processes

MDPI

Article

A Multi-Granularity Random Mutation Genetic Algorithm for Steel Cold Rolling Scheduling Optimization

Hairong Yang [1,2], Xiao Ji [1,*], Haiyan Sun [2], Yonggang Li [1] and Weidong Qian [2]

[1] School of Automation, Central South University, Changsha 410083, China; yanghairong@baosight.com (H.Y.); liyonggang@csu.edu.cn (Y.L.)
[2] Shanghai Baosight Software Co., Ltd., Shanghai 201203, China; sunhaiyan_195299@baosight.com (H.S.); qianweidong@baosight.com (W.Q.)
* Correspondence: season_jx@csu.edu.cn

Abstract

Cold rolling is the precision finishing stage in the steel production process, and its scheduling optimization is essential for enhancing production efficiency. To address the complex process constraints and objectives, this paper proposes a multi-granularity random mutation genetic algorithm (MGRM-GA) for cold rolling scheduling optimization. First, a multi-objective collaborative optimization model is established to integrate the production cost and process constraints. Then, high-quality initial solutions are generated based on greedy heuristic rules to fulfill the cold rolling constraints. Finally, four random mutation strategies are designed at different task granularities and unit levels to search diverse candidates. The standard flexible job shop scheduling problem (FJSP) datasets and practical cold rolling production data are studied to validate the feasibility and competitiveness of the MGRM-GA. Experimental results show that the MGRM-GA achieves a 94.2% improvement in objective function optimization, a 14.8-fold increase in throughput, and a 94.8% reduction in execution time on cold rolling data. Compared with the heuristic mutation algorithm, MGRM-GA increases population heterogeneity and avoids premature convergence, which enhances global search ability and scheduling performance.

Keywords: cold rolling scheduling; genetic algorithm; flexible job shop scheduling problem (FJSP); multi-objective optimization

check for updates

Academic Editors: Gongzhuang Peng and Shenglong Jiang

Received: 11 September 2025
Revised: 6 October 2025
Accepted: 14 October 2025
Published: 16 October 2025

Citation: Yang, H.; Ji, X.; Sun, H.; Li, Y.; Qian, W. A Multi-Granularity Random Mutation Genetic Algorithm for Steel Cold Rolling Scheduling Optimization. *Processes* **2025**, *13*, 3311. https://doi.org/10.3390/pr13103311

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The steel industry is a fundamental pillar of the national economy, playing a crucial role in the economic development and holding significant strategic importance for national security [1–3]. However, the industry still faces severe challenges, including reliance on imports of high-grade steel and vulnerabilities within the industrial and supply chains systems [4,5]. As a critical finishing stage in steel production, the cold rolling process is characterized by complex workflows [6], stringent quality control, and the need to meet diverse customer requirements [7,8]. It plays a decisive role in product competitiveness, production-line efficiency, and resource allocation, making it indispensable within the steel industry [9]. However, cold rolling production involves multi-stage and multi-unit collaboration across annealing, rolling, and coating [10], with long process flows and complex constraints [11]. On the one hand, it must simultaneously satisfy multiple objectives such as delivery deadlines, batch production, and specification continuity. On the other hand, differences in machine capacities and equipment conditions require cross-unit load balancing while ensuring batch compatibility and minimizing machine changeover times [12].

Therefore, cold rolling scheduling must seek global optimality under multiple objectives while maintaining feasibility and computational efficiency, making it a highly complex and challenging optimization problem [13].

The flexible job shop scheduling problem (FJSP) [14,15] is a complex scheduling optimization problem that involves assigning processing sequences for multiple jobs on multiple alternative machines to minimize the total makespan, and it provides an effective approach for addressing this type of scheduling challenge. At present, researchers worldwide primarily employ approaches such as mathematical programming approaches, intelligent optimization and metaheuristic algorithms, and hybrid and evolutionary algorithms to address the FJSP. Nouiri et al. [16] applied a particle swarm optimization algorithm to solve flexible job-shop scheduling problems, addressing machine assignment and operation sequencing to minimize makespan. Jiang et al. [17] proposed a discrete grey wolf optimization algorithm with adaptive mutation and variable neighborhood search to solve job shop and flexible job shop scheduling problems, aiming to minimize makespan. Pan et al. [18] proposed a learning-based multipopulation evolutionary optimization algorithm to solve flexible job shop scheduling problems with finite transportation resources, improving solution quality and robustness. Luan et al. [19] proposed a discrete whale optimization algorithm to solve low-carbon flexible job shop scheduling problems, aiming to minimize completion time and energy consumption costs. Li et al. [20] proposed a hybrid algorithm combining genetic algorithm and tabu search to solve flexible job shop scheduling problems, aiming to minimize makespan. Ishikawa et al. [21] proposed a hierarchical multi-space competitive distributed genetic algorithm to solve flexible job shop scheduling problems efficiently, reducing computational cost without problem-specific operators. Gao et al. [22] proposed an improved artificial bee colony algorithm to solve flexible job shop scheduling problems with fuzzy processing times, aiming to minimize fuzzy completion time and machine workload.

Some studies have also focused on specific processes in steel production, developing tailored solutions based on their unique characteristics. Li et al. [23] proposed an adaptive multi-objective differential evolution algorithm based on deep reinforcement learning to solve multi-objective scheduling problems in steel annealing, balancing setup, earliness, and tardiness costs. Zhao et al. [24] proposed an improved discrete differential evolution algorithm for a supply chain scheduling problem in cold rolling, realizing production capacity balance and changeover cost minimization across multiple lines. Zhao et al. [25] proposed a differential evolution algorithm (Sa-PDDE), by which coil scheduling problems in parallel continuous annealing lines are solved with the aim of reducing changeover cost and improving capacity utilization.

Although the FJSP has been extensively studied as a complex scheduling problem [26], existing research rarely systematically investigates the specific characteristics of the cold rolling process. Consequently, current approaches fail to adequately capture the multi-objective and multi-scenario requirements and provide limited analysis of cross-stage and multi-scenario adaptive scheduling issues. Therefore, it is essential to develop a scheduling optimization model tailored to the practical requirements of the cold rolling process. To this end, this study proposes a scheduling optimization method for cold rolling production lines based on a multi-granularity random mutation genetic algorithm (MGRM-GA), enabling effective production scheduling.

To address the coexistence of multiple processes and objectives in cold rolling scheduling, a multi-objective collaborative optimization model for intelligent cold rolling scheduling is constructed. The model integrates critical constraints such as batch production, specification succession, and delivery deadlines, thereby aligning more closely with the practical scheduling requirements of cold rolling production lines. To address the issue of

low-quality initial solutions and frequent generation of infeasible solutions in traditional genetic algorithms for cold rolling scheduling, a greedy heuristic-based initialization method is proposed. This approach improves the quality of initial solutions while maintaining population diversity, strictly adheres to cold rolling process constraints, and effectively avoids infeasibility caused by random operations. To overcome the rule-dependency and poor adaptability of traditional heuristic mutation strategies, MGRM-GA is proposed. Four types of mutation mechanisms are systematically designed that incorporate differences in both task granularity and unit hierarchy, thereby enhancing the algorithm's generalization ability and practical applicability.

Furthermore, the proposed method is validated on both standard job shop benchmark datasets and practical cold rolling production data, and the remainder of the paper sequentially introduces the mathematical modeling of the FJSP and the cold rolling scheduling problem, details the proposed method, presents the experimental results, and finally concludes with discussions and future directions.

2. Problem Formulation

2.1. Basic Model of the FJSP

In the cold rolling scheduling problem, FJSP serves as a fundamental framework for modeling the solution [27,28]. The FJSP is defined as scheduling n jobs, where each operation can be assigned to an available set of machines $M = \{m_1, m_2, ..., m_p\}$. Each job consists of k operations, represented by $J = \{j_{11}, j_{12}, ..., j_{1k}, ..., j_{n1}, j_{n2}, ..., j_{nk}\}$, where k denotes the maximum number of operations among all tasks in the problem to be solved. The operations must be processed in a strict order, while being executed on different machines. The objective of solving the FJSP is to assign each operation to an appropriate machine and determine its start time in order to optimize specific performance measures of the entire system [29]. Depending on the range of machine options available for each operation, the FJSP can be classified into Total FJSP (T-FJSP) and Partial FJSP (P-FJSP) [30]. In T-FJSP, all operations can be processed on any machine, whereas in P-FJSP, at least one operation is restricted to a subset of the available machines [31]. In the steel cold rolling scheduling problem, the formulation is based on the P-FJSP. In addition, the scheduling process is subject to satisfy the following fundamental constraints [32]:

(1) A machine can process only one job at any given time.
(2) Each operation of a job can be assigned to only one machine at a time.
(3) Once the processing of an operation begins, it cannot be interrupted.
(4) All jobs are available for processing at time zero.

For ease of reference in the following sections, the notations are defined in Abbreviations.

2.2. Model of the Steel Cold Rolling Scheduling Problem

The scheduling problem of the steel cold rolling production line is characterized as a multi-stage coil production scheduling problem. In this process, the materials to be processed by each unit are determined and their production sequence is decided on the basis of operation requirements and inventory conditions, so that rationality and efficiency in steel production can be ensured.

2.2.1. Objective Function

In the cold rolling scheduling problem studied in this paper, the production process consists of annealing, rolling, and coating operations. Among them, annealing and rolling are each performed by a single unit, while coating is handled by three units, with one selected for processing. The primary objective of this study is to minimize overall production

costs while satisfying all process constraints, leading to the construction of the following multi-objective optimization model:

$$\min F = \alpha_1 f_1 + \alpha_2 f_2 + \alpha_3 f_3 + \alpha_4 f_4 \tag{1}$$

where f_1~f_4 represent the penalties associated with delivery performance, production cost and roll-change losses, specification switching, and inventory balance, respectively. In particular, f_1 accounts for penalties due to both delayed and premature deliveries, f_2 is designed to reduce switching losses and roll wear costs caused by specification changes, f_3 is formulated to minimize specification changes during production through reasonable batching, and f_4 ensures production continuity while avoiding excessive inventory accumulation. The mathematical formulations are formulated as follows:

$$\begin{cases} f_1 = \sum\limits_{j\in J} \omega_j \cdot \max(0, C_j - D_j) + \sum\limits_{j\in J} \max(0, D_j - C_j) \\ f_2 = \sum\limits_{i\in M} \sum\limits_{o\in O} x_{io} \cdot c^1_{io} + \sum\limits_{r,r'\in R} y_{rr'} \cdot (c^2_{rr'} + \alpha \cdot t^s_{rr'}) + \sum\limits_{i,i'\in M} \sum\limits_{o\in O} x_{io} \cdot (z_{oops} c^3_{oops} + s_{ii'o} c^4_{ii'o}) \\ f_3 = \sum\limits_{g\in G} (\gamma_g \cdot L^{trans}_g) \\ f_4 = \sum\limits_{t\in T} (\mathrm{WIP}_t - \mathrm{WIP}_0)^2 \end{cases} \tag{2}$$

2.2.2. Constraints

(1) Basic Constraints

The cold rolling production process of steel coils involves multiple operations, multiple units, and multiple materials, which constitutes an NP-hard problem conforming to the FJSP framework. Accordingly, it is subject to the following basic time-related constraints:

$$\sum_{i\in M_o} x_{io} = 1, \quad \forall o \in O \tag{3}$$

$$t_o \geqslant t_{o'} + p_{io'} - L(1 - y_{oo'}), \quad \forall o, o' \in O, i \in M \tag{4}$$

$$y_{oo'} + y_{o'o} = 1, \quad \forall o, o' \in O, o \neq o' \tag{5}$$

$$t_o \geqslant t_{o'} + p_{io'} - L(2 - x_{io} - x_{io'} + y_{oo'}), \quad \forall i \in M, o, o' \in O \tag{6}$$

$$t_{o'} \geqslant t_o + p_{io} - L(3 - x_{io} - x_{io'} - y_{oo'}), \quad \forall i \in M, o, o' \in O \tag{7}$$

where Equations (3)–(7) are derived from the classical modeling framework of the FJSP and represent the fundamental constraints widely adopted in this field. Specifically, Equation (3) restricts each operation to be assigned to only one machine at any given time. Equations (4) and (5) ensure that a single machine can process only one operation at a time. Equations (6) and (7) further require that the processing intervals of any two operations on the same machine do not overlap, thereby preventing temporal conflicts.

(2) Cold Rolling Constraints

In the steel cold rolling production line, in addition to the basic temporal constraints defined in the FJSP model, additional production-specific constraints must also be satisfied. Steel coils are required to be produced in batches, and within a production cycle, width and thickness continuity must be maintained to minimize capacity loss. Meanwhile, the total processing time of all operations assigned to a given machine

must not exceed the machine's maximum available capacity. In addition, material inventory must meet safety stock requirements, and advance production beyond demand is prohibited.

$$\sum_{o\in O}\sum_{o'\in O} x_{io}x_{io'}z_{oo'} \geqslant \Gamma_i^{batch}, \quad \forall i \in M \tag{8}$$

$$|x_{io}h_o - x_{io'}h_{o'}| \leqslant \Delta H_{max}, \quad \forall i \in M, (o,o') \in A_i \tag{9}$$

$$|x_{io}w_o - x_{io'}w_{o'}| \leqslant \Delta W_{max}, \quad \forall i \in M, (o,o') \in A_i \tag{10}$$

$$\sum_{o\in O} x_{io}p_{io} \leqslant T_i^{max}, \quad \forall i \in M \tag{11}$$

$$I_i^{min} \leqslant \sum_{o\in O_i} x_{io}r_{io} \leqslant I_i^{max}, \quad \forall i \in M \tag{12}$$

where Equations (8)–(12) are constructed based on the typical process requirements of steel cold rolling production, aiming to capture the key constraints encountered in practical manufacturing [33]. Specifically, Equation (8) formulates the batching constraint in cold rolling production. Equations (9) and (10) formulate the specification transition constraints. Equation (11) specifies the machine capacity constraint. Equation (12) formulates the safety stock constraint.

3. Methods

3.1. Heuristic Initialization of Solutions

This paper proposes a scheduling optimization method referred to as MGRM-GA. The approach first generates heuristic initial solutions and subsequently applies a genetic algorithm with multi-granularity random mutation to solve the cold rolling scheduling problem. The overall workflow is depicted in Figure 1.

Traditional methods for solving the FJSP often rely on random strategies to generate initial solutions, which often result in low-quality solutions, thereby hindering subsequent optimization. To address this issue, this paper proposes a heuristic initialization method that leverages a greedy strategy to effectively enhance both the rationality and quality of the initial solutions. Specifically, a two-stage penalty function, $\text{Cost}_{\text{mac}}(j_{nk}, m_j)$ and $\text{Cost}_{\text{seq}}(j_{nk}, m_j)$, is designed to generate high-quality initial solutions under heuristic rules. For the initialization of unit assignments, each operation j_{nk} is assessed across all candidate machines $\{m_1, m_2, ..., m_p\}$ by computing its penalty value $\text{Cost}_{\text{mac}}(j_{nk}, m_j)$, taking into account batch attributes $\text{Cost}_{\text{camp}}(j_{nk}, m_j)$, capacity balance $\text{Cost}_{\text{balance}}(j_{nk}, m_j)$, machine preference $\text{Cost}_{\text{mac_priority}}(j_{nk}, m_j)$, and raw material penalties $\text{Cost}_{\text{mat}}(j_{nk}, m_j)$. The machine with the minimum $\text{Cost}_{\text{mac}}(j_{nk}, m_j)$ is selected, as is which j_{nk} is assigned to that machine. To determine the processing sequence on each machine, a sequence penalty function $\text{Cost}_{\text{seq}}(j_{nk}, m_j)$ is introduced to account for specification continuity $\text{Cost}_{\text{concat}}(j_{nk}, m_j)$, delayed delivery $\text{Cost}_{\text{deadline}}(j_{nk}, m_j)$, machine idleness $\text{Cost}_{\text{utility}}(j_{nk}, m_j)$, and scheduling priority penalties $\text{Cost}_{\text{priority}}(j_{nk}, m_j)$. A greedy strategy is employed, under which the penalty values of the unscheduled operations on each machine are dynamically calculated and the lowest penalty operation is iteratively appended to the machine's schedule to generate the initial production sequence. The resulting initialization process is formulated as shown in Equation (13). Heuristic initialization provides a well-structured starting point in the solution space for random mutations. Building on high-quality initial solutions,

random mutation strategies are applied to achieve a complementary balance between heuristic local prioritization and randomized global exploration.

$$\begin{cases} \text{Cost}_{\text{mac}}(j_{nk}, m_j) = \text{Cost}_{\text{camp}}(j_{nk}, m_j) + \text{Cost}_{\text{balance}}(j_{nk}, m_j) \\ \qquad\qquad + \text{Cost}_{\text{mac_priority}}(j_{nk}, m_j) + \text{Cost}_{\text{mat}}(j_{nk}, m_j), \\ \text{Cost}_{\text{seq}}(j_{nk}, m_j) = \text{Cost}_{\text{concat}}(j_{nk}, m_j) + \text{Cost}_{\text{deadline}}(j_{nk}, m_j) \\ \qquad\qquad + \text{Cost}_{\text{utility}}(j_{nk}, m_j) + \text{Cost}_{\text{priority}}(j_{nk}, m_j). \end{cases} \tag{13}$$

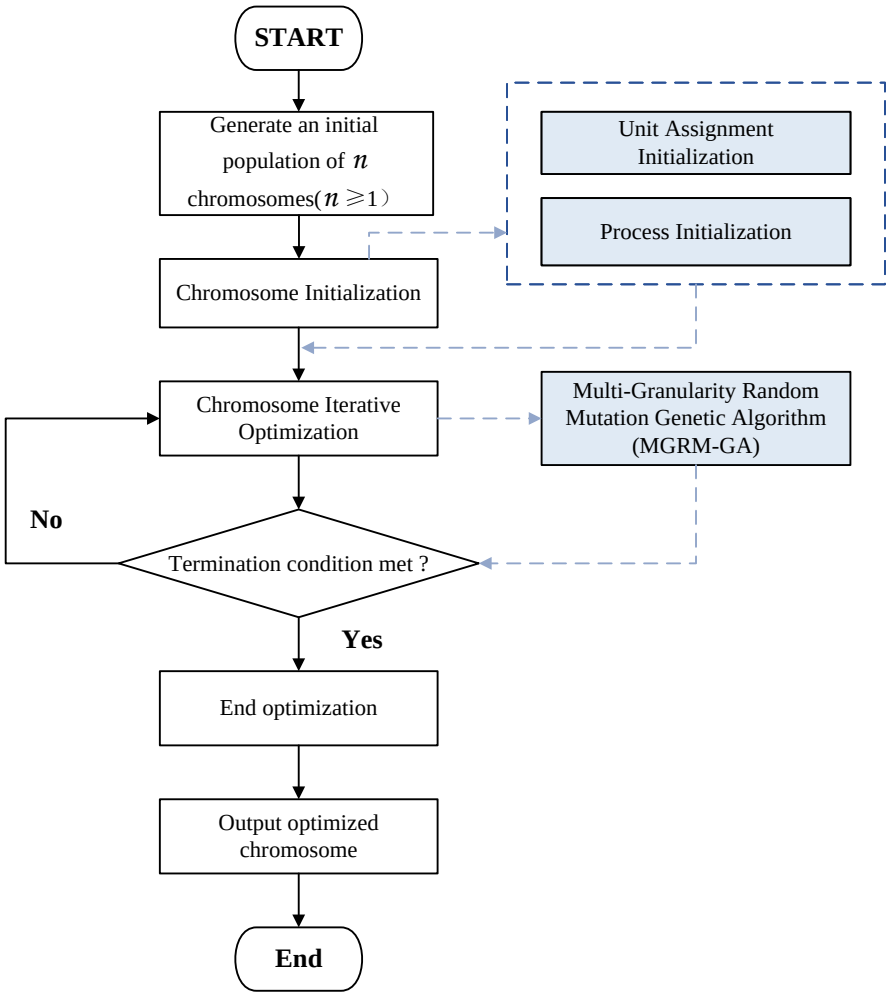


Figure 1. Flowchart of the proposed method.

3.2. Multi-Granularity Random Mutation Genetic Algorithm (MGRM-GA)

During the iterative optimization process, this study introduces targeted modifications to the crossover and mutation steps of the traditional Genetic Algorithm (GA) to improve the feasibility and stability of chromosomal solutions. First, in the crossover stage, considering the multiple process constraints in cold rolling scheduling—such as batch production and specification continuity—the traditional approach of generating offspring by exchanging parent chromosome segments often causes significant disruption to the overall solution structure, frequently leading to infeasible solutions that violate critical constraints. To avoid the high repair cost associated with such structural disruptions and to reduce interference with the feasible solution space, the crossover operation is removed in the proposed optimization process. Second, in the mutation stage, a multi-granularity random mutation approach is proposed to overcome the strong reliance of traditional heuristic mutation strategies on task-specific rules and domain expertise, thereby enhancing the

adaptability and scalability of the algorithm. The random mutation strategy introduces stochastic perturbations into the current solution within evolutionary algorithms to avoid premature convergence to local optima and enhance global search capability. This method incorporates four types of mutation operations: (i) single-task movement within the same unit, (ii) batch-task movement within the same unit, (iii) single-task movement across different units, and (iv) batch-task movement across different units. To ensure the stability and computational efficiency of these mutation strategies, a unified mutation control mechanism is developed. The key parameter is the maximum number of mutation attempts for each task, denoted by *submit_times*. This mechanism balances solution quality and computational efficiency through ensuring sufficient mutation attempts while controlling computational overhead. The calculation is formulated as Equation (14):

$$\text{submit_times} = \frac{N_{\max}}{N_{\text{now}}} \tag{14}$$

3.2.1. Random Single-Task Movement Within the Same Unit

This strategy controls the mutation frequency of each candidate task based on its normalized scheduling objective penalty, thereby emphasizing the role of task importance in directing the mutation process. For the task sequence within the same unit, all tasks except the currently executing first task are traversed sequentially. Each task is encoded as a gene, and the normalized value of F, denoted as $F' \in [0, 1]$, is calculated. The actual number of mutation attempts, denoted as times, is then obtained by multiplying F' with the maximum mutation attempt limit *submit_times*. For each attempt, an insertion point is generated uniformly at random, and the current task is inserted into that position to create a mutated individual. If the insertion violates scheduling constraints, the mutation attempt is discarded. An illustration of this procedure is presented in Figure 2. For example, in a given scheduling instance, the current task sequence of unit A is represented as $\{G_1, G_2, G_3, G_4, G_5, G_6\}$. If task G_2 is randomly selected and inserted after task G_5, the new sequence becomes $\{G_1, G_3, G_4, G_5, G_2, G_6\}$.

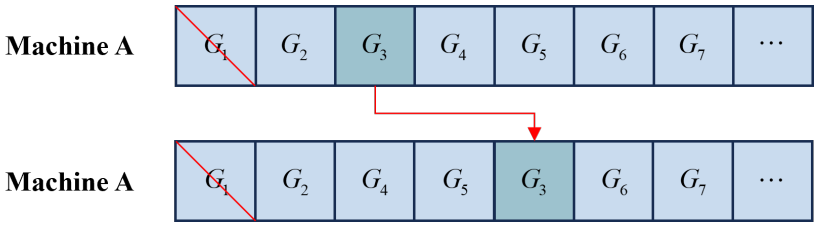


Figure 2. Random single-task movement within the same unit.

3.2.2. Random Batch-Task Movement Within the Same Unit

The random single-task movement strategy is further extended to the movement of consecutive tasks formulated as a gene segment. The initial segment length is set to $gene_num = 1$, and the segment is incrementally expanded from the current task onward. For each subsequent task that can potentially be included in the segment, its inclusion is determined with a probability of $1 - F'$, where F' denotes the normalized penalty value. The expansion terminates once the probability condition is not satisfied or the end of the task sequence is reached. Subsequently, a random insertion point within the same unit is selected, and the entire segment is inserted at that position to generate a new chromosome. An illustration of this process is presented in Figure 3. For example, in a given scheduling instance, the current task sequence of unit A is represented as $\{G_1, G_2, G_3, G_4, G_5, G_6\}$. If a consecutive task segment $\{G_2, G_3, G_4\}$ is randomly selected and inserted after G_6, the new sequence becomes $\{G_1, G_5, G_6, G_2, G_3, G_4\}$.

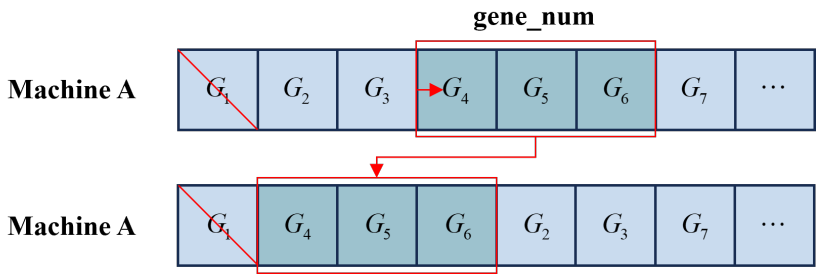


Figure 3. Random batch-task movement within the same unit.

3.2.3. Random Single-Task Movement Across Different Units

To enhance the global search capability of scheduling resources, a cross-unit task migration strategy is proposed. For each task, all feasible units are examined, and within each candidate unit, multiple insertion positions are generated using the same F weighted random mutation logic. The task is then inserted at a randomly selected position in the target unit to generate a new scheduling solution. If the insertion fails due to resource limitations or other constraints, the mutation attempt is discarded. An illustration of this process is illustrated in Figure 4. For example, in a given scheduling instance, the current task sequence of unit A is represented as $\{G_1, G_2, G_3\}$, and that of unit B as $\{H_1, H_2, H_3\}$. If a task G_2 from unit A is moved and inserted after H_2 in unit B, the resulting sequences are updated as follows: unit A becomes $\{G_2, G_3\}$, while unit B becomes $\{H_1, H_2, G_2, H_3\}$.

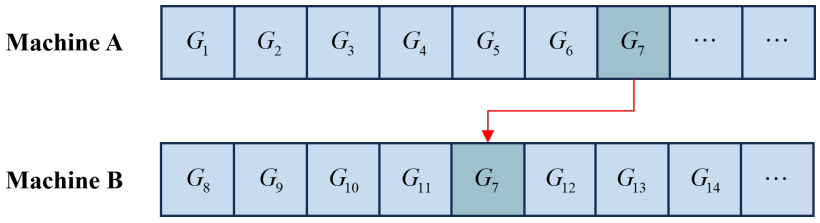


Figure 4. Random single-task movement across different units.

3.2.4. Random Batch-Task Movement Across Different Units

This strategy integrates batch-task mutation with cross-unit allocation. First, a chromosome segment is constructed based on $1 - F'$. The feasibility of all tasks within the segment on the target unit is subsequently evaluated. If all tasks are feasible, a random insertion is performed. The insertion point is randomly generated within the target unit, and the entire segment is inserted to that position. If any task in the segment cannot be executed on the target unit, the mutation attempt is discarded. An illustration of this process is illustrated in Figure 5. For example, in a given scheduling instance, the current task sequence of unit A is represented as $\{G_1, G_2, G_3, G_4\}$, and that of unit B as $\{H_1, H_2, H_3\}$. If a consecutive task segment $\{G_2, G_3\}$ is randomly selected from unit A and inserted after H_2 in unit B, the resulting sequences are updated as follows: unit A becomes $\{G_1, G_4\}$, while unit B becomes $\{H_1, H_2, G_2, G_3, H_3\}$.

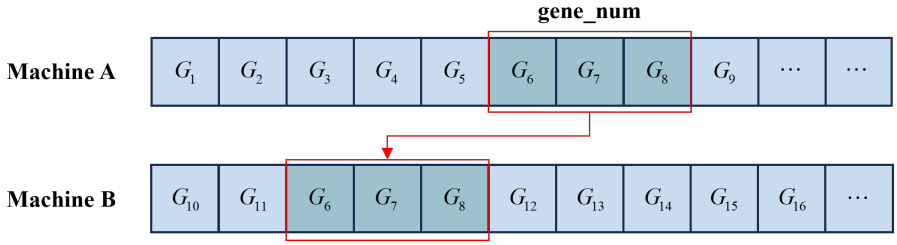


Figure 5. Random batch-task movement across different units.

Single-task and batch-task movements within the same unit correspond to scenarios where the sequence on a single production line needs to be adjusted due to small-batch insertions or specification transitions. In contrast, cross-unit single-task and batch-task movements reflect the need to reallocate tasks among parallel units to balance production capacity or adapt to line switching. These four mutation operators are well aligned with the flexible scheduling requirements of real-world production.

4. Experiments

4.1. Experimental Parameter Settings

In this section, the performance and effectiveness of the proposed method are comprehensively evaluated through a series of experiments. The basic experimental configuration, including the operating environment and parameter settings, is also presented. The experiments were conducted on a server equipped with dual Intel Xeon Gold 6146 CPUs (24 cores and 48 threads), 256 GB of memory, running the CentOS 7.4 operating system.

A simulation environment was constructed based on historical data from a cold rolling production line, using daily orders as the basic unit of analysis. Each order includes three processing stages: annealing, rolling, and coating. The production system is composed of an annealing machine (M11), a rolling machine (M21), and three coating machines (M31–M33). Order data covering six days were collected, with approximately 800 jobs recorded per day. Key parameters of the orders, including width, thickness, and temperature, were interpolated and statistically analyzed, as shown in Table 1, demonstrating the feasibility of continuous production across orders. The algorithm parameters were set with a population size of 14 and three iterations. Since customized multi-granularity random mutation operators were employed, mutation operations were enforced in every generation, eliminating the need for a separately defined mutation probability.

Table 1. Parameter settings of actual cold rolling production line data.

Physical Attributes	Value Range	Process Flexibility	Allowable Deviation
Width (mm)	730, 1100	ΔWidth (mm)	<100
Thickness (mm)	0.14, 0.30	ΔThickness (mm)	<0.15
Temperature (°C)	730, 1100	ΔTemperature (°C)	<40

4.2. Comparative Experiments

Existing general genetic algorithms or other metaheuristic approaches struggle to directly satisfy the stringent process constraints and real-time requirements of cold rolling production lines. Therefore, comparative experiments were conducted between the MGRM-GA and the rule-based heuristic mutation approach. The experimental datasets include both standard scheduling benchmark datasets and six days of order data collected from a cold rolling production line. The evaluation metrics comprise runtime, objective function value, and throughput. A shorter runtime indicates higher computational efficiency, a smaller objective function value reflects better scheduling quality, and a higher throughput represents the ability to process a larger number of mutated chromosomes per unit time.

4.2.1. Comparison on Benchmark Datasets

The experimental results on 74 standard FJSP benchmark datasets are presented in Figure 6. Due to differences in dataset scale, process complexity, and machine availability, both runtime and makespan exhibit some fluctuations. On average, the proposed algorithm requires 10.04 s, which is slightly longer than the 1.32 s of the rule-based heuristic mutation algorithm. However, in terms of the core metric of makespan, it achieves an average reduction of approximately 66% compared with the heuristic approach. These results

demonstrate that the proposed multi-granularity random mutation strategy effectively improves solution quality in standard scheduling scenarios, confirming its validity and strong generalization capability in solving generic scheduling problems.

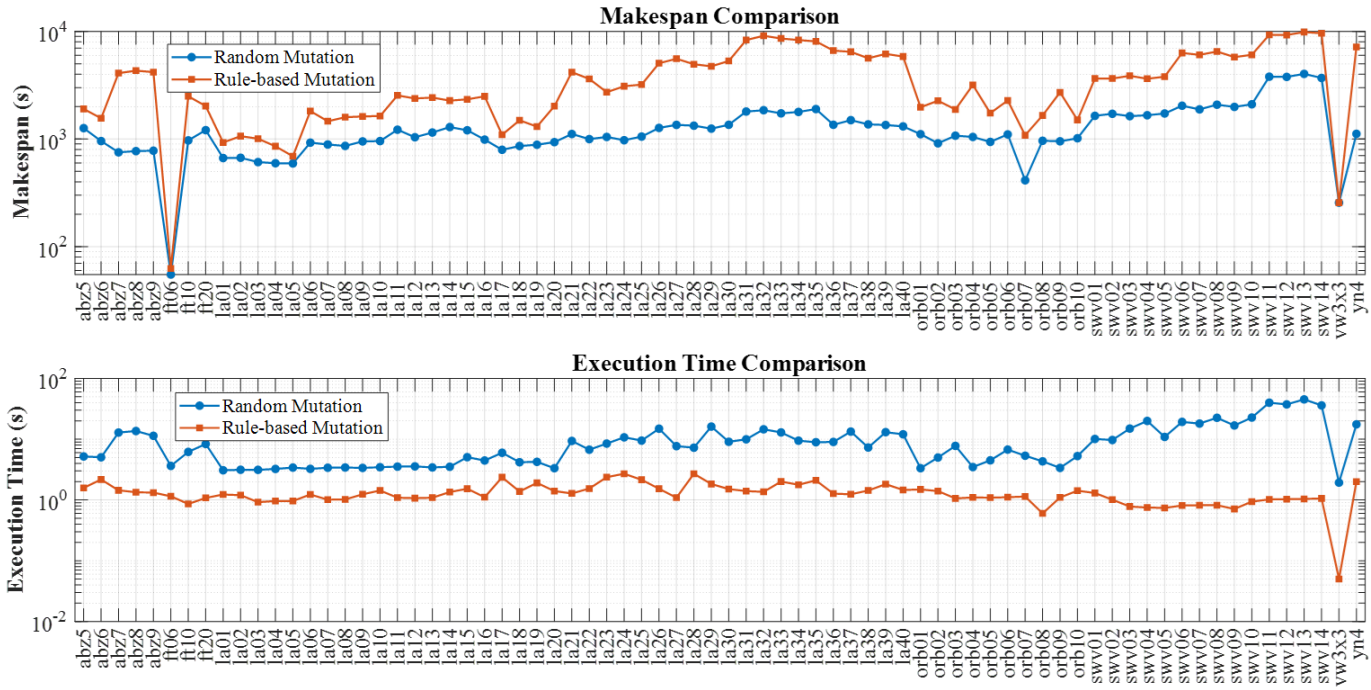


Figure 6. Parameter comparison on benchmark datasets between the MGRM-GA method and the rule-based heuristic mutation method.

4.2.2. Comparison on Cold Rolling Production Line Data

The experimental results on six days of real cold rolling production orders are presented in Figures 7–9. Since the order structure and production load vary from day to day, the overall production conditions also differ; thus, average values are reported for comparison. The proposed algorithm achieves an average runtime of 7.91 s, representing a reduction of approximately 95% compared with the 153.13 s required by the rule-based heuristic mutation algorithm. The average objective function penalty is 33,479.40, a reduction of about 94% from 579,881.86, while the average throughput reaches 24,121.22, nearly 14.8 times higher than 1628.98. The optimized scheduling results are illustrated in the Gantt chart in Figure 10, showing more compact task sequencing, significantly reduced idle time, and an improved overall production rhythm. Moreover, the proposed MGRM-GA has been integrated into the industrial scheduling system, demonstrating stable operation and consistent results. These findings indicate that, in real-world industrial settings, the method not only ensures high-quality scheduling solutions but also substantially enhances computational efficiency, highlighting its superior engineering applicability and practical value.

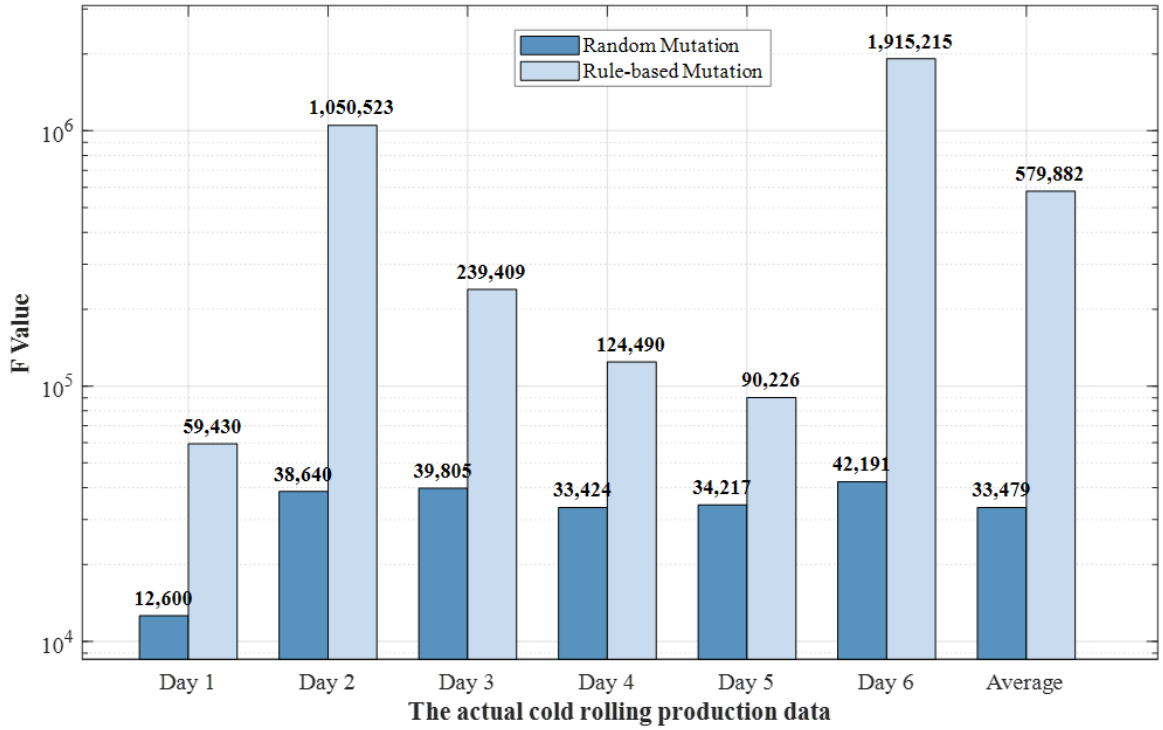


Figure 7. Comparison of F value between the MGRM-GA method and the rule-based heuristic mutation method on cold rolling production line data.

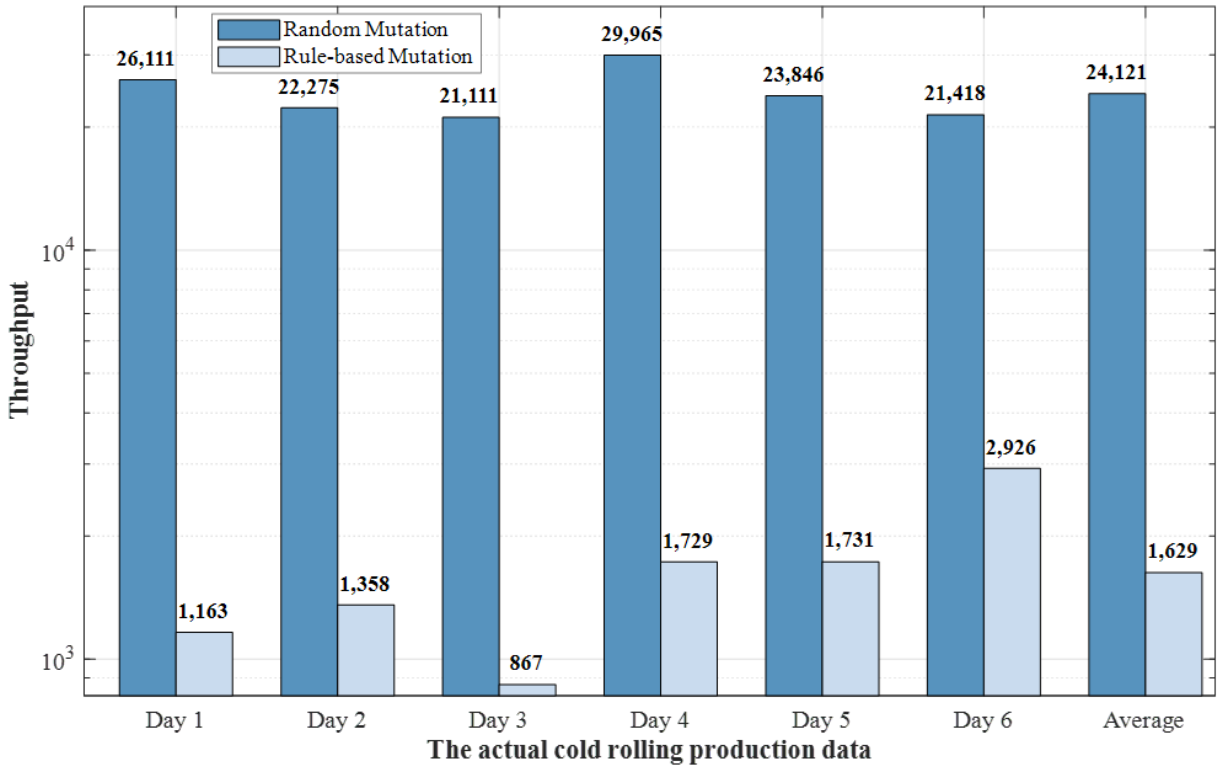


Figure 8. Comparison of throughput between the MGRM-GA method and the rule-based heuristic mutation method on cold rolling production line data.

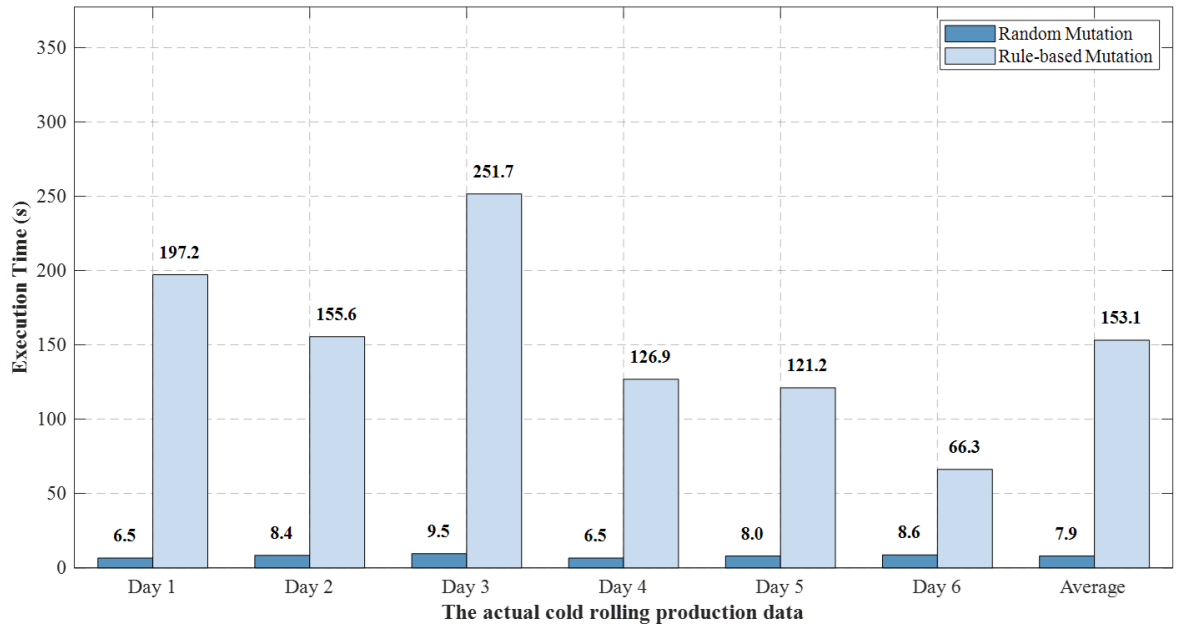


Figure 9. Comparison of execution time between the MGRM-GA method and the rule-based heuristic mutation method on cold rolling production line data.

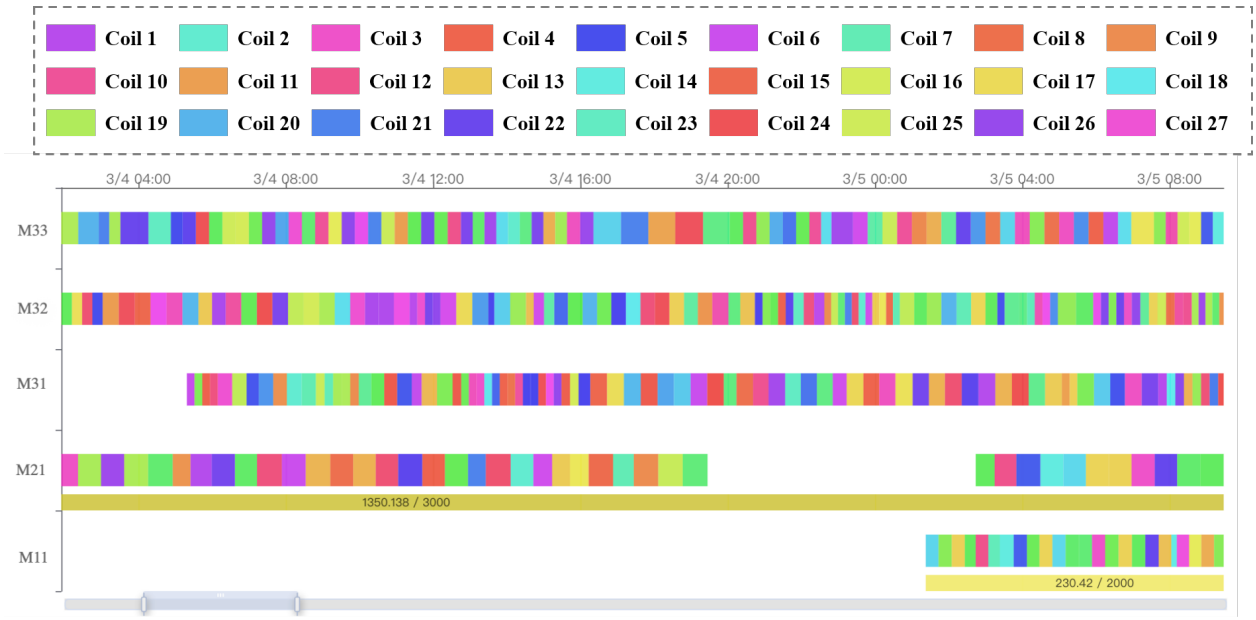


Figure 10. Gantt chart of the MGRM-GA method applied to the actual cold rolling production line (different colors represent different types of steel coils).

5. Conclusions

This study proposes a scheduling optimization method for steel cold rolling production lines based on the MGRM-GA, addressing the complex engineering characteristics of cold rolling scheduling. By constructing a multi-objective collaborative optimization model, introducing a greedy heuristic initialization strategy, and designing four random mutation operators covering both task- and unit-level granularities, the method achieves efficient scheduling optimization under multi-process, multi-machine, and multi-constraint conditions. Although MGRM-GA demonstrates excellent performance on both real cold rolling production data and standard FJSP benchmark datasets, certain limitations remain, including the risk of local optima, scalability challenges, and data sensitivity. Future

research may incorporate multi-population collaboration and adaptive search strategies to enhance the algorithm's capability of escaping local optima.

In addition, as a deliberate design choice based on the characteristics of the problem, MGRM-GA omits crossover operations since the complex process constraints in cold rolling scheduling can easily lead to infeasible solutions under traditional crossover and search mechanisms, making direct comparisons with algorithms such as NSGA-II or PSO impractical.

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

Funding: This work was supported by the National Natural Science Foundation of China (Grant No. 62273362) and the National Key Research and Development Program of China (Grant No. 2024YFB3312100).

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

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

Abbreviations

The following abbreviations are used in this manuscript:

Parameter	Description	Parameter	Description
J	Set of all production orders (jobs)	p_{io}	Processing time of operation o on machine i (h)
M	Set of all machines	$t^{s}_{rr'}$	Roll changing time of machine s from type r to r' (h)
O	Set of all operations	$T_i^{\max}$	Maximum available time of machine i (h)
R	Set of all roll types	$\alpha_{1-4}, \beta_{1-2}$	Weighting coefficients
T	Set of scheduling time points	x_{io}	Binary variable, equals 1 if operation o is executed on machine i, and 0 otherwise
M_o	Set of machines capable of executing operation o	$y_{oo'}$	Binary variable, equals 1 if operation o is executed before operation o', and 0 otherwise
j	Index of order	$y_{rr'}$	Binary variable, equals 1 if roll type changes from r to r', and 0 otherwise
i,i',k	Index of machine	$z_{oo'}$	Binary variable, equals 1 if a family switch occurs from operation o to o', and 0 otherwise
o,o'	Index of operation	$s_{ii'o}$	Binary variable, equals 1 if operation o is transferred from machine i to i', and 0 otherwise
r,r'	Index of roll type	c^{1}_{io}	Basic processing cost of executing operation o on machine i (CNY)
t	Index of time point	$c^{2}_{rr'}$	Roll changing cost from type r to r'
P	Total number of production lines	h	Index of warehouse (CNY)
n	Total number of orders	s	Total number of warehouses
t_o	Start time of operation o (h)	c^{3}_{oops}	Operation family switching configuration cost (CNY)
C_j	Completion time of order j (h)	$c^{4}_{ii'o}$	Transfer cost of the same job across different machines (CNY)
D_j	Due date of order j	h_o, w_o	Thickness and width of coil corresponding to operation o (mm)
G	Set of specification groups	$\Delta H_{\max}, \Delta W_{\max}$	Maximum allowed thickness/width variation between consecutive operations (mm)
L_g^{trans}	Transition length (mm)	Γ_i^{batch}	Minimum batch size requirement of machine i (t)
ω_j	Priority weight of order j	$I_i^{\min}, I_i^{\max}$	Minimum/maximum inventory level of buffer in front of machine i (t)
WIP_t	Work-in-process inventory level at time t (t)	r_{io}	Raw material consumption of operation o on machine i
WIP_0	Target work-in-process inventory level	γ	Specification switching cost coefficient
$N_{\max}$	Maximum mutation protection count	N_{now}	Number of tasks on the preceding unit

References

1. Du, H.; Cheng, L.; Qian, Z.; Zhou, Y.; Gao, Z.; Hou, L.; Wei, Y. Study on Smelting Process Parameters of a Blast Furnace with Hydrogen-Rich Gas Injection Using Coalbed Methane. *Processes* **2025**, *13*, 1702. [CrossRef]
2. He, J.; Ji, X.; Chen, Z.; Shen, L.; Gui, W. High-Definition Light and Shadow Reconstruction of Blast Furnace Burden Surface Images Based on a Spatiotemporal Multi-Domain Guided Generative Model. *IEEE Trans. Autom. Sci. Eng.* **2025**, *22*, 18243–18254. [CrossRef]
3. Ma, S.H.; Wen, Z.G.; Chen, J.N.; Wen, Z.C. Mode of circular economy in China's iron and steel industry: A case study in Wu'an city. *J. Clean. Prod.* **2014**, *64*, 505–512. [CrossRef]
4. Li, K.; Tian, H. A multiobjective variable neighborhood search with learning and swarm for permutation flowshop scheduling with sequence-dependent setup times. *Processes* **2022**, *10*, 1786. [CrossRef]
5. Sun, W.; Wang, Q.; Zhou, Y.; Wu, J. Material and energy flows of the iron and steel industry: Status quo, challenges and perspectives. *Appl. Energy* **2020**, *268*, 114946. [CrossRef]
6. Jiang, S.L.; He, L.; Cao, L.; Sun, L.; Peng, G. A learning-enhanced ant colony optimization algorithm for integrated planning and scheduling in hot rolling production lines under uncertainty. *Swarm Evol. Comput.* **2025**, *93*, 101851. [CrossRef]
7. Jacobs, L.; Rèche, D.; Bán, A.; Colla, V.; Toscanelli, O.; Raulf, M.; Schlupp, M.; Smeulders, B.; Cook, M.; Filemon, W. Development of Oil-Free Lubricants for Cold Rolling of Low-Carbon Steel. *Processes* **2025**, *13*, 1234. [CrossRef]
8. Wang, Y.; Li, C.; Jin, X.; Xiang, Y.; Li, X. Multi-objective optimization of rolling schedule for tandem cold strip rolling based on NSGA-II. *J. Manuf. Process.* **2020**, *60*, 257–267. [CrossRef]
9. Babajamali, Z.; Aghadavoudi, F.; Farhatnia, F.; Eftekhari, S.A.; Toghraie, D. Pareto multi-objective optimization of tandem cold rolling settings for reductions and inter stand tensions using NSGA-II. *ISA Trans.* **2022**, *130*, 399–408. [CrossRef]
10. Peng, G.; Cheng, Y.; Zhang, Y.; Shao, J.; Wang, H.; Shen, W. Industrial big data-driven mechanical performance prediction for hot-rolling steel using lower upper bound estimation method. *J. Manuf. Syst.* **2022**, *65*, 104–114. [CrossRef]
11. Yang, H.; Du, Y.; Li, Y.; Qian, W.; Hu, B. A Heuristic Mutation Based Genetic Algorithm for Fast Parallel Scheduling of Steel Cold Rolling. *Chin. J. Mech. Eng.* **2025**, *38*, 100. [CrossRef]
12. Jiang, M.; Yu, H.; Chen, J. Improved self-learning genetic algorithm for solving flexible job shop scheduling. *Mathematics* **2023**, *11*, 4700. [CrossRef]
13. Li, J.; Li, H.; He, P.; Xu, L.; He, K.; Liu, S. Flexible job shop scheduling optimization for green manufacturing based on improved multi-objective wolf pack algorithm. *Appl. Sci.* **2023**, *13*, 8535. [CrossRef]
14. Gao, K.; Cao, Z.; Zhang, L.; Chen, Z.; Han, Y.; Pan, Q. A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems. *IEEE/CAA J. Autom. Sin.* **2019**, *6*, 904–916.
15. Shi, J.; Liu, W.; Yang, J. An enhanced multi-objective evolutionary algorithm with reinforcement learning for energy-efficient scheduling in the flexible job shop. *Processes* **2024**, *12*, 1976.
16. Nouiri, M.; Bekrar, A.; Jemai, A.; Niar, S.; Ammari, A.C. An effective and distributed particle swarm optimization algorithm for flexible job-shop scheduling problem. *J. Intell. Manuf.* **2018**, *29*, 603–615. [CrossRef]
17. Jiang, T.; Zhang, C. Application of grey wolf optimization for solving combinatorial problems: Job shop and flexible job shop scheduling cases. *IEEE Access* **2018**, *6*, 26231–26240. [CrossRef]
18. Pan, Z.; Wang, L.; Zheng, J.; Chen, J.F.; Wang, X. A learning-based multipopulation evolutionary optimization for flexible job shop scheduling problem with finite transportation resources. *IEEE Trans. Evol. Comput.* **2022**, *27*, 1590–1603. [CrossRef]
19. Luan, F.; Cai, Z.; Wu, S.; Liu, S.Q.; He, Y. Optimizing the low-carbon flexible job shop scheduling problem with discrete whale optimization algorithm. *Mathematics* **2019**, *7*, 688. [CrossRef]
20. 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]
21. Ishikawa, S.; Kubota, R.; Horio, K. Effective hierarchical optimization by a hierarchical multi-space competitive genetic algorithm for the flexible job-shop scheduling problem. *Expert Syst. Appl.* **2015**, *42*, 9434–9440. [CrossRef]
22. Gao, K.Z.; Suganthan, P.N.; Pan, Q.K.; Chua, T.J.; Chong, C.S.; Cai, T.X. An improved artificial bee colony algorithm for flexible job-shop scheduling problem with fuzzy processing time. *Expert Syst. Appl.* **2016**, *65*, 52–67. [CrossRef]
23. Li, T.; Meng, Y.; Tang, L. Scheduling of continuous annealing with a multi-objective differential evolution algorithm based on deep reinforcement learning. *IEEE Trans. Autom. Sci. Eng.* **2023**, *21*, 1767–1780. [CrossRef]
24. Zhao, S.; Yang, Y.; Guo, Q.; Tang, L. A differential evolution algorithm with double-mode crossover for supply chain scheduling in cold rolling. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016; pp. 1062–1069.
25. Zhao, S.; Yang, Y. Differential evolution algorithm for solving coil scheduling problem in parallel continuous annealing lines. In Proceedings of the 2016 Chinese Control and Decision Conference (CCDC), Yinchuan, China, 28–30 May 2016; pp. 3417–3422.
26. Wang, C.; Chen, J.; Xu, B.; Liu, S. A Discrete Improved Gray Wolf Optimization Algorithm for Dynamic Distributed Flexible Job Shop Scheduling Considering Random Job Arrivals and Machine Breakdowns. *Processes* **2025**, *13*, 1987. [CrossRef]

27. Song, L.; Li, Y.; Xu, J. Dynamic job-shop scheduling based on transformer and deep reinforcement learning. *Processes* **2023**, *11*, 3434. [CrossRef]
28. Xu, Y.; Wang, D.; Zhang, M.; Yang, M.; Liang, C. Quantum particle swarm optimization with chaotic encoding schemes for flexible job-shop scheduling problem. *Swarm Evol. Comput.* **2025**, *93*, 101836. [CrossRef]
29. Sun, L.; Lin, L.; Gen, M.; Li, H. A hybrid cooperative coevolution algorithm for fuzzy flexible job shop scheduling. *IEEE Trans. Fuzzy Syst.* **2019**, *27*, 1008–1022. [CrossRef]
30. Wan, L.; Fu, L.; Li, C.; Li, K. An effective multi-agent-based graph reinforcement learning method for solving flexible job shop scheduling problem. *Eng. Appl. Artif. Intell.* **2025**, *139*, 109557. [CrossRef]
31. Gao, K.Z.; Suganthan, P.N.; Pan, Q.K.; Chua, T.J.; Cai, T.X.; Chong, C.S. Discrete harmony search algorithm for flexible job shop scheduling problem with multiple objectives. *J. Intell. Manuf.* **2016**, *27*, 363–374. [CrossRef]
32. Zhao, L.; Cheng, W.; Meng, L.; Zhang, C.; Ren, Y.; Zhang, B.; Duan, P. MILP modeling and optimization of flexible job shop scheduling problem with preventive maintenance. *Comput. Ind. Eng.* **2025**, *201*, 110861. [CrossRef]
33. Ayuba, E.O.; Bolu, C.A.; John, T.M.; Abioye, A.A. Dataset showing steel cold rolling process parameters for a 6-high cold rolling mill in Nigeria. *Data Brief* **2018**, *18*, 891–901. [CrossRef] [PubMed]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.