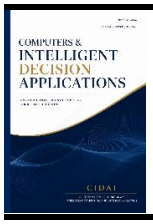
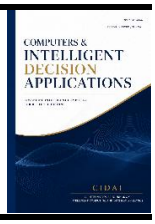


Computers & Intelligent Decision Applications

Journal homepage: www.cidai-journal.org
eISSN: 3123-125X



A Hybrid NSGA-II for Delivery-Carbon Bi-objective Flexible Job Shop Scheduling of Electric Motor Housing Machining with Fixture and Inspection Constraints

Jie Wan [1], Min Kong [1,*]

[1] School of Economics and Management, Anhui Normal University, Wuhu, 241003, China

ARTICLE INFO

Article history:
Received 30 May 2026
Received in revised form 3 July 2026
Accepted 8 August 2026
Available online 13 August 2026

Keywords:
Flexible job-shop scheduling; Motor housing; Total weighted tardiness; Carbon emissions; Fixture constraints; Final inspection; NSGA-II.

ABSTRACT

Precision machining of new-energy vehicle motor housings requires coordinated decisions on flexible machines, processing modes, limited fixtures, sequence-dependent setup, transport, and mandatory final inspection. A bi-objective flexible job-shop model is formulated to minimize total weighted tardiness and carbon emissions. Its energy account includes processing, setup, machine idling, transport, and inspection, and fixtures remain occupied throughout datum-retention blocks. The proposed HILS-NSGA-II uses an OS-MMS-FPA representation, a common serial decoder, hybrid initialization, and probabilistic Pareto local search. Gurobi experiments on small instances verify feasibility and objective calculations. Tests on 30 instances from six scale groups compare the method with NSGA-II, SPEA2, and MOPSO. Mean HV is 3.70% and 4.28% higher than that of NSGA-II and SPEA2, and mean IGD is 31.36% and 35.74% lower. Ablation results attribute most of the gain to hybrid initialization, with a smaller complementary contribution from local search. Due-date tightness has the largest effect on tardiness in the production sensitivity analysis. Fixture availability and final-inspection efficiency also alter waiting and completion performance when those resources are restrictive. The resulting nondominated sets provide delivery-oriented, low-carbon, and compromise schedules for motor-housing production.

1. Introduction

The global automotive industry is undergoing a rapid transition toward electrification and low-carbon manufacturing. Electric vehicles have moved from an emerging technology to an important part of the automobile market. Global electric car sales increased from approximately 3.0 million units in 2020 to 20.9 million units in 2025, as shown in Figure 1. Electric cars accounted for approximately one-quarter of all new cars sold worldwide in 2025 [1]. This growth has increased the demand for

* *Corresponding author.*
E-mail address: minkong@ahnu.edu.cn

electric-drive components and their supporting manufacturing capacity.

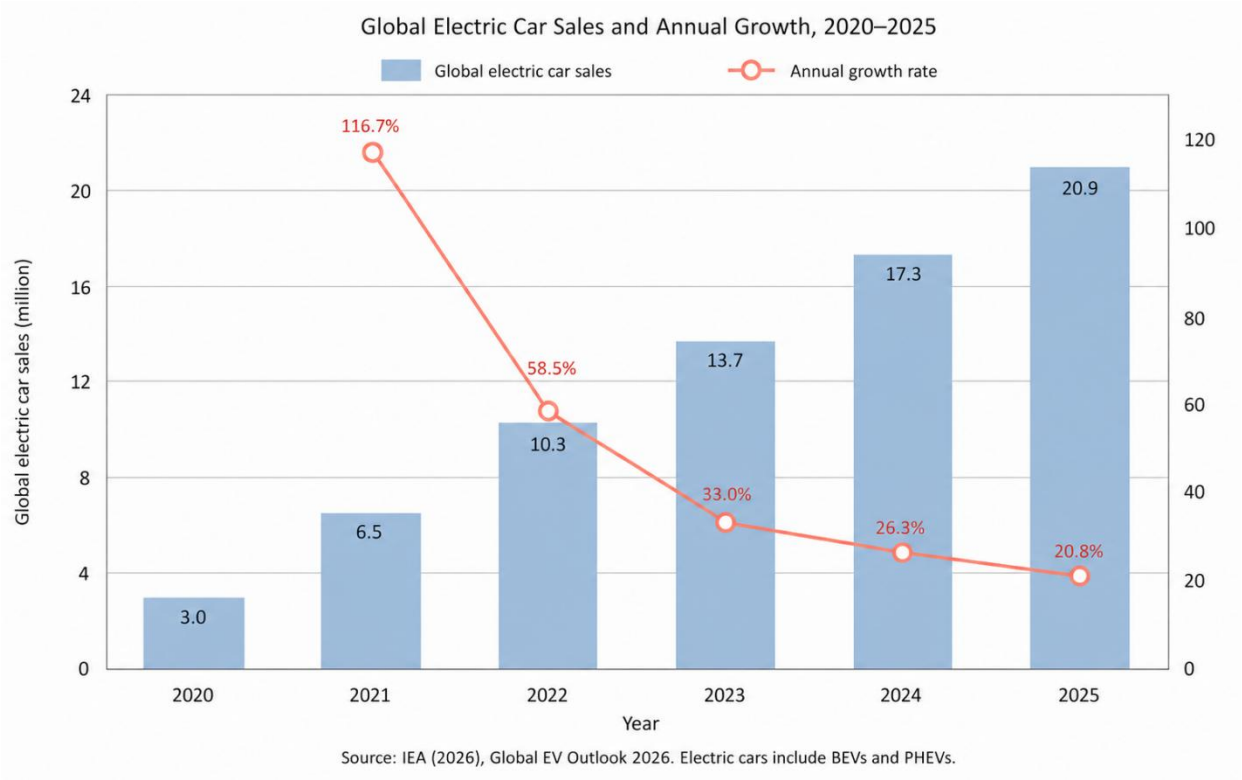


Fig. 1. Global electric car sales and annual growth from 2020 to 2025

Motor housing is an important structural component of an electric drive system. Its dimensional accuracy and assembly quality directly affect the installation of the motor stator, rotor, bearings, and cooling structure. The production of motor housings usually involves datum machining, rough machining, semi-finish and finish machining, cleaning, transport, and final inspection. Different operations may be processed by different machines. They may also require different processing modes, fixtures, and inspection resources. Production planning must therefore coordinate operation sequencing, machine assignment, processing-mode selection, fixture allocation, and inspection scheduling.

The flexible job-shop scheduling problem provides a suitable basis for modeling alternative machine assignments and operation sequences. It extends the classical job shop by allowing each operation to be processed on one of several eligible machines. This flexibility introduces a machine-assignment decision in addition to the sequencing decision. The resulting problem is a difficult combinatorial optimization problem with extensive industrial applications [2, 3]. However, the conventional flexible job-shop model mainly focuses on machines and processing operations. It does not fully describe the coupled resources and technological requirements of motor-housing precision machining.

Energy consumption and carbon emissions have become important criteria in production scheduling. Scheduling can reduce energy use at the system level without requiring major changes to existing equipment [4]. Liu *et al.,* [5] formulated a job-shop scheduling problem that minimizes total weighted tardiness and energy consumption. Their results demonstrated the conflict between delivery performance and energy use. Zhang and Chiong [6] further developed a multi-objective genetic algorithm with local improvement strategies for the same two objectives. In a flexible job shop, alternative machines and processing speeds provide more opportunities for energy-aware

decisions. Yin *et al.,* [7] treated the machining speed as a decision variable because it affects both processing time and power consumption.

Several shop-floor features are still rarely handled in one model. Product changes create sequence-dependent setups that consume machine time and energy. Inter-machine transfers add transport time and energy [8]. Fixtures are finite physical resources, so machine availability alone does not guarantee a feasible schedule. Wu *et al.,* [9] showed the value of scheduling machines and fixtures together, while Zhou *et al.,* [10] reported that machine-fixture-pallet compatibility can alter resource allocation and production delay.

Motor-housing machining imposes an additional fixture requirement. Some precision operations belong to a datum-retention block. A compatible fixture must remain occupied from the beginning of the first operation in the block until the completion of the last operation. It cannot be released during waiting or inter-machine transport within the block. Final inspection is also mandatory. It requires a qualified inspection resource, consumes time and energy, and determines the actual completion time of each job. A schedule that ignores fixture continuity or inspection congestion may therefore appear feasible while being difficult to execute on the shop floor.

Another challenge arises from the conflict between delivery and environmental objectives. High-speed processing can shorten completion times and reduce tardiness, but it normally requires higher power. Energy-saving processing reduces operating power but may extend machine occupation. Fixture shortages and inspection congestion can further amplify this conflict. A single-objective schedule cannot fully represent these alternatives. A set of nondominated solutions is more useful because it allows managers to select delivery-oriented, low-carbon-oriented, or balanced production plans.

This paper studies a delivery-low-carbon flexible job-shop scheduling problem for precision machining of electric-motor housings. The two objectives are total weighted tardiness and carbon emissions. The emission calculation covers processing, inspection, sequence-dependent setup, internal machine idling, and inter-machine transport. The model also includes flexible machines, high-speed, normal, and energy-saving modes, product-family setups, fixture compatibility, continuous fixture occupation over datum-retention blocks, and mandatory final inspection.

The coupled decisions create a large discrete search space, which limits exact optimization to small instances. Multi-objective evolutionary search is more practical for the larger cases. NSGA-II combines fast nondominated sorting with crowding distance [11], but its standard initialization and operators do not use the delivery and energy structure of this problem. HILS-NSGA-II retains the NSGA-II selection framework and adds a three-layer representation, preference-guided initialization, and probabilistic Pareto local search.

The main contributions of this study are as follows:

i. A delivery-low-carbon scheduling model is developed for electric motor-housing precision machining. The model integrates machine-mode decisions, sequence-dependent setups, transport, fixtures, datum-retention blocks, and final inspection in a unified framework.
ii. A three-layer representation is designed for operation sequences, machine-mode selections, and fixture assignments. A unified serial decoder converts each encoded individual into a complete schedule and evaluates total weighted tardiness and carbon emissions.
iii. HILS-NSGA-II is proposed to solve the problem. Its initial population combines random, delivery-oriented, and low-carbon-oriented solutions. A probabilistic Pareto local search further explores the operation-sequence, machine-mode, and fixture-assignment layers.

iv. A systematic computational study is conducted. The mathematical model is first checked using small instances and Gurobi. The algorithm is then evaluated on 30 test instances covering six production scales. Parameter confirmation, algorithm comparison, ablation analysis, and production-factor sensitivity analysis are also performed.

Section 2 reviews the related literature. Section 3 defines the scheduling problem and mathematical model. Section 4 describes HILS-NSGA-II. Section 5 presents the experiments and results, and Section 6 closes the paper.

2. Literature Review

The flexible job-shop scheduling problem extends the classical job shop by allowing each operation to use one of several eligible machines. Machine assignment and operation sequencing must therefore be decided together. The added flexibility can balance workloads, but it also enlarges the search space. Dauzère-Pérès *et al.,* [3] noted that makespan remains the most common objective, although due dates, energy use, multiple resources, and industrial constraints are receiving more attention. Li *et al.,* [12] reviewed integrated FJSP models that connect scheduling with process planning, maintenance, transport, and assembly. The literature is moving beyond machine-centered models, yet many practical constraints are still treated in separate extensions.

Energy-efficient scheduling is an important part of this development. Gahm *et al.,* [4] showed that production scheduling can reduce manufacturing energy demand without replacing production equipment. The resulting schedules must usually balance production efficiency against energy performance. Liu *et al.,* [5] formulated a job-shop problem that minimizes total weighted tardiness and total energy consumption. Their results showed that reducing energy consumption may increase delivery delay. Zhang and Chiong [6] considered the same two objectives and introduced machine speed scaling. Faster processing could reduce tardiness but increase energy consumption. These studies provide a direct basis for examining the delivery-energy trade-off. However, they were developed for classical job shops and did not consider flexible machine assignment.

Energy and carbon objectives have subsequently been introduced into flexible job-shop models. Yin *et al.,* [7] treated spindle speed as an independent scheduling decision and jointly considered productivity, energy efficiency, and noise. Their encoding linked operation sequencing, machine assignment, and spindle-speed selection. Meng *et al.,* [13] developed MILP models for energy-aware FJSP. Their models considered processing energy, idle energy, and machine switching decisions. These studies confirm that machine assignment and operating conditions should be optimized together. Their production criteria, however, mainly concern makespan or productivity. Order release times, due dates, and priority-weighted tardiness are not their central criteria. Limited fixtures and final-inspection resources are also outside their modeling scope.

Sequence-dependent setup is another important feature of practical job shops. The setup required on a machine may depend on the product families of two adjacent jobs. Ignoring this relationship may underestimate machine occupation and delay subsequent operations. Shen *et al.,* [8] formulated an FJSP with sequence-dependent setup times and developed a tabu-search method based on the structural properties of the problem. Their model minimized makespan and did not include energy consumption or secondary resources. Rossi and Dini [14] considered routing flexibility, sequence-dependent setup, and transportation time in a flexible manufacturing system. Their study demonstrated that machine assignment, machine sequencing, setup, and material movement should be coordinated. It did not include carbon emissions, processing modes, or fixture competition.

Transportation further connects machine assignment with schedule performance. If two consecutive operations are assigned to different machines, additional transportation time and energy

are required. Transportation may also delay the release of fixtures or the start of inspection. Existing setup- and transport-aware FJSP studies improve the representation of machine-related activities. However, transportation is usually modeled without continuous fixture occupation or mandatory final inspection. Its effect on total weighted tardiness has also received less attention than its effect on makespan.

Limited auxiliary resources form another branch of FJSP research. Wu *et al.,* [9] studied a dual-resource FJSP involving machines and fixtures and considered fixture loading and unloading. Their results showed that fixture availability can affect operation start times and production efficiency. Liu *et al.,* [15] considered machine-fixture-pallet combinations in a multi-resource FJSP. They introduced a correction strategy to resolve conflicts between machine and fixture selections. Zhou *et al.,* [10] further integrated machines, fixture-pallets, and loading and unloading activities in a pallet automation system. These studies confirm that auxiliary resources may restrict the feasible machine set and become production bottlenecks. Their main objective is makespan. Fixture allocation is mainly associated with an individual operation or a fixture-pallet combination. They do not consider the requirement that one fixture remain continuously occupied across several datum-sensitive operations.

Inspection is less common in low-carbon FJSP models. Zhu *et al.,* [16] studied operation inspection in a dynamic distributed FJSP, where inspection outcomes could trigger rework, scrapping, transport, and rescheduling. Their findings show that inspection can affect both production efficiency and energy use. Precision machining has a different requirement: final inspection is a mandatory last task. It uses a limited inspection resource, requires transport from the last production machine, and determines the actual completion time. Its sequence can therefore change total weighted tardiness.

Multi-objective evolutionary algorithms are commonly used to solve extended FJSP models. NSGA-II provides nondominated sorting, crowding-distance assignment, and elitist environmental selection for multi-objective optimization [11]. Liu *et al.,* [5] used NSGA-II to examine the trade-off between tardiness and energy consumption. Zhang and Chiong [6] combined a multi-objective genetic algorithm with an enhanced local search. Yin *et al.,* [7] designed a problem-oriented representation and evolutionary operators for low-carbon FJSP. Fixture-constrained studies have also introduced special chromosomes, feasibility correction, and neighborhood search. These methods indicate that a general evolutionary framework must be supported by a representation and decoder that preserve the constraints of the specific production problem.

Table 1 groups the representative studies according to their main modeling focus. Full and partial coverage are distinguished because some studies consider only one component of a combined feature. For example, some models include flexible machine assignment without processing-mode selection. Inspection-related models may include intermediate inspection but not mandatory final inspection.

The studies in Table 1 establish the separate foundations of the present problem. Some examine the delivery-energy conflict, others combine flexible machines with operating decisions, and a smaller group considers setup, transport, fixtures, pallets, or inspection. Few models connect these elements within the same schedule.

The gap lies in this limited integration rather than in any single missing feature. None of the representative models jointly covers the delivery-carbon trade-off, machine-mode flexibility, sequence-dependent setup and transport, limited compatible fixtures, continuous fixture occupation over datum-retention blocks, and mandatory final inspection. These features occur together in electric-motor-housing machining and are modeled together here.

Table 1
Comparison of the main scheduling features in related studies

Research stream	Delivery-carbon trade-off	Machine-mode flexibility	Setup and transport	Limited fixtures	Continuous datum retention	Mandatory final inspection
[5, 6]	✓	△	—	—	—	—
[7, 13]	△	✓	—	—	—	—
[8, 14]	—	△	✓	—	—	—
[9, 10, 15]	—	△	—	✓	—	—
[16]	△	△	△	—	—	
This study	✓	✓	✓	✓	✓	✓

Note: ✓ indicates full coverage of the listed feature; △ indicates partial or related coverage; — indicates that the feature is not included. Machine-mode flexibility requires both alternative-machine assignment and processing-mode selection. Mandatory final inspection refers to a capacity-constrained last operation whose completion determines job completion.

The combined setting creates direct dependencies among decisions. A machine-mode assignment changes processing time, power, transport, and fixture feasibility. A fixture remains occupied throughout its datum-retention block, while a separate inspection resource determines final job completion. The model and decoder must preserve all of these links. The proposed bi-objective formulation minimizes total weighted tardiness and carbon emissions, and HILS-NSGA-II generates schedules with different delivery and carbon preferences.

3. Problem Description and Mathematical Model

This section describes the production setting and formulates the scheduling problem. It first introduces the process flow, resource interactions, and scheduling decisions. A bi-objective mathematical model is then developed to minimize total weighted tardiness and carbon emissions under machine, processing-mode, fixture, transport, setup, and final-inspection constraints.

3.1 Problem Description

This study addresses a low-carbon flexible job shop scheduling problem in the precision machining of electric-motor housings. Each order has a release time, due date, priority weight, and product family. It follows a predefined route through datum machining, rough machining, finish machining, cleaning, and final inspection. Most production operations can use several eligible machines. The schedule must determine both the operation sequence and machine assignments, which form the two basic decisions of an FJSP [3].

Figure 2 summarizes the process, resource relationships, and objectives. Figure 2(a) shows a typical route for an electric-motor housing. Figure 2(b) presents the coupled decisions for machines, processing modes, fixtures, and final-inspection devices. Figure 2(c) illustrates the trade-off between delivery performance and carbon emissions.

After an operation is assigned to an eligible machine, a permitted processing mode must also be selected. The model uses high-speed, normal, and energy-saving modes, denoted by H, N, and E. Each mode has its own processing time and operating power. High-speed processing is faster but usually draws more power. Energy-saving processing lowers power but may occupy the machine longer. Normal mode provides an intermediate option. Machine and mode decisions therefore affect completion time, tardiness, and energy use together. Similar delivery-energy conflicts have been reported in energy-aware scheduling [5, 17, 18].

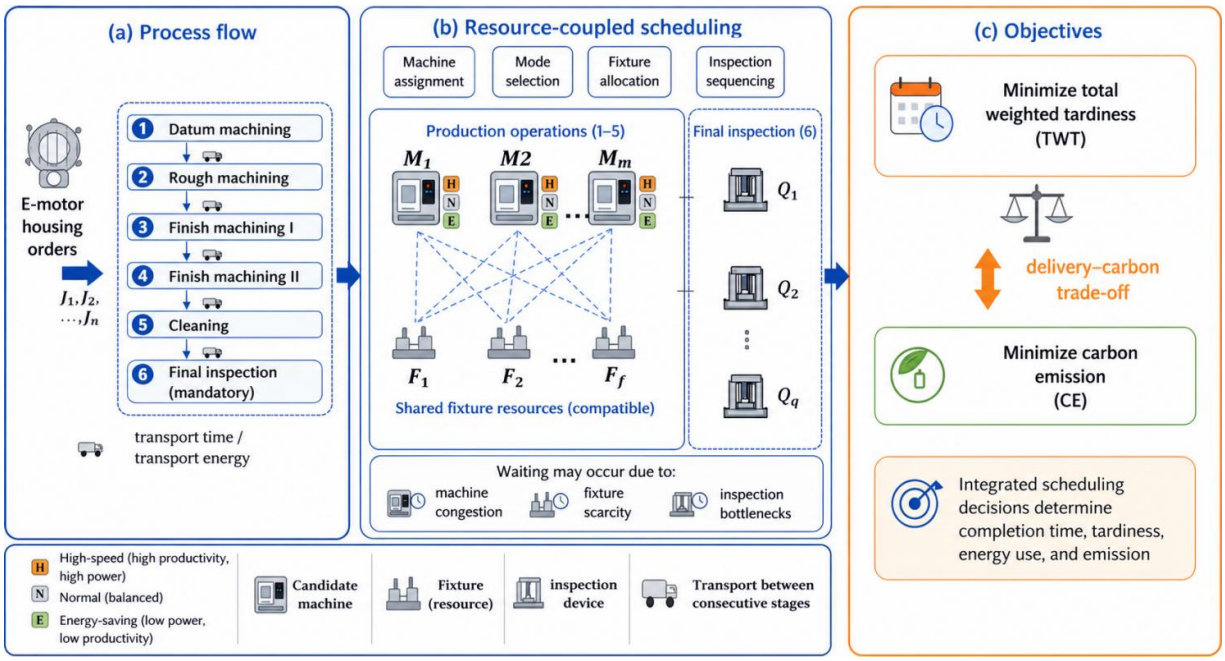


Fig. 2. Low-carbon scheduling of electric-motor housings with machines, processing modes, fixtures, and mandatory inspection

Some finish-machining operations require a stable positioning datum. The process route therefore contains a datum-retention block. In Figure 2, the block covers Finish Machining I and II. Before the block starts, the order receives a fixture that is compatible with both the product and the selected machines. The fixture is occupied from the start of the first block operation until the last block operation finishes. It remains attached during waiting and inter-machine transport and cannot serve another order during this interval.

This rule differs from treating a fixture as an auxiliary resource for one operation only. Fixture capacity is limited, and occupation is continuous across the datum-retention block. Operation feasibility thus depends on machine capability, job-fixture compatibility, and machine-fixture compatibility. Multi-resource FJSP studies have shown that fixtures and pallets can change eligible-machine sets and operation start times [9, 15].

When consecutive tasks from different product families use the same machine, a sequence-dependent setup is required. Its duration and energy use depend on the two product families and the machine. Setup occupies the machine and delays the following task, so it must be determined with the machine sequence. Shen *et al.,* [8] also showed that sequence-dependent setup is important to the feasibility and quality of FJSP schedules.

A workpiece must be transported when consecutive operations use different devices. Transport adds time and energy. Both values are zero when consecutive production operations use the same machine. The model covers transport between production machines and transport from the last production machine to the final-inspection device. Waiting may arise from machine congestion, fixture scarcity, or inspection-device occupation. Waiting inside a datum-retention block remains part of the fixture-occupation interval. Including transport in the schedule gives a more faithful account of internal logistics and its delivery and environmental effects [19].

Every order undergoes final quality inspection after all production operations. This task cannot be omitted or moved earlier and must use an eligible inspection device. When several orders share one device, their inspection sequence must also be determined. Job completion is defined by the end of

mandatory final inspection, not by the end of the last machining operation. Limited inspection capacity can therefore create waiting and increase tardiness.

A complete schedule determines four linked decisions: the production-operation sequence, the machine-mode assignment of each operation, the fixture assigned to each datum-retention block, and the device and sequence used for final inspection. The schedule must satisfy technological precedence, unit machine capacity, mode eligibility, fixture compatibility and continuous occupation, sequence-dependent setup, transport, and final-inspection constraints.

Two criteria evaluate a schedule. Total weighted tardiness measures delivery performance and accounts for both priority and delay. Carbon emissions include processing, inspection, setup, internal machine idle time, and transport. Faster modes can lower tardiness at higher energy use, while energy-saving modes may occupy machines for longer. The output is therefore a nondominated set containing delivery-oriented, low-carbon, and compromise schedules rather than one scalar optimum.

3.2 Notation

The notation extends the standard job-operation-machine structure of the FJSP with processing modes, fixtures, product families, and inspection devices. Table 2 lists the sets, indices, parameters, and decision variables used in the model.

Table 2
Notations Used in the Mathematical Model

Sets	
J	Set of jobs, $J = \{1,2,\ldots,n\}$
O_j	Set of operations of job j
O	Set of all productionoperations, $O = \cup_{j\in J} O_j$
$\mathcal{P}_j$	Set of immediate technological precedence relations of job j; if $(o,o') \in \mathcal{P}_j$, operation o immediately precedes operation o'.
M	Set of production machines.
Q	Set of final-inspection devices.
R	Set of all production and inspection devices, $R = M \cup Q$.
K	Set of processing modes, $K = \{H, N, E\}$.
F	Set of physical fixtures.
G	Set of product families.
B_j	Datum-retention block of job j, $B_j \subseteq O_j$.
M_{jo}	Set of eligible machines for operation o of job j.
K_{jom}	Set of modes available to operation o on machine m.
F_j	Set of fixtures compatible with job j.
M_f	Set of machines compatible with fixture f.
Q_j	Set of devices eligible for the final inspection of job j.
r_j	Release time of job j.
d_j	Due date of job j.
w_j	Tardiness weight of job j.
g_j	Product family of job j.
$\bar{o}_j$	Last production operation of job j.
o_j^{B-}	First operation in the datum-retention block of job j.
o_j^{B+}	Last operation in the datum-retention block of job j.
p_{jomk}	Processing time of operation o on machine m in mode k.
P_{jomk}^{P}	Processing power of operation o on machine m in mode k.
p_{jq}^{I}	Final-inspection time of job j on device q.
P_{jq}^{I}	Operating power for the final inspection of job j on device q.

Table 2
Continued

Sets	
$s_{gg'm}$	Setup time on machine m when switching from family g to family g'.
P_m^S	Setup power of machine m.
$t_{ll'}^T$	Transport time from device l to device l'.
$e_{ll'}^T$	Transport energy from device l to device l'.
P_m^{idle}	Internal-idle power of machine m.
γ^C	Carbon-emission factor per unit of electricity.
ρ_t	Time-to-energy conversion factor; it equals 1/60 when time is measured in minutes.
M_0	A sufficiently large constant used to linearize disjunctive relations.
Indices	
j, j'	Job index, $j, j' \in J$.
o, o′	Operation index.
m, m'	Production-machine index, $m, m' \in M$.
q	Final-inspection-device index, $q \in Q$.
l, l'	Production- or inspection-device index, $l, l' \in R$.
k	Processing-mode index, $k \in K$.
f	Fixture index, $f \in F$.
g, g'	Product-family index, $g, g' \in G$.
Decision variables	
x_{jomk}	1 if operation o is processed on machine m in mode k; 0 otherwise.
Φ_{jf}	1 if the datum-retention block of job j uses fixture f; 0 otherwise.
y_{jq}	1 if the final inspection of job j is assigned to device q; 0 otherwise.
$z_{jo,j'o',m}$	1 if operations (j, o) and (j', o') are assigned to machine m and the former precedes the latter; 0 otherwise.
$\xi_{jo,j'o',m}$	1 if operation (j, o) is the immediate predecessor of operation (j', o') on machine m; 0 otherwise.
$\xi_{0,jo,m}$	1 if the dummy source 0 is the immediate predecessor of operation (j, o) on machine m, indicating that (j, o) is the first operation processed on machine m; 0 otherwise.
$\xi_{jo,E,m}$	1 if operation (j, o) is the immediate predecessor of the dummy sink E on machine m, indicating that (j, o) is the last operation processed on machine m; 0 otherwise.
$\theta_{joo'mm'}$	1 if consecutive operations o and o' of job j are assigned to m and m'; 0 otherwise.
$v_{jj'f}$	1 if jobs j and j' use fixture f and the datum-retention block of job j occupies it first; 0 otherwise.
$u_{jj'q}$	1 if jobs j and j' use inspection device q and job j precedes job j'; 0 otherwise.
η_{jmq}	1 if the last production operation of job j uses machine m and final inspection uses device q; 0 otherwise.
Auxiliary quantities and objective values	
E^P	Processing and final-inspection energy.
E^S	Sequence-dependent setup energy.
E^{idle}	Internal-idle energy of production machines.
E^T	Transport energy for production operations and final inspection.
CE	Carbon emissions of a schedule.
TWT	Total weighted tardiness of all jobs.
S_{jo}	Start time of operation o.
C_{jo}	Completion time of operation o.
S_j^I	Start time of the final inspection of job j.
C_j	Completion time of final inspection for job j, also the actual job completion time.
T_j	Tardiness of job j.
$G_{jo,j'o',m}$	Internal idle time between operation (j, o) and its immediate successor (j', o') on machine m.

In Table 2, O_j denotes only the production operations of job j. Final inspection is a separate task performed after production. It is described by assignment variable y_{jq} and inspection-time variables S_j^I and C_j. A job is complete only after final inspection; therefore, C_j is both its final-inspection completion time and actual completion time.

Fixture variable ϕ_{jf} applies only to the datum-retention block of job j. The fixture is occupied when the first block operation o_j^{B-} starts and is released when the last block operation o_j^{B+} finishes. Its continuous occupation interval is therefore

$$\left[S_{j,o_j^{B-}}, C_{j,o_j^{B+}}\right] \tag{1}$$

Transport time and energy depend on the origin and destination devices. To describe both transport between consecutive production machines and transport from the last production machine to the final-inspection device, define the complete device set $R = M \cup Q$, with indices l and l' denoting the origin and destination. No inter-device transport occurs when consecutive operations of the same job use the same machine; hence,

$$t_{mm}^T = 0, e_{mm}^T = 0, \forall m \in M \tag{2}$$

Machine-order variable $z_{jo,j'o',m}$ and immediate-predecessor variable $\xi_{jo,j'o',m}$ serve different purposes. The former sets the relative order of any two operations assigned to the same machine and enforces machine capacity. The latter identifies only adjacent tasks in a machine sequence and is used to calculate setup time, setup energy, and internal-idle energy. A dummy source and sink can be linked through ξ to the first and last tasks; no additional order variables are required.

$E^P, E^S, E^{idle}, E^T, CE$ and TWT are derived from scheduling and time variables. They are auxiliary quantities or objective values rather than independent scheduling decisions. The next section defines the energy components, total energy, and carbon emissions.

3.3 Energy Consumption and Carbon Emissions

Once the operation sequence, machine-mode assignments, fixture allocation, and final inspections are fixed, the energy use of the schedule can be calculated. Total energy is divided into processing energy, sequence-dependent setup energy, internal machine idle energy, and transport energy. This scope covers both machine states and internal logistics and follows common energy-accounting practice in low-carbon scheduling [5, 19].

Fixtures are passive auxiliary resources, so their occupation does not consume energy directly. Facility loads such as lighting and air conditioning are also excluded because they do not vary materially across schedules. Energy used while an inspection device performs a task is included in processing energy, but idle energy between inspection tasks is not counted.

Let ρ_t be the time-to-energy conversion factor. It equals $\rho_t = 1/60$ when time is measured in minutes and power in kW, and the following value when time is measured in hours: $\rho_t = 1$. The electricity carbon-emission factor was fixed at $\gamma^C = 0.68$kg CO_2/kWh in the instance-generation configuration and applied to all electricity use.

3.3.1 Processing energy

Processing energy includes production and final-inspection energy. The processing time and power of an operation depend on its machine-mode assignment. Their combined energy consumption is

$$E^p = \rho_t \left(\sum_{j\in J} \sum_{o\in O_j} \sum_{m\in M_{jo}} \sum_{k\in K_{jom}} P_{jomk}^P \, p_{jomk} x_{jomk} + \sum_{j\in J} \sum_{q\in Q_j} P_{jq}^I \, p_{jq}^I y_{jq} \right) \tag{3}$$

The first term is the operating energy of all production operations. The second is the energy used by mandatory final inspection. Each operation receives one machine-mode combination and each job receives one inspection device, so Eq. (3) gives the processing energy of a schedule directly.

3.3.2 Sequence-dependent setup energy

A machine requires adjustment when it processes consecutive tasks from different product families. The setup time depends on the two families and the machine. If operation (j, o) immediately precedes operation (j', o') on machine m, the setup energy is the product of setup power and setup time. Total setup energy is

$$E^S = \rho_t \sum_{m \in M} \sum_{(j,o) \in \mathcal{O}_m} \sum_{\substack{(j',o') \in \mathcal{O}_m \\ (j',o') \neq (j,o)}} P_m^S s_{g_j g_{j'} m} \xi_{jo,j'o',m} \tag{4}$$

The setup time is zero when two adjacent tasks belong to the same product family. No initial setup is counted before the first task on a machine. Equation (4) therefore includes only adjacent production tasks with an actual predecessor.

3.3.3 Internal machine idle energy

After finishing a task, a production machine may wait because the next operation is not ready, a fixture is occupied, or another resource is unavailable. Only idle periods between adjacent tasks in a machine sequence are counted. Time before the first task and after the last task is excluded.

When operation (j, o) immediately precedes operation (j', o') on machine m, let $G_{jo,j'o',m}$ denote the idle interval after excluding the required setup. Total internal-idle energy is

$$E^{idle} = \rho_t \sum_{m \in M} \sum_{(j,o) \in \mathcal{O}_m} \sum_{\substack{(j',o') \in \mathcal{O}_m \\ (j',o') \neq (j,o)}} P_m^{idle} G_{jo,j'o',m} \tag{5}$$

where $G_{jo,j'o',m}$ can be positive only when $\xi_{jo,j'o',m} = 1$; otherwise, it is zero. For two adjacent tasks, this idle interval satisfies

$$G_{jo,j'o',m} = max\left\{0, S_{j'o'} - C_{jo} - s_{g_j g_{j'} m}\right\} \tag{6}$$

The maximum relation in Eq. (6) is linearized in the mathematical model. Subtracting setup time prevents the same interval from being counted as both setup energy and idle energy.

3.3.4 Inter-device transport energy

A workpiece is transported when consecutive operations of the same job use different machines. It is also moved from the last production machine to the selected final-inspection device. Transport energy depends on the path rather than on one operation alone.

Define the following auxiliary binary variable for the machine pair used by consecutive production operations:

$$\theta_{joo'mm'} = \begin{cases} 1, \text{if job } j \text{ has consecutive operations } o \text{ and o}' \text{ processed} \\ \qquad\qquad \text{on machines } m \text{ and } m', \text{respectively} \\ 0, otherwise \end{cases} \tag{7}$$

also define

$$\eta_{jmq} = \begin{cases} 1, \text{if job } j \text{ finishes production on machine } m \text{ and} \\ \qquad\qquad \text{uses inspection device } q \\ 0, othewise \end{cases} \tag{8}$$

The inter-device transport energy is then

$$E^T = \sum_{j\in J}\sum_{(o,o\prime)\in P_j}\sum_{m\in M_{jo}}\sum_{m\prime\in M_{jo\prime}} e^T_{mm\prime}\cdot\theta_{joo\prime mm\prime} + \sum_{j\in J}\sum_{m\in M_{j\bar{o}_j}}\sum_{q\in Q_j} e^T_{mq}\cdot\eta_{jmq} \tag{9}$$

The first term in Eq. (9) represents transport between consecutive production machines. The second represents transport from the last production machine to the final-inspection device. When consecutive production operations use the same machine, define

$$e^T_{mm} = 0, \forall m \in M \tag{10}$$

Thus, consecutive processing on the same machine adds no transport energy.

3.4 Mathematical Model

Based on the problem description, assumptions, and energy accounting above, a bi-objective mixed-integer linear programming model is formulated to minimize total weighted tardiness and carbon emissions. It jointly determines operation sequences, machine-mode assignments, fixture allocation for datum-retention blocks, and final-inspection devices and sequences.

For compact notation, define

$$X_{jom} = \sum_{k\in K_{jom}} x_{jomk} \tag{11}$$

where, if operation o of job j is assigned to machine m, then $X_{jom} = 1$; otherwise, $X_{jom} = 0$. Also let

$$J^B = \{j \in J | B_j \neq \emptyset\} \tag{12}$$

denote the set of jobs that contain a datum-retention block, and define

$$\mathcal{O}_m = \{(j,o) | j \in J, o \in O_j, m \in M_{jo}\} \tag{13}$$

is the candidate-operation set for machine m. Each machine sequence also has dummy source 0 and dummy sink E.

Objective:

$$minf_1 = TWT = \sum_{j\in J} w_j \cdot T_j \tag{14}$$

$$minf_2 = CE = \gamma^C\left(E^P + E^S + E^{idle} + E^T\right) \tag{15}$$

Subject to:

$$\sum_{m\in M_{jo}}\sum_{k\in K_{jom}} x_{jomk} = 1, \forall j \in J, o \in O_j \tag{16}$$

$$\sum_{f\in F_j} \phi_{jf} = 1, \forall j \in J^B \tag{17}$$

$$\sum_{q\in Q_j} y_{jq} = 1, \forall j \in J \tag{18}$$

$$C_{jo} = S_{jo} + \sum_{m\in M_{jo}}\sum_{k\in K_{jom}} p_{jomk}x_{jomk}, \forall j \in J, o \in O_j \tag{19}$$

$$S_{jo} \geq r_j, \forall j \in J, o \in O_j \tag{20}$$

$$S_{jo'} \geq C_{jo} + \sum_{m \in M_{jo}} \sum_{m' \in M_{jo'}} t^T_{mm'} \cdot \theta_{joo'mm'}, \forall j \in J, (o, o') \in \mathcal{P}_j \tag{21}$$

$$\theta_{joo'mm'} \leq X_{jom}, \forall j \in J, (o, o') \in \mathcal{P}_j, m \in M_{jo}, m' \in M_{jo'} \tag{22}$$

$$\theta_{joo'mm'} \leq X_{jo'm'}, \forall j \in J, (o, o') \in \mathcal{P}_j, m \in M_{jo}, m' \in M_{jo'} \tag{23}$$

$$\theta_{joo'mm'} \geq X_{jom} + X_{jo'm'} - 1, \forall j \in J, (o, o') \in \mathcal{P}_j, m \in M_{jo}, m' \in M_{jo'} \tag{24}$$

$$z_{jo,j'o',m} + z_{j'o',jo,m} \leq X_{jom}, \forall m \in M, (j, o), (j', o') \in \mathcal{O}_m, (j, o) \neq (j', o') \tag{25}$$

$$z_{jo,j'o',m} + z_{j'o',jo,m} \leq X_{j'o'm}, \forall m \in M, (j, o), (j', o') \in \mathcal{O}_m, (j, o) \neq (j', o') \tag{26}$$

$$z_{jo,j'o',m} + z_{j'o',jo,m} \geq X_{jom} + X_{j'o'm} - 1, \forall m \in M, (j, o), (j', o') \in \mathcal{O}_m, (j, o) \neq (j', o') \tag{27}$$

$$S_{j'o'} \geq C_{jo} - M_0\left(1 - z_{jo,j'o',m}\right), \forall m \in M, (j, o), (j', o') \in \mathcal{O}_m, (j, o) \neq (j', o') \tag{28}$$

$$S_{jo} \geq C_{j'o'} - M_0\left(1 - z_{j'o',jo,m}\right), \forall m \in M, (j, o), (j', o') \in \mathcal{O}_m, (j, o) \neq (j', o') \tag{29}$$

$$\xi_{0,jo,m} + \sum_{\substack{(j'o') \in \mathcal{O}_m \\ (j'o') \neq (j,o)}} \xi_{j'o',jo,m} = X_{jom}, \forall m \in M, (j, o) \in \mathcal{O}_m \tag{30}$$

$$\xi_{jo,E,m} + \sum_{\substack{(j'o') \in \mathcal{O}_m \\ (j'o') \neq (j,o)}} \xi_{jo,j'o',m} = X_{jom}, \forall m \in M, (j, o) \in \mathcal{O}_m \tag{31}$$

$$\sum_{(j,o) \in \mathcal{O}_m} \xi_{0,jo,m} = \sum_{(j,o) \in \mathcal{O}_m} \xi_{jo,E,m} \leq 1, \forall m \in M \tag{32}$$

$$\xi_{jo,j'o',m} \leq z_{jo,j'o',m}, \forall m \in M, (j, o), (j', o') \in \mathcal{O}_m, (j, o) \neq (j', o') \tag{33}$$

$$S_{j'o'} \geq C_{jo} + s_{g_j g_{j'} m} - M_0\left(1 - \xi_{jo,j'o',m}\right), \forall m \in M, (j, o), (j', o') \in \mathcal{O}_m, (j, o) \neq (j', o') \tag{34}$$

$$G_{jo,j'o',m} \geq S_{j'o'} - C_{jo} - s_{g_j g_{j'} m} - M_0\left(1 - \xi_{jo,j'o',m}\right), \forall m \in M, (j, o), (j', o') \in \mathcal{O}_m, (j, o) \neq (j', o') \tag{35}$$

$$G_{jo,j'o',m} \leq S_{j'o'} - C_{jo} - s_{g_j g_{j'} m} + M_0\left(1 - \xi_{jo,j'o',m}\right), \forall m \in M, (j, o), (j', o') \in \mathcal{O}_m, (j, o) \neq (j', o') \tag{36}$$

$$0 \leq G_{jo,j'o',m} \leq M_0 \cdot \xi_{jo,j'o',m}, \forall m \in M, (j, o), (j', o') \in \mathcal{O}_m, (j, o) \neq (j', o') \tag{37}$$

$$X_{jom} \leq \sum_{\substack{f \in F_j: \\ m \in M_f}} \phi_{jf}, \forall j \in J^B, o \in B_j, m \in M_{jo} \tag{38}$$

$$v_{jj'f} + v_{j'jf} \leq \phi_{jf}, \forall j, j' \in J^B, j < j', f \in F_j \cap F_{j'} \tag{39}$$

$$v_{jj'f} + v_{j'jf} \leq \phi_{j'f}, \forall j, j' \in J^B, j < j', f \in F_j \cap F_{j'} \tag{40}$$

$$v_{jj'f} + v_{j'jf} \geq \phi_{jf} + \phi_{j'f} - 1, \forall j, j' \in J^B, j < j', f \in F_j \cap F_{j'} \tag{41}$$

$$S_{j',o_{j'}^{B-}} \geq C_{j,o_j^{B+}} - M_0\left(1 - v_{jj'f}\right), \forall j, j' \in J^B, j < j', f \in F_j \cap F_{j'} \tag{42}$$

$$S_{j,o_j^{B-}} \geq C_{j',o_{j'}^{B+}} - M_0\left(1 - v_{j'jf}\right), \forall j, j' \in J^B, j < j', f \in F_j \cap F_{j'} \tag{43}$$

$$S_j^I \geq C_{j\bar{o}_j} + \sum_{m \in M_{j\bar{o}_j}} \sum_{q \in Q_j} t_{mq}^T \cdot \eta_{jmq}, \forall j \in J \tag{44}$$

$$\eta_{jmq} \leq X_{j\bar{o}_j m}, \forall j \in J, m \in M_{j\bar{o}_j}, q \in Q_j \tag{45}$$

$$\eta_{jmq} \leq y_{jq}, \forall j \in J, m \in M_{j\bar{o}_j}, q \in Q_j \tag{46}$$

$$\eta_{jmq} \geq X_{j\bar{o}_j m} + y_{jq} - 1, \forall j \in J, m \in M_{j\bar{o}_j}, q \in Q_j \tag{47}$$

$$C_j = S_j^I + \sum_{q \in Q_j} p_{jq}^I \cdot y_{jq}, \forall j \in J \tag{48}$$

$$u_{jj'q} + u_{j'jq} \leq y_{jq}, \forall j, j' \in J, j < j', q \in Q_j \cap Q_{j'} \tag{49}$$

$$u_{jj'q} + u_{j'jq} \leq y_{j'q}, \forall j, j' \in J, j < j', q \in Q_j \cap Q_{j'} \tag{50}$$

$$u_{jj'q} + u_{j'jq} \geq y_{jq} + y_{j'q} - 1, \forall j, j' \in J, j < j', q \in Q_j \cap Q_{j'} \tag{51}$$

$$S_{j'}^I \geq C_j - M_0\left(1 - u_{jj'q}\right), \forall j, j' \in J, j < j', q \in Q_j \cap Q_{j'} \tag{52}$$

$$S_j^I \geq C_{j'} - M_0\left(1 - u_{j'jq}\right), \forall j, j' \in J, j < j', q \in Q_j \cap Q_{j'} \tag{53}$$

$$T_j \geq C_j - d_j, \forall j \in J \tag{54}$$

$$T_j \geq 0 \tag{55}$$

$$x_{jomk}, \phi_{jf}, y_{jq}, z_{jo,j'o',m}, \xi_{jo,j'o',m}, \xi_{0,jo,m}, \xi_{jo,E,m} v_{jj'f}, u_{jj'q}, \theta_{joo'mm'}, \eta_{jmq} \in \{0,1\}$$
for all valid indices. (56)

$$S_{jo}, C_{jo}, S_j^I, C_j, T_j, G_{jo,j'o',m} \geq 0 \tag{57}$$

Objectives (14) and (15) minimize total weighted tardiness and carbon emissions, respectively. Constraints (16)-(18) assign one machine-mode pair to each production operation, one compatible fixture to each datum-retention block, and one eligible inspection device to each job. Constraint (19) defines operation completion times. Constraints (20)-(24) enforce release times, technological precedence, inter-machine transport, and the machine-pair indicators for consecutive operations. Constraints (25)-(29) establish the relative order and non-overlap of operations assigned to the same machine. Constraints (30)-(34) identify adjacent machine tasks and impose sequence-dependent setup times. Constraints (35)-(37) define internal machine idle time. Constraint (38) enforces machine-fixture compatibility, while Constraints (39)-(43) prevent overlapping occupation of a shared fixture. Constraints (44)-(53) model transport to final inspection, inspection completion, and non-overlap on inspection devices. Constraints (54) and (55) define job tardiness, and Constraints (56) and (57) specify the variable domains.

4. Solution Approach

The model in Section 3 combines operation sequencing, machine-mode selection, fixture

allocation, sequence-dependent setup, transport, and mandatory final inspection. Exact solution becomes costly as the numbers of jobs and candidate resources increase. HILS-NSGA-II uses the NSGA-II framework with a hybrid initial population and Pareto local search on OS-MMS-FPA offspring. A common decoder enforces precedence, machine capacity, mode eligibility, continuous fixture occupation, setup, transport, and final inspection. The following subsections describe the representation, decoding, initialization, genetic operators, local search, environmental selection, and archive.

4.1 HILS-NSGA-II Framework

Figure 3 outlines HILS-NSGA-II. The search begins with hybrid initialization and evaluation, then alternates offspring generation, local search, environmental selection, and archive update. Fast nondominated sorting, crowding distance, and elitist selection follow NSGA-II. The encoding, decoder, initialization, and local search are tailored to the resources and decisions considered here.

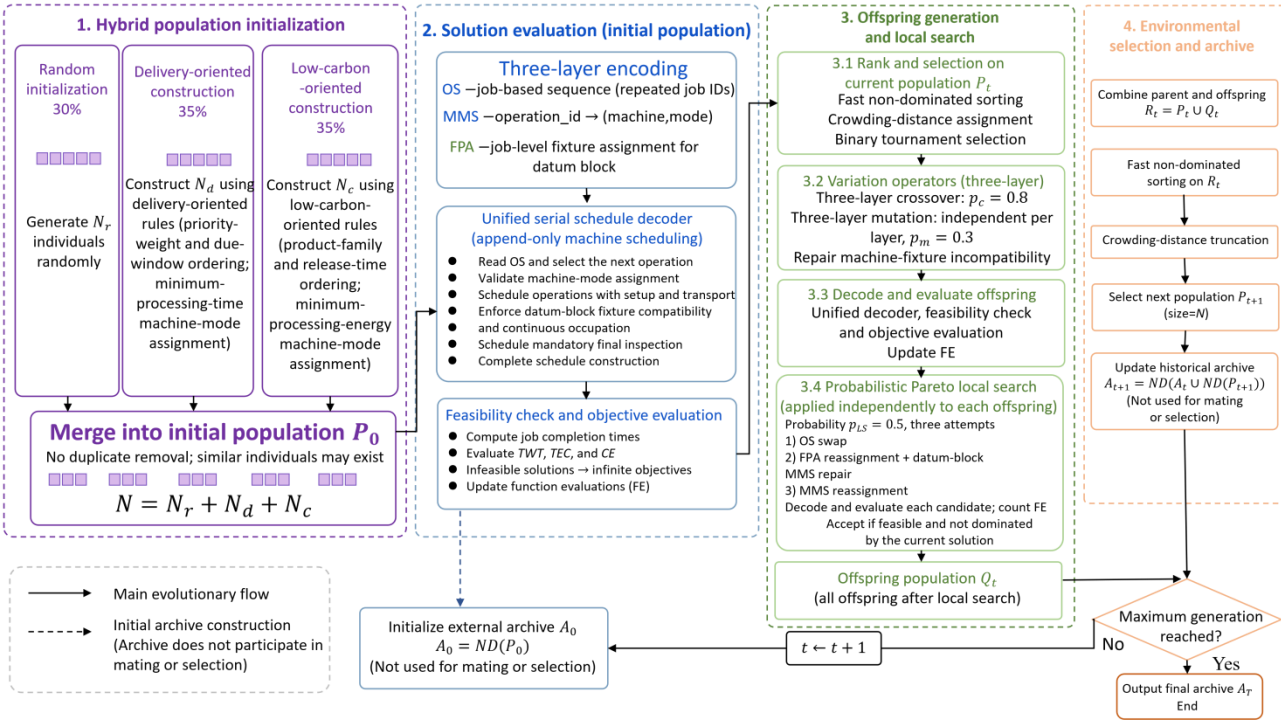


Fig. 3. Framework of HILS-NSGA-II

The initial population contains 30% random individuals, 35% delivery-oriented individuals, and 35% low-carbon-oriented individuals. Random individuals use randomly generated OS, MMS, and FPA layers. Delivery-oriented construction gives priority to orders under greater delivery pressure and favors short machine-mode assignments. Low-carbon construction groups jobs by product family and release time and favors low-energy assignments. The three groups are merged into the initial population P_0. Chromosome duplicates are not removed, so deterministic construction may produce identical or similar individuals. The purpose is to provide starting points with different objective preferences, not to claim greater diversity.

Each individual has three layers. OS uses repeated job identifiers to define the operation-selection order. MMS maps each operation to a machine-mode combination. FPA assigns a fixture to each job-level datum-retention block. The unified serial decoder reads OS in order, selects the next unscheduled operation of the indicated job, and validates its machine-mode assignment. It then

applies append-only machine scheduling while enforcing setup, transport, fixture compatibility and continuous occupation, and mandatory final inspection.

After decoding the initial population, the algorithm calculates job completion times, total weighted tardiness, total energy consumption, and carbon emissions. If a chromosome cannot produce a feasible schedule, its objective values are set to infinity.

Strict nondominance filtering creates the initial historical archive. The archive stores historical nondominated schedules but is not used for parent selection, crossover, or mutation. At generation t, fast nondominated sorting and crowding distance are calculated for the current population P_t. Binary tournaments select the parents. Three-layer crossover and mutation then create offspring. The crossover probability is $p_c = 0.8$ and the mutation probability is $p_m = 0.3$. Mutation is tested independently on OS, MMS, and FPA. Because MMS and FPA can be inherited separately, machine-fixture compatibility within each datum-retention block is checked after variation, and incompatible machine choices are repaired.

Every offspring is decoded and evaluated before local search. Pareto local search is triggered independently with probability $p_{LS} = 0.5$. It makes at most three attempts: an OS swap, job-level FPA reassignment with MMS repair, and operation-level MMS reassignment. Every candidate is decoded, evaluated, and counted in FE. A feasible candidate is accepted if the current solution does not dominate it. The resulting individuals form the offspring population Q_t.

Environmental selection first combines the parent and offspring populations:

$$R_t = P_t \cup Q_t \tag{58}$$

Fast nondominated sorting is applied to R_t. Nondomination rank and crowding distance are then used to select a fixed-size next population P_{t+1}. The historical nondominated archive is updated as follows:

$$A_{t+1} = ND\big(A_t \cup ND(P_{t+1})\big) \tag{59}$$

If the generation limit has not been reached, set $t \leftarrow t + 1$ and use P_{t+1} as the new current population. Evolution returns to ranking and selection; initialization and construction of A_0 are not repeated. The algorithm stops when $t = T_{max} = 1000$ and outputs the final archive A_T.

4.2 Solution Representation and Decoding Procedure

The chromosome stores the decisions that evolutionary operators can change, while the decoder builds a timed schedule that respects precedence and resource constraints. A single sequence is not enough to represent operation order, machine-mode choice, and fixture allocation. HILS-NSGA-II therefore uses OS, MMS, and FPA layers. Figure 4 illustrates the three-layer representation and decoding process for three jobs and eight production operations. Figure 5 gives the resulting feasible schedule.

A complete chromosome is written as

$$C = (C^{OS}, C^{MMS}, C^{FPA}) \tag{60}$$

where C^{OS} is the job-based operation-sequence layer, C^{MMS} is the operation-level machine-mode layer, and C^{FPA} is the job-level fixture-allocation layer. These layers store the scheduling order, processing-resource configuration, and datum-retention resource configuration. Start times, completion times, and occupation intervals are generated by the serial decoder rather than encoded directly.

OS represents production-operation order by repeated job identifiers. The kth occurrence of a job identifier refers to the kth unscheduled production operation of that job. Its number of occurrences equals the number of production operations, so OS length equals the total number of production

operations. Final inspection is not an OS gene. The decoder schedules it as the mandatory last task after all production operations of a job have finished.

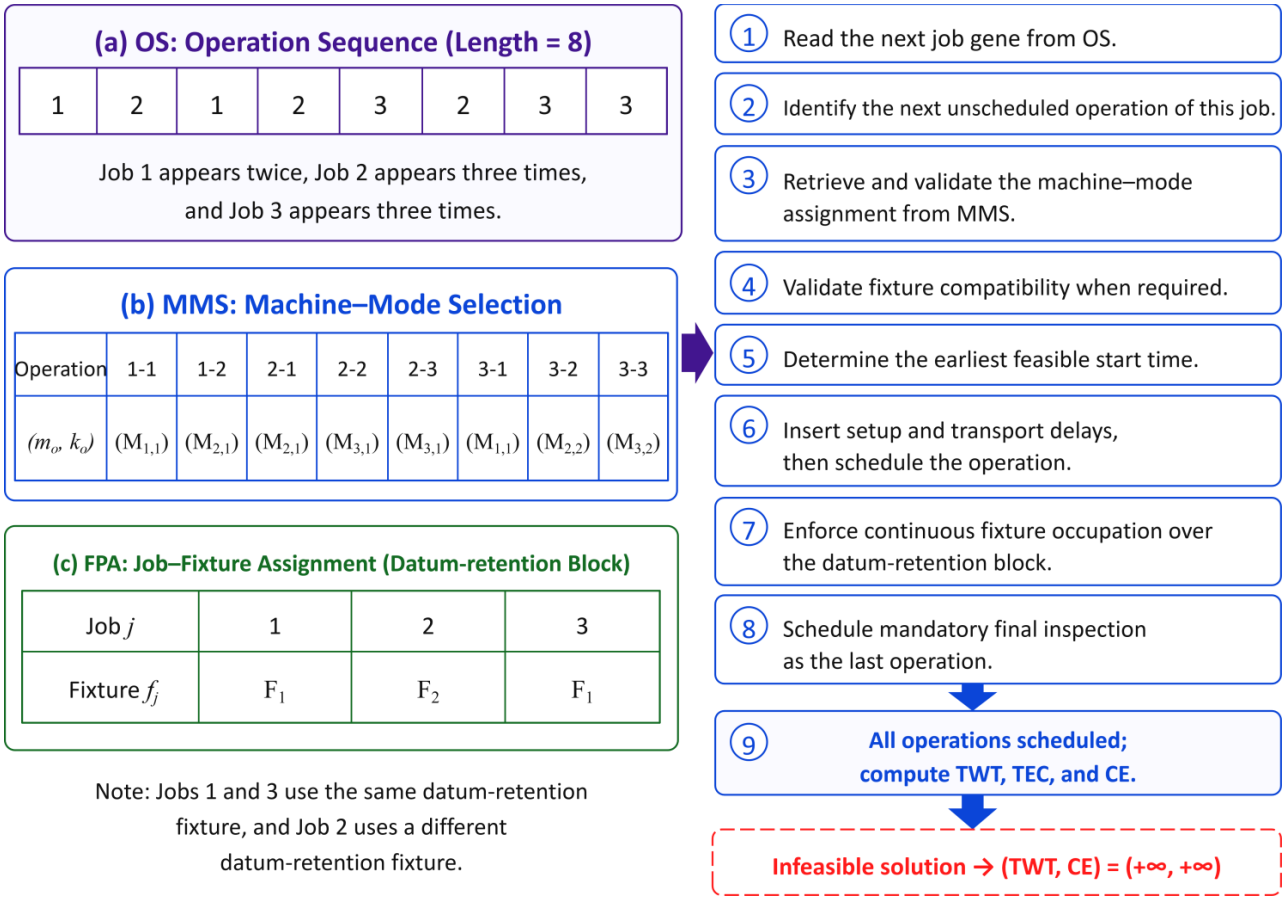


Fig. 4. Three-layer chromosome and unified serial decoding process

The OS in Figure 4 is

$$C^{OS} = [1,2,1,2,3,2,3,3] \tag{61}$$

Job 1 appears twice and refers to operations1-1 and 1-2. Job 2 appears three times and refers to operations 2-1, 2-2, and 2-3. Job 3 appears three times and refers to operations 3-1, 3-2, and 3-3. OS length therefore equals the number of production operations. Final inspection is scheduled after all production operations of a job.

MMS is indexed by operation and stores the resource-mode assignment of each production, cleaning, and final-inspection task. Ordinary operations use a production machine and processing mode. Cleaning and inspection use the relevant resource and normal mode. For job j, its oth operation O_{jo} has the following MMS gene:

$$C_{jo}^{MMS} = (m_{jo}, k_{jo}) \tag{62}$$

where m_{jo} is the production machine or inspection resource assigned to operation o_{jo}, and k_{jo} is the processing mode used on resource m_{jo}. For ordinary production operations, m_{jo} is a production machine and k_{jo} is an eligible processing mode. Cleaning and final inspection allow normal mode only. Storing the resource and mode together enables the decoder to obtain processing time, operating power, and the related energy parameters from (m_{jo}, k_{jo}).

For the three-job example in Figures 4 and 5, the complete MMS is

$$C^{MMS} = \{1-1\colon (M_1,1), 1-2\colon (M_2,1), I_1\colon (Q_1,N), \\ 2-1\colon (M_2,1), 2-2\colon (M_3,1), 2-3\colon (M_3,1), I_2\colon (Q_1,N), \\ 3-1\colon (M_1,1), 3-2\colon (M_2,1), 3-3\colon (M_3,1), I_3\colon (Q_1,N)\} \tag{63}$$

where I_j is the final-inspection task of job j; Q_1 is an inspection resource; and N is normal mode. The production-operation part of the MMS corresponds to Figure 4. The final-inspection assignments I_1, I_2, and I_3 are stored separately in the same MMS structure and are used when the decoder schedules the mandatory inspection tasks. Operation 3-2 uses machine M_2 in mode 2, operation 3-3 uses machine M_3 in mode 2, and final-inspection task I_3 uses resource Q_1 in normal mode. The decoder checks whether m_{jo} is eligible for operation o_{jo} and whether k_{jo} belongs to its permitted mode set. For an operation in a datum-retention block, the selected machine must also be compatible with the FPA fixture. An ineligible machine or inspection resource, an invalid mode, or a machine-fixture conflict makes the assignment invalid and the chromosome infeasible. The decoder does not choose a replacement machine, mode, or fixture at this stage.

FPA assigns fixtures at job level and has the form

$$j \to f_j \tag{64}$$

where f_j is the fixture assigned to the datum-retention block of job j. FPA applies only to jobs with such a block. It means that fixture f_j is used continuously across that block, not across every production operation of the job.

The FPA in Figure 4 is

$$C^{FPA} = \{1\colon F_1, 2\colon F_2, 3\colon F_1\} \tag{65}$$

Jobs 1 and 3 use fixture F_1, while Job 2 uses fixture F_2. Their occupation intervals on F_1 cannot overlap. Job 3 can use it only after Job 1 completes its datum-retention block and releases F_1.

MMS stores machine-mode choices, whereas FPA stores fixture assignments. Before decoding, the algorithm checks machine eligibility, mode eligibility, and machine-fixture compatibility inside each datum-retention block. The unified decoder validates these configurations but does not select replacement resources. An ineligible machine, invalid mode, or machine-fixture conflict makes the chromosome infeasible.

After the three layers are built, the serial decoder follows the job-gene order in OS. It initializes the availability of production machines, inspection resources, and fixtures and records the next production operation and predecessor completion time of every job. Each job gene selects that job's next unscheduled production operation. After all production operations of a job are complete, the decoder schedules final inspection using the inspection resource-mode assignment stored in MMS.

For the current operation, the decoder reads the resource-mode pair from MMS and checks resource and mode eligibility. If the operation belongs to a datum-retention block, it also reads the fixture from FPA and checks job-fixture and machine-fixture compatibility. A failed check makes the chromosome infeasible; the decoder does not replace the resource.

An operation start time depends on job readiness, machine availability, sequence-dependent setup, inter-device transport, and fixture occupation. For operation O_{jo}, its earliest feasible start time is summarized as

$$S_{jo} = max\{R_{jo}^{pre}, R_{jo}^{mach}, R_{jo}^{fix}\} \tag{66}$$

where R_{jo}^{pre} is the operation-ready time determined by job release, predecessor completion, and required transport; R_{jo}^{mach} is the selected machine's availability after its current last task and any required setup; and R_{jo}^{fix} is the earliest time at which the selected fixture can start or continue the

current datum-retention block. Fixture availability is omitted from Eq. (66) for operations outside such a block.

The decoder uses append-only machine scheduling. A new operation is placed only at the end of the selected machine's current sequence. Existing idle intervals are not searched, and the operation is not inserted between scheduled tasks. If the current last task and the new task belong to different product families, sequence-dependent setup is inserted before the new operation. Setup occupies the machine but is not encoded as a separate OS or MMS operation.

The OS reading order in Figure 4 is

$$1-1,2-1,1-2,2-2,3-1,2-3,3-2,3-3$$

The decoder reads these operations in order and uses MMS and FPA to obtain their machine, mode, and fixture assignments. Append-only scheduling then produces the feasible schedule shown in Figure 5.

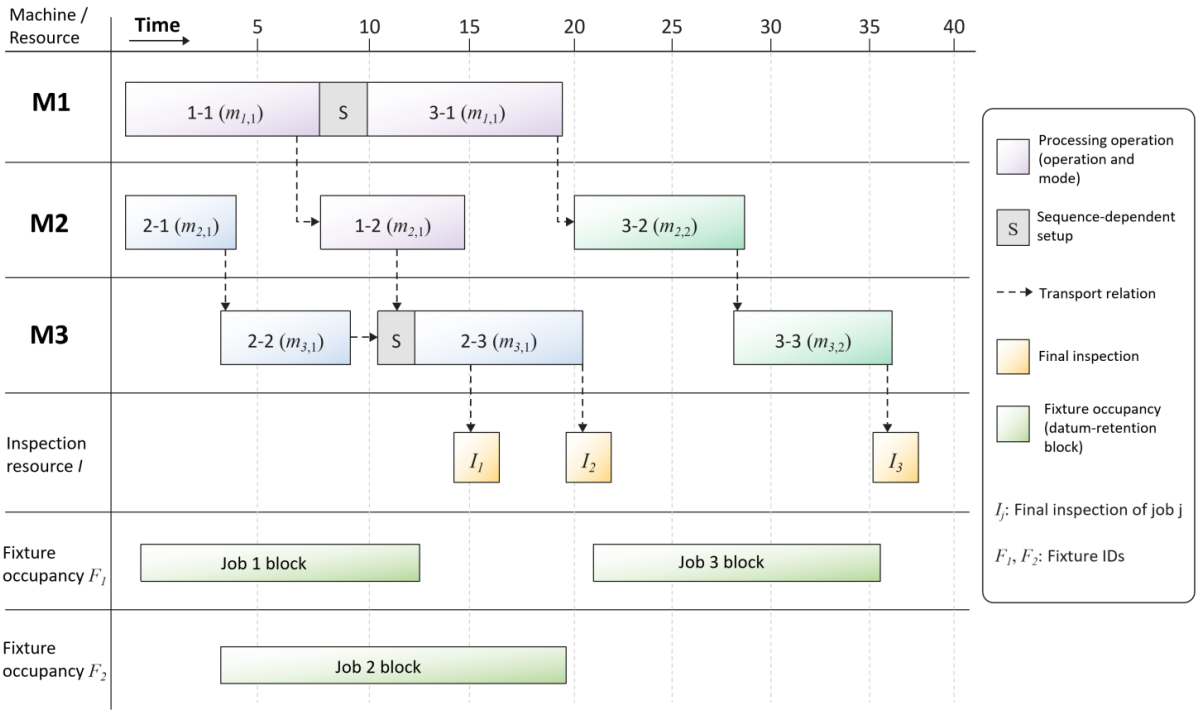


Fig. 5. Illustrative feasible schedule generated by the unified serial decoder

In Figure 5, operation 1-1 of Job 1 uses machine M_1 in mode 1. After transport, operation 1-2 uses machine M_2 in mode 1 and is followed by final inspection I_1. Its route is

$$1-1 \rightarrow 1-2 \rightarrow I_1$$

Operation 2-1 of Job 2 first uses machine M_2 and is transported to machine M_3 for operation 2-2. Operations 2-2 and 2-3 both use machine M_3. After the required setup, operation 2-3 is followed by final inspection I_2. Its route is

$$2-1 \rightarrow 2-2 \rightarrow 2-3 \rightarrow I_2$$

Operations 3-1, 3-2, and 3-3 of Job 3 use machinesM_1,M_2, and M_3, with modes 1, 2, and 2. Consecutive operations satisfy precedence and transport requirements. Final inspection I_3 follows the last production operation. The complete route is

$$3-1 \rightarrow 3-2 \rightarrow 3-3 \rightarrow I_3$$

In Figure 5, Jobs 1 and 3 share fixture F_1. Their occupation intervals do not overlap. Job 3 starts only after Job 1 releasesF_1. Job 2 uses fixtureF_2. Each bar covers only the relevant datum-retention interval.

Final inspection is the mandatory last task in every job route, but it is not represented by an OS gene in Figure 4. Once all production operations of a job are complete, the decoder reads the preassigned inspection resource-mode pair from MMS. Inspection cannot start before the last production operation and required transport are complete, and the selected inspection resource must be available. The final-inspection start time of job j is

$$S_j^I = max\left\{C_{j\bar{o}_j} + t^T_{m_{j\bar{o}_j},q}, A^I_q\right\} \tag{67}$$

where $\bar{o}_j$ is the last production operation of job j; $C_{j\bar{o}_j}$ is the completion time of job j's last operation; $m_{j\bar{o}_j}$ is its production machine; $t^T_{m_{j\bar{o}_j},q}$ is the transport time to inspection resource q; A^I_q is the availability time of inspection resource q; final-inspection completion is the actual completion time of job j.

After each task, the decoder updates machine or inspection-resource availability, the product family last processed on the machine, current job completion, and fixture occupation. A fixture is occupied from the start of the first operation in a datum-retention block until the last block operation finishes. After all OS operations have been scheduled, the decoder checks for omissions and calculates the following measures from the complete timetable:

$$TWT, CE$$

The optimization objectives are TWT and CE, where CE is calculated from processing, final-inspection, sequence-dependent setup, internal machine idle, and inter-device transport energy.

If the decoder finds an invalid machine-mode assignment, machine-fixture conflict, invalid fixture, unknown OS job, or omitted operation, the chromosome is infeasible. The decoder does not replace resources. Its objective values are

$$(TWT, CE) = (+\infty, +\infty) \tag{68}$$

Feasible individuals always outrank infeasible ones, so no extra penalty coefficient is required. Figure 4 summarizes the chromosome and serial decoder, and Figure 5 illustrates a feasible decoded schedule. The next subsection explains the three initialization rules.

4.3 Hybrid Population Initialization

Initialization shapes the early search. Random construction produces varied starting points but offers no quality control, whereas one deterministic rule may concentrate the population in a narrow objective region. The initial population therefore mixes random, delivery-oriented, and low-carbon-oriented constructions.

Let the population size be N. The three proportions are ρ_R,ρ_D, ρ_C and satisfy:

$$\rho_R + \rho_D + \rho_C = 1 \tag{69}$$

The corresponding numbers of individuals are, N_R,N_D, and N_C, satisfying:

$$N_R + N_D + N_C = N \tag{70}$$

When a theoretical count is not an integer, its integer part is retained first. The remaining positions are assigned in descending order of fractional remainder. Ties follow the fixed order random, delivery-oriented, and low-carbon-oriented. All three groups use the OS-MMS-FPA representation described in Section 4.2.

Random initialization introduces no objective preference. For OS, the algorithm creates repeated job identifiers according to the number of operations and shuffles the sequence. The kth occurrence of a job still refers to its kth unscheduled operation, so technological order is preserved. FPA selects a compatible fixture for each job at random. MMS then selects a feasible resource-mode pair for every operation. Within a datum-retention block, a candidate machine must be both eligible for the

operation and compatible with the selected fixture. Cleaning and final inspection use normal mode only.

Delivery-oriented initialization favors lower total weighted tardiness. Jobs are sorted by descending tardiness weight. Ties are resolved by ascending due-date slack $d_j - r_j$ and then by job identifier. OS follows this job order. FPA selects the first fixture in each job's compatible list. MMS chooses the feasible resource-mode pair with the shortest processing time. A tie is resolved by lower direct processing energy.

Low-carbon-oriented initialization favors lower direct processing energy. Jobs are sorted by product family, release time, and job identifier before OS is formed. This keeps jobs from the same family relatively close but adds no load-balancing rule or random perturbation. FPA selects the first compatible fixture. For each operation, MMS minimizes processing time multiplied by processing power over all feasible resource-mode pairs. Ties are resolved by shorter processing time.

The three initialization methods construct only OS, MMS, and FPA. They do not set start or completion times. The common serial decoder validates resource configurations, builds schedules, and evaluates objectives. It checks machine and mode eligibility, job-fixture compatibility, and machine-fixture compatibility, but it does not replace an invalid resource during decoding. A chromosome that contains an invalid assignment or cannot schedule every operation is marked infeasible and receives inferior objective values.

Initialization neither removes duplicate chromosomes nor regenerates infeasible individuals. Deterministic delivery- or low-carbon-oriented rules may therefore create identical chromosomes, and random construction may produce similar ones by chance. All individuals are decoded and evaluated directly to form the initial population:

$$P_0 = P_0^R \cup P_0^D \cup P_0^C \tag{71}$$

where P_0^R,P_0^D and P_0^C denote the random, delivery-oriented, and low-carbon-oriented sets. Strict nondominance filtering then creates the initial historical archive:

$$A_0 = ND(P_0) \tag{72}$$

where $ND(\cdot)$ denotes strict nondominance filtering. The mixed population becomes the first HILS-NSGA-II population. The three rules provide different initial preferences, but no claim is made that they remove duplicates or increase population diversity.

4.4 Genetic Operators

Genetic operators recombine and perturb the three-layer chromosome. This allows operation order, resource-mode assignments, and fixture allocation to change during evolution. Each layer uses operators suited to its data structure.

Parents are selected by binary tournament. Two individuals are sampled at random. The individual with the lower nondomination rank is preferred; if their ranks are equal, the one with the larger crowding distance is selected. The selected parents undergo three-layer crossover with probability p_c. If crossover is not triggered, they are copied and then passed to mutation.

OS uses job-subset precedence-preserving crossover. Each job enters a selected subset with probability 0.5. For the first child, genes of the selected jobs retain their positions from Parent 1. The remaining positions are filled in the relative order of unselected-job genes in Parent 2. Parent roles are reversed for the second child. Job occurrence counts and OS length are unchanged. Because the kth occurrence of a job always refers to its kth unscheduled operation, internal technological order is preserved.

MMS uses operation-level uniform crossover. For every operation, the child inherits the resource-

mode gene from either parent with equal probability. Machine, inspection-resource, and mode assignments are recombined without changing operation indices.

FPA uses job-level uniform crossover. Each fixture gene is inherited from either parent with equal probability. Because MMS and FPA are inherited separately, a machine in a datum-retention block may conflict with the inherited fixture. The fixture is retained, and a machine is selected at random from the intersection of the operation's eligible machines and the fixture-compatible machines. The mode is unchanged. This repair addresses only machine-fixture conflicts created by crossover.

OS, MMS, and FPA share mutation probability p_m, but each layer tests it independently. An offspring may therefore mutate in none, one, two, or all three layers.

When OS mutation is triggered, two distinct positions are selected at random and their job identifiers are exchanged. The identifiers may be equal, so the mutation can be a no-op. No insertion mutation is used, and job occurrence counts remain unchanged.

When FPA mutation is triggered, one job is selected at random and receives a randomly chosen compatible fixture. The new fixture may equal the current one. Machines are then reselected for every operation in that job's datum-retention block. Each candidate must be eligible for the operation and compatible with the fixture; the processing mode is retained. The mutation can be a no-op if the selected fixture and machines match the current configuration.

When MMS mutation is triggered, one operation is selected at random. A machine or inspection resource is drawn from its eligible set, and a mode is drawn from its permitted modes. For a datum-retention operation, the machine must also be compatible with the current FPA fixture. The new resource-mode pair may equal the old one, so MMS mutation can also be a no-op. Cleaning and final inspection allow normal mode only.

Table 3 summarizes the genetic operators used for the three chromosome layers.

Table 3
Genetic operators for the three chromosome layers

Encoding layer	Crossover operator	Mutation operator	Feasibility treatment
OS	Job-based precedence-preserving crossover	Random position-swap mutation	The number of occurrences of each job ID is preserved
MMS	Operation-level uniform crossover	Random feasible resource-mode reassignment	The resource and mode are selected from the eligible sets
FPA	Job-level uniform crossover	Random compatible-fixture reassignment	Machine assignments in the datum-retention block are updated according to the selected fixture

After crossover and mutation, every offspring is scheduled and evaluated by the unified serial decoder in Section 4.2. The decoder checks machine eligibility, mode eligibility, fixture assignment, and machine-fixture compatibility, but performs no general resource repair. An invalid resource assignment, unknown job identifier, or missing operation makes the offspring infeasible and gives it inferior objective values. The algorithm does not reassign fixtures, restore parents, select new parents, or repeat the genetic operators at this stage.

These operators alter operation order, resource-mode choices, and fixture allocation at the population level. Probabilistic Pareto local search is then applied to selected decoded offspring.

4.5 Pareto Local Search

Pareto local search is applied after offspring decoding and objective evaluation. Each offspring triggers the local search independently with probability p_L. When it is triggered, three search attempts are performed in sequence.

Each attempt applies a composite perturbation to the current solution. The perturbation contains three operations. First, two positions in the OS layer are selected and exchanged. Second, one job is selected in the FPA layer. A compatible fixture is assigned to this job. The machine choices of the operations in its datum-retention block are then repaired according to the new fixture. Third, one operation is selected in the MMS layer. A feasible machine and an admissible processing mode are assigned to this operation. The selected gene or resource may be the same as its current value. Therefore, a perturbation may occasionally produce no effective change.

Each candidate is decoded by the unified serial decoder. Its feasibility, TWT, and CE are then evaluated. Each candidate evaluation is counted as one function evaluation. Let x be the current solution and x' be the candidate. The candidate is accepted when it is feasible and is not dominated by the current solution:

$$x \leftarrow x', if\ Feasible(x') \wedge F(x) \prec F(x') \tag{73}$$

This rule accepts a candidate that dominates the current solution. It also accepts a candidate that is mutually nondominated with the current solution. A candidate with the same objective values can also be accepted. Once a candidate is accepted, it becomes the starting solution for the next attempt.

The local search does not identify critical operations, bottleneck machines, or tardy jobs. It does not use a separate local archive. It also does not apply crowding-distance selection or an early-stopping rule. Its purpose is to introduce additional local perturbations into the offspring population while preserving different trade-offs between TWT and CE.

4.6 Historical Nondominated Archive

A historical nondominated archive is maintained to preserve nondominated solutions found during the search. After the initial population P_0 is decoded and evaluated, the archive is initialized according to Eq. (72).

At the end of generation t, the parent and offspring populations are combined. Environmental selection produces the next population P_{t+1}. The historical archive is then updated by

$$A_{t+1} = ND(A_t \cup ND(P_{t+1})) \tag{74}$$

During the update, dominated solutions are removed. Duplicate objective points are also removed. Only one solution is retained when several solutions have identical TWT and CE values.

The archive has no predefined capacity. It is not truncated by crowding distance. It does not participate in binary tournament selection, crossover, mutation, or environmental selection. The evolutionary process is driven only by the current population. The archive is used to preserve historical nondominated solutions and to produce the final oIts two added search components are P3 hybrid initialization and probabilistic Pareto local search.utput. When the maximum number of generations is reached, the final archive A_T is returned as the Pareto solution set obtained by the algorithm.

4.7 HILS-NSGA-II Procedure

HILS-NSGA-II keeps the fast nondominated sorting, crowding distance, binary tournament, and elitist environmental selection of NSGA-II. Its two added search components are the hybrid initialization strategy described in Section 4.3 and probabilistic Pareto local search.

The initial population contains three types of solutions. Random solutions account for 30% of the population. Delivery-oriented solutions account for 35%. Low-carbon-oriented solutions account for the remaining 35%. Every individual contains an OS layer, an MMS layer, and an FPA layer. All individuals are evaluated by the same serial schedule decoder.

At the beginning of each generation, the current population is ranked by fast nondominated

sorting. The crowding distance of each individual is then calculated. Two parents are selected through binary tournaments. An individual with a lower nondominated rank is preferred. If two individuals have the same rank, the one with a larger crowding distance is preferred.

Three-layer crossover is performed with probability p_c. Otherwise, the two parents are copied. The OS, MMS, and FPA layers share the same mutation probability p_m. However, mutation is triggered independently for each layer. Therefore, a child may experience no mutation, mutation in one layer, or mutation in several layers.

After mutation, all offspring are decoded and evaluated. Pareto local search is then independently applied to each offspring with probability p_{LS}. The parent and offspring populations are combined after the local search. Fast nondominated sorting and crowding-distance truncation are used to select the next population. Finally, the historical archive is updated. This process continues until G_{max} generations have been completed.

HILS-NSGA- Ⅱ

Input: Problem instance Ω; population size N; maximum number of generations G_{max}; crossover probability p_c; mutation probabilities p_m; local-search probability p_{LS}; Number of local-search attempts $L = 3$.

Set $t \leftarrow 0$ and $FE \leftarrow 0$
Generate the initial population P_0 using the hybrid initialization strategy in Section 4.3:
  30% random solutions
  35% delivery-oriented solutions
  35% low-carbon-oriented solutions
for each individual x in P_0 do
  Decode x using the unified serial decoder
  Evaluate $TWT(x)$ and $CE(x)$
  $FE \leftarrow FE + 1$
end for
$A_0 \leftarrow ND(P_0)$
while $t < G_{max}$ do
  Perform fast nondominated sorting on P_t
  Calculate the crowding distance of each individual
  $Q_t \leftarrow \emptyset$
  while $|Q_t| < N$ do
    Select two parents by binary tournament selection
    if $rand() < p_c$ then
      Perform three-layer crossover
      Repair machine-fixture incompatibility if required
    else
      Copy the two parents
    end if
    for each generated child c do
      if $rand() < p_m$ then
        Apply OS mutation
      end if
      if $rand() < p_m$ then
        Apply FPA mutation
        Repair the MMS choices in the datum-retention block
      end if
      if $rand() < p_m$ then
        Apply operation-level MMS mutation
      end if
      Add c to Q_t
    end for
  end while

35: Keep the first N offspring if $|Q_t| > N$
36: for each offspring c in Q_t do
37: Decode and evaluate c
38: $FE \leftarrow FE + 1$
39: end for
40: for each offspring c in Q_t do
41: if $rand() < p_{LS}$ then
42: $c \leftarrow \text{ParetoLocalSearch}(\text{c}, \text{L} = 3)$
43: Update FE for all evaluated local candidates
44: end if
45: end for
46: $R_t \leftarrow P_t \cup Q_t$
47: Perform fast nondominated sorting on R_t
48: Calculate crowding distances
49: Select N individuals to form P_{t+1}
50: $A_{t+1} \leftarrow ND\big(A_t \cup ND(P_{t+1})\big)$
51: Remove duplicate objective points from A_{t+1}
52: Record the current generation, TWT, CE, current nondominated-set size, and FE
53: $t \leftarrow t + 1$
54: End while
55: Return $A_{G_{max}}$

Output: Final historical nondominated archive $A_{G_{max}}$.

The environmental selection in Line 49 follows the standard NSGA-II procedure. Complete nondominated fronts are added first. If the last acceptable front cannot be fully included, its individuals are sorted by crowding distance. Individuals with larger crowding distances are selected until the population size reaches N.

The final output contains the nondominated archive, objective values, three-layer encodings, and decoded schedule information. The generation history and cumulative function evaluations are also recorded.

5. Computational Experiments

The experiments examine model solvability, parameter selection, comparative performance, component effects, and sensitivity to production conditions. All compared algorithms use the same instances, population or swarm size, generation limit, decoder, and objective evaluator.

5.1 Experimental Design and Parameter Settings

To evaluate the proposed HILS-NSGA-II, 30 test instances were generated for the precision machining scheduling of new-energy vehicle motor housings. The instances were divided into six scale groups, denoted by S1, S2, M1, M2, L1, and L2, with five independently generated instances in each group. The problem size ranged from 6 jobs and 36 operations to 20 jobs and 140 operations. Table 4 summarizes the principal characteristics of the test instances.

The instance data were generated according to the machining characteristics of motor housings and the parameter-generation practices used in energy-aware flexible job shop scheduling. The processing-time and machine-energy settings were developed with reference to Wu and Sun [18] and Meng *et al.,* [13]. The multiple processing modes and their time-energy trade-offs followed the variable-speed scheduling principles of Wei *et al.,* [20], while the sequence-dependent setup and inter-machine transportation settings were established with reference to Du *et al.,* [21]. The numerical ranges were further adjusted to reflect the operational characteristics of motor-housing precision machining.

Table 4
Characteristics of the test instances

Scale group	Instances	Jobs	Total operations	Processing machines	Inspection machines	Fixtures
S1	EM01, EM07, EM13, EM19, EM25	6	36	5	1	3
S2	EM02, EM08, EM14, EM20, EM26	8	48	6	1	4
M1	EM03, EM09, EM15, EM21, EM27	10	60	7	1	4
M2	EM04, EM10, EM16, EM22, EM28	12	72	8	2	5
L1	EM05, EM11, EM17, EM23, EM29	15	105	10	2	6
L2	EM06, EM12, EM18, EM24, EM30	20	140	12	2	8

Note: The total number of operations includes the mandatory final inspection operations.

Ordinary production operations could be processed under high-speed, normal, and energy-saving modes, whereas cleaning and final inspection operations used only the normal mode. The high-speed mode shortened processing time but increased energy consumption, while the energy-saving mode reduced energy consumption at the cost of a longer processing time. Machine eligibility was determined jointly by processing capability and precision requirements. Fixture compatibility depended on product family, fixture interface, machine interface, and precision level. For each datum-retention block, at least one common feasible fixture-machine path was guaranteed, and the selected fixture remained continuously occupied until the block was completed.

Jobs were classified as urgent, regular, or relaxed, with tardiness weights of 3, 2, and 1, respectively. Release times and due dates were generated according to the estimated job workload and average machine workload. An independent baseline schedule based on the earliest-due-date rule, the earliest available eligible machine, and the normal processing mode was used to calibrate the due dates. The calibration maintained the proportion of tardy jobs between 35% and 70% without using the results of any compared algorithm.

The proposed HILS-NSGA-II was compared with NSGA-II, SPEA2, and MOPSO. Each algorithm was independently executed ten times on every instance using random seeds 201-210. The population or swarm size was fixed at 100, and the stopping criterion was 1000 generations. The complete experiment therefore comprised 1200 independent runs. For HILS-NSGA-II, the crossover probability, mutation probability, and local-search rate were set to 0.8, 0.3, and 0.5, respectively. Its hybrid initialization consisted of 30% randomly generated individuals, 35% delivery-oriented individuals, and 35% low-carbon-oriented individuals. All algorithms used the same instances, random seeds, stopping criterion, feasibility rules, and objective-evaluation procedure to ensure a consistent comparison.

5.2 Validation of the Mathematical Model

To examine the small-scale solvability of the proposed mathematical formulation, four instances of increasing size were solved using Gurobi. For each instance, Gurobi was used to solve the minimum-TWT and minimum-CE endpoint problems and the intermediate Pareto subproblems required to construct the nondominated solution set. The endpoint status, number of certified

nondominated solutions, and maximum optimality gap are reported in Table 5.

Table 5
Gurobi results for model validation

Instance	Size (J/O)	Min-TWT	Min-CE	Certified ND points	Max gap
EX01	3/18	0 (OPT)	19.153 (OPT)	8	0.0021%
EX02	4/24	277 (OPT)	33.247 (OPT)	8	0
EX03	5/30	107 (TL)	45.448 (OPT)	1	100%
EX04	6/36	6 (OPT)	51.578 (OPT)	2	100%

Note: J/O denotes the numbers of jobs and operations, respectively; OPT indicates proven optimality; TL indicates that the time limit was reached; ND denotes nondominated. Max gap is the largest optimality gap among the solved Pareto subproblems.

Both endpoint problems were proven optimal for EX01 and EX02. Each instance yielded eight certified nondominated points, and the maximum optimality gaps were negligible. The small cases verify that the formulation produces feasible schedules and consistent objective values.

The exact problems became markedly harder as size increased. For EX03, Gurobi proved the minimum-CE endpoint but did not prove the minimum-TWT endpoint within the time limit. For EX04, both endpoints were optimal, but only two certified nondominated points were found and several intermediate subproblems retained large gaps. Optimal endpoints alone do not guarantee an exactly constructed Pareto front.

Taken together, the Gurobi runs validate the formulation on small instances and expose the cost of exact Pareto-front construction as jobs, operations, and coupled resources increase. Exact solutions were therefore used only as small-instance benchmarks. The evolutionary algorithms were tested on all six scale groups.

5.3 Performance Indicators

HV and IGD were used to evaluate the nondominated solution sets. HV measures the objective-space coverage dominated by an approximation set. A larger HV is better. IGD measures the distance from an empirical reference front to an approximation set. A smaller IGD is better. The number of nondominated solutions (NPS) was also reported as a supplementary indicator of the size of the final approximation set.

A separate empirical reference front was constructed for each instance and each experimental phase. The final nondominated solutions obtained by all methods under comparison and all random seeds were pooled. Here, a "method" refers to a parameter setting in the tuning experiments, a parameter candidate in the confirmation experiment, or an algorithm in the comparative experiment. Duplicate and dominated objective points were removed:

$$R_i = ND\left(\bigcup_{m\in M}\bigcup_{s\in S} P_{i,m,s}\right) \tag{75}$$

where $P_{i,m,s}$ is the final nondominated set obtained by method m on instance i with seed s. The resulting R_i is an empirical reference front rather than the theoretical Pareto front. Reference fronts were constructed separately for different experimental phases.

TWT and CE have different scales, and their ranges vary across instances. Both objectives were therefore normalized within each instance:

$$\widehat{TWT}_i(x) = \frac{TWT_i(x) - TWT_i^{min}}{TWT_i^{max} - TWT_i^{min}} \tag{76}$$

$$\widehat{\mathrm{CE}}_i(x) = \frac{CE_i(x) - CE_i^{min}}{CE_i^{max} - CE_i^{min}} \tag{77}$$

The bounds were calculated from all final objective points in the corresponding experimental phase. The same bounds were used for all methods on the same instance.

$$r^{HV} = (1.1,1.1) \tag{78}$$

HV was calculated in the normalized objective space. The reference point was set to
For a normalized approximation set $\hat{P}$, HV is defined as

$$HV(\hat{P}) = \lambda\left(\bigcup_{x\in\hat{P}}\left[\widehat{TWT}(x),1.1\right] \times \left[\widehat{CE}(x),1.1\right]\right) \tag{79}$$

where $\lambda(\cdot)$ denotes the area in the two-dimensional objective space. A larger HV indicates better convergence, wider coverage, or both. Since the maximum reference rectangle has an area of $1.1 \times 1.1 = 1.21$, a normalized HV greater than 1 is possible.

IGD was calculated as

$$IGD(\hat{P},\hat{R}) = \frac{1}{|\hat{R}|}\sum_{r\in\hat{R}} \min_{x\in\hat{P}} \|r - x\|_2 \tag{80}$$

A smaller IGD indicates that the approximation set is closer to the empirical reference front and provides better coverage of its objective points.

5.4 Parameter Tuning of HILS-NSGA-II

To determine an effective parameter configuration for HILS-NSGA-II, an $L_9(3^4)$ orthogonal design was adopted. Four factors were considered: the crossover probability p_c, mutation probability p_m, local-search probability p_{LS}, and initialization scheme. The three levels of p_c were 0.7, 0.8, and 0.9. The levels of p_m were 0.1, 0.2, and 0.3, while those of p_{LS} were 0.1, 0.3, and 0.5. Three initialization schemes were examined. P1 consisted of 70% randomly generated individuals, 15% delivery-oriented individuals, and 15% low-carbon-oriented individuals. P2 used proportions of 50%, 25%, and 25%, respectively. P3 used proportions of 30%, 35%, and 35%. The nine orthogonal combinations and their tuning results are presented in Table 6.

The tuning experiment was conducted on six representative instances, T04-T09, covering the six problem scales from S1 to L2. Each parameter combination was independently executed using five common random seeds, namely 51-55. The population size and the maximum number of generations were fixed at 100 and 1000, respectively. Consequently, each parameter combination involved 30 runs. The results of the five seeds were first averaged for each instance, after which the six instance-level means were equally weighted.

Normalized HV was used as the primary evaluation indicator, while normalized IGD was used as the secondary indicator. When the HV difference between two settings was below 0.5%, the setting with the lower IGD was preferred. Function evaluations were considered only when the two quality indicators produced very similar results. The main effects of the four parameters on HV and IGD are illustrated in Figure 6.

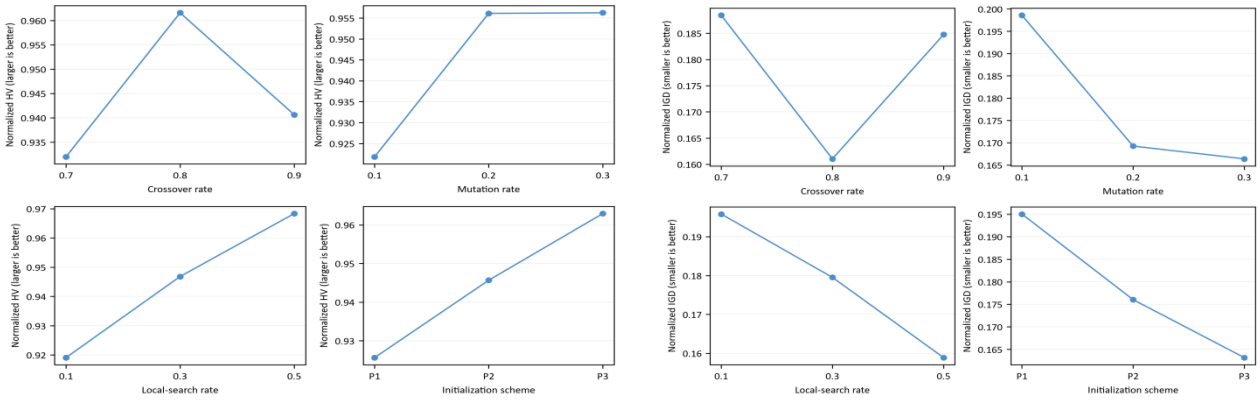
As shown in Table 6, L9-3 achieved the highest observed mean HV and the lowest observed mean IGD among the nine orthogonal combinations. However, the main-effect analysis in Figure 6 shows that the best average performance was obtained with $p_c = 0.8$, $p_m = 0.3$, $p_{LS} = 0.5$ and P3

initialization. Both HV and IGD produced the same preferred parameter configuration.

Table 6
Orthogonal design matrix and tuning results

Run	p_c	p_m	p_{LS}	Initialization	Mean HV	Mean IGD
L9-1	0.7	0.1	0.1	P1	0.864	0.244
L9-2	0.7	0.2	0.3	P2	0.946	0.179
L9-3	0.7	0.3	0.5	P3	0.985	0.143
L9-4	0.8	0.1	0.3	P3	0.959	0.168
L9-5	0.8	0.2	0.5	P1	0.977	0.150
L9-6	0.8	0.3	0.1	P2	0.948	0.165
L9-7	0.9	0.1	0.5	P2	0.942	0.184
L9-8	0.9	0.2	0.1	P3	0.945	0.179
L9-9	0.9	0.3	0.3	P1	0.935	0.191

Note: Larger HV and smaller IGD indicate better performance. P1, P2, and P3 denote the three predefined initialization schemes described above.



(a) Main effects on normalized HV (b) Main effects on normalized IGD.

Fig. 6. Main effects of parameter levels on normalized HV and normalized IGD

The main-effect curves also indicate that the local-search rate had the strongest influence on algorithm performance. Increasing p_{LS} from 0.1 to 0.5 consistently improved HV and reduced IGD. P3 produced better mean HV and IGD values than P1 and P2 under the tested conditions. This result supports the use of a larger proportion of delivery-oriented and low-carbon-oriented individuals in the initial population. The crossover probability had a comparatively smaller effect, with $p_c = 0.8$ providing the best balance. The mutation results favored $p_m = 0.3$, although its HV performance was close to that obtained with $p_m = 0.2$.

The observed best orthogonal combination, the main-effect prediction, and an independent baseline configuration were further evaluated in a confirmation experiment. Six separate instances, T10-T15, were used. These instances covered the six scale groups from S1 to L2. Each candidate was independently tested using five new random seeds, from 61 to 65. The confirmation experiment therefore contained $3 \times 6 \times 5 = 90$ runs. For each candidate, the five runs were first averaged within each instance. The six instance-level means were then equally weighted. Table 7 presents the confirmation results.

Table 7
Results of the independent parameter confirmation

Candidate	p_c	p_m	p_{LS}	Initialization	Mean HV	Mean IGD	Decision
C_0	0.8	0.2	0.3	P2	1.1949	0.0238	-
L9-3	0.7	0.3	0.5	P3	1.1962	0.0188	-
PRED-R	0.8	0.3	0.5	P3	1.1967	0.0187	Selected

PRED-R had the highest mean HV and the lowest mean IGD in Table 7. Relative to C0, HV increased by 0.15% and IGD fell by 21.37%; the differences from L9-3 were smaller. The Friedman test found no significant difference among the three candidates at the 0.05 level. The predefined rule ranked HV first and IGD second, under which PRED-R was selected for the remaining experiments.

The final configuration used $p_c = 0.8$, $p_m = 0.3$, $p_{LS} = 0.5$ and P3 initialization. Under P3, the initial population contained 30% randomly generated individuals, 35% delivery-oriented individuals, and 35% low-carbon-oriented individuals. The population size was 100, and the maximum number of generations was 1,000. A common mutation probability was used for OS, MMS, and FPA. Mutation was triggered independently for each chromosome layer. This configuration was used in all subsequent experiments.

5.5 Comparison of Algorithm Performance

Three established multi-objective algorithms were selected as the baselines. NSGA-II uses fast non-dominated sorting, crowding distance, and elitist environmental selection [11]. SPEA2 uses strength-based fitness assignment, density estimation, and archive truncation [22]. MOPSO applies Pareto dominance and an external archive to particle swarm optimization [23]. These methods represent three common multi-objective search frameworks. HILS-NSGA-II is the proposed algorithm. It combines the NSGA-II framework with hybrid initialization and Pareto local search.

All algorithms were implemented independently. They used the same decoder, objective evaluator, and feasibility rules. Their parameters were fixed before the computational experiments. The baseline algorithms used predefined standard parameter settings. These settings remained unchanged during the experiments.

The comparison covered 30 instances in six scale groups, with five instances per group. Each algorithm was run ten times on every instance, giving 1,200 runs. Every retained solution satisfied the model constraints, and recalculation found no TWT or CE error.

The ten runs were averaged within each algorithm-instance pair. The resulting 30 instance-level means formed the samples for comparison and statistical testing, so every instance received equal weight.

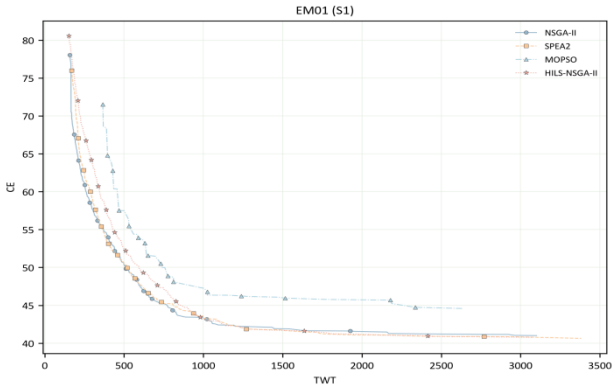
The Friedman test was used to examine the overall differences among the four algorithms. The significance level was set to 0.05. HILS-NSGA-II was then compared with each baseline using the two-sided Wilcoxon signed-rank test. Holm correction was applied to the three pairwise p-values for each indicator. Mean ranks and rank-biserial effect sizes were also reported. Win-tie-loss counts were calculated from the 30 instance-level means. A relative difference below 0.5% for HV or below 2% for IGD was treated as a tie.

Table 8 reports the results across the 30 instances. HILS-NSGA-II obtained a mean HV of 1.1268, which was 3.70% above NSGA-II and 4.28% above SPEA2. Its mean IGD was 0.0845, or 31.36% below NSGA-II and 35.74% below SPEA2. It also returned the largest mean nondominated set. MOPSO recorded the lowest HV and the highest IGD.

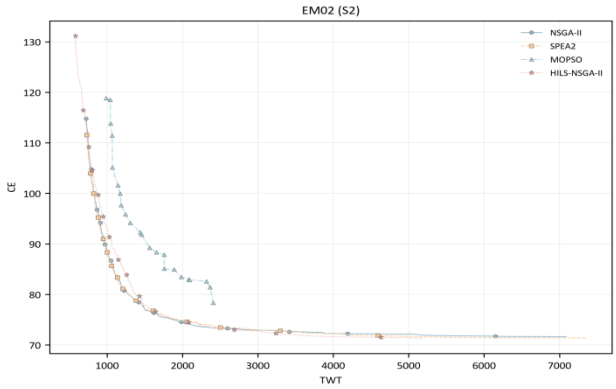
Table 8
Comparative results of the four algorithms

Algorithm	Mean HV	Mean IGD	Mean NPS
NSGA-II	1.0866	0.1231	56.35
SPEA2	1.0806	0.1315	54.63
MOPSO	0.5698	0.5548	10.72
HILS-NSGA-II	1.1268	0.0845	63.22

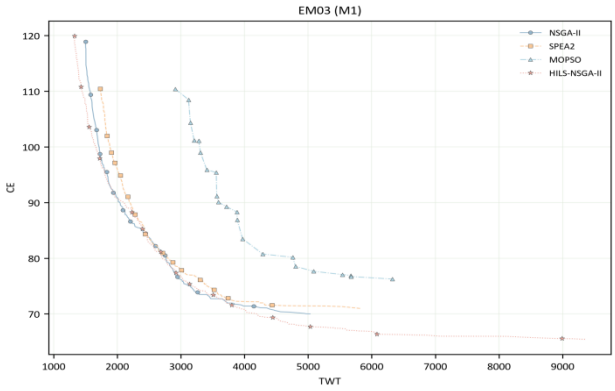
Figure 7 plots the Pareto fronts for EM01-EM06, one instance from each scale group. NSGA-II and SPEA2 remain competitive on several small cases. On most medium and large cases, HILS-NSGA-II spans a wider objective range and reaches lower TWT or CE in parts of the front. MOPSO is generally farther from the lower-left region. Because several fronts overlap, these plots are read together with HV and IGD.



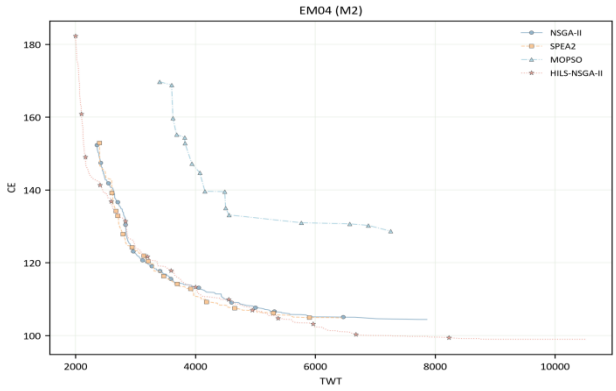
(a) EM01(S1)



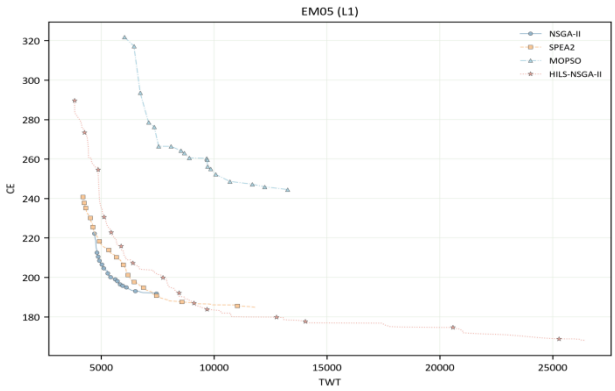
(b) EM02(S2)



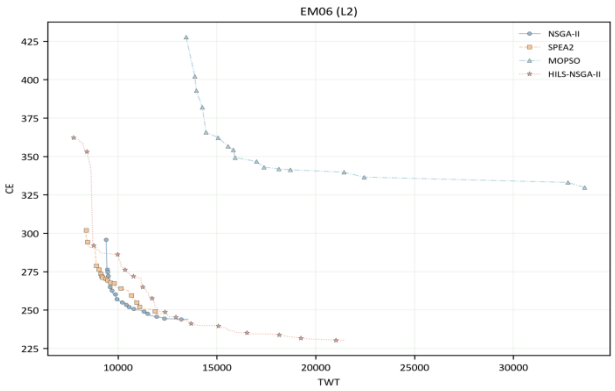
(c) EM03(M1)



(d) EM04(M2)



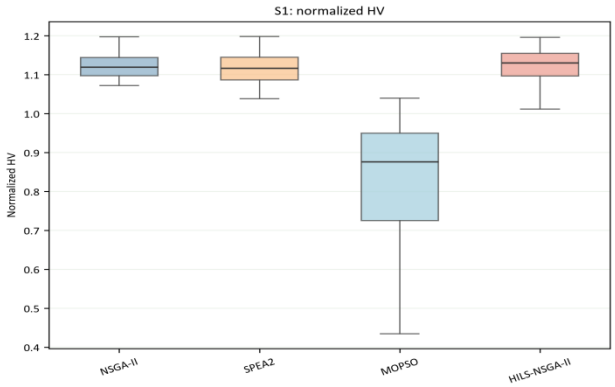
(e) EM05(L1)



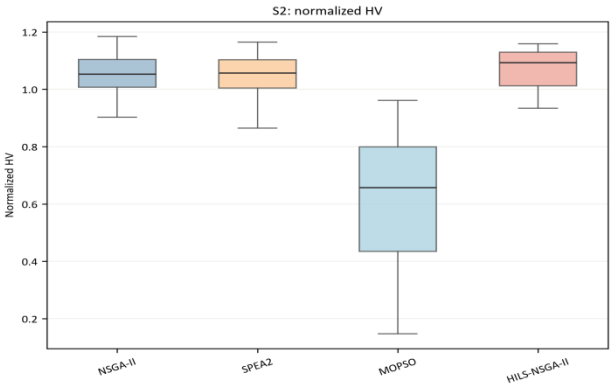
(f) EM06(L2)

Fig. 7. Pareto fronts obtained by the four algorithms on EM01-EM06

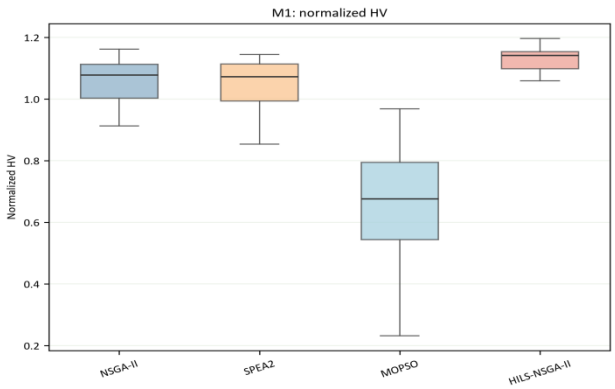
Figure 8 presents normalized HV for the same six representative instances. The boxplots use ten independent runs. HILS-NSGA-II has the highest median on every instance. The gap is modest on EM01 and EM02 but widens on the medium and large cases. MOPSO usually has both the lowest HV and the widest spread.



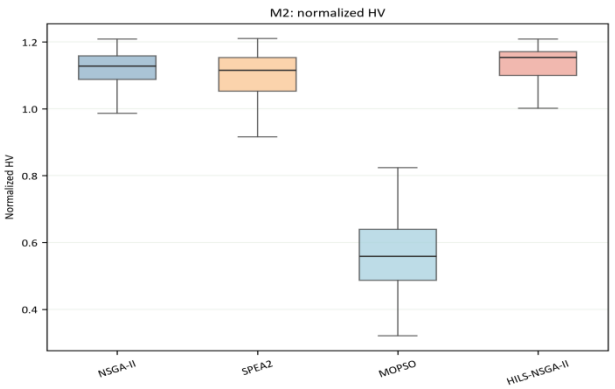
(a) EM01(S1)



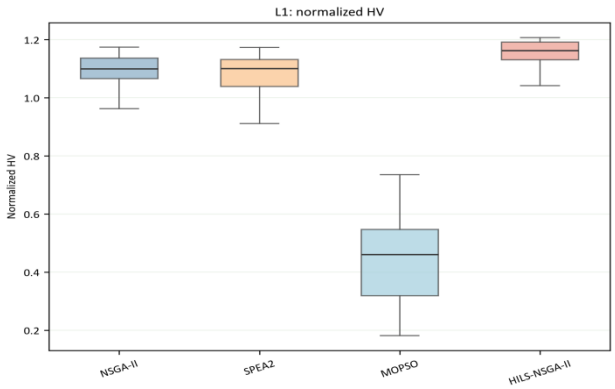
(b) EM02(S2)



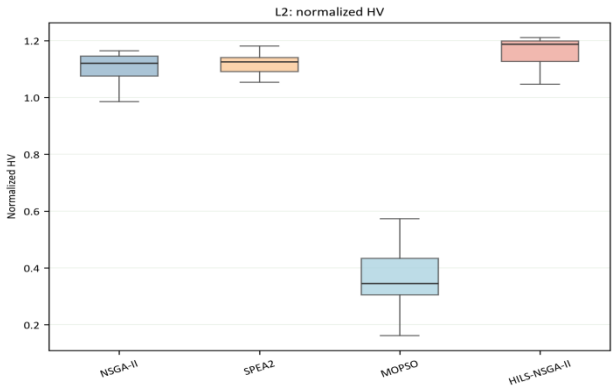
(c) EM03(M1)



(d) EM04(M2)



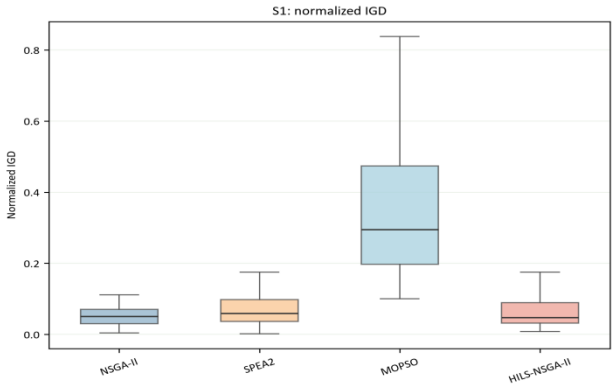
(e) EM05(L1)



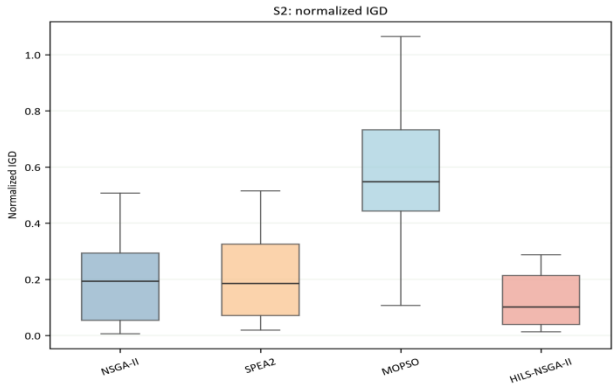
(f) EM06(L2)

Fig. 8. Distributions of normalized HV for the four algorithms on six representative instances

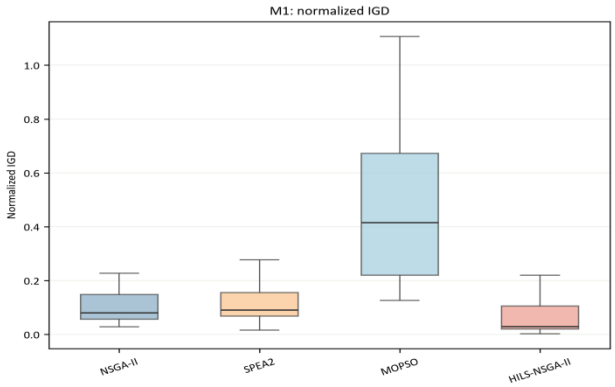
Figure 9 presents normalized IGD for the six instances. HILS-NSGA-II usually has the lowest median and a compact distribution. NSGA-II and SPEA2 are competitive on the small cases, but their IGD values are higher on most medium and large cases. MOPSO records the highest IGD on most instances.



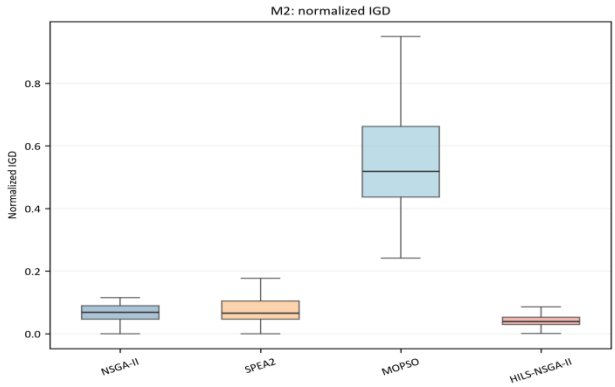
(a) EM01(S1)



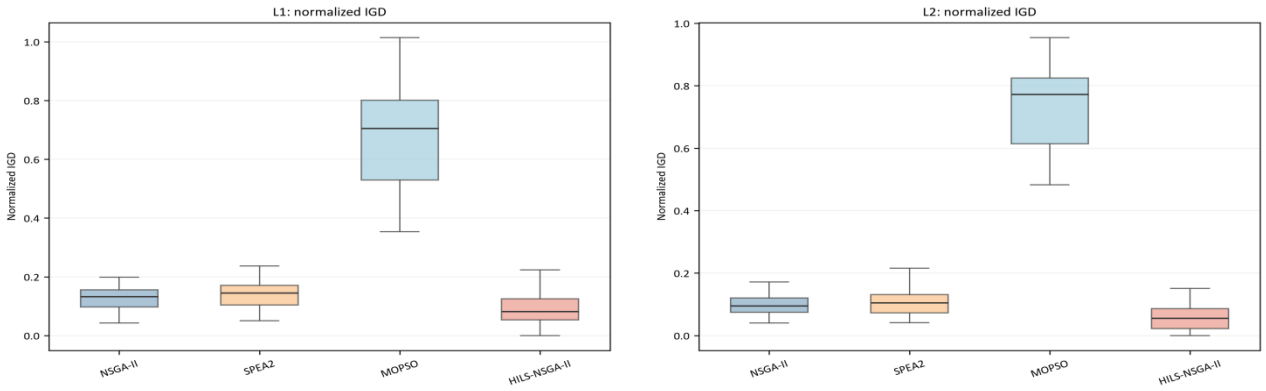
(b) EM02(S2)



(c) EM03(M1)



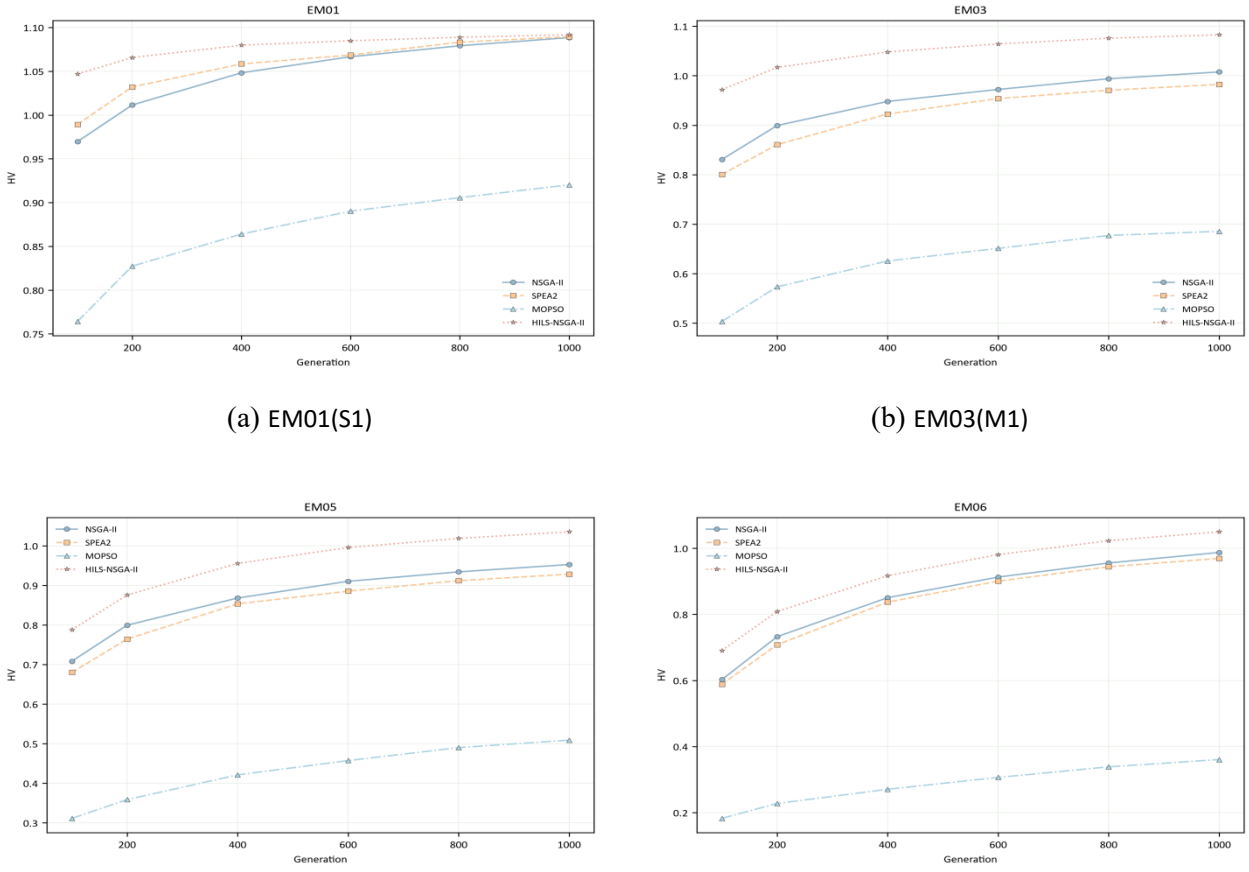
(d) EM04(M2)



(e) EM05(L1) (f) EM06(L2)

Fig. 9. Distributions of normalized IGD for the four algorithms on six representative instances

Figure 10 traces normalized HV over 1,000 generations on four representative instances. HV rises for all algorithms. HILS-NSGA-II ends with the highest value in each case, with a clearer separation on EM03, EM05, and EM06. NSGA-II and SPEA2 follow similar trajectories, while MOPSO converges to lower values.



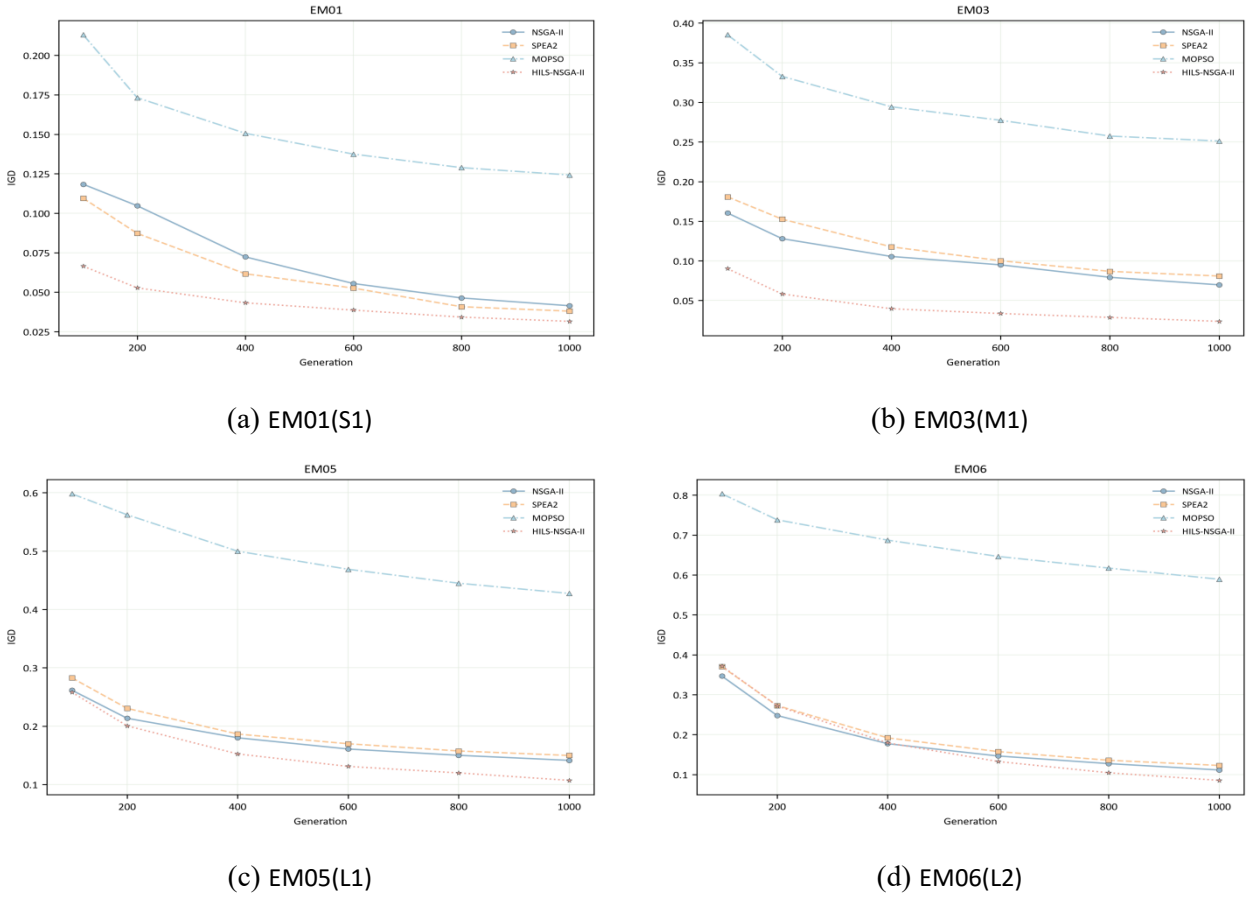
(a) EM01(S1) (b) EM03(M1)

(c) EM05(L1) (d) EM06(L2)

Fig. 10. Convergence curves of normalized HV over generations on four representative instances

Figure 11 gives the corresponding IGD curves. IGD falls as the search progresses, and HILS-NSGA-II reaches the lowest final value on all four instances. The separation is clearer on the medium and large cases. NSGA-II and SPEA2 are close on EM01, whereas MOPSO remains higher throughout the

search.



(a) EM01(S1) (b) EM03(M1)

(c) EM05(L1) (d) EM06(L2)

Fig. 11. Convergence curves of normalized IGD over generations on four representative instances

Statistical tests support these observations. The Friedman test detected significant differences among the four algorithms. The p-values were 1.50×10^{-15} for HV and 4.27×10^{-15} for IGD. The Wilcoxon signed-rank tests with Holm correction showed that HILS-NSGA-II significantly outperformed NSGA-II, SPEA2, and MOPSO in both final HV and final IGD. HILS-NSGA-II also obtained the best average ranks of 1.23 for HV and 1.33 for IGD. The tests therefore support the advantage of HILS-NSGA-II on the evaluated instances.

5.6 Ablation Study

An ablation study was conducted to examine the effects of hybrid initialization and Pareto local search. Four algorithm variants were considered. Base-NSGA-II used random initialization and did not apply local search. HI-NSGA-II replaced random initialization with the proposed hybrid initialization. LS-NSGA-II retained random initialization and added Pareto local search. HILS-NSGA-II included both components.

The experiments used 12 test instances. Two instances were selected from each scale group. The selected instances were EM01-EM06 and EM25-EM30. Each variant was independently run ten times on every instance. The same population size, generation limit, crossover rate, and mutation rate were used for all variants.

Table 9 reports the mean results of the ablation variants. HI-NSGA-II raises HV from 0.9228 to 1.0220 and lowers IGD from 0.1737 to 0.1008, changes of 10.75% and 41.95%, respectively. LS-NSGA-

II also improves both indicators when used alone, but by only 0.95% in HV and 4.56% in IGD.

Table 9
Definitions and mean results of the ablation variants

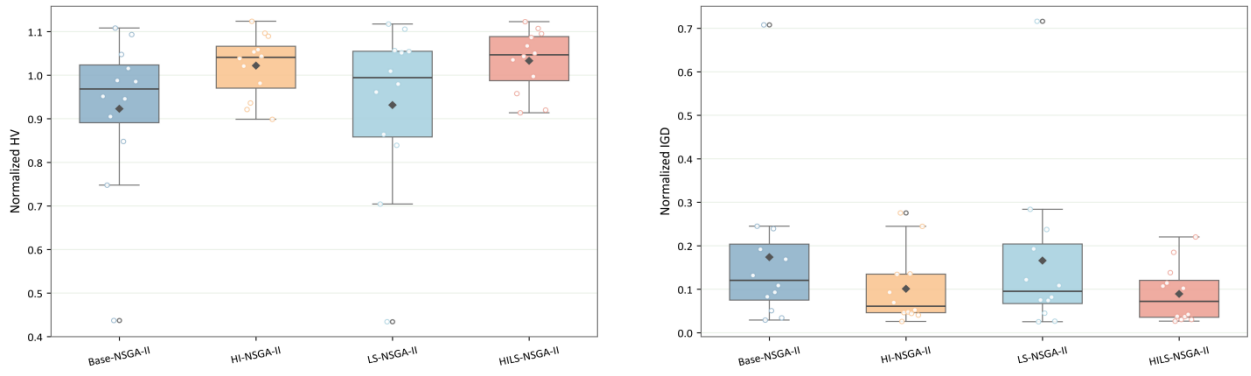
Variant	Hybrid initialization	Pareto local search	Mean HV	Mean IGD	Mean NPS
Base-NSGA-II	No	No	0.9228	0.1737	94.39
HI-NSGA-II	Yes	No	1.0220	0.1008	110.95
LS-NSGA-II	No	Yes	0.9316	0.1657	95.50
HILS-NSGA-II	Yes	Yes	1.0332	0.0893	108.91

HILS-NSGA-II has the best mean HV and IGD. Against Base-NSGA-II, HV rises by 11.96% and IGD falls by 48.59%. Against HI-NSGA-II, the corresponding changes are 1.09% and 11.43%. Hybrid initialization accounts for most of the gain, while local search adds a further improvement when the two components are combined.

Figure 12 compares the distributions of normalized HV and IGD. Each boxplot contains 12 instance-level means obtained after averaging ten runs per variant and instance. HILS-NSGA-II usually ranks first, followed by HI-NSGA-II. Base-NSGA-II and LS-NSGA-II are less competitive on the medium and large cases.

The Friedman test detected significant differences among the four variants. The p-values were 0.0013 for HV and 0.0042 for IGD. The Wilcoxon signed-rank test with Holm correction showed that HILS-NSGA-II significantly outperformed Base-NSGA-II in both HV and IGD. The adjusted p-values were 0.0244 and 0.0342, respectively. HILS-NSGA-II also obtained the best average ranks of 1.58 for HV and 1.67 for IGD.

The instance-level counts lead to the same interpretation. HILS-NSGA-II beats Base-NSGA-II on 10 of 12 instances for HV, with one tie and one loss, and on 11 instances for IGD. Against HI-NSGA-II, it records eight wins for each indicator. Those differences do not remain significant after Holm correction. Hybrid initialization is the stronger component; Pareto local search supplies a smaller complementary gain.



(a) normalized HV (b) normalized IGD

Fig. 12. Performance distributions of the four ablation variants

The ablation evidence supports both added components but assigns them different roles. Preference-guided initialization provides the main improvement, and local search refines the final nondominated set.

5.7 Sensitivity Analysis of Production Parameters

Energy-efficient scheduling requires a balance between delivery performance and carbon emissions. Previous studies have shown that due-date settings can substantially affect tardiness-oriented scheduling results [24]. Energy-aware job-shop studies have also reported a clear conflict between delivery-related objectives and energy consumption [6, 25]. In addition, tooling availability can affect flow time, tardiness, and the number of tardy jobs [26]. Recent research has further identified limited fixture-pallet resources as an important production constraint [15]. Based on these observations, three production factors were examined: due-date tightness, fixture availability, and final-inspection efficiency.

A one-factor-at-a-time design was adopted. Six representative instances, EM01-EM06, were selected from the six scale groups. Each scenario was independently evaluated using five random seeds, namely 201-205. The experiment contained one baseline scenario and six modified scenarios. This produced 210 scenario-instance-seed observations.

The due dates were modified according to

$$d_j' = r_j + k_d\left(d_j - r_j\right) \tag{81}$$

where r_j and d_j denote the release time and original due date of job j, respectively. Values of $k_d = 0.8$, 1.0, and 1.2 represent tight, baseline, and loose due dates.

Fixture availability was divided into scarce, baseline, and abundant levels. The scarce level retained approximately 75% of the original fixture capacity. A deterministic coverage rule ensured that every job and datum-retention block remained feasible. The abundant level increased the physical fixture capacity to approximately 125% of the baseline level. Additional fixtures were cloned from existing fixture types. Therefore, this scenario increased resource capacity without changing the original compatibility structure.

Final-inspection efficiency was controlled by multiplying every inspection time by k_I. The values k_I =0.8, 1.0, and 1.2 represent fast, baseline, and slow inspection, respectively. Table 10 summarizes the experimental levels.

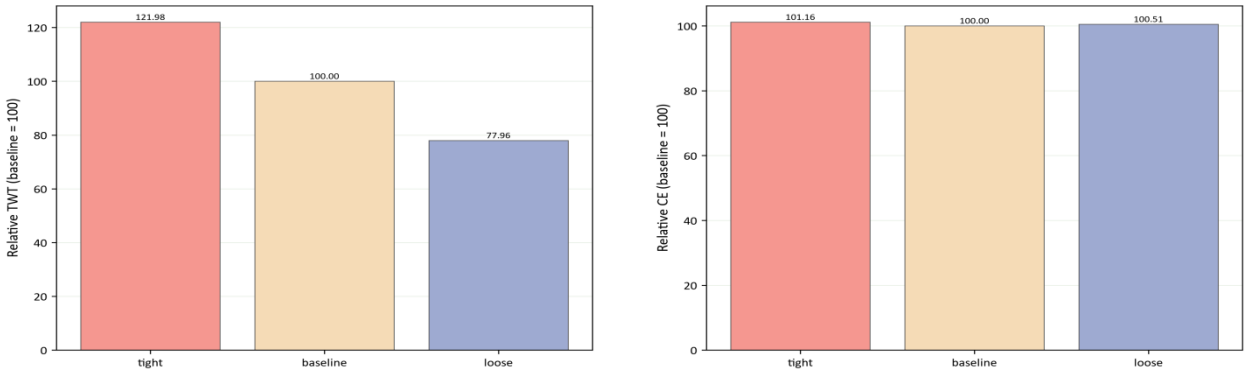
Table 10
Production factors and levels used in the sensitivity analysis

Factor	Modified level 1	Baseline	Modified level 2
Due-date tightness k_d	0.8, tight	1.0	1.2, loose
Fixture availability	Scarce, approximately 75%	Original capacity	Abundant, approximately 125%
Inspection-time multiplier k_I	0.8, fast	1.0	1.2, slow

HV and IGD were not compared across different scenarios because each parameter change defines a different scheduling problem and reference front. Instead, three representative solutions were selected from each Pareto front: the minimum-TWT solution, the minimum-CE solution, and the compromise solution. The compromise solution was identified by the minimum normalized distance to the ideal point. TWT, CE, the number of tardy jobs, fixture waiting time, inspection waiting time, and the four energy components were compared with the baseline. For graphical presentation, TWT and CE were converted into relative indices. The baseline value was set to 100. An index above 100 indicates an increase from the baseline, while an index below 100 indicates a decrease.

Figure 13 reports the due-date experiment. The baseline TWT and CE indices are 100. Tight due

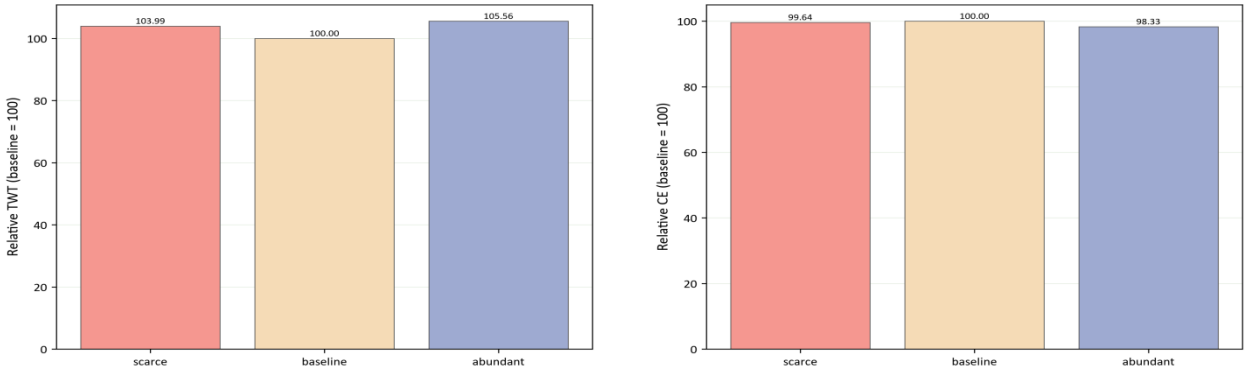
dates raise the compromise-solution indices to 121.98 and 101.16, whereas loose due dates change them to 77.96 and 100.51. The delivery-oriented endpoint is more sensitive: its TWT rises by 63.36% under tight dates and falls by 36.53% under loose dates. Due-date tightness mainly changes delivery performance; its effect on emissions is small.



(a) TWT (b) CE

Fig. 13. Relative TWT and CE under different due-date tightness levels

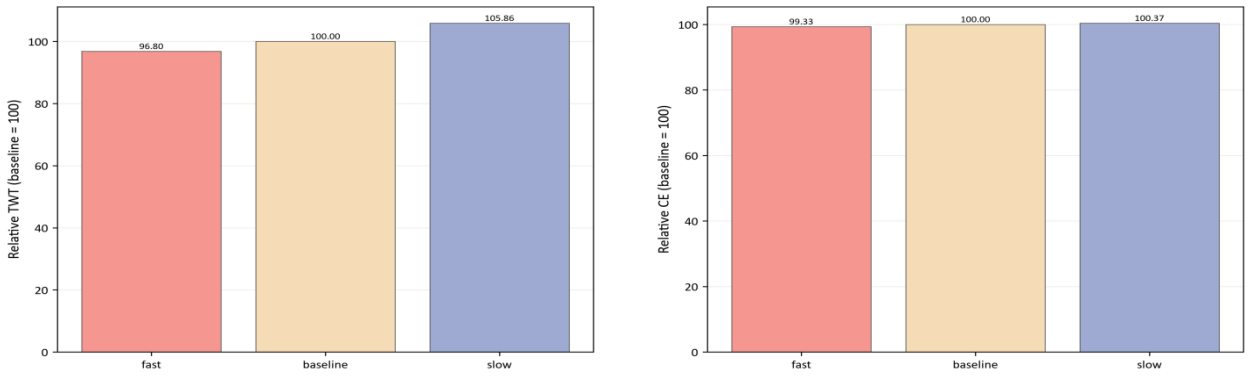
Figure 14 reports the fixture experiment. For the compromise solution, scarce and abundant fixtures produce TWT indices of 103.99 and 105.56 and CE indices of 99.64 and 98.33. Scarcity adds 13.97 time units of mean fixture waiting, while extra fixtures remove 26.63 time units. More fixture capacity reduces fixture congestion and slightly lowers CE, but it does not lower TWT for the selected compromise solutions. Machine queues, datum-retention requirements, and inspection can remain binding.



(a) TWT (b) CE

Fig. 14. Relative TWT and CE under different fixture-availability levels

Figure 15 reports the inspection experiment. Fast inspection changes the TWT and CE indices to 96.80 and 99.33. Slow inspection raises them to 105.86 and 100.37 and adds 11.19 time units of mean inspection waiting. The delivery-oriented solution is more exposed, with a 13.10% TWT increase under slow inspection. Final inspection is therefore a relevant downstream resource, although its effect on emissions remains limited.



(a) TWT (b) CE

Fig. 15. Relative TWT and CE under different final-inspection efficiency levels

Table 11 summarizes the changes in TWT and CE for the three representative solution types. The percentages are calculated relative to the corresponding baseline solution.

Table 11
Mean changes in TWT and CE under different production scenarios

Scenario	Representative solution	TWT change	CE change
Tight due dates	Minimum TWT	+63.36%	−0.75%
	Compromise	+21.98%	+1.16%
	Minimum CE	+6.06%	−0.15%
Loose due dates	Minimum TWT	−36.53%	−3.21%
	Compromise	−22.04%	+0.51%
	Minimum CE	−10.37%	+0.06%
Scarce fixtures	Minimum TWT	+7.68%	−1.72%
	Compromise	+3.99%	−0.36%
	Minimum CE	+0.56%	−0.44%
Abundant fixtures	Minimum TWT	+11.81%	−3.46%
	Compromise	+5.56%	−1.67%
	Minimum CE	+5.39%	−0.84%
Fast inspection	Minimum TWT	−2.12%	−2.81%
	Compromise	−3.20%	−0.67%
	Minimum CE	+1.13%	−1.03%
Slow inspection	Minimum TWT	+13.10%	−1.14%
	Compromise	+5.86%	+0.37%
	Minimum CE	+7.85%	−0.06%

Among the three factors, due-date tightness has the largest effect on TWT. Inspection efficiency has a smaller but consistent delivery effect. Fixture capacity mainly changes fixture waiting and congestion; its TWT response is not monotonic because other resources can remain restrictive. CE changes little in most scenarios, suggesting that machine assignment, processing mode, and the energy mix play a larger role in that objective. All 25,557 retained solutions passed the feasibility and objective-recalculation audits without a detected violation.

6. Conclusions

This paper formulates a delivery-low-carbon flexible job-shop scheduling problem for precision

machining of new-energy vehicle motor housings. The model minimizes total weighted tardiness and carbon emissions while linking flexible machines, processing modes, sequence-dependent setup, transport, fixture capacity, datum-retention blocks, and mandatory final inspection. It extends the conventional FJSP toward the multi-resource setting highlighted by Dauzère-Pérès *et al.,* [3] and Kasapidis *et al.,* [27], while retaining the delivery-energy perspective of Liu *et al.,* [5], Gahm *et al.,* [4], and Zhang and Chiong [6].

The OS-MMS-FPA representation links operation order, machine-mode choice, and job-level fixture allocation. Its serial decoder enforces precedence, machine eligibility, fixture compatibility, continuous fixture occupation, and final inspection before evaluating the objectives. Exact small-case solutions and independent recalculation support the model's feasibility and energy accounting, including both productive and non-productive energy terms [4, 5].

HILS-NSGA-II retains the nondominated sorting and crowding-distance mechanisms of NSGA-II [11] and adds hybrid initialization and Pareto local search. Across the 30 test instances, its mean HV exceeds NSGA-II and SPEA2 by 3.70% and 4.28%, while mean IGD is lower by 31.36% and 35.74%. The statistical tests support these differences. The pattern is consistent with earlier evidence that population search can benefit from local improvement in multi-objective scheduling [6, 28].

The ablation study locates most of the gain in hybrid initialization. Pareto local search contributes less on its own but improves the full method. Relative to Base-NSGA-II, HILS-NSGA-II raises mean HV by 11.96% and lowers mean IGD by 48.59%. Because the local-search gain varies by instance, the method is best described as preference-guided initialization supported by Pareto local refinement, rather than as a local-search algorithm alone [28].

The Pareto fronts also expose the delivery-carbon conflict. Faster assignments tend to reduce completion times and tardiness at higher energy use. Low-carbon schedules accept longer completion times when this reduces emissions. No schedule is best for both objectives in every case, so the nondominated set offers delivery-oriented, low-carbon, and compromise choices. Comparable trade-offs have been reported by Liu *et al.,* [5] and Zhang and Chiong [6].

Production conditions affect the two objectives differently. Due-date tightness changes weighted tardiness most strongly but has little effect on carbon emissions, in line with Zhou *et al.,* [24]. Fixture shortages increase waiting, yet extra fixtures do not always reduce TWT because machines or inspection may remain bottlenecks. Fixture capacity should therefore be coordinated with other resources [10, 15]. Faster final inspection also shortens completion times. Capacity improvement is most useful when it targets the resource that is actually binding.

The model assumes known processing data and resource availability. It does not cover dynamic arrivals, breakdowns, rework, inspection failures, or uncertain processing times, and it uses a constant carbon factor. Dynamic and uncertain FJSPs remain natural extensions [3]. Time-varying electricity prices and carbon intensity could also be included [29]. Future search neighborhoods may use tardy jobs, fixture conflicts, inspection queues, or machine bottlenecks. Industrial validation with long-term motor-housing data is also needed.

Funding
This research received no external funding.

Data Availability Statement
The datasets generated and analyzed during the current study are available from the corresponding author upon reasonable request.

Conflicts of Interest

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 paper.

Acknowledgment

The author gratefully acknowledges the support from the Youth Project of Anhui Provincial Philosophy and Social Science Planning Program under Grant AHSKQ2022D070.

References

[1] International Energy Agency. (2026). *Global EV Outlook 2026*. IEA. https://www.iea.org/reports/global-ev-outlook-2026

[2] 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

[3] Dauzère-Pérès, S., Ding, J., Shen, L., & Tamssaouet, K. (2024). The flexible job shop scheduling problem: A review. *European Journal of Operational Research*, 314(2), 409–432. https://doi.org/10.1016/j.ejor.2023.05.017

[4] Gahm, C., Denz, F., Dirr, M., & Tuma, A. (2016). Energy-efficient scheduling in manufacturing companies: A review and research framework. *European Journal of Operational Research*, 248(3), 744–757. https://doi.org/10.1016/j.ejor.2015.07.017

[5] Liu, Y., Dong, H., Lohse, N., Petrovic, S., & Gindy, N. (2014). An investigation into minimising total energy consumption and total weighted tardiness in job shops. *Journal of Cleaner Production*, 65, 87–96. https://doi.org/10.1016/j.jclepro.2013.07.060

[6] Zhang, R., & Chiong, R. (2016). Solving the energy-efficient job shop scheduling problem: A multi-objective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption. *Journal of Cleaner Production*, 112, 3361–3375. https://doi.org/10.1016/j.jclepro.2015.09.097

[7] Yin, L., Li, X., Gao, L., Lu, C., & Zhang, Z. (2017). Energy-efficient job shop scheduling problem with variable spindle speed using a novel multi-objective algorithm. *Advances in Mechanical Engineering*, 9(4). https://doi.org/10.1177/1687814017695959

[8] Shen, L., Dauzère-Pérès, S., & Neufeld, J. S. (2018). Solving the flexible job shop scheduling problem with sequence-dependent setup times. *European Journal of Operational Research*, 265(2), 503–516. https://doi.org/10.1016/j.ejor.2017.08.021

[9] Wu, X., Peng, J., Xiao, X., & Wu, S. (2021). An effective approach for the dual-resource flexible job shop scheduling problem considering loading and unloading. *Journal of Intelligent Manufacturing*, 32(3), 707–728. https://doi.org/10.1007/s10845-020-01697-5

[10] Zhou, Y., Du, S., Liu, M., & Shen, X. (2024). Machine-fixture-pallet resources constrained flexible job shop scheduling considering loading and unloading times under pallet automation system. *Journal of Manufacturing Systems*, 73, 143–158. https://doi.org/10.1016/j.jmsy.2024.01.010

[11] Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2), 182–197. https://doi.org/10.1109/4235.996017

[12] Li, X., Guo, X., Tang, H., Wu, R., Wang, L., Pang, S., Liu, Z., Xu, W., & Li, X. (2022). Survey of integrated flexible job shop scheduling problems. *Computers & Industrial Engineering*, 174, 108786. https://doi.org/10.1016/j.cie.2022.108786

[13] Meng, L., Zhang, C., Shao, X., & Ren, Y. (2019). MILP models for energy-aware flexible job shop scheduling problem. *Journal of Cleaner Production*, 210, 710–723. https://doi.org/10.1016/j.jclepro.2018.11.021

[14] Rossi, A., & Dini, G. (2007). Flexible job-shop scheduling with routing flexibility and separable setup times using ant colony optimisation method. *Robotics and Computer-Integrated Manufacturing*, 23(5), 503–516. https://doi.org/10.1016/j.rcim.2006.06.004

[15] Liu, M., Lv, J., Du, S., Deng, Y., Shen, X., & Zhou, Y. (2024). Multi-resource constrained flexible job shop scheduling problem with fixture-pallet combinatorial optimisation. *Computers & Industrial Engineering*, 188, 109903. https://doi.org/10.1016/j.cie.2024.109903

[16] Zhu, K., Gong, G., Peng, N., Zhang, L., Huang, D., Luo, Q., & Li, X. (2023). Dynamic distributed flexible job-shop scheduling problem considering operation inspection. *Expert Systems with Applications*, 224, 119840. https://doi.org/10.1016/j.eswa.2023.119840

[17] Mokhtari, H., & Hasani, A. (2017). An energy-efficient multi-objective optimization for flexible job-shop scheduling problem. *Computers & Chemical Engineering*, 104, 339–352. https://doi.org/10.1016/j.compchemeng.2017.05.004

[18] Wu, X., & Sun, Y. (2018). A green scheduling algorithm for flexible job shop with energy-saving measures. *Journal of Cleaner Production*, 172, 3249–3264. https://doi.org/10.1016/j.jclepro.2017.10.342

[19] Dai, M., Tang, D., Giret, A., & Salido, M. A. (2019). Multi-objective optimization for energy-efficient flexible job shop scheduling problem with transportation constraints. *Robotics and Computer-Integrated Manufacturing*, 59, 143–157. https://doi.org/10.1016/j.rcim.2019.04.006

[20] Wei, Z., Liao, W., & Zhang, L. (2022). Hybrid energy-efficient scheduling measures for flexible job-shop problem with variable machining speeds. *Expert Systems with Applications*, 197, 116785. https://doi.org/10.1016/j.eswa.2022.116785

[21] Du, Y., Li, J., Li, C., & Duan, P. (2024). A reinforcement learning approach for flexible job shop scheduling problem with crane transportation and setup times. *IEEE Transactions on Neural Networks and Learning Systems*, 35(4), 5695–5709. https://doi.org/10.1109/tnnls.2022.3208942

[22] Zitzler, E., Laumanns, M., & Thiele, L. (2001). *SPEA2: Improving the strength Pareto evolutionary algorithm* (TIK Report No. 103). ETH Zurich, Computer Engineering and Networks Laboratory. https://doi.org/10.3929/ethz-a-004284029

[23] Coello, C. A. C., Pulido, G. T., & Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3), 256–279. https://doi.org/10.1109/tevc.2004.826067

[24] Zhou, H., Cheung, W., & Leung, L. C. (2009). Minimizing weighted tardiness of job-shop scheduling using a hybrid genetic algorithm. *European Journal of Operational Research*, 194(3), 637–649. https://doi.org/10.1016/j.ejor.2007.10.063

[25] Luan, F., Zhao, H., Liu, S. Q., He, Y., & Tang, B. (2023). Enhanced NSGA-II for multi-objective energy-saving flexible job shop scheduling. *Sustainable Computing: Informatics and Systems*, 39, 100901. https://doi.org/10.1016/j.suscom.2023.100901

[26] Melnyk, S. A., Ghosh, S., & Ragatz, G. L. (1989). Tooling constraints and shop floor scheduling: A simulation study. *Journal of Operations Management*, 8(2), 69–89. https://doi.org/10.1016/0272-6963(89)90013-2

[27] Kasapidis, G. A., Paraskevopoulos, D. C., Mourtos, I., & Repoussis, P. P. (2025). A unified solution framework for flexible job shop scheduling problems with multiple resource constraints. *European Journal of Operational Research*, 320(3), 479–495. https://doi.org/10.1016/j.ejor.2024.08.010

[28] Jaszkiewicz, A. (2002). Genetic local search for multi-objective combinatorial optimization. *European Journal of Operational Research*, 137(1), 50–71. https://doi.org/10.1016/s0377-2217(01)00104-7

[29] Gong, X., De Pessemier, T., Martens, L., & Joseph, W. (2019). Energy- and labor-aware flexible job shop scheduling under dynamic electricity pricing: A many-objective optimization investigation. *Journal of Cleaner Production*, 209, 1078–1094. https://doi.org/10.1016/j.jclepro.2018.10.289