Taylor & Francis
Taylor & Francis Group

Engineering Optimization

ISSN: 0305-215X (Print) 1029-0273 (Online) Journal homepage: www.tandfonline.com/journals/geno20

An improved non-dominated sorting genetic algorithm-II with neighbourhood search for the multi-objective flexible job-shop problem considering limited automated guided vehicles

Shiduo Ning, Yuanfei Fang, Yanjun Shi, Kaiyu Zhang & Chengjia Yu

To cite this article: Shiduo Ning, Yuanfei Fang, Yanjun Shi, Kaiyu Zhang & Chengjia Yu (15 Aug 2025): An improved non-dominated sorting genetic algorithm-II with neighbourhood search for the multi-objective flexible job-shop problem considering limited automated guided vehicles, Engineering Optimization, DOI: 10.1080/0305215X.2025.2537241

To link to this article: https://doi.org/10.1080/0305215X.2025.2537241

ENGINEERING OPTIMIZATION
https://doi.org/10.1080/0305215X.2025.2537241

Taylor & Francis
Taylor & Francis Group

An improved non-dominated sorting genetic algorithm-II with neighbourhood search for the multi-objective flexible job-shop problem considering limited automated guided vehicles

Shiduo Ning[a,b], Yuanfei Fang[c], Yanjun Shi [a,b], Kaiyu Zhang[a,b] and Chengjia Yu[a,b]

[a]School of Mechanical Engineering, Dalian University of Technology, Dalian, People's Republic of China; [b]State Key Laboratory of High-Performance Precision Manufacturing, Dalian University of Technology, Dalian, People's Republic of China; [c]The 36th Research Institute of China Electronics Technology Group Corporation, Jiaxing, People's Republic of China

ABSTRACT

The multi-objective flexible job-shop scheduling problem with limited automated guided vehicles (MFJSPLA) focuses on integrating scheduling decisions with constrained automated guided vehicles (AGVs), considering transportation time. As AGVs are energy-intensive, improving their efficiency aids in reducing energy consumption, promoting green manufacturing. To solve this, we propose an improved non-dominated sorting genetic algorithm-II with neighbourhood search (INSGA-II_NS). The method includes a machine selection, operation sequencing and AGV selection representation encoding for mapping scheduling and allocation decisions; multiple heuristic operators to generate high-quality initial solutions and guide local search and a neighbourhood search mechanism within NSGA-II for optimal trade-off convergence. Experimental results show that INSGA-II_NS outperforms conventional multi-objective algorithms, demonstrating the effectiveness of the heuristic operators and neighbourhood search in improving solution quality and efficiency. This methodology is robust and practical for addressing complex scheduling challenges in advanced manufacturing systems.

ARTICLE HISTORY
Received 20 November 2024
Accepted 16 July 2025

KEYWORDS
Multi-objective optimization; AGV delivery; NSGA-II; energy consumption

1. Introduction

In the context of intelligent manufacturing, the importance of logistics in the scheduling of production workshops has become more and more prominent. Automated guided vehicles (AGVs), as an intelligent logistics tool, are widely used in production workshops, bringing great convenience (Özgüven, Özbakır, and Yavuz 2023; Wu, Liu, and Zhao 2021). In actual production, to achieve flexible and free logistics and transportation, multiple AGVs are often used to realize production workshop scheduling, and have replaced manually driven transport vehicles in many situations (Balas 1969). According to the tasks issued by the intelligent production workshop scheduling system, the task allocation and path planning may be carried out through intelligent algorithms under the Internet of Things (Cao *et al*. 2020), and important handling and transportation tasks, including material transportation, parts supply, equipment support, workshop scheduling and inventory management, can be completed (Burmeister, Guericke, and Schryen 2024; Zhou, Tan, Wu, *et al*. 2025). Such systems can achieve safer, standardized and efficient handling of production factors, automated material

CONTACT Yanjun Shi shiyj@dlut.edu.cn

transportation and production process optimization, and reduce labour costs and material losses (Liang *et al.* 2023).

In the scenario of shop floor scheduling, there are multiple machines and multiple production tasks to be completed in the production workshop, and specific workshop scheduling goals are formulated according to the requirements of the production tasks, such as completing all production tasks within a limited time, so as to formulate a scheduling plan that meets the production tasks or even achieves higher efficiency (Z. Li and Chen 2021). The key challenges of shop-floor scheduling problems include determining which machines are used for each task, the machining sequence for each task and the machining time for each task (Cheng *et al.* 2024; H. Wang and Zhu 2023). Factors such as task priority, machine availability and dependencies between tasks are usually taken into account; therefore, based on these factors, shop floor scheduling issues can be divided into different categories (Brandimarte 1993; Chang *et al.* 2023). The flexible job-shop scheduling problem (FJSP) was originally discussed as a job-shop scheduling problem with machine flexibility (Zhang *et al.* 2020). It solves both machine assignment and operation sequencing problems and has proven to be an NP challenge. Production scheduling becomes even more complex if additional AGV selection subproblems are taken into account (Gu, Huang, and Liang 2020; Sze, Salhi, and Wassan 2016).

Considering the FJSP research on multi-AGV transportation, this is a research direction with rich technical value and great social significance under intelligent manufacturing. As shown in Figure 1, this article focuses on research on the process of production and processing in the production and processing process of modern intelligent processing workshops (Yang *et al.* 2023). The combination of a flexible operation workshop and AGV technology helps to enhance the logistics and transportation capacity of the flexible operation workshop, promote the rational allocation of production workshop elements, realize the flexible layout of the production line and the rapid adjustment of production methods, reduce human intervention in the transportation process, reduce errors and accidents, and improve production quality (Chen, Zou, and Wang 2023; Y. Wang and Zhu 2023). The AGV and the production factors in the flexible workshop are coordinated and synchronized through information technology to improve the efficiency of production process management and monitoring.

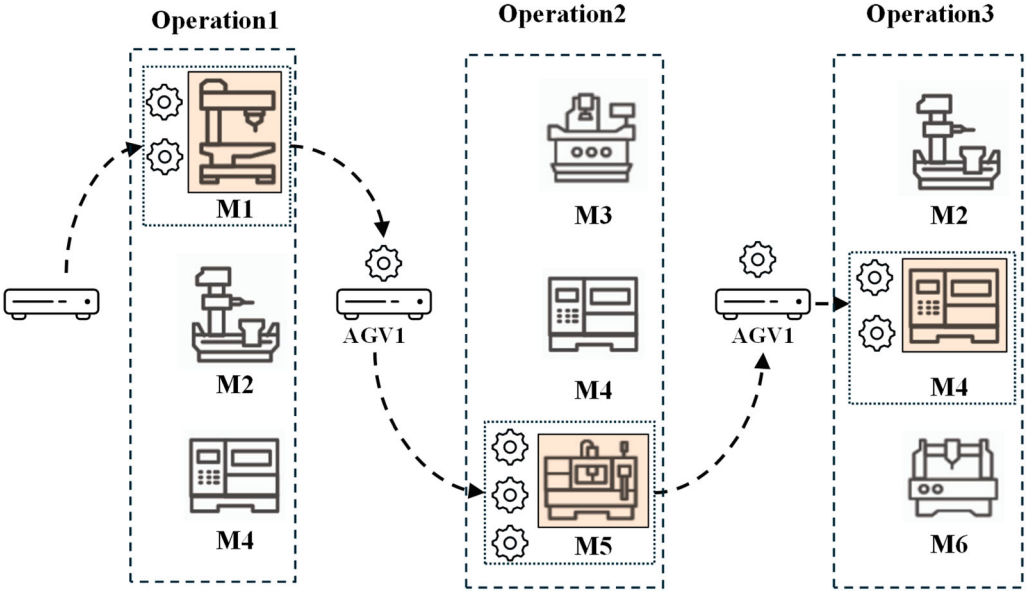

Figure 1. Illustration of integrated production and inventory scheduling problem. M = machine; AGV = automated guided vehicle.

The application of this technology can promote progress in discrete manufacturing technology, promote the development of digitalization, networking, intelligence and low-carbon green development of the manufacturing industry, and create higher economic and social benefits for manufacturing enterprises (Jiménez Tovar *et al*. 2024; Zhou, Tan, Zhao, *et al*. 2024).

This article advances the field through three core contributions. First, it redefines the classical FJSP by integrating AGV coordination and transportation time into a unified multi-objective flexible job-shop scheduling problem considering limited automated guided vehicles (MFJSPLA), explicitly capturing the interdependence between machine scheduling and material handling, which is a critical gap that has been overlooked in prior AGV-related FJSP studies. Secondly, it introduces a novel three-layer integer encoding strategy (operation sequence, machine allocation and AGV routing triple-layer encoding), enabling efficient exploration of the expanded solution space through tailored neighbourhood search operators while preserving problem-specific stochastic characteristics. Thirdly, it prioritizes multi-objective optimization by simultaneously minimizing makespan, energy consumption and logistics costs, reflecting real-world demands for sustainable and cost-effective manufacturing. Experimental validations demonstrate that these innovations collectively enhance solution quality and robustness.

Research on the scheduling problem of flexible job shops under AGV coordination mainly focuses on two aspects: one is the modelling, and the other is the study of optimization algorithms for solving the problem (Ho *et al*. 2024; Johnson, Chen, and Lu 2022). At present, many researchers are also focusing on the job-shop problem of transferring jobs between machines *via* vehicle paths (JSVRP). JSVRP typically focuses on optimizing the transportation routes for vehicles to deliver goods between predefined locations, and the number of AGVs transported is usually unlimited. In contrast, MFJSPLA integrates machines, operation sorting and AGV routing into a unified framework. In the current study, AGVs are not merely transportation tools but also key resources, and their allocation directly affects production schedules and energy consumption. This integration establishes an interdependent relationship between job processing and material processing, and joint optimization of these two dimensions is required.

Several studies have been carried out on modelling and optimization. Sang, Tan, and Liu (2020) proposed a high-dimensional multi-objective green FJSP. They pointed out that achieving green manufacturing involves factors such as carbon emissions, noise and energy consumption, and incorporated them as part of the objectives in the model. They also defined machine idle time as part of the model attributes, enriching the constraints under energy consumption objectives to achieve a more complex model. Fontes and Homayouni (2019) modelled the integration scheduling problem of AGVs and equipment as a mixed-integer linear programming model. This model uses two sets of chained decisions interconnected with each other: one set for machines and another set for AGVs. Yao *et al*. (2024) proposed a new mixed-integer linear programming model, which decomposes the coordinated scheduling problem of AGVs and flexible job shops into four interrelated and dependent subproblems. Sanogo *et al*. (2022) considered the issue of AGV collisions during transportation and proposed a simulation method based on multi-agent systems. Xu, Bao, and Zhang (2023) proposed a multi-objective mixed-integer programming model for the integrated green scheduling problem of flexible job shops and AGVs. The aim is to minimize both makespan (total time) and total energy consumption simultaneously. Gong *et al*. (2024) proposed a green hierarchical integrated scheduling method. At the workshop level, a batch splitting model was introduced to obtain the optimal processing task set for each production cycle with minimal expected costs (start-up costs, tardiness costs and holding costs). At the process unit level, a task allocation model was provided to allocate the optimal workload to each machine tool with minimal processing energy consumption and maximum machine tool load. At the machine tool level, an operation sequencing model was established to obtain the optimal machining sequence for each machine tool with minimal standby energy consumption and manufacturing span.

After completing the modelling work for the problem, it is necessary to use optimization algorithms properly to solve the problem. The selection and implementation of optimization algorithms

will directly affect the efficiency of the solution and the quality of the results. The optimization research for the algorithm part mainly includes the following aspects. Schworm *et al.* (2024) proposed a quantum annealing-based solving algorithm, which combines quantum annealing with classical techniques to solve the FJSP. This algorithm considers the multi-criteria scheduling problem of minimizing completion time, total workload and job priorities. It uses a Hamiltonian formula with Lagrange parameters to integrate the constraints and objectives of the problem. By assigning appropriate weights to the objectives, it allows certain objectives to take precedence over others. Combining multi-criteria problem formulations with an improved iterative method accelerates computation time and uses bottleneck factors to achieve better solutions. Hamida, Azzouz, and Said (2023) proposed an adaptive variable neighbourhood search for the green FJSP with variable processing speeds. This method improves upon the basic variable neighbourhood search algorithm by incorporating an adaptive mechanism that alternates between exploring and exploiting the search space. The objective is to minimize both completion time and total energy consumption. Yu *et al.* (2024) studied the green scheduling of distributed flexible job-shop problems with the objective of minimizing completion time and total energy consumption. They proposed a knowledge-guided dual-population evolutionary algorithm, which includes various types of evolution operator with adaptive strategies to guide the efficient evolution of the dual populations. They also devised energy-saving strategies based on knowledge to further reduce energy consumption. Yang *et al.* (2023) used a decomposition-based memetic algorithm to address the FJSP with fuzzy processing times. They introduced a novel Tchebycheff aggregation method based on a subproblem decomposition strategy to select high-quality individuals and design acceptance criteria. The results indicate that the algorithm achieves satisfactory solutions when solving problems involving fuzzy processing times. Chen, Zou, and Wang (2023) proposed a multi-objective scheduling strategy for a fuzzy flexible job shop with three uncertain factors. They used a hybrid particle swarm optimization with variable neighbourhood search strategy to solve this problem. The process priority encoding method combined with several initialization strategies was used to generate the initial population. Genetic algorithm crossover and mutation were introduced, and the positions and velocities of particles were reconstructed based on particle swarm for the global search. Subsequently, three neighbourhood structures were used to conduct a neighbourhood search for optimal particles to obtain the Pareto feasible solution set. Finally, satisfactory solutions were selected through grey relational analysis (Jing *et al.* 2024).

The above-mentioned research on the integrated scheduling problem involving AGV has been able to accurately describe the modelling work of the problem, and the proposed algorithms can also quickly converge and achieve high-quality solutions (J. Li *et al.* 2021; Saqlain, Ali, and Lee 2023). However, previous research has overlooked the time and energy consumption associated with transportation tasks. Therefore, this article focuses on the scheduling problem of flexible job shops with AGV collaboration, which is called MFJSPLA. By visualizing and controlling transportation tasks, AGVs can undertake the handling tasks efficiently. Transportation time is incorporated as an attribute value into the production factors, aiming to achieve green collaborative scheduling between machines and AGVs. A mathematical model is proposed, which is used to construct constraints and objective functions. To find the optimal solution to this MFJSPLA, an OSMSTS encoding method (incorporating operation, machine and transportation sequences) is proposed to represent a unique allocation of production factors for a production task. Based on the modelling and encoding, the problem is solved using a multi-objective optimization algorithm. Owing to the excessively large solution space of the encoding, which leads to slow convergence of the algorithm, this study proposes the improved non-dominated sorting genetic algorithm-II with neighbourhood search (INSGA-II_NS) to minimize the objective values in the initial solution generation. By obtaining solutions under a multi-objective scenario, the optimization with regard to convergence speed and precision of the algorithm is proved. Validation is provided by comparison with other multi-objective algorithms.

The remainder of this article is structured as follows. Section 2 provides a description of the MFJSPLA, followed by the proposed mathematical model in Section 3. The method for solving the

scheduling problem of flexible job shops using INSGA-II_NS is described in Section 4. Comparative experiments and analysis of the results are presented in Section 5. Section 6 concludes the article and suggests directions for future work.

2. Problem formulation

2.1. Problem description

MFJSPLA is defined as the processing of $|I|$ jobs on $|K|$ machines, with $|V|$ AGVs responsible for transporting jobs between different machines. Each job consists of multiple operations, and each operation can be processed on at least one machine, although the processing time may vary depending on the machine. MFJSPLA incorporates multiple objectives: not only assigning the most suitable machines and transport vehicles for each operation, but also optimizing for minimal total processing time, reduced energy consumption, and controlled production and transportation costs. While adhering to the basic constraints of FJSP, this problem adds the complexity of logistics scheduling to achieve maximum resource efficiency and overall system performance optimization. The following assumptions also need to be considered in MFJSPLA while satisfying the constraints of FJSP.

- The time and energy consumption of all AGVs when they are first called are negligible, assuming that the time and energy consumption of tasks corresponding to new AGV identifiers in the AGV code is 0.
- All jobs are ready at the machines' side when they start processing the first step of the operation, assuming that the time consumed for each initial operation transport task by an AGV is 0.
- Machines and AGVs adopt a shutdown/restart strategy, meaning that machines will be shut down when there are no ongoing processing tasks, to save energy, and restarted when a processing task arrives. Similarly, AGVs will go into standby mode to save energy when there are no transport tasks or travel tasks to destinations, with the standby energy consumption of AGVs being negligible.
- The loading and unloading times of jobs on processing machines and AGVs are ignored.
- All AGVs have the same performance, and transporting jobs does not affect the energy consumption when AGVs are working. The energy consumption for transporting different jobs is the same.

Table 1 and Figure 2 show small-scale examples of this problem.

The notation used in MFJSPLA, energy consumption and transportation resources, and throughout the article, is listed in Table 2.

Table 1. Example of the multi-objective flexible job-shop scheduling problem considering limited automated guided vehicles (MFJSPLA).

		Processing time					
Case	Operation	M_1	M_2	M_3	M_4	M_5	M_6
J_1	$O_{1,1}$	96	–	–	–	–	102
	$O_{1,2}$	–	98	–	137	95	–
	$O_{1,3}$	68	–	–	75	–	84
J_2	$O_{2,1}$	–	79	91	–	83	75
	$O_{2,2}$	152	–	–	134	175	–
J_3	$O_{3,1}$	96	–	83	–	64	74
	$O_{3,2}$	135	117	–	–	–	–
AGV_1	Starting point M_6	57	91	80	43	62	48
AGV_2	Starting point M_5	47	33	35	41	67	62
AGV_3	Starting point M_1	65	75	59	61	47	84
	Starting point M_5	47	86	64	65	71	63

Note: AGV = automated guided vehicle.

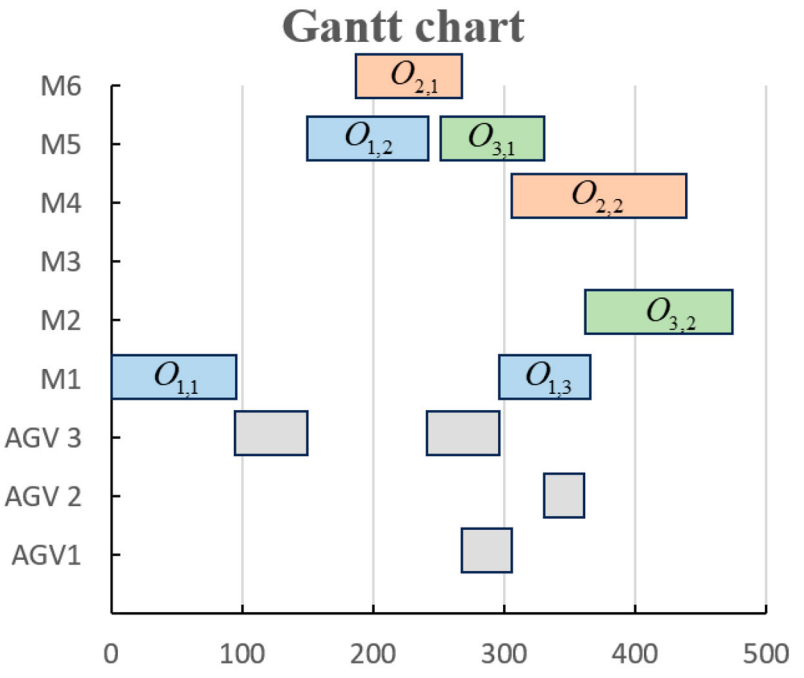

Figure 2. Gantt chart of the example. M = machine; AGV = automated guided vehicle.

3. The proposed mathematical model

In the MFJSPLA, a mathematical model is developed to minimize the makespan, AGV operation times and costs. The mathematical model consists of an objective function and a set of constraints. To define the objective function, energy consumption components within the production workshop are also taken into account.

3.1. Time consumption model in a manufacturing system

In the production task formed by scheduling production elements, the minimum maximum processing time represents the shortest time required to complete the task. To maximize the on-site utilization of AGVs and achieve green and efficient production, this article incorporates the average operation time of AGVs into the time objective function. The total AGV transport time reflects the time required for AGVs to transport jobs between different machines during various operations. The total AGV empty travel time represents the time needed for AGVs to travel from their current location to the job's location before performing the transport task associated with processing on the respective machine. To enhance AGV utilization in the workshop, the average AGV operation time measures the standard deviation of all AGV operation times (including transport and travel time) for the given production task, providing an indicator of time distribution efficiency (Yuan *et al.* 2024).

$$T_{carry} = \sum_{i\in I}\sum_{i\in J_i}\sum_{k\in K_{i,j}}\sum_{k'\in K_{i,j+1}}\sum_{q\in Q} X_{i,j,k}X_{i,j+1,k'}A_{i,j,q}tt_{k,k'} \tag{1}$$

Table 2. Summary of notation.

Notation	Explanation
Indices	
i, i'	Index of jobs, $i, i' = 1, 2, \ldots, N$
j, j'	Index of operations, $j, j' = 1, 2, \ldots, n_i$
k, k'	Index of machines, $k, k' = 1, 2, \ldots, \lvert M\rvert$
q, q'	Index of AGVs, $q, q' = 1, 2, \ldots, \lvert V\rvert$
Parameters	
I	Set of total jobs
K	Set of total machines
J	Set of total operations
V	Set of total AGVs
J_i	Set of operations of job i
$K_{i,j}$	Set of available machines of operation $O_{i,j}$
n_i	Total operation number of job i
$O_{i,j}$	Operation j of job i
$pt_{i,j,k}$	Processing time of $O_{i,j}$ on machine k
$tt_{k,k'}$	Travel time for the AGV from machine k to machine k'
$tt'_{k,k'}$	Empty travel time for the AGV from machine k to machine k'
$p_{i,j,k}$	Processing power of $O_{i,j}$ on machine k
p'_k	Idle power of the machine/standby power of the machine
P_q	Transporting power of the P_0 AGV q
P'_q	Idle power of the AGV q
P_0	Public power, which represents the power consumption of other equipment maintaining the workshop environment
$pc_{i,j,k}$	Processing cost of $O_{i,j}$ on machine k
$pc'_{i,j,k}$	Disposal cost related to waste generated by job $O_{i,j}$ on machine k
$C_{\max}$	Maximum completion time
J_1	An extreme value
Decision variables	
$Y_{i,j,i',j'}$	Equals 1 if $O_{i,j}$ is processed before $O_{i',j'}$, otherwise 0
$Z_{i,j,i',j',q}$	Equals 1 if $O_{i,j}$ is transported before $O_{i',j'}$ on AGV q, otherwise 0
$X_{i,j,k}$	Equals 1 if $O_{i,j}$ is assigning machine k, otherwise 0
$A_{i,j,q}$	Equals 1 if transportation task $O_{i,j}$ between $O_{i,j+1}$ assigning to AGV q, otherwise 0
$W_{i,j,k,q}$	Equals 1 if both $A_{i,j,q}$ and $X_{i,j,k}$ are equal to 1, otherwise 0
$B_{i,j,k}$	Start time of processing $O_{i,j}$ on machine k
$D_{i,j,q}$	Start time of transporting $O_{i,j}$ on AGV q

Note: AGV = automated guided vehicle.

$$T_{idle} = \sum_{i\in I}\sum_{i'\in I}\sum_{j\in J_i}\sum_{j'\in J_{i'}}\sum_{k\in K_{i,j}}\sum_{k'\in K_{i',j'}}\sum_{q\in Q} A_{i,j,q}A_{i',j',q}X_{i,j,k}X_{i',j',k'}Z_{i,j,i',j',q}tt'_{k,k'} \tag{2}$$

$$T_{carry,q} = \sum_{i\in I}\sum_{j\in\{1,\ldots,J_i-1\}}\sum_{k'\in K_{i',j'}}\sum_{k\in K_{i,j}} A_{i,j,q}X_{i,j,k}X_{i,j+1,k'}tt_{k,k'}, \forall q \in Q \tag{3}$$

$$T_{idle,q} = \sum_{i\in I}\sum_{i'\in I}\sum_{j\in J_i}\sum_{j'\in J_{i'}}\sum_{k\in K_{i,j}}\sum_{k'\in K_{i',j'}} A_{i,j,q}A_{i',j',q}X_{i,j,k}X_{i',j',k'}Z_{i,j,i',j',q}tt'_{k,k'} \tag{4}$$

$$T_{trans} = \sigma\left(T_{carry,q} + T_{idle,q}, \frac{T_{carry} + T_{idle}}{V}\right) \tag{5}$$

3.2. Energy consumption model in a manufacturing system

The energy consumption in the flexible operation workshop includes the resources required to ensure smooth production from start to finish, such as energy for workshop lighting, exhaust fans and other auxiliary systems. In this article, this is denoted as the public energy consumption per unit time in the workshop. The total machine energy consumption represents the sum of energy used by all machines during the job operations necessary to complete the production task (Wen *et al.* 2023). Similarly, the total AGV processing energy consumption refers to the energy consumed by all AGVs assigned to

the job operations, while the total AGV transport energy consumption accounts for the energy used by AGVs during the transportation tasks related to those job operations.

$$P_{pub} = P_0 C_{\max} \tag{6}$$

$$E_P = \sum_{i \in I} \sum_{j \in J_i} \sum_{k \in K_{i,j}} X_{i,j,k} pt_{i,j,k} pc_{i,j,k} \tag{7}$$

$$E_Q = T_{trans} P_q \tag{8}$$

$$E'_Q = T_{idle} PQ_q \tag{9}$$

3.3. Cost consumption model in a manufacturing system

The cost objective function comprises both the total production cost and the total waste disposal cost. Given the differing dimensions of costs, normalization is required for accurate comparison. Here, the maximum production cost for a production element allocation in the given production task is multiplied by the number of all job operations, and similar adjustments are made for other cost components. Production cost refers to the consumption of resources necessary to complete the task, including raw materials, tool wear and coolant usage. In line with green manufacturing principles aimed at minimizing environmental pollution from production processes, this study also addresses the waste generated during processing. The total waste disposal cost reflects the expense incurred in handling the waste produced by the task, such as the disposal of cutting chips and the filtration and reuse of coolant. The processing time for machines, representing the time required to perform all job operations, is also considered. Here, $W_{P\max}$ represents the maximum production cost of a production element allocation for the given production task, multiplied by the number of all job operations, and $W_{P\min}$, $W_{idle\max}$, $W_{idle\min}$ are similar.

$$W_{\arg} = \varepsilon_1 \frac{W_P - W_{P\min}}{W_{P\max} - W_{P\min}} + \varepsilon_2 \frac{W_{idle} - W_{idle\min}}{W_{idle\max} - W_{idle\min}}, \varepsilon_1 + \varepsilon_2 = 1, \varepsilon_1, \varepsilon_2 \in \mathbb{R} \tag{10}$$

$$W_P = \sum_{i \in I} \sum_{j \in J_i} \sum_{k \in K_{i,j}} X_{i,j,k} pc_{i,j,k} \tag{11}$$

$$W_{idle} = \sum_{i \in I} \sum_{k \in K_{i,j}} \sum_{j \in J} X_{i,j,k} pc'_{i,j,k} \tag{12}$$

According to the problem description, the scheduling optimization objectives are to minimize the maximum completion time of all operations (*i.e.* the makespan) and the total energy consumption. Therefore, the mathematical model of MFJSPLA requiring the minimization of the makespan and energy consumption is formulated as follows:

$$\min f_1 = \alpha_1 C_{\max} + \alpha_2 T_{trans}, \alpha_1 + \alpha_2 = 1, \alpha_1, \alpha_2 \in \mathbb{R} \tag{13}$$

$$\min f_2 = \beta_1 P_{pub} + \beta_2 E_P + \beta_3 E_Q + \beta_4 E'_Q, \beta_1 + \beta_2 + \beta_3 + \beta_4 = 1, \beta_1, \beta_2, \beta_3, \beta_4 \in \mathbb{R} \tag{14}$$

$$\min f_3 = \varepsilon_1 \frac{W_P - W_{P\min}}{W_{P\max} - W_{P\min}} + \varepsilon_2 \frac{W_{idle} - W_{idle\min}}{W_{idle\max} - W_{idle\min}}, \varepsilon_1 + \varepsilon_2 = 1, \varepsilon_1, \varepsilon_2 \in \mathbb{R} \tag{15}$$

subject to:

$$\sum_{k \in K_{i,j}} X_{i,j,k} = 1, \forall i \in I, j \in J_i \tag{16}$$

$$\sum_{q\in Q} A_{i,j,q} = 1, \forall i \in I, j \in \{1,\ldots,n_i - 1\} \tag{17}$$

$$\sum_{k\in K_{i,j}} B_{i,j,k} + \sum_{k\in M_{i,j}} X_{i,j,k} pt_{i,j,k} \leq \sum_{k\in K_{i,j+1}} B_{i,j+1,k}, \forall i \in I, j \in \{1,\ldots,n_i - 1\} \tag{18}$$

$$\sum_{k\in K_{i,j}} B_{i,j,k} + \sum_{k\in M_{i,j}} X_{i,j,k} pt_{i,j,k} \leq \sum_{q\in Q} D_{i,j,q}, \forall i \in I, j \in \{1,\ldots,n_i - 1\} \tag{19}$$

$$\sum_{q\in Q} D_{i,j,q} + \sum_{k\in K_{i,j}} \sum_{k'\in K_{i,j+1}} \sum_{q\in Q} X_{i,j,k} X_{i,j+1,k'} A_{i,j,q} tt_{k,k'} \leq \sum_{k\in K_{i,j}} B_{i,j+1,k},$$
$$\forall i \in I, j \in \{1,\ldots,n_i - 1\} \tag{20}$$

$$\sum_{k\in K_{i,n_i}} B_{i,n_i,k} + \sum_{k\in M_{i,j}} X_{i,n_i,k} pt_{i,n_i,k} \leq C_{\max}, \forall i \in I \tag{21}$$

$$B_{i.j.k} + pt_{i,j,k} \leq B_{i',j',k} + M(3 - Y_{i,j,i',j'} - X_{i',j',k} - X_{i,j,k}),$$
$$\forall i \in I, i' \in I, j \in J_i, j \in J_{i'}, k \in K_{i,j} \cap K_{i',j'} \tag{22}$$

$$B_{i',j',k} + pt_{i',j',k} \leq B_{i.j.k} + M(2 + Y_{i,j,i',j'} - X_{i',j',k} - X_{i,j,k}),$$
$$\forall i \in I, i' \in I, j \in J_i, j \in J_{i'}, k \in K_{i,j} \cap K_{i',j'} \tag{23}$$

$$D_{i,j,q} + \sum_{k\in K_{i,j}} \sum_{k'\in K_{i',j'}} tt_{k,k'} X_{i,j,k} X_{i',j',k'} \leq D_{i',j',q} + M(3 - Z_{i,j,i',j',q} - A_{i,'j',q} - A_{i,j,q}),$$
$$\forall i \in I, i' \in I, i < i', j \in J_i, j' \in J_{i'}, q \in Q \tag{24}$$

$$D_{i',j',q} + \sum_{k\in M_{i,j}} \sum_{k'\in M_{i',j'}} tt_{k,k'} X_{i,j,k} X_{i',j',k'} \leq D_{i,j,q} + M(2 + Z_{i,j,i',j',q} - A_{i,'j',q} - A_{i,j,q}),$$
$$\forall i \in I, i' \in I, i < i', j \in J_i, j' \in J_{i'}, q \in Q \tag{25}$$

$$B_{i,j,k} \geq 0, \forall i \in I, j \in J, k \in K_{i,j} \tag{26}$$

$$B_{i,j,k} \leq M X_{i,j,k}, \forall i \in I, j \in J, k \in K_{i,j} \tag{27}$$

$$D_{i,j,q} \geq 0, \forall i \in I, j \in \{1,\ldots,J_i - 1\}, q \in V \tag{28}$$

$$D_{i,j,q} \leq M A_{i,j,q}, \forall i \in I, j \in \{1,\ldots,J_i - 1\}, q \in V \tag{29}$$

Equations (13)–(15) show three objective functions, comprising production/transport time, total energy, and total waste disposal cost. Constraints (16) and (17) determine that that each operation is exclusively processed on a single machine and transported by only one AGV, thereby preventing simultaneous allocation to multiple resources. Constraint (18) ensures that the processing and transportation of each operation are completed before the subsequent operation in the job sequence starts, maintaining the temporal order of operations. Constraint (19) is similar to Constraint (18); it ensures that after the processing and transportation of an operation, the start time of the subsequent operation respects the scheduled timeline, preventing overlapping within the job. Constraint (20) ensures that the total transportation time assigned to each AGV does not exceed its capacity, preventing overloading of AGV resources. Constraint (21) limits the total processing time on each machine to ensure that the workload on a machine remains within its available capacity. Constraints (22) and (23) ensure that two operations do not overlap on the same machine. Specifically, these two constraints enforce that the start and end times of any two processes meet the non-overlapping condition (*i.e.* the completion time of the previous operation must be less than or equal to the start time of the subsequent operation). Constraints (24) and (25) guarantee the same AGV performing only a single transportation

task at the same time; these paired constraints, often referred to as dual constraints, collectively maintain the proper sequencing of operations and transportation tasks, preventing resource conflicts and ensuring the smooth execution of the production schedule. Constraints (26) and (28) ensures that the start times for both machine processing and AGV transportation are non-negative, reflecting realistic scheduling. Constraint (27) enforces an upper limit on the start time of operations on machines, ensuring that all tasks begin within the allowable production schedule. Constraint (29) limits the start time for AGV transportation tasks, ensuring that all transport operations begin within the required timeframe to maintain an efficient production flow.

4. The proposed INSGA-II_NS

Based on the preceding description of the algorithm , the entire algorithm flow is illustrated in Figure 3.

In the first stage of the proposed INSGA-II_NS, population initialization is achieved through a hybrid approach combining random coding and heuristic rule-based coding. Specifically, for the machine code segment, three heuristic rules are designed to prioritize three distinct objectives: minimizing makespan, energy consumption and logistics costs. These rules generate high-quality initial solutions by strategically assigning machines to operations based on processing times, energy efficiency and cost metrics. For a given process code (operation sequence) and derived machine code (machine assignments), the AGV code (transportation path) is generated by selecting the AGV with the shortest combined travel and handling time from available options, ensuring efficient material

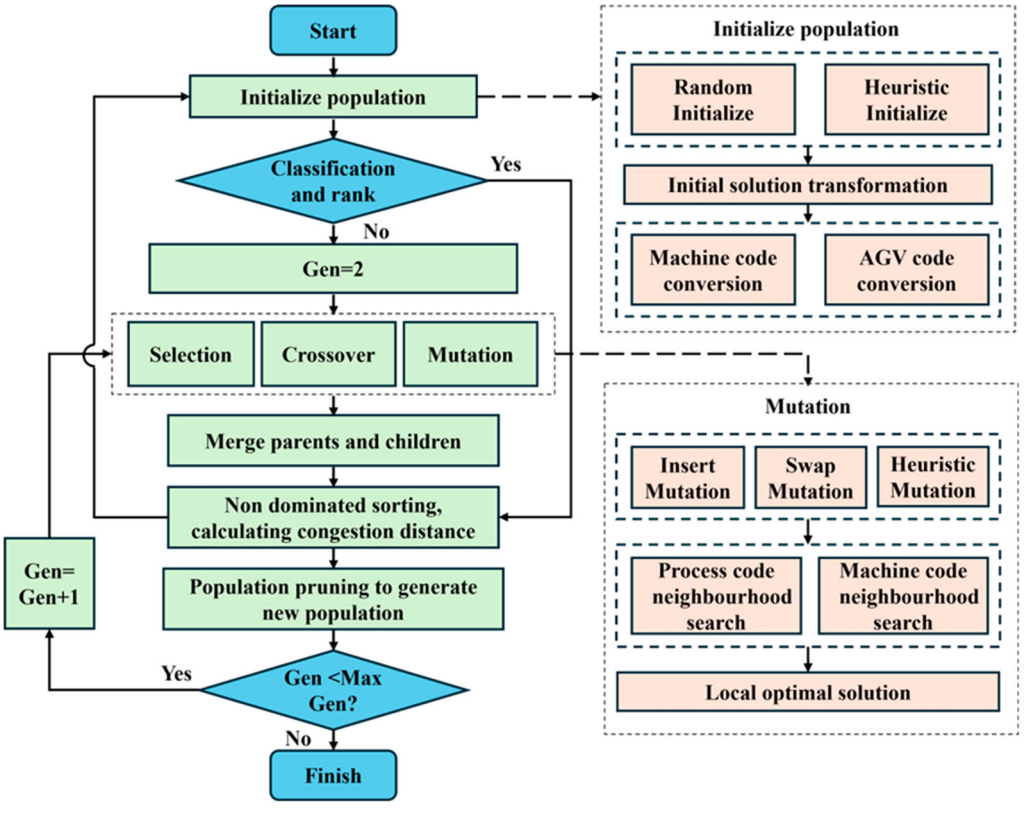

Figure 3. Flowchart of the proposed improved non-dominated sorting genetic algorithm-II with neighbourhood search (INSGA-II_NS). AGV = automated guided vehicle.

handling. Each initialized individual is represented as a set of decision variables (operation sequence, machine allocation, AGV routing), and the initial solution is transformed through machine code transformation and AGV code transformation to meet the problem constraints. This hybrid initialization strategy balances exploration (*via* random coding) and exploitation (*via* heuristic rules) to produce diverse, high-quality initial solutions, thereby enhancing the algorithm's convergence towards optimal Pareto fronts.

Then, the evaluated individuals are selected, the next generation of populations is selected from the current population according to the non-dominated sorting and congestion distance, and the individuals with higher grades and larger congestion distances are generally selected (Durst, Jia, and Li 2023; Ren *et al.* 2022). Crossover and mutation operations are then performed to perform crossover and mutation on selected individuals to generate the next-generation population. The mutation operation includes three methods: insertion, swap and heuristic, and the local optimal solution is obtained by searching the neighbourhood of the process code and the machine code. The process code forms a variety of neighbourhood structures by combining the exchange operation and the insertion operation, and the machine code forms a variety of neighbourhood structures by combining the random operation and the heuristic operation. The optimal solution set of the problem is effectively searched by repeating the steps of fitness assessment, non-dominant sorting, calculation of crowding distance, selection operation, crossover and variation (He *et al.* 2022; Meng *et al.* 2023).

4.1. Solution representation

Traditional job-shop scheduling problems only consider the processing sequence constraints between different operations, and generally employ a process-based encoding method. However, FJSPs not only require arranging the processing sequence of each operation, but also need to assign suitable processing machines for each operation. In the case of MFJSPLA, it is necessary to append coding related to the transportation AGVs alongside the process and machine coding, expanding the coding structure to three layers; thus, the encoding is called OSMSTS (Wu, Liu, and Zhao 2021). The OSMSTS encoding consists of three parts:

- **Operation sequence (OS)**: The OS is used to determine the sequence of all operations (O_{11}, O_{12}, O_{13}, $O_{2,1}$, *etc.*) for each job in the production process, when calculated from left to right; the frequency of the occurrence of the coded number determines the position of the processing sequence of a particular job, and different coding methods can represent different operation paths. The formula $length_{OS} = \sum_{i \in I} n_i$ indicates that the gene length of each process is equal to the sum of all the operations, where n_i represents the number of operation steps per job, reflecting the overall complexity of the action selection process.
- **Machine sequence (MS)**: The MS is used to assign a specific machine to each operation of the job, and the position of the coded number determines the operation in which the job is processed. Different operations have different lists of available machines, *e.g.* the list of available machines for operation $O_{1,1}$ includes machines m_1 and m_6, and the number in the corresponding position of the MS coding block indicates the machine number selected from K_{11}. The formula $length_{MS} = \sum_{i \in I} n_i$ means that the gene length of each machine is equal to the sum of all job processes.
- **Transportation (AGV) sequence (TS)**: The TS is used to assign a transportation path to a specific AGV; the position of the coded number determines the transport object and path of the AGV, and the number at the corresponding position of the TS coded block indicates the number of the AGV. For example, as shown in Figure 4, AGV_3 transports job J_2 from machine m_4 to machine m_5. The equation $length_{TS} = \sum_{i \in I} (n_i - 1)$ indicates that the gene length of the transport path is equal to the sum of the number of operations minus the number of jobs.

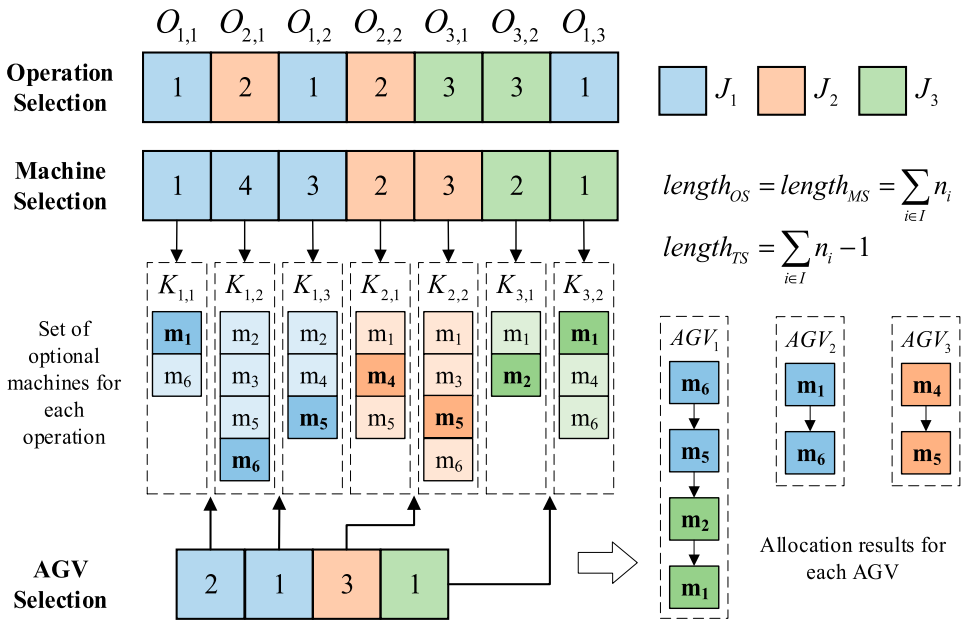

Figure 4. An example of three-layer integer encoding. AGV = automated guided vehicle.

Based on the assumption that all first-step operations for each job have already been placed at the corresponding processing machines at the start of the production task, thus requiring no transportation, it follows that for each job, the AGV encoding will have one less digit compared to the machine encoding, as shown in the equation $length_{TS} = \sum_{i \in I} (n_i - 1)$. Based on the assumption that there is no difference in performance between AGVs, the set of AGVs available for each job operation is the complete set of AGVs. The distribution result of each AGV can be obtained through the above coding process. For example, the transport path of AGV_1 is to transport the job J_1 from machine m_6 to machine m_5, and then transport job J_3 taken from machine m_2 to machine m_1 to be processed.

In this article, the classical semi-active decoding strategy is applied to the two-vector encoding scheme. Semi-active decoding sequentially visits an OS vector to find the operation to be assigned, and then identifies the processing configuration by the MS vector. After that, it is necessary to calculate the starting time and completion time for the operation to obtain operation $O_{i,j}$, which depends on its job predecessor operation $JP_{O_{i,j}}$, machine predecessor operation $MP_{O_{i,j}}$ and AGV predecessor operation $AP_{O_{i,j}}$, completion times of which are denoted as $C_{JP_{O_{i,j}}}$, $C_{MP_{O_{i,j}}}$ and $C_{AP_{O_{i,j}}}$, respectively. Pseudo-codes of the semi-active decoding strategy are given in Algorithm 1, where $CS_p(X)$, $MS_p(X)$ and $TS_p(X)$, respectively, represent the p^{th} elements in the CS vector, MS vector and TS vector of solution X.

4.2. Initialization

In a multi-objective optimization algorithm, the population initialization process, as shown in Figure 5, determines the initial iterative solutions' position distribution and overall quality in the solution space. The position distribution determines the size of the search domain in the subsequent iteration process. For a multi-objective problem, owing to its large solution space, using widely distributed initial solutions can achieve better globality, preventing solutions from converging to local optima. On the other hand, overall quality refers to the ability of the generated solutions to obtain smaller

Algorithm 1. Semi-active decoding for the two-vector encoding scheme.

Input: OS, MS and TS vectors of solution X
Output: A feasible schedule for solution X represented by the CS, MS and TS vectors

for $p = 1$ *to* $\sum_{i}^{n} N_i$ **do**
  $i = OS_p(X)$
  This is jth time that i appears in the OS vector
  $\Rightarrow O_{i,j}$ is the operation to be scheduled
  if $j = 1$ **then**
  $C_{JP_{O_{i,j}}} = 0$
  else
  $C_{JP_{O_{i,j}}} = C_{O_{i,j-1}}$
  Check the position of $O_{i,j}$ in the TS vector
    $\Rightarrow$The transportation task between $O_{i,j}$ and $O_{i,j-1}$ is assigned to A_q
    $\Rightarrow$Identify the last operation on A_q as $AP_{O_{i,j}}$
    end
    Check the position of $O_{i,j}$ in the MS vector
    $\Rightarrow O_{i,j}$ is assigned to be processed by M_k as $M_{O_{i,j}}$
    $\Rightarrow$Identify the last operation on M_k as $MP_{O_{i,j}}$
    if $j > 1$ **and** $MP_{O_{i,j}} != MP_{O_{i,j-1}}$ **then**
    $\Rightarrow$Identify $JP_{O_{i,j}}$ is processed by $M_{k'}$
    $\Rightarrow$Identify AGV position as $M_{k''}$
    $B_{i,j} = \max C_{JP_{O_{i,j}}} + t_{kk'}, C_{AP_{O_{i,j}}} + t_{kk'} + t_{k'k''}, C_{MP_{O_{i,j}}}$
    else
    $B_{i,j} =$
    $C_{i,j} = B_{i,j} + pt_{i,j,k}$
  End

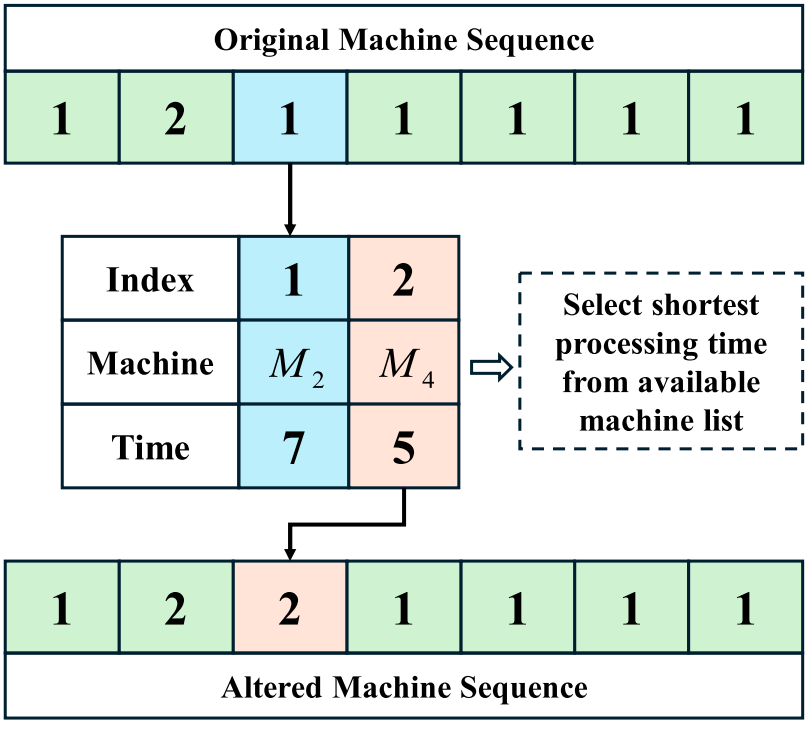

Figure 5. An example of the heuristic rule for machine code population initialization.

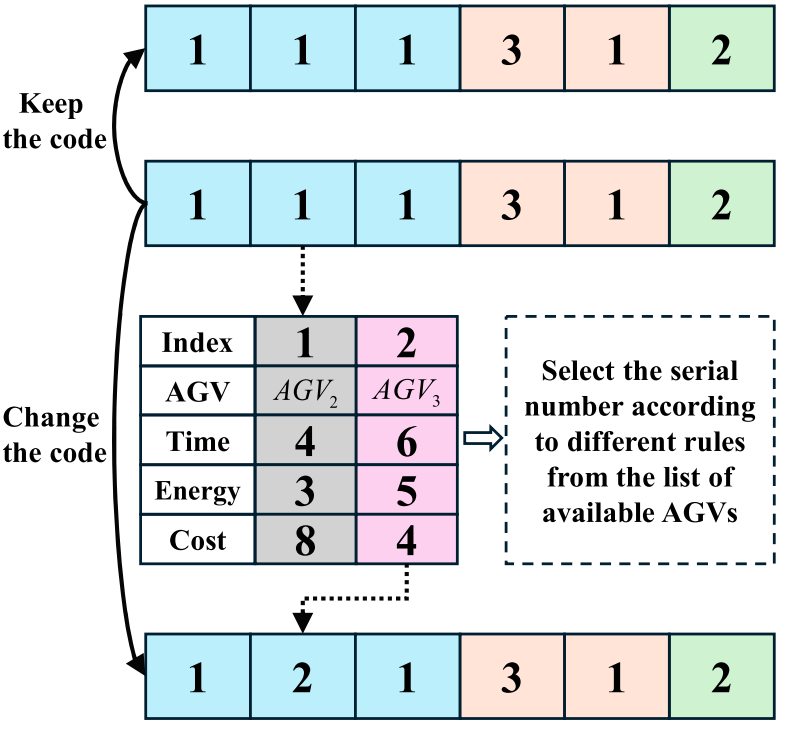

Figure 6. An example of the transport code neighbourhood operation. AGV = automated guided vehicle.

objective function values or smaller function values for certain indicators, guiding the algorithm towards iterative reduction in the direction of the objective function.

Therefore, the initialization of the population needs to satisfy a certain level of globality while ensuring a certain level of quality. In this study, a mixed approach of random encoding and heuristic rule encoding is adopted for population initialization.

In previous studies, for a given set of operation codes, the machine code was generated in ascending order based on the processing time of the available machines for that operation. The machine with the shortest processing time was selected as the machine for the operation. Similarly, in the generation of initial solutions during the iteration of multi-objective algorithms where there is more than one objective function based on the model and objectives established earlier, this article uses three heuristic rules for population initialization: namely, selecting individually according to the shortest processing time, the minimum energy consumption, and the minimum cost.

The solutions generated by these heuristic rules are relatively limited in number and lack a certain level of randomness. Although the random generation of initial solutions can ensure randomness, its dominance in the overall solution space prohibits providing an impetus for the algorithm to iterate towards optimization. Therefore, it is necessary to perform neighbourhood operations on the solutions obtained from heuristic rules to obtain more high-quality suboptimal solutions.

For a set of solutions generated by heuristic rules, the transport codes undergo neighbourhood transformations on one or even multiple operations based on the number of available AGVs for each operation, as shown in Figure 6. There is a certain probability that the transport code will remain the same, and it may also undergo neighborhood operations according to three different rules.

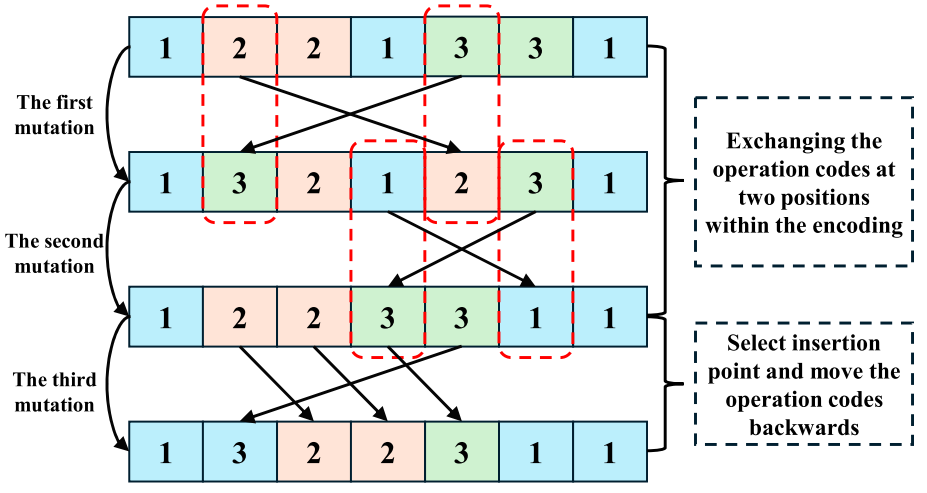

Figure 7. An example of operation code swap and insert operation.

4.3. Local search operators

Excellent population initialization enables the algorithm to start iterating towards global optimal solutions from a good solution space, while the iteration's crossover and mutation processes determine the speed and quality of convergence. The multiple neighbourhood structures adopted in this article can effectively improve the search efficiency and obtain the global optimal solution more quickly.

4.3.1. Operation code neighbourhood search

The operation code uses a combination of swap and insert operations to form multiple neighbourhood searches. The swap operation involves exchanging the operation codes at two positions within the encoding. This process is performed n times during each mutation, where $n, n \in [1, \text{len(Code)}]$, balancing global and local search capabilities. The insert operation involves selecting an operation code position and choosing another position as the insertion point for that code. All operation codes after the insertion point are shifted back by one position. This operation, based on the randomness of the relative distance between the two randomly selected positions, also balances global and local search capabilities. An example of two swap operations (first and second mutations) and one insertion operation that occur during a mutation is shown in Figure 7.

5. Computational experiments and results

This study employed a systematic experimental set-up using Python 3.8 on a personal computer with an 11th Gen Intel Core i7-11800H processor (2.4 GHz) and 32 GB RAM. Ten representative MK-AGV1 test cases (MK01-LA to MK10-LA) with varying job counts (n = 20–50) and machines (m = 5–15) were selected to evaluate the performance of INSGA-II_NS. Parameters (population size $N_g = 50$, offspring size $N_s = N_g$, crossover $P_c = 0.8$, mutation $P_m = 0.35$) were optimized using Taguchi orthogonal arrays (L16) to balance efficiency and convergence. INSGA-II_NS was compared against the indicator-based evolutionary algorithm (IBEA) and strength Pareto evolutionary algorithm (SPEA) under identical encoding, decoding and termination criteria (500 iterations), with three metrics, namely, the dominance ratio ($C(A, B)$), inverted generational distance (IGD) and Spread (SP), quantifying the quality of the Pareto front. Each algorithm was run 20 times per case with fixed random seeds for reproducibility, and statistical significance was validated using the

Table 3. Parameter settings.

Parameter name	Parameter value	Parameter name	Parameter value
α_1	0.9	β_3	0.1
α_2	0.1	β_4	0.05
β_1	0.05	ε_1	0.5
β_2	0.8	ε_2	0.5

Table 4. Parameter results.

	Factor level			
Parameter	1	2	3	4
N_g	50	100	150	200
N_s	$0.6N_g$	$0.8N_g$	N_g	$1.2N_g$
P_c	0.75	0.8	0.85	0.9
P_m	0.2	0.25	0.3	0.35

Table 5. Parameter combinations.

	Factor level					Factor level			
Test	N_g	N_s	P_c	P_m		N_g	N_s	P_c	P_m
1	1	1	1	1	9	3	1	3	4
2	1	2	2	2	10	3	2	4	3
3	1	3	3	3	11	3	3	1	2
4	1	4	4	4	12	3	4	2	1
5	2	1	2	3	13	4	1	4	2
6	2	2	1	4	14	4	2	3	1
7	2	3	4	1	15	4	3	2	4
8	2	4	3	2	16	4	4	1	3

Wilcoxon signed-rank test. This design ensured robust validation of scalability, efficiency and real-world applicability for AGV-integrated flexible job-shop scheduling. In the experiment, the handling time of the AGV was calculated as follows:

$$Tij = \gamma \cdot \frac{Pi + Pj}{2 \cdot P\max} \tag{30}$$

where Tij is the handling time of the process from machine tool Mi to machine tool Mj, Pi is the processing time of the workpiece on machine tool i, Pj is the processing time of the workpiece on machine tool i, $P\max$ is the maximum processing time of the workpiece, and γ is the handling efficiency coefficient, and takes the value [0,1].

5.1. Data generation

The parameter values for the objective functions in this experiment are set as shown in Table 3.

To achieve optimal performance of the proposed algorithm, parameter selection for INSGA-II_NS is crucial. INSGA-II_NS comprises four main parameters: population size N_g, offspring population size N_s, crossover probability P_c and mutation probability P_m. Table 4 shows the factor levels of every parameter in settings for INSGA-II_NS.

Using an orthogonal array to determine the parameter combinations, the parameter combinations are as shown in Table 5.

5.2. Performance measurement

To compare the performance of multi-objective algorithms, the algorithms are evaluated using three

metrics: the IGD, SP and S-metric (Hypervolume). In Equation (32), $C(A, B)$ is the proportion of solutions in set B that are dominated by at least one solution in set A. A larger value of $C(A, B)$ indicates that solution set A is superior to solution set B:

$$C(A,B) = \frac{|\{b \in B\}|, \exists a \in A, a \prec b}{|B|} \tag{31}$$

The IGD is the average distance of each point in the optimal reference set to the solution set. IGD is calculated using Equation (32), where $dist(i, PF)$ is the minimum Euclidean distance from the ith solution in the optimal reference set to the solution set PF, and PF^* is the number of solutions in the optimal reference set. A smaller IGD value indicates better overall performance of the algorithm.

$$\text{IGD} = \frac{\sum_i^{|PF^*|} dist(i, PF)}{|PF^*|} \tag{32}$$

SP can reflect evenness of the Pareto-optimal set A obtained by the algorithm, and measures the standard deviation of the minimum distance from each solution in A to the other solutions. A smaller SP is more favourable. SP is given by:

$$\text{SP} = \sqrt{\frac{1}{|A|}\sum_{i=1}^{|A|}(d_i - \bar{d})^2} \tag{33}$$

where d_i denotes the minimum Euclidean distance from the ith solution in A to the other solutions, and $\bar{d}$ represents the mean value of all d_i.

The S-metric shows the extent of the obtained non-dominated front by calculating the extreme function values of the solutions in the non-dominated set ndA, which is obtained by getting all non-dominated solutions that are not dominated by solutions in the true Pareto front P from the obtained Pareto-optimal set A. If no non-dominated solution is in ndA, the S-metric is assigned the value of -1. The closer the S-metric is to 1, the more area of the true Pareto front is covered by the obtained Pareto front.

$$S(A) = \sqrt{\frac{1}{M}\sum_{k=1}^{M}\left[\frac{f_k^{ndA,\max} - f_k^{ndA,\min}}{f_k^{P,\max} - f_k^{P,\min}}\right]^2} \tag{34}$$

where M is the number of objective functions, $f_k^{ndA,\max}$ and $f_k^{ndA,\min}$, respectively, represent the maximum and minimum of the kth objective in A, and $f_k^{P,\max}$ and $f_k^{P,\min}$, respectively, represent the maximum and minimum of the kth objective in the true Pareto-optimal set P.

5.3. Results

Under various combinations of parameters, INSGA-II_NS is independently run 10 times on the MK-AGV1 case with four AGVs, each for 500 iterations. The results are evaluated using the IGD metric to assess the parameter combinations, and the impact of each parameter on the algorithm's performance is plotted.

Based on Figure 8, the optimal parameter settings are: $N_g = 50$, $N_s = N_g$, $P_c = 0.8$ and $P_m = 0.35$. The following experiments were performed with these optimal parameter settings.

To verify the effectiveness of population initialization and mutation with multiple neighbourhood structures in the algorithm, variants of INSGA-II_NS were constructed for comparison. These variants were labelled INSGA-II_NS-α, INSGA-II_NS-β and INSGA-II_NS-γ . Here, INSGA-II_NS-α considers population initialization optimization but does not have multiple neighbourhood mutation, INSGA-II_NS-γ does not consider population initialization optimization but takes into

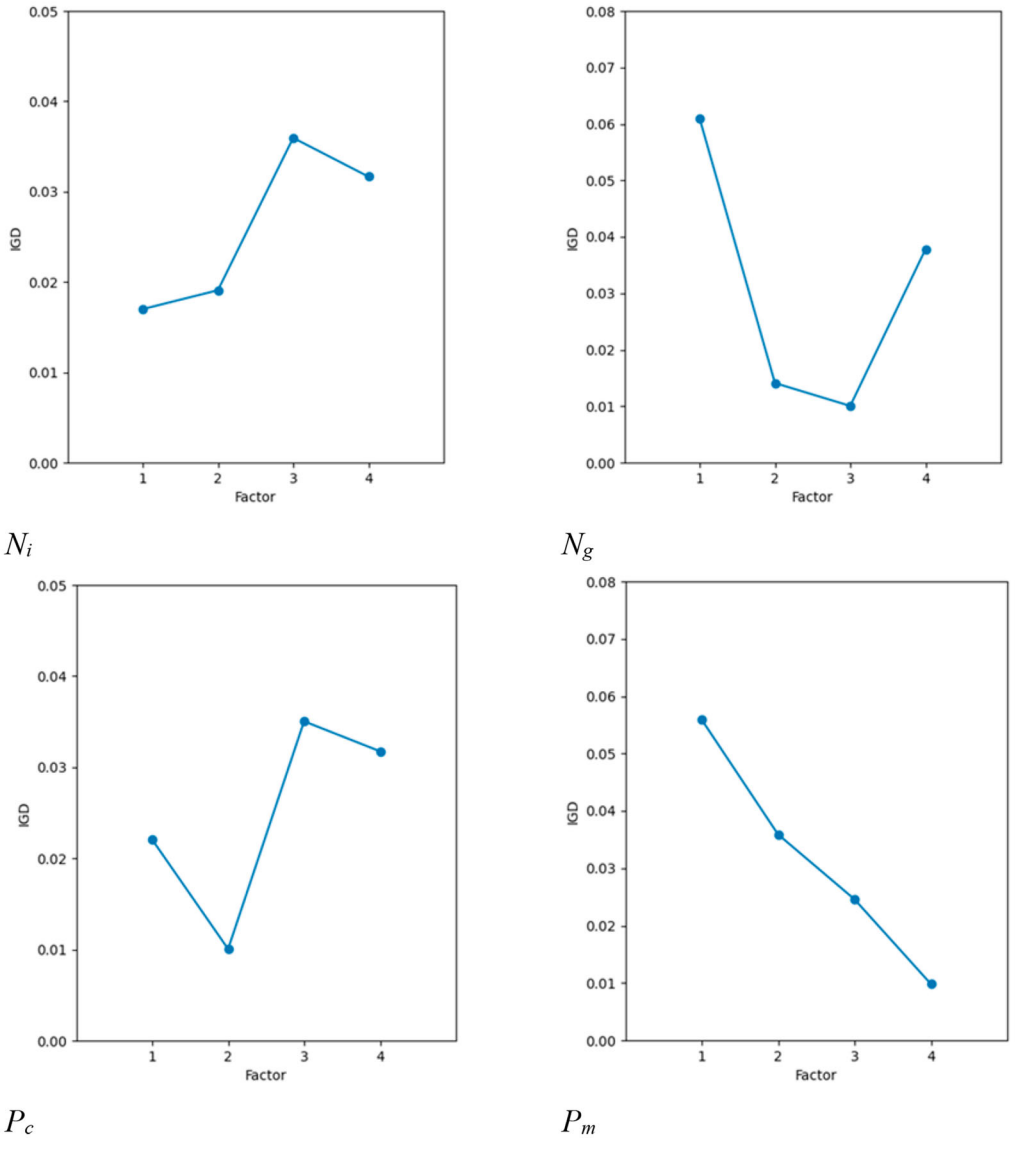

Figure 8. Factor-level comparison.

account multiple neighbourhood mutation, and INSGA-II_NS-β is the prototype algorithm of the non-dominated sorting genetic algorithm-II (NSGA-II).

Based on the analysis of Table 6, it can be observed that INSGA-II_NS exhibits significant advantages over the various mutation algorithms. In comparisons between INSGA-II_NS and its variants INSGA-II_NS-α and INSGA-II_NS-γ, the values for $C(A, B)$ are generally greater than $C(B, A)$, indicating the dominant position of the proposed optimized algorithm, INSGA-II_NS. However, when comparing with variant INSGA-II_NS-γ, there are cases where the value of $C(A, B)$ is lower than $C(B, A)$. When INSGA-II_NS is compared with the algorithm prototype INSGA-II_NS-β, the $C(A, B)$ of INSGA-II_NS is overwhelmingly greater than the $C(B, A)$ of INSGA-II_NS-β, which indicates that population initialization optimization and multiple neighbourhood mutation are necessary. In the instances where $C(A, B)$ is greater than $C(B, A)$, the ratio $\frac{C(A,B)}{C(B,A)}$ is relatively large, while in the

Table 6. Experimental results and comparison of the proposed algorithm and its variants.

Instance	A: INSGA-II_NS $C(A,B)$	B: INSGA-II_NS-α $C(B,A)$	A: INSGA-II_NS $C(A,B)$	B: INSGA-II_NS-β $C(B,A)$	A: INSGA-II_NS $C(A,B)$	B: INSGA-II_NS-γ $C(B,A)$
MK01-LA	0.86	0.48	0.86	0.48	0.60	0.44
MK02-LA	0.46	0.58	0.64	0.64	0.80	0.30
MK03-LA	0.74	0.04	1.00	0.00	1.00	0.00
MK04-LA	0.64	0.32	1.00	0.00	0.98	0.24
MK05-LA	0.68	0.00	1.00	0.00	1.00	0.00
MK06-LA	0.74	0.14	1.00	0.00	1.00	0.00
MK07-LA	0.56	0.48	1.00	0.00	0.94	0.18
MK08-LA	0.66	0.00	0.56	0.08	0.88	0.06
MK09-LA	0.76	0.08	0.26	0.00	0.86	0.00
MK10-LA	0.84	0.08	0.52	0.16	0.82	0.06

Note: INSGA-II_NS = improved non-dominated sorting genetic algorithm-II with neighbourhood search; INSGA-II_NS-α = INSGA-II_NS with population initialization optimization but without multiple neighbourhood mutation; INSGA-II_NS-β = prototype algorithm of NSGA-II; INSGA-II_NS-γ = INSGA-II_NS without population initialization optimization but with multiple neighbourhood mutation.

cases where $C(A,B)$ is lower than $C(B,A)$, the ratio $\frac{C(B,A)}{C(A,B)}$ is not greater than the ratio $\frac{C(A,B)}{C(B,A)}$ seen in the $C(A,B)$ cases. This indicates that population initialization optimization and multiple neighbourhood mutation improve the performance of the algorithm much more than the negative impact on the algorithm occasionally.

In the comparison between INSGA-II_NS and INSGA-II_NS-α, only in case MK02-LA, the effect of INSGA-II_NS-α is slightly better than INSGA-II_NS, and the other INSGA-II_NS variants have better effects, especially in the examples MK03-LA, MK05-LA and MK08-LA, where the proposed INSGA-II_NS is overwhelmingly better than INSGA-II_NS-α. In the comparison between INSGA-II_NS and INSGA-II_NS-β, except for the MK02-LA example, the performance of the two algorithms is equal, and the performance of INSGA-II_NS in all the other studies is better than that of INSGA-II_NS-β, and it shows absolute suppression in 50% of the examples. In the third set of experimental INSGA-II_NS compared with INSGA-II_NS-γ, the proposed algorithm performs very well in all cases, especially in large-scale cases.

INSGA-II_NS-β, which is the prototype of NSGA-II, has the lowest dominant position compared to several other mutation and optimization algorithms. INSGA-II_NS-α introduces population initialization optimization compared to the prototype, which improves the quality of initial solutions and contributes to the stability of solution generation during iterations, although it does not guarantee solution quality. On the other hand, INSGA-II_NS-γ introduces optimization for multiple neighbourhood search mutations compared to the prototype, ensuring both global and local search capabilities during iteration and providing a directional heuristic neighbourhood search based on a certain number of iterations, thereby improving the quality of iterative solutions. INSGA-II_NS takes into account the stability of understanding generation and global search ability by introducing population initialization optimization and multiple neighbourhood mutation.

The evaluation metrics examine the quality of the iterative solutions at the end of iteration, and since INSGA-II_NS-γ is also an optimization method aimed at improving the quality of iterative solutions, as a variant of INSGA-II_NS, it would achieve comparable results to INSGA-II_NS.

Convergence plots for INSGA-II_NS and its three variants with respect to time objectives in 10 different scenarios are presented in Figure 9. This provides a more intuitive understanding of how the optimization content of INSGA-II_NS has different effects in different scenarios.

As can be seen from Figure 9, in the first two calculation examples, MK01-LA and MK02-LA, the convergence speeds of the four algorithms are similar, while the differences in the stable values reached by convergence are relatively obvious. Although INSGA-II_NS performs better than the other three algorithms, its performance improvement is relatively small owing to the small data scale. Besides, INSGA-II_NS-β has the worst performance, and INSGA-II_NS-α and INSGA-II_NS-γ perform successively better. In the calculation examples MK03-LA, MK04-LA and MK05-LA, with a

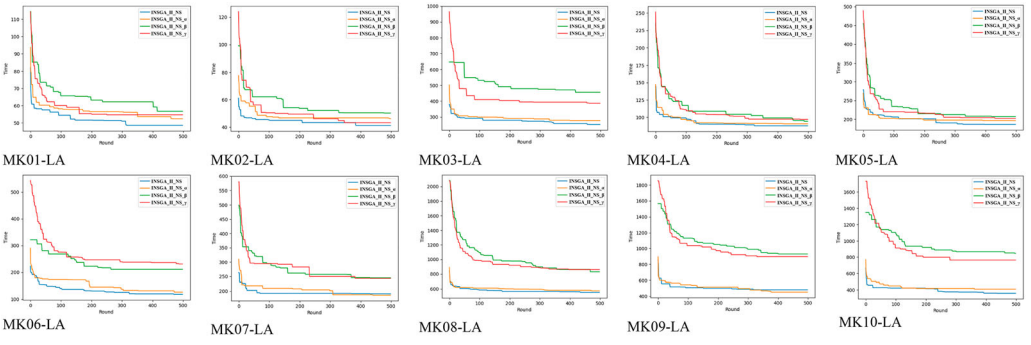

Figure 9. Time objective (makespan) convergence comparison. INSGA-II_NS = improved non-dominated sorting genetic algorithm-II with neighbourhood search.

medium data scale, a phenomenon emerges that the performance within the two groups (INSGA-II_NS and INSGA-II_NS-α, as well as INSGA-II_NS-β and INSGA-II_NS-γ) is consistent, and the former group outperforms the latter, although the performance improvement is not very significant. In the last five calculation examples, from MK06-LA to MK10-LA, the optimization effects of INSGA-II_NS and INSGA-II_NS-α are significantly better than those of INSGA-II_NS-β and INSGA-II_NS-γ . Moreover, as the data scale increases, the differences in the experimental results between the two groups grow larger. In these five groups of experiments, except for the occasional cases where the effect of INSGA-II_NS-α is slightly better than that of INSGA-II_NS in the calculation examples of MK07-LA and MK09-LA, INSGA-II_NS is always superior to INSGA-II_NS-α, and the convergence speed and convergence effects of INSGA-II_NS-γ are always better than those of INSGA-II_NS-β.

INSGA-II_NS is better optimized than the other three variants, with faster convergence and smaller convergence values. Comparing the figures, it can be seen that the convergence speed of INSGA-II_NS and INSGA-II_NS-α with population initialization optimization is faster than that of the other two, and the convergence value is also smaller, which is significantly better than the effect of considering multiple neighbourhood mutation. The last cases are more complex than the others, particularly owing to an increase in the number of jobs and machines, leading to more intricate job–machine assignment crossovers. Although population initialization is effective in these scenarios, it only provides a slight improvement in the initial objective function value compared to the NSGA-II prototype. This may be due to the heuristic rule of the population initialization optimization, which selects the machine with the shortest processing time for each operation at the initial machine code. However, for complex scenarios, there is an increased possibility of multiple operations being assigned to the same machine based on this rule. In the overall production scheduling, this arrangement may result in an imbalance of tasks across machines, causing congestion in certain machines while leaving others relatively idle. This could lead to an increase in the objective function for the entire scheduling. Nevertheless, this rule still outperforms most random initial solutions.

Regarding the optimization related to multiple neighbourhood searches, in addition to using insertion and exchange mutation methods for job codes, a heuristic rule such as population initialization is used for the mutation of machine codes. However, the scale of mutation is more random compared to the population initialization method. The mutation factor set in this article is a random number within a certain length of the encoding interval or the entire length n, $n \in [1, \text{len(Code)}]$. As a result, it is less likely to create the issue of congested tasks on a single machine. Moreover, it plays a role similar to population initialization in bringing solutions to a relatively excellent position in the early stages of iteration.

To visually represent the iterative solutions of the algorithms, a comparison and analysis of the Pareto scatter plots of INSGA-II_NS and its variant algorithms is conducted.

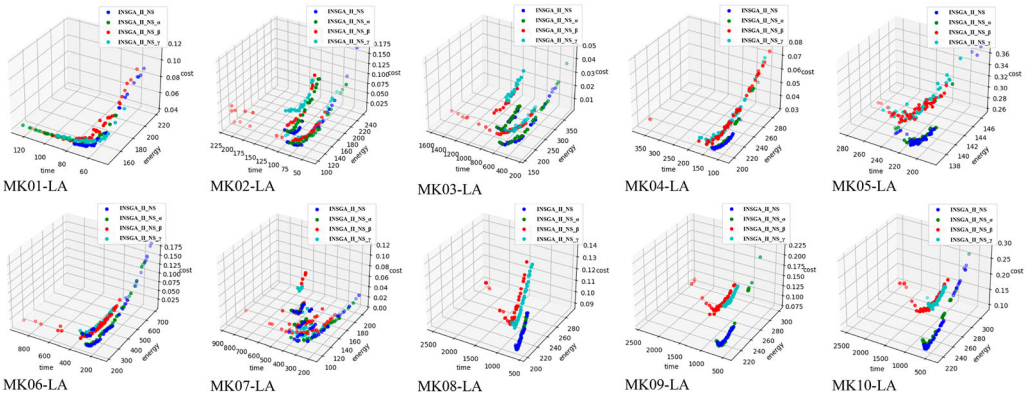

Figure 10. Pareto scatter comparison with three variants. INSGA-II_NS = improved non-dominated sorting genetic algorithm-II with neighbourhood search.

Figure 10 shows that in the calculation examples of MK01-LA and MK03-LA, the scatter plots of various algorithms are relatively scattered, which indicates that when the data scale is small, the performance differences among different algorithms are relatively significant. As can be seen from Figure 10, the positions of the scatter plots corresponding to different algorithms in the three-dimensional space are rather dispersed without an obvious clustering tendency, meaning that each algorithm performs differently in terms of convergence speed, cost, energy and other indicators, and there is no situation where the performances are close to each other. As the calculation examples range from MK04-LA to MK07-LA, the scatter plots of some algorithms begin to show a clustering tendency, demonstrating that at the medium data scale, the performances of some algorithms start to become similar. In the calculation examples from MK06-LA to MK10-LA, the scatter plots of INSGA-II_NS and INSGA-II_NS-α are significantly superior to those of INSGA-II_NS-β and INSGA-II_NS-γ in most cases, indicating that the first two algorithms perform outstandingly at the larger data scale. Moreover, as the data scale increases, this superiority becomes more and more evident. Therefore, the data scale has a profound impact on the performance of algorithms. Under the large data scale, the advantages of some algorithms begin to stand out, which also provides a basis for algorithm selection.

The above analysis suggests that the INSGA-II_NS proposed in this article is significantly superior to the traditional NSGA-II, and the optimization related to population initialization is unrelated to the quality of the final iterative solution. To verify the influence of heuristic initialization on the performance of the algorithm, a hybrid coding strategy is adopted to construct the initial population: the global exploration ability is guaranteed through random coding, and at the same time, high-quality heuristic solutions are generated by combining machine priority rules, energy consumption efficiency rules and logistics cost rules. Experiments show that this hybrid initialization method increases the effective front coverage rate and reduces the average number of iterations, verifying the necessity for the constructive heuristic to accelerate convergence and improve the quality of solutions in complex multi-objective optimization problems.

In addition, the mutation method of multiple neighbourhood searches plays a certain role in improving the quality of the iterative solutions. From MK01-LA to MK10-LA, as the number of jobs and machines increases, the Pareto point of INSGA-II_NS is usually located at the Pareto front and the optimization effect is also more obvious.

The comparative experiments with the variants have preliminarily demonstrated that INSGA-II_NS exhibits improvements in iteration speed and iterative solution quality compared to the original algorithm.

To explore the performance of the proposed algorithm compared to other multi-objective optimization model algorithms, two other similar algorithms are listed for comparison through experiments: IBEA and SPEA. Each algorithm is subjected to 10 experiments with 500 iterations under the scenario of four AGVs, applying the parameter settings described at the beginning of this subsection.

Convergence plots for time, energy consumption and cost objectives, Pareto scatter plots and boxplot distributions of iterative solutions are presented for different scenarios, to compare the performance among these algorithms.

It can be seen from Figure 11 that in the calculation examples of MK01-LA and MK02-LA, the scatter plots of each algorithm are relatively dispersed, and the overlapping parts of each algorithm are large. This indicates that when the data scale is small, the three algorithms show small differences in terms of time, cost and energy metrics, and there is no significant performance difference. As the calculation examples range from MK03-LA to MK06-LA, the scatter plots of some algorithms begin to show a clustering trend, and the performance differences among the algorithms become more obvious. INSGA-II_NS is significantly better than IBEA and SPEA, and there are also cases where IBEA and SPEA alternately take the lead. This shows that the superiority of INSGA-II_NS's performance begins to emerge at a medium-sized data scale. MK07-LA is a special calculation example. In this example, the scatter plots of the three algorithms are extremely dispersed, but the points on the Pareto front are still from INSGA-II_NS. In the calculation examples from MK08-LA to MK10-LA, the scatter plots of INSGA-II_NS are significantly better than those of IBEA and SPEA in most cases, and the scatter plots of each algorithm are also very concentrated. Moreover, as the data scale increases, this advantage becomes more and more obvious. In summary, the solutions obtained by INSGA-II_NS are predominantly located along the Pareto front, indicating a significant advantage over IBEA and SPEA. From MK01 to MK10, as the number of jobs and machines increases, the Pareto point of the INSGA-II_NS is usually located at the Pareto front, and the optimization effect is also more obvious. As shown in Figure 12, in terms of the target energy, on the whole, the target energy curves in each chart all exhibit fluctuations, although the extent of these fluctuations varies. The minimum values and the average values are relatively stable. However, the maximum values fluctuate significantly, and most of them show a zig-zag pattern. This is the case for examples such as MK01-LA, MK02-LA, MK06-LA and MK09-LA. The remaining examples tend to become stable after convergence. Regarding the target time, whether it is the maximum value, the minimum value or the average value, all the other groups are extremely stable except for MK01-LA, which has relatively large fluctuations. Moreover, as the data scale increases, this stable trend becomes even more pronounced. When it comes to the target cost, the fluctuation patterns of the curves are rich and diverse. Some groups present relatively regular fluctuations similar to sine waves, *e.g.* MK03-LA, MK06-LA and MK07-LA. Some groups have irregular multi-peak and multi-valley patterns, and some groups are relatively stable. For instance, in examples MK04-LA and MK10-LA, the cost fluctuations are relatively small. Therefore, the proposed INSGA-II_NS has lower objective function values in the initial iterations. In addition, the overall convergence speed of iterations is faster, and the obtained final iterative solutions are generally smaller.

Table 7 displays the specific data performance of three algorithms, namely INSGA-II_NS, IBEA and SPEA, on three evaluation indicators, IGD, SP and S, under 10 calculation examples from MK01-LA to MK10-LA, with the minimum values among the three algorithms highlighted in bold. The average processing time of INSGA-II_NS was 27.6 s, while the average processing times of IBEA and SPEA were 32.9 and 36.1 s, respectively. In terms of the IGD indicator, only in the three calculation examples of MK02-LA, MK04-LA and MK05-LA does IBEA or SPEA show superior performance. In the remaining seven calculation examples, INSGA-II_NS demonstrate more outstanding performance. Moreover, as the data scale continues to increase, the margin by which INSGA-II_NS leads the other two algorithms also keeps expanding, which prominently reflects the powerful advantage of this algorithm in handling large-scale data to optimize the IGD indicator. Regarding the SP indicator, only in the calculation examples of MK02-LA and MK10-LA does IBEA perform slightly better. However, its advantage over INSGA-II_NS is extremely slight, and the two are basically on the same level.

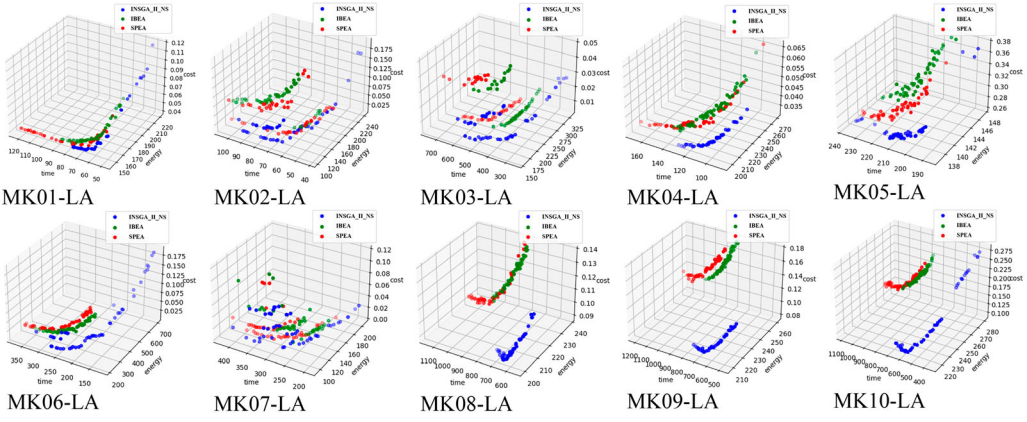

Figure 11. Pareto scatter comparison of the improved non-dominated sorting genetic algorithm-II with neighbourhood search (INSGA-II_NS) with the indicator-based evolutionary algorithm (IBEA) and strength Pareto evolutionary algorithm (SPEA).

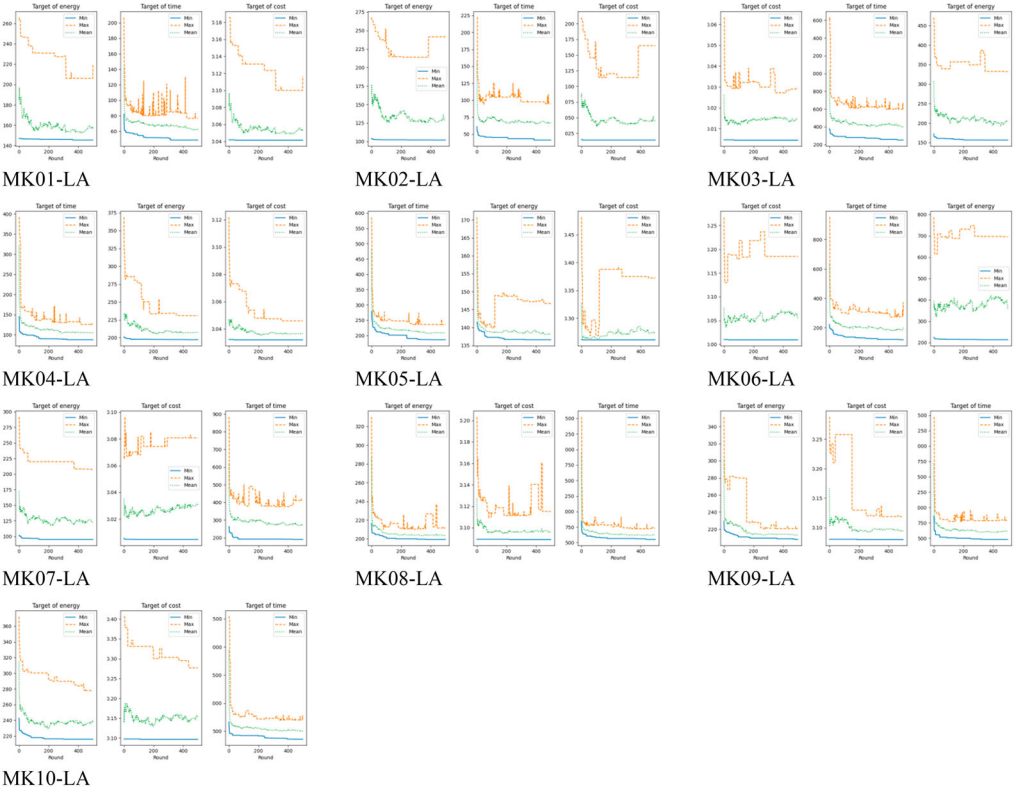

Figure 12. Performance for each objective comparison.

In terms of the S indicator, in approximately 70% of the calculation examples, the proposed algorithm performs better than the other two algorithms. These calculation examples are mainly concentrated in small- and medium-scale calculation examples, indicating that this algorithm has a unique advantage when dealing with the S indicator tasks of such scale calculation examples. Therefore, from Table

7, it can be seen that the IGD, S and SP values of INSGA-II_NS are mostly better than those of IBEA and SPEA, and the optimization effect is more obvious as the number of jobs and machines increases.

The IGD of the final iterative solutions was calculated for each algorithm, and boxplots were created to assess the quality of their solutions (Figure 13). It can be observed that the solutions obtained by INSGA-II_NS have smaller IGD values in the majority of scenarios compared to the other two algorithms, with a more compact distribution of IGD values. In the calculation examples of MK01-LA, MK03-LA, MK06-LA, MK08-LA and MK10-LA, the IGD values of INSGA-II_NS are very concentrated in their distribution, and these values are much smaller than those of IBEA and SPEA. Although the distribution is more dispersed in other calculation examples, the average IGD value is still smaller than that of IBEA and SPEA. Therefore, the proposed INSGA-II_NS must be superior to IBEA and SPEA. It achieves faster convergence in eight out of 10 scenarios, particularly under large-scale conditions (*e.g.* 15% shorter makespan than IBEA in MK10-LA), and attains lower IGD values (*e.g.* 37.7% improvement *vs* IBEA in MK08-LA), indicating denser Pareto-front approximation. These advantages stem from integrating AGV transportation time into the multi-objective optimization and adaptive neighbourhood search mechanisms.

The performance of IBEA and SPEA alternately takes the lead, and IBEA performs poorly in large-scale calculation examples. Therefore, it may be tentatively concluded that INSGA-II_NS outperforms IBEA and SPEA in terms of performance in the MK-AGV scenario proposed in this study.

To study whether there is a statistical difference between the solving methods, the relative percentage deviation (RPD) of each target of each example of each algorithm was calculated, as in Equation (35), and then normalized. At the significance level of 0.05, the Wilcoxon signed-rank test was performed on the paired RPD observations of the comparison algorithm. The null hypothesis of each significance test is defined as H0: DRPD $= 0$, which means that there is no difference between RPD values. Accordingly, the alternative hypothesis is H1: DRPD $\neq 0$, indicating that there is a significant difference between the RPD values obtained by the comparison algorithm. The results of the Wilcoxon signed-rank test are shown in the last three columns of Table 7.

$$\mathrm{RPD} = \frac{F - Fbest}{Fbest} \cdot 100 \tag{35}$$

The experimental results verify the superiority of the algorithm over IBEA, SPEA and the multi-objective evolutionary algorithm (MOEA). Statistical analysis ($p \leq 0.05$) shows that the performance of the proposed method is significantly different from that of all the comparison algorithms on multiple examples, and the three other algorithms do not show statistically significant differences in comparison, indicating that their performance levels are similar. This gap highlights the effectiveness of the new initialization strategy and adaptive neighbourhood structure in the solution space of this problem.

Finally, the article presents the Gantt chart generated by decoding the three-level encoding of the optimization objective function values obtained through iterations of the proposed INSGA-II_NS in different scenarios, resulting in the configuration of production elements for the respective scenarios. The effectiveness is depicted in Figure 14, which displays the Gantt chart after 500 iterations for MK-AGV1. The Gantt chart clearly illustrates the sequence of job changes scheduled on the machines and AGVs over time stamps, as well as the interrelationships among them.

6. Conclusions and future work

In summary, this study has accomplished the following. First, it introduces AGV collaboration and incorporates transportation time, extending the FJSP to MFJSPLA, and proposes modelling, constraints and objective functions based on the conditions of the problem; and secondly, it presents the optimization algorithm INSGA-II_NS based on the NSGA-II suitable for MFJSPLA, validates its effectiveness through experiments, and demonstrates its advantages in terms of convergence and solution quality compared to other algorithms in solving MFJSPLA.

Table 7. Experimental results and comparison of four algorithms.

	INSGA-II_NS			IBEA			SPEA			MOEA			Significant? ($p \leq 0.05$)		
Case	IGD	SP	S	IGD	SP	S	IGD	SP	S	IGD	SP	S	INSGA-II_NS vs IBEA	INSGA-II_NS vs SPEA	INSGA-II_NS vs MOEA
MK01-LA	**0.0621**	**0.2841**	**3859.67**	0.0657	0.7929	1325.96	0.0676	0.8291	948.87	0.0659	0.5873	2658.14	Yes	Yes	Yes
MK02-LA	0.0776	1.0145	**26,179.20**	0.1923	**0.9062**	5032.43	**0.0502**	1.1603	13, 225.31	0.0859	1.2523	9671.23	Yes	No	Yes
MK03-LA	**0.0162**	**3.1311**	**26,074.21**	0.0379	3.4997	20, 982.86	0.0677	3.5208	13, 979.58	0.0237	4.0258	2421.17	Yes	Yes	Yes
MK04-LA	0.9271	**0.4176**	361.72	**0.4604**	1.0379	1489.11	0.6968	1.5866	**4116.47**	0.5867	0.5987	1784.26	No	No	Yes
MK05-LA	1.7395	**0.7079**	**1681.43**	0.7702	0.7176	278.97	**0.4231**	1.4754	407.57	0.9852	0.8574	852.37	Yes	Yes	No
MK06-LA	**0.5620**	**2.2206**	**37,185.02**	0.6386	2.2798	13, 801.32	0.9547	2.2344	24, 023.85	0.7589	2.4182	15, 847.58	Yes	Yes	Yes
MK07-LA	**0.5863**	**1.6739**	**38,191.50**	0.9948	2.7071	27, 571.69	0.6531	3.6249	30, 495.92	0.8135	3.0519	29, 453.12	Yes	Yes	Yes
MK08-LA	**0.5172**	**2.0332**	**1252.70**	1.4356	2.1863	724.69	0.6770	4.3178	672.82	1.2847	3.1587	984.39	Yes	Yes	Yes
MK09-LA	**0.1681**	2.0928	2314.01	0.7869	2.2071	1728.05	0.1909	6.5023	**2452.89**	0.5469	**2.0651**	2017.14	Yes	No	No
MK10-LA	0.0847	2.9628	6808.27	1.6081	**0.9850**	3506.86	0.1281	6.8257	**15,075.33**	**0.0748**	1.4769	9821.47	No	No	Yes

Note: INSGA-II_NS = improved non-dominated sorting genetic algorithm-II with neighbourhood search; IBEA = indicator-based evolutionary algorithm; SPEA = strength Pareto evolutionary algorithm; MOEA = multi-objective evolutionary algorithm; IGD = inverted generational distance; SP = Spread; S = S-metric.
The minimum values among the three algorithms are highlighted in bold.

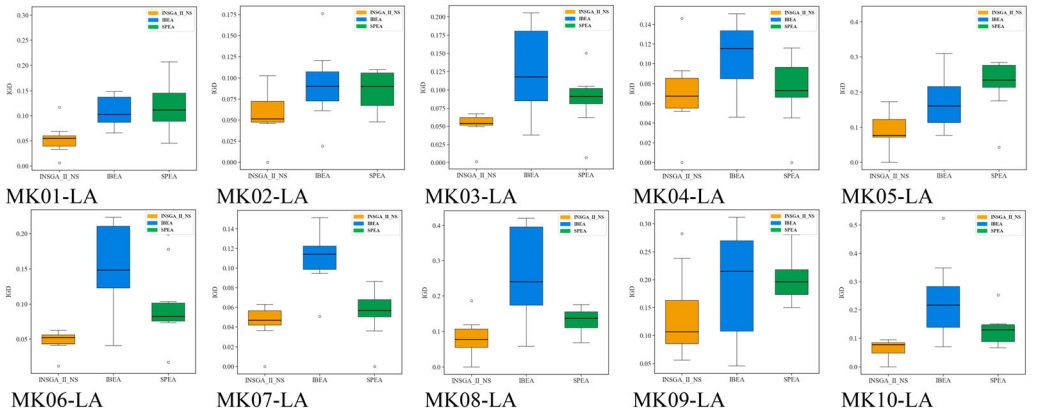

Figure 13. Boxplots of inverted generational distance (IGD) comparison of the improved non-dominated sorting genetic algorithm-II with neighbourhood search (INSGA-II_NS) with the indicator-based evolutionary algorithm (IBEA) and strength Pareto evolutionary algorithm (SPEA).

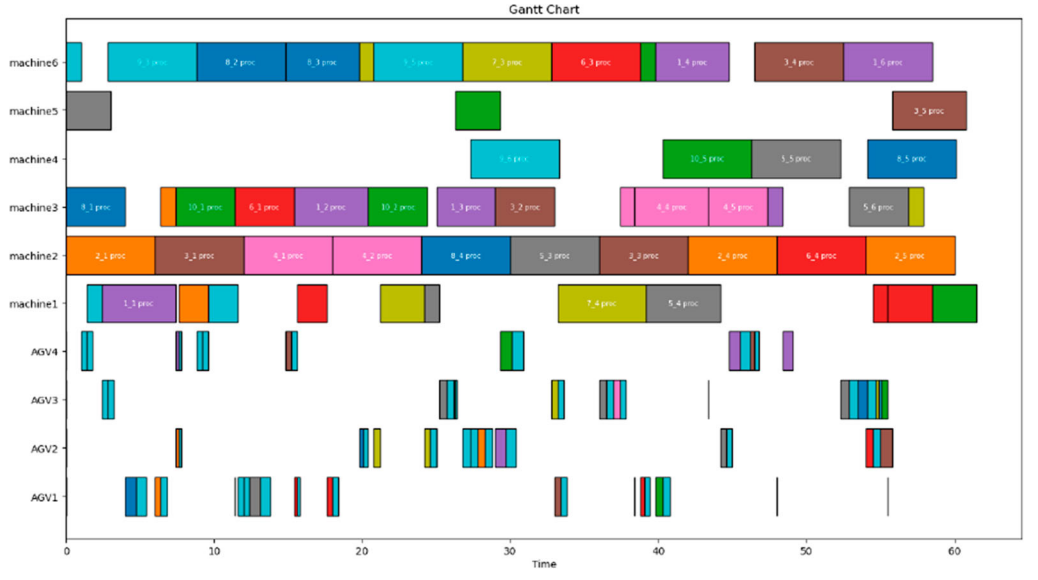

Figure 14. An example Gantt chart of a solution on MK01-LA. AGV = automated guided vehicle.

However, in this study, the extension of the MK-AGV case is relatively straightforward, lacking practicality in setting parameters such as AGV and machine power, and lacking universality in setting objective function parameters. Improvements need to be made based on specific shop-floor situations, and it is hoped that more real data can be obtained to validate the model and algorithm. In addition, although the algorithm shows certain advantages over others, unexpected phenomena still exist in different MK-AGV cases, such as some optimization effects becoming less significant, requiring further exploration into the reasons behind these issues. Lastly, the research on AGV parameters remains superficial, with the experiment in this article setting the number of AGVs at 4, whereas there could be more variations to compare different scenarios with varying AGV numbers in the same case (Sang, Tan, and Liu 2020).

The proposed INSGA-II_NS exhibits some notable advantages. First, it integrates multi-objective optimization by simultaneously minimizing makespan, energy consumption and logistics costs, addressing the sustainability and cost-effectiveness demands of modern manufacturing. Secondly, its innovative three-layer integer encoding scheme (operation sequence, machine allocation and AGV routing) effectively captures the interdependencies between machine scheduling and material handling, enabling comprehensive exploration of the solution space. However, potential limitations exist. The computational complexity of the multi-layer encoding and neighbourhood search may increase runtime for highly complex instances. While the algorithm shows robustness across diverse scenarios, its performance is sensitive to parameter settings (*e.g.* population size and mutation probability), necessitating careful tuning. Furthermore, the current model assumes a fixed number of AGVs (four in the experiments), limiting its adaptability to scenarios with dynamic AGV allocation. The energy consumption model primarily focuses on transportation and processing times, potentially overlooking dynamic factors such as AGV acceleration/deceleration or payload-dependent energy usage.

The authors look forward to the opportunity to discuss more content with scholars interested in MFJSPLA in the future. The model has more parametric characteristics waiting to be explored and studied. Furthermore, green, low-carbon initiatives are crucial today, with the introduction of AGVs and consideration of waste handling costs significantly impacting the environmental benefits of improving the FJSP. The authors hope that this article will provide assistance in addressing both the FJSP and green industry issues.

Disclosure statement

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this article.

Funding

This work is supported in part by the National Key Research and Development Program of China [grant numbers 2024YFD2400200 and 2024YFD2400204], in part by the Science and Technology Development Programme for the Two Zones [grant number 2023LQ02004], and in part by the Fundamental Research Funds for the Central Universities [grant number DUT24BK046].

Data availability

The data that support the findings of this study are available from the corresponding author, Yanjun Shi, upon reasonable request.

ORCID

Yanjun Shi http://orcid.org/0009-0007-0231-1581

References

Balas, E. 1969. "Machine Sequencing Via Disjunctive Graphs: An Implicit Enumeration Algorithm." *Operations Research* 17 (6): 941–957. https://doi.org/10.1287/opre.17.6.941

Brandimarte, P. 1993. "Routing and Scheduling in a Flexible Job Shop by Tabu Search." *Annals of Operations Research* 41 (3): 157–183. https://doi.org/10.1007/BF02023073

Burmeister, S. C., D. Guericke, and G. Schryen. 2024. "A Memetic NSGA-II for the Multi-objective Flexible Job Shop Scheduling Problem with Real-Time Energy Tariffs." *Flexible Services and Manufacturing Journal* 36 (4): 1530–1570. https://doi.org/10.1007/s10696-023-09517-7

Cao, J., Z. Guan, L. Yue, S. Ullah, and R. A. K. Sherwani. 2020. "A Bottleneck Degree-Based Migrating Birds Optimization Algorithm for the PCB Production Scheduling." *IEEE Access* 8:209579–209593. https://doi.org/10.1109/ACCESS.2020.3033002

Chang, D., H. Shi, C. Han, and F. Meng. 2023. "Research on Production Scheduling Optimization of Flexible Job Shop Production with Buffer Capacity Limitation Based on the Improved Gene Expression Programming Algorithm." *International Journal of Precision Engineering and Manufacturing* 24 (12): 2317–2336. https://doi.org/10.1007/s12541-023-00897-2

Chen, Z., J. Zou, and W. Wang. 2023. "Digital Twin Oriented Multi-objective Flexible Job Shop Scheduling Model and Its Hybrid Particle Swarm Optimization." *Proceedings of the Institution of Mechanical Engineers. Part B, Journal of Engineering Manufacture* 237 (8): 1269–1282. https://doi.org/10.1177/09544054221121921

Cheng, Y., Z. Xie, Y. Xin, K. Chen, and R. Zarei. 2024. "Flexible Job Shop Scheduling Method for Optimizing Mold Resource Setup Time." *IEEE Access* 12:33486–33503. https://doi.org/10.1109/ACCESS.2024.3505944

Durst, P., X. Jia, and L. Li. 2023. "Multi-objective Optimization of AGV Real-Time Scheduling Based on Deep Reinforcement Learning." In *2023 42nd Chinese Control Conference (CCC)*. Vol. 5, 5535–5540. https://doi.org/10.23919/CCC58697.2023.10240797

Fontes, D. B. M. M., and S. M. Homayouni. 2019. "Joint Production and Transportation Scheduling in Flexible Manufacturing Systems." *Journal of Global Optimization* 74 (4): 879–908. https://doi.org/10.1007/s10898-018-0681-7

Gong, Q., J. Li, Z. Jiang, and Y. Wang. 2024. "A Hierarchical Integration Scheduling Method for Flexible Job Shop with Green Lot Splitting." *Engineering Applications of Artificial Intelligence* 129:107595. https://doi.org/10.1016/j.engappai.2023.107595

Gu, X., M. Huang, and X. Liang. 2020. "A Discrete Particle Swarm Optimization Algorithm with Adaptive Inertia Weight for Solving Multi Objective Flexible Job-Shop Scheduling Problem." *IEEE Access* 8:33125–33136. https://doi.org/10.1109/ACCESS.2020.2974014

Hamida, M. B., A. Azzouz, and L. B. Said. 2023. "An Adaptive Variable Neighborhood Search Algorithm to Solve Green Flexible Job Shop Problem." In *2023 9th International Conference on Control, Decision and Information Technologies (CoDIT)*, Rome, Italy, 1403–1408. https://doi.org/10.1109/CoDIT58514.2023.10284046

He, L., R. Chiong, W. Li, G. S. Budhi, and Y. Zhang. 2022. "A Multi-objective Evolutionary Algorithm for Achieving Energy Efficiency in Production Environments Integrated with Multiple Automated Guided Vehicles." *Knowledge-Based Systems* 243:108315. https://doi.org/10.1016/j.knosys.2022.108315

Ho, K., J. Cheng, J. Wu, F. Chiang, Y. Chen, Y. Wu, and I. Wu. 2024. "Residual Scheduling: A New Reinforcement Learning Approach to Solving Job Shop Scheduling Problem." *Ieee Access* 12:1–1.

Jiménez Tovar, M., J. Acevedo-Chedid, H. Ospina-Mateus, K. Salas-Navarro, and S. S. Sana. 2024. "An Optimization Algorithm for the Multi-objective Flexible Fuzzy Job Shop Environment with Partial Flexibility Based on Adaptive Teaching–Learning considering Fuzzy Processing Times." *Soft Computing (Berlin, Germany)* 28:1459–1489.

Jing, X., X. Yao, M. Liu, and J. Zhou. 2024. "Multi-agent Reinforcement Learning Based on Graph Convolutional Network for Flexible Job Shop Scheduling." *Journal of Intelligent Manufacturing* 35 (1): 75–93. https://doi.org/10.1007/s10845-022-02037-5

Johnson, D., G. Chen, and Y. Lu. 2022. "Multi-agent Reinforcement Learning for Real-Time Dynamic Production Scheduling in a Robot Assembly Cell." *IEEE Robotics and Automation Letters* 7 (3): 7684–7691. https://doi.org/10.1109/LRA.2022.3184795

Li, Z., and Y. Chen. 2023. "Minimizing the Makespan and Carbon Emissions in the Green Flexible Job Shop Scheduling Problem with Learning Effects." *Scientific Reports* 13:6369–6369. https://doi.org/10.1038/s41598-023-33615-z

Li, J., Z. Liu, C. Li, and Z. Zheng. 2021. "Improved Artificial Immune System Algorithm for Type-2 Fuzzy Flexible Job Shop Scheduling Problem." *IEEE Transactions on Fuzzy Systems* 29 (11): 3234–3248. https://doi.org/10.1109/TFUZZ.2020.3016225.

Liang, X., Y. Song, Q. Lei, and A. Sun. 2023. "Research on Flexible Job-Shop Scheduling Problems Considering Optimization of Worker Number Allocation#br." *Zhongguo Jixie Gongcheng China Mechanical Engineering* 34:2065.

Meng, L., W. Cheng, B. Zhang, W. Zou, W. Fang, and P. Duan. 2023. "An Improved Genetic Algorithm for Solving the Multi-aGV Flexible Job Shop Scheduling Problem." *Sensors (Basel, Switzerland)* 23 (8): 3815. https://doi.org/10.3390/s23083815.

Özgüven, C., L. Özbakır, and Y. Yavuz. 2010. "Mathematical Models for Job-Shop Scheduling Problems with Routing and Process Plan Flexibility." *Applied Mathematical Modelling* 34 (6): 1539–1548. https://doi.org/10.1016/j.apm.2009.09.002

Ren, W., Y. Yan, Y. Hu, and Y. Guan. 2022. "Joint Optimisation for Dynamic Flexible Job-Shop Scheduling Problem with Transportation Time and Resource Constraints." *International Journal of Production Research* 60 (18): 5675–5696. https://doi.org/10.1080/00207543.2021.1968526

Sang, Y., J. Tan, and W. Liu. 2020. "Research on Many-objective Flexible Job Shop Intelligent Scheduling Problem Based on Improved NSGA-III." *IEEE Access* 8:1–1.

Sanogo, K., A. Mekhalef Benhafssa, M. H. Sahnoun, B. Bettayeb, and A. Bekrar. 2022. *Multi-agent Simulation for Flexible Job-Shop Scheduling Problem with Traffic-Aware Routing*. Cham: Springer International, 573–583.

Saqlain, M., S. Ali, and J. Y. Lee. 2023. "A Monte-Carlo Tree Search Algorithm for the Flexible Job-Shop Scheduling in Manufacturing Systems." *Flexible Services and Manufacturing Journal* 35 (2): 548–571. https://doi.org/10.1007/s10696-021-09437-4

Schworm, P., X. Wu, M. Klar, M. Glatt, and J. C. Aurich. 2024. "Multi-objective Quantum Annealing Approach for Solving Flexible Job Shop Scheduling in Manufacturing." *Journal of Manufacturing Systems* 72:142–153. https://doi.org/10.1016/j.jmsy.2023.11.015

Sze, J. F., S. Salhi, and N. Wassan. 2016. "A Hybridisation of Adaptive Variable Neighbourhood Search and Large Neighbourhood Search: Application to the Vehicle Routing Problem." *Expert Systems with Applications* 65:383–397. https://doi.org/10.1016/j.eswa.2016.08.060

Wang, Y., and Q. Zhu. 2021. "A Hybrid Genetic Algorithm for Flexible Job Shop Scheduling Problem with Sequence-Dependent Setup Times and Job Lag Times." *IEEE Access* 9:104864–104873. https://doi.org/10.1109/ACCESS.2021.3096007

Wang, H., and G. Zhu. 2023. "Multiobjective Optimization for FJSP Under Immediate Predecessor Constraints Based OFA and Pythagorean Fuzzy Set." *IEEE Transactions on Fuzzy Systems* 31 (9): 1–13. https://doi.org/10.1109/TFUZZ.2022.3180223.

Wen, X., Y. Fu, W. Yang, H. Wang, Y. Zhang, and C. Sun. 2023. "An Effective Hybrid Algorithm for Joint Scheduling of Machines and AGVs in Flexible Job Shop." *Measurement and Control (London)* 56 (9–10): 1582–1598. https://doi.org/10.1177/00202940231173750

Wu, X., X. Liu, and N. Zhao. 2019. "An Improved Differential Evolution Algorithm for Solving a Distributed Assembly Flexible Job Shop Scheduling Problem." *Memetic Computing* 11 (4): 335–355. https://doi.org/10.1007/s12293-018-00278-7

Xu, G., Q. Bao, and H. Zhang. 2023. "Multi-objective Green Scheduling of Integrated Flexible Job Shop and Automated Guided Vehicles." *Engineering Applications of Artificial Intelligence* 126:106864. https://doi.org/10.1016/j.engappai.2023.106864

Yang, J., H. Xu, J. Cheng, R. Li, and Y. Gu. 2023. "A Decomposition-Based Memetic Algorithm to Solve the Biobjective Green Flexible Job Shop Scheduling Problem with Interval Type-2 Fuzzy Processing Time." *Computers & Industrial Engineering* 183:109513. https://doi.org/10.1016/j.cie.2023.109513

Yao, Y., Q. Liu, L. Fu, X. Li, Y. Yu, L. Gao, and W. Zhou. 2024. "A Novel Mathematical Model for the Flexible Job-Shop Scheduling Problem with Limited Automated Guided Vehicles." *IEEE Transactions on Automation Science and Engineering* 22 (4): 1–14.

Yu, F., C. Lu, J. Zhou, L. Yin, and K. Wang. 2024. "A Knowledge-Guided BI-Population Evolutionary Algorithm for Energy-Efficient Scheduling of Distributed Flexible Job Shop Problem." *Engineering Applications of Artificial Intelligence* 128:107458. https://doi.org/10.1016/j.engappai.2023.107458

Yuan, M., L. Zheng, H. Huang, K. Zhou, F. Pei, and W. Gu. 2025. "Research on Flexible Job Shop Scheduling Problem with AGV Using Double DQN." *Journal of Intelligent Manufacturing* 36: 509–535. https://doi.org/10.1007/s10845-023-02252-8

Zhang, G., Y. Hu, J. Sun, and W. Zhang. 2020. "An Improved Genetic Algorithm for the Flexible Job Shop Scheduling Problem with Multiple Time Constraints." *Swarm and Evolutionary Computation* 54:100664. https://doi.org/10.1016/j.swevo.2020.100664

Zhou, K., C. Tan, Y. Wu, B. Yang, and X. Long. 2024. "Research on Low-Carbon Flexible Job Shop Scheduling Problem Based on Improved Grey Wolf Algorithm." *The Journal of Supercomputing* 80 (9): 12123–12153. https://doi.org/10.1007/s11227-024-05915-2

Zhou, K., C. Tan, Y. Zhao, J. Yu, Z. Zhang, and Y. Wu. 2024. "Research on Solving Flexible Job Shop Scheduling Problem Based on Improved GWO Algorithm SS-GWO." *Neural Processing Letters* 56 (1): 26. https://doi.org/10.1007/s11063-024-11488-1