International Journal of Industrial Engineering Computations 16 (2025) ***–***

Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

An adaptive local search for large-scale parallel machine scheduling in textile production with release dates and sequence-dependent setup times

Mariane Emanuelle Pessoa Santos[a*], Yuri Laio Teixeira Veras Silva[a] and Maria Creuza Borges de Araújo[a]

[a]*Federal University of Campina Grande, Brazil*

C H R O N I C L E

Article history:
Received December 31 2024
Received in Revised Format March 9 2025
Accepted April 4 2025
Available online April 4 2025

Keywords:
Scheduling
Parallel machines
Local search
Total tardiness minimization
Release dates

A B S T R A C T

This study proposes an adaptive local search heuristic to solve a real-world large-scale parallel machine scheduling problem with release dates and setup times, aiming to minimize total tardiness. The complexity of the problem stems from the need to synchronize machine availability, job release dates, and setup durations, which are crucial for meeting production deadlines and ensuring operational efficiency. Traditional optimization approaches often struggle to deliver timely solutions for large-scale industrial applications. Our heuristic method effectively explores the search space to identify schedules that significantly reduce total tardiness while adhering to the constraints of the production system. The approach was tested using real production data, and the results indicate that the heuristic consistently generated high-quality solutions within short computational times. The approach proved viable and efficient, offering a practical tool for improving scheduling performance and minimizing total tardiness in industries with similar operational constraints.

1. Introduction

In recent years, the manufacturing industry has faced increasing pressure to optimize production processes due to rising competition, evolving customer demands, and the need for cost-efficiency. One of the key challenges within the manufacturing sector, particularly in the textile industry, is scheduling production on parallel machines. The textile industry is characterized by a high degree of product variety and fluctuating order volumes, which can make production planning complex (Albayrak & Onuet, 2024; Pei et al., 2021; Fuchigami & Rangel, 2018). Specifically, textile companies often deal with parallel machine scheduling problems (PMSP), where multiple machines operate concurrently, and each job must be assigned to a machine while considering operational constraints such as release dates and setup times (Wang & Zhang, 2023). These constraints significantly impact production efficiency, and their improper handling can lead to delays, increased costs, and lower customer satisfaction. The PMSP is a well-known NP-hard problem, meaning that finding an optimal solution becomes computationally intractable as the problem size grows (Liu et al., 2020; Wu & Che, 2019). In practice, exact methods often prove inefficient due to the complexity of real-world scheduling scenarios, particularly when operational constraints like release dates and setup times are considered. Release dates dictate when a job becomes available for processing, while setup times represent the time required to prepare a machine for a specific job. Both factors must be accounted for in an efficient production schedule to minimize total tardiness and optimize resource utilization (Hu et al., 2024; Safarzadeh & Niaki, 2023). Setup times, particularly sequence-dependent setup times (SDST), play a critical role in the textile production process. Different products often require different machine settings, material changes, or even cleaning operations between jobs, all of which contribute to downtime if not managed effectively (Xue et al., 2024; Kim & Kim, 2020). The scheduling problem becomes particularly challenging when both setup time optimization and release date constraints must be addressed concurrently. This dual requirement compounds the computational difficulty of generating efficient production schedules, especially in industries with high product mix variability such as textile manufacturing (Wu et al., 2025).

* Corresponding author
E-mail marianeperson1@gmail.com (M.E.P. Santos)
ISSN 1923-2934 (Online) - ISSN 1923-2926 (Print)
2025 Growing Science Ltd.
doi: 10.5267/j.ijiec.2025.4.004

Heuristics have long been a popular choice for solving complex scheduling problems due to their ability to provide good-quality solutions within a reasonable computational time, especially for NP-hard problems like PMSP with SDST and release dates (Krimi & Benmansour, 2024; Lee & Kim, 2021). Unlike exact methods, which guarantee optimal solutions but can be computationally expensive for large-scale problems, heuristics offer a trade-off between solution quality and computational efficiency, making them particularly attractive for real-world industrial applications where quick decision-making is crucial (Huynh & Chien, 2018; Prasad et al., 2022).

This paper proposes a novel heuristic approach to treat the identical parallel machine scheduling problem with release dates and setup times, specifically tailored for a real-world textile manufacturing environment. The proposed heuristic is based on a local search strategy that iteratively explores the solution space by reassigning jobs to different machines and adjusting job sequences to minimize total tardiness, a common objective in industries where meeting customer deadlines is critical (Li et al., 2024; Demirtas, 2022). The development of this heuristic was motivated by the need for a method that could handle the large-scale, complex scheduling requirements of the textile industry while providing solutions that are both practical and efficient.

The real-world application of the proposed heuristic was tested in a textile company that produces a wide range of fabrics, each with varying production requirements. The company's scheduling problem involved multiple parallel machines, with each machine capable of processing different types of fabrics but requiring significant setup times when switching between jobs. Additionally, the release dates of different fabric orders added another layer of complexity to the scheduling process, as jobs could not be scheduled in a continuous manner. The proposed heuristic reduces mean total tardiness by 27.4% compared to standard dispatching rules in computational experiments on industrial datasets. The method achieves strict dominance (100% outperformance rate) across all test instances, with a feasible average runtime for large-scale problems (200+ jobs, 40+ parallel machines), demonstrating both solution quality and scalability for real-world deployment.

2. Literature review

The scheduling problem is a fundamental challenge in operations research and computer science, involving the allocation of resources over time to perform a collection of tasks/jobs. This problem is crucial in various areas, including manufacturing, healthcare, and project management, where efficient scheduling can significantly enhance productivity and reduce operational costs (Meng et al., 2024; Rocholl & Mönch, 2019). The textile industry, with its complex and highly variable production processes, greatly benefits from optimization approaches in scheduling. These techniques enable manufacturers to efficiently allocate resources, minimize delays, and reduce operational costs. By implementing advanced scheduling algorithms, textile enterprises can enhance their agility and responsiveness to market demands, ensuring timely deliveries and improving overall productivity (Dermitas, 2022). Moreover, optimized scheduling fosters better utilization of machinery and workforce, which is essential in maintaining a competitive edge in a rapidly evolving industry. Thus, the application of optimization methods in scheduling is not only crucial for operational excellence but also for sustaining long-term growth and profitability (Noor et al., 2022). Table 1 compares key contributions in production scheduling research, with emphasis on real-world textile manufacturing applications.

Table 1
Overview of recent scheduling studies in the textile industry

Work	Approach	Textile Manufacturing Problem
Celikbilek et al. (2016)	Genetic Algorithm	Bottleneck machine scheduling
Zaharie et al. (2017)	Exact Approach	Order acceptance, delivery date setting and scheduling
Zhou et al. (2017)	Genetic Algorithm	Water-saving scheduling
Zhang et al. (2017)	Particle Swarm Optimization and Local Search	Production planning with environmental considerations
Ortíz-Barrios et al. (2018)	Hybrid Dispatching Algorithm	Operation selection in the flexible job-shop scheduling
Huynh et al. (2018)	Genetic Algorithm	Batch dyeing scheduling
Lorente-Leyva et al. (2019)	Genetic Algorithm	Master production scheduling
Nugraheni et al. (2020)	Genetic Programming and Hyper-Heuristic	Adaptive flow and criteria in flexible flow shop
Tsao et al. (2020)	Simulated Annealing and Genetic Algorithm	Cut ordering planning
Berthier et al. (2022)	Exact Approach and Genetic Algorithm	Unrelated parallel machines
Demirtas (2022)	Local Search	Unrelated parallel dedicated machines scheduling
Prassad et al. (2022)	Exact Approach	Cut ordering planning
Tsao et al. (2022)	Particle Swarm Optimization	Marker planning problem
Wang et al. (2023)	Tabu Search	Batch processing machines

Scheduling is a critical component for optimizing production efficiency and meeting stringent market demands. In this context, Identical Parallel Machine Scheduling (IPMS) is one of the most common and impactful scheduling issues in the textile industry. Addressing this problem is essential for optimizing machinery usage, reducing operational bottlenecks, and enhancing overall production flow (Li & Liu, 2024; Min et al., 2024). The IPMS problem is a well-known optimization challenge in modern manufacturing. It involves scheduling a set of jobs (n) on identical machines (m) with the objective of optimizing certain performance metrics, such as minimizing total tardiness or makespan. This problem is particularly relevant in environments where multiple machines perform the same jobs, and efficient scheduling can significantly impact productivity and operational costs (Lu et al., 2024; Kim et al., 2020). Problems involving identical parallel machines are considered NP-hard, meaning that no algorithm is known to solve these problems in polynomial time, making the attainment of precise solutions more complex as the number of jobs increases. An example of a problem that falls into the NP-hard class is the scheduling on identical parallel machines with the objective of minimizing total tardiness, denoted by $P_m| \sum T_j$ (Zhang et al., 2024; Lee, 2018; Pinedo, 2016).

According to recent studies, the problem of minimizing total tardiness in identical parallel machines can be described as follows: Consider a set $N = \{1, 2, \dots, n\}$ consisting of n jobs that need to be scheduled on a set $M = \{1, 2, \dots, m\}$ of m identical parallel machines. Each job must be assigned to one machine, and the goal is to sequence these jobs in a way that minimizes the total tardiness across all jobs, as illustrated in Fig. 1.

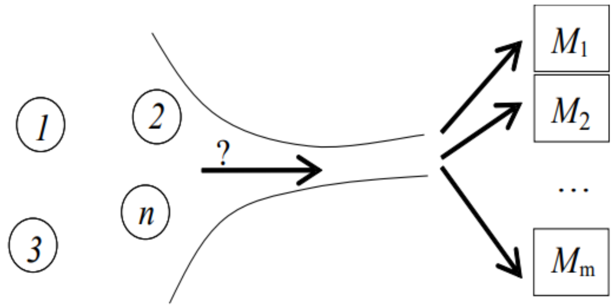

Fig. 1. Decision-making in parallel machines scheduling problems

In IPMS, each job (i) has a processing time (p_i) and a due date (d_i). Furthermore, it is assumed that all machines are identical in terms of their capabilities and processing speeds. Any job can be processed on any machine without any preference or difference in processing time.

In its classical form, it is also assumed that all jobs are ready for execution at time zero, and that the processing of a job can begin immediately after the completion of the previous job. The tardiness $T_i(S)$ of a job i in a schedule is determined by $T_i = max\{0, c_i - d_i\}$, where c_i represents the time at which the processing of job i is completed. Thus, the objective function representing the problem is formalized as minimizing the total tardiness, which is determined by the sum of the tardiness of each job (Feng & Peng, 2024). The textile industry presents unique challenges in scheduling due to the necessity of frequent machine setups between fabric types, a factor that has motivated the development of customized heuristic approaches of different classes (Demirtas, 2022).

Evolutionary and population-based heuristics have been widely applied to scheduling problems in textile environments due to their ability to explore large solution spaces effectively. Genetic algorithms (GAs) are among the most popular methods in this category (Zhou et al., 2017; Celikbilek et al., 2016). Particle swarm optimization (PSO) has also been employed for problems in this field of application. A recent study by Tsao et al. (2022) successfully applied PSO to the marker planning problem. The study highlighted PSO's ability to converge quickly to high-quality solutions.

Local search and tabu-based heuristics are known for their ability to intensively explore the neighborhood of solutions, making them suitable for fine-tuning schedules. In a study by Dermitas (2022), a local search (LS) algorithm was applied to minimize the total tardiness in an unrelated parallel dedicated machine scheduling problem. The results showed LS's ability to escape local optima. Tabu search (TS) is another powerful heuristic in this category (Wang et al., 2023).

Hybrid heuristics enhance solution quality and computational efficiency by combining the strengths of different heuristic methods. A hybrid simulated annealing and genetic algorithm (SA-GA) approach was proposed by Tsao et al. (2020) for cut ordering planning problems. The hybrid method leveraged the global search capability of GAs and the local search strength of SA, resulting in superior performance compared to standalone heuristics. Another hybrid approach combining particle swarm optimization and local search (PSO-LS) was investigated by Zhang et al. (2017), applied to production planning problems with environmental considerations.

The IPMS problem has been extensively studied in recent years, with a notable increase in interest, particularly in the context of minimizing the total tardiness in job scheduling. In this context, several studies have been conducted on parallel machine scheduling problems considering different constraints, such as sequence-dependent setup times (Goli & Keshavarz, 2022), release dates (Elidrisse et al., 2024; Li & Chen, 2023), batch orders (Beldar et al., 2022; Shahvari et al., 2022), precedence relationships (Caselli et al., 2022), heterogeneous machines (Bastos & Rosendo, 2020; Pan et al., 2020; Ekici et al., 2019), among others.

3. Problem description

The studied company operates in the textile industry, is based in Brazil, and maintains multinational operations. Its manufacturing processes focus on towel weaving, carried out across two distinct production lines, which will be referred to in this study as lines A and B. Fig. 2 illustrates the distribution of the production lines and their respective outputs.

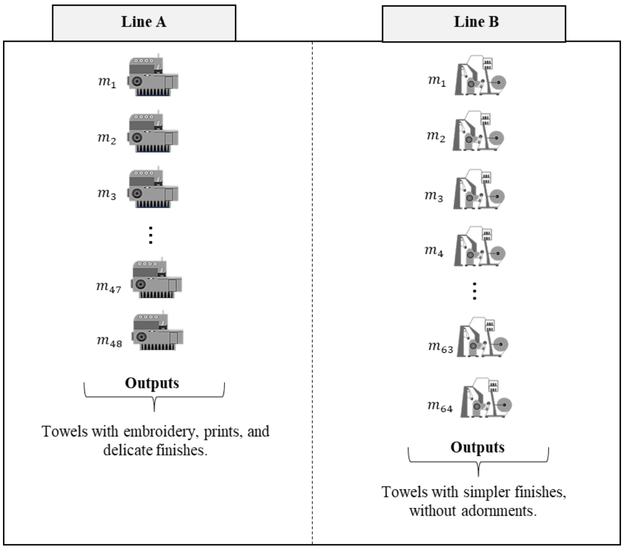

Fig. 2. Company production lines representation

Line A consists of 48 identical looms operating in parallel. Due to the advanced technology employed in these machines, the products processed on this line exhibit more refined and complex details, such as delicate embroidery, prints, and special finishes. Conversely, Line B operates with a total of 64 identical looms, also arranged in parallel. However, the products from this line have a simpler finish, as the machines are from an older generation of weaving technology.

In the IPMS addressed in this study, in addition to traditional constraints, there are also release date constraints and sequence-dependent setup times. To better understand these aspects, consider an instance with $n = 6$ jobs, where in addition to processing times (p_i) and due dates (d_i), there are also release dates (r_i) and the type of each job (k_i), as shown in Table 2.

Table 2
Example of an instance for the IPMS with additional constraints

i	p_i	r_i	k_i	d_i
1	4	3	2	10
2	3	5	1	10
3	5	0	1	9
4	3	5	3	13
5	1	0	3	5
6	2	3	2	7

Each of the orders shown in Table 2 is categorized as type 1, 2, or 3, with this distinction determining whether a setup time is required between orders. Additionally, each order has its own release date. This implies that, for instance, job $i = 2$ can only begin processing at time 5, even if it is the first in the sequence. Consequently, the process will naturally have an idle period that prevents it from starting at $t = 0$, as illustrated in Fig. 3.

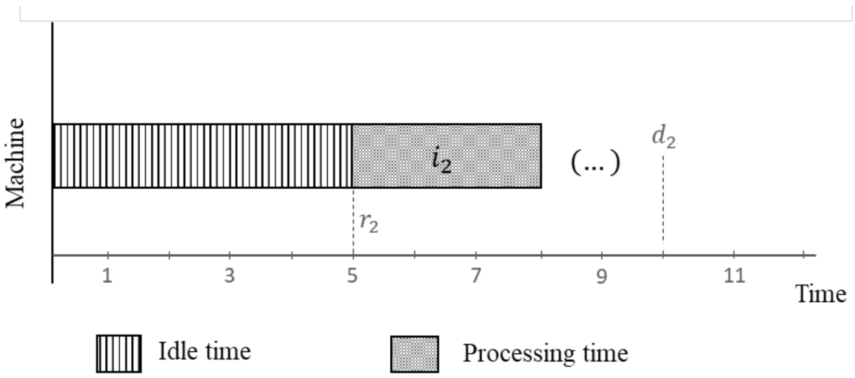

Fig. 3. Graphical representation of the release date in a solution

Additionally, it is considered that there are sequence-dependent setup times, which are intrinsically related to the machine configuration. This occurs due to the need to adjust operational elements of the machine according to the type of product, such as changing the thread/fabric for towel production, its quantity per cubic meter, among other adjustments. Thus, when job j is sequenced immediately after job i, and they belong to different order types, there is a machine setup time s_{ij}. Conversely, when there are no machine setup times, $s_{ij} = 0$, indicating that both jobs belong to the same order type, as shown in Table 3.

Table 3
Example of a sequence-dependent setup time matrix

Type (k_i)	1	2	3
1	0	1	1
2	1	0	1
3	1	1	0

Fig. 4 illustrates an arbitrary sequencing for the IPMS discussed in this study, based on the instance example presented in Table 2, with the respective setup times shown in Table 3.

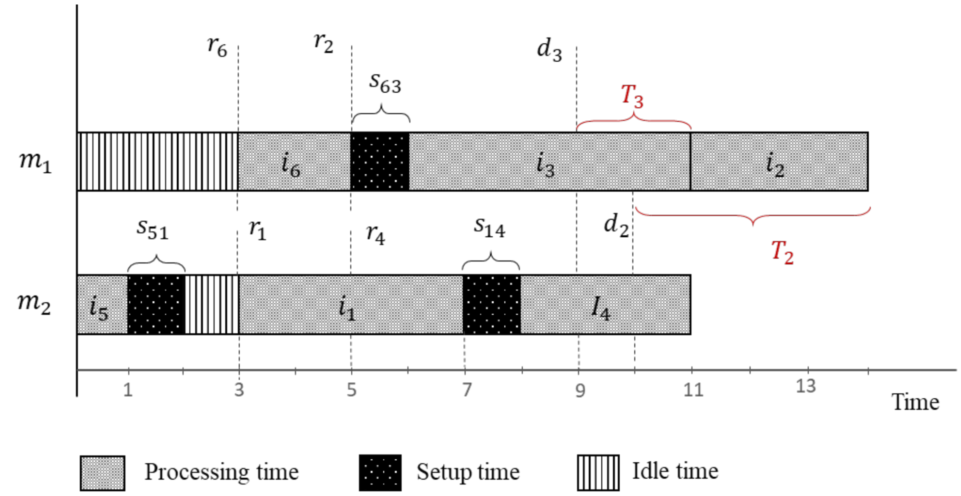

Fig. 4. Arbitrary schedule for the complete instance example

Fig. 4 shows that orders 2, 3, and 6 are allocated to machine 1, while orders 1, 4, and 5 are allocated to machine 2. In the adopted sequencing, machine 1 begins processing at time $t = 3$, due to order 6 being the first in the sequence with a release date of 3 units. Additionally, setup times are observed, as some consecutively sequenced orders belong to different types, such as orders 1 and 4, resulting in a setup time of $s_{14} = 1$. In the presented example, it is also noted that machine 1 has two delayed orders, orders 2 and 3. Jobs 2 and 3 were completed four and two units after their due dates, respectively. For this solution, the total tardiness is 6 units.

3.1. Mathematical Formulation

The proposed mathematical model for this problem was developed using Mixed Integer Linear Programming (MILP) and can be represented as $P_m|r_i, s_{ij}|\sum T_i$. In this modeling approach, we consider the number of jobs n to be sequenced and the number of machines m, with $n \geq m$. The decision variables, parameters used in the model, and the mathematical formulation of the problem are described as follows:

- p_i: processing time of job i, $(i = 1, \dots, n)$;
- d_i: due date of job i, $(i = 1, \dots, n)$;
- s_{ij}: setup time between jobs i and j, $i, j \in \{0, 1, \dots, n, n+1\}$;
- r_i: release date of job i, $(i = 1, \dots, n)$;
- c_i: completion time of job i, $(i = 1, \dots, n)$;
- x_{ij}: Binary variable. If job i is schedule immediately before j, $x_{ij} = 1$. Otherwise, $x_{ij} = 0$;
- T_i: Tardiness of job i $(i = 1, \dots, n)$.

$$\min Z = \sum_{i=1}^{n} T_i \tag{1}$$

Subject to:

$$\sum_{j=1}^{n} x_{0j} = m \tag{2}$$

$$\sum_{j=1}^{n+1} x_{ij} = 1, \qquad \forall i = 1, \dots, n \tag{3}$$

$$\sum_{i=0}^{n} x_{ij} = 1, \qquad \forall j = 1, \dots, n \tag{4}$$

$$c_0 = 0 \tag{5}$$

$$c_j \geq p_j + \sum_{i=0}^{n} s_{ij}(s_{ij} + c_i), \qquad \forall j = 1, \dots, n \tag{6}$$

$$c_j \geq r_j + p_j + \sum_{i=0}^{n} x_{ij} s_{ij}, \qquad \forall j = 1, \dots, n \tag{7}$$

$$T_i \geq c_i - d_i, \qquad \forall i = 1, \dots, n \tag{8}$$

$$T_i \geq 0, \qquad \forall i = 1, \dots, n \tag{9}$$

$$x_{ij} \in \{0,1\}, \qquad \forall i, j \in \{0, 1, \dots, n, n+1\} \tag{10}$$

$$x_{ij} = 0, \qquad i = j \tag{11}$$

$$c_j \geq 0, \qquad \forall j = 0, 1, \dots, n \tag{12}$$

Equation (1) defines the objective function, which aims to minimize the total tardiness of the sequenced jobs. Constraints (2) ensure that the sequencing occurs within a set of m machines. The sets of constraints (3) and (4) guarantee that each job is selected and sequenced on only one machine. Constraint (5) establishes the completion time of the fictitious job 0, allowing for the use of constraints (6) and (7), which establish the relationships between the completion times of jobs.

Constraints (6) and (7) ensure that jobs only begin processing after the completion of the previous job, in conjunction with the release date of the job in question and the completion of the required setup, depending on the specifics of the previous and current jobs. Constraints (8) and (9) define that the tardiness for each job must be equal to or greater than the difference between its completion time and its delivery date, as well as greater than zero. Finally, the sets of constraints (10) and (11) set the limits for the values of the decision variables x_{ij}, while constraints (12) indicate that the completion time of each job must be greater than or equal to zero.

4. Adaptive Local Search Approach

To solve the IPMS, the study proposed the development of a heuristic approach based on local search, named Adaptive Local Search (ALS). This approach includes three neighborhood search mechanisms: intra swap, inter swap, and insertion, along with a perturbation procedure. Figure 5 illustrates the proposed algorithm.

The process begins with the generation of an initial solution, prioritizing the sequencing of jobs with the nearest delivery dates and considering the probability of allocating jobs to machines with shorter processing times. The initial solution is evaluated based on the objective function of minimizing total tardiness. Subsequently, the heuristic enters the iterative local search loop, applying three neighborhood strategies to find new solutions. Each new solution is evaluated in the same manner as the initial one, and if an improvement is identified by the algorithm, it is stored as the best solution.

After the improvement check, the algorithm assesses whether the perturbation criterion has been met. If so, a perturbation technique is applied to the solution to avoid stagnation in local optima. Next, the stopping criterion, based on the maximum number of iterations, is evaluated. If the perturbation criterion is not satisfied, the algorithm proceeds directly to the evaluation of the stopping criterion. Upon meeting this criterion, the code terminates, and the best solution found is reported as the result. However, if the stopping criterion is not met, the code returns to the beginning of the iterative local search loop and repeats the entire procedure, continuously seeking to improve the solution.

The solution is represented by a two-dimensional matrix, where the quantities of machines and jobs vary according to the instance, which depends on the month and the production line. In each position (i, j) of the matrix, i represents the machine to which the job is allocated, and j indicates the sequential position of the job on that machine for its processing. An illustrative example of this representation can be seen in Fig. 6.

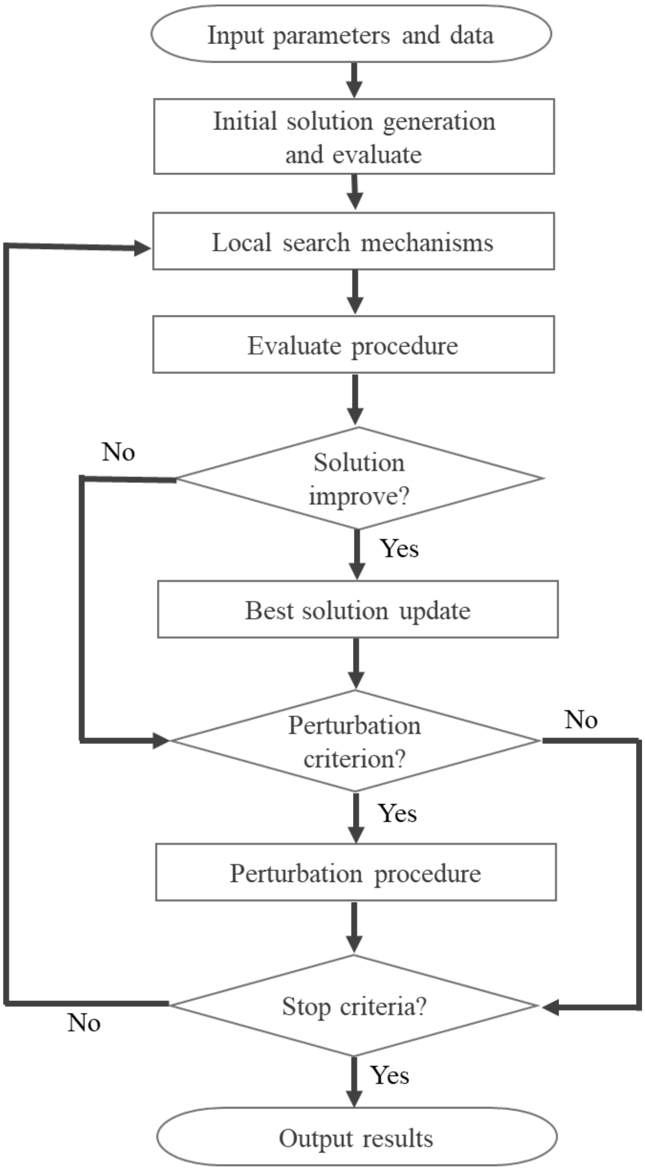

Fig. 5. Steps of the proposed heuristic approach

		1	**2**	**3**	**4**	
Machines	**1**	5	8	10	2	
	2	11	9	1	3	→ Job
	3	7	12	4	6	

Fig. 6. Representation of an Arbitrary Solution

In Fig. 6, the allocation of twelve orders across three machines is observed. For example, on machine 1, the orders to be processed are [5, 8, 10, 2], following the same logic for the other machines. The initial solution generation prioritizes orders with the closest delivery dates. Additionally, there is a 20% probability of allocating jobs to machines with the shortest processing times to diversify the initial solutions. The solution generation process is completed when there are no more orders to be allocated.

4.1. Solution evaluation

The objective function $f\ (s)$ aims to minimize total tardiness. To achieve this goal, not only the initial solution but all subsequent solutions throughout the iterations are evaluated. Algorithm 1 provides the procedure used to determine the objective function value of the current solution.

Algorithm 1 – Solution evaluation

1: **Procedure** Evaluation
2: **in:** s, C_i, d_i, r_i , $s_{(i-1,\ i)}$, P_i, T_i
3: **out:** $f\ (s)$
4: **for** i = 1, ..., Q_m
5:   **for** j = 1, ..., Q_t
6:     **if** $C_{i-1} \geq r_i$
7:       $C_i = C_{i-1} + s_{(i-1,\ i)} + P_i$
8:     **else**
9:       $C_i = r_i + s_{(i-1,\ i)} + P_i$
10:     **end if**
11:     **if** $C_i \leq d_i$
12:       $T_i = 0$
13:     **else**
14:       $T_i = C_i - d_i$
15:     **end if**
16:   **end for**
17: **end for**
18: Return $f\ (s)$
19: **end** Evaluation

To calculate the total tardiness, the algorithm must be executed repeatedly, encompassing the total number of machines in the instance, represented by Q_m, as well as the number of jobs per machine Q_t. When calculating the completion time, two scenarios are considered. The first scenario occurs when the completion time of the previous job is equal to or greater than the release date of the current job, as described in line 6. Thus, the subsequent job can commence processing after the setup time, if applicable. In this case, the completion time is given by the following equation:

$$C_i = C_{i-1} + s_{(i-1,\ i)} + P_i \tag{11}$$

Eq. (11) calculates the completion time of the job by considering three main components: the completion time of the previous job C_{i-1}, the setup time of the machine from the previous job to the current job $s_{(i-1,\ i)}$, and the processing time of the current order P_i. If the job is the first to be sequenced, the completion time will be equal to its own processing time.

However, there is a second scenario to consider, where the previous job (j_{i-1}) may be completed before the release date of the next job (j_i). In this case, the release date must be included in the calculation, and the equation becomes:

$$C_i = r_i + s_{(i-1,\ i)} + P_i \tag{12}$$

Once the completion time of each job is calculated, the individual tardiness for each job can be determined. This tardiness is given by the difference between the job's completion time and its due date, as shown in Eq. (13):

$$T_i = C_i - d_i \tag{13}$$

As noted in lines 11 and 12 of Algorithm 1, if a job is completed before its due date, there is no delay, thus $T_i = 0$. Finally, the total tardiness *f*(*s*) is obtained from Equation (14), which sums all these individual tardiness:

$$f\ (s) = \sum_{i=1}^{n} T_i \tag{14}$$

4.2. Local search procedure

Effective neighborhood operators are critical for local search performance in parallel machine scheduling. We adopt three fundamental neighborhood strategies to ensure comprehensive exploration of the solution space.

***Intra-machine swap*:** This operator selects two distinct job positions within a single machine and exchanges their assignments. Fig. 7 illustrates the procedure, where jobs 6 and 1 swap positions while preserving all other job assignments.

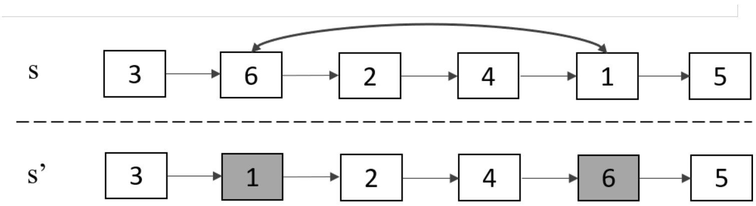

Fig. 7. Swap intra-machine procedure

***Inter-machine swap*:** The inter-machine swap operator generates neighboring solutions by selecting two distinct machines and exchanging one randomly chosen job position from each: given machines m_1 and m_2 with selected jobs $j_4 \in m_1$ and $j_{12} \in m_2$, the operator performs the assignment swap, as demonstrated in Fig. 8.

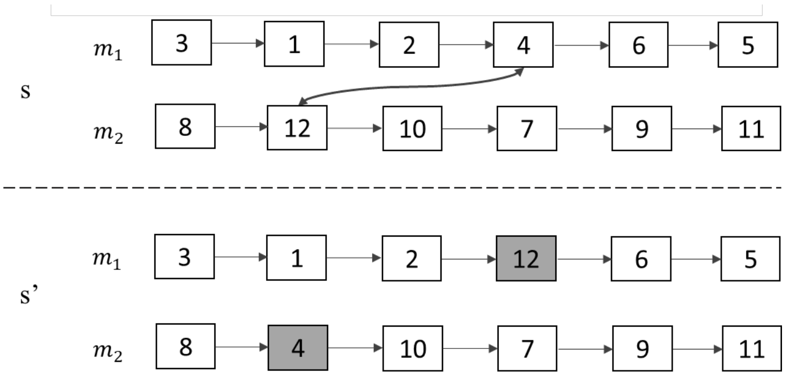

Fig. 8. Swap inter-machines procedure

***Insertion*:** The insertion operator generates new solutions by relocating a single job j from its current position p_1 in machine m_1 to a new position p_2 in machine m_2 (where $m_1 \neq m_2$). Fig. 9 illustrates the relocation of job 3 from machine 1 to a position preceding job 9 on machine 2.

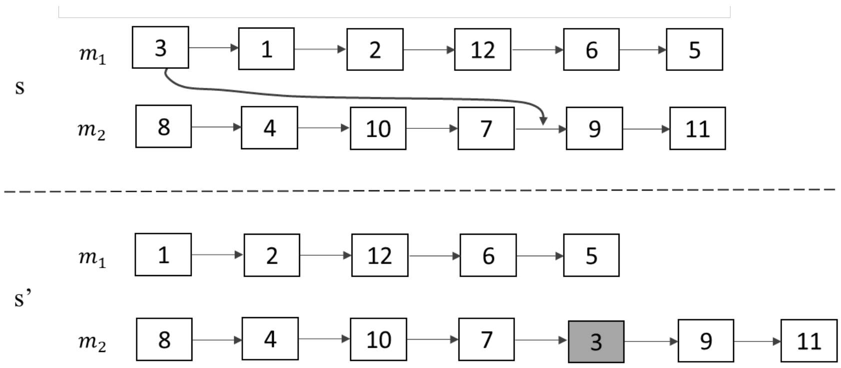

Fig. 9. Insertion procedure

Algorithm 2 describes the pseudocode for the three implemented local search neighborhoods. The variable s represents the best current solution, which will be used to find new neighborhoods, while s' indicates the solution obtained through local search strategies. The parameters C_i, s_{ij}, P_i represent, respectively, the completion times, setup times, and processing times for each job. The term d_i indicates the delivery date, and r_i the release date. All these variables are fundamental for evaluating the objective functions $f(s')$ and $f(s)$, which quantify the total tardiness.

The first strategy employed is the intra-machine swap, which performs a number of positional exchanges denoted by Q_{si}. The second strategy refers to the inter-machine swap, which also has a specific number of exchanges Q_{se}. Finally, the third strategy performs Q_{in} job insertions, generating a new solution s'. After applying the strategies, s' is evaluated considering the objective function. If $f(s') < f(s)$, then s' is adopted as the current best solution.

The perturbation procedure in the proposed LS is triggered when the criterion C_P is met. This criterion analyzes whether the total tardiness of the current function $f(s)$ is 20% greater than the total tardiness of the best current solution $f(s_{best})$, and if the number of recorded improvements c_m is greater than 1% of the maximum number of iterations defined in the local search. If these conditions are satisfied, the insertion strategy is applied to the best current solution s_{best}, generating a new solution s'.

Algorithm 2 – Local Search procedure

```
Procedure LocalSearch (swap intra, swap inter, insertion)
in: s, C_i, d_i, r_i, s_ij, P_i
out: s'
for Iter = 1, ..., MaxIter
  for i = 1, ..., Q_si
    if v > Prob
      Select two different jobs
      Perform intra swap between the selected jobs
    end if
  end for
  for i = 1, ..., Q_se
    if v > Prob
       Select two jobs from different machines
       Perform inter swap between the jobs
    end if
  end for
  for i = 1, ..., Q_in
    if v > Prob
      Select two positions (P_1, P_2) on different machines (M_1, M_2)
      Insert the job associated with P_1 of M_1 into position P_2 of M_2
    end if
  end for
  Evaluate the objective function for s'
   if f(s') < f(s)
         s ← s'
   end if
  if perturbation criterion is met
    Perform the perturbation procedure
  end if
end for
Return s'
end LocalSearch
```

5. Experimental results

5.1. Parameters calibration

To improve the efficiency of the results, heuristic parameters were adjusted, with a particular emphasis on local search strategies. Table 4 presents the calibrated parameters along with their corresponding values.

Table 4
Local search parameters

Parameters	Values
Probability of initializing local search strategies	Rnd ( ) > 0.20
	Rnd ( ) > 0.35
	Rnd ( ) > 0.50
# swap intra	Q_m
# swap inter	$Q_m/12$
	$Q_m/4$
# insertion	$Q_m/12$
	$Q_m/4$
Criterion for returning to the best solution	$f(s) > 1.20 \cdot f(s_{best})$
	$f(s) > 1.35 \cdot f(s_{best})$
	$f(s) > 1.50 \cdot f(s_{best})$

As presented in Table 5, the following parameters were calibrated: the execution probability of each strategy, the number of intra- and inter-swaps, the insertion quantity, and the improvement threshold for restarting the local search with the current best solution. The parameter values were determined based on relevant literature and the experiments conducted in this study. To achieve proper heuristic calibration, multiple parameter combinations were tested, assessing average tardiness, average execution time, and the standard deviation of tardiness. Each parameter combination was tested across 10 independent runs, with local search termination triggered at 2,000 iterations.

The best performance was achieved with the parameter combination that include a probability of 0.5 for executing the three neighborhood strategies, an intra-swap count equal to the number of machines in the instance Q_m, an inter-swap and insertion count set to $Q_m/12$, and a restart condition for returning to the best solution defined as $f(s) > 1.20 \cdot f(s_{best})$. This parameter configuration resulted in an average tardiness of 149.63 days with a standard deviation of 7.874.

5.2. Instance structuring

During the study, eight instances were collected and structured directly with the company, with data gathered through the organization's ERP and subsequently processed. During the structuring phase, any doubts regarding data interpretation were clarified with the organization's managers to ensure that the structured instances faithfully represented the observed reality within the organization.

The collected data pertain to the months of September, October, November, and December 2024, resulting in the creation of four distinct instances for each production line. Each instance is determined by its production line, which defines the number of machines, and by the month, which determines the number of jobs. The first four instances correspond to production line A with 48 looms (machines), while the next four belong to line B with 64 looms, as detailed in Table 5.

Table 5
Structured instances for the problem

Instance	# Machines	# Jobs
A1	48	240
A2	48	183
A3	48	264
A4	48	297
B1	64	319
B2	64	372
B3	64	344
B4	64	415

5.3. Computational results

The experiments were conducted on a computer equipped with an Intel Core i5 processor clocked at 2.3 GHz, 8 GB of RAM, and running Windows 10. For the computational experiments, 10 runs were performed for each of the eight instances, totaling 80 executions. Additionally, a maximum limit of 5,000 iterations was established for the developed local search approach. The results obtained for the instances are reported in Table 6, including the minimum, average, and standard deviation values.

Table 6
Summary of achieved experimental results

Instance	Total tardiness			Time		
	Min.	Avg.	SD	Min.	Avg.	SD.
A1	430.2	446.2	11.68	29.77	30.32	0.29
A2	329.3	352.8	14.92	26.46	26.69	0.13
A3	310.1	340.5	13.45	33.21	33.74	0.27
A4	311.8	328.4	9.45	36.12	36.30	0.13
B1	436.7	454.3	9.57	54.95	59.14	2.38
B2	477.2	503.3	12.49	55.33	58.64	2.21
B3	368.3	399.6	17.69	55.22	57.34	1.61
B4	207.3	222.8	7.49	70.29	71.55	0.55

Despite being the largest instance, B4 achieved the lowest total tardiness (minimum of 207.3 days and an average of 222.8 days). This scenario is intrinsically linked to the magnitude of the order information related to this instance, particularly regarding the processing time data, which are lower compared to other instances. Being the largest instance in the study, it also had the highest execution times, with an average of 71.55 seconds and a standard deviation of 0.55.

Analyzing the results for production line A, instance A3 stands out with the smallest total tardiness of 310.1 days. However, considering the average total tardiness, instance A4 achieved the lowest average, corresponding to 328.4 days. The highest standard deviations concerning total tardiness were found in instances A2 and B3, with 14.92 and 17.69, respectively. The smallest deviations were observed in instances B4 and A4, for the month of September, with 7.49 and 9.45, respectively.

Regarding computational times, the heuristic obtained the highest times when executed with information from production line B instances. This phenomenon occurs due to the larger number of machines and jobs in these instances, which naturally demands a longer period for the algorithm execution and solution analysis. Concerning the deviations, they were smaller for line A, demonstrating greater consistency, as the data have low variability around the mean.

To compare the developed heuristic with the company's current method, an algorithm was created that exactly simulates the decision-making process adopted by the organization during the study. The algorithm sequences jobs based on the closest delivery date, order, and allocation to machines with the shortest accumulated processing times. The results of the simulation experiments are shown in Table 7 and were validated by the organization as reflecting the reality observed during the periods of the collected data.

Table 7
Results of the simulation of the company's adopted Method

Instance	$\sum T_i$ (days)
A1	585.7
A2	468.5
A3	587.0
A4	426.8
B1	522.1
B2	685.2
B3	634.2
B4	315.9

Table 8 presents a comparison between the results obtained by the proposed heuristic approach and the results achieved by the organization on the same data instances, highlighting the superior performance achieved by the proposed approach compared to the strategy currently adopted by the organization.

Table 8
Comparison of results obtained by the proposed heuristic and the company's approach

Instance	ALS $\sum T_i$ (days)			Company Approach $\sum T_i$ (days)	Percentage Improvement (%)		
	Min.	Avg.	Max.	-	Min.	Avg.	Max.
A1	430.2	446.2	465.4	585.7	26.5	23.8	20.5
A2	329.3	352.8	371.0	468.5	29.7	24.7	20.8
A3	310.1	340.5	353.9	587.0	47.2	42.0	39.7
A4	311.8	328.4	342.2	426.8	26.9	23.0	19.8
B1	436.7	454.3	464.8	522.1	16.4	13.0	11.0
B2	477.2	503.3	515.6	685.2	30.4	26.5	24.8
B3	368.3	399.6	422.9	634.2	41.9	37.0	33.3
B4	207.3	222.8	232.7	315.9	34.4	29.5	26.3
Avg.	358.9	381.0	396.0	528.2	31.7	27.4	24.5

The LS heuristic achieved significant reductions, particularly in total tardiness for instances A3 and B3, improving by 47.2% and 41.9% respectively compared to the company's method. Across all instances, the average total tardiness improvement is 27.4%. The best solutions from the heuristic show an average improvement of 31.7%, while even the worst scenarios see a 24.5% enhancement. Fig. 10 demonstrates the total tardiness achieved using the LS heuristic in the best scenarios, compared with the company's method.

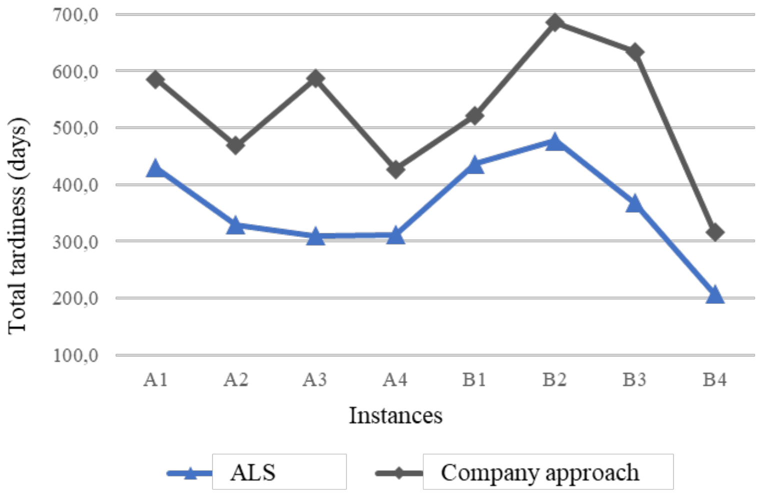

Fig. 10. Comparison of the best solutions obtained by ALS and the company's approach

In all instances, the ALS heuristic demonstrated superior results, consistently finding solutions with lower total tardiness compared to the company's method. The difference between the approaches is most pronounced in instances A3, B2, and B3. Fig. 11 provides a visual representation of the maximum total tardiness obtained by the ALS heuristic in contrast to the company's method.

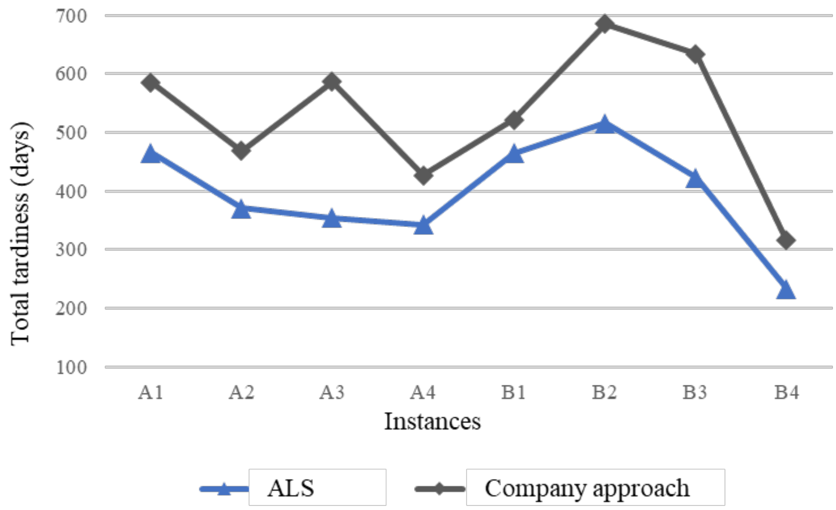

Fig. 11. Comparison of the worst solutions obtained by ALS and the company's approach

Although Fig. 11 presents the performance of solutions with the highest total tardiness obtained by the heuristic, it still demonstrates superiority in terms of quality. The discrepancy between the solutions remains significant in instances A3, B2, and B3. On the other hand, instance B1 recorded a result close to that obtained by the company's method, with a difference of 57.3 days.

6. Conclusions

Due to the complexity of solving production scheduling problems, they have become a significant area of interest among researchers and, notably, in industries that continuously seek to improve their processes and satisfactorily meet customer demands. Consequently, the need for effective and agile resolution methods is increasingly pertinent. Heuristics have been widely utilized, as they excel in finding viable and high-quality solutions despite not guaranteeing the best solution in the search space.

The objective of this study was to develop a heuristic approach, based on local search techniques, to optimize the job scheduling problem on identical parallel machines, applicable to the operational scenario of a textile company. The approach focused on minimizing total tardiness, considering the complexity of variables such as release dates and setup times. The proposed algorithm ALS incorporated three local search strategies: intra-machine swap, inter-machine swap, and insertion, along with a perturbation mechanism to avoid stagnation in local optima and expand the search space.

The proposed heuristic was applied to eight real instances provided by the company under study. The results demonstrated that the Local Search (LS) heuristic outperformed the company’s method in all eight instances analyzed, presenting an average total tardiness improvement of 27.4%. Considering the best-obtained solutions, the average improvement was 31.7%.

The proposed heuristic proves to be an excellent alternative for optimizing production scheduling for similar problems, especially in industrial environments that prioritize timely deliveries and seek more agile scheduling solutions. As encountered limitations and possible future research proposals, we consider the implementation of efficient dynamic approaches to allow production scheduling for the addressed problem according to the demand receipt times.

References

Albayrak, E., & Onuet, S. (2024). Energy-efficient scheduling for a flexible job shop problem considering rework processes and new job arrival. *International Journal of Industrial Engineering Computations*, *15*(4), 871-886.

Bastos, C. E. N., & Resendo, L. C. (2020). Two-step approach for scheduling jobs to non-related parallel machines with sequence dependent setup times applying job splitting. *Computers & Industrial Engineering, 145*, 106500.

Beldar, P., Moghtader, M., Giret, A., & Ansaripoor, A. H. (2022). Non-identical parallel machines batch processing problem with release dates, due dates and variable maintenance activity to minimize total tardiness. *Computers & Industrial Engineering, 168*, 108135

Berthier, A., Yalaoui, A., Chehade, H., Yalaoui, F., Amodeo, L., & Bouillot, C. (2022). Unrelated parallel machines scheduling with dependent setup times in textile industry. *Computers & Industrial Engineering, 174*, 108736.

Caselli, G., Delorme, M., Iori, M., & Magni, C. A. (2022). Mixed integer linear programming for a real-world parallel machine scheduling problem with workforce and precedence constraints. In *Optimization in Artificial Intelligence and Data Sciences: ODS, First Hybrid Conference, Rome, Italy* (pp. 61-71). Springer International Publishing.

Celikbilek, C., Tunacan, T., Süer, G. A., & Dulkadir, O. (2016). Scheduling comparison between multi-objective mathematical models and genetic algorithms approach in the textile industry. *International Journal of Services and Operations Management, 25*(2), 236-258.

Demirtas, Y. E. (2022). Unrelated parallel dedicated machine scheduling with sequence dependent setup times: an application in a textile company. *International Journal of Applied Decision Sciences, 15*(6), 733-752.

Ekici, A., Elyasi, M., Özener, O. Ö., & Sarıkaya, M. B. (2019). An application of unrelated parallel machine scheduling with sequence-dependent setups at Vestel Electronics. *Computers & Operations Research, 111*, 130-140.

Elidrissi, A., Benmansour, R., Zufferey, N., Benbrahim, M., & Duvivier, D. (2024). Minimization of maximum lateness on parallel machines with a single server and job release dates. *4OR, 22*, 351-385.

Feng, X., & Peng, H. (2024). Robust identical parallel machine scheduling with two-stage time-of-use tariff and not-all-machine option. *International Journal of Production Research, 62*(1), 380-403.

Fuchigami, H. Y., & Rangel, S. (2018). A survey of case studies in production scheduling: Analysis and perspectives. *Journal of Computational Science, 25*, 425-436.

Goli, A., & Keshavarz, T. (2022). Just-in-time scheduling in identical parallel machine sequence-dependent group scheduling problem. *Journal of Industrial and Management Optimization, 18*(6), 3807-3830.

Hu, C., Zheng, R., Lu, S., & Liu, X. (2024). Parallel machine scheduling with position-dependent processing times and deteriorating maintenance activities. *Journal of Global Optimization.*

Huynh, N. T., & Chien, C. F. (2018). A hybrid multi-subpopulation genetic algorithm for textile batch dyeing scheduling and an empirical study. *Computers & Industrial Engineering, 125*, 615-627.

Kim, J. G., Song, S., & Jeong, B. (2020). Minimizing total tardiness for the identical parallel machine scheduling problem with splitting jobs and sequence-dependent setup times. *International Journal of Production Research, 58*(6), 1628-1643.

Kim, J., & Kim, H. J. (2021). Parallel machine scheduling with multiple processing alternatives and sequence-dependent setup times. *International Journal of Production Research, 59*(18), 5438-5453.

Krimi, I., & Benmansour, R. (2024). Self-adaptive General Variable Neighborhood Search algorithm for parallel machine scheduling with unrelated servers. *Computers & Operations Research, 163*, 106480.

Lee, C. H. (2018). A dispatching rule and a random iterated greedy metaheuristic for identical parallel machine scheduling to minimize total tardiness. *International Journal of Production Research, 56*(6), 2292-2308.

Lee, J. H., & Kim, H. J. (2021). A heuristic algorithm for identical parallel machine scheduling: splitting jobs, sequence-dependent setup times, and limited setup operators. *Flexible Services and Manufacturing Journal, 33*, 992-1026.

Li, K., Xie, F., Zhao, X., Chen, J., & Zhou, T. (2024). Parallel machine scheduling with a total energy consumption limitation for minimizing total completion time. *Engineering Optimization*, 1-28.

Li, S. S., & Chen, R. X. (2023). Competitive two-agent scheduling with release dates and preemption on a single machine. *Journal of Scheduling, 26*(3), 227-249.

Li, X., & Liu, C. (2024). Energy-efficient scheduling in an identical parallel machine environment with peak power consumption and deadline constraints. *Computers & Operations Research, 170*, 106777.

Liu, X., Chu, F., Zheng, F., Chu, C., & Liu, M. (2021). Parallel machine scheduling with stochastic release times and processing times. *International Journal of Production Research, 59*(20), 6327-6346.

Lorente-Leyva, L. L., et al. (2019). Optimization of the Master Production Scheduling in a Textile Industry Using Genetic Algorithm. In *Hybrid Artificial Intelligent Systems. HAIS 2019. Lecture Notes in Computer Science* (Vol. 11734). Springer.

Lu, H., Shi, Y., & Pei, Z. (2024). A distributionally robust approach for the parallel machine scheduling problem with optional machines and job tardiness. *Computers & Operations Research, 170*, 106776.

Meng, L., Cheng, W., Zhang, B., Zou, W., & Duan, P. (2024). A novel hybrid algorithm of genetic algorithm, variable neighborhood search and constraint programming for distributed flexible job shop scheduling problem. *International Journal of Industrial Engineering Computations*, *15*, 813-832.

Min, S. H., Lee, S. W., & Kim, H. J. (2024). Parallel Machine Scheduling With Peak Energy Consumption Limits. *IEEE Transactions on Automation Science and Engineering*, 1-13.

Noor, A., Saeed, M. A., Ullah, T., Uddin, Z., & Ullah Khan, R. M. W. (2022). A review of artificial intelligence applications in apparel industry. *The Journal of The Textile Institute, 113*, 505-514.

Nugraheni, C. E., Abednego, L., & Widyarini, M. (2020). A Genetic Programming based Hyper-Heuristic for Production Scheduling in Apparel Industry. In *International Conference on Machine Learning Techniques and NLP.*

Ortíz-Barrios Ortíz-Barrios, M., Neira-Rodado, D., Jiménez-Delgado, G., & Hernández-Palma, H. (2018). Using FAHP-VIKOR for operation selection in the flexible job-shop scheduling problem: A case study in textile industry. In *Advances in Swarm Intelligence: 9th International Conference, ICSI 2018, Shanghai, China*. Springer International Publishing.

Pan, Z., Lei, D., & Wang, L. (2020). A knowledge-based two-population optimization algorithm for distributed energy-efficient parallel machines scheduling. *IEEE Transactions on Cybernetics, 52*(6), 5051-5063.

Pei, J., Song, Q., Liao, B., Liu, X., & Pardalos, P. (2021). Parallel-machine serial-batching scheduling with release times under the effects of position-dependent learning and time-dependent deterioration. *Annals of Operations Research, 298*, 407-444.

Pinedo, M. L. (2016). *Scheduling: Theory, algorithms, and systems* (5th ed.). Springer.

Prasad, S., Panghal, M., & Ali, T. M. (2022). Developing a cost-effective and heuristic tool to solve cut order planning problems in the apparel industry. *International Journal of Mathematics in Operational Research, 21*(1), 26-45.

Rocholl, J., & Mönch, L. (2019). Decomposition heuristics for parallel machine multiple orders per job scheduling problems with a common due date. *Journal of the Operational Research Society, 72*(8), 1737-1753.

Safarzadeh, H., & Niaki, S. (2023). Unrelated parallel machine scheduling with machine processing cost. *International Journal of Industrial Engineering Computations*, *14*(1), 33-48.

Shahvari, O., Logendran, R., & Tavana, M. (2022). An efficient model-based branch-and-price algorithm for unrelated-parallel machine batching and scheduling problems. *Journal of Scheduling, 25*(5), 589-621.

Tsao, Y. C., Delicia, M., & Vu, T. L. (2022). Marker planning problem in the apparel industry: Hybrid PSO-based heuristics. *Applied Soft Computing, 123*, 108928.

Tsao, Y. C., Vu, T. L., & Liao, L. W. (2020). Hybrid heuristics for the cut ordering planning problem in apparel industry. *Computers & Industrial Engineering, 144*, 106478.

Wang, S., & Zhang, H. (2023). A matheuristic for flowshop scheduling with batch processing machines in textile manufacturing. *Applied Soft Computing, 145*, 110594.

Wu, C. C., Chen, J. H., Lin, W. C., Zhang, X., Ren, T., Wu, Z. L., & Chung, Y. H. (2025). A robust single-machine scheduling problem with scenario-dependent processing times and release dates. *International Journal of Industrial Engineering Computations*, *16*(1), 37-50.

Wu, X., & Che, A. (2019). A memetic differential evolution algorithm for energy-efficient parallel machine scheduling. *Omega, 82*, 155-165.

Xue, H., Meng, L., Duan, P., Zhang, B., Zou, W., & Sang, H. (2024). Modeling and optimization of the hybrid flow shop scheduling problem with sequence-dependent setup times. *International Journal of Industrial Engineering Computations*, *15*(2), 473-490.

Zaharie, B., Islak, G., Dehelean, C., & Miclea, L. (2017). A hierarchical approach of order acceptance and delivery date setting problems in the apparel industry. In *18th International Carpathian Control Conference (ICCC), Sinaia, Romania* (pp. 267-272).

Zhang, R., Chang, P. C., Song, S., & Wu, C. (2017). Local search enhanced multi-objective PSO algorithm for scheduling textile production processes with environmental considerations. *Applied Soft Computing, 61*, 447-467.

Zhang, X., Angel, E., Chu, F., & Regnault, D. (2024). Minimizing total completion time and makespan for a multi-scenario bi-criteria parallel machine scheduling problem. *European Journal of Operational Research.*

Zhou, L., Xu, K., Cheng, X., Xu, Y., & Jia, Q. (2017). Study on optimizing production scheduling for water-saving in textile dyeing industry. *Journal of Cleaner Production, 141*, 721-727.