buildings

MDPI

Article

Multi-Objective Optimization of Production Sequence and Layout of Precast Concrete Components on the Mold Table Under Limited Mold Quantity

Junyong Liang [1,2], Yong Liu [2], Xiaotao Sun [3], Wenxiang Xu [4,*] and Baigang Du [5]

[1] School of Civil Engineering, Sichuan University of Science and Engineering, Zigong 643000, China; liangjunyong@suse.edu.cn
[2] Luzhou Key Laboratory of Intelligent Construction and Low-Carbon Technology, Luzhou 646000, China
[3] Building Environment Engineering Technology Research Center, Sichuan University of Arts and Science, Dazhou 635000, China
[4] School of Mechanical Engineering, Hubei University of Arts and Science, Xiangyang 441000, China
[5] School of Mechanical and Electronic Engineering, Wuhan University of Technology, Wuhan 430070, China
* Correspondence: xuwenxiang910327@126.com

Abstract

Precast concrete components, as one of the important structural systems in prefabricated buildings, have received widespread attention due to their efficient manufacturing characteristics on the production line. Their production sequence and layout on the mold table have a crucial impact on production energy consumption. However, a critical constraint is often overlooked in the first step of precast concrete manufacturing: the production sequence and layout of molds are planned without considering the limited availability of molds for each component type. Therefore, this article proposes a mixed-integer programming model for the production sequence and layout of precast concrete components under a limited number of molds, aiming to simultaneously minimize production energy consumption, fluctuation coefficients of mold table utilization, and mold switching time. To obtain high-quality solutions for production sequence and mold layout, a multi-objective genetic flatworm algorithm with a Tabu mapping mechanism is developed to efficiently determine the production sequence and the positions of molds on the mold tables. Through three production cases of precast concrete components with different scales, the proposed model and algorithm have been demonstrated to be highly effective in assisting decision-makers in quickly formulating the optimal production sequence and layout schemes for precast concrete components.

Keywords: precast concrete components; production sequence; layout optimization; multi-objective genetic flatworm algorithm; Tabu mapping mechanism

Received: 29 December 2025
Revised: 30 January 2026
Accepted: 25 February 2026
Published: 28 February 2026

1. Introduction

With the accelerated pace of global urbanization and the booming development of the construction industry, the pursuit of efficiency, quality, and environmental standards in construction projects has become increasingly stringent. Prefabricated buildings, with their innovative models of prefabricated production and rapid on-site assembly, are gradually becoming an important driving force for the transformation and upgrading of the construction industry [1]. By manufacturing precast concrete components (PC/PCs) in factories and assembling them efficiently on-site, the model not only realizes the standardization, industrialization, and informatization of the construction process but also greatly improves the

construction efficiency and reduces the construction cost [2–4], while effectively reducing construction waste and environmental pollution and exploring a new path for the pursuit of sustainability in construction practices [5].

Relevant government departments in China attach great importance to the development of prefabricated buildings. The '14th Five-Year Plan' for the development of the construction industry clearly proposed that, by 2025, a target be set requiring that prefabricated buildings account for over 30% of all new construction [6,7]. Sichuan Province has set a high target of over 40% and emphasized that the assembly rate should not be less than 50%. In this context, PCs, as the core component of prefabricated building systems, have a dual guarantee of production efficiency and quality, which is crucial for ensuring the successful implementation of the entire building project.

The production of PCs is a discrete manufacturing process highly dependent on mold technology [8]. As a key tool in the production process, the rationality of the mold design and the efficiency of the layout directly determine the overall performance of the production line. Mold layout is the optimal strategy for planning the layout of PCs within a limited space on the mold tables, aiming to maximize mold tables' utilization, shorten production cycle times, and reduce costs [9]. This process not only requires designers to possess exceptional intelligence and innovation capabilities but also tests the coordination and scheduling abilities of production managers. In view of the continuous growth of construction demand and the increasingly fierce competition in the market, it is particularly important to study the optimization of the mold layout for the production of precast PCs. Optimization of mold layout can bring multiple benefits: Firstly, it significantly improves production efficiency by reducing the frequency of mold replacement through scientific and reasonable layout design, shortening the production cycle time, and improving the continuous operation capability of the production line, thereby achieving rapid production growth while ensuring component quality. Secondly, optimizing the layout of PC molds can avoid mold turnover waste, improve mold table utilization, reduce mold wear and maintenance costs, and ultimately lower total production costs, creating greater economic benefits for the enterprise. Finally, it can also promote the standardization and automation of the PC production process, further enhancing the management level and market competitiveness of enterprises [10–12].

Although scholars have made many research achievements in the field of PC production planning and scheduling, there is still a lack of in-depth research on the optimization of PC production mold layout [9]. Especially in the production environment of limited mold resources and multiple varieties and small batches, effectively evaluating the impact of mold layout on PC production, balancing the workload of the mold tables, saving energy consumption, and minimizing the negative effects of mold switching have become complex and challenging topics. The main contribution of this paper includes the following aspects:

1. A mixed-integer programming (MIP) model is constructed for the production sequence and layout optimization under mold constraints (PC-PLO-MC), which considers the limitation of mold quantity and the production spacing requirements, enhancing the practical relevance of the PC-PLO-MC model.
2. With respect to the current concerns of PC production managers, three optimization objectives are defined in the PC-PLO-MC model, including minimizing the production energy consumption, balancing the utilization rate of the mold tables, and reducing mold switching time.
3. A multi-objective genetic flatworm algorithm with a Tabu mapping mechanism (GFA–Tabu) is developed to solve PC-PLO-MC, and a PC positioning selection strategy called weighted matching filling is designed. Finally, the effectiveness of the algo-

rithm in optimizing PC production sequence and layout is verified by three cases of different scales.

The rest of this article is organized as follows. Section 2 reviews the relevant areas of focus in current PC production. Section 3 describes the optimization problem of PC production layout under mold constraints and constructs the MIP model. Section 4 details the logical steps of the work of the proposed GFA–Tabu. Section 5 validates the applicability of the proposed PC-PLO-MC model and the computational efficiency of the GFA–Tabu method through a total of four numerical cases of three different scales. The last section summarizes the conclusions of PC-PLO-MC.

2. Related Works

PC-PLO-MC has been proven to be similar to the cutting stock problem [13] and the two-dimensional bin packing problem [14,15], but it is far more complex than both. A great deal of theoretical research and practical work has been conducted by researchers and PC producers, respectively. This section mainly focuses on three aspects: the PC production process and production organization, PC production scheduling optimization, and molds in the PC production.

2.1. PC Production Process and Production Organization

The production process of PC usually undergoes the following procedures [16]. (1) Mold preparation and assembly. The molds matching the PCs to be produced are selected and precisely fixed on the mold table. (2) Installation of steel reinforcement skeleton and embedded parts. Accurately place and fix the elements to be pre-buried in the PC, such as electrical conduits, junction boxes, etc., to the designated locations. (3) Concrete pouring. The concrete is evenly cast into the mold, and the process needs to ensure that there are no air bubbles and voids inside the concrete, and the surface is flat and smooth. (4) Curing stage. The PCs are moved to a specialized curing room with a high-temperature and high-humidity environment for the PC to accelerate the hardening process as soon as possible. (5) Demolding. When the concrete strength reaches 75% or more of the design strength, the PCs can be removed from the curing room together with the molds, and the molds can be removed so that the PC is completely exposed. (6) Post-processing and inspection. After demolding, conduct a detailed inspection of the surface of the PC, and if necessary, perform precision machining such as polishing and repairing.

In terms of PC production organization, two main modes currently exist: fixed-mold table production and flow mold table production [17]. Despite their differences, both share the six core processes outlined above. The key distinction lies in the role of the mold table: in fixed-mold table production, the table remains stationary while operators, reinforcement, and concrete move between tables. This method offers broad applicability and strong versatility, making it suitable for producing non-standard PCs [18]. The other method is a mode of continuous production through automated or semi-automated production lines in a specific sequence and rhythm. The PC production line is organized into multiple workstations corresponding to the processes outlined above. Each workstation is equipped with dedicated operators and tools to perform one or a few specific tasks. During the production process, the operators remain stationary while the mold tables are transported between workstations by the production line system. This method has the advantages of high automation, high productivity, and low production costs, and it is particularly suitable for large-scale, standardized production of PCs [19].

2.2. PC Production Scheduling Optimization

Optimizing the production sequence and layout of PCs plays a key role in production planning and scheduling [18]. Since the initial proposal by Warszawski [20], numerous scholars have dedicated significant effort to this field of research. Subsequently, Chan et al. developed a foundational flow shop scheduling model for PC production, laying the groundwork for subsequent studies on PC production scheduling [10,19,21]. Subsequent research has explored various dimensions of this problem. For instance, production scheduling in a hybrid flow shop environment—where standard PCs are made-to-stock and non-standard units are made-to-order—has been examined [22]. The scheduling in multi-factory PC production has also been investigated to improve delivery performance [23]. Recognizing the continuous nature of concrete pouring and curing processes, researchers have introduced blocking constraints between mold tables to enhance model realism [7]. Dynamic scheduling models accounting for uncertainties in construction project progress have been developed, reducing the likelihood of delivery delays [12,24,25]. More recently, a comprehensive model incorporating mold preparation, PC inventory, and delivery has been proposed to improve the accuracy of the whole supply chain [17]. Recently, a production redundancy scheduling optimization model has been developed to address the issue of unstable production interruptions in PCs to minimize interruption losses in PC production [26].

The primary objectives of optimizing the production planning and scheduling of PCs can be broadly categorized into four main areas: minimizing the makespan [16,18,19,21,27–29], reducing production costs [7,30–32], minimizing inventory costs [23,33,34], and mitigating the financial penalties incurred due to delayed or early delivery. The reduction in bottlenecks and construction delays [35], the maximization of profitability and supply security [36], and so forth. The field of production scheduling is inherently complex, with the NP-hard problem representing a significant challenge [23,37]. The complexity of PC production scheduling has prompted the development of three optimization paradigms: exact algorithms, heuristics, and meta-heuristics. Exact methods—including mathematical analysis [38], MIP [39], and branch and bound [40]—provide optimal solutions but face scalability issues. Heuristic algorithms constructed based on intuition or experience can optimize the makespan with acceptable time consumption [41]. As problem size increases, however, both categories encounter limitations, driving research toward meta-heuristic algorithms. This class of methods encompasses a range of techniques, including the classical genetic algorithm [16,42], particle swarm optimization [43,44], simulated annealing [37], and hybrid algorithms such as the hybrid differential firefly algorithm [45], genetic gray wolf optimizer [46], and others. Recently, simulation [47] and reinforcement learning-based approaches [48,49] have also demonstrated efficacy in PC scheduling.

2.3. Mold Resources in PC Manufacturing

PCs flow unidirectionally between workstations on the mold table throughout a production cycle, and all of the above literature has examined the problem of production sequencing of PCs. Regarding the role of mold resources in PC production, the inclusion of identical parallel molds with shared resources in PC production was studied, aiming to reduce the total product cost [50]. Lim addressed the challenge of optimizing both production scheduling and stacking area design for PCs, considering mold table limitations [51]. The influence of custom mold availability on PC processing time was explored by Benjaoran et al. and Yang et al. [52,53]. Hu combined tool scheduling with flow shop optimization to achieve balanced tool use and reduce peak-demand-induced waste in PC production [10].

2.4. Research Gap

Research on PC production sequence and layout on the mold table remains limited. Zheng et al. applied grouping techniques to PC allocation on the mold table, addressing scheduling challenges in mixed flow production [54,55]. However, their reliance on a 0–1 random assignment method, which neglected the geometric and positional relationships between components, failed to provide a viable solution for mold assembly—a significant gap relative to actual production practices. Wang et al. addressed this limitation by proposing a MIP model that incorporates order demand, mold table dimensions, and constraints related to component positioning. Their approach maximizes the average utilization of mold tables during assembly [9]. However, this approach suffers from two shortcomings. One is its failure to account for mold quantity constraints arising from high production costs. The other is its reliance on a predefined, limited number of PC arrangement patterns on the mold table, limiting real-world usefulness.

Therefore, optimizing the production sequence and layout of PCs subject to the limited molds is of considerable importance. This study focuses on addressing such efficiency bottlenecks in PC manufacturing practice. A mixed-integer programming model is developed, which establishes three core optimization objectives: minimizing production line energy consumption, balancing mold table utilization, and reducing mold switching time. Building on this, a multi-objective genetic flatworm algorithm incorporating a Tabu mapping mechanism is proposed to solve the problem of PC-PLO-MC, with its performance effectively validated through practical case studies.

3. Mathematical Model for PC-PLO-MC

3.1. Problem Statement

All six PC fabrication processes are carried out on a series of mold tables along the production line. As illustrated in Figure 1, two mold tables at the assembly workstation hold eight components spanning five different sizes.

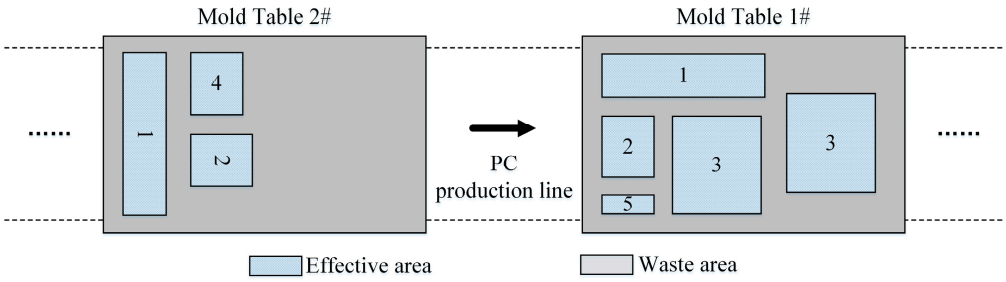


Figure 1. Diagram of the assembly of PC molds on the production line.

The space occupied by PCs constitutes the effective utilization area of the mold table, while the remaining area is wasted. A larger total occupied area of PCs on the mold tables corresponds to a higher utilization rate. For the curing process with high energy consumption, a higher number of PCs can be cured simultaneously with the limited capacity of the curing kiln. Therefore, increasing the overall utilization of the mold tables contributes to a marked reduction in energy consumption.

Table 1 presents the order information for four types of PCs, with each column representing the corresponding mold quantity, order demand quantity, PC entity size for the concrete casting part, and the length of the extended steel bars in four directions. It should be noted that the area occupied by a PC on the mold table includes both its concrete section and the extended reinforcement portion. The impact of the production sequence and layout of PCs in Table 1 on energy consumption, production line balance, and mold switching time will be elaborated upon in Section 3.3.

Table 1. Production information for 4 types of PCs (P4).

PC	Mold Quantity	Order Demand	Geometric Size (mm)					
			Length	Width	L_{left}	L_{right}	L_{top}	L_{bottom}
1#	2	4	1610	1260	90	90	90	150
2#	1	2	3520	960	90	90	90	150
3#	2	3	4020	2400	90	90	0	0
4#	3	5	2520	1560	90	90	90	150

3.2. Assumptions

1. The types, geometries, and order demand quantities of all PCs waiting to be produced are known.
2. The number of molds for each PC type is predetermined, regardless of the combination of molds. During the production of each type of PC, the corresponding molds participate in the layout according to the maximum available quantity.
3. Parallel alignment with the mold table edge is maintained for all PC molds in the layout, and non-parallel placement is not considered.
4. The number of mold tables is sufficient.
5. There is no difference in the demand deadline for all PCs in the production order.
6. All coefficients directly related to PC production are known.

3.3. Mathematical Model of PC-PLO-MC

When arranging PC molds on the mold table, the following geometric dimensions, positional constraints, and associated quantitative constraints should be satisfied:

$$0 \le P_{is} \le Q_i \quad \forall i \in I, s \in S \tag{1}$$

$$\sum_{s=1}^{N_s} P_{is} = d_i \quad \forall i \in I \tag{2}$$

$$d/2 \le x_i \le L - (1 - r_i)l_i - r_i w_i - d/2 \quad \forall i \in I \tag{3}$$

$$d/2 \le y_i \le W - (1 - r_i)w_i - r_i l_i - d/2 \quad \forall i \in I \tag{4}$$

$$x_i + (1 - r_i)l_i + r_i w_i + d \le x_j \quad \forall i, j \in I \tag{5}$$

$$x_j + (1 - r_j)l_j + r_j w_j + d \le x_i \quad \forall i, j \in I \tag{6}$$

$$y_i + (1 - r_i)w_i + r_i l_i + d \le y_j \quad \forall i, j \in I \tag{7}$$

$$y_j + (1 - r_j)w_j + r_j l_j + d \le y_i \quad \forall i, j \in I \tag{8}$$

$$\sum_{i=1}^{I} z_{ip} = 1 \quad \forall p \in P \tag{9}$$

$$r_i = 0 \text{ or } 1, r_j = 0 \text{ or } 1 \quad \forall i, j \in I \tag{10}$$

Equation (1) denotes that the quantity of molds allocated to the ith PC does not exceed a given value in the sth layout. Equation (1) denotes that all PCs are produced after several layouts, and the demand of the order is satisfied. Equations (3) and (4) indicate that the molds for any PC are placed inside the mold table, and the spacing between each PC and the edge of the mold table is not less than half of the production spacing, where the production spacing $d = 300$ mm. On the mold table, any two PCs cannot overlap each other, and their feasible relative positional relationships can be up-down or left-right. Therefore, the coordinate constraint relationship between them can be expressed as Equations (5)–(8) for a set of redundant constraints, and only one of which needs to be satisfied in practice. In a PC's production sequence, the PC's category at any location is uniquely determined

and can be constrained by Equation (9). Equation (10) represents the feasible arrangement direction of the PCs, where 0 corresponds to parallel orientation between the PC length and mold table length, and 1 means that the two directions are perpendicular to each other.

Based on the above layout constraints, the aim is to minimize the following three objectives:

$$\begin{aligned} \min f_1 &= f_{11} + f_{12} \\ f_{11} &= (w_1 + w_2) \sum_{i \in I} l_i w_i h_i \\ f_{12} &= w_3 \left(\alpha \sum_{i \in I} l_i w_i h_i + \beta \sum_{k \in K} y_k \right) \end{aligned} \tag{11}$$

$$\begin{aligned} \min f_2 &= \sqrt{\sum_{k \in K} (\eta_k - \max\{\eta_k\})^2 / \sum_{k \in K} y_k} \\ \eta_k &= \sum_{i \in I} l_i^k w_i^k h_i^k / LW \end{aligned} \tag{12}$$

$$\min f_3 = w_4 \sum_{k \in K} y_k + w_5 \sum_{p=1}^{P} z_p \tag{13}$$

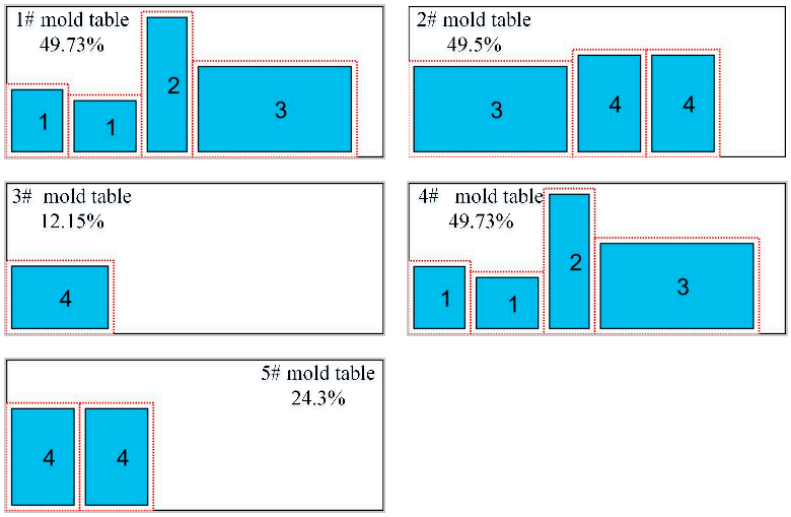
Equation (11) represents the total energy consumption of the PCs, which, in terms of energy type, consists mainly of water, electricity, and natural gas apportioned to the PCs by the production line system. In terms of the production process, f_1 can be divided into two parts, where f_{11} is called the process energy consumption, which consists of the energy consumption of the hot curing process and five other processes, and is positively proportional to the volume of the PCs. f_{12} is called the transportation energy consumption (also known as the non-process energy consumption), which is mainly related to the weight of the PCs to be produced and of the utilized mold tables. It is assumed that each cubic meter of reinforced concrete is 2.5 tons, and each mold table is 10 m long, 4 m wide, and weighs 6 tons, with a thickness of 60 mm for all PCs. For calculation purposes, energy consumption is measured in kilograms of standard coal equivalent (kgce). Based on the norm of energy consumption per unit of PCs, w_1 is set to 25 kgce/m^3, w_2 is set to 5 kgce/m^3, and w_3 is set to 6 kgce/t. It should be noted that the total energy consumption does not include the energy consumed by the PCs from the factory delivery to the construction site. Thus, it is clear that energy consumption is closely related to carbon emissions, with lower total energy consumption indicating a relatively cleaner production process.

η_k in Equation (12) represents the utilization efficiency of the kth mold table, expressed as the ratio of the occupied area of all PCs to the whole area of the utilized mold tables. f_2 represents the balance coefficient of the utilization of the production line's mold tables during the completion of all processes, which indicates the fluctuation degree of the utilization efficiency across all mold tables or the operators' workload. Very similar to the standard deviation, the smaller the indicator, the more balanced the workload of different mold tables at the same workstation in the entire assembly line; to a large extent, the situation of sometimes idle and sometimes busy is avoided.

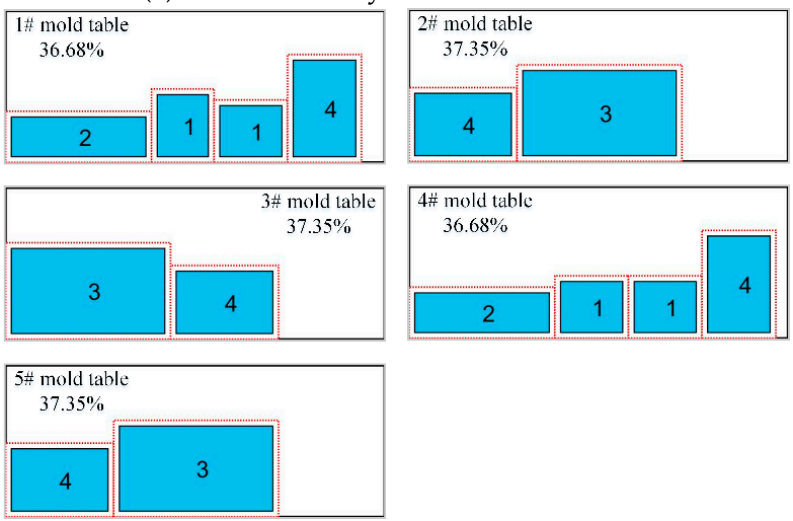
f_3 represents the mold switching time, which consists of two parts. w_4 is the switching time between two mold tables, which is assumed to be 10 min, and w_5 is the mold adjustment time for two different PCs, which is set to 2 min, and we assume that the mold adjustment time for two consecutive identical PCs is 0. In a PC layout sequence, the total number of switching times for PCs can be stated as the $\sum_{p=1}^{P} z_p$. The switching time for the first precast component is counted as $z_1 = 1$. The lower the value, the less non-value-added time it takes for mold switching and adjustment, which is highly favored by workshop managers.

The effect of the PC production sequence and layout scheme on the optimization objectives can be demonstrated using P4 in Table 1, assuming that there are three dif-

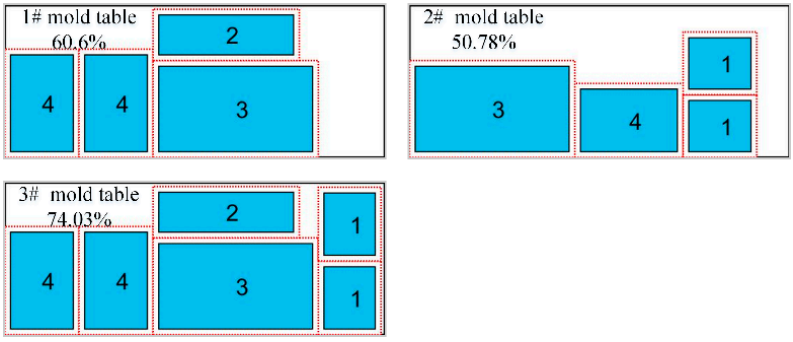
ferent PC production sequences for the same production task, S_1 = {11233444,112344}, S_2 = {21144334,211443}, and S_3 = {44334211,443211}. Obviously, the limitations on the number of molds result in all three scenarios requiring the mold to cycle at least two times to satisfy the order demand. The expression of production sequences will be detailed in the encoding part. Subject to the above constraints, three different production scenarios are shown in Figure 2, with the dark rectangles indicating the PCs placed on the mold tables and the red dashed boxes indicating the production process interval boundaries.



(**a**) Production layout solution for $\boldsymbol{S_1}$.



(**b**) Production layout solution for $\boldsymbol{S_2}$.



(**c**) Production layout solution for $\boldsymbol{S_3}$.

Figure 2. Examples of three production layout schemes.

Intuitively, five mold tables are required by both S_1 and S_2 to complete the layout task, while S_3 requires only three, leading to significantly higher energy consumption in S_1 and S_2. The usage rate per mold table is given in the blank areas in Figure 2, respectively. According to Equation (12), the mold table utilization balance rate for scheme S_1 is 20.29, while it is only 0.43 for S_2. Although the balanced utilization of all three mold tables in scheme S_3 is above 50%, which is significantly higher than that of the other two schemes, its f_2 is 15.5, which indicates that the production line has a better rhythmicity under scheme S_2. Ideally, the shorter the mold switching time is when all the PCs of the same type are arranged in a layout together, so scheme S_1 has the shortest switching time of 16 min. However, considering the limitation of the number of molds, the different placement order of PCs, and the preparation time for switching and cleaning the mold tables, the scheme shown in Figure 2c saves more non-value-added labor time.

PC-PLO-MC simultaneously minimizes energy consumption, mold utilization balance index, and mold switching time. The case in Figure 2 also proves the conflict between the three objectives; that is, the three objectives cannot be minimized simultaneously. Therefore, PC-PLO-MC belongs to multi-objective optimization problems. To evaluate the quality of the solution, the idea of the Pareto optimal solution (also known as the Pareto non-dominated solution) is employed, and a detailed description can be found in the classical non-dominated sorting genetic algorithm (NSGAII) [56]. Based on the characteristics of the combinatorial explosion of PC production layout schemes under the constraint of mold quantities, approaches to maximize mold table utilization while ensuring balanced workloads at each workstation, while reducing the energy consumption of the production process, and minimizing the non-value-added operation time make PC-PLO-MC more challenging and complex.

4. Genetic Flatworm Algorithm with Tabu Mapping Mechanism

The flatworm algorithm (FA), inspired by the unique growth, splitting, and regeneration mechanisms of flatworms in nature, has been ingeniously transformed and designed by Tseng and his partners [57] into an efficient search and optimization algorithm. This algorithm not only cleverly draws on the unique biological characteristics of flatworms, making its principles and operation process intuitive and easy to understand, but also stands out in the field of combinatorial optimization with its global search ability, such as disassembly sequence planning. However, the single-starting-point design of FA limits its parallel computing capabilities, making it difficult to realize its full potential in multi-core or distributed computing environments, which in turn affects search efficiency.

The standard genetic algorithm (GA) also has strong global optimal solution search ability, but extensive practice has shown that GA has the shortcomings of poor local search ability and premature convergence, while the above-mentioned FA has superior performance in local search. In response to this limitation, our research team has integrated genetic algorithms' population diversity and hybridization advantages into FA and successfully constructed an optimized version called the genetic flatworm algorithm (GFA) [58] for multipoint parallel computing.

Although this improvement has significantly enhanced the algorithm's parallel processing capability, the combination of FA's built-in growth, splitting, and regeneration heuristics and the stochastic nature of the genetic operation during execution inevitably leads to the emergence of overlapping search paths and duplicate solutions, which in turn may degrade the search efficiency and quality. Therefore, we redesigned the famous Tabu list and embedded it into the genetic flatworm algorithm to develop a multi-objective genetic flatworm algorithm with a Tabu mapping mechanism (GFA–Tabu). The specific

calculation process of GFA–Tabu is shown in Algorithm 1, and the following section will deeply elaborate on its working logic in solving the PC-PLO-MC problem.

Algorithm 1 GFA-Tabu for the PC-PLO-MC

Input: Information of PC production order, molds and GFA-Tabu parameters.
Output: PC production sequence and layout schemes (*PS & LS*) for PC-PLO-MC.
Begin:
Import PC order demand and mold information. **// Model and parameter initialization**
GFA-Tabu parameters initialization, including: population size n, coefficients p, empty Tabu list, etc.
Generate an initial population with n individuals (*Pop*) by PC production sequence representation.
Divide *Pop* into two subgroups randomly, named *Pop1* with *np* individuals, and *Pop2* with (*n- np*) individuals.
Determine the positioning strategy of PCs by all individuals in the *Pop* and update the Tabu list.
While (termination criteria is satisfied) **do** **// Main loop**
For individuals (marked as I_i) in *Pop1* **// Genetic manipulations**
Crossover and mutation;
Check result by performing Tabu mapping mechanism; **// Tabu mapping mechanism**
If offspring I_i* performs better than I_j **and** the result is not tabued, **then** I_i* ← I_i, and update tabu list;
Else I_i ← select one with relatively better performance in I_i*, and update tabu list;
Determine the positioning strategy of PCs by all individuals in the *Pop1* and update the Tabu list.
EndFor
Update *Pop1*. **// Selection**
For individuals (marked as I_j) in P_2 **// Flatworm manipulations**
Growth, splitting, and regeneration;
Check result by performing Tabu mapping mechanism; **// Tabu mapping mechanism**
· · · · · · **// Operation is similar to that in *Pop1***
EndFor
Update *Pop2*.
Endwhile
PS & LS ← *PF*(*Pop1*∪*Pop2*). **// Pareto front (*PF*)**
End

4.1. PC Production Sequence Representation

To facilitate the application of GFA–Tabu in PC-PLO-MC, our first task is to determine the production sequence of the PCs before placing them on the mold tables. This process is called encoding in the GFA–Tabu, which is the foundation of the entire optimization process. A reasonable PC production sequence encoding must fulfill three core criteria: firstly, the encoding must be able to cover the required demand of all PC types in the production order. Secondly, the encoding needs to satisfy the limitation of mold availability on the production line. For example, for the 1# PC in Table 1, the demand is 4 pieces while the corresponding molds are only two pieces, so the two pieces of 1# PC molds have to be cycled twice to satisfy the order. For PC 4#, the situation is relatively complicated. In the second mold cycle layout, only two out of three pieces of molds are involved in production, and the other one piece is idle. Finally, a rational encoding individual in GFA–Tabu must be able to efficiently and stably perform genetic operations, as well as be easily recorded and released under the Tabu mapping mechanism. It should be noted that the strategy of keeping the PC mold layout as consistent as possible between two adjacent PC mold layouts is adopted when generating encoded individuals, which ensures that there will be the same or similar mold arrangement in different layout rounds, being more conducive to the workers' operation.

Based on the above criteria, we have constructed a feasible PC production sequence in Table 1, as shown in Figure 3. The numbers in the encoding circles represent the PC types, with a total of 14 PC elements. The arrows from left to right in the encoding scheme in Figure 3 indicate the sequence in which PCs are produced, with the PCs at the end of the arrow taking precedence over those at the beginning.

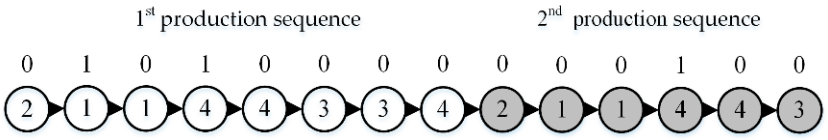


Figure 3. Three feasible encoding solutions.

According to the second criterion, we can easily determine that completing the order requires all PC molds to undergo two cycles of placement, with the only difference being that 3# and 4# PC molds are only partially involved in the second round. Therefore, in the encoding, the segment without a background color represents the first round of mold layout, while those with a dark background indicate the second round. The numbers 0 and 1 above the circle indicate the direction in which the PC molds are located relative to the mold table. Next, the population in the GFA–Tabu algorithm can be created based on the above encoding rules.

4.2. PC Mold Positioning Strategy

After determining the production sequence of the PCs, for a specific PC, the way and position of its layout need to be precisely determined when it is placed on the mold table. There are various ways to arrange the molds on the mold tables, and an improper layout has a significant impact on production energy consumption and other goals. As mentioned earlier, the arrangement of molds on the mold tables can be regarded as a two-dimensional bin packing problem. For medium and large computational problems, the most commonly used positioning methods are the bottom-left positioning algorithm [59], the minimum horizontal line method [60], and the rectangular interval generation and selection (*IGS*) method [61,62], while the bottom-left and minimum horizontal line methods are excellent for the bin packing problems with a small geometric size. Considering the special characteristics of the PC-PLO-MC problem, that is, each PC has a larger rectangular area and each mold table can generally only hold four to six PCs, the *IGS* method has greater advantages in dealing with the layout of large-sized PCs. Therefore, the *IGS* method is adopted in this paper, and in order to make the *IGS* have better adaptability to the PC-PLO-MC problem, we design a PC localization selection strategy called weighted matching filling (*WMF*) and embed it in the *IGS* localization method. The process of determining the positioning of the PC molds on the mold tables is detailed in Algorithm 2.

Algorithm 2 Determine the exact positions of molds on the mold tables

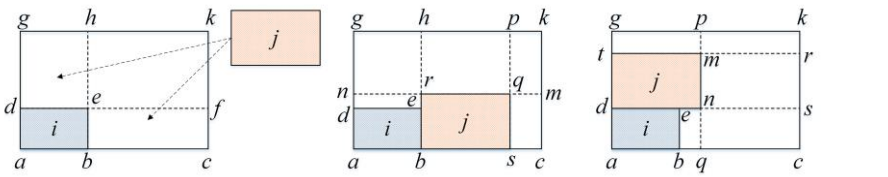
Input: PC production sequence and a certain number of available mold tables.
Output: Production layout plan for PC molds.
Begin
Select the first element in the PC production sequence for placement, and marked the first element index as $i = 1$;
Initialize the empty mold tables (without any molds on them), and marked the first empty mold table index $k = 1$;
While Not all molds in the PC production sequence have been placed on the mold tables
    If the mold table is empty, **then** the current mold i is placed in the lower left corner of the current mold table k;
    Else // At least one set of PC molds has been placed on the mold table
        Identify a set of rectangular intervals (mark as *RI*) on the current mold table; // *RSG* method

Algorithm 2 *Cont.*

For each rectangular interval in *RI*
Calculate the weighted matching value for each rectangular interval; // *WMF* method
EndFor
If there are multiple rectangular intervals with the same maximum weighted matching value
Select one (mark as *ri*) of the rectangular intervals randomly;
Else
Select the rectangular interval (mark as *ri*) with the maximum *WMF* value;
EndIf
If current PC mold *i* can be placed in the *ri* on the current mold table *k*
Record the placement sequence of PC mold *i* as *PS_tabu;*
Record the current position coordinates of the PC mold *i* on mold table *k*;
i = *i* + 1; // Jump to the *(i + 1) th* PC mold to be continued
Else // current PC mold *i* cannot be placed in the *ri* on the current mold table *k*
Save the PCs that have been placed on the current mold table *k*;
Record layout scheme on mold table *k* as *LS_tabu* and *PS_tabu* in the Tabu list;
k = *k* + 1; // Skip to the *(k + 1) th* mold table to continue placing the rest PC molds
EndIf
EndWhile
Output the determined layout theme.
End

4.2.1. Rectangular Set Generation (RSG) Method

The generation of a rectangular set involves two processes: partitioning and merging rectangles. As shown in Figure 4a, for the mold table 'ackg', the set of rectangular intervals can be represented as *RSG* = {'*ackg*'}. After the PC mold is placed, the remaining space on the mold table is divided into five rectangular areas—'*dehg*', '*dfkg*', '*bcfe*', '*bckh*', and '*efkh*'—and the corresponding set of rectangular intervals can be denoted as *RSG* = {'*dehg*', '*dfkg*', '*bcfe*', '*bckh*', '*efkh*'}. Then, the rectangular spaces that contain each other are deleted from the set of rectangular intervals, and the rectangles that cross and overlap each other are retained. Both '*dehg*' and '*efkh*' in the above set of rectangles are contained by the rectangle '*dfkg*', so they are deleted. Similarly, the rectangle '*bcfe*' is deleted. Therefore, the set of rectangular intervals becomes *RSG* = {'*dfkg*', '*bckh*'}. For the next PC mold *j* waiting to be placed, it is necessary to select a rectangle from '*dfkg*' and '*bckh*' in the set of rectangular intervals for placement. After selecting two different scenarios for placement, there are differences in the newly obtained set of rectangular intervals. As shown in Figure 4b,c, when the PC mold *j* is placed in the rectangular interval '*dfkg*', according to the generation process of the rectangular interval mentioned above, then *RSG* becomes {'*dehg*', '*nmkg*', '*sckp*'}. Similarly, when *j* is placed in the rectangular interval '*bckh*', *RSG* becomes {'*trkg*', '*qckp*', '*bcse*'}. It should be noted that the spacing between PC molds is not considered here to simplify the *RSG* method elaboration.



(a) Before j is placed. (b) Scheme 1 after j is placed. (c) Scheme 2 after j is placed.

Figure 4. Process of rectangular interval generation and selection.

4.2.2. Weighted Matching Filling (WMF) Method

Section 4.3.1 explained that selecting different rectangular intervals has a significant impact on the subsequent placement of PC molds. However, the general selection process is as follows: firstly, the rectangular interval selected from the set of rectangular intervals must satisfy the size requirements of the PC molds, that is, the size of the rectangular interval must be greater than or equal to the size of the PC molds to be placed. Secondly, the selection principle of rectangular intervals is that there may exist more than one that meets the size requirements. Choosing a rectangular interval to place PC molds is the key problem to be solved. Traditional random selection is a fast method, but not necessarily the most beneficial for saving energy and other objectives. This paper proposes a rectangular interval selection method based on weighted matching filling, as shown in Equation (14).

$$Max\ WMF = \lambda_0 \cdot \left(\lambda_1 \cdot \frac{l_i}{L_t} + \lambda_2 \cdot \frac{w_i}{W_t} + \lambda_3 \cdot \frac{l_i \cdot w_i}{L_t \cdot W_t} \right) \quad \forall i \in I, t \in RI \tag{14}$$

Equation (14) represents the selection of a rectangular interval, which is jointly determined by the weighted matching function generated in terms of the length, width, and area of the rectangular interval and the PC mold. λ_0 indicates whether the rectangular interval can accommodate the current PC mold; a value equal to 1 means yes, and 0 means no. λ_1, λ_2, and λ_3 are three weight factors for the length ratio, width ratio, and area ratio of the two, with values of 0.25, 0.25, and 0.5, based on a comprehensive consideration of the following two aspects: firstly, through regression analysis of historical production data and layout schemes, it was found that the area utilization rate has the most significant impact on the overall layout compactness, so it is given a higher weight of 0.5. Secondly, the dimensions of the length and width have a relatively balanced constraint effect on the utilization of the mold table, so they are each given a weight of 0.25 to reflect their equal importance in avoiding local congestion and to ensure the process spacing between components. The higher the value of *WMF*, indicating that at least in one of the three dimensions, the rectangular interval is highly matched with the PC mold and is therefore prioritized for selection. The theoretical maximum value of *WMF* is 1, corresponding to the scenario where the rectangular interval is perfectly aligned with the size of the PC mold.

4.3. Genetic Manipulations

Genetic operations are performed on all encoded individuals in Pop1. This also reflects one of the advantages of the GFA–Tabu, as the existence of subpopulations eliminates the need for additional definitions of difficult-to-determine crossover and mutation probabilities. Regarding the PC-PLO-MC problem, each operation was elaborated as follows.

4.3.1. Crossover

Due to the distinctiveness of the PC-PLO-MC problem, the encoding individuals generally consist of two or more sub-segments, and starting from the second and subsequent sub-segments, their PC production sequence is closely related to the first segment. Therefore, we only perform a two-point mapping crossover of the PC production sequence of the first round, but the two-point mapping crossover may result in infeasible encoding individuals, which requires a repair operation. Then, the sequence rearrangement is performed for segment 2 and subsequent sub-segments, similar to the encoding process. A crossover scheme for the two encoded individuals is given in Figure 5 with the following steps.

Step 1: Randomly select two different PC production sequences in Pop1 as parent individuals, denoted as P1 and P2.

Step 2: Generate two random integers as crossover points between the values of 1 and the length of the first sub-segment. The crossover points in Figure 5 are PCs 3# and 6# from left to right.

Step 3: P1 is used as the crossover individual, and P2 is used as the reference individual. Swap the PC subsequence {2,3,3,4} between the crossover points of P1 with {1,4,4,3} in P2, as well as swap the layout directions corresponding to each other, and keep the others unchanged. An offspring individual of P1 is obtained, denoted as O1.

Step 4: Based on the PC production order, check if the encoded offspring is feasible. If feasible, proceed to step 6; otherwise, repair operations are required. In the offspring O1, we found that the first sub-segment of P1 is {1,1,2,3,4,4,4}, while O1 is {1,1,1,4,4,3,4,4}. It is evident that all PCs of type 2 have been lost, one PC of type 3 is missing, and one more PC of types 1 and 4 has been added compared to the original. Therefore, O1 needs to be repaired.

Step 5: Repair. Specifically, after the crossover, the sub-segment of offspring 1 that has been crossed is extracted from the complement set in the original parent 2, reordered according to the order of its appearance in parent 1, and then repaired and replaced. The sub-segment of O1 after crossover is {1,4,4,3}, and its complement set is {2,1,3,4}, which is reordered as {1,2,3,4}, and O1 after repair is {1,2,1,4,4,3,3,4}.

Step 6: Rearrange the second round and subsequent sub-segments in offspring 1 by referencing the crossed sub-segment. The second sub-segment {1,1,2,3,4,4,} in O1 is rearranged to {1,2,1,4,3}.

Step 7: Similarly, swap the roles of the two parent individuals and repeat step 3 to get the offspring O2 of P2.

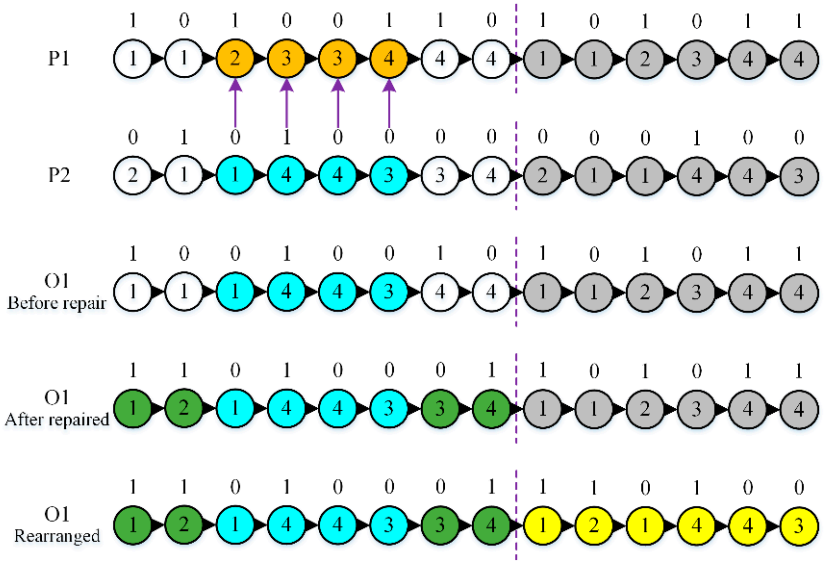


Figure 5. The crossover manipulation.

4.3.2. Mutation

Individuals in the Pop1 subpopulation will perform mutations, and the mutation process of P1 is depicted in Figure 6. Firstly, in the first round of the PC sequence, randomly select a mutation point, assuming that the first 3# PC from left to right is selected. Then, the PC number at the mutation point can be randomly swapped with the PC number at the front or back position. Finally, similar to the crossover process, it is necessary to adjust the PC sequence of the second and subsequent rounds according to the sequence of the mutated PC in the first round to create a new mutated individual.

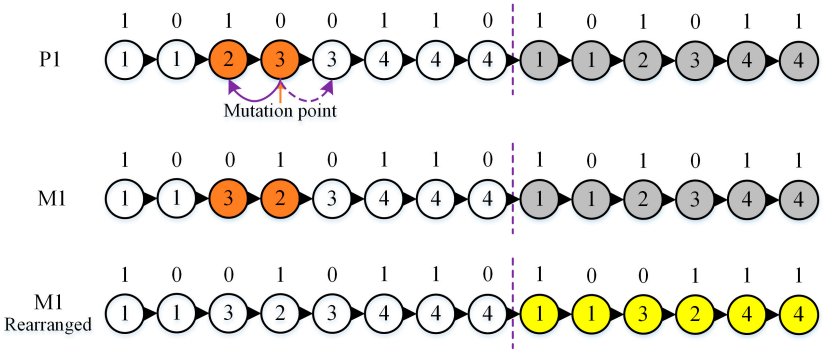


Figure 6. The mutation manipulation.

4.3.3. Selection

Crossover and mutation applied to Pop1 produced new individuals after genetic manipulation. These new individuals merged with the initial population to create a larger population that temporarily expanded to three times the size of the initial population. In order to maintain the appropriate population size and promote the continuation of the evolutionary process, a selection mechanism based on the Pareto dominance relation ranking and crowding distance of NSGAII is employed. Firstly, the individuals in the merged population are hierarchically sorted using the Pareto dominance relationship to identify the non-dominated solution set (also known as the Pareto front), which has reached a relatively optimal state on multiple optimization objectives, and no other solution exists that outperforms them on all objectives. Next, in order to select an equal number of individuals from the non-dominated solution set as the initial population for the next iteration, the crowding distance is used to evaluate the local crowding level of other solutions surrounding each non-dominated solution, and individuals with lower density will be retained. These selected individuals not only represent excellent solutions in the current iteration but also maintain sufficient diversity. Afterwards, crossover, mutation, and selection over hundreds or thousands of iterations drive the exploration towards a better solution space.

4.4. Flatworm Manipulations

In contrast to Section 4.3, flatworm manipulations consist primarily of the processes of growth, splitting, and regeneration of independent individuals. The working principles of each process can be found in Holstein [63], Tseng [57], and Liang [58] and will not be detailed here.

4.4.1. Growth

The growth operation is still only carried out for the first round of the production sequence of encoded individuals, as shown in P1 of Figure 7. Assuming that the number of PCs participating in growth does not exceed ten percent of the total number of PCs involved in the first round of production sequence and is at least one. The first production sequence in Figure 7 has a total of 8 PCs, so only one PC can participate in growth. Assuming that the second 3# PC from left to right undergoes replication growth, since there is no production priority relationship between the 8 PCs, then there is a total of seven theoretically available for insertion of the 3# PC after growth. Here, it is assumed that the replicated 3# PC is inserted before the second 1# PC from left to right. At this point, the growth process of P1 comes to an end. It should be noted that the layout direction of self-replicating PCs remains consistent with before. After the growth process, the first production sequence of the encoded individual will always have ten percent more PCs than the original, so the new encoded individual will no longer be a feasible production sequence, as it does not

meet the order requirements. Also, because of this, the second and subsequent rounds of production sequences will not undergo any repair operations temporarily after this process.

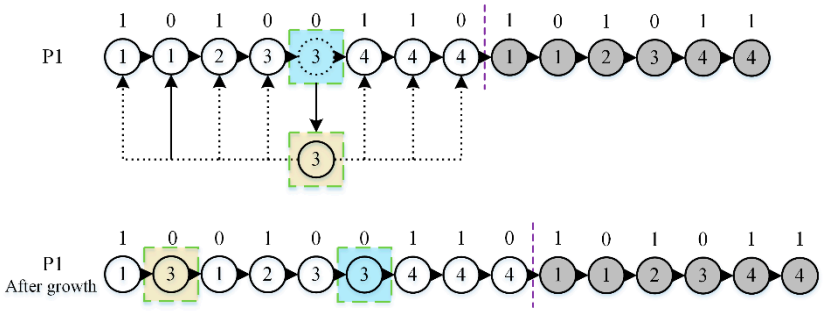


Figure 7. The flatworm growth process.

4.4.2. Splitting

The flatworm immediately undergoes the splitting process after growth. The splitting points are all in the vicinity of that position where self-replication of the PC occurs, and the sub-segments after the break can inherit and bond with the second and subsequent production sequences, but the individual after bonding no longer fulfills the encoding requirements. Figure 8 shows the splitting process after the growth of the individual in Figure 7. The splitting point occurs on the left side of the self-replicating PC. After splitting, the first round of the production sequence is cut into two parts; the left side is called the head, and the right is called the body. There is a special situation that needs attention: if the breakpoint in Figure 8 is on the right side, after the splitting, it will be found that there are three 3# PCs on the left head, and its corresponding molds only have two sets, which no longer meet the mold number constraint, this time we need to delete the redundant PC elements randomly in the regeneration.

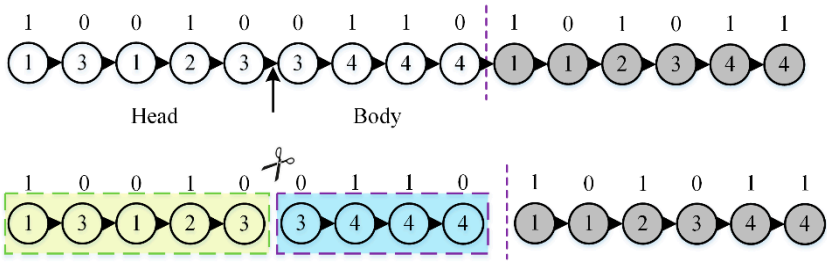


Figure 8. Splitting process after growth.

4.4.3. Regeneration

The fractured individual immediately starts the regeneration process, and the main implementation steps are as follows.

Step 1: Confirm whether the PC element redundancy scenario described above occurs. If so, the redundant PC elements will be randomly deleted until the mold quantity constraint is satisfied.

Step 2: For each sub-segment, identify all the missing PC information based on the order demand and the number of molds. Then, complete the missing PC information to get a new first-round production sequence, which is called the regeneration segment.

Step 3: The second and subsequent rounds of the production sequence are patched, and the process is the same as before and will not be described in detail.

The regeneration process is usually considered to occur instantaneously and is not directly affected by the length of the encoded individual. After regeneration and repair, the head and body segments in Figure 8 are shown in Figure 9.

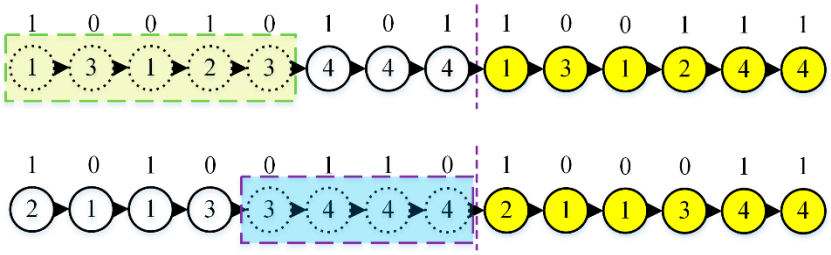


Figure 9. Regeneration process after splitting.

4.5. Tabu Mapping Mechanism

After the previous genetic and flatworm operations, an extremely rich production sequence scheme has been generated, but it is also prone to causing duplicate search problems in the production sequence schemes. Therefore, the Tabu mechanism derived from the Tabu search algorithm is incorporated into GFA–Tabu, where the Tabu search algorithm has already achieved great success in the fields of combinatorial optimization and production scheduling [64]. The so-called Tabu is to prohibit the repetition of the previous operation, which can improve search efficiency to a large extent. The traditional Tabu search generally sets the encoded neighborhood transformation structure as the Tabu object, such as the classical vehicle routing problem [65].

However, for the PC-PLO-MC problem, the storage structure of Tabu objects becomes quite complex and redundant when the number of PC orders demanded reaches hundreds or more. There are two very important operations in the PC-PLO-MC problem: one is the PC production sequence in the encoding, and the other is the PC layout scheme, significantly affected by the former. As shown in Figure 2, from the perspective of energy conservation, a higher density of PC layout on the mold table implies less energy consumption. As mentioned in Section 4.2, each mold table can usually only accommodate from four to six PCs. If the combination of PC layout on the mold table is used as the Tabu object, and then the production sequence can be easily deduced based on the order in which the PCs are placed, the complexity of the storage structure of the Tabu object can be greatly reduced. In other words, we record the Tabu object in the PC mold positioning process and map and check the state of this Tabu object during the PC production sequence generation process, which is called the Tabu mapping mechanism. Other parameters, such as Tabu length and aspiration criteria, are consistent with the Tabu search. The workflow of the Tabu mapping mechanism is shown in Algorithms 1 and 2.

5. Computational Experiments and Analysis

The GFA–Tabu algorithm was implemented in MATLAB 2021a and executed on a Windows 11 system equipped with a 3.10 GHz Intel® Core™ i5-10500 processor and 8 GB RAM. To comprehensively assess the performance of the GFA–Tabu algorithm in solving the PSLO-PC-MRC problem, three PC production cases of different sizes are carefully designed for validation.

It is worth noting that, despite researchers having accumulated considerable knowledge in PC production planning and management, detailed size information about PCs and their supporting molds is rarely mentioned in the existing literature. In view of this, we adopted a multi-source strategy to construct the case studies: for the small- and medium-scale test problems, we referred to the 'Atlas of Truss-Reinforced Concrete Stacked Slabs,' authoritatively released in China, which ensured the professionalism of the data and compliance with the industry standards. For the convenience of case design, the types and related size data of PCs remain unchanged, with problem scale, both order demand and mold quantity per PC type increase proportionally, but the mold resources do not exceed

the PC order demand. Each type of PC size consists of two parts: the length and width of the solid part for the PC cast by concrete, and the length of the steel bars extending from the solid part in four directions.

Related literature shows NSGAII is most frequently adopted in multi-objective PC production scheduling and management [66]. PSLO-PC-MRC is a multi-objective optimization problem; therefore, we compare the performance of the proposed GFA–Tabu with GFA, NSGAII, and the classical Tabu search (TS) algorithm. Meanwhile, to assess the performance of the proposed weighted matching filling in PC mold positioning, GFA–Tabu, GFA, NSGAII, and TS with *WMF* strategy were designed and compared with those without *WMF* strategy, respectively.

These algorithms adopt disparate parameter sets and termination rules. GFA, GFA–Tabu, and NSGAII involve the common computational parameter of population size, whereas NSGAII additionally involves the crossover probability and mutation probability. GFA, GFA–Tabu, and TS also include Tabu length and aspiration criteria. Except for the subpopulation partitioning ratio n in GFA and GFA–Tabu, all other parameters are determined using analysis of variance based on equivalent scale cases from relevant literature [58,67]. The available values of p are 0.9, 0.7, and 0.5, respectively. At a significance level of 0.05, the hypervolume [68] is used as the evaluation metric to perform analysis of variance on cases of all sizes. The results indicate that hypervolume has a high significance when p is 0.7. All algorithms undergo five independent runs to ensure testing fairness and are forced to terminate when a single run exceeds time $T = |I| \cdot \sum_{i=1}^{I} d_i \cdot Q_i / 10$ (in seconds).

5.1. Small-Scale Case

The small-scale case is detailed on the left side of Table 2. There are 20 types with a total of 60 PCs, and the number of molds is 28. The algorithm operates with a population size of 60, a crossover rate of 0.7, and a mutation rate of 0.3. The Tabu length is set to half the number of PC molds. The production results of PCs on the mold tables are detailed in Table 3.

Table 2. Information on small-scale and medium-scale cases.

Precast Component	Small-Scale Case		Medium-Scale Case 1		Medium-Scale Case 2		Geometric Size (mm)					
	Mold Quantity	Order Demand	Mold Quantity	Order Demand	Mold Quantity	Order Demand	Length	Width	L_{left}	L_{right}	L_{top}	L_{bottom}
1#	1	3	2	5	3	7	2220	1020	150	150	90	90
2#	2	4	2	7	3	10	2340	1320	90	90	90	90
3#	1	3	2	4	3	5	2580	660	150	90	150	150
4#	2	4	3	5	4	8	2520	1500	150	150	150	150
5#	2	5	3	7	4	10	2940	1200	90	90	0	0
6#	2	4	3	5	4	7	3120	1700	0	0	150	150
7#	1	2	2	6	3	7	3240	1260	90	90	0	0
8#	2	3	3	4	4	8	3120	1260	150	150	150	150
9#	1	4	2	6	3	7	3420	1200	150	150	0	0
10#	1	2	2	3	3	5	3420	1320	150	150	90	90
11#	1	2	2	3	3	5	3840	900	90	90	150	150
12#	1	4	3	7	5	13	4020	1320	0	0	90	90
13#	2	3	3	8	3	9	4020	1260	150	150	0	0
14#	1	3	2	5	3	10	4140	1560	90	90	0	0
15#	1	2	2	5	3	7	4620	1050	0	0	0	150
16#	1	2	2	3	3	5	4440	1860	90	90	150	150
17#	2	3	3	5	4	7	4620	1460	150	150	150	150
18#	1	2	2	4	3	6	4920	1980	150	150	90	90
19#	2	3	3	5	4	6	5520	660	0	0	150	150
20#	1	2	2	3	3	5	5520	2160	150	150	0	0

Table 3. PCs production results on the mold tables for small cases.

Method	No.	PCs Production Results on the Mold Tables	f_1	f_2	f_3
GFA–Tabu with WMF	1	{6,6,5,8,12},{5,4,4,8,17},{20,10,19,1},{17,15,19,7,3},{13,13,14,16},{9,2,2,11,18}, {6,6,5,8,12},{5,4,4,17,10},{20,19,1,7,3},{15,13,14,16},{9,2,2,11,18},{5,12,1,14,9,3},{12,9}	691.56	12.4	236
	2	{8,9,3,6},{6,13,12},{16,14,4},{4,7,20},{19,17,5,5},{19,17,2,2},{11,13,10,15},{18,1,8}, {8,9,3,6},{6,13,12},{16,14,4},{4,7,20},{19,17,5,5},{2,2,11,10},{15,18,1},{9,3,12},{14,5,1},{9,12}	721.56	11.23	284
	3	{6,19,5,4},{4,3,15,11,5},{20,2,9},{8,1,18},{14,19,10,2},{13,7,6,13},{8,16,12},{17,17}, {6,19,5,4},{4,3,15,11,5},{20,2,9},{8,1,18},{14,13,10},{2,7,6,12},{16,17},{5,3,9,1},{14,12},{9,12}	721.56	10.95	294
	4	{3,11,2,2,13,19,1},{4,4,13,6,9},{6,20,15,5},{17,16,19,5},{8,8,14,17},{10,18,7,12}, {3,11,2,2,13,19,1},{4,4,6,9,8},{6,20,15,5},{17,16,5,10},{14,18,7,12},{3,9,14,5,1,12},{9,12}	691.56	13.33	230
	5	{3,11,2,2,13,19,1},{4,4,6,9,13},{6,20,15,5},{17,17,8,8},{16,14,19,5},{10,18,7,12}, {3,11,2,2,13,19,1},{4,4,6,9,8},{6,20,15,5},{17,16,5,10},{14,18,7,12},{3,9,14,5,1,12},{9,12}	691.56	13.35	228
GFA–Tabu	1	{6,6,5,8,12},{5,4,4,8,17},{20,10,19,1},{17,15,19,7,3},{13,13,14,16},{9,2,2,11,18},{6,6,5,8,12}, {5,4,4,17,10},{20,19,1,7,3},{15,13,14,16},{9,2,2,11,18},{5,12,1,14,9,3},{12,9}	691.56	12.4	236
	2	{11,13,20,1,3},{5,5,17,7,9},{17,13,19,8},{18,19,8,4},{15,16,10,2,2},{4,6,6,14,12},{11,13,20,1,3}, {5,5,17,7,9},{18,19,8,4},{15,16,10,2,2},{4,6,6,14,12},{1,5,3,14,12,9},{12,9}	691.56	15.96	228
	3	{4,4,19,9,5,3},{5,8,6,6,1,2},{11,20,2,10},{18,13,19,8},{15,13,14,17},{17,16,7,12},{4,4,19,9,5,3}, {5,8,6,6,1,2},{11,20,2,10},{18,13,15,14},{17,16,7,12},{9,5,1,3,14,12},{9,12}	691.56	13.55	232
GFA with WMF	1	{7,6,8,8,1,2},{19,20,6,5},{18,14,19,5},{4,4,10,13,13},{16,3,15,9,11},{12,17,17,2},{7,6,8,19}, {20,6,1},{18,14,5},{5,4,4,10,13},{16,3,15,9,11},{12,17,2,2},{14,1,5,3},{12,9},{12,9}	703.56	19.64	252
	2	{5,5,10,4,4,3},{16,9,17},{17,13,13,19},{8,8,15,12},{11,20,2,2},{7,6,14,1},{6,18,19}, {5,5,10,4,4,3},{16,9,17},{13,8,15,12},{11,20,2,2},{7,6,14,1},{6,18,19},{5,3,9,1},{12,14},{9,12}	709.56	16.33	256
	3	{7,19,5,4},{4,3,15,11,5},{20,2,9},{8,1,18},{14,19,10,2},{13,6,13,8},{6,16,12},{17,17},{7,19,5,4}, {4,3,15,11,5},{20,2,9},{8,1,18},{14,13,10},{2,6,6,12},{16,17},{5,3,9,1},{14,12},{9,12}	721.56	12.2	290
	4	{6,19,5,4},{4,3,15,11,5},{20,2,9},{8,1,18},{14,19,10,2},{13,7,6,13},{8,16,12},{17,17},{6,19,5,4}, {4,3,15,11,5},{20,2,9},{8,1,18},{14,13,10},{2,7,6,12},{16,17},{5,3,9,1},{14,12},{9,12}	721.56	10.95	294
GFA	1	{6,19,5,4},{4,3,15,11,5},{20,2,9},{8,1,18},{14,19,10,2},{13,7,6,13},{8,16,12},{17,17},{6,19,5,4}, {4,3,15,11,5},{20,2,9},{8,1,18},{14,13,10},{2,7,6,12},{16,17},{5,3,9,1},{14,12},{9,12}	697.56	14.64	244
	2	{19,9,3,11,19,2},{14,17,17,5},{5,2,16,7,6},{6,18,15,10},{13,13,4,4,8},{8,20,1,12},{19,9,3,11,2}, {14,17,5,5,2},{16,7,6},{6,18,15,10},{13,4,4,8},{20,1,12},{9,3,14,5,1,12},{9,12}	697.56	16.13	238
	3	{2,2,19,19,8,8},{6,6,15,4,4},{18,11,13,3},{20,13,9,10},{7,16,5,5,1},{17,17,14,12},{2,2,19,8,6}, {6,15,4,4},{18,11,13,3},{20,9,10},{7,16,5,5,1},{17,14,12},{9,5,3,1,14,12},{9,12}	697.56	18.4	236
NSGAII with WMF	1	{9,11,10,15,19},{13,13,18,1,3},{17,17,7,6},{20,19,6,5},{12,16,5,2,2},{14,8,8,4,4}, {9,11,10,15,19},{13,18,1,7,3},{17,6,5,12},{20,6,5},{16,14,8},{4,4,2,2},{9,1,3,5},{12,14},{9,12}	703.56	19.51	252
	2	{8,9,3,6},{6,13,12},{16,14,4},{4,7,20},{19,17,5,5},{19,17,2,2},{11,13,10,15},{18,1,8},{8,9,3,6}, {6,13,12},{16,14,4},{4,7,20},{19,17,5,5},{2,2,11,10},{15,18,1},{9,3,12},{14,5,1},{9,12}	721.56	11.23	284
	3	{11,9,10,15,19},{13,13,18,1,3},{17,17,7,6},{20,19,6,5},{12,16,2,5,2},{14,8,8,4,4}, {11,9,10,15,19},{13,18,1,7,3},{17,6,5,12},{20,6,2},{16,5,14},{8,4,4,2},{9,1,3,5},{12,14},{9,12}	703.56	19.42	256
NSGAII	1	{17,13,19,2},{15,11,2,7,8},{9,13,8,1,4},{17,5,5,19,3},{18,10,6},{16,6,14},{20,4,12},{17,13,19,2}, {15,11,2,7,8},{9,1,4,5,5,3},{18,10,6},{16,6,14},{20,4,12},{9,1,5,3,14,12},{9,12}	703.56	13.08	260
	2	{2,17,2,9,1},{18,8,6},{6,5,13,19,3},{13,8,17,5},{16,19,7,11},{20,15,12},{4,4,10,14},{2,17,2,9,1}, {18,8,6},{6,5,13,19,3},{5,16,7,15},{20,11,12},{4,4,10,14},{9,1,5,3,12,14},{9,12}	703.56	13.12	258
	3	{17,3,8,19},{5,5,4,15,13},{11,14,4,2,2},{18,13,12},{20,8,10},{6,6,7,1,9},{16,17,19},{17,3,8,19}, {5,5,4,15,13},{11,14,4,2,2},{18,12,10},{20,6,7},{6,16,1,9},{3,5,14,12,1,9},{12,9}	703.56	13.16	256
TS with WMF	1	{1,14,8,2,2},{20,10,5},{16,12,17},{17,5,19,6},{19,6,8,9},{4,4,15,7,11},{18,13,13,3},{1,14,8,2,2}, {20,10,5},{16,12,17},{5,19,6,6},{9,4,4,15,7},{18,11,13,3},{1,14,5,12,9,3},{12,9}	703.56	13.41	252
	2	{1,6,14,6,8},{17,16,2,2,10},{20,19,5,12},{19,17,5,15,3},{8,4,4,9,7},{13,13,18,11},{1,6,14,6,8}, {17,16,2,2,10},{20,19,5,12},{5,15,4,4,9,3},{7,13,18,11},{1,14,5,12,9,3},{12,9}	691.56	18.65	238
TS	1	{11,7,17,6},{6,2,2,16},{12,13,1,19,3},{17,19,9,8},{18,14,13},{8,5,5,4,4},{10,15,20},{11,7,17,6}, {6,2,2,16},{12,13,1,19,3},{18,14,9},{8,5,5,4,4},{10,15,20},{12,1,14},{9,3,5},{12,9}	709.56	13.01	260
	2	{15,4,1,19},{17,14,10},{2,7,20},{13,19,4,9},{16,6,8},{18,11,8},{17,5,5,3},{2,6,13,12}, {15,4,1,19},{17,14,10},{2,7,20},{13,4,9,8},{16,6,11},{18,5,5,3},{2,6,12},{1,14,9,5},{12,3},{9,12}	721.56	11.12	296

From Table 3, it can be intuitively seen that GFA–Tabu with WMF obtained five optimal Pareto solutions, while other algorithms obtained relatively fewer. The average energy consumption in all solutions is 704.81 kgce, the highest and lowest are 721.56 kgce and 691.56 kgce, respectively, and the lowest values are mainly concentrated in GFA–Tabu.

From the quality of optimal Pareto solutions, the first solution by GFA–Tabu, the fourth solution by GFA-WMF, and the second solution by NSGAII-WMF can all be found in the solution obtained by GFA–Tabu with WMF. If observing carefully, we will find that the solutions obtained by GFA–Tabu with WMF partially dominate the solutions obtained by other algorithms, such as the fifth solution (691.56,13.35,228) by GFA–Tabu with WMF dominating the second solution (691.56,15.96,228) and the third solution (691.56,13.55,232) of GFA–Tabu. The first and second solutions by GFA–Tabu with WMF dominate the third solution of NSGAII-WMF and the third solution of GFA-WMF. The two solutions by TS-WMF and TS are dominated by the solutions in GFA–Tabu with WMF. All of these demonstrate the effectiveness of the Tabu mechanism and the WMF method.

5.2. Medium-Scale Cases

The medium-sized cases are constructed based on the small-sized case, both of which have 20 PCs, and the data are detailed in Table 2. The first case involves 100 PCs and 48 molds, while the second comprises 147 PCs and 68 molds. For these cases, the population sizes are set to 80 and 100, respectively, with crossover and mutation probabilities fixed at 0.75 and 0.25. The Tabu length is set to half the number of molds of the respective cases. After running, the best five optimal Pareto solutions were obtained for the two cases, as shown in Tables 4 and 5.

Table 4. Results for the medium-sized case with 100 PCs.

Method	No.	GFA–Tabu			GFA			NSGAII			TS		
		f_1	f_2	f_3	f_1	f_2	f_3	f_1	f_2	f_3	f_1	f_2	f_3
WMF	1	1176.19	14.34	364	1164.19	16.47	358	1188.19	11.49	436	1164.19	15.99	362
	2	1164.19	15.71	366	1164.19	16.52	354	1188.19	11.51	434	1164.19	15.88	368
	3	1164.19	16.65	350	1170.19	15.52	356	1164.19	16.34	386	1176.19	13.46	388
	4	1158.19	18.91	344	1176.19	14.49	386	1164.19	16.54	380	1176.19	13.67	378
	5	1176.19	12.12	390	1176.19	13.72	398	1176.19	14.28	404	1176.19	12.77	390
No WMF	1	1188.19	10.67	446	1170.19	14.61	388	1170.19	15.67	374	1176.19	14.85	378
	2	1188.19	11.1	444	1176.19	15.39	382	1188.19	11.78	466	1164.19	16.82	404
	3	1188.19	12.29	426	1164.19	21.24	406	1164.19	17.01	390	1164.19	16.71	410
	4	1170.19	16.12	390	1170.19	15.87	378	1170.19	15.77	370	1176.19	14.6	388
	5	1188.19	12.27	428	1176.19	15.43	380	1188.19	13.92	438	1176.19	14.59	390

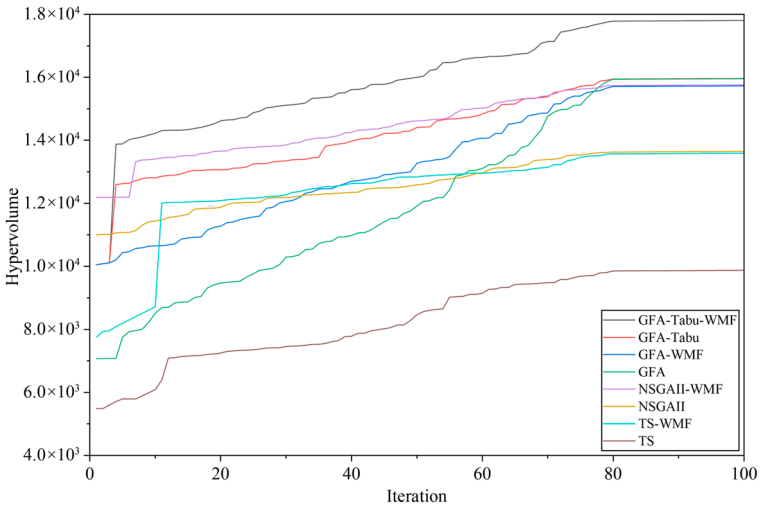
Table 5. Results for the medium-sized case with 147 PCs.

Method	No.	GFA–Tabu			GFA			NSGAII			TS		
		f_1	f_2	f_3	f_1	f_2	f_3	f_1	f_2	f_3	f_1	f_2	f_3
WMF	1	1740.13	18.13	472	1734.13	18.07	480	1746.13	10.53	498	1746.13	12.86	518
	2	1770.13	8.66	600	1734.13	18.36	476	1752.13	9.4	556	1746.13	12.77	526
	3	1734.13	18.07	480	1740.13	13.05	476	1746.13	11.29	490	1740.13	15.97	508
	4	1740.13	13.05	476	1740.13	13.02	480	1746.13	11.14	494	1740.13	13.64	530
	5	1740.13	10.12	500	1752.13	11.58	548	1740.13	11.98	502	1740.13	16.02	500
No WMF	1	1740.13	13.05	476	1740.13	10.12	500	1770.13	8.66	600	1752.13	11.54	540
	2	1734.13	18.51	474	1746.13	11.14	494	1746.13	13.06	550	1752.13	11.55	538
	3	1740.13	13.02	480	1746.13	11.29	490	1752.13	11.4	560	1746.13	15.06	510
	4	1752.13	11.58	548	1740.13	10.09	552	1764.13	11.04	610	1740.13	15.7	512
	5	1740.13	12.82	490	1752.13	9.91	596	1752.13	11.29	562	1746.13	16.83	504

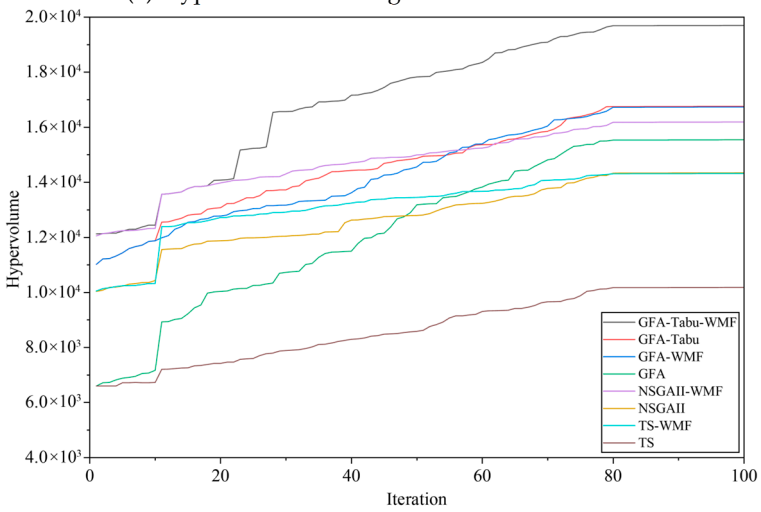
For the medium-sized case with 100 PCs, among all the production sequences and layout schemes obtained by the algorithms in Table 4, GFA–Tabu has the smallest energy consumption of 1158.19 kgce, while the other algorithms have the smallest value

of 1164.19 kgce. The upper half of Table 4 shows that each algorithm is embedded with the WMF method, while the lower half does not have the WMF method. We find the fact that the optimal Pareto solutions obtained by all four algorithms without the WMF method are dominated by algorithms with the WMF method. For example, the fifth solution (1176.19,12.12,390) by GFA–Tabu with WMF dominates the third solution (1188.19,12.29,426) and the fifth solution (1188.19,12.27,428) of GFA–Tabu. The fifth solution by GFA with WMF dominates the fifth solution of GFA, the first solution by NSGAII with WMF dominates the second and fifth solutions of NSGAII, and the first solution by TS with WMF dominates the second and third solutions of TS. This indicates that the proposed WMF method plays a better role in guiding the layout of PCs onto the mold tables. Combining all the solutions, we found that the solutions obtained by other algorithms are dominated by GFA–Tabu with WMF. As shown in Table 4, except for GFA without WMF, the fifth solution of all other algorithms is dominated by the fifth solution of GFA–Tabu with WMF, and there are also cases where the solutions in GFA without WMF are dominated by GFA–Tabu with WMF.

The hypervolume convergence curve in Figure 10 also confirms the above conclusion, and similar conclusions exist when the number of PCs in medium-sized cases grows to 147, as shown in Table 5, and will not be elaborated further. The results indicate that the proposed GFA–Tabu with WMF outperforms all compared algorithms, proving the effectiveness of the Tabu mechanism and WMF method.



(**a**) Hypervolume convergence curve of 100 PCs.



(**b**) Hypervolume convergence curve of 147 PCs.

Figure 10. Hypervolume convergence curve for medium-scale cases.

5.3. Analysis of Computational Complexity

The computational complexity of the proposed algorithm is primarily analyzed from two aspects. First, the complexity of the genetic operations is mainly constrained by the number of optimization objectives and the population size, expressed as O (objNum × n^2) [56], where objNum denotes the number of simultaneously optimized objectives. Second, the complexity of the flatworm operations—during the growth, splitting, and regeneration stages—is closely related to both the chromosome length and the population size, expressed as O (objNum × chromLength), where chromLength represents the chromosome encoding length. Therefore, overall, the computational complexity of the proposed GFA–Tabu algorithm is O (objNum × n^2) + O (objNum × chromLength) ≈ O (n^2), indicating that its computational complexity does not exceed that of the classical NSGA-II.

6. Large-Scale Application Case

6.1. Case Data

For the large-scale case, we directly obtained the production order data from a PC manufacturer in Chengdu, Sichuan Province, thus enhancing the practicality and relevance of the study. The order consists of 44 laminated floor slab variants, with an overall quantity of 272 units and 95 mold configurations. The length and width of the solid part of each type of PC, as well as the length of the steel bars extending in four directions, are described in Table 6.

Table 6. Information on large-scale cases.

Precast Component	Large-Scale		Geometric Size (mm)					
	Mold Quantity	Order Demand	Length	Width	L_{left}	L_{right}	L_{top}	L_{bottom}
1#	1	4	2260	795	150	90	150	150
2#	2	7	2260	2420	90	150	0	0
3#	4	8	2500	2205	0	0	150	90
4#	1	3	2320	2520	90	90	0	0
5#	3	11	2545	855	0	0	90	150
6#	2	8	2545	2180	0	0	90	150
7#	2	5	2545	2265	0	0	90	90
8#	1	4	2245	2340	150	150	90	90
9#	2	5	2665	1940	90	90	150	150
10#	2	4	2895	1720	0	0	90	150
11#	2	6	2655	2100	90	150	0	0
12#	2	6	2765	1820	90	90	0	0
13#	2	9	2645	1920	150	150	90	90
14#	3	8	3170	1800	0	0	150	150
15#	2	7	3645	1800	0	0	150	150
16#	2	4	3515	1800	90	90	150	150
17#	3	9	3565	1800	90	90	150	150
18#	2	6	3495	2240	150	150	0	0
19#	2	4	3665	1780	90	90	90	90
20#	3	10	3605	1860	90	150	150	90
21#	3	9	3630	1800	90	150	150	150
22#	2	6	3595	1860	150	150	150	90
23#	1	2	3680	1800	150	90	150	150
24#	2	5	3765	1515	90	90	90	90
25#	3	12	3645	1620	150	150	90	90
26#	1	4	3765	1800	90	90	150	150
27#	2	6	3790	1395	90	90	150	150
28#	1	3	3790	1500	90	90	150	150
29#	2	8	3730	1800	150	90	150	150

Table 6. *Cont.*

Precast Component	Large-Scale		Geometric Size (mm)					
	Mold Quantity	Order Demand	Length	Width	L_{left}	L_{right}	L_{top}	L_{bottom}
30#	2	3	3680	1900	150	150	90	90
31#	2	6	3680	1800	150	150	150	150
32#	2	7	3695	1800	150	150	150	150
33#	5	9	3840	2100	90	90	0	0
34#	5	10	3890	1860	90	90	150	90
35#	3	6	3915	1440	90	90	90	90
36#	1	3	3915	1920	90	90	90	90
37#	1	2	3965	1360	90	90	150	90
38#	2	5	3965	1920	90	90	90	90
39#	2	9	3895	1920	150	150	90	90
40#	1	3	3945	1860	150	150	150	90
41#	4	9	4305	1200	90	150	150	150
42#	1	3	4305	1280	90	150	90	150
43#	2	10	4305	1650	150	90	0	0
44#	2	4	4640	2100	90	90	0	0

6.2. Comparative Testing and Result Discussion

GFA, NSGAII, and TS are still selected for performance comparison with GFA–Tabu, and the presence or absence of WMF methods will be integrated into each of the four algorithms separately. The population sizes involved in all algorithms are all set to 160, the crossover and mutation probabilities involved are set to 0.8 and 0.2, respectively, the Tabu length is set to half of the number of PC types, which is set to 22, and the coefficient p for dividing subpopulations is set to 0.7.

After each algorithm is run independently five times, the optimal Pareto solutions obtained by all algorithms are merged to form a more comprehensive and diverse solution set. Subsequently, the dominant relationship between these merged solutions is reevaluated, and the newly obtained Pareto optimal solution is approximated as the true Pareto front of the large-scale problem. On this basis, the hypervolume indexes of the solutions obtained by each algorithm are calculated, and their distribution is shown in Figure 11, which can intuitively reflect the coverage and degree of superiority of each algorithm in the solution space. Overall, the obtained hypervolume mean values by all algorithms with WMF are connected by red dashed lines, and it is clear that the mean values show a decreasing trend. If the hypervolume mean values obtained by the algorithms with WMF from those without WMF are viewed separately, they are roughly two parallel dashed lines. Moreover, each algorithm with WMF has a higher hypervolume mean value than those without WMF, which indicates the effectiveness of the WMF method. Looking at the four algorithms on the left side of Figure 11 individually, their corresponding hypervolume means are 1.57×10^6, 1.38×10^6, 1.38×10^6, and 1.24×10^6, indicating that the proposed GFA–Tabu performs the best in convergence and stability, while the GFA and NSGAII have comparable performances, and TS has the worst performance.

The obtained optimal Pareto solutions by the algorithms with WMF from those without WMF are depicted in Figure 12, which exhibits significant differences in spatial distribution. In terms of energy consumption, GFA–Tabu with WMF successfully obtained optimal Pareto solutions for twelve different energy consumption levels, in contrast to the other algorithms, which only provide solutions for five to eight energy consumption levels. It is noteworthy that GFA–Tabu with WMF, together with the other two algorithms, reached the lowest energy consumption, which is 4039.17 kgce. Not only that, but GFA–Tabu with WMF can also find solutions in relatively high energy consumption ranges, such as the two energy

consumption points of 4099.17 kgce and 4105.17 kgce. Although these two solutions have relatively high energy consumption, the fluctuation coefficients of mold table utilization in the corresponding PC production sequence and layout scheme are relatively small, with specific values of 12.87 and 13.04, respectively.

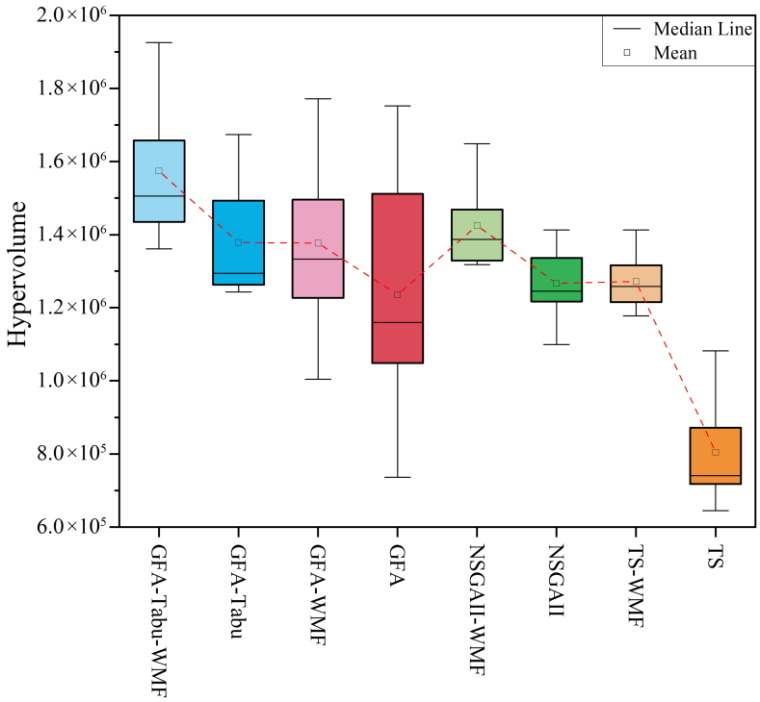


Figure 11. Hypervolume distributions of the tested algorithms.

6.3. Analysis of Mold Layout Schemes

To assist in the comparative analysis of PC mold layout alternatives for decision-makers, Figure 13 presents the layout scheme when each objective is the smallest, obtained by GFA–Tabu with WMF. The horizontal axis corresponds to the PC coverage area per mold table, bounded by 40 m^2 (10 m × 4 m), and the vertical axis reflects the total mold tables utilized. Since the number of mold tables consumed in this case is all over 90, the utilized mold tables are exhibited on the left and right sides in the figure. In Figure 12a, 89 mold tables are used, and the total energy consumption is 4039.17 kgce, of which the process energy consumption and transportation energy consumption are 3505.2 kgce and 534 kgce, respectively. The lowest utilization rate is 22.02% on the 89th mold table, the highest is 73.94% on the 36th mold table, the average utilization rate is 54.7%, and the mold table utilization fluctuation coefficient is 21.34. The corresponding switching time of mold tables and molds is 1298 min. In Figure 13b, 99 mold tables are used with a total energy consumption of 4099.17 kgce. The lowest utilization rate is 15.46% on the 99th mold table, and the highest is 59.52% on the 98th mold table, with an average utilization rate of 49.17% and a coefficient of fluctuation of the utilization rate of 12.87. The mold switching time is 1524 min. In Figure 13c, 92 mold tables are used, with a total energy consumption of 4057.17 kgce; the mold table utilization fluctuation coefficient is 26.08, and the mold switching time is 1268 min.

Although the energy consumption in Figure 13a is the lowest, the mold table utilization fluctuation coefficient is almost 40% higher than that in Figure 13b. Meanwhile, Figure 13b has the least fluctuating mold table utilization but sacrifices as much as 256 min of mold switching time compared to the scenario in Figure 13c. The result reveals that the key factor affecting total energy consumption is the number of mold tables consumed to fulfill PC order demand; that is, increasing the average utilization rate of mold tables to reduce

their usage is beneficial for reducing PC production energy consumption. However, if energy consumption is the sole consideration, we may observe large swings in mold table utilization, which translate into workload differences across tables and serious disruptions to line balance. Each option has its own advantages and disadvantages, and the PC production manager can choose the preferred production sequence and layout plan based on the actual situation of the workshop.

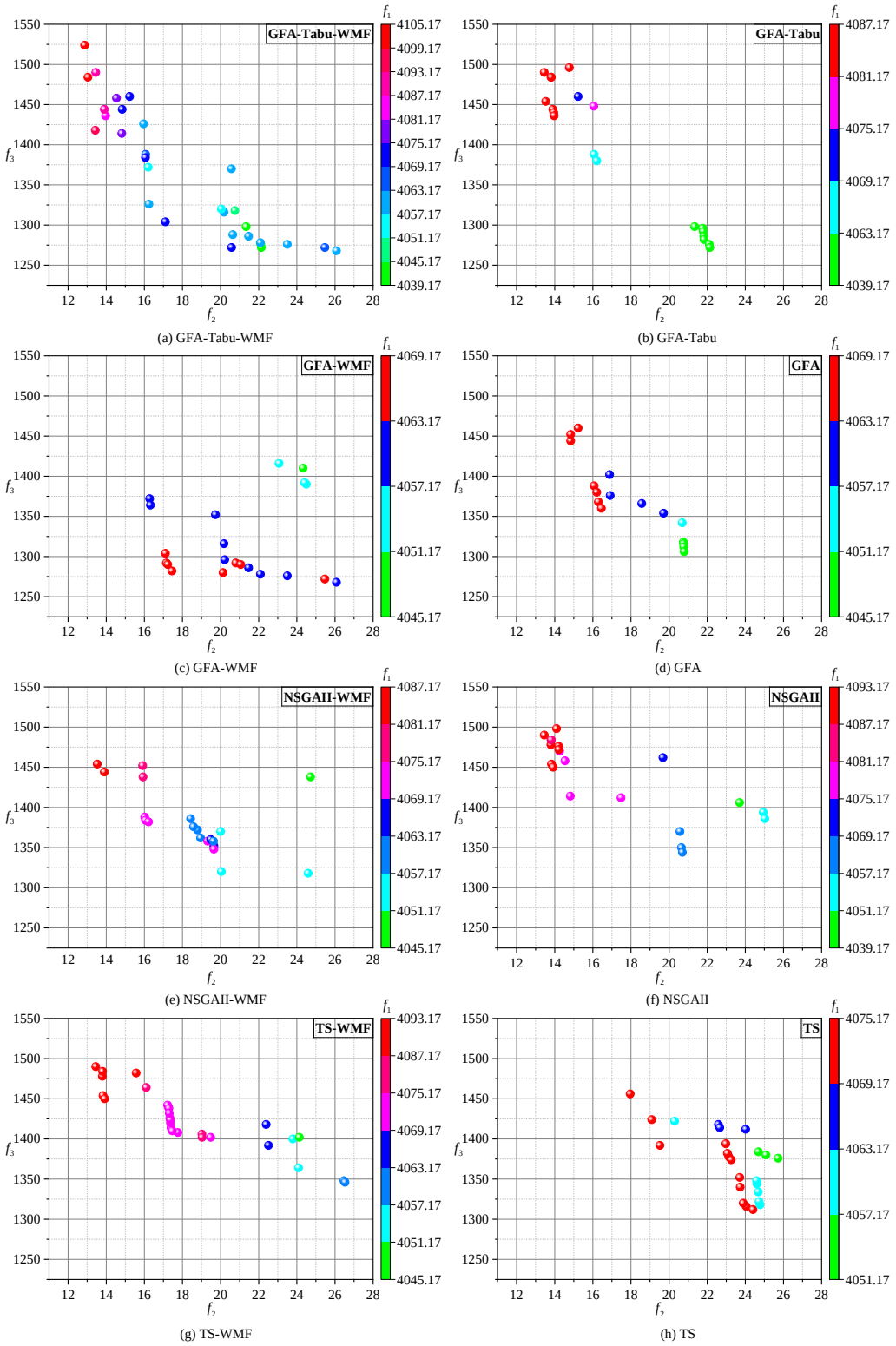
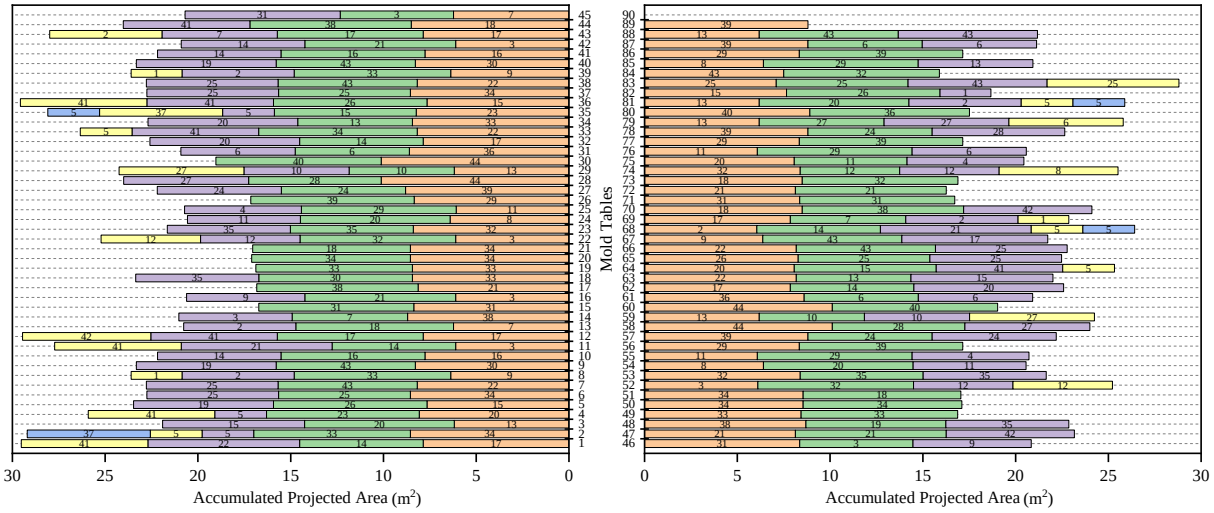
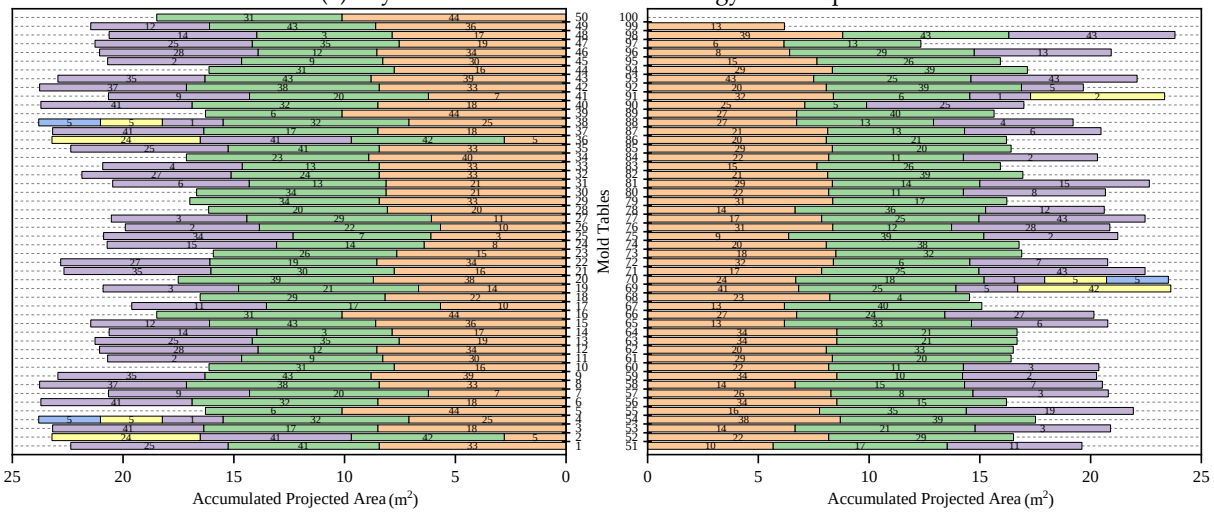


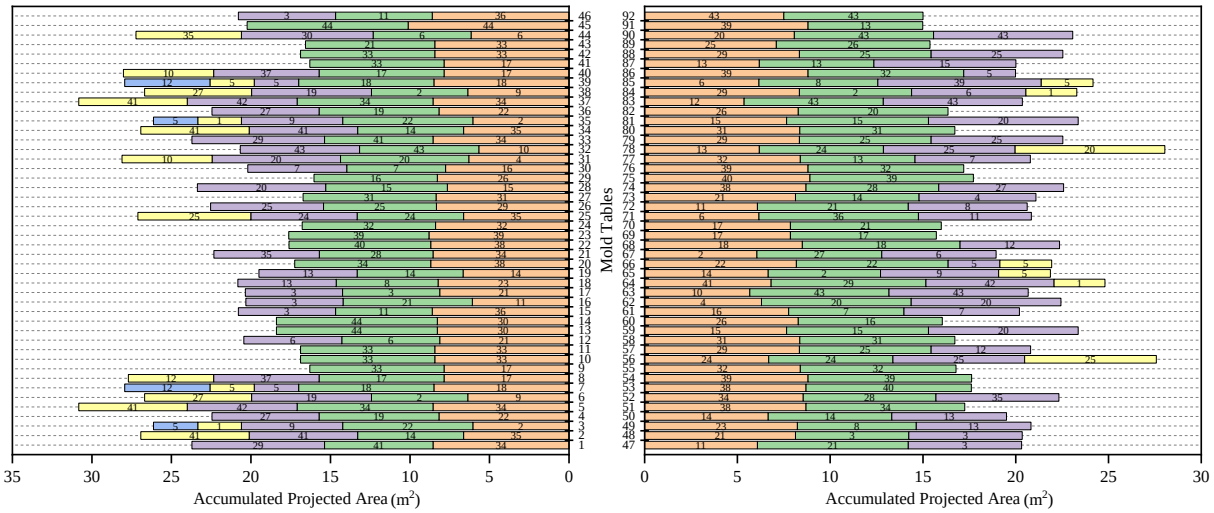
Figure 12. Distributions of optimal Pareto solutions obtained by each algorithm.



(**a**) Layout scheme for minimum energy consumption.



(**b**) Layout scheme for minimum fluctuation of mold table utilization rates.



(**c**) Layout scheme for minimum mold switching time.

Figure 13. Mold layout schemes by GFA–Tabu with WMF.

7. Conclusions

This paper proposes a model for optimizing the production sequencing and layout of precast concrete components, which considers how to efficiently and reasonably allocate PC to the mold tables in the actual situation where the number of molds is limited. Subse-

quently, a multi-objective genetic flatworm algorithm with a Tabu mapping mechanism is developed, and a rectangular interval selection method based on a weighted matching filling strategy is designed to carry out the positioning of PC molds accurately to minimize the total energy consumption, fluctuation coefficient of mold table utilization, and mold switching time. To assess the performance of the model and algorithm, three types of cases covering different scales are carefully constructed for testing. The results indicate that:

1. In the small-scale case, the proposed model achieved minimum values of 691.56 kgce for production line energy consumption, 10.95 for the mold table utilization balance coefficient, and 228 min for mold switching time. Compared to the control group algorithms, the proposed GFA–Tabu algorithm with the weighted matching filling method not only obtained an average of two additional Pareto-optimal solutions but also demonstrated superior solution quality, with average reductions of 0.98 kgce, 2.54 in balance coefficient, and 1.5 min in switching time across the three objectives, respectively.
2. For the two medium-scale cases involving 100 and 147 PCs, the proposed model achieved minimum values of the three objectives at 1158.19 kgce, 12.12, 344 min, and 1734.13 kgce, 8.66, and 472 min, respectively. Compared to the control group algorithms, it achieved average reductions of 575.94 kgce, 3.46, and 130 min across these objectives.
3. In the large-scale application case, the proposed model and algorithm exhibit a dual-core advantage. Its average hypervolume performance metric reaches 1.57×10^6, which is 10.48% higher than the second-best algorithm of NSGAII-WMF and represents improvements of 27.39% and 95.55% over the traditional GFA and TS algorithms, respectively. In terms of stability, its coefficient of variation for hypervolume is 11.27%, lower than that of GFA-WMF (16.44%) and GFA (25.79%), indicating strong robustness.

Therefore, the case study results consistently show that the proposed model and algorithm effectively address the production sequence and layout optimization problem for precast concrete components, providing decision-makers with efficient and practical solutions for PC production.

Author Contributions: Conceptualization, J.L. and Y.L.; methodology, J.L. and Y.L.; software, B.D.; validation, Y.L., X.S. and J.L.; formal analysis, J.L.; investigation, X.S. and Y.L.; resources, X.S.; data curation, X.S. and W.X.; writing—original draft preparation, J.L.; writing—review and editing, W.X. and Y.L.; visualization, B.D.; supervision, X.S. and Y.L.; project administration, B.D.; funding acquisition, B.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by a grant from Luzhou Key Laboratory of Intelligent Construction and Low-carbon Technology, the Key Laboratory of Higher Education of Sichuan Province for Enterprise Informationalization and Internet of Things (grant number 2024WYY03), the National Natural Science Foundation of China (grant number 52105513), and the Natural Science Foundation of Hubei Province (grant number 2023AFB593), and by Building Environment Engineering Technology Research Center in Dazhou (Sichuan University of Arts and Sciences, grant number SDJ2024ZB-11).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing is not applicable.

Acknowledgments: We thank all reference authors who gave us inspiration and help. The authors thank the editors and anonymous commentators for their valuable comments, which have improved the quality of this paper, and the experimental institutions that provide financial support for the paper.

Conflicts of Interest: The authors declare no conflicts of interest.

Notations

Indices	
i, j	Index of PC, $i, j \in I$
k	Index of mold table, $k \in K$
s	Index of the cycle that the PC mold is to be used, $s \in S$
t	Index of rectangular interval, $t \in RI$
Parameters	
I	Set of categories of precast components
K	Set of mold tables
S	The set of cycles that the PCs mold needs to be used
RI	A set of rectangular intervals on the mold table
L	The length of the mold table
W	The width of the mold table
P	The total required quantity of all precast components, $P = \sum_{i \in I} d_i$
l_i, l_j	The length of precast components i and j, $i, j \in I$
w_i, w_j	The width of precast components i and j, $i, j \in I$
h_i, h_j	The thickness of precast components i and j, $i, j \in I$
l_i^k, w_i^k, h_i^k	The length, width, and thickness of the precast component i on the kth mold table, $i \in I$, $k \in K$
r_i, r_j	Direction of placed precast concrete components i and j, $i, j \in I$
L_t	The length of the rectangular interval t, $t \in RI$
W_t	The width of the rectangular interval t, $t \in RI$
P_{is}	Number of molds of type I participated in the sth cycle layout, $P_{is} \geq 0$
Q_i	Available quantity of mold i, $Q_i \geq 0$
d_i	The order demand of PC i, $d_i > 0$
d	Production spacing, $d > 0$
z_{ip}	In a production sequence, if position p is component i, then $z_{ip} = 1$, otherwise $z_{ip} = 0$
Decision variables	
x_i, y_i	Positioning coordinates of the lower left corner of PC i on the mold table
η_k	The utilization efficiency of the kth mold table
y_k	1 if the kth mold table is utilized; 0 otherwise
z_p	In a production sequence, if positions p and p-1 are different components, then $z_p = 1$, otherwise $z_p = 0$
Indicator variables	
w_1	Unit energy consumption coefficient of PC in the hot curing process
w_2	Unit energy consumption coefficient of PC in the other five processes
w_3	Unit energy consumption coefficient of PC in transportation
w_4	Switching time between two mold tables
w_5	Mold adjustment time for two different PCs
α	The weight of reinforced concrete per cubic meter for PC
β	The weight of each mold table
λ_0	A coefficient indicating whether the rectangular interval can accommodate the current PC mold
λ_1	Weight coefficient for the length ratio
λ_2	Weight coefficient for the width ratio
λ_3	Weight coefficient for the area ratio

References

1. Hong, J.; Shen, G.Q.; Mao, C.; Li, Z.; Li, K. Life-Cycle Energy Analysis of Prefabricated Building Components: An Input-Output-Based Hybrid Model. *J. Clean. Prod.* **2016**, *112*, 2198–2207. [CrossRef]
2. Kong, L.; Li, H.; Luo, H.; Ding, L.; Luo, X.; Skitmore, M. Optimal Single-Machine Batch Scheduling for the Manufacture, Transportation and JIT Assembly of Precast Construction with Changeover Costs within Due Dates. *Autom. Constr.* **2017**, *81*, 34–43. [CrossRef]

3. Aye, L.; Ngo, T.; Crawford, R.H.; Gammampila, R.; Mendis, P. Life Cycle Greenhouse Gas Emissions and Energy Analysis of Prefabricated Reusable Building Modules. *Energy Build.* **2012**, *47*, 159–168. [CrossRef]
4. Liu, W.; Tao, X.; Mao, C.; He, W. Scheduling Optimization for Production of Prefabricated Components with Parallel Work of Serial Machines. *Autom. Constr.* **2023**, *148*, 104770. [CrossRef]
5. Zhang, R.; Feng, X.; Mou, Z.; Zhang, Y. Green Optimization for Precast Production Rescheduling Based on Disruption Management. *J. Clean. Prod.* **2023**, *420*, 138406. [CrossRef]
6. Xu, A.; Zhu, Y.; Wang, Z. Carbon Emission Evaluation of Eight Different Prefabricated Components during the Materialization Stage. *J. Build. Eng.* **2024**, *89*, 109262. [CrossRef]
7. Dan, Y.; Liu, G.; Fu, Y. Optimized Flowshop Scheduling for Precast Production Considering Process Connection and Blocking. *Autom. Constr.* **2021**, *125*, 103575. [CrossRef]
8. Du, J.; Zhang, J.; Castro-Lacouture, D.; Hu, Y. Lean Manufacturing Applications in Prefabricated Construction Projects. *Autom. Constr.* **2023**, *150*, 104790. [CrossRef]
9. Wang, D.; Liu, G.; Li, K.; Wang, T.; Shrestha, A.; Martek, I.; Tao, X. Layout Optimization Model for the Production Planning of Precast Concrete Building Components. *Sustainability* **2018**, *10*, 1807. [CrossRef]
10. Hu, H. A Study of Resource Planning for Precast Production. *Archit. Sci. Rev.* **2007**, *50*, 106–114. [CrossRef]
11. Khalili, A.; Chua, D.K. Integrated Prefabrication Configuration and Component Grouping for Resource Optimization of Precast Production. *J. Constr. Eng. Manag.* **2014**, *140*, 04013052. [CrossRef]
12. Du, J.; Dong, P.; Sugumaran, V.; Castro-Lacouture, D. Dynamic Decision Support Framework for Production Scheduling Using a Combined Genetic Algorithm and Multiagent Model. *Expert Syst.* **2021**, *38*, e12533. [CrossRef]
13. de Athayde Prata, B.; Pitombeira-Neto, A.R.; de Moraes Sales, C.J. An Integer Linear Programming Model for the Multiperiod Production Planning of Precast Concrete Beams. *J. Constr. Eng. Manag.* **2015**, *141*, 04015029. [CrossRef]
14. Xiang, Y.; Ma, K.; Mahamadu, A.M.; Florez-Perez, L.; Zhu, K.; Wu, Y. Embodied Carbon Determination in the Transportation Stage of Prefabricated Constructions: A Micro-Level Model Using the Bin-Packing Algorithm and Modal Analysis Model. *Energy Build.* **2023**, *279*, 112640. [CrossRef]
15. Zou, Y.; Gao, Q.; Wang, S. Optimization of the Stacking Plans for Precast Concrete Slab Based on Assembly Sequence. *Buildings* **2022**, *12*, 1538. [CrossRef]
16. Ko, C.H.; Wang, S.F. GA-Based Decision Support Systems for Precast Production Planning. *Autom. Constr.* **2010**, *19*, 907–916. [CrossRef]
17. Wang, Z.; Hu, H. Improved Precast Production–Scheduling Model Considering the Whole Supply Chain. *J. Comput. Civ. Eng.* **2017**, *31*, 04017013. [CrossRef]
18. Chan, W.T.; Hu, H. Precast Production Scheduling with Genetic Algorithms. *Proc. IEEE Conf. Evol. Comput.* **2000**, *2*, 1087–1094. [CrossRef]
19. Chan, W.T.; Hu, H. An Application of Genetic Algorithms to Precast Production Scheduling. *Comput. Struct.* **2001**, *79*, 1605–1616. [CrossRef]
20. Warszawski, A. Production Planning in Prefabrication Plant. *Build. Environ.* **1984**, *19*, 139–147. [CrossRef]
21. Chan, W.T.; Hu, H. Production Scheduling for Precast Plants Using a Flow Shop Sequencing Model. *J. Comput. Civ. Eng.* **2002**, *16*, 165–174. [CrossRef]
22. Jiang, W.; Wu, L. Flow Shop Optimization of Hybrid Make-to-Order and Make-to-Stock in Precast Concrete Component Production. *J. Clean. Prod.* **2021**, *297*, 126708. [CrossRef]
23. Xiong, F.; Chu, M.; Li, Z.; Du, Y.; Wang, L. Just-in-Time Scheduling for a Distributed Concrete Precast Flow Shop System. *Comput. Oper. Res.* **2021**, *129*, 105204. [CrossRef]
24. Du, J.; Dong, P.; Sugumaran, V. Dynamic Production Scheduling for Prefabricated Components Considering the Demand Fluctuation. *Intell. Autom. Soft Comput.* **2020**, *26*, 715–723. [CrossRef]
25. Kim, T.; Kim, Y.-W.; Cho, H. Dynamic Production Scheduling Model under Due Date Uncertainty in Precast Concrete Construction. *J. Clean. Prod.* **2020**, *257*, 120527. [CrossRef]
26. Wang, Z.; Xu, H. Scheduling Redundancy Optimization for Precast Production to Enhance Disruption Management in Prefabricated Construction. *Comput. Ind. Eng.* **2024**, *192*, 110251. [CrossRef]
27. Leu, S.-S.; Hwang, S.T. A GA-Based Model for Maximizing Precast Plant Production under Resource Constraints. *Eng. Optim.* **2001**, *33*, 619–642. [CrossRef]
28. Ko, C.H.; Wang, S.F. Precast Production Scheduling Using Multi-Objective Genetic Algorithms. *Expert Syst. Appl.* **2011**, *38*, 8293–8302. [CrossRef]
29. Leu, S.-S.; Hwang, S.T. GA-Based Resource-Constrained Flow-Shop Scheduling Model for Mixed Precast Production. *Autom. Constr.* **2002**, *11*, 439–452. [CrossRef]
30. Li, X.; Li, Z.; Wu, G. Lean Precast Production System Based on the CONWIP Method. *KSCE J. Civ. Eng.* **2018**, *22*, 2167–2177. [CrossRef]

31. Du, J.; Xue, Y.; Sugumaran, V.; Hu, M.; Dong, P. Improved Biogeography-Based Optimization Algorithm for Lean Production Scheduling of Prefabricated Components. *Eng. Constr. Archit. Manag.* **2023**, *30*, 1601–1635. [CrossRef]
32. Huo, Z.; Wu, X.; Cheng, T. Scheduling Model for Prefabricated Component Assembly, Production, and Transportation Stage Based on GA for Prefabricated Buildings. *IEEE Access* **2024**, *12*, 60826–60838. [CrossRef]
33. Chan, W.T.; Hu, H. Constraint Programming Approach to Precast Production Scheduling. *J. Constr. Eng. Manag.* **2002**, *128*, 513–521. [CrossRef]
34. Ahmad, S.; Soetanto, R.; Goodier, C. Lean Approach in Precast Concrete Component Production. *Built Environ. Proj. Asset Manag.* **2019**, *9*, 457–470. [CrossRef]
35. Nam, S.; Yoon, J.; Kim, K.; Choi, B. Optimization of Prefabricated Components in Housing Modular Construction. *Sustainability* **2020**, *12*, 10269. [CrossRef]
36. Liu, Y.; Dong, J.; Shen, L. A Conceptual Development Framework for Prefabricated Construction Supply Chain Management: An Integrated Overview. *Sustainability* **2020**, *12*, 1878. [CrossRef]
37. Podolski, M. Effective Allocation of Manpower in the Production of Precast Concrete Elements With the Use of Metaheuristics. *J. Civ. Eng. Manag.* **2022**, *28*, 247–260. [CrossRef]
38. Chang, C.; Han, M. Production Scheduling Optimization of Prefabricated Building Components Based on DDE Algorithm. *Math. Probl. Eng.* **2021**, *2021*, 6672753. [CrossRef]
39. Liu, Z.; Liu, Z.; Liu, M.; Wang, J. Optimization of Flow Shop Scheduling in Precast Concrete Component Production via Mixed-Integer Linear Programming. *Adv. Civ. Eng.* **2021**, *2021*, 6637248. [CrossRef]
40. Zhang, Z.; Song, X.; Huang, H.; Yin, Y.; Lev, B. Scheduling Problem in Seru Production System Considering DeJong's Learning Effect and Job Splitting. *Ann. Oper. Res.* **2022**, *312*, 1119–1141. [CrossRef]
41. Wang, L.; Zhao, Y.; Yin, X. Precast Production Scheduling in Off-Site Construction: Mainstream Contents and Optimization Perspective. *J. Clean. Prod.* **2023**, *405*, 137054. [CrossRef]
42. Du, J.; Dong, P.; Wang, X. *Multi-Objective Genetic Algorithm Based Prefabricated Component Production Process Optimization Considering the Demand Fluctuations in the Construction Site*; Springer International Publishing: Cham, Switzerland, 2019; Volume 842, ISBN 9783319987750.
43. Zhang, H.; Yu, L. Dynamic Transportation Planning for Prefabricated Component Supply Chain. *Eng. Constr. Archit. Manag.* **2020**, *27*, 2553–2576. [CrossRef]
44. Zhang, H.; Yu, L. Site Layout Planning for Prefabricated Components Subject to Dynamic and Interactive Constraints. *Autom. Constr.* **2021**, *126*, 103693. [CrossRef]
45. Zou, C.; Zhu, J.; Ma, S.; Lou, K.; Lu, N.; Li, L. Optimal Transportation Scheduling of Prefabricated Components Based on Improved Hybrid Differential Firefly Algorithm. *Math. Probl. Eng.* **2022**, *2022*, 3302983. [CrossRef]
46. Wang, S.; Zhang, X. Production Scheduling of Prefabricated Components Considering Delivery Methods. *Sci. Rep.* **2023**, *13*, 15094. [CrossRef]
47. Marasini, R.; Dawood, N.S.; Hobbs, B. Stockyard Layout Planning in Precast Concrete Products Industry: A Case Study and Proposed Framework. *Constr. Manag. Econ.* **2001**, *19*, 365–377. [CrossRef]
48. Kim, T.; Kim, Y.W.; Lee, D.; Kim, M. Reinforcement Learning Approach to Scheduling of Precast Concrete Production. *J. Clean. Prod.* **2022**, *336*, 130419. [CrossRef]
49. Du, Y.; Li, J. qing A Deep Reinforcement Learning Based Algorithm for a Distributed Precast Concrete Production Scheduling. *Int. J. Prod. Econ.* **2024**, *268*, 109102. [CrossRef]
50. Tharmmaphornphilas, W.; Sareinpithak, N. Formula Selection and Scheduling for Precast Concrete Production. *Int. J. Prod. Res.* **2013**, *51*, 5195–5209. [CrossRef]
51. Lim, J.; Kim, D.Y. Integrated Management Model of Production and Yard Stock for In-Situ Precast Concrete Production. *J. Asian Archit. Build. Eng.* **2023**, *22*, 286–302. [CrossRef]
52. Benjaoran, V.; Dawood, N.; Hobbs, B. Flowshop Scheduling Model for Bespoke Precast Concrete Production Planning. *Constr. Manag. Econ.* **2005**, *23*, 93–105. [CrossRef]
53. Yang, Z.; Ma, Z.; Wu, S. Optimized Flowshop Scheduling of Multiple Production Lines for Precast Production. *Autom. Constr.* **2016**, *72*, 321–329. [CrossRef]
54. Zheng, R.; Li, Z.; Li, L.; Dou, Y.; Yuan, M.; Yin, X. Proposing a Lean-Optimized Scheduling Model of Mixed-Flow Prefabricated Component Production in Off-Site Construction. *J. Constr. Eng. Manag.* **2024**, *150*, 04024088. [CrossRef]
55. Zheng, R.; Li, Z.; Li, L.; Ma, S.; Li, X. Group Technology Empowering Optimization of Mixed-Flow Precast Production in off-Site Construction. *Environ. Sci. Pollut. Res.* **2024**, *31*, 11781–11800. [CrossRef]
56. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
57. Tseng, H.E.; Huang, Y.M.; Chang, C.C.; Lee, S.C. Disassembly Sequence Planning Using a Flatworm Algorithm. *J. Manuf. Syst.* **2020**, *57*, 416–428. [CrossRef]

58. Liang, J.; Guo, S.; Du, B.; Liu, W.; Zhang, Y. Restart Genetic Flatworm Algorithm for Two-Sided Disassembly Line Balancing Problem Considering Negative Impact of Destructive Disassembly. *J. Clean. Prod.* **2022**, *355*, 131708. [CrossRef]
59. Fırat, H.; Alpaslan, N. An Effective Approach to the Two-Dimensional Rectangular Packing Problem in the Manufacturing Industry. *Comput. Ind. Eng.* **2020**, *148*, 106687. [CrossRef]
60. Hu, Y.; Zuo, Y.; Sun, Z. Combination of Simulated Annealing Algorithm and Minimum Horizontal Line Algorithm to Solve Two-Dimensional Pallet Loading Problem. In Proceedings of the 2022 Winter Simulation Conference (WSC), Singapore, 11–14 December 2022; pp. 1956–1966. [CrossRef]
61. Lai, K.K.; Chan, J.W.M. Developing a Simulated Annealing Algorithm for the Cutting Stock Problem. *Comput. Ind. Eng.* **1997**, *32*, 115–127. [CrossRef]
62. Zhao, H.; Wang, Y.; Feng, X. A Surplus Rectangle Fill Algorithm for Petrochemical Industry Area-Wide Layout Optimization with Key Plant Constraint. *Chem. Eng. Trans.* **2017**, *61*, 961–966. [CrossRef]
63. Holstein, T.W. What Makes Flatworms Go to Pieces. *Nature* **2019**, *572*, 593–594. [CrossRef] [PubMed]
64. Li, X.; Gao, L. An Effective Hybrid Genetic Algorithm and Tabu Search for Flexible Job Shop Scheduling Problem. *Int. J. Prod. Econ.* **2016**, *174*, 93–110. [CrossRef]
65. Gmira, M.; Gendreau, M.; Lodi, A.; Potvin, J.Y. Tabu Search for the Time-Dependent Vehicle Routing Problem with Time Windows on a Road Network. *Eur. J. Oper. Res.* **2021**, *288*, 129–140. [CrossRef]
66. Arashpour, M.; Kamat, V.; Bai, Y.; Wakefield, R.; Abbasi, B. Optimization Modeling of Multi-Skilled Resources in Prefabrication: Theorizing Cost Analysis of Process Integration in off-Site Construction. *Autom. Constr.* **2018**, *95*, 1–9. [CrossRef]
67. Gao, K.Z.; He, Z.M.; Huang, Y.; Duan, P.Y.; Suganthan, P.N. A Survey on Meta-Heuristics for Solving Disassembly Line Balancing, Planning and Scheduling Problems in Remanufacturing. *Swarm Evol. Comput.* **2020**, *57*, 100719. [CrossRef]
68. Shang, K.; Ishibuchi, H.; He, L.; Pang, L.M. A Survey on the Hypervolume Indicator in Evolutionary Multiobjective Optimization. *IEEE Trans. Evol. Comput.* **2021**, *25*, 1–20. [CrossRef]