Received: 06 July 2025; Revised: 23 Sep. 2025; Accepted: 19 Nov. 2025
https://doi.org/10.22306/al.v13i1.734

Optimizing logistics and production flows for sustainability in green flexible job-shops

Asma Fekih
University of Tunis El Manar, LR-OASIS, National Engineering School of Tunis, 1002 Tunis, Tunisia,
asmafekih@hotmail.fr (corresponding author)
Mohamed Karim Hajji
American University of the Middle East, College of Engineering and Technology, Egaila 54200, Kuwait,
Mohammad.Hajji@aum.edu.kw

Keywords: industrial logistics, green flexible job-shop, constraint programming, particle swarm algorithm.
Abstract: In industrial operations, the effective optimization of logistics and production flows, including material, energy, and information flows, is essential for improving operational performance while reducing environmental impacts. The combinatorial complexity of the green flexible job-shop scheduling problem (GFJSP) and its connection to sustainable manufacturing have attracted significant attention in recent research. This study aims to minimize total flow time and carbon emissions by optimizing the management of industrial flows, while accounting for technical elements of logistics such as production sequencing, resource allocation, and coordination of material and energy flows. Two complementary solution approaches are proposed: an exact constraint programming (CP) model for small-sized instances, ensuring optimality, and a particle swarm optimization (PSO) algorithm enhanced with an energy-aware encoding scheme and logistic map-based population initialization for medium and large-scale problems. Numerical experiments demonstrate the effectiveness of both approaches in addressing operational efficiency and environmental objectives. The novelty of this work lies in integrating the management of logistics and production flows with sustainable scheduling through a hybrid exact–metaheuristic strategy. Sensitivity analysis further validates the robustness of the PSO, highlighting how exact and metaheuristic methods complement each other to tackle complex industrial scheduling challenges. Overall, this study provides both scientific insights and practical guidance for managing logistics and production flows in sustainable industrial systems.

1 Introduction

Efficient management of material, energy, and information flows is vital to manufacturing performance, impacting productivity, lead times, and environmental sustainability. The flexible job-shop scheduling problem (FJSP) represents one of the most complex combinatorial optimization challenges due to its dual decision layers: machine assignment (routing), where operations can be processed on multiple eligible machines, and operation sequencing [1]. This dual-layered structure makes the FJSP a strongly NP-hard problem, even more challenging than the classical job-shop scheduling problem [2]. Traditionally, the FJSP has been addressed with the primary objective of minimizing the makespan [3], but recent industrial and environmental concerns require considering multiple flows and their management throughout the production system. In particular, minimizing total carbon emissions has become increasingly critical, in line with the principles of green manufacturing, low-carbon production planning, and sustainable logistics [4]. The industrial sector remains one of the major contributors to greenhouse gas emissions. For instance, the manufacturing industry accounts for about 30–35% of global energy use and nearly 40% of total CO_2 emissions worldwide [5]. In recent years, most research on FJSP has emphasized traditional performance indicators, such as makespan, tardiness, and machine utilization, while paying limited attention to environmental and flow-oriented concerns. Although the FJSP reflects the complexity and flexibility of real industrial environments, sustainability-oriented criteria, particularly carbon emission reduction and total flow time minimization, have rarely been addressed together in this context. The existing studies have dealt extensively with energy-aware scheduling mostly in hybrid flow shops [6] or batch systems [7], leaving a gap in the integration of environmental objectives in green flexible job-shop problems (GFJSP) and the management of material and energy flows. However, the need for sustainable logistics and integrated flow management is becoming increasingly critical due to regulatory pressure and market expectations.

From a practical viewpoint, total flow time is a key determinant of production efficiency and responsiveness, while carbon emissions are a major environmental cost. To the best of the authors' knowledge, this study is among the first to simultaneously address the minimization of total flow time and carbon emissions in a GFJSP setting, using both exact and metaheuristic approaches, contributing both scientifically, by introducing a novel bi-objective formulation incorporating machine-dependent emission rates and energy consumption, and practically, by offering insights into the management of multiple flows in GFJSP. In this paper, a Constraint Programming (CP) was developed for solving small instances, and a Particle Swarm Optimization (PSO) metaheuristic was used to solve medium and large-scale instances very efficiently. In this respect, the study presents a novel issue in sustainable scheduling by addressing both productivity and environmental performance in flexible production systems.

The remainder of the paper is structured as follows. Section 2 provides a review of the most relevant literature. Section 3 describes the problem under consideration. Section 4 presents an exact CP model and a metaheuristic PSO designed to handle large-scale instances. Section 5 presents the experimental results evaluations, ANOVA analysis, and discussion of results. Finally, Section 6 concludes the paper by summarizing the key contributions and outlining possible directions for future research.

2 Literature review

This review presents the contributions made since 2015, to reflect the recent developments in the FJSP mainly in terms of carbon emissions and total flow time minimization. This period shows an increasing interest of the research community towards integrating sustainability concerns, particularly energy consumption and environmental impact, into some of the classical scheduling objectives, such as makespan. This review examines the existing works based on their optimization criteria as well as solution methods. The next sub-sections discuss in detail methodologies aimed at carbon emission reduction and total flow time reduction in FJSP, referring to the state-of-the-art and how this helps identify research gaps.

2.1 Carbon emission minimization in FJSP

New methods have been introduced in the recent literature on the integration of carbon emissions with traditional performance criteria. For instance, the model, proposed by [8], combines deep reinforcement learning with programming techniques to minimize machine emissions and the carbon footprint of idle time. In [9], authors introduced an improved metaheuristics NSGA-II that optimizes makespan, operating cost and carbon emissions simultaneously. Authors in [10] developped a two-objective genetic algorithm that minimizes job tardiness and carbon emission. More recently, Ref. [11] proposed a GFJSP model for CO2 emissions directly into a mixed-integer linear programming (MILP) model, along with automated guided vehicle (AGV) transport and consumable usage. These contributions show how important it is to incorporate environmental criteria, specifically carbon emissions, into scheduling models, beyond establishing thresholds in reliability and productivity. Table 1 summarizes key studies related to low-carbon in FJSP, along with their methods, objectives, and limitations.

Table 1 Summary of previous studies of low-carbon approaches in FJSP

Reference	Method	Objective fonction(s)	Gaps
[8]	Deep Reinforcement Learning	Emissions from both machines and idle states	Lacks granularity in the emission rate per machine
[9]	NSGA-II	Makespan, energy cost, and carbon emissions	Uses uniform emission factors across machines
[10]	Genetic Algorithm	Total carbon footprint and tardiness	Lacks machine-dependent emission modeling
[11]	MILP + Logistics modeling	Emissions from AGVs, coolant, lubricants; includes CO_2	Emissions are aggregated; no per-machine emission modeling
[12]	Whale Optimization Algorithm	Carbon emissions	Simplified emission modeling; no machine granularity
[13]	Memetic Algorithm	Makespan, carbon emissions, and worker costs	Uniform emission rates; no machine-specific differentiation
[14]	Deep Reinforcement Learning	Carbon emissions and idle-time emissions	No machine-specific emission coefficients
[15]	MILP + Heuristic	Makespan and emissions in an industrial case	Uses general/global emission coefficients
[16]	Multi-objective MILP under ToU	Minimizes cost, carbon emissions, and maximizes customer satisfaction under ToU pricing	Focus on electricity tariffs; limited integration of machine-specific emissions

2.2 Total flow time minimization in FJSP

In contrast to carbon emission objectives, minimizing flow time remains a classical but less extensively integrated goal in flexible job-shop scheduling, especially when combined with environmental criteria. Flow time, representing the total time jobs spend in the system from release to completion, is crucial for improving production efficiency and responsiveness. In previous studies, research has focused mostly on traditional criteria such as makespan, tardiness, or workload balance [17-28], while few papers have considered total flow time minimization in conjunction with sustainability goals. Most of the scheduling research on environmental impact has concentrated on hybrid flow shops or batch processing systems [6,7], leaving a notable gap in the FJSP context. Given the increasing industrial demand for low-carbon manufacturing, addressing total flow time minimization alongside carbon emission reduction in FJSP is both timely and relevant (Table 2).

Table 2 Review of flow-time oriented approaches in FJSP

Reference	Method	Objective function(s)
[17]	Tabu search + Genetic Algorithm	Makespan
[18]	Multi-agent chemical reaction optimization + Tabu search	Makespan
[19]	MILP, Tabu search	Makespan
[20]	Greedy heuristics	Makespan
[21]	Jaya algorithm	Makespan
[22]	Artificial Bee Colony algorithm	Total flow time
[23]	Hybrid genetic algorithm + tabu search	Total flow time
[24]	Particle Swarm Optimization	Makespan, total workload
[25]	NSGA-II	Makespan, maximum tardiness
[26]	Particle Swarm Optimization	Makespan
[27]	Hybrid Genetic Algorithm + Variable Neighborhood Search	Makespan, total workload
[28]	Constraint Programming	Makespan, maximum flow time, maximum tardiness

In response to the identified research gaps, this study introduces both an exact and a metaheuristic to solve the GFJSP, targeting the simultaneous minimization of total flow time and carbon emissions, a practically relevant problem and yet an underexplored combination of objectives to our knowledge. We highlight total flow time as a way to improve resource continuity and reduce work-in-progress inventory, which differs from most previous studies that mainly concentrate on makespan or tardiness. Our goal is to improve scalability and solution diversity for large-scale instances by creating a CP model and a PSO algorithm that integrates a novel energy-aware encoding scheme.

3 Problem description and model formulation

3.1 Problem description and assumptions

The GFJSP addressed in this paper involves scheduling a set of independent jobs on a set of heterogeneous machines while optimizing both total flow time and carbon emissions. Formally, let $J = \{J_1, J_2, J_3, \ldots, J_n\}$ be a set of n jobs to be processed on a set of m machines $M = \{M_1, M_2, M_3, \ldots, M_k\}$. Each job J_j consists of a specific sequence of n_j operations $\{O_{1j}, O_{2j}, O_{3j}, \ldots, On_{jj}\}$which must respect precedence constraints. (The operation cannot start before the completion of its predecessor). Each operation O_{ij} (with $i = 1, 2, \ldots, n_j$) can be processed by one machine from a **subset** $M_{ij} \subseteq M$ of eligible machines. The processing time P_{ij}^k, energy consumption E_{ij}^k, and carbon emission rate C_k depend on the machine $M_k \in M_{ij}$ chosen. Once started, an operation must run to completion without interruption (**non-preemptive scheduling**), and each machine can process at most one operation at a time (**disjunctive constraint**). All jobs and machines are assumed to be available at time zero. There are no precedence constraints between operations of different jobs, but the internal order of operations within each job must be preserved (conjunctive constraint). The GFJSP is thus a two-level decision problem: assign each operation to a machine from its eligible set (routing decision), and sequence the operations on each machine (scheduling decision). To illustrate the problem setting, Table 3 presents a sample instance with 4 jobs, 3 machines, and 10 operations, including the processing times and energy consumption of each operation on each machine.

Table 3 Data of the 4×3 GFJS instance

Job	*O_{ij}*	*Processing times (min) / Machines*			*Energy Consumption (kWh)*		
		M1	*M2*	*M3*	*M1*	*M2*	*M3*
1	O_{11}	3	5	2	0.20	0.25	0.15
	O_{21}	4	3	1	0.30	0.20	0.10
	O_{31}	1	5	6	0.20	0.28	0.35
2	O_{12}	8	3	3	0.18	0.14	0.13
	$O_{2,2}$	2	3	7	0.31	0.24	0.42
	O_{32}	4	5	5	0.17	0.21	0.20
3	O_{13}	9	1	3	0.13	0.10	0.12
	O_{23}	2	4	5	0.26	0.30	0.33
4	O_{14}	5	9	5	0.19	0.27	0.21
	O_{24}	1	3	5	0.21	0.22	0.24

Depending on the structure of machine eligibility [29]: If each operation O_{ij} can be executed on all machines ($M_{ij} = M$), the problem is referred to as a Total GFJSP. But if $M_{ij} \subset M$ for at least one operation, it becomes a Partial GFJSP, which is inherently more difficult to solve due to reduced flexibility. The illustrated example corresponds to a Total GFJSP, where each operation can be performed on any of the three machines.

3.2 Model formulation

To address the studied GFJSP we formulate the following bi-objective model. This formulation integrates machine-level energy consumption and emission factors in a detailed manner, allowing for a more realistic representation of scheduling decisions in carbon-sensitive environments. It distinguishes itself by explicitly considering job-level total flow time, which better captures the responsiveness of the production system, and linking it with machine-specific emission contributions through the assignment of operations. By explicitly incorporating machine-dependent emission rates, the energy-aware formulation builds upon previous work on low carbon scheduling [9,10,14]. The notations of sets, parameters, and decision variables are presented in Table 4.

Table 4 List and description of the notations used

Notation	*Description*
Sets	
$J = \{J_1, J_2, ..., J_n\}$	Set of jobs (j: index of jobs)
$M = \{M_1, M_2, ..., M_k\}$	Set of machines (indexed by k)
O_{ij}	Operation i of job j *(with* $i = 1, 2, \ldots, n_j$)
Parameters	
P_{ij}^k	Processing time of operation O_{ij} on machine M_k
E_{ij}^k	Energy consumption of O_{ij} on machine M_k
F_k	Carbon emission factor of machine M_k (kg CO_2/kWh)
L	A sufficiently large positive number (used in sequencing constraints)
Decision Variables	
$x_{ij}^k \in \{0,1\}$	1 if operation O_{ij} is assigned to machine M_k; 0 otherwise
$S_{n_j j} \geq 0$	Start time of the last operation of job j
$C_{n_j,j} \geq 0$	Completion time of the last operation n_j of job j
$y_{ij,hz}^k \in \{0,1\}$	1 if O_{ij} is scheduled after O_{hz} on machine M_k; 0 otherwise

The studied problem is formulated as follows:

$$\min f_1 = \sum_{j=1}^{n} C_{n_j,j} \tag{1}$$

$$\min f_2 = \sum_{j=1}^{n} \sum_{i=1}^{n_j} \sum_{k=1}^{m} x_{ij}^k \,.\, E_{ij}^k \,.\, F_k \tag{2}$$

Subject to :

$$\sum_{k=1}^{m} x_{ij}^k = 1, \qquad \forall j \in J, \forall i \tag{3}$$

$$S_{i+1,j} \geq S_{ij} + \sum_{k=1}^{m} x_{ij}^k \,.\, P_{ij}^k\,, \quad \forall j \in J, \forall i < \textit{last operation of } j \tag{4}$$

$$S_{ij} \ \geq \ S_{hz} + P_{kz}^k \ - L \cdot \left(3 - y_{ij,hz}^k - x_{ij}^k - x_{hz}^k\right), \quad \forall i,j,h,z, i \neq h \textit{ or } j \neq z, \forall k \in M \tag{5}$$

$$S_{hz} \ \geq \ S_{ij} + P_{ij}^k \ - L \cdot \left(y_{ij,hz}^k + 2 - x_{ij}^k - x_{hz}^k\right), \quad \forall i,j,h,z, i \neq h \textit{ or } j \neq z, \forall k \in M \tag{6}$$

$$C_{n_j,j} \geq S_{n_j j} + \sum_{k=1}^{m} x_{n_j j}^k \,.\, P_{n_j j}^k \qquad \forall j \in J \tag{7}$$

$$x_{ij}^k \in \{0, 1\}, \ \ y_{ij,hz}^k \in \{0, 1\}, \ \ S_{ij} \geq 0, \ \ C_{n_j,j} \geq 0 \tag{8}$$

Objective (1) seeks to minimize the total flow time (f_1), which is defined as the sum of completion times of all jobs and reflects overall system responsiveness and throughput. Objective (2) minimizes total carbon emissions by summing, for each scheduled operation, the product of its energy consumption and the emission factor of the assigned machine. Constraint set (3) ensures that each operation is assigned to one machine to guarantee the feasibility of machine allocation. Constraint (4) specifies the sequential technological precedence constraints within each job: No operation should start before its predecessor is completed. Constraint sets (5) and (6) enforce the sequencing of operations assigned to the same machine to avoid overlapping (one operation starting when the previous one finishes). Constraint set (7) states that the end time of a job is the time that the last operation in that job completes. Finally, constraint set (8) states that both assignment and sequencing variables should be binary, while start and finish times are non-negative.

4 Methods

4.1 *Constraint programming model*

We formulate the GFJSP using a CP model that simultaneously minimizes the total flow time and controls the total carbon emissions associated with machine usage. This model is especially suitable for combinatorial scheduling problems due to its expressiveness in modeling and its ability to represent sequencing and resource allocation constraints without linearization. According to [30], the CP is composed of three core elements: a set of decision variables, the domains of possible values associated with each variable, and a set of constraints. In scheduling, two key variable types are interval variables and sequences. An interval is defined by a process time, a start time, and an end time. A sequence is a collection of non-overlapping intervals assigned to the same resource. The following notations are defined in this model:

- $O[j,o,k]$: interval of operation o of job j on machine k with size duration $dur[j][o][k]$.
- S_k: sequence of operation scheduled on machine k;
- $endOf(O[j,o_{last},k])$: Completion time of job j, defined as the end of its last scheduled operation.

The CP model formulation is expressed as follows:

$$min\left(w_1 \cdot \sum_{j\in J} max_{k\in M}\, endOf(O[j,o_{last},k])\right)$$

$$+\left(w_2 \cdot \sum_{j\in J}\sum_{o\in O_j}\sum_{k\in M} PresenceOf(O[j,o,k]) \cdot energy[j][o][k] \cdot emissionFactor[k]\right)$$

Under constraint

$$alternative(O[j,o],\{O[j,o,k]|\ k\ \in\ M\} \quad \forall j\in J, \forall o\in O_j \tag{9}$$

$$endBeforeStart\big(O[j,o,k_1],O[j,o+1,k_2] \quad \forall j\in J, \forall o\in O_j\backslash\{o_{last}\}, \forall k_1,k_2\in M \tag{10}$$

$$NoOverlap(S_k) \quad \forall\ k\ \in\ M \tag{11}$$

$$endOf(O[j,o_{last},k]) \geq endOf(O[j,o,k]) \quad \forall j\in J, \forall o\in O_j, \forall k\ \in\ M \tag{12}$$

Using a weighted aggregation with weights w_1 and w_2,the objective function simultaneously minimizes the total flow time and total carbon emissions. The *alternative()* constraint (9) ensures that each operation is assigned to exactly one machine. By ensuring that each operation begins only after the completion of its immediate predecessor, the precedence constraint (10) ensures that the operations within the same job are processed in the proper order. The ***NoOverlap*** constraint (11) ensures that no two operations overlap on the same machine, thereby enforcing the unary resource constraint. Constraint (12) defines the **completion time of each job** as the latest end time among all operations of that job.

4.2 *Particle swarm optimization*

In addition to the development of a CP model, an innovative PSO metaheuristic was proposed to solve the studied GFJSP. This method was a stochastic and population-based optimization inspired by the social behavior observed in bird flocks or fish schools. The search space of this algorithm is considered to be d-dimensional, and each particle $\boldsymbol{l}$ in the swarm represents a potential solution, defined as a position vector $\boldsymbol{X}=(\boldsymbol{x_{l1}},\boldsymbol{x_{l2}},\dots,\boldsymbol{x_{ld}})$ and a velocity vector $\boldsymbol{V}=(\boldsymbol{\vartheta_{l1}},\boldsymbol{\vartheta_{l2}},\dots,\boldsymbol{\vartheta_{ld}})$.

- **Particle encoding**

To enhance solution diversity and embed energy-awareness into the optimization process, we propose a novel encoding strategy based on three continuous-valued vectors: Temporal Execution Priority Vector (TEPV), Machine Efficiency Preference Vector (MEPV), and Dispatchability Perturbation Vector (DPV). Each of size N, where N is the total number of operations in the GFJSP instance. This hybrid encoding is well-suited for PSO and explicitly captures time sensitivity, machine efficiency, and controlled randomness in scheduling decisions.

Each element in the TEPV is a real number in the range [0,1] representing the urgency of scheduling the corresponding operation. Operations are sorted in ascending order of their TEPV values to determine the scheduling sequence; lower values correspond to earlier execution priority. Unlike traditional priority encodings, TEPV can flexibly incorporate external scheduling constraints such as due dates, peak energy periods, or time-dependent carbon taxation, thereby increasing the realism and responsiveness of the scheduling process.

Additionally, the MEPV is a vector of real values in [0,1], each of which represents an operation. Based on a Machine Efficiency Index (MEI), it encodes the preference for assigning the operation to one of its eligible machines. A variety of component, such as processing time, energy consumption, and carbon emission factors, are combined in the MEI. The MEI (13) is defined as follows for each operation O_{ij} and eligible machine M_k:

$$MEI_{ij}^{k} = \alpha \cdot \frac{1}{P_{ij}^{k}} + \beta \cdot \frac{1}{E_{ij}^{k}} + \gamma \cdot \frac{1}{C_k} \tag{13}$$

Where α, β, and γ are user-defined weights that reflect the decision-maker's preferences. The machines are ranked in descending order of their MEI values, and the MEPV value is proportionally mapped to this ranked list to determine the machine assignment.

A controlled perturbation mechanism is introduced into the decoding process by the DPV, a third real-valued vector in [0,1]. During decoding, it adds adaptive randomness by modulating both TEPV and MEPV. Particularly in subsequent iterations, this stochastic modulation helps diversify the search and avoids premature convergence. A higher DPV value corresponds to stronger deviation from the deterministic decoding path, thereby encouraging exploration of alternative yet still feasible schedules.

The TEPV–MEPV–DPV encoding scheme offers a compact yet powerful representation that integrates time sensitivity, energy efficiency, and decision flexibility. It encourages the creation of more environmentally friendly schedules by directly integrating energy and emission factors into machine selection. While the DPV vector adds controlled randomness that strengthens global search capabilities, its responsiveness is improved by its flexibility in response to external factors like dynamic energy pricing or emission constraints. All things considered, this hybrid encoding offers a strong basis for creating effective and environmentally conscious metaheuristic solutions to the GFJSP.

To demonstrate the effectiveness of the proposed particle representation, we make use of the simplified GFJSP instance previously introduced in Table 1, which involves 4 jobs and 3 machines. The total number of operations is 10, so each particle is encoded as three real-valued vectors of length 10:

Table 5 Proposed encoding scheme

Operation	$O_{1,1}$	$O_{2,1}$	$O_{3,1}$	$O_{1,2}$	$O_{2,2}$	$O_{3,2}$	$O_{1,3}$	$O_{2,3}$	$O_{1,4}$	$O_{2,4}$
Index	1	2	3	4	5	6	7	8	9	10
TEPV	0.35	0.88	0.12	0.60	0.45	0.21	0.75	0.33	0.50	0.15
MEPV	0.60	0.20	0.90	0.25	0.70	0.45	0.80	0.10	0.55	0.30
DPV	0.05	0.12	0.10	0.08	0.03	0.15	0.04	0.20	0.18	0.06

Let δ = 0.2 be the perturbation parameter for decoding TEPV (Table 5).

- **Particle decoding**

To ensure feasibility and transform the encoded particle representation into a valid schedule, a decoding procedure is applied in three stages, corresponding to the three components of the proposed encoding scheme: Temporal Execution Priority Vector (TEPV), Machine Efficiency Preference Vector (MEPV), and Dispatchability Perturbation Vector (DPV). This procedure guarantees adherence to the routing and scheduling constraints of the GFJSP.

Step 1: Operation sequencing from TEPV and DPV

Operations are first sorted in ascending order of their TEPV values. To avoid strict determinism and encourage search diversification, each TEPV value is adjusted using its corresponding DPV component. The adjusted priority score is calculated as (14):

$$Priority_{index} = TEPV_{index} + \delta \cdot DPV_{index} \tag{14}$$

Where δ ∈ [0,1] is a perturbation parameter controlling the influence of randomness. While accepting slight changes that may help in escaping local optima, this controlled noise preserves the overall order. The final scheduling sequence is determined by sorting operations based on these adjusted priority values. Using the illustrative example, the adjusted value softens the strict priority order by incorporating a small perturbation from the DPV, helping the algorithm to avoid deterministic traps. The adjusted $\boldsymbol{Priority_{index}}$ values are (Table 6):

Table 6 Adjusted $Priority_{index}$

Operation Index	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$Priority_{index}$	0.36	0.904	0.14	0.616	0.456	0.24	0.758	0.37	0.536	0.162

The operations are then sorted in ascending order of their Priority values, resulting in the following scheduling sequence:
[3,10,1,8,6,5,4,9,7,2] ⇒ [$O_{3,1}$, $O_{2,4}$, $O_{1,1}$, $O_{2,3}$, $O_{3,2}$, $O_{2,2}$, $O_{1,2}$, $O_{1,4}$, $O_{1,3}$, $O_{2,1}$]

Step 2: Machine assignment (Based on MEPV):

For each operation, the MEPV value is used to select one of the eligible machines. Assuming full eligibility of all machines (M1, M2, M3) for this illustration, the selected machine is calculated using (15):

$$Machine_k = [MEPV_k \times 3] + 1 \tag{15}$$

This yields the following machine assignments (Table 7):

Table 7 Machine assignment using MEPV decoding

	$O_{1,1}$	**$O_{2,1}$**	**$O_{3,1}$**	**$O_{1,2}$**	**$O_{2,2}$**	**$O_{3,2}$**	**$O_{1,3}$**	**$O_{2,3}$**	**$O_{1,4}$**	**$O_{2,4}$**
Eligible Machines	M1, M2, M3	M1, M2, M3	M1, M2, M3	M1, M2, M3	M1, M2, M3	M1, M2, M3	M2, M3	M1, M2, M3	M1, M2, M3	M1, M2, M3
MEPV	**0.60**	**0.20**	**0.90**	**0.25**	**0.70**	**0.45**	**0.80**	**0.10**	**0.55**	**0.30**
Assigned Machine	M2	M1	M3	M1	M3	M2	M3	M1	M2	M1

Step 3: Schedule construction with constraint enforcement

Based on the sequencing and machine assignment obtained from the previous steps, the final schedule is constructed by enforcing key problem constraints. Precedence constraints are satisfied by ensuring that each operation starts only after the completion of its predecessor within the same job. To guarantee non-overlapping operations, machine availability is tracked, and each operation is assigned to the earliest feasible time slot on its designated machine. If any constraint violations occur, such as incorrect operation ordering or resource conflicts, they are penalized through a dynamic penalty function integrated into the fitness evaluation, thereby steering the optimization process toward feasible and high-quality solutions. This decoding mechanism effectively transforms each continuous-valued particle into a discrete and feasible schedule, allowing the hybrid PSO to navigate the solution space of the GFJSP while incorporating energy efficiency and carbon emissions into considerations.

- **Population initialization using logistic map**

Instead of using only uniform random sampling, a chaotic logistic map is used to create the initial swarm of particles in order to guarantee diversity and viability right away. The pseudo-random numbers produced by this nonlinear map, which is defined as $l_{n+1} = r \cdot l_n \cdot (1 - l_n)$ with $r = 4$, improve population diversity and aid in escaping local optima. The logistic sequence is used to determine the values of the triplet of continuous vectors (TEPV, MEPV, and DPV) that are initialised for each particle. The steps outlined in Section 4.2 are followed during the decoding process. Energy-conscious criteria incorporated into the MEI impact machine assignments during initialization, biassing the population towards more environmentally friendly scheduling arrangements. This hybrid initialisation approach preserves practical feasibility while improving early exploration.

- **Fitness evaluation**

Each decoded particle corresponds to a complete and feasible schedule, which is evaluated using a multi-objective fitness function. Since classical PSO requires a scalar fitness value, a normalized weighted sum approach is adopted to aggregate the two objectives (16):

$$Fitness_l = w_1 \cdot \frac{FT_l}{FT_{ref}} + w_2 \cdot \frac{CE_l}{CE_{ref}} \tag{16}$$

Where FT_l and CE_l are the total flow time and carbon emissions of particle l, FT_{ref} and CE_{ref} are reference values, typically obtained from an initial heuristic solution. These reference values are computed once at the beginning of the optimization process and remain constant throughout all iterations, to ensure consistency and fairness in the evaluation of particles. The weights w_1, w_2 are user-defined and belong to the interval $[0, 1]$, with the constraint $w_1 + w_2 = 1$. They

Acta Logistica, Volume 13, Issue 1, Pages 114-126, 2026 ISSN 1339-5629

reflect the decision-maker's priorities between productivity and sustainability. By adjusting these weights, different trade-offs between the two objectives can be explored.

- **Velocity and position update**

The standard PSO update equations are adapted to handle the continuous encoding of the TEPV–MEPV–DPV scheme. Each particle's velocity and position are updated using (17):

$$\vartheta_l^{t+1} = \omega \cdot \vartheta_l^t + c_1 \cdot r_1 \cdot (p_l^t - x_l^t) + c_2 \cdot r_2 \cdot (g^t - x_l^t)$$
$$x_l^{t+1} = x_l^t + \vartheta_l^{t+1} \tag{17}$$

Where:
- x_l^t, ϑ_l^t: current position and velocity of particle l at iteration t,
- p_l^t: best-known position of particle i,
- g^t: global best-known position,
- ω: inertia weight,
- c_1, c_2: cognitive and social learning factors,
- r_1, r_2: random numbers uniformly drawn from [0,1].

After each update, particle components (TEPV, MEPV, DPV) are clipped to remain within [0,1], ensuring compatibility with the decoding process. To improve exploration in the discrete domain, a sigmoid-based probability transformation is optionally applied to velocity values, influencing the likelihood of structural changes in scheduling decisions.

- **Stopping Criteria**

To guarantee computational effectiveness and solution quality, the PSO algorithm ends based on two conditions. First, if the maximum number of iterations (T_{max}), which stands for a predetermined exploration limit, is reached, the algorithm stops. Second, if there is no improvement in the global best fitness value over a series of iterations (T_{stall}), which indicates convergence stagnation, an early stopping mechanism is initiated. In order to avoid needless computational effort, this set of criteria makes sure that the search process ends either after enough investigation or when more iterations are unlikely to yield better answers.

5 Results and discussion

To evaluate the performance of the proposed methods, computational experiments were carried out on a computer with Intel i7, 16 GB RAM under a Windows XP environment. The PSO was implemented in the programming language C# and the CP model was implemented using the Cplex CP Optimizer solver. To evaluate the performance of the proposed methods on small-sized instances, a dataset comprising ten classes of ten randomly generated instances each was constructed. The number of jobs was selected from the set {4, 8, 10, 12, 15, 20, 50}, while the number of operations per job was randomly drawn from the range {2, ..., 10}. The number of available machines was chosen from {5, 7, 10, 15, 20}. Processing times were generated according to a uniform distribution over the interval [1, 100]. For medium- and large-scale instances, the benchmark dataset used is the well-known rdata collection proposed by [30]. In this dataset, the number of jobs varies between 6 and 20, the number of machines ranges from 5 to 15, the number of operations per job spans from 5 to 15, and the total number of operations across all jobs ranges from 36 to 300.

5.1 Parameter settings

The PSO parameters were carefully calibrated to ensure a good balance between exploration and exploitation of the search space. The swarm size was set to 100 particles, and the maximum number of iterations T_{max} was fixed at 1000 and T_{stall} at 20. The inertia weight ω was decreased linearly from 0.9 to 0.4 over the iterations to balance global and local search. The cognitive and social acceleration coefficients c_1 and c_2 were both set to 2. The multi-objective fitness function aggregated the two objectives using a normalized weighted sum, as described in Section 4.1, with weights $w_1 = 0.5$ and $w_2 = 0.5$. These reference values remain constant throughout the optimization process to maintain a consistent evaluation framework. The carbon emission factor F_k of the machines is 0.54 (kg CO_2 /kWh).

5.2 Exact resolution results

In order to evaluate the CP model, we generate 6 classes of 20 instances (*Cl1-Cl6*) with several combinations for the parameters $n \in \{6, 10, 15, 20\}$ and $m \in \{5, 6, 10\}$. The processing times are randomly generated using the uniform distribution over the interval [1, 100]. Table 8 summarizes the average, minimum, and maximum computation times in seconds over 20 generated instances for each class configuration. The time limit of 600s was imposed: any instance not solved within this limit was considered unsolvable. The results show that the CP model was very efficient and able to solve instances up to 100 operations, a significant size compared to problems handled by other exact solution approaches. The model reliably solves *Cl1–Cl3* (≤ 75 operations) in under 9 seconds, affirming its strength on small to medium problem sizes. For *Cl4*, the average CPU time of 15 s and a 100% solve rate indicate that job count, more than total operations, drives CP complexity. In *Cl5*, the problem density increases; although many instances finish quickly, complexity spikes produce outliers up to 55 s, giving a 92% solve rate. Finally, *Cl6* presents the highest complexity: average time is 38 s, but difficult cases require up to 120 s, so 14% of instances timed out, yielding an 86% success rate (Table 8).

Table 8 Resolution results for CP model

Class	Size $(n \times m, \sum n_j)$	CPU$_{Avg}$ (in s)	{Min, Max} Times (in s)	% Resolution
Cl1	(6×6, 36)	0.1	{0.03, 0.13}	100%
Cl2	(10×5, 50)	0.1	{0.06, 0.2}	100%
Cl3	(15×5, 75)	4	{1.2, 8.5}	100%
Cl4	(10×5, 100)	0.1	{5, 30}	100%
Cl5	(10×10, 100)	12	{3, 55}	92%
Cl6	(20×5, 100)	38	{10, 120}	86%

5.3 PSO evaluation results

Table 9 presents the performance results obtained by the implemented PSO algorithm. The second column indicates the size of each class of the rdata benchmark, expressed as $(n \times m, \sum n_j)$. To evaluate the quality of the proposed metaheuristic, we compute the Relative Percentage Deviation (RPD) for each objective: total flow time (f_1) and total carbon emissions (f_2). For each instance, the PSO algorithm is executed 20 times. The best result among these runs is used as a reference to compute the RPD of that instance. At the class level (comprising 5 instances), the RPD is calculated as the average of the RPD values obtained for each instance within the class:

$$RPD_1^{Class} = \frac{1}{5}\sum_{I=1}^{5}[\frac{1}{20}\sum_{r}^{20}(\frac{f_{1,I}^{r} - f_{1,I}^{best}}{f_{1,I}^{best}}) \times 100]; \qquad RPD_2^{Class} = \frac{1}{5}\sum_{I=1}^{5}[\frac{1}{20}\sum_{r}^{20}(\frac{f_{2,I}^{r} - f_{2,I}^{best}}{f_{2,I}^{best}}) \times 100]$$

Where $f_{1,I}^{r}$ and $f_{2,I}^{r}$ are the total flow time and carbon emissions obtained at run r on instance I, and $f_{1,I}^{best}$ and $f_{2,I}^{best}$ are the best values obtained across 20 runs on instance I. To assess the robustness of the method, we also calculate the standard deviation of the relative deviations across the 5 instances in each class. These deviations are first computed per run as:

$$Dev_{1,I}^{r} = \left(\frac{f_{1,I}^{r} - f_{1,I}^{best}}{f_{1,I}^{best}}\right) \times 100; \qquad Dev_{2,I}^{r} = \left(\frac{f_{2,I}^{r} - f_{2,I}^{best}}{f_{2,I}^{best}}\right) \times 100$$

The standard deviation per benchmark class is then computed as the standard deviation of the average deviations per instance: $\sigma_1^{Class} = StdDev(\{\frac{1}{20}\sum_{r=1}^{20} Dev_{1,I}^{r}\}_{I=1}^{5})$ and $\sigma_2^{Class} = StdDev(\{\frac{1}{20}\sum_{r=1}^{20} Dev_{2,I}^{r}\}_{I=1}^{5})$. This evaluation provides insight into both the average performance and the consistency of the PSO algorithm across different instances and benchmark classes.

Table 9 PSO performance for GFJSP from rdata [30]

Benchmark Class	*Size* $(n \times m, \sum n_j)$	RPD_1 (%)	RPD_2 (%)	$StdDev_1$(%)	$StdDev_2$(%)	*CPU$_{Avg}$ (in s)*
la01-la05	(10×5, 50)	1.45	1.12	0.45	0.35	0.75
la06-la010	(15×5, 75)	3.05	2.34	0.85	0.60	3.82
la11-la15	(20×5, 100)	6.21	3.58	1.75	1.20	68.35
la16-la20	(10×10, 100)	5.38	4.80	1.30	1.10	95.58
la21-la25	(15×10, 150)	9.73	6.22	2.25	1.80	110.10
la26-la30	(20×10, 200)	12.05	7.66	2.80	2.15	175.05
la31-la35	(30×10, 300)	10.89	8.35	3.10	2.45	291.42
la36-la40	(15×15, 300)	13.12	8.70	3.45	2.70	336.15
Average		**7.73**	**5.35**	**1.99**	**1.55**	**135.15**

The PSO performs effectively across medium- and large-sized instances. For the smallest classes (la01–la05), RPD_1 and RPD_2 remain low (1.45% and 1.12%, respectively) with minimal variability and negligible computation time (0.75 s), demonstrating efficiency and robustness. Moderately sized classes (la06–la15) maintain high solution quality (RPDs 3.05–6.21%) with low dispersion among runs. For larger instances (la26–la40), RPD values and variability increase moderately, reflecting the expected challenges in extensive search spaces, yet the algorithm remains competitive in solution quality and computation time (average 175–291 s). Overall, the PSO balances solution quality, stability, and computational efficiency, with average RPD_1 and RPD_2 of 7.73% and 5.35%, and standard deviations of 1.99% and 1.55%, confirming its suitability for solving GFJSP.

The ANOVA analysis was executed on RPD values across benchmark classes to determine whether mean differences were statistically significant, evaluating the method's stability and robustness to increased instance size and complexity. The results indicate a statistically significant difference between classes with *p-value* < 0.05. A closer examination of the box plots, represented in Figure 1, revealed a rather tight distribution for classes la01-la05 and la06-la10, indicating that variance among these classes is low, and the PSO algorithm can be assumed to be highly robust for small-sized problems. As the instance size increases, especially starting from class la31-la35, a trend is seen in both the median RPD and the interquartile range to increase, suggesting a subsequent drop in solution quality with an increase in variation. This being said, very few outliers across the entire set of classes lend credence to the conclusion of algorithm robustness. In conclusion, PSO performs well and is stable on small to medium-sized instances, with certain gradual degradation observed on larger problem sets.

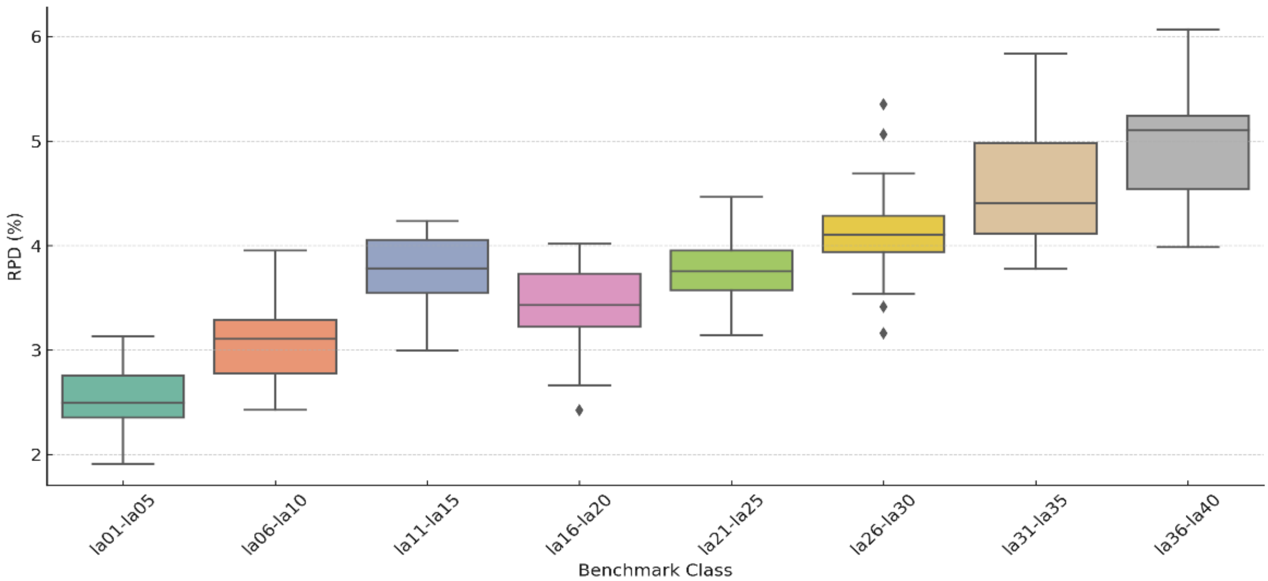

Figure 1 Box plots of RPD values across benchmark classes (20 runs per instance)

5.4 Discussion

The findings of this study highlight the relevance of proposing the two complementary methods, CP and PSO, enhanced with an energy-aware encoding scheme and population initialization via a logistic map, for addressing the GFJSP. Unlike traditional approaches that primarily aim to minimize makespan [3], results demonstrate that it is possible to simultaneously optimize both operational efficiency and sustainability criteria, particularly carbon emissions. This dual-objective perspective reflects a broader trend in industrial logistics research, where material, human, and energy flows are increasingly managed with sustainability considerations in mind. When benchmarked against existing heuristics and metaheuristics, the two proposed methods show competitive performance in reducing logistics and production flow while minimizing environmental impact. The sensitivity analysis and ANOVA tests further confirm the robustness of the PSO: changes in key algorithmic parameters lead to only minor variations in performance, while statistical analyses reveal that differences between solutions are significant. These results highlight both the stability and reliability of the proposed approach across small, medium, and large-scale problem instances.

Beyond numerical improvements, the model emphasizes the importance of simultaneously managing multiple flows, providing a more realistic representation of the complexity inherent in industrial logistics. From a managerial perspective, this approach offers a practical and structured tool to assess trade-offs between productivity and environmental responsibility. For example, minimizing waiting times while reducing energy consumption can directly translate into lower operational costs and enhanced compliance with sustainability regulations.

Nevertheless, some limitations remain. Real-time disruptions and information flows are not modeled, and the experiments rely on benchmark datasets that may not capture all real-world variability. Future research could explore adaptive PSO strategies, stochastic disruptions, or integration with IoT and digital twin technologies to further improve responsiveness, robustness, and sustainability in dynamic, Industry 4.0 contexts.

Overall, the discussion emphasizes that these complementary methods not only contribute to academic knowledge but also offer practical guidance for industrial logistics managers seeking to align operational performance with environmental responsibility.

6 Conclusion and future research

This paper addressed the optimization of logistics and production flows in a green flexible job-shop environment, with the dual aim of enhancing operational efficiency (total flow time) and promoting sustainable performance (carbon emissions). Two complementary methods were developed: an exact CP model for small-sized instances, and a PSO algorithm enhanced with an energy-aware encoding scheme and population initialization via a logistic map for medium and large-scale problems. The results show that the proposed framework effectively reduces both total flow time and carbon emissions, demonstrating the value of combining exact and heuristic methods to tackle complex industrial logistics challenges. From a practical perspective, the methods improve material flow coordination, reduce waiting times, and promote more sustainable use of resources in flexible manufacturing systems. The CP model ensures optimal solutions for smaller instances, while the PSO algorithm produces high-quality solutions for larger instances within reasonable computation times. The sensitivity analysis confirmed the robustness of the PSO under varying parameter settings, further supporting the reliability of the proposed approach. This demonstrates how exact and metaheuristic methods can complement each other to address sustainable scheduling and flow management in industrial logistics.

Despite the promising results, some limitations remain, including the exclusion of information flows and real-time disruptions, which are increasingly relevant in Industry 4.0 contexts. Incorporating these aspects would increase the applicability of the scheduling models. Future research could explore multi-swarm or cooperative PSO strategies to improve convergence and solution quality in large-scale instances, and integrate stochastic disruptions to reflect dynamic production conditions. Additionally, combining PSO with machine learning models may enable predictive and adaptive scheduling, allowing the system to adjust in real time to production changes. The use of IoT-enabled data streams and digital twins could further enhance responsiveness, operational efficiency, and sustainability in modern flexible job-shop environments. Overall, this study provides both scientific insights and practical guidance for managing industrial logistics in a sustainable manner, offering a foundation for continued research in adaptive, data-driven, and environmentally conscious scheduling.

References

[1] ZHANG, H., XU, G., PAN, R., GE, H.: A novel heuristic method for the energy-efficient flexible job-shop scheduling problem with sequence-dependent set-up and transportation time, *Engineering Optimization*, Vol. 54, No. 10, pp. 1646-1667, 2021. https://doi.org/10.1080/0305215X.2021.1949007

[2] GAREY, M.R., JOHNSON, D.S., SETHI, R.: The Complexity of Flow Shop and Job-Shop Scheduling, *Mathematics of Operations Research*, Vol. 1, No. 2, pp. 117-129, 1976. https://doi.org/10.1287/moor.1.2.117

[3] PINEDO, M.L.: *Scheduling: Theory, Algorithms and Systems*, Prentice-Hall, Englewood Cliffs, New Jersey, USA, 2002. https://doi.org/10.1007/978-0-387-78935-4

[4] ZHANG, S., DU, H., BORUCKI, S., JIN, S., HOU, T., LI, Z.: Dual Resource Constrained Flexible Job Shop Scheduling Based on Improved Quantum Genetic Algorithm, *Machines*, Vol. 9, No. 6, 108, pp. 1-15, 2021. https://doi.org/10.3390/machines9060108

[5] CITARISTI, I.: *International Energy Agency—IEA*, *The Europa Directory of International Organizations 2022*, 24th ed., Routledge, pp. 701-702, 2022.

[6] LIU, C.H., HUANG, D.H.: Reduction of power consumption and carbon footprints by applying multi-objective optimisation via genetic algorithms, *International Journal of Production Research*, Vol. 52, No. 2, pp. 337-352, 2013. https://doi.org/10.1080/00207543.2013.825740

[7] LIU, C.H.: Approximate trade-off between minimisation of total weighted tardiness and minimisation of carbon dioxide (CO_2) emissions in bi-criteria batch scheduling problem, *International Journal of Computer Integrated Manufacturing*, Vol. 27, No. 8, pp. 759-771, 2013. https://doi.org/10.1080/0951192X.2013.834479

[8] TANG, Y., SHEN, L., HAN, S.: Low-carbon flexible job shop scheduling problem based on deep reinforcement learning, *Sustainability*, Vol. 16, No. 11, 4544, pp. 1-23, 2024. https://doi.org/10.3390/su16114544

[9] MEI, Z., LU, Y., LV, L.: Research on multi-objective low-carbon flexible job shop scheduling based on improved NSGA-II, *Machines*, Vol. 12, No. 9, 590, pp. 1-15, 2024. https://doi.org/10.3390/machines12090590

[10] PIROOZFARD, H., WONG, K.Y., WONG, W.P.: Minimizing total carbon footprint and total late work criterion in flexible job shop scheduling by using an improved multi-objective genetic algorithm, *Resources, Conservation and Recycling*, Vol. 128, pp. 267-283, 2018. https://doi.org/10.1016/j.resconrec.2016.12.001

[11] WEN, X., FU, Y., YANG, W., WANG, H., ZHANG, Y., SUN, C.: An effective hybrid algorithm for joint scheduling of machines and AGVs in flexible job shop, *Measurement and Control*, Vol. 56, No. 9-10, pp. 1582-1598, 2023. https://doi.org/10.1177/00202940231173750
[12] LUAN, F., CAI, Z., WU, S., LIU, S.Q., HE, Y.: Optimizing the low-carbon flexible job shop scheduling problem with discrete whale optimization algorithm, *Mathematics*, Vol. 7, No. 8, 688, pp. 1-17, 2019. https://doi.org/10.3390/math7080688
[13] ZHU, H., DENG, Q., ZHANG, L., HU, X., LIN, W.: Low carbon flexible job shop scheduling problem considering worker learning using a memetic algorithm, *Optimization and Engineering*, Vol. 21, pp. 1691-1716, 2020. https://doi.org/10.1007/s11081-020-09494-y
[14] WANG, S., LI, J., TANG, H., WANG, J.: CEA-GFJSP: Carbon emission-aware flexible job-shop scheduling based on deep reinforcement learning, *Frontiers in Environmental Science*, Vol. 10, 1059451, pp. 1-13, 2022. https://doi.org/10.3389/fenvs.2022.1059451
[15] LI, Z., CHEN, Y.: Minimizing the makespan and carbon emissions in the green flexible job shop scheduling problem with learning effects, *Scientific Reports*, Vol. 13, 6369, pp. 1-15, 2023. https://doi.org/10.1038/s41598-023-33615-z
[16] JIA, S., YANG, Y., LI, S., WANG, S., LI, A., CAI, W., LIU, Y., HAO, J., HU, L.: The green flexible job-shop scheduling problem considering cost, carbon emissions, and customer satisfaction under time-of-use electricity pricing, *Sustainability*, Vol. 16, No. 6, 2443, pp. 1-22, 2024. https://doi.org/10.3390/su16062443
[17] YI, W., LI, X., PAN, B.: Solving flexible job shop scheduling using an effective memetic algorithm, *International Journal of Computer Applications in Technology*, Vol. 53, No. 2, pp. 157-163, 2016. https://doi.org/10.1504/IJCAT.2016.074454
[18] MARZOUKI, B., DRISS, O.B., GHEDIRA, K.: Multi-agent model based on combination of chemical reaction optimisation metaheuristic with Tabu search for flexible job shop scheduling problem, *International Journal of Intelligent Engineering Informatics*, Vol. 6, pp. 242-265, 2018. https://doi.org/10.1504/IJIEI.2018.091875
[19] SHEN, L., DAUZÈRE-PÉRÈS, S., NEUFELD, J.S.: Solving the flexible job shop scheduling problem with sequence-dependent setup times, *European Journal of Operational Research*, Vol. 265, No. 2, pp. 503-516, 2018. https://doi.org/10.1016/j.ejor.2017.08.021
[20] BEKKAR, A., BELALEM, G., BELDJILALI, B.: Iterated greedy insertion approaches for the flexible job shop scheduling problem with transportation times constraint, *International Journal of Manufacturing Research*, Vol. 14, No. 1, pp. 43-66, 2019. https://doi.org/10.1504/IJMR.2019.096746
[21] CALDEIRA, R.H., GNANAVELBABU, A.: Solving the flexible job shop scheduling problem using an improved Jaya algorithm, *Computers & Industrial Engineering*, Vol. 137, 106064, 2019. https://doi.org/10.1016/j.cie.2019.106064
[22] MENG, T., PAN, Q.K., SANG, H.Y.: A hybrid artificial bee colony algorithm for a flexible job shop scheduling problem with overlapping in operations, *International Journal of Production Research*, Vol. 56, pp. 5278-5292, 2018. https://doi.org/10.1080/00207543.2018.1467575
[23] FEKIH, A., HADDA, H., KACEM, I., HADJ-ALOUANE, A.B.: A hybrid genetic Tabu search algorithm for minimising total completion time in a flexible job-shop scheduling problem, *European Journal of Industrial Engineering*, Vol. 14, No. 6, pp. 763-781, 2020. https://doi.org/10.1504/EJIE.2020.112479
[24] HUANG, S., TIAN, N., WANG, Y., JI, Z.: Multi-objective flexible job-shop scheduling problem using modified discrete particle swarm optimization, *Springerplus*, Vol. 5, No. 1, 1432, 2016. https://doi.org/10.1186/s40064-016-3054-z
[25] VITAL-SOTO, A., BAKI, M.F., AZAB, A.: A multi-objective mathematical model and evolutionary algorithm for the dual-resource flexible job-shop scheduling problem with sequencing flexibility, *Flexible Services and Manufacturing Journal*, Vol. 35, pp. 626-668, 2023. https://doi.org/10.1007/s10696-022-09446-x
[26] GIRISH, B.S., JAWAHAR, N.: *A particle swarm optimization algorithm for flexible job shop scheduling problem*, 2009 IEEE International Conference on Automation Science and Engineering, Bangalore, India, 2009, pp. 298-303, 2009. https://doi.org/10.1109/COASE.2009.5234153
[27] SUN, K., ZHENG, D., SONG, H., CHENG, Z., LANG, X., YUAN, W., WANG, J.: Hybrid genetic algorithm with variable neighborhood search for flexible job shop scheduling problem in a machining system, *Expert Systems with Applications*, Vol. 215, 119359, 2023. https://doi.org/10.1016/j.eswa.2022.119359
[28] HUTTER, D., STEINBERGER, T., HELLWIG, M.: A Benchmarking Environment for Worker Flexibility in Flexible Job Shop Scheduling Problems, *Neural and Evolutionary Computing*, arXiv preprint arXiv:2501.16159, Vol. 2025, pp. 1-40, 2025. https://arxiv.org/abs/2501.16159
[29] KACEM, I., HAMMADI, S., BORNE, P.: Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic, *Mathematics and Computers in Simulation*, Vol. 60, No. 3-5, pp. 245-276, 2002. https://doi.org/10.1016/S0378-4754(02)00019-8

[30] HURINK, J., JURISCH, B., THOLE, M.: Tabu search for the job-shop scheduling problem with multi-purpose machines, *OR Spectrum*, Vol. 15, No. 4, pp. 205-215, 1994.

Review process

Single-blind peer review process.