Journal of Engineering, Project, and Production Management
2026, 16(1), 2025-152

Integrating Particle Swarm, Genetic Algorithms, and Differential Evolution for Prefabricated Formwork Optimization

Fang Luo[1] and Songfeng Fang[2]
[1] Associate Professor, School of Civil Engineering, Zhengzhou College of Finance and Economics, Zhengzhou, 450000, China, E-mail: luo_luofang@outlook.com (corresponding author).
[2] Senior Engineer, Henan Zhengtong Construction and Installation Engineering Co., Ltd., Zhengzhou, 450000, China

Project and Production Management
Received August 8, 2025; revised September 23, 2025; accepted September 24, 2025
Available online December 24, 2025

Abstract: The increasing adoption of prefabricated construction has made formwork a key factor that directly affects project cost, construction time, and quality. Traditional optimization methods, however, often struggle to address the variability of prefabricated components and the complexity of production environments. To overcome these limitations, this study develops an optimization model for prefabricated formwork by combining particle swarm optimization, genetic algorithm, and differential evolution within a three-swarm cooperative framework. The model considers objectives related to cost, construction period, and quality, and applies the analytic hierarchy process to support multi-criteria decision-making. Experimental validation with real production data shows that the model reduced the production time of a single component from 34.93 hours to 29.89 hours. For a complete order of 17 components, total labor hours decreased by 27.32% and costs by 23.56%. These results confirm that the proposed approach not only improves optimization performance but also delivers practical value for scheduling and resource allocation in prefabricated construction projects.

Keywords: AHP, architecture, formwork, optimization, prefabrication.

DOI 10.32738/JEPPM-2025-152

1. Introduction

The prefabricated building market is experiencing substantial growth, fueled by the advancement of “dual carbon” goals and the rise of construction industrialization. As the main component in prefabricated building production, formwork directly impacts cost, construction duration, and quality consistency (Fazeli et al., 2022). Data indicate that optimizing formwork can cut material waste by 15% to 20% and reduce construction time by 10% to 15%, making it a key factor in lowering costs and boosting efficiency in the industry (Han et al., 2023). Traditional optimization methods mainly depend on mathematical modeling and operations research to find the best solutions. However, due to varying component sizes and changing site conditions, overly simplified models often create gaps between theoretical outcomes and real-world practice. Particle Swarm Optimization (PSO) offers benefits such as a simple structure and quick convergence, making it effective for finding near-optimal solutions (Liu et al., 2022). Differential Evolution (DE), based on swarm intelligence, demonstrates strong global search ability and robustness (Bujok et al., 2023). Genetic Algorithm (GA), inspired by natural selection and genetic evolution, is suitable for large-scale, complex problems requiring global optimization (Alam and Arya, 2022). Nevertheless, each algorithm also faces limitations, such as a tendency to get stuck in local optima, limited local search accuracy, low computational efficiency, and premature convergence. To tackle these challenges, this study develops a three-swarm differential PSO algorithm that combines the strengths of all three methods. By improving data processing, clarifying problem constraints, and including diversity maintenance strategies, the model aims to optimize formwork for prefabricated building components. Its goal is to lower costs, shorten construction times, and improve quality. This study presents a new method by applying these three algorithms within a cooperative swarm structure, offering innovative solutions for formwork optimization in prefabricated construction.

2. Related Works

In the field of optimization algorithms, intelligent optimization methods have developed rapidly to address complex problems. Studies on multi-swarm Differential Evolution (DE) and Improved Particle Swarm Optimization (IPSO) have

been widely conducted by scholars from different countries. These methods have been applied in various domains. For example, Pirozmand et al. (2023) proposed a multi-adaptive learning PSO algorithm to address the impact of task scheduling on service quality in cloud computing environments. By defining two types of particles to reduce initial population diversity, their experiments showed that this method achieved better efficiency and accuracy in multi-objective tests. Yang et al. (2022) put forward a clustering-based competitive PSO algorithm to solve the problem of sparse optimization being overlooked in Pareto frontiers. This method combined PSO with competitive group optimizers to balance the search process. Nama and Saha (2022) proposed an improved multi-swarm Backtracking Search Algorithm to enhance global optimization performance. They introduced a new mutation strategy and adaptive parameter updates. Ankita and Sahana (2022) developed a balanced PSO algorithm to solve the challenge of effective application scheduling in dynamic grid environments. By optimizing scheduling, the algorithm reduced job execution time, improved resource utilization, and offered scalability. Experiments confirmed that it outperformed various deterministic and heuristic methods. Parouha and Verma (2022) addressed the limitation of traditional meta-heuristic algorithms that often fall into local optima. They introduced a hybrid algorithm based on DE and PSO, adding new strategies and parameter adjustments to balance global and local search capabilities. Results showed that this hybrid algorithm performed better than other similar approaches.

Although extensive studies have been conducted on multi-swarm DE and IPSO, their application in the field of prefabricated buildings has remained limited. Some progress has been made using other approaches in this domain. For example, Du et al. (2023) proposed an improved biogeography-based optimization method to address the impact of ignoring participant behavior in the production of prefabricated components. Case studies demonstrated that this method optimized production planning and improved the ability to handle uncertainty, ultimately enhancing project efficiency. Zou and Feng (2023) developed a multi-objective simulation optimization method to overcome the negative effect of uncertainty on the performance of prefabricated construction. By refining activity logic to balance objectives, their experiments demonstrated significant improvements in construction time and cost, thereby supporting informed project decision-making. Almashaqbeh and El-Rayes (2022) developed an optimization model to reduce the high costs associated with the transportation and storage of prefabricated modules. The model was constructed by identifying variables, defining objective functions, and setting constraints. Zou and Feng (2023) developed an optimized prefabricated component library and simulation model to enhance efficiency and reduce costs in Building Information Modeling (BIM) for prefabricated buildings. Their experiments demonstrated that the method increased efficiency, lowered costs, and improved overall project performance. Xiao and Bhola (2022) carried out collaborative design using Building Information Modeling to address its limitations in the design of prefabricated buildings.

Although existing research has made progress in optimisation algorithms and certain aspects of prefabricated construction, most work either emphasizes general algorithmic innovation or focuses on localized issues such as transportation, storage, or component libraries. Systematic research remains relatively scarce concerning the optimisation of prefabricated formwork, which directly impacts production costs, construction schedules, and quality. Furthermore, existing methods exhibit limited robustness testing across diverse production scenarios and offer insufficient guidance for construction management practices. Consequently, a significant research gap persists in developing an integrated, application-oriented framework for optimising prefabricated formwork. To address this deficiency, this study proposes a tri-population collaborative optimisation model integrating PSO, GA, and DE algorithms, supplemented by the Analytic Hierarchy Process (AHP) for multi-objective decision-making. Unlike previous studies, this research explicitly integrates computational intelligence methods with scheduling and resource allocation problems in prefabricated construction processes. While achieving methodological innovation, it further emphasises its application value in engineering management practice.

3. Model Construction for Template Optimization of Prefabricated Components

3.1. Design of Three-Swarm Differential PSO Combined with GA and DE

Prefabricated component templates must be optimized in terms of cost control and production cycle to enhance the production efficiency and quality of prefabricated buildings (Pereira et al., 2022). This study integrates GA and DE with PSO through parallel and cooperative operations to computationally solve the above problem. PSO shares information among particles in the swarm, enabling different individuals to explore various areas in the solution space to find the optimal solution by continuously updating particle positions and velocities (Demir and Sahin 2023). The velocity update equation is shown in Eq. (1).

$$v_{i\dim}(k+1) = \omega v_{i\dim}(k) + c_1 r_1 [pbest_{i\dim}(k) - x_{i\dim}(k)] + c_2 r_2 [gbest_{i\dim}(k) - x_{i\dim}(k)] \tag{1}$$

In Eq. (1), k denotes the number of iterations. $v_{i\dim}(k)$ is the velocity of particle i in the dimensional space. x is the position of the particle. ω is the inertia weight. c_1 and c_2 are the cognitive and social learning factors, respectively. r is a random number between 0 and 1. $gbest$ represents the personal best, and $pbest$ represents the global best. The position update equation is shown in Eq. (2).

$$x_{i\dim}(k+1) = x_{i\dim}(k) + v_{i\dim}(k+1) \tag{2}$$

In Eq. (2), k is the number of iterations. $x_{i\dim}(k)$ is the current position of particle i in the dimensional space. v is

the particle's velocity. In traditional PSO, the inertia weight is usually constant, limiting the ability to adapt velocity over iterations. However, in practice, the inertia weight should vary with iterations. A nonlinear inertia weighting method accelerates the reduction of speed in the early and middle stages, improving global search performance. The nonlinear inertia weight is calculated by Eq. (3).

$$\omega = \omega_{\max} - (\omega_{\max} - \omega_{\min}) \ln(1 + iter / iter_{\max}) \tag{3}$$

In Eq. (3), $\omega_{\max}$ and $\omega_{\min}$ represent the maximum and minimum values of the inertia weight, respectively. $iter$ is the current iteration number, and $iter_{\max}$ is the maximum iteration number. The learning factors affect the global optimization ability of PSO. Instead of using fixed values, the learning factors are optimized as shown in Eq. (4).

$$\begin{cases} c_1 = c_{1s} - (c_{1s} - c_{1e}) \times \dfrac{iter}{iter_{\max}} \\ c_2 = c_{2s} - (c_{2s} - c_{2e}) \times \dfrac{iter}{iter_{\max}} \end{cases} \tag{4}$$

In Eq. (4), c_{1s} and c_1 are the initial values of c_{2s} and c_2, while c_{1e} and c_{2e} are their final values, respectively. Learning factors that vary with iterations enhance global search ability and speed, making it easier to achieve the global optimum through fine local search. The flowchart of the IPSO is shown in Fig. 1.

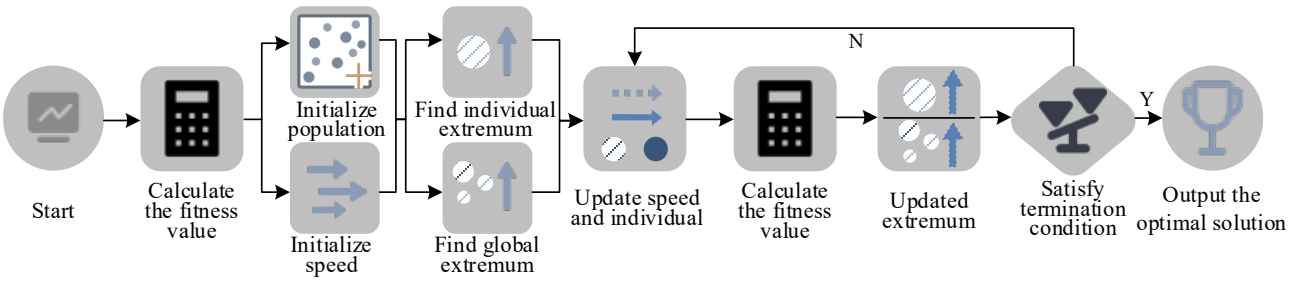


Fig. 1. Schematic diagram of the workflow of the IPSO algorithm

As shown in Fig. 1, the process begins by calculating the fitness value, followed by the initialization of the swarm and particle velocities. It then searches for the personal and global best positions. The particle velocity and position are updated using the improved equations. Fitness values are then recalculated. If the termination condition is met, the optimal solution is output. Otherwise, the process continues to iterate. Although IPSO offers several advantages, running it independently may result in premature convergence, which lowers accuracy and limits particle diversity (Parouha and Verma, 2022; Bujok et al., 2023). DE addresses this issue by evolving solutions through mutation, crossover, and selection. It is simple in principle, converges quickly, and offers strong robustness (Tiwari et al., 2024). Therefore, DE is integrated to generate better individuals using its mutation mechanism. The mutation operation is shown in Eq. (5).

$$\theta_{ir} = x_{b_1}^t + \eta \times (x_{b_2}^t + x_{b_3}^t) \tag{5}$$

In Eq. (5), b denotes a random and distinct function. i represents the target individual. t is the current iteration, and r is the dimensional index. x is the target vector, and θ is the mutant vector's value in the same dimension, i.e., the new mutant individual. η is the scaling factor. The crossover operation is shown in Eq. (6).

$$P_{ir} = \left(1 - rand(S_C)\right) \times \theta_{ir} + rand(S_C) \times x_{ir} \tag{6}$$

In Eq. (6), P is the crossover vector, or the new offspring. $rand$ is the function used to generate a random number. The comparison of the second-generation fitness values is shown in Eq. (7).

$$X_i^{t+1} = \begin{cases} P_{ir}, P_i \le F(X_i) \\ X_i^t \end{cases} \tag{7}$$

In Eq. (7), X_i^t is the position vector before iteration, and $F(X_i)$ is the fitness function value of the target vector. The DE algorithm flowchart is shown in Fig. 2.

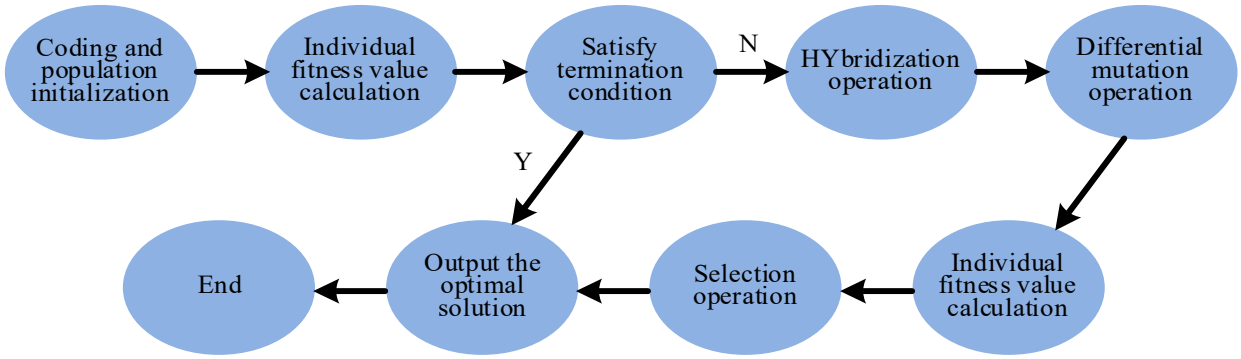


Fig. 2. Schematic diagram of DE algorithm operation process

As shown in Fig. 2, the process begins by initializing individuals and calculating their fitness values. New individuals are generated through mutation and crossover. Their fitness is evaluated, and the process iterates until the termination condition is met, after which the optimal solution is output. When IPSO stagnates, DE helps it escape from local optima. However, combining the two still faces the challenge of a single search strategy. In complex, high-dimensional problems, performance may degrade, convergence may slow, and the algorithm may still fall into local optima. To address this, GA is introduced to complement the deficiencies by simulating natural selection, crossover, and mutation to iteratively explore potential solutions. In the selection phase, the probability of an individual being selected for reproduction is proportional to its fitness. The probability calculation is shown in Eq. (8).

$$p_i = \frac{f(a_i)}{\sum_{j=1}^{n} f(a_j)} \tag{8}$$

In Eq. (8), a represents a vector encoding the individual's features. p is the probability of being selected. f is the fitness value. n is the total number of individuals. i and j are individuals in n. In the crossover phase, selected individuals generate offspring through crossover, and the resulting offspring are represented in Eq. (9).

$$a' = (a_{i1}, a_{i2}, ..., a_{ik}, a_{j(k+1)}, ..., a_{jn}) \tag{9}$$

In Eq. (9), a' represents the offspring of a. The DE algorithm modifies certain genes of newborn individuals with a small probability, thereby introducing mutations and increasing the diversity of the population. The mutation operation is shown in Eq. (10).

$$a''_{nk} = a_{nk} + \delta, with probability \mu \tag{10}$$

In Eq. (10), a'' is the mutated gene. μ is the mutation rate. δ is a small random change. Repeating these steps across generations enables GA to gradually improve the quality of solutions and approach the optimal. However, GA may encounter difficulties such as a large search space and local optima in complex problems. ANN has a strong learning ability, and its rapid learning feature can guide GA to enhance search efficiency (Stojanović et al., 2022). The operation flow chart of ANN-GA is shown in Fig. 3.

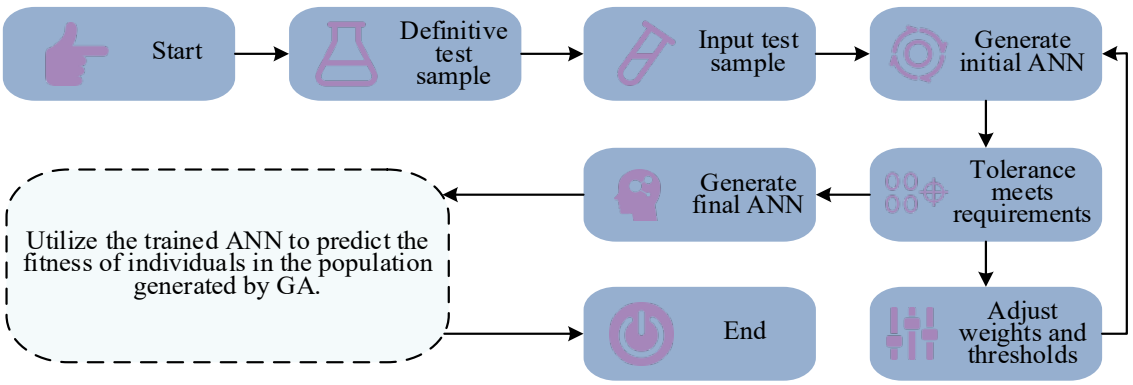


Fig. 3. ANN-GA operation flow chart

As shown in Fig. 3, the process begins with random sampling to select test samples, which are then input into the ANN prediction module for training. Next, the initialization module is created, and a tolerance analysis is performed. If the tolerance meets the requirements, the final ANN module is generated. Otherwise, weights and thresholds are adjusted. The initial parameters of ANN-GA are set to generate the optimal prediction module. Based on the defined constraints and

optimization goals, the fitness value is calculated. If the termination condition is met, the result is output. If not, the selected individuals undergo crossover and mutation to create a new population, returning to the parameter setting step for the next iteration. These three algorithms run in parallel and work together. IPSO provides an initially optimized region for DE and ANN-GA, narrowing the search scope. DE continuously generates diverse individuals during the search process, providing enriched material for ANN-GA. ANN-GA further refines the solutions produced by IPSO and DE, selecting better results. Their cooperation enhances global search capabilities, local refinement, and multi-objective handling.

3.2. Development of the Template Optimization Model for Prefabricated Components

After developing the three-swarm differential PSO, this study builds an optimization model for prefabricated building component formworks based on the algorithm and practical requirements. The optimization process needs to consider cost, construction period, and quality. The cost includes materials, manufacturing, and transportation. The cost objective function sums all items based on material types and unit prices. The construction period is calculated by summing the time for each stage from design to installation. The quality is measured by accuracy and strength. These three objectives are shown in Eq. (11).

$$\begin{cases} F_C = \sum_{i=1}^{n_m} p_i q_i + \sum_{j=1}^{n_f} r_j h_j + \sum_{k=1}^{n_t} s_k l_k \\ F_T = \sum_{u=1}^{n_T} t_u \\ F_Q = \omega_1 \dfrac{1}{E} + \omega_2 S \end{cases} \tag{11}$$

In Eq. (11), F_C represents the cost objective function. p_i is the unit price of material, and q_i is the quantity. r_j is the unit cost of each manufacturing step, and h_j is the workload. s_k is the unit cost of transportation, and l_k is the transportation distance. n_m, n_f, and n_t are the number of material types, process steps, and transportation segments, respectively. F_T represents the construction period objective function. t_u is the time of each stage, and n_T is the number of stages. F_Q represents the quality objective function. E and S are the indicators of accuracy, error and strength. ω_1 and ω_2 are the corresponding weight coefficients. In formwork optimization, population diversity affects whether the algorithm can find high-quality solutions (Kunakh et al., 2023). This matches the use of the three-swarm differential PSO. Before running the algorithm, this study introduces the average Euclidean distance among individuals in the population as the diversity evaluation index. The Euclidean distance and its average expression are shown in Eq. (12).

$$\begin{cases} d_{ij} = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2} \\ D = \dfrac{2}{N(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} d_{ij} \end{cases} \tag{12}$$

In Eq. (12), N is the number of individuals. x_{jk} is the value of the individual j on the k-th parameter. d_{ij} is the Euclidean distance between individuals, and D is the average Euclidean distance. In multi-dimensional space, population diversity is measured using individual distances. A lower average indicates insufficient diversity. Some individuals are reinitialized by randomly resetting parameters or adding small random disturbances. The evaluation process is shown in Fig. 4.

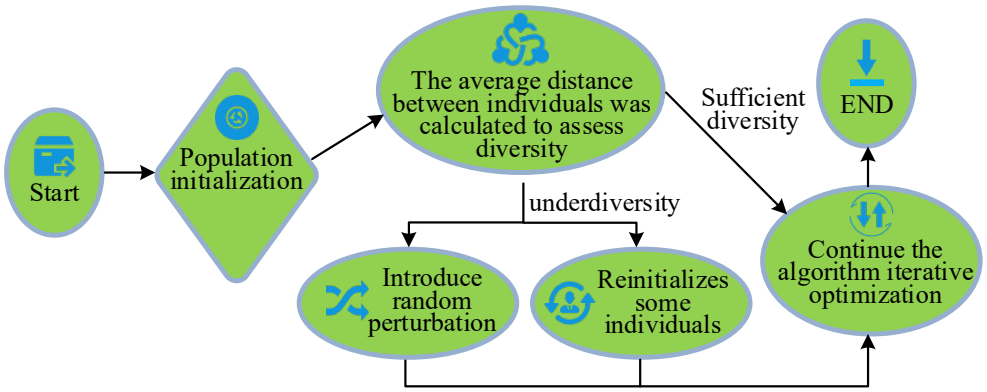


Fig. 4. Schematic diagram of the diversity assessment process

As shown in Fig. 4, after initializing the algorithm, it calculates the average distance between individuals to assess diversity. If the diversity is sufficient, the algorithm proceeds with iterative optimization. Otherwise, it introduces random disturbances and reinitializes some parameters. After improving diversity, the algorithm performs parallel computation using the three-swarm differential PSO. The geometric mean of the results is used to improve stability and adaptability. In the cooperation stage, the algorithm exchanges top-performing individuals based on fitness ranking to enhance optimization. However, the objectives of formwork optimization may conflict. This study applies the AHP to support decision-making. The AHP structure is shown in Fig. 5.

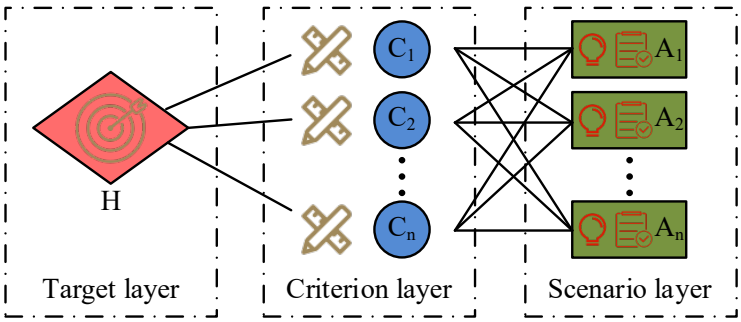


Fig. 5. Schematic diagram of the mechanism of AHP

As shown in Fig. 5, AHP first divides the multi-objective problem into a goal layer, a criteria layer, and an alternatives layer. Then it constructs a judgment matrix to compare the importance of each element and determine its weight. Finally, it calculates the comprehensive score of each candidate under different objectives, ranks them, and selects the optimal one that meets practical needs. This method provides scientific and rational decision-making for real production and helps to find better solutions in multi-objective optimization. By combining the objective functions, diversity evaluation, AHP, and the proposed three-swarm differential PSO, the study builds an optimization model for formwork. The complete process is shown in Fig. 6.

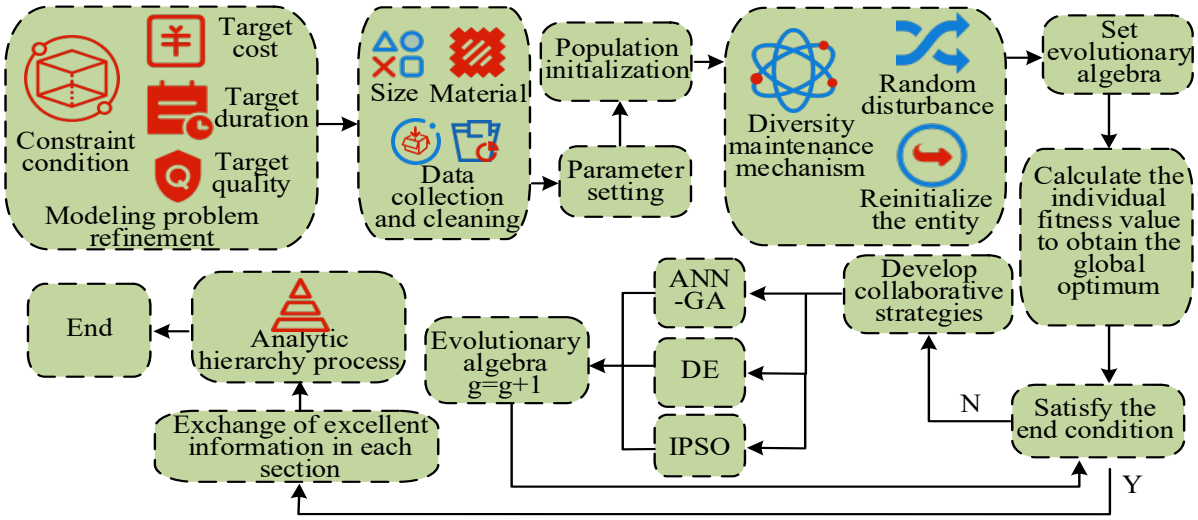


Fig. 6. Optimization process of prefabricated building formwork

As shown in Fig. 6, the process begins by defining objectives and constraints for cost, time, and quality, followed by data collection and cleaning. After parameter setting and population initialization, a diversity maintenance mechanism is applied through random disturbances or reinitialization. The algorithm then evaluates fitness, identifies the global best, and checks termination conditions. If unmet, a cooperation strategy is executed until convergence. The three-swarm differential PSO operates in parallel, increments the generation count, and repeats the process. Once the termination condition is satisfied, the algorithm exchanges top-performing individuals among swarms. It then applies AHP to process the results and outputs the optimized formwork solution.

4. Experimental Analysis of the Proposed Prefabricated Building Model

4.1. Experimental Setup

To ensure the fairness and validity of the proposed optimization model, this study designed the following experimental setup. Firstly, during the algorithm performance validation phase, the Holder function, Rastrigin function, and Zitzler-Deb-Thiele (ZDT) test set were selected as benchmark functions to evaluate the convergence and accuracy of single-objective and multi-objective optimization. Secondly, during the practical application validation phase, production data from a precast component factory were collected through field surveys and structured interviews. This included key indicators such as process duration, material consumption, and defect rates. The specific details of the experimental platform's software and hardware environment are presented in Table 1.

Table 1. Experimental hardware and software environment configuration

Category	Project	Configuration notes
Hardware environment	Processor	Intel Core i7-12700H, 2.3 GHz, 14cores
	Memory	16 GB DDR4
	Storage	512 GB SSD
	Graphics card	NVIDIA RTX 3060, 6 GB
Software environment	Operating system	Windows 10 (64 bites) / Ubuntu 20.04
	Programming environment	Python 3.6 / MATLAB R2021a
	Libraries/Toolkits	NumPy, SciPy, Matplotlib, OpenCV

4.2. Performance Verification of the Improved Three-Swarm Differential PSO

To evaluate the performance of the improved multi-population differential PSO, the proposed algorithm was compared with two hybrid algorithms: Whale Optimization Algorithm combined with PSO (WOA-PSO) and DE combined with Artificial Bee Colony (DE-ABC). The three algorithms were tested on the Rastrigin and Holder benchmark functions. The Rastrigin function was executed 20 times, and the results are shown in Fig. 7.

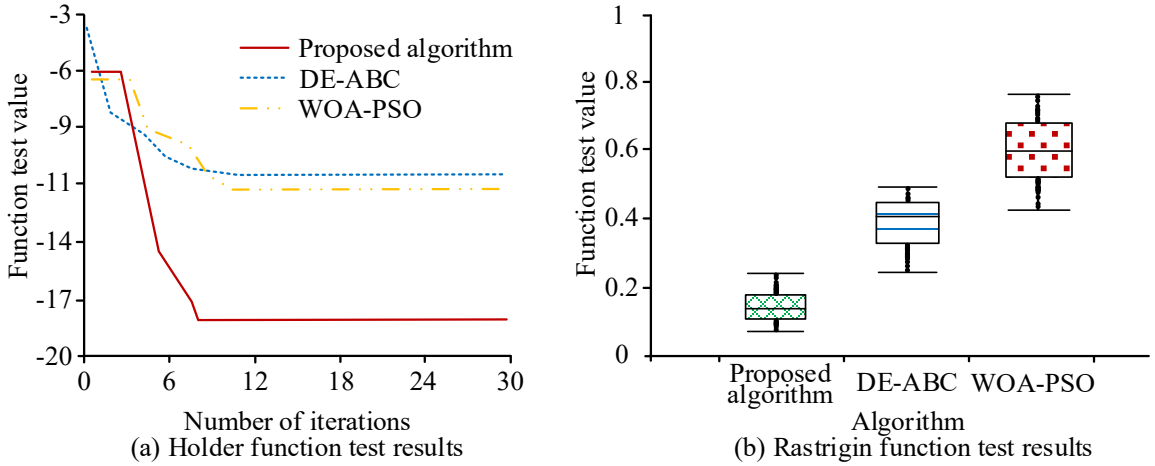


Fig. 7. Performance comparison of test functions

As shown in Fig. 7(a), in the Holder function test, the proposed algorithm achieved a rapid decline in function values as the number of iterations increased, reaching -18 at the 8th iteration and then stabilizing. DE-ABC showed a decrease from -2 to -10.5 by the 11th iteration and then stabilized. According to Fig. 7(b), during the 20 runs of the Rastrigin function test, the function values of the proposed algorithm ranged from 0.08 to 0.23, with an average of 0.16. These results indicated that the proposed algorithm achieved a good balance between computational efficiency and accuracy, while also demonstrating faster convergence. To analyze the algorithm's capability in solving single-objective problems, a series of standard test functions and the ZDT benchmark set were selected for further evaluation. The specific expressions of the functions are shown in Table 2.

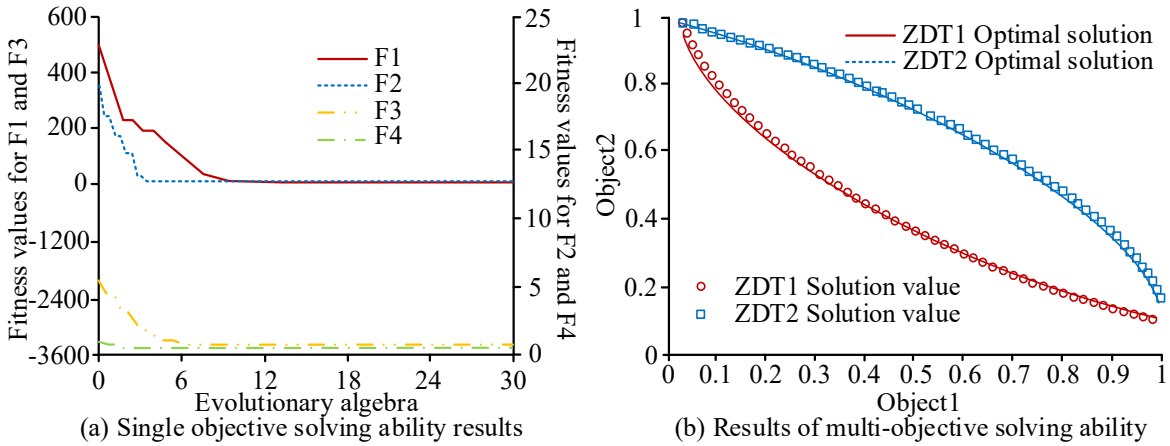
Table 2. Test functions used in the experiment

Function	Boundary	Optimal value	Optimal solution condition	Function feature	Variable dimension
$F1=\sum_{i=1}^{20}x_i^2+(\sum_{i=1}^{20}0.5ix_i)^2+(\sum_{i=1}^{20}0.5ix_i)^4$	[-5,10]	0	/	/	/
$F2=-20\exp(-0.2\sqrt{\frac{1}{20}\sum_{i=1}^{20}x_i^2})-\exp(\frac{1}{20}\sum_{i=1}^{20}\cos(2\pi x_i^2))+20+e$	[-32,32]	0	/	/	/
$F3=-(\frac{3}{0.05+x^2+y^2})^2-(x^2+y^2)^2$	[-32,32]	-3600	/	/	/
$F4=1+\sin^2(x)+\sin^2(y)-0.1\exp(-(x^2+y^2)^2$	[-10,10]	0.9	/	/	/

Table 2. Test functions used in the experiment (continued)

Function	Boundary	Optimal value	Optimal solution condition	Function feature	Variable dimension
$ZDT1\begin{cases} f_1(x)=x_1 \\ f_2(x)=g(x)(1-\sqrt{f_1(x)/g(x)}) \\ g(x)=1+9\sum_{i=2}^{n}(x_1)/(n-1) \end{cases}$	[0,1]	/	$\begin{cases} x_1\in[0,1] \\ x_i=0 \\ i=2,\ldots,n \end{cases}$	Convex	$n=30$
$ZDT2\begin{cases} f_1(x)=x_1 \\ f_2(x)=g(x)[1-(f_1(x)/g(x))^2] \\ g(x)=1+9\sum_{i=2}^{n}(x_1)/(n-1) \end{cases}$	[0,1]	/	$\begin{cases} x_1\in[0,1] \\ x_i=0 \\ i=2,\ldots,n \end{cases}$	Non-convex	$n=30$

As presented in Table 2, F1 consisted of a combination of power terms, F2 the Ackley function, F3 involved fractional and power operations, and F4 combined trigonometric and exponential components. ZDT1 and ZDT2 were two sets of multi-objective functions. Through this diverse set of functions, the study evaluated the proposed algorithm's performance in solving both convex and non-convex functions in multi-objective optimization scenarios. The optimization results are displayed in Fig. 8.



(a) Single objective solving ability results (b) Results of multi-objective solving ability

Fig. 8. Optimization curves of different objective functions

As shown in Fig. 8(a), the initial fitness values of functions F1 to F4 varied in the single-objective tests. A noticeable downward trend emerged after 5, 4, 6, and 3 generations, respectively, eventually approaching zero or a stable value. Fig. 8(b) shows that, in the multi-objective tests, the proposed algorithm generated tightly clustered solutions when solving ZDT1 and ZDT2, which have convex and non-convex characteristics, respectively. These findings demonstrated that the proposed algorithm achieved accurate convergence and evenly distributed solutions when handling functions with different features, delivering strong performance in both single-objective and multi-objective optimization.

4.3. Performance Verification of the Component Optimization Model

After validating the performance of the proposed algorithm, further tests were conducted to assess its practical application in optimizing the formwork scheduling model for prefabricated components in construction. Experiments were conducted using MATLAB R2021a on a Windows 10 system with 16GB RAM and an Intel Core i7-12700H CPU. Prefabricated component order data were collected from a local manufacturer through field visits and interviews. The proposed model was compared with two hybrid algorithms: PSO combined with Ant Colony Optimization (PSO-ACO) and Tabu Search combined with GA (TS-GA). One prefabricated component was selected; the average processing duration was calculated for multiple repetitions of each operation. Operations N1 to N9 represented stages such as initial preparation, rebar processing, and embedded part production. The Gantt charts of the optimization results from the three models are presented in Fig. 9.

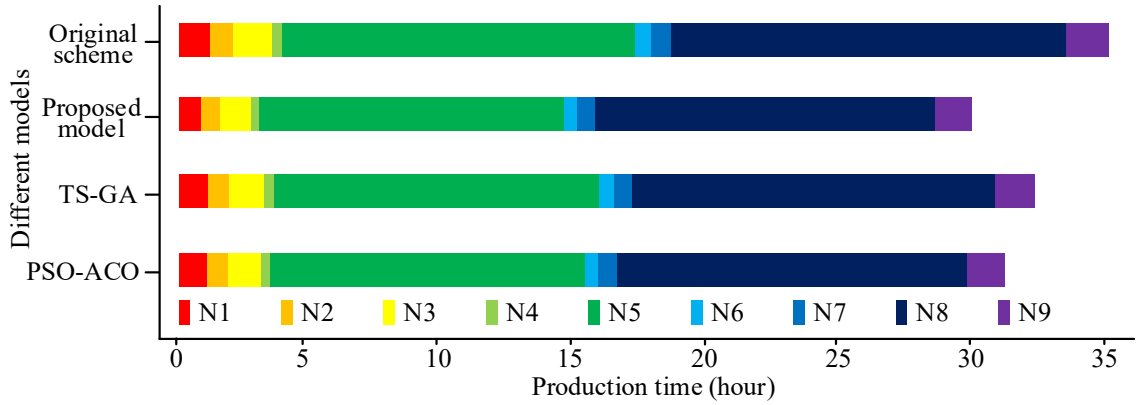


Fig. 9. Optimization comparison for a single component by different models

As shown in Fig. 9, the total production time of the original plan was 34.93 hours, while the proposed model reduced it to 29.89 hours. TS-GA required 33.15 hours, and PSO-ACO required 32.11 hours. Among all operations, the proposed model achieved the best improvement in operation N8, reducing the time from 16.24 hours in the original plan to 12.56 hours—a reduction of 3.68 hours or 22.66%. The least improvement was observed in operation N6, which was reduced from 0.61 hours to 0.56 hours—a decrease of 0.05 hours or 8.20%. Overall, the proposed model shortened the total production time by 5.04 hours, representing a 14.43% reduction. This highlighted the proposed model's advantage in optimizing the formwork process for a single prefabricated component. To demonstrate the model's optimization advantages from multiple perspectives, four additional component types were included. The experiments considered factors such as waiting time for production resources, rework time within acceptable defect rates, and penalties for late delivery. The results are shown in Fig. 10.

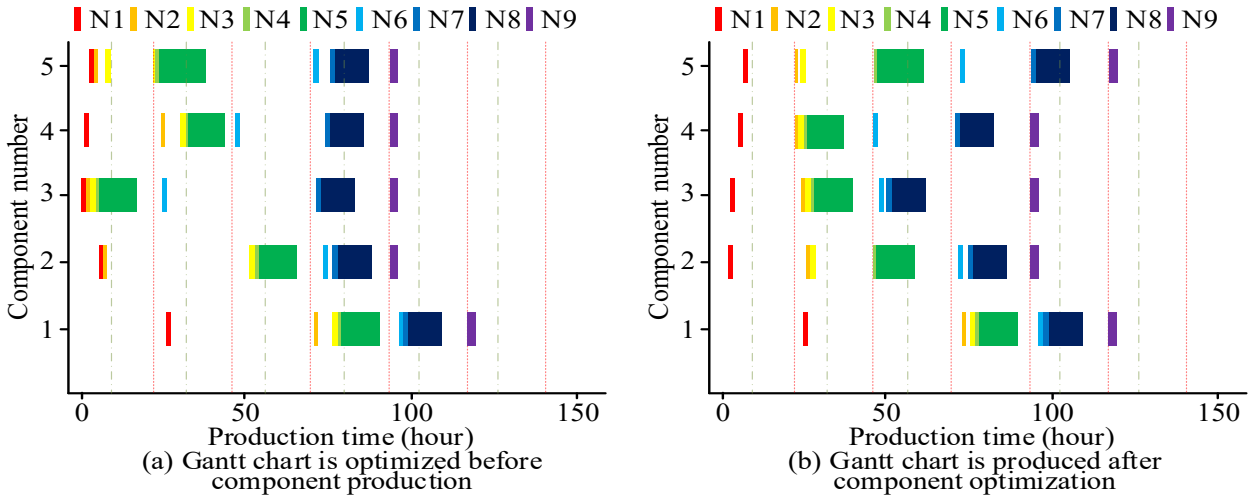


Fig. 10. Multi-component optimization comparison by different models

As shown in Fig. 10(a), the original production sequence was 2, 3, 4, 5, 1. The distribution of operation durations was relatively scattered, resulting in an extended total production time. This suggested inefficiencies and a lack of compact scheduling. Fig. 10(b) illustrates that, after optimization using the proposed model, the operations became more compact, with significantly reduced idle times. The new production sequence was adjusted to 3, 4, 5, 2, 1. For instance, the operation sequence of component 3 became more continuous, allowing for better time utilization and reducing overall production time. These results demonstrated that the optimized plan significantly improved scheduling efficiency and the arrangement of operations. To verify whether the optimization results of the proposed model translated into improvements in cost and labor hours, a complete order containing 17 prefabricated components was optimized using different models. The results are shown in Table 2 and illustrated in Fig. 11.

As shown in Fig. 11(a), the production sequence of the 17 components was arranged as 8, 6, 7, 14, 3, 9, 17, 4, 5, 12, 10, 16, 11, 1, 13, 15, 2. The production sequences optimized by TS-GA and the proposed model were noticeably different from the original plan, while the sequence from PSO-ACO partially overlapped with the original. Fig. 11(b) reveals that the original plan required 167.50 hours with a cost of 3026 yuan. TS-GA reduced the time to 136.50 hours and the cost to 2694 yuan. PSO-ACO reduced the time to 149.25 hours and the cost to 2815 yuan. The proposed model performed the best, reducing labor hours to 121.75 hours—a 27.32% decrease—and cutting costs to 2313 yuan—a 23.56% reduction. These comparisons showed that each model had different effects on labor hours, costs, and production sequences. The proposed model demonstrated clear advantages in both time and cost savings.

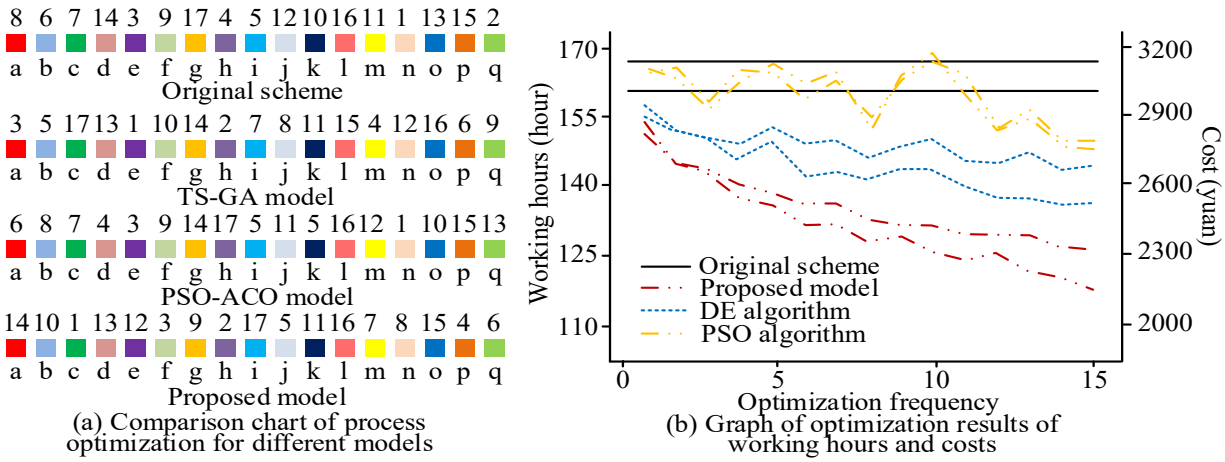


Fig. 11. Optimization comparison for the complete component order

5. Conclusion

With the rapid development of the prefabricated construction industry, optimizing component templates has become a key step in improving the efficiency of construction projects. This study proposes a multi-population differential PSO approach and develops a corresponding optimization model. Experimental results showed that the proposed multi-population differential PSO reduced the test value of the Holder function to -18 within 8 iterations and then remained stable. In the 20 runs of the Rastrigin function test, the results ranged from 0.08 to 0.23, with an average of 0.16, all of which were better than those of the DE-ABC and WOA-PSO algorithms. In the optimization of a single-component production task, the proposed model reduced the total production time from 34.93 hours to 29.89 hours, a reduction of 5.04 hours or 14.43%. When considering waiting times for production resources, rework within acceptable loss rates, and penalties for late delivery, the proposed optimization approach improved production efficiency. For the full order of 17 components, the proposed model reduced the total working hours from 167.50 hours to 121.75 hours, a 27.32% reduction, and decreased the cost from 3026 yuan to 2313 yuan, a 23.56% reduction. Although the model improved task scheduling and enhanced production efficiency, it remained relatively computationally complex when handling large-scale and complex projects. In addition, limited attention was given to robustness and generalization. Future research can focus on addressing these issues in depth.

Author Contributions

Fang Luo contributes to manuscript editing, methodology. Songfeng Fang contributes to analysis, investigation, and data collection.

Funding

The research is supported by Henan Provincial Science and Technology Research Project, Research on Optimization Design and Application of Prefabricated Building Component Templates Based on Machine Learning (No.: 252102320314).

Institutional Review Board Statement

Not applicable.

References

Alam, S. J. and Arya, S. R. (2022). Volterra LMS/F based control algorithm for UPQC with multi-objective optimized PI controller gains. *IEEE Journal of Emerging and Selected Topics in Power Electronics*, 11(4), 4368-4376.

Almashaqbeh, M. and El-Rayes, K. (2022). Minimizing transportation cost of prefabricated modules in modular construction projects. *Engineering, Construction and Architectural Management*, 29(10), 3847-3867.

Ankita and Sahana, S. K. (2022). Ba-PSO: A balanced PSO to solve multi-objective grid scheduling problem. *Applied Intelligence*, 52(4), 4015-4027.

Bujok, P., Lacko, M., and Kolenovský, P. (2023). Differential evolution and engineering problems. *Mendel*, 29(1), 45-54.

Demir, S. and Sahin, E. K. (2023). Predicting occurrence of liquefaction-induced lateral spreading using gradient boosting algorithms integrated with particle swarm optimization: PSO-XGBoost, PSO-LightGBM, and PSO-CatBoost. *Acta Geotechnica*, 18(6), 3403-3419.

Du, J., Xue, Y., Sugumaran, V., Hu, M., and Dong, P. (2023). Improved biogeography-based optimization algorithm for lean production scheduling of prefabricated components. *Engineering, Construction and Architectural Management*, 30(4), 1601-1635.

Fazeli, A., Jalaei, F., Khanzadi, M., and Banihashemi, S. (2022). BIM-integrated TOPSIS-Fuzzy framework to optimize selection of sustainable building components. *International Journal of Construction Management*, 22(7), 1240-1259.

Han, Y., Yan, X., and Piroozfar, P. (2023). An overall review of research on prefabricated construction supply chain management. *Engineering, Construction and Architectural Management*, 30(10), 5160-5195.

Kunakh, O. M., Volkova, A. M., Tutova, G. F., and Zhukov, O. V. (2023). Diversity of diversity indices: which diversity measure is better? *Biosystems Diversity*, 31(2), 131-146.

Liu, H., Yang, L., and Yang, H. (2022). Cooperative optimal control of the following operation of high-speed trains. *IEEE Transactions on Intelligent Transportation Systems*, 23(10), 17744-17755.

Nama, S. and Saha, A. K. (2022). A bio-inspired multi-population-based adaptive backtracking search algorithm. *Cognitive Computation*, 14(2), 900-925.

Parouha, R. P. and Verma, P. (2022). A systematic overview of developments in differential evolution and particle swarm optimization with their advanced suggestion. *Applied Intelligence*, 52(9), 10448-10492.

Pereira, J. L. J., Oliver, G. A., Francisco, M. B., Cunha Jr, S. S., and Gomes, G. F. (2022). A review of multi-objective optimization: methods and algorithms in mechanical engineering problems. *Archives of Computational Methods in Engineering*, 29(4), 2285-2308.

Pirozmand, P., Jalalinejad, H., Hosseinabadi, A. A. R., Mirkamali, S., and Li, Y. (2023). An improved particle swarm optimization algorithm for task scheduling in cloud computing. *Journal of Ambient Intelligence and Humanized Computing*, 14(4), 4313-4327.

Stojanović, B., Gajević, S., Kostić, N., Miladinović, S., and Vencl, A. (2022). Optimization of parameters that affect wear of A356/Al2O3 nanocomposites using RSM, ANN, GA and PSO methods. *Industrial Lubrication and Tribology*, 74(3), 350-359.

Tiwari, P., Mishra, V. N., and Parouha, R. P. (2024). Developments and design of differential evolution algorithm for non-linear/non-convex engineering optimization. *Archives of Computational Methods in Engineering*, 31(4), 2227-2263.

Xiao, Y. and Bhola, J. (2022). Design and optimization of prefabricated building system based on BIM technology. *International Journal of System Assurance Engineering and Management*, 13(1), 111-120.

Yang, J. Q., Yang, Q. T., Du, K. J., Chen, C. H., Wang, H., Jeon, S. W., and Zhan, Z. H. (2022). Bi-directional feature fixation-based particle swarm optimization for large-scale feature selection. *IEEE Transactions on Big Data*, 9(3), 1004-1017.

Zou, Y. F. and Feng, W. H. (2023). Cost optimization in the construction of prefabricated buildings by using BIM and finite element simulation. *Soft Computing*, 27(14), 10107-10119.

Fang Luo, Associate Professor, is affiliated with the School of Civil Engineering, Zhengzhou College of Finance and Economics. Her main research interests and directions include civil engineering mechanics and structural engineering, as well as research on the development of intelligent green buildings.

Songfeng Fang, Senior Engineer, is employed by Henan Zhengtong Jian'an Construction Engineering Co., Ltd. His research interests and directions include prefabricated bridge construction and trenchless technology research.