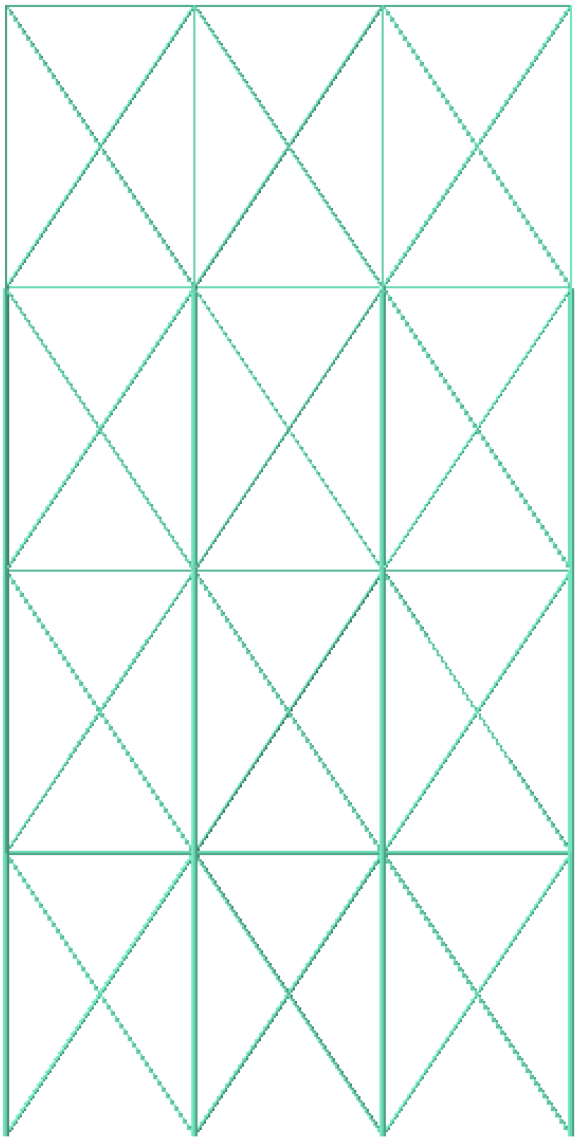


Fig. 13 The diagram of optimized model

3.3.3. Optimization design of 200-bar plane truss

The grid structure in this problem is a plane truss composed of 200 elements. As shown in Fig. 14, the elements are divided into 29 groups according to the cross-sectional area, the group number is only identified by the number without brackets in the semi-structure because of symmetry, the horizontal right concentrated force is applied to the nodes (1), (6), (11), (16), (21), (26), (31),

(36), (41), (46) and (51), and the vertical downward concentrated force is applied to the nodes (1) to (55). The objective is the mass of the structure, and the 200 constraints include stress constraints of 200 elements and no displacement constraints of nodes. The decision variable is the cross-sectional area of each element, and 29 in total.

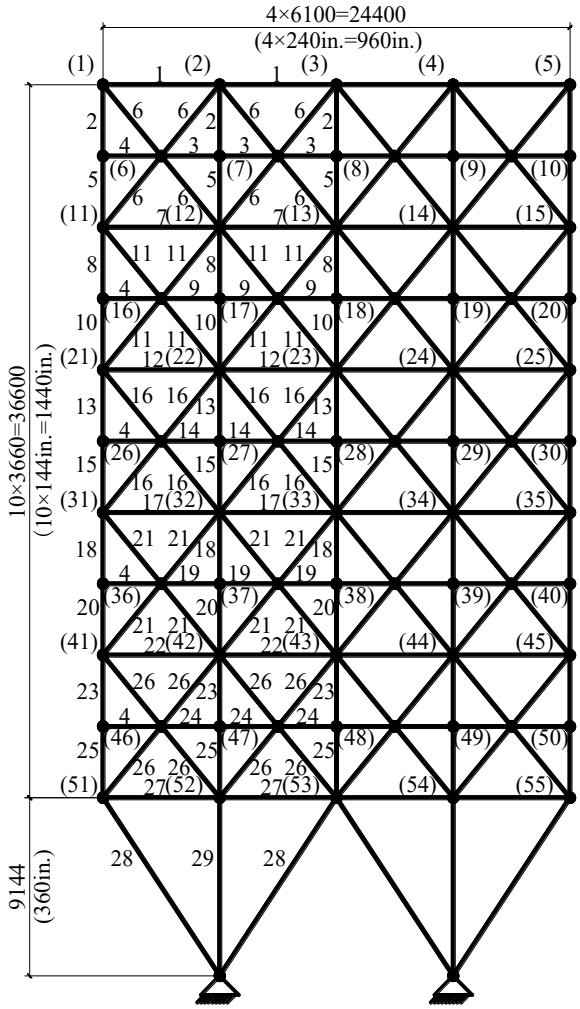


Fig. 14 Diagram of the 200-bar plane truss

The optimization framework was run once to solve the next problems. Similarly, the results were statistically analyzed as shown in Table 8.

Table 8
Optimization results

Objective (t)	Constraint violation	Time (h)
12.32858	0	28.3835

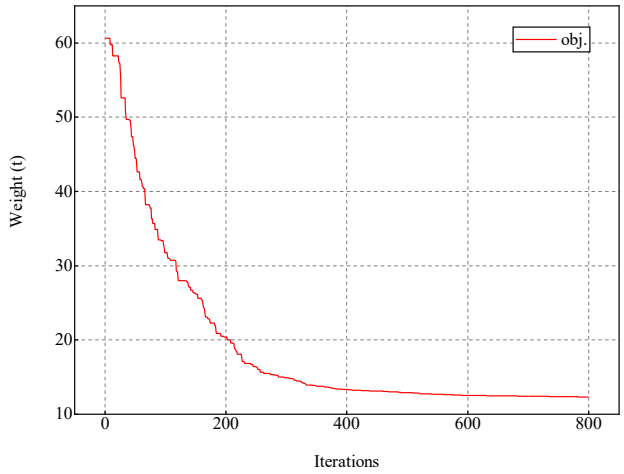


Fig. 15 The evolution process of the objective

Similarly, the constraint violation of the solution is 0, indicating that the solution is a feasible solution. IDPGA takes about 28h to solve this problem. The evolution process diagram of the objective was drawn, as shown in Fig. 15.

It can be seen that the objective decreases rapidly before 400 iterations, and then slowly decreases and gradually converges. After about 600 iterations, the algorithm has obtained a satisfactory solution. Finally, some of the best results reported by other existing algorithms were selected and compared with the results of this algorithm. All the results were listed in Table 9.

Table 9
The best results reported by other algorithms

Algorithm	ESASS[25]	DE[22]	AEDE[22]
Objective (t)	12.73483	12.65594	12.63640
Algorithm	FWA[23]	IFWA[24]	
Objective (t)	12.59987	12.45077	

It can be seen that the best value of 12.32858t obtained by IDPGA is better than the results reported above, indicating that the algorithm has good accuracy. Accordingly, the optimized model is shown in Fig. 16, where the maximum stress of the tension element is 68.88MPa, the maximum stress of the compression element is -68.90MPa.

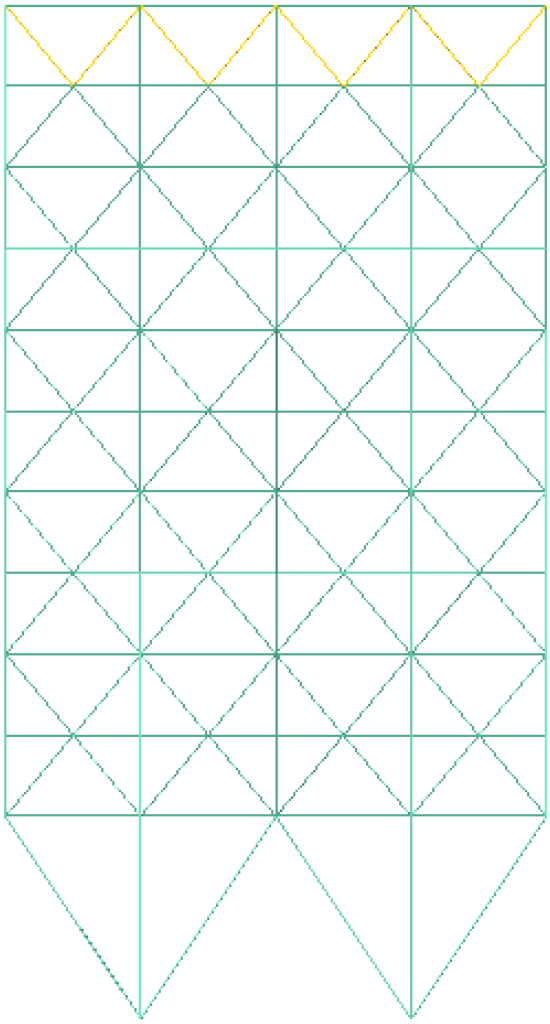


Fig. 16 The diagram of optimized model

3.3.4. Optimization design of 120-bar single-layer lattice shell

The grid structure in this problem is a single-layer lattice shell composed of 120 elements. As shown in Fig. 17, the elements are divided into 7 groups according to the cross-sectional area, the group number is only identified by the number without brackets in a single block because of symmetry, and the vertical downward concentrated force is applied to the nodes (1), (2), (3), and other symmetric nodes respectively.

The material is steel, and the allowable stresses of the tension and

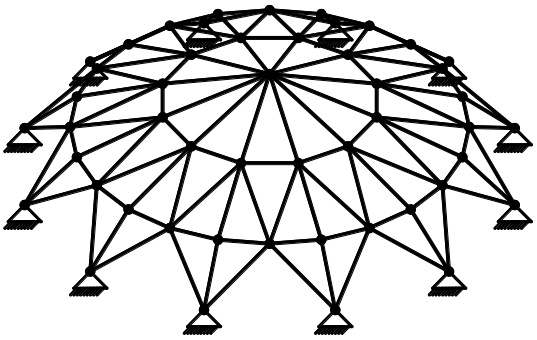
compression elements are different. According to the definition of the American code for the design of steel structures, AIC-ASD[26], the allowable stress of the tension element adopts the form as Eq. (4), where $[\sigma]_{\text{tension}}$ is the allowable stress, f_y is the yield strength of steel.

$$[\sigma]_{\text{tension}} = 0.6 f_y \tag{4}$$

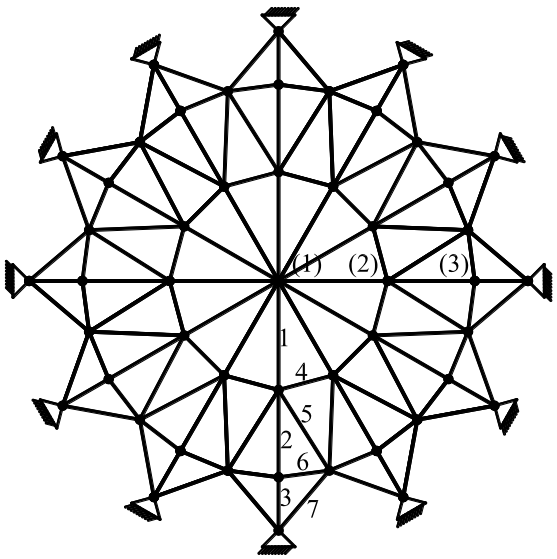
The allowable stress of the compression element is calculated according to two failure modes of elastic buckling and plastic buckling respectively as Eq. (5). E is the elastic modulus of steel, C_c is the critical slenderness ratio parameter, $C_c = \sqrt{\frac{2\pi^2 E}{f_y}}$, λ_i is the slenderness ratio of the ith element, $\lambda_i = \frac{k_i L_i}{r_i}$, k_i is the calculated length coefficient of the ith element, and is 1 for all the elements in this problem, r_i is the section rotation radius of the ith element, and the relationship between it and the cross-sectional area can be defined as $r_i = aA_i^b$ [27], a and b are two constants related to the type of section, and are 0.4993 and 0.6777 for pipes respectively.

$$[\sigma]_{\text{compression}i} = \begin{cases} \dfrac{\left(1 - \dfrac{\lambda_i^2}{2C_c^2}\right) f_y}{\dfrac{5}{3} + \dfrac{3\lambda_i^2}{8C_c} - \dfrac{\lambda_i^3}{8C_c^3}} & \lambda_i < C_c \ (\textit{plastic buckling}) \\ \dfrac{12\pi^2 E}{23\lambda_i^2} & \lambda_i \geq C_c \ (\textit{elastic buckling}) \end{cases} \tag{5}$$

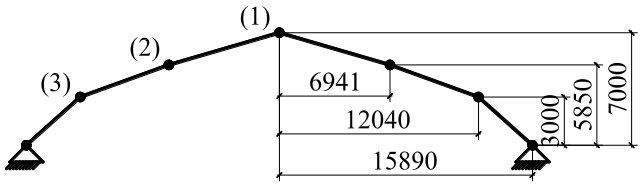
The objective is the mass of the structure. The constraints include the stress constraints of 120 elements and the displacement constraints of all free nodes in three directions. Due to the symmetry, only the constraints of a single block are taken, and there are 19 constraints in total. The decision variable is the cross-sectional area of each element, and 7 in total.



(a) 3D view



(b) Top view



(c) Side view

Fig. 17 Diagram of the 120-bar single-layer lattice shell

The optimization framework was run once to solve the next problems. Similarly, the results were statistically analyzed as shown in Table 10.

Table 10
Optimization results

Objective (t)	Constraint violation	Time (h)
14.44850	0	32.9462

Similarly, the constraint violation of the solution is 0, indicating that the solution is a feasible solution. IDPGA takes about 33h to solve this problem. The evolution process diagram of the objective was drawn, as shown in Fig. 18.

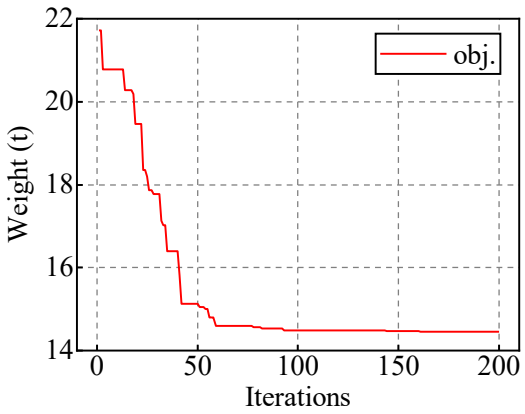


Fig. 18 The evolution process of the objective

It can be seen that the objective decreases rapidly before 60 iterations, and then slowly decreases and gradually converges. After about 100 iterations, the algorithm has obtained a satisfactory solution. Finally, some of the best results reported by other existing algorithms were selected and compared with the results of this algorithm. All the results were listed in Table 11.

Table 11
The best results reported by other algorithms

Algorithm	PSO[28]	MSPSO[28]	HPSSO[29]
Objective (t)	15.08283	15.08250	15.08197
Algorithm	CA[30]	EHO[31]	EHOC[32]
Objective (t)	15.08441	15.34355	15.34355

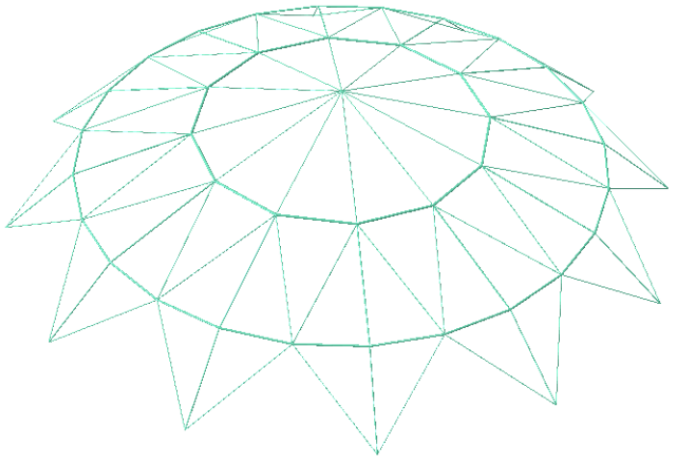


Fig. 19 The diagram of optimized model

It can be seen that the best value of 14.44850t obtained by IDPGA is better than the results reported above, indicating that the algorithm has good accuracy. Accordingly, the optimized model is shown in Fig. 19, where the maximum stress of the tension element is 11.95MPa, the maximum stress of the compression element is -20.10MPa, the maximum displacements of the node are 0.93mm in x direction and y direction, and -5.00mm in z direction.

4. Dynamic performance optimization problem of grid structure

4.1. Description of problems

The optimization design problem of dynamic performance of grid structure with constraints of frequency can be described as follows: the objective is the mass of grid structure, the constraints are several order frequency constraints of grid structure, and the decision variable is the cross-sectional area of each member, and the problem is to find the minimum mass of grid structure. Correspondingly, the mathematical expression of the problem is shown as Eq. (6), where $\omega_j(\mathbf{x})$ and $[\omega]_j$ are the natural frequency and allowable frequency of the *j*th order.

$$\begin{aligned} &\min f(\mathbf{x}) = \sum_{i=1}^{n} \rho A_i L_i \\ &s.t. \begin{cases} \omega_j(\mathbf{x}) \geq [\omega]_j & (j = 1,2,...,q) \\ A_{\min} \leq A_i \leq A_{\max} & (i = 1,2,...,n) \end{cases} \end{aligned} \tag{6}$$

The normalization of frequency constraints adopts the form as Eq. (7).

$$\frac{-\omega_j(\mathbf{x}) + [\omega]_j}{[\omega]_j} \leq 0, \qquad j = 1,2,...,q \tag{7}$$

4.2. Simulation settings

The optimization framework was utilized to solve design problems of 10-bar plane truss, 37-bar plane truss, 200-bar plane truss and 1410-bar double-layer lattice shell successively. Similarly, computers with different hardware were used for different problems, as shown in Table 12. Hardware 1 and Hardware 2 are the same as above.

Table 12
Computer properties of different problems

Grid structure	Hardware	Whether to adopt the two acceleration strategies
10-bar plane truss	1	No
37-bar plane truss	2	Yes
200-bar plane truss	2	Yes
1410-bar double-layer lattice shell	2	Yes

Parameters of IDPGA: The maximum number of generations was different for different problems. Other parameters are the same as above.

All results were automatically written to Excel files after the optimization framework finished running.

4.3. Problems solving

4.3.1. Optimization design of 10-bar plane truss

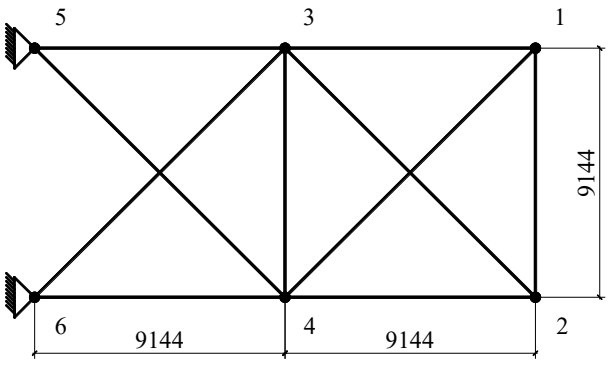


Fig. 20 Diagram of the 10-bar plane truss

The grid structure in this problem is a plane truss composed of 10 elements. As shown in Fig. 20, the concentrated mass is applied to nodes 1, 2, 3 and 4 respectively. The objective is the mass of the structure, and the constraints include natural frequencies of the first 3 orders of structure. The decision variable is the cross-sectional area of each element, and 10 in total.

The optimization framework was run once to solve the next problems. Similarly, the results were statistically analyzed as shown in Table 13.

Table 13
Optimization results

Objective (t)	Constraint violation	Time (h)
0.54183	0	22.5678

Similarly, the constraint violation of the solution is 0, indicating that the solution is a feasible solution. IDPGA takes about 23h to solve this problem. The evolution process diagram of the objective was drawn, as shown in Fig. 21.

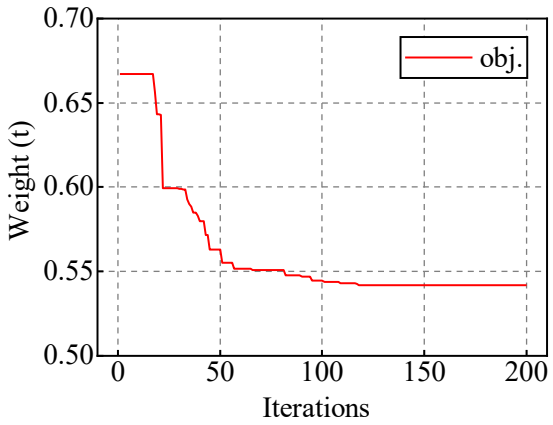


Fig. 21 The evolution process of the objective

It can be seen that the objective decreases rapidly before 50 iterations, and then slowly decreases and gradually converges. After about 100 iterations, the algorithm has obtained a satisfactory solution. Finally, some of the best results reported by other existing algorithms were selected and compared with the results of this algorithm. All the results were listed in Table 14.

Table 14
The best results reported by other algorithms

Algorithm	GA[33]	PSO[34]	CSS[35]
Objective (t)	0.54275	0.53798	0.53195
Algorithm	ECSS[35]	CSS-BBBC[36]	
Objective (t)	0.52925	0.52909	

It can be seen that the best value of 0.54183t obtained by IDPGA is better than the results reported by GA, worse than that reported by the other four algorithms. This phenomenon does not indicate that the solution obtained by IDPGA is poor, because the algorithm was randomly run once, while the solutions of other algorithms in the table are their own best solutions of many runs. In fact, the solution obtained by IDPGA is close to the best of other algorithms, and the first 3 natural frequencies of the structure obtained are 7.0014Hz, 15.691Hz and 20.004Hz respectively. Such results are acceptable in engineering, and IDPGA still has good accuracy. Accordingly, the optimized model is shown in Fig. 22.

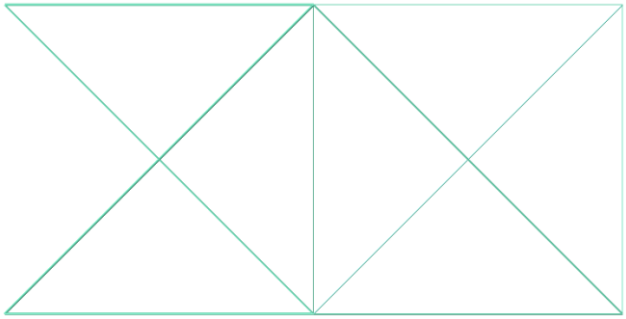


Fig. 22 The diagram of optimized model

4.3.2. Optimization design of 37-bar plane truss

The grid structure in this problem is a plane truss composed of 37 elements. As shown in Fig. 23, the positions of the nodes in the lower chords are fixed, the concentrated mass is applied to nodes in the lower chords respectively. The objective is the mass of the structure, and the constraints include natural frequencies of the first 3 orders of structure. The decision variables are the cross-sectional areas of the upper chords and the web members and the vertical ordinates of the nodes in the upper chords, considering symmetry, 19 in total.

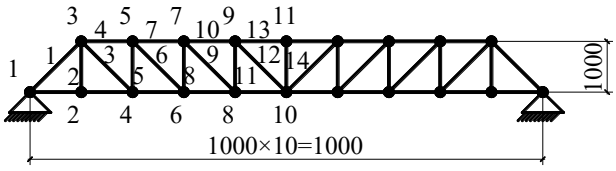


Fig. 23 Diagram of the 37-bar plane truss

The optimization framework was run once to solve the next problems. Similarly, the results were statistically analyzed as shown in Table 15.

Table 15
Optimization results

Objective (t)	Constraint violation	Time (h)
0.36777	0	25.7763

Similarly, the constraint violation of the solution is 0, indicating that the solution is a feasible solution. IDPGA takes about 26h to solve this problem. The evolution process diagram of the objective was drawn, as shown in Fig. 24.

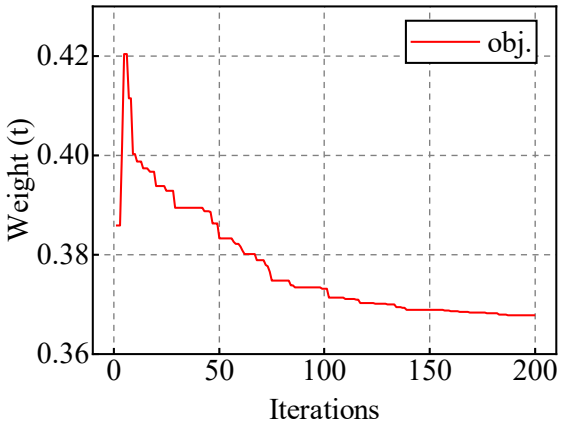


Fig. 24 The evolution process of the objective

It can be seen that the objective decreases rapidly before 100 iterations, and then slowly decreases and gradually converges. After about 180 iterations, the algorithm has obtained a satisfactory solution. Finally, some of the best results reported by other existing algorithms were selected and compared with the results of this algorithm. All the results were listed in Table 16.

Table 16
The best results reported by other algorithms

Algorithm	GA[33]	PSO[37]	NHPGA[38]
Objective (t)	0.36884	0.37720	0.36303
Algorithm	CSS[35]	ECSS[35]	HRPSO[39]
Objective (t)	0.36284	0.36238	0.36472

It can be seen that the best value of 0.36777t obtained by IDPGA is better than the results reported by GA and PSO, worse than that reported by the other four algorithms. Similarly, this phenomenon does not indicate that the solution obtained by IDPGA is poor, the solution obtained by IDPGA is close to the best of other algorithms, and the first 3 natural frequencies of the structure obtained are 20.009Hz, 40.063Hz and 60.004Hz respectively. Such results are acceptable in engineering, and IDPGA still has good accuracy. Accordingly, the optimized model is shown in Fig. 25.

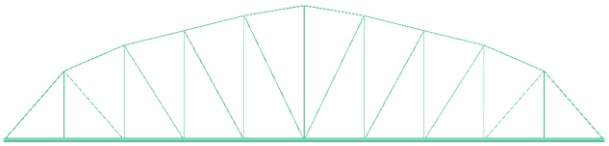


Fig. 25 The diagram of optimized model

4.3.3. Optimization design of 200-bar plane truss

The grid structure in this problem is a plane truss composed of 200 elements. As shown in Fig. 26, the concentrated mass is applied to nodes (1), (2), (3), (4) and (5) respectively. The objective is the mass of the structure, and the constraints include natural frequencies of the first 3 orders of structure. The decision variables are the cross-sectional areas of each member, considering symmetry, 29 in total.

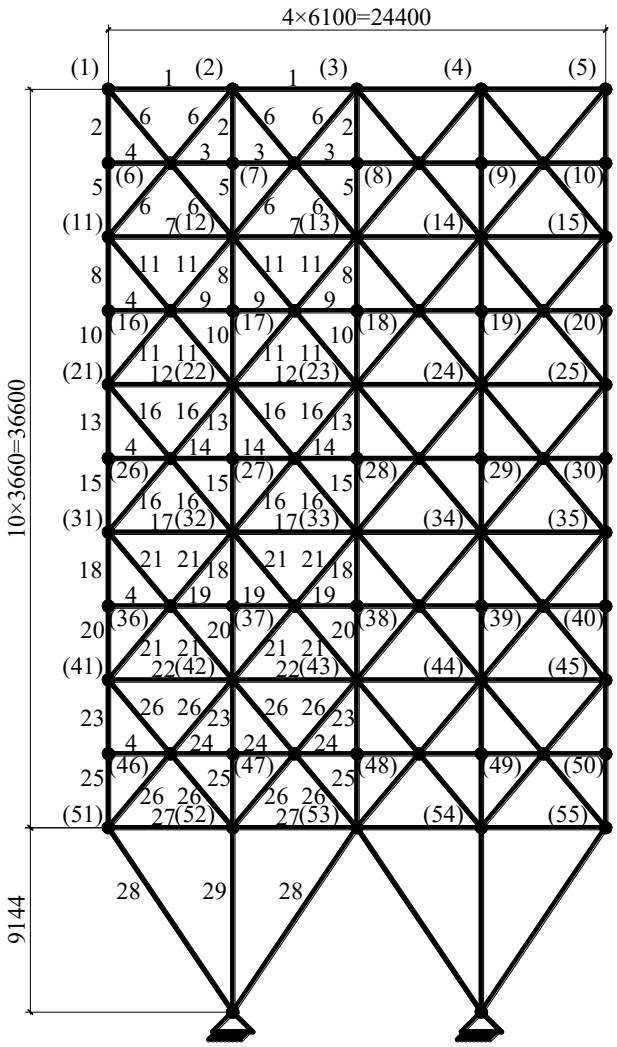


Fig. 26 Diagram of the 200-bar plane truss

The optimization framework was run once to solve the next problems. Similarly, the results were statistically analyzed as shown in Table 17.

Table 17
Optimization results

Objective (t)	Constraint violation	Time (h)
2.20373	0	29.2922

Similarly, the constraint violation of the solution is 0, indicating that the solution is a feasible solution. IDPGA takes about 29h to solve this problem. The evolution process diagram of the objective was drawn, as shown in Fig. 27.

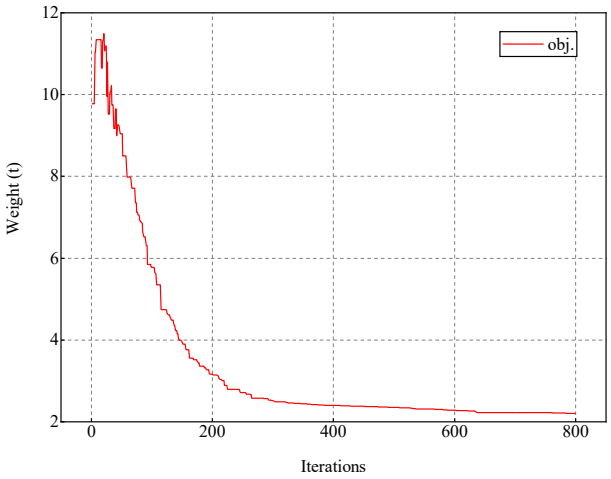


Fig. 27 The evolution process of the objective

It can be seen that the objective decreases rapidly before 300 iterations, and then slowly decreases and gradually converges. After about 600 iterations, the algorithm has obtained a satisfactory solution. Finally, some of the best results reported by other existing algorithms were selected and compared with the results of this algorithm. All the results were listed in Table 18.

Table 18
The best results reported by other algorithms

Algorithm	CSS[35]	ECSS[35]	CBO[40]
Objective (t)	2.25986	2.29861	2.16115
Algorithm	ECBO[40]	SOS-ABF[41]	CSS-BBBC[36]
Objective (t)	2.15808	2.16488	2.29861

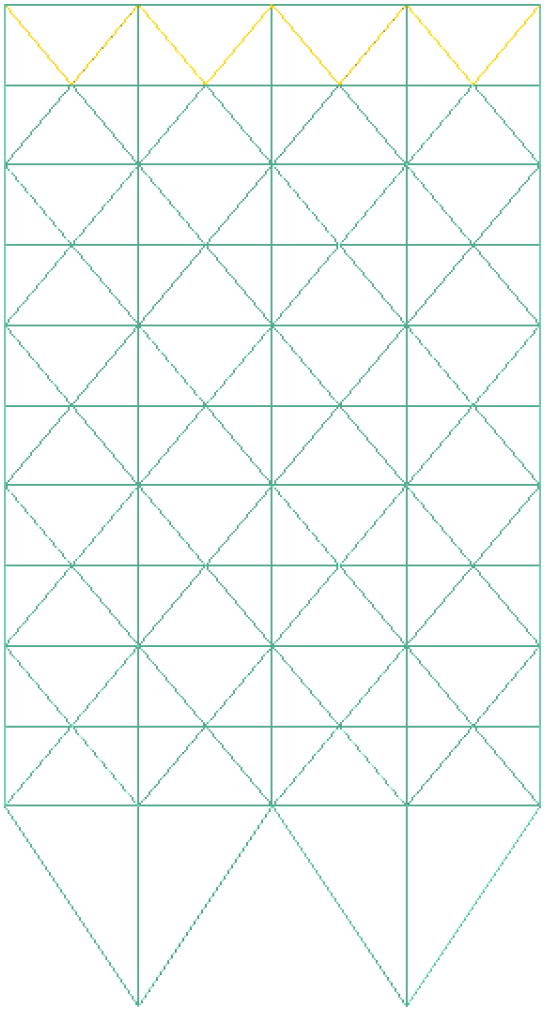
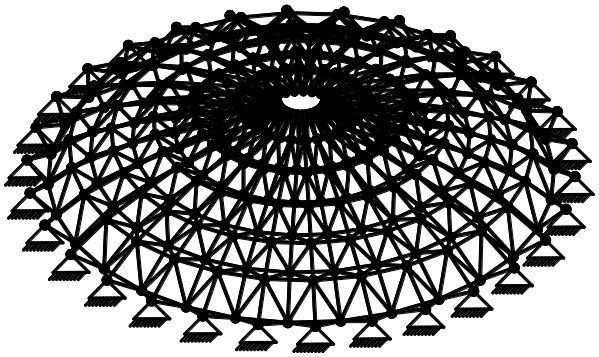


Fig. 28 The diagram of optimized model

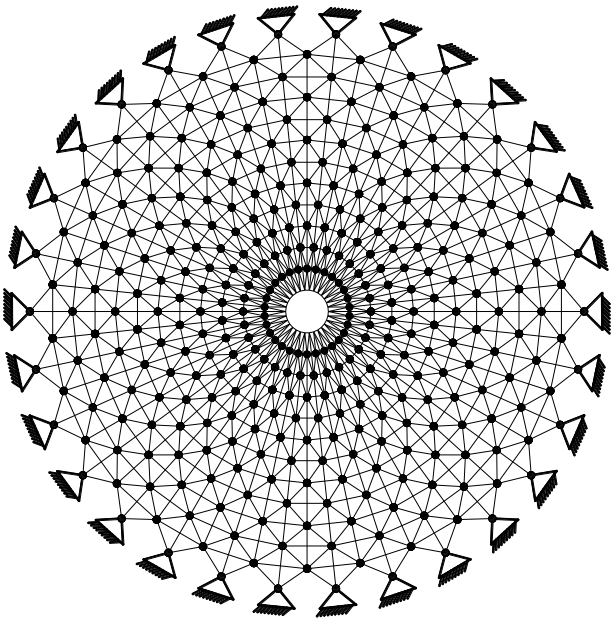
It can be seen that the best value of 2.20373t obtained by IDPGA is better than the results reported by CSS, ECSS and CSS-BBBC, worse than that reported by the other three algorithms. Similarly, this phenomenon does not indicate that the solution obtained by IDPGA is poor, the solution obtained by IDPGA is close to the best of other algorithms, and the first 3 natural frequencies of the structure obtained are 5.0057Hz, 12.708Hz and 15.128Hz respectively. Such results are acceptable in engineering, and IDPGA still has good accuracy. Accordingly, the optimized model is shown in Fig. 28.

4.3.4. Optimization design of 1410-bar double-layer lattice shell

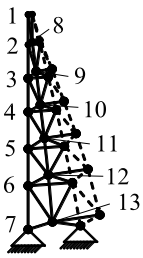
The grid structure in this problem is a double-layer lattice shell composed of 1410 elements. As shown in Fig. 29, the concentrated mass is applied to all free nodes respectively. The objective is the mass of the structure, and the constraints include natural frequencies of the first and third orders of structure. The decision variables are the cross-sectional areas of each member, considering symmetry, 47 in total in a single block.



(a) 3D view



(b) Top view (line width not shown)



(c) Diagram of a single block

Fig. 29 Diagram of the1410-bar double-layer lattice shell

The optimization framework was run once to solve the next problems. Similarly, the results were statistically analyzed as shown in Table 19.

Table 19
Optimization results

Objective (t)	Constraint violation	Time (h)
10.69489	0	89.7813

Similarly, the constraint violation of the solution is 0, indicating that the solution is a feasible solution. IDPGA takes about 90h to solve this problem. The evolution process diagram of the objective was drawn, as shown in Fig. 30.

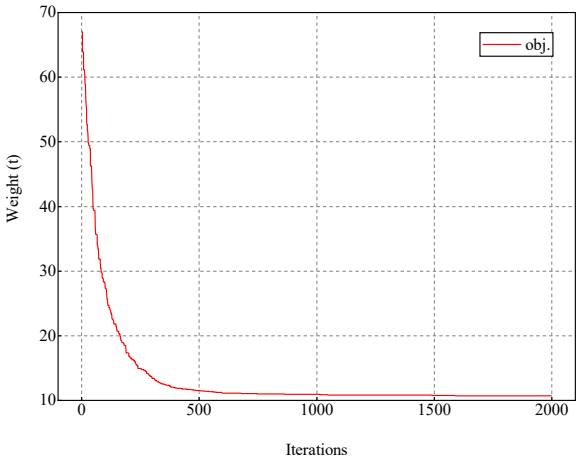


Fig. 30 The evolution process of the objective

It can be seen that the objective decreases rapidly before 500 iterations, and then slowly decreases and gradually converges. After about 700 iterations, the algorithm has obtained a satisfactory solution. Finally, some of the best results reported by other existing algorithms were selected and compared with the results of this algorithm. All the results were listed in Table 20.

Table 20
The best results reported by other algorithms

Algorithm	ECBO[42]	ECBO-Cascade[42]	DPSO[43]
Objective (t)	10.73919	10.50420	10.45384
Algorithm	VPS[44]	BB-BC[45]	HS[45]
Objective (t)	10.49183	10.77211	10.92270

It can be seen that the best value of 10.69489t obtained by IDPGA is better than the results reported by ECBO, BB-BC and HS, worse than that reported by the other three algorithms. Similarly, this phenomenon does not indicate that the solution obtained by IDPGA is poor, the solution obtained by IDPGA is close to the best of other algorithms, and the first 3 natural frequencies of the structure obtained are 7.0001Hz, 7.0001Hz and 9.0008Hz respectively. Such results are acceptable in engineering, and IDPGA still has good accuracy. Accordingly, the optimized model is shown in Fig. 31.

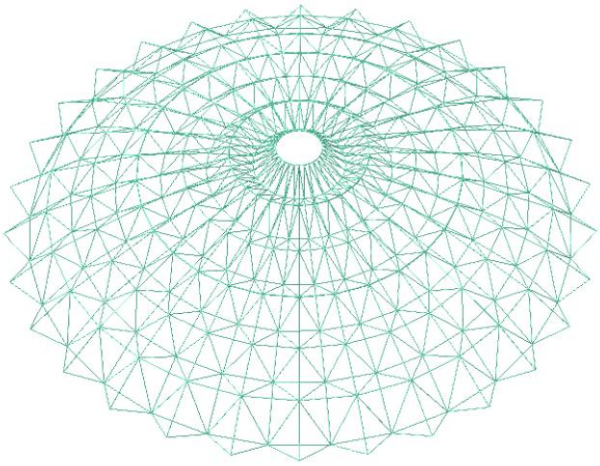


Fig. 31 The diagram of optimized model

5. Conclusions

Based on IDPGA proposed before, combined with ABAQUS, a simulation optimization framework is developed to optimize the static and dynamic performance of grid structures. In order to improve the solving efficiency, two strategies are introduced in the optimization framework, including grouping and parallel computing strategy and comparison strategy between parent and offspring. The feasibility and effectiveness of IDPGA for grid structure optimization design are verified by solving 8 engineering problems, and it proves that the algorithm has great accuracy and robustness once again.

The results show that the algorithm can converge when the iteration reaches a certain number for different problems. At the beginning of the iteration, the objective decreases rapidly and then decreases slowly and converges gradually. Before reaching the maximum number of iterations, the algorithm has found a satisfactory solution, a completely acceptable solution in the engineering field. "No free lunch theory" points out that there is no universal algorithm. In other words, no algorithm can be effective for all problems. In this paper, when solving the grid structure optimization problem, in most cases, it is randomly run once, and the obtained results are close to or better than the best values of other algorithms, which is sufficient to show that the algorithm in this paper is suitable for solving the grid structure optimization problem.

This research further expands the application of genetic algorithm and promotes the intelligent design of grid structure. Meanwhile, this research enriches the structure optimization script completed by Python language, and provides some inspiration for the application of IDPGA in the optimization design of other structures.

It is worth noting that the optimization framework takes a long time to solve the grid structure optimization problem. Some reasons are analyzed. Firstly, the idea of "dual population evolution" determines that two populations participate in the evolution at the same time, which takes longer than a single population. Secondly, due to the limitation of computer hardware, the solving efficiency will be greatly improved if the computing platform suitable for large-scale simulation is selected. Thirdly, limited by the expertise of researchers, there is still room for improvement of the interactive script between IDPGA and ABAQUS. Therefore, further improving the efficiency of the optimization framework is a research direction. For example, among the best results of adjacent 10 generations, if the difference between the best and the worst results is within 5%, it is considered that the algorithm has converged and the iteration is terminated.

In addition, parameter tuning is a current hot issue in the field of algorithms. The optimal parameter combination of an algorithm needs to be determined by other methods, which is also a future research direction of this paper. The algorithm parameters in this paper cannot be guaranteed to be optimal, but the searching results show that the combination of these parameters is reliable and effective.

Author Contributions

Formal analysis, Xuchen Xu; Investigation, Hongbo Liu; Methodology, Zhihua Chen; Software, Xuchen Xu; Supervision, Zhihua Chen; Validation, Hongbo Liu; Visualization, Zhihua Chen; Writing – original draft, Xuchen Xu; Writing – review & editing, Hongbo Liu & Ting Zhou.

Declarations

Conflict of Interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Replication of Results All scripts of algorithms are completed in Python and all contours are plotted with Matplotlib. Some or all data, models, or code that support the findings of this study are available from the corresponding author upon reasonable request.

References

[1] DOGAN E;SEKER S;SAKA M P, et al. Investigating the Effect of Joint Behavior on the Optimum Design of Steel Frames via Hunting Search Algorithm. Adv Steel Constr, v. 14, n. 2, p. 166-83, 2018.
[2] TALASLIOGLU T. Design Optimization of Tubular Lattice Girders. Adv Steel Constr, v. 15, n. 3, p. 274-87, 2019.
[3] TALASLIOGLU T. Design Optimization of Lattice Girders According to Member and Joint-Related Design Constraints. Adv Steel Constr, v. 17, n. 2, p. 181-98, 2021.
[4] KUNZ F F;SANTOS P D E;CARDOSO E U, et al. Teaching-Learning Based Optimization Method Considering Buckling and Slenderness Restriction for Space Trusses. Adv Steel Constr, v. 18, n. 1, p. 446-52, 2022.
[5] SRINIVASA K G;SRIDHARAN K;SHENOY P D, et al. A Dynamic Migration Model for self-adaptive Genetic Algorithms. Springer-Verlag Berlin, Berlin, 2005.
[6] LI Y M;ZHANG S J, ZENG X P. Research of multi-population agent genetic algorithm for feature selection. Expert Systems with Applications, v. 36, n. 9, p. 11570-81, 2009.
[7] PARK T, RYU K R. A Dual-Population Genetic Algorithm for Adaptive Diversity Control. Ieee Transactions on Evolutionary Computation, v. 14, n. 6, p. 865-84, 2010.

[8] UMBARKAR A J;JOSHI M S, HONG W C. Multithreaded Parallel Dual Population Genetic Algorithm (MPDPGA) for unconstrained function optimizations on multi-core system. Appl Math Comput, v. 243, p. 936-49, 2014.
[9] POURVAZIRI H, NADERI B. A hybrid multi-population genetic algorithm for the dynamic facility layout problem. Applied Soft Computing, v. 24, n. p. 457-69, 2014.
[10] MU Z G; CHEN Y Z; XIU L. Application of Genetic Algorithm in Optimum Design of Space Grid Structures. Spatial Structures, v. 52, p. 4, 2003, (in Chinese)
[11] MU Z G; LIANG J; SUI J, et al. Study of Optimum Design of Single Layer Dome Structures Based on Niche Genetic Algorithm. Journal of Building Structures, v. 115, p. 9, 2006. (in Chinese)
[12] LI Y M; LU X Y. Topological optimization of reticulated structures based on Genetic Algorithms. Journal of Shandong University of Architecture and Engineering, v. 8, p. 11+90, 2004. (in Chinese)
[13] CHEN Z;XU X, LIU H. Improved Dual-Population Genetic Algorithm: A Straightforward Optimizer Applied to Engineering Optimization. Sustainability, v. 15, n. 20, p. 14821, 2023.
[14] WOLPERT D H, MACREADY W G. Coevolutionary free lunches. IEEE Transactions on Evolutionary Computation, v. 9, n. 6, p. 721-35, 2005.
[15] Python 3.10.9 documentation. Python Software Foundation, BeOpen.com, Corporation for National Research Initiatives, Stichting Mathematisch Centrum. 2022.
[16] SIMULIA User Assistance 2022. Dassault Systèmes Simulia Corp, Providence, RI, USA, 2022.
[17] LI L J;HUANG Z B, LIU F. A heuristic particle swarm optimization method for truss structures with discrete variables. Computers & Structures, v. 87, n. 7, p. 435-43, 2009.
[18] TOĞAN V, DALOĞLU A T. An improved genetic algorithm with initial population strategy and self-adaptive member grouping. Computers & Structures, v. 86, n. 11, p. 1204-18, 2008.
[19] CAMP C;PEZESHK S, CAO G. Optimized Design of Two-Dimensional Structures Using a Genetic Algorithm. Journal of Structural Engineering, v. 124, n. 5, p. 551-9, 1998.
[20] KAZEMZADEH AZAD S;HASANÇEBI O, SAKA M P. Guided stochastic search technique for discrete sizing optimization of steel trusses: A design-driven heuristic approach. Computers & Structures, v. 134, n. p. 62-74, 2014.
[21] SADOLLAH A;ESKANDAR H;BAHREININEJAD A, et al. Water cycle, mine blast and improved mine blast algorithms for discrete sizing optimization of truss structures. Computers & Structures, v. 149, n. p. 1-16, 2015.
[22] HO-HUU V;NGUYEN-THOI T;VO-DUY T, et al. An adaptive elitist differential evolution for optimization of truss structures with discrete design variables. Computers & Structures, v. 165, n. p. 59-75, 2016.
[23] TAN Y, ZHU Y. Fireworks Algorithm for Optimization; Advances in Swarm Intelligence. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
[24] GHOLIZADEH S, MILANY A. An improved fireworks algorithm for discrete sizing optimization of steel skeletal structures. Engineering Optimization, v. 50, n. 11, p. 1829-49, 2018.
[25] AZAD S K, HASANÇEBI O. An elitist self-adaptive step-size search for structural design optimization. Applied Soft Computing, v. 19, n. p. 226-35, 2014.
[26] AISC. Manual of Steel Construction, Allowable Stress Design, 9th edition. American Institutes of Steel Construction, Inc, Chicago, Illinois, USA, 1989.
[27] SAKA M P. Optimum Design of Pin-Jointed Steel Structures With Practical Applications. Journal of Structural Engineering, v. 116, n. 10, p. 2599-620, 1990.
[28] TALATAHARI S;KHEIROLLAHI M;FARAHMANDPOUR C, et al. A multi-stage particle swarm for optimum design of truss structures. Neural Computing and Applications, v. 23, n. 5, p. 1297-309, 2013.
[29] KAVEH A;BAKHSHPOORI T, AFSHARI E. An efficient hybrid Particle Swarm and Swallow Swarm Optimization algorithm. Computers & Structures, v. 143, n. p. 40-59, 2014.
[30] REYNOLDS R G. An Introduction to Cultural Algorithms. Proceedings of the third annual conference on evolutionary programming. World Scientific, San Diego, California, USA, 1994.
[31] WANG G G;DEB S, COELHO L D S. Elephant Herding Optimization; 2015 3rd International Symposium on Computational and Business Intelligence (ISCBI). IEEE Computer Society, NW Washington, DC, US, 2015.
[32] JAFARI M;SALAJEGHEH E, SALAJEGHEH J. An efficient hybrid of elephant herding optimization and cultural algorithm for optimal design of trusses. Engineering with Computers, v. 35, n. 3, p. 781-801, 2019.
[33] LINGYUN W;MEI Z;GUANGMING W, et al. Truss optimization on shape and sizing with frequency constraints based on genetic algorithm. Computational Mechanics, v. 35, n. 5, p. 361-8, 2005.
[34] GOMES H M. Truss optimization with dynamic constraints using a particle swarm algorithm. Expert Systems with Applications, v. 38, n. 1, p. 957-68, 2011.
[35] KAVEH A, ZOLGHADR A. Shape and size optimization of truss structures with frequency constraints using enhanced charged system search algorithm. Asian Journal of Civil Engineering, v. 12, n. p. 2011.
[36] KAVEH A, ZOLGHADR A. Truss optimization with natural frequency constraints using a hybridized CSS-BBBC algorithm with trap recognition capability. Computers & Structures, v. 102-103, n. p. 14-27, 2012.
[37] MIGUEL L F F, FADEL MIGUEL L F. Shape and size optimization of truss structures considering dynamic constraints through modern metaheuristic algorithms. Expert Systems with Applications, v. 39, n. 10, p. 9458-67, 2012.
[38] WEI L;TANG T;XIE X, et al. Truss optimization on shape and sizing with frequency constraints based on parallel genetic algorithm. Structural and Multidisciplinary Optimization, v. 43, n. 5, p. 665-82, 2011.
[39] KAVEH A, JAVADI S M. Shape and size optimization of trusses with multiple frequency constraints using harmony search and ray optimizer for enhancing the particle swarm optimization algorithm. Acta Mechanica, v. 225, n. 6, p. 1595-605, 2014.
[40] KAVEH A, GHAZAAN M I. Enhanced Colliding Bodies Algorithm for Truss Optimization with Frequency Constraints. J Comput Civil Eng, v. 29, n. 6, p. 04014104, 2015.
[41] TEJANI G G;SAVSANI V J, PATEL V K. Adaptive symbiotic organisms search (SOS) algorithm for structural design optimization. Journal of Computational Design and Engineering, v. 3, n. 3, p. 226-49, 2016.
[42] KAVEH A, GHAZAAN M I. Optimal design of dome truss structures with dynamic frequency constraints. Structural and Multidisciplinary Optimization, v. 53, n. 3, p. 605–21, 2016.
[43] KAVEH A. Optimal analysis and design of large-scale domes with frequency constraints. Smart Structures and Systems, v. 18, n. p. 733-54, 2016.
[44] KAVEH A, ILCHI GHAZAAN M. A new hybrid meta-heuristic algorithm for optimal design of large-scale dome structures. Engineering Optimization, v. 50, n. 2, p. 235-52, 2018.
[45] KAVEH A, ZOLGHADR A. Optimal design of cyclically symmetric trusses with frequency constraints using cyclical parthenogenesis algorithm. Advances in Structural Engineering, v. 21, n. 5, p. 739-55, 2018.