International Journal of Industrial Engineering Computations 17 (2026) 709–720

Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

# An effective iterated greedy heuristic for the flow shop scheduling with heterogeneous workers

**Fernando Luis Rossi[a], Esra Boz[b] and Marcelo Seido Nagano[c*]**

[a]*Federal Institute of São Paulo, Department of Industrial Engineering, São Paulo, Brazil*
[b]*KTO Karatay University, Department of Industrial Engineering, Konya, Türkiye*
[c]*University of São Paulo, São Carlos School of Engineering, Department of Production Engineering, São Paulo, Brazil*

C H R O N I C L E

*Article history*:
Received October 30 2025
Received in Revised Format January 3 2026
Accepted February 5 2026
Available online February 5 2026

Keywords:
*Flow shop*
*Heterogeneous workers*
*Iterated greedy*
*Scheduling*
*Metaheuristics*

A B S T R A C T

This paper addresses the Permutation Flow Shop Scheduling Problem with Heterogeneous Workers (PFSP-HW), an extension of the classical problem in which processing times depend not only on the job and machine, but also on the assigned worker. This variant better reflects practical environments where worker capabilities and proficiencies vary significantly. We propose a new Iterated Greedy (IG) heuristic adapted to handle worker heterogeneity. The IG heuristic combines destruction and reconstruction mechanisms with a local search procedure tailored for the problem. We develop two versions of the proposed algorithm and compare them with adapted state-of-the-art heuristics and metaheuristics from related problems. The algorithms were tested on a large benchmark set comprising 360 instances generated under various shop configurations. The suggested IG heuristics surpass current approaches in terms of solution quality and execution time, as determined by computational and statistical evaluations, making them reliable and efficient tools for solving the PFSP-HW.



## 1. Introduction

The Permutation Flow Shop Scheduling Problem (PFSP) has been widely studied in the scheduling literature due to its theoretical significance and numerous industrial applications. In its traditional form, a group of tasks must be completed on a group of machines in the same order, to optimize a performance measure such as makespan ($C_{\max}$), total flowtime ($\sum C_j$), or total tardiness ($\sum T_j$) (Ruiz & Maroto, 2005; Framinan et al., 2003). The problem is NP-hard for more than two machines (Garey & Johnson, 1977), and numerous heuristics and metaheuristics have been developed to provide high-quality solutions within reasonable computational times (Nawaz et al., 1983; Fernandez-Viagas et al., 2014; Rossi & Nagano, 2021).

In practical production environments, the assumption of homogeneous workers is often unrealistic. Operators frequently differ in skills, experience, and productivity, which leads to heterogeneous processing times even for the same job–machine combination. This gives rise to the PFSP-HW, an extension of the classical problem in which processing times depend on the job, the machine, and the assigned worker (Ruiz & Stützle, 2008; Moreira et al., 2012). The PFSP-HW requires simultaneous determination of job sequences, worker assignments, and machine schedules, which considerably enlarges the search space and increases the problem's complexity.

The PFSP-HW has practical relevance in industries such as electronics assembly, furniture manufacturing, or sheltered work centers, where workers with different abilities perform tasks in parallel (Corominas et al., 2008). Despite its importance, the literature on PFSP-HW is still limited compared to the classical PFSP, and few effective algorithms have been specifically designed for this variant (Benavides et al., 2014; Fernandez-Viagas & Framinan, 2022).

Among the metaheuristics applied to difficult scheduling issues, the IG method has earned recognition for its simplicity and effectiveness. First suggested for the permutation flow shop issue (Ruiz & Stützle 2007), IG has been successfully adapted to variants such as blocking flow shops (Pan et al., 2014), no-idle flow shops (Tasgetiren et al., 2013), and sequence-dependent setup time problems (Fernandez-Viagas & Ruiz, 2017), often outperforming more sophisticated approaches in both quality and computational time. However, to the best of our knowledge, IG has not yet been adapted to address the additional decision layer of heterogeneous worker assignment in flow shop scheduling.

* Corresponding author
E-mail drnagano@usp.br (M. S. Nagano)
ISSN 1923-2934 (Online) - ISSN 1923-2926 (Print)
2026 Growing Science Ltd.
doi: 10.5267/j.ijiec.2026.2.001

In real-world production environments, workers vary in speed, skill, and experience, and thus processing times are not only job- and machine-dependent but also worker-dependent. However, the overwhelming majority of classical flow shop scheduling literature assumes homogeneous workers, overlooking this important source of variability. There is a need to capture this type of heterogeneity to achieve realistic and human-centered scheduling solutions that better reflect actual production dynamics and improve shop-floor performance. In the meantime, the Flow Shop Scheduling Problem with Heterogeneous Workers (FSP-HW) introduces strong interdependence between worker–machine assignment and job sequencing, whereby classical heuristics are no longer sufficient. Existing research has failed to provide an efficient and high-quality algorithm that can effectively solve this composite optimization problem. The motivation behind this research is therefore the need for an efficient, simple metaheuristic solution. By developing a specially tailored the IG algorithm to handle worker heterogeneity, the research aims to offer a practical and effective tool that bridges the gap between modeling realism and computational tractability.

This paper proposes a new IG heuristic specifically designed for the PFSP-HW. The method incorporates problem-specific destruction and reconstruction phases, a local search tailored to the joint job–machine–worker assignment structure, and an acceleration procedure for efficiently evaluating candidate solutions. The algorithm is evaluated on a benchmark of 360 instances generated under different shop configurations and worker heterogeneity levels. Computational and statistical analyses reveal that the suggested technique outperforms adapted state-of-the-art methods in terms of solution quality and runtime.

In this described setting, this study seeks answers to the following research questions:

*Research Question 1*: How does the structure and performance of the classical flow shop scheduling problem change with the addition of worker heterogeneity?

*Research Question 2*: Can an IG heuristic effectively co-optimize job sequencing and worker–machine assignment in the PFSP-HW?

*Research Question 3*: How does the implemented IG algorithm perform in terms of solution quality, stability, and efficiency in processing time compared to existing metaheuristics and heuristics?

*Research Question 4*: How responsive is the proposed approach to variations in workers' heterogeneity levels and shop configurations, and can it maintain scalability for big instances?

This is how the rest of the paper is structured: Section 2 reviews the literature on flow shop scheduling with heterogeneous resources and IG-based approaches. Section 3 formally defines the PFSP-HW. Section 4 details the proposed IG heuristic. The results and computational experiments are shown in Section 5. Finally, Section 6 ends the work and suggests options for future research.

## 2. Literature review

Research on flow shop scheduling has evolved along two intertwined tracks: (i) the classical Permutation Flow Shop Problem (PFSP) with homogeneous processing times and (ii) richer settings that embed more realistic shop-floor features such as worker heterogeneity, machine unavailability, setup effects, or no-idle/no-wait constraints.

The PFSP-HW is based on the PFSP and has been developed in detail by incorporating the employee assignment process into this problem. Therefore, a literature review on the problem addressed in this study was first conducted on the PFSP, followed by a field study on the PFSP -HW, and consequently, gaps in the literature were identified. In this context, studies addressing the PFSP topic in the literature were first reviewed. Due to the abundance of studies, we focused on recent studies on the PFSP. A summary of these studies is provided in Table 1. (For detailed information, see: Tosun et al. (2020), Komaki et al. (2019)). Existing studies have utilized the classic flow shop scheduling problem under actual production conditions, generating more advanced models that consider human, time, energy, and resource differences. Bai et al., (2022) and Mansouri (2023) sought to minimize production flow delays by incorporating time requirements, including waiting time, blocking, and latency. Irmouli et al. (2023) optimized operator selection through deep reinforcement learning, providing contributions to AI-based dynamic planning systems. Such studies aim to offer more flexibility and adaptability in scheduling by considering variables such as heterogeneous human resources, imprecise task durations, and production activity time windows. Moreover, distributed manufacturing and sustainability studies are also highlighted in the literature on the flow shop. Gogos (2023), Nicosia et al. (2024), Singla et al. (2024), and Becker et al. (2025) maximized the allocation of resources by incorporating logistics factors like multi-site scheduling, flexible allocation of tasks, cost of renting machines, and transportation delays into their models. Park et al. (2025), Liu et al. (2025), and Su et al. (2025) proposed multi-objective approaches considering energy usage, speed fluctuations of machines, and continuous production environments. These studies usually position the flow shop scheduling problem within the realm of sustainable and integrated production systems via the incorporation of parameters such as energy efficiency, operating flexibility, and process continuity.

**Table 1**
Related studies in PFSP

| Reference | Objective Function | Solution Method | Handled Problem |
|---|---|---|---|
| Bai et al. (2022) | Minimize makespan, maximum lateness | Heuristic algorithm | A flow shop problem that simultaneously models release date and blocking constraints. |
| Irmouli et al. (2023) | Minimize makespan | Genetic algorithm with deep reinforcement learning | A permutation flow shop model whose operator selection is optimized using deep reinforcement learning. |
| Mansouri et al. (2023) | Minimize makespan under waiting-time | Particle swarm optimization algorithm | A time-windowed flow shop model that includes waiting time and job release time constraints. |
| Gogos (2023) | Minimize makespan | Constrain programming | A distributed permutation flow shop scheduling in a multi-plant environment. |
| Nicosia et al. (2024) | Minimize makespan with blocking and flexible assignments | Matheuristic | A flow shop model that combines flexible task assignment and blocking constraints. |
| Singla et al. (2024) | Minimize makespan and machine rental cost | Novel heuristic | A two-stage model that includes machine rental costs under a no-idle constraint. |
| Becker et al. (2025) | Minimize makespan including transport delay | IG algorithm | A distributed flow shop model that considers intermediate product transport between plants. |
| Park et al. (2025) | Minimize total weighted completion time | Approximation/ensemble algorithm | A machine-dependent flow shop problem that includes machine-dependent speed differences. |
| Liu et al. (2025) | Minimize makespan+processing delay ratio+electricity cost | NSGA-II, GRASP | A multi-objective flexible flow shop problem that includes energy consumption and production delays. |
| Su et al. (2025) | Minimize makespan | IG algorithm | A PPFSP model that considers a continuous processing environment in a proposed model for process manufacturing such as a steel mill. |

Modeling human variability explicitly is comparatively recent in flow shops. The Assembly Line Worker Assignment and Balancing Problem (ALWABP) showed that heterogeneity in execution times and incompatibilities can, and should, be handled natively in optimization models and heuristics (Miralles et al., 2007; Corominas et al., 2008; Blum & Miralles, 2011; Moreira et al., 2012). For flow shops specifically, Benavides et al. (2014) introduced the Heterogeneous Flow Shop Scheduling Problem (Het-PFSP), where the processing time $p_{i,k,h}$ of operation $o_{ij}$ depends on the worker assigned to machine $M_k$. The solution couples two intertwined decisions: (1) a permutation (or non-permutation) schedule of jobs and (2) a one-to-one assignment of workers to machines. This coupling increases the search space by a factor of $m!$ relative to the homogeneous case and renders even small instances challenging for exact MILPs. Their scatter search with path relinking combined positional recombination for worker permutations with critical-path neighborhoods for job sequences, setting a baseline for the problem.

**Table 2**
Related studies in PFSP, including human factors

| Reference | Solution Method | Handled Problem |
|---|---|---|
| Cheng and Kovalyov (2003) | Dynamic programming | A single server with two machines PFSP |
| Benkalai et al. (2019) | Heuristic algorithm | PFSP with operators |
| Samarghandi (2015) | Genetic algorithm | No wait PFSP with sequence-dependent set-up times |
| Chu et al. (2018) | Genetic algorithm + hill climbing algorithm | Reentrant Flow Shop Scheduling with Multi resource Qualification Matching |
| Fekri et al. (2024) | Genetic algorithm + simulated annealing algorithm | Multi-skilled Resource-Constrained Flexible Flow Shop Scheduling Problem |
| Li et al. (2024) | Grey wolf optimization algorithm | |
| Gong et al. (2020) | Evolutionary algorithm | Energy-efficient FFSP with worker flexibility |
| Fernandez-Viagas et al. (2022) | NEH, IG | A PPFSP with human resource constraints and setup times. |
| Liu et al. (2023) | Combining reinforcement learning and genetic algorithms for simulation-based optimization | PFSP with multi-skilled workers and fatigue factors |
| Benavides et al. (2014) | Scatter search algorithm | PFSP with heterogeneous workers |
| This study | IG algorithm | PFSP with heterogeneous workers |

More detailed studies in the literature, specific to the study, which include human factors, are also examined and listed in Table 2. Table 2 provides an overview of the development of approaches integrating human factors in flow shop scheduling problems over the years. In the previous studies, Cheng and Kovalyov (2003) solved a two-machine, single-server PFSP using dynamic programming, while Benkalai et al. (2019) developed a heuristic algorithm based on the influence of operators. Samarghandi (2015) and Chu et al. (2018) optimized decision variables such as human-caused setup times and quality compatibility in no-wait or reentrant production systems by using a genetic algorithm and a hybrid genetic algorithm and hill climbing, respectively. Subsequent studies integrated human factors into the model in a more comprehensive manner, considering issues such as multi-skilled labor and energy efficiency. Fekri et al. (2024) and Li et al. (2024) solved multi-skilled, resource-constrained flexible PFSPs using hybrid metaheuristics (GA+SA) and the gray wolf optimization algorithm, respectively. Gong et al. (2020) considered both worker flexibility and energy efficiency, while Liu et al. (2023) modeled multi-skilled workers and fatigue factors using the combination of genetic algorithms and reinforcement learning. Fernandez-

Viagas et al. (2022) considered human resources and setup times with the NEH and the IG methods, while Benavides et al. (2014) was the first study that introduced the concept of "flow shop scheduling with heterogeneous workers" to the literature. In this study, the same problem structure was handled with the IG algorithm.

## 3. Research Gaps

Despite the increasing interest in human-centric concerns in flow shop scheduling, the PFSP-HW remains methodologically underexplored. Except for the seminal work by Benavides et al. (2014), who introduced worker heterogeneity via a scatter search approach, the literature still lacks dedicated heuristic frameworks that effectively integrate worker–machine assignment decisions and sequencing optimization. Existing work typically treats worker assignment and job scheduling as independent subproblems or applies general-purpose algorithms that overlook the strong interdependence between these components. Moreover, no IG-based method has been specifically designed to (i) co-optimize worker–machine assignment and job sequence within a unified destruction–construction framework, (ii) employ critical-path–aware local search neighborhoods with worker-dependent processing times, and (iii) employ efficient delta evaluation mechanisms to ensure scalability. The methodological gap is therefore the absence of a lightweight yet high-performance IG framework to efficiently combat the combinatorial complexity of Het-PFSP.

## 4. Contributions of the Study

This study fills the mentioned methodological gap by proposing a novel IG heuristic specifically tailored to the PFSP-HW. The proposed approach provides first-class treatment to worker permutations as decision variables, allowing worker–machine assignments and job sequences to be optimized simultaneously within a single iterative loop. The process includes critical-path–guided insertion and destruction moves in order for local improvements to be sensitive to heterogeneity-induced processing time variability. Additionally, a calibrated acceptance mechanism is incorporated to help the search process escape assignment-induced local optima, fostering exploration without compromising computational efficiency. Comprehensive computational testing over an extended set of benchmark instances shows that the proposed IG algorithm surpasses tailored state-of-the-art heuristics and metaheuristics in solution quality and execution time, uniformly over all instances, confirming its robustness and scalability in large-scale heterogeneous flow shop environments.

## 4. Problem Description

In the classical the PFSP, the processing time of job j on machine m is fixed and independent of the operator. But in the heterogeneous worker extension, the processing time depends on the job, the machine, and the assigned worker. According to the three-field notation of Graham et al. (1979), the problem can be denoted as:

$$F_m \mid \text{prmu, heterogeneous workers} \mid C_{max}$$

Let $J = \{J_1, J_2, \dots, J_n\}$ be a set of $n$ jobs to be processed on a set of $m$ machines $M = \{M_1, M_2, \dots, M_m\}$ in the same processing order. Each machine m is operated by exactly one worker selected from a set $W = \{W_1, W_2, \dots, W_k\}$, where $|W| = k$. The notation of developed mathematical model is given in Table 3.

**Table 3**
Notations

| | |
|---|---|
| **Indices and Sets** | |
| $J$ | Jobs |
| $M$ | Machines |
| $W$ | Workers |
| $\pi$ | Sequence of $n$ jobs |
| **Decision Variables** | |
| $x_{jr}$ | 1, if job j is assigned to position r; 0 otherwise |
| $z_{mv}$ | 1, if worker w is assigned to machine m; 0 otherwise |
| $y_{jrmw}$ | 1, job j at position r is processed on machine m by worker w; 0 otherwise |
| $p_{rm}$ | Processing time of the job assigned to position r on machine m |
| $C_{rm}$ | Completion time of the job assigned to position r on machine m |
| $C_{max}$ | The makespan of the schedule |
| **Parameters** | |
| $p_{jmw}$ | The processing time of job j on machine m when operated by worker w |

$$\min \mathrm{C}_{max} \tag{1}$$

$$\sum_{r \in R} x_{jr} = 1 \qquad \forall j \in J \tag{2}$$

$$\sum_{j \in J} x_{jr} = 1 \qquad \forall r \in R \tag{3}$$

$$\sum_{w \in W} z_{mv} = 1 \qquad \forall m \in M \qquad (4)$$
$$y_{jrmw} = x_{jr} * z_{mv} \qquad \forall j, r, m, w \qquad (5)$$
$$\sum_{w \in W} y_{jrmw} = x_{jr} \qquad \forall j, r, m \qquad (6)$$
$$p_{rm} = \sum_{j \in J} \sum_{w \in W} p_{jmw} y_{jrmw} \qquad \forall r \in R, m \in M \qquad (7)$$
$$C_{r1} \geq C_{(r-1)1} + p_{r1} \qquad \forall r \in R \qquad (8)$$
$$C_{rm} \geq C_{r(m-1)} + p_{rm} \qquad \forall r \in R, m = 2, .., m \qquad (9)$$
$$C_{rm} \geq C_{(r-1)m} + p_{rm} \qquad \forall r \in R, m = 2, .., m \qquad (10)$$
$$C_{max} \geq C_{rm} \qquad \forall r \in R, m \in M \qquad (11)$$
$$x_{jr} \in \{0,1\} \qquad (12)$$
$$z_{mw} \in \{0,1\} \qquad (13)$$
$$y_{jrmw} \in \{0,1\} \qquad (14)$$
$$p_{rm} \geq 0 \qquad (15)$$
$$C_{rm} \geq 0 \qquad (16)$$
$$C_{max} \geq 0 \qquad (17)$$

The goal of function (1) is to reduce the makespan, which is the highest number of task completion times in the schedule. Each job is assigned to a single position thanks to Eq. (2). Eq. (3) guarantees that each position will have a single task. Eq. (4) represents that one worker may be assigned to each machine. Eq. (5) represents whether the job and worker are being assigned simultaneously. Eq. (6) ensures job-worker association. Eq. (7) decides the process times. Eqs. (8-10) decide the completion times on the machines. Eq. (11) ensures that the makespan variable must not be less than the maximum completion time of all machines and jobs. Eqs. (12-17) are sign constraints.

***Linearization***: Because the Constraint (6) is not linear, the mathematical model is not linear, it is a nonlinear formulation. In this case, it is necessary to linearize this constraint with given constraints as follows:

$$y_{jrmw} \leq x_{jr} \qquad \forall \mathrm{j, r, m, w} \qquad (18)$$
$$y_{jrmw} \leq z_{mv} \qquad \forall \mathrm{j, r, m, w} \qquad (19)$$
$$y_{jrmw} \geq \mathrm{x}_{jr} + z_{mv} - 1 \qquad \forall \mathrm{j, r, m, w} \qquad (20)$$

## 5. Novel Iterated Greedy Heuristic

In this study, we propose an IG algorithm specifically tailored for the PFSP-HW, referred to as IG-HW. The method comprises three main components: an initialization phase, a destruction–reconstruction phase, and a final refinement stage that employs a new local search procedure based on worker reassignment.

A variation of the NEH heuristic is used to generate the initial solution, denoted as FRB4, a NEH variant proposed by Rad et al. (2009). The FRB4 is based on reinsertion strategy and tie-breaking rules to improve NEH's performance. Like NEH, FRB4 builds the job sequence incrementally but applies a smarter reinsertion phase to intensify the local search around each partial solution. Among these, FRB4 emerged as one of the most effective and is still widely used as a baseline constructive heuristic for makespan minimization in the flow shop literature. We adapted the FRB4 to consider worker-dependent processing times by generating a randomized worker assignment. With this initial solution, we employed the Referenced Local Search (RLS) approach. The RLS method, introduced by Ruiz and Stützle (2007), refines the current solution through insertion-based moves guided by a reference solution. This solution serves as the starting point for the iterative deconstruction-reconstruction process.

At each iteration, a destruction phase is performed in which, randomly, a portion of d jobs is eliminated from the current order. The construction phase then reinserts these jobs one at a time into the position that yields the best increase in solution quality. A referenced local search procedure is subsequently applied, exploring the neighborhood with insertion procedures, while also adjusting worker assignments to exploit heterogeneity in processing capabilities. The resulting solution is evaluated using a makespan calculation that explicitly accounts for worker–machine–job dependencies. This solution is accepted if it outperforms the best that has been discovered thus far; if not, it may nevertheless be accepted with a certain probability to enable diversification and escape from local optima. In this instance, a simulated annealing acceptance criterion with a constant temperature was applied.

After the job-sequence improvement step (RLS), we apply a dedicated local search on the worker–machine assignment while keeping the job sequence fixed. We denoted this new method Worker Insertion Local Search (WILS). The worker assignment is represented as a permutation $z$ of size $m$, where $z_i$denotes the worker assigned to machine $i$. The proposed neighborhood is an insertion move in the worker permutation: a worker currently assigned to machine $p$ is removed and inserted into another machine position $q$, shifting the remaining workers accordingly. Each candidate assignment is checked for feasibility with respect to worker–machine compatibility, and the makespan is evaluated considering worker-dependent processing times. The procedure iterates until no improving insertion move is found. This local improvement is applied once after initialization and once per IG iteration. This method is presented in Algorithm 1. This iterative procedure keeps going until a halting condition, like a predetermined number of iterations or a computing time limit, is met. The algorithm outputs the best sequence and worker assignment found, along with its makespan, which is then used for comparative performance analysis against other heuristics such as Scatter Search (Benavides et al., 2014). Algorithm 2 contains the algorithm's pseudocode.

**Algorithm 1 – Worker Insertion Local Search (WILS)**

$improved \leftarrow true$
**while** $improved = true$ **do**
    $improved \leftarrow false$
    **for** $p = 1$ **to** $m$ **do** // origin machine position
        **for** $q = 1$ **to** $m$ **do** // target machine position
            **if** $q = p$ **then** continue
            $z' \leftarrow InsertWorker(z, p, q)$     // remove worker at $p$, insert at $q$
            **if** $z'$is infeasible (compatibility) **then** continue
            **if** $C_{\max}(\pi, z') < C_{\max}(\pi, z)$ **then**
                $z \leftarrow z'$
                $improved \leftarrow true$
            **end if**
        **end for**
    **end for**
**end while**
**return** $z$

**Algorithm 2 – Iterated Greedy for Heterogeneous Workers (IG–HW)**

**Generate** a feasible worker assignment $z$
$\pi \leftarrow FRB4(z)$
$\pi \leftarrow RLS(\pi, z)$
$z \leftarrow WILS(\pi, z)$    (Algorithm 1)
$(\pi^{*}, z^{*}) \leftarrow (\pi, z)$
**while CPU < CPU**$_{max}$ **do**
    $\pi' \leftarrow \pi$
    **Remove** $d$ jobs **from** $\pi'$producing list $D = \{D_1, \dots, D_d\}$
    **for** $i = 1$ **to** $d$ **do**
        **Insert** job $D_i$ in the best position of $\pi'$ (minimum $C_{\max}$)
    **end for**
    $\pi' \leftarrow RLS(\pi', z)$
    $z' \leftarrow WILS(\pi', z)$     (Algorithm 1)
    $S' \leftarrow (\pi', z')$
    **if** $C_{\max}(S') < C_{\max}(\pi, z)$ **then**
        $(\pi, z) \leftarrow S'$
        **if** $C_{\max}(\pi, z) < C_{\max}(\pi^{*}, z^{*})$ **then**
            $(\pi^{*}, z^{*}) \leftarrow (\pi, z)$
        **end if**
    **else if** $rand \leq \exp\left(-(C_{\max}(S') - C_{\max}(\pi, z))/T\right)$ **then**
        $(\pi, z) \leftarrow S'$
    **end if**
**end while**
**return** $(\pi^{*}, z^{*})$

## 6. Computational Results

The computational evaluation was carried out using the well-known Taillard benchmark set (Taillard 1993), which consists of 360 PFSP instances with sizes ranging from $20 \times 5$ to $500 \times 20$ jobs and machines. In this study, these instances were adapted to the flow shop scheduling problem with heterogeneous workers by introducing worker-dependent processing times for each job-machine pair. The adapted benchmark was organized into three groups of 120 instances each, corresponding to different levels of heterogeneity in worker performance. Each group preserves the original structure of the Taillard instances while modifying the processing time matrix according to heterogeneous worker configurations, ensuring a consistent and challenging test bed for the proposed algorithms.

The performance measure employed was the Relative Percentage Deviation (RPD) calculated according to Eq. (21):

$$RPD(C_{max}(\pi_h)) = 100 \cdot (C_{max}(\pi_h) - C^*_{max})/C^*_{max} \tag{21}$$

The makespan supplied by the sequence $\pi_h$ produced by heuristic h is the value of $C_{max}(\pi_h)$. $C^*_{max}$ is the optimal answer among all the compared heuristics. ARPD stands for Average Relative Percentage Deviation. The heuristic performs better when the RPD value is smaller since its answers will be closer to the optimal outcome among all the approaches compared.

All the compared heuristics were implemented in C++, compiled with Intel C++ 16.0, and ran on an Intel i9 9900 processor operating at 4.7 GHz with 16 GB of RAM. The following techniques were chosen for comparison:

- NEH: well-known constructive heuristic from Nawaz et al. (1983).
- Scatter Search (Benavides et al., 2014).
- IG: original Iterated Greedy adapted from Ruiz and Stützle (2007).
- IG-HW: proposed the IG algorithm in this study for the heterogeneous worker problem (Section 4).

The NEH constructive heuristic (Nawaz et al., 1983), the original IG (Ruiz & Stützle, 2007), and the Scatter Search metaheuristic (Benavides et al., 2014) were used to examine the performance of the suggested Iterated Greedy Heuristic with Heterogeneous Worker Local Search (IG-HW). The ARPD was used to evaluate the modified Taillard benchmark.

The computational results are summarized in Tables 4–6 using the ARPD as a performance metric. Four methods are compared: the NEH constructive heuristic, the original IG, the Scatter Search approach, and the proposed IG-HW. The destruction size parameter $d$ was fixed to $d = 10$, following the standard configuration used in the original IG algorithm proposed by Ruiz and Stützle for the flow shop scheduling problem. The inclusion of the original IG allows a clearer assessment of the impact of explicitly incorporating worker–machine assignment decisions within the IG framework.

**Table 4**
ARPD values for the compared heuristic.

| Set | n | NEH | IG | Scatter Search | IG-HW |
|---|---|---|---|---|---|
| 1 | 20 | 20.03 | 16.22 | **0.37** | 1.28 |
| 1 | 50 | 10.83 | 6.36 | 4.73 | **0.16** |
| 1 | 100 | 6.59 | 5.33 | 3.07 | **0.23** |
| 1 | 200 | 5.81 | 3.06 | 4.32 | **0.72** |
| 1 | 500 | 1.59 | 1.56 | 1.33 | **0.14** |
| **Average** | | 10.46 | 7.62 | 2.87 | **0.55** |
| 2 | 20 | 20.76 | 14.71 | 0.40 | **1.57** |
| 2 | 50 | 9.46 | 5.68 | 4.83 | **0.13** |
| 2 | 100 | 6.56 | 5.26 | 7.42 | **0.09** |
| 2 | 200 | 3.28 | 1.39 | 10.14 | **0.39** |
| 2 | 500 | 2.16 | 1.10 | 1.22 | **0.60** |
| **Average** | | 9.92 | 6.74 | 4.95 | **0.56** |
| 3 | 20 | 18.26 | 14.88 | **0.14** | 1.60 |
| 3 | 50 | 9.65 | 5.27 | 5.15 | **0.18** |
| 3 | 100 | 6.37 | 3.94 | 5.50 | **0.51** |
| 3 | 200 | 3.77 | 1.58 | 10.77 | **0.11** |
| 3 | 500 | 2.01 | 0.39 | 0.92 | **0.43** |
| **Average** | | 9.36 | 6.32 | 4.57 | **0.63** |
| **ARPD** | | 9.91 | 6.89 | 4.13 | **0.58** |

Considering all 360 instances, the IG-HW clearly achieves the best overall performance, presenting the lowest global ARPD value of 0.58. This result represents a significant improvement over Scatter Search, which attains an overall ARPD of 4.13, the original IG with 6.89, and NEH with 9.91. These results indicate that the proposed IG-HW substantially outperforms both classical constructive heuristics and metaheuristics that do not explicitly address worker heterogeneity.

**Table 5**
ARPD data for the comparative heuristic is summarized by the number of machines.

| Set | m | NEH | IG | Scatter Search | IG-HW |
|---|---|---|---|---|---|
| 1 | 5 | 10.27 | 9.41 | 0.82 | **0.64** |
| 1 | 10 | 11.11 | 8.10 | 5.37 | **0.73** |
| 1 | 20 | 10.06 | 6.16 | 2.11 | **0.35** |
| **Average** | | 10.46 | 7.62 | 2.87 | **0.55** |
| 2 | 5 | 9.62 | 7.53 | 1.00 | **0.95** |
| 2 | 10 | 11.17 | 7.46 | 4.56 | **0.58** |
| 2 | 20 | 9.09 | 5.68 | 7.64 | **0.32** |
| **Average** | | 9.92 | 6.74 | 4.95 | **0.56** |
| 3 | 5 | 8.18 | 7.77 | **0.45** | 0.65 |
| 3 | 10 | 10.81 | 6.77 | 5.39 | **0.76** |
| 3 | 20 | 8.91 | 5.09 | 6.38 | **0.50** |
| **Average** | | 9.36 | 6.32 | 4.57 | **0.63** |
| **ARPD** | | 9.91 | 6.89 | 4.13 | **0.58** |

Fig. 1 reports the average ARPD values together with 95% confidence intervals for the compared methods. The confidence interval associated with IG-HW does not overlap with those of NEH, IG, and Scatter Search, indicating that the mean ARPD achieved by IG-HW is statistically different at the 95% confidence level. Moreover, IG-HW presents the smallest confidence interval, suggesting greater stability and lower variability across the tested instances. These results provide additional statistical evidence of the superiority of the proposed method.

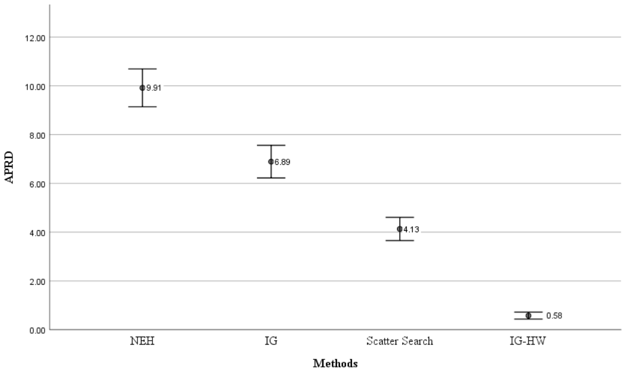


**Fig. 1.** Average ARPD across methods

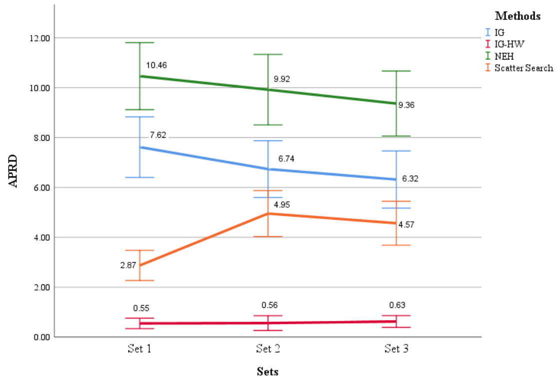


**Fig. 2.** ARPD by instance set

When the results are analyzed by heterogeneity set, IG-HW consistently yields the lowest average ARPD values across all groups. In Set 1, IG-HW attains an average ARPD of 0.55, while Scatter Search, IG, and NEH present values of 2.87, 7.62, and 10.46, respectively. Similar behavior is observed in Set 2, where IG-HW achieves an average ARPD of 0.56 compared to 4.95 for Scatter Search, 6.74 for IG, and 9.92 for NEH. In Set 3, IG-HW maintains its superiority with an average ARPD of 0.63, again outperforming Scatter Search, IG, and NEH. These findings show that the suggested approach is resilient to varying degrees of worker heterogeneity. Fig. 2 illustrates the average ARPD values with 95% confidence intervals for the four methods across the three heterogeneity sets. The proposed IG-HW consistently achieves the lowest ARPD values in all sets, with narrow confidence intervals, indicating both superior performance and high stability. In contrast, NEH and the original IG exhibit higher ARPD levels and larger variability, while Scatter Search shows noticeable fluctuations across the sets. The results indicate that IG-HW is robust to different levels of worker heterogeneity and maintains a statistically superior performance across all instance groups. The effect of the number of machines is further analyzed by aggregating ARPD values for different machine sizes. For all heterogeneity sets, IG-HW consistently achieves the lowest ARPD values for small, medium, and large numbers of machines. The advantage of IG-HW becomes more pronounced as the number of machines increases, indicating that the worker-oriented local search scales well with the machine dimension.

A detailed analysis by job size shows that the superiority of IG-HW is particularly evident for medium and large instances. For instances with $n \geq 100$ jobs, IG-HW consistently achieves ARPD values below 1.0 across all heterogeneity sets. In contrast, Scatter Search and NEH often exhibit ARPD values above 1.0, while the original IG, although improving upon NEH, remains significantly less competitive. These results suggest that the proposed algorithm effectively exploits worker heterogeneity to maintain high solution quality as problem size increases. The comparison between the original IG and IG-HW highlights the importance of explicitly integrating worker–machine assignment decisions into the search process. While the original IG improves substantially over NEH, its performance remains far from that of IG-HW. This performance gap confirms that a straightforward adaptation of IG without dedicated worker reassignment mechanisms is insufficient for solving the Flow Shop Scheduling Problem with Heterogeneous Workers. A key factor in attaining the noted gains is the worker insertion local search included into IG-HW. Overall, the computational experiments confirm that IG-HW consistently outperforms NEH, the original IG, and Scatter Search across all tested scenarios. The low and stable ARPD values achieved by IG-HW demonstrate its robustness, scalability, and effectiveness for addressing the Flow Shop Scheduling Problem with Heterogeneous Workers. In general, the graphs given in Fig. 3a and Fig. 3b show a summary of the overall situation. Fig. 3a shows a comparison graph of ARPD with other solution methods for each set. Accordingly, it is seen that the IG-HW algorithm achieved the best results by a significant margin compared to tests performed on the same data. Furthermore, Fig. 3b analyzes the results of all methods according to the number n. Accordingly, the IG-HW method yielded better results than the other methods.

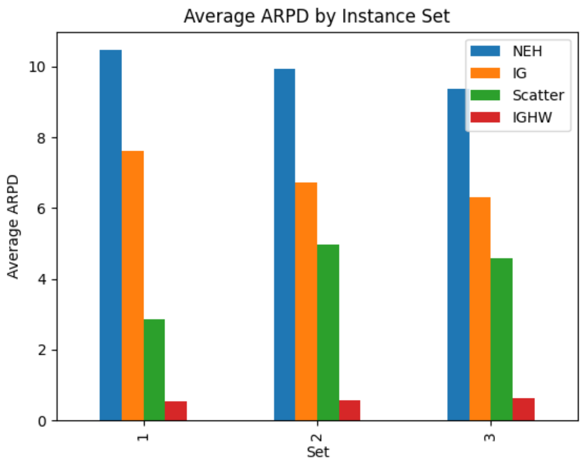


**Fig. 3a.** ARPD by instance set

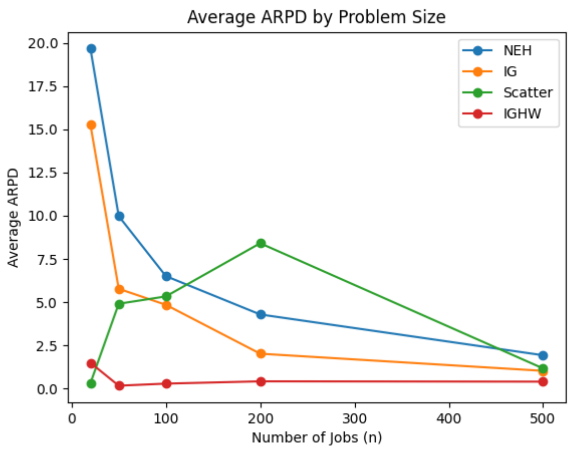


**Fig. 3b.** ARPD by the problem size

**Table 6**
Detailed ARPD values for the compared heuristic, summarized by number of jobs, machines, and data set.

| Set | n | m | NEH | IG | Scatter Search | IG-HW |
|---|---|---|---|---|---|---|
| 1 | 20 | 5 | 16.46 | 13.49 | **0.26** | 1.07 |
| 1 | 20 | 10 | 21.15 | 18.29 | **0.00** | 2.44 |
| 1 | 20 | 20 | 22.48 | 16.88 | 0.85 | **0.31** |
| 1 | 50 | 5 | 8.79 | 6.00 | 1.24 | **0.40** |
| 1 | 50 | 10 | 10.77 | 6.21 | 5.85 | **0.07** |
| 1 | 50 | 20 | 12.92 | 6.87 | 7.09 | **0.00** |
| 1 | 100 | 5 | 5.57 | 8.73 | 0.94 | **0.44** |
| 1 | 100 | 10 | 6.94 | 4.27 | 7.05 | **0.23** |
| 1 | 100 | 20 | 7.25 | 2.98 | 1.21 | **0.03** |
| 1 | 200 | 10 | 5.56 | 3.62 | 8.56 | **0.20** |
| 1 | 200 | 20 | 6.06 | 2.50 | **0.07** | 1.25 |
| 1 | 500 | 20 | 1.59 | 1.56 | 1.33 | **0.14** |
| 2 | 20 | 5 | 16.80 | 9.97 | **0.29** | 2.19 |
| 2 | 20 | 10 | 24.03 | 16.40 | **0.19** | 1.79 |
| 2 | 20 | 20 | 21.44 | 17.77 | 0.74 | **0.74** |
| 2 | 50 | 5 | 7.06 | 6.04 | 1.00 | **0.39** |
| 2 | 50 | 10 | 10.50 | 6.13 | 4.00 | **0.00** |
| 2 | 50 | 20 | 10.81 | 4.86 | 9.49 | **0.00** |
| 2 | 100 | 5 | 5.01 | 6.59 | 1.70 | **0.26** |
| 2 | 100 | 10 | 6.80 | 5.38 | 6.97 | **0.00** |
| 2 | 100 | 20 | 7.86 | 3.82 | 13.58 | **0.00** |
| 2 | 200 | 10 | 3.35 | 1.94 | 7.11 | **0.52** |
| 2 | 200 | 20 | 3.20 | 0.83 | 13.17 | **0.27** |
| 2 | 500 | 20 | 2.16 | 1.10 | 1.22 | **0.60** |
| 3 | 20 | 5 | 11.38 | 12.99 | 0.26 | **0.12** |
| 3 | 20 | 10 | 21.27 | 14.93 | **0.02** | 2.63 |
| 3 | 20 | 20 | 22.12 | 16.73 | **0.12** | 2.06 |
| 3 | 50 | 5 | 7.15 | 4.56 | 0.82 | **0.54** |
| 3 | 50 | 10 | 12.18 | 6.26 | 7.79 | **0.00** |
| 3 | 50 | 20 | 9.62 | 4.99 | 6.85 | **0.00** |
| 3 | 100 | 5 | 6.02 | 5.76 | **0.28** | 1.31 |
| 3 | 100 | 10 | 5.99 | 4.05 | 5.73 | **0.21** |
| 3 | 100 | 20 | 7.09 | 2.00 | 10.49 | **0.00** |
| 3 | 200 | 10 | 3.82 | 1.85 | 8.03 | **0.18** |
| 3 | 200 | 20 | 3.71 | 1.32 | 13.51 | **0.03** |
| 3 | 500 | 20 | 2.01 | 0.39 | 0.92 | **0.43** |
| | **ARPD** | | 9.91 | 6.89 | 4.13 | **0.58** |

## 7. Conclusion

This paper investigated the Flow Shop Scheduling Problem with Heterogeneous Workers and proposed an IG algorithm that explicitly incorporates worker–machine assignment decisions. The proposed IG-HW combines job-sequence optimization with a dedicated local search on worker assignments, enabling the algorithm to exploit worker heterogeneity effectively. IG-HW consistently outperforms NEH, the original IG, and Scatter Search in all evaluated cases, as indicated by computational results. The proposed method achieves the lowest overall ARPD values and maintains stable performance across different levels of heterogeneity, machine sizes, and job counts. The comparison with the original IG highlights that optimizing job sequences alone is insufficient in the heterogeneous worker setting, and that explicit worker reassignment is essential. Overall, the results confirm that IG-HW is an effective and robust solution approach for the Flow Shop Scheduling Problem with Heterogeneous Workers. The integration of worker-oriented neighborhoods within the IG framework represents a promising direction for scheduling problems involving human resource heterogeneity. This study can be expanded in several ways: (i) The data under consideration can be solved using different algorithms to test whether better results can be obtained. (ii) The proposed method can be tested with different datasets and the results analyzed. (iii) The problem can be applied in any production or assembly field using the mathematical model and the proposed heuristic method. Furthermore, sensitivity analysis can be performed as a continuation of this study to observe the effects of the parameters on the objective functions.

**Acknowledgments**

This work was supported by the National Council for Scientific and Technological Development - Brazil under grant number 312585/2021-7 and 404819/2023-0.

## References

Bai, D., Bai, X., Li, H., Pan, Q. K., Wu, C. C., Gao, L., & Lin, L. (2022). Blocking flowshop scheduling problems with release dates. *Swarm and Evolutionary Computation, 74*, 101140.

Becker, T., Neufeld, J., & Buscher, U. (2025). The distributed flow shop scheduling problem with inter-factory transportation. *European Journal of Operational Research, 322*(1), 39–55.

Benavides, A. J., Ritt, M., & Miralles, C. (2014). Flow shop scheduling with heterogeneous workers. *European Journal of Operational Research, 237*(3), 713–722.

Benkalai, I., Rebaine, D., & Baptiste, P. (2019). Scheduling flow shops with operators. *International Journal of Production Research, 57*(2), 338–356.

Blum, C., & Miralles, C. (2011). An introduction to beam search and its application to assembly line worker assignment and balancing problems. *Computers & Operations Research, 38*(1), 301–311.

Cheng, T. E., & Kovalyov, M. Y. (2003). Scheduling a single server in a two-machine flow shop. *Computing, 70*(2), 167–180.

Chu, F., Liu, M., Liu, X., Chu, C., & Jiang, J. (2018). Reentrant flow shop scheduling considering multiresource qualification matching. *Scientific Programming, 2018*, 2615096.

Corominas, A., Pastor, R., & Plans, J. (2008). Balancing assembly lines with heterogeneous workers. *European Journal of Operational Research, 177*(1), 264–280.

Fekri, M., Heydari, M., & Mazdeh, M. M. (2024). Bi-objective optimization of flexible flow shop scheduling problem with multi-skilled human resources. *Engineering Applications of Artificial Intelligence, 133,* 108094.

Fernandez-Viagas, V., & Framinan, J. M. (2014). On insertion tie-breaking rules in heuristics for the permutation flowshop scheduling problem. *Computers & Operations Research, 45*, 60–67.

Fernandez-Viagas, V., Ruiz, R., & Framiñán, J. M. (2017). A new vision of approximate methods for the permutation flowshop to minimise makespan: State-of-the-art and computational evaluation. *European Journal of Operational Research, 257*(3), 707–721.

Fernandez-Viagas, V., Sánchez-Mediano, L., Angulo-Cortés, A., Gómez-Medina, D., & Molina-Pariente, J. M. (2022). The permutation flow shop scheduling problem with human resources: MILP models, decoding procedures, NEH-based heuristics, and an iterated greedy algorithm. *Mathematics, 10*(19), 3446.

Framiñán, J. M., Leisten, R., & Rajendran, C. (2003). Different initial sequences for the heuristic of Nawaz, Enscore and Ham to minimize makespan, idletime or flowtime in the static permutation flowshop sequencing problem. *International Journal of Production Research, 41*(1), 121–148.

Garey, M. R., & Johnson, D. S. (1977). Two-processor scheduling with start-times and deadlines. *SIAM Journal on Computing, 6*(3), 416–426.

Gogos, C. (2023). Solving the distributed permutation flow-shop scheduling problem using constrained programming. *Applied Sciences, 13*(23), 12562.

Gong, G., Chiong, R., Deng, Q., Han, W., Zhang, L., Lin, W., & Li, K. (2020). Energy-efficient flexible flow shop scheduling with worker flexibility. *Expert Systems with Applications, 141*, 112902.

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Rinnooy Kan, A. H. G. (1979). Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics, 5*, 287–326.

Irmouli, M., Benazzoug, N., Adimi, A. D., Rezkellah, F. Z., Hamzaoui, I., Hamitouche, T., & Tayeb, F. S. (2023). Genetic algorithm enhanced by deep reinforcement learning in parent selection mechanism and mutation: Minimizing makespan in permutation flow shop scheduling problems. arXiv preprint arXiv:2311.05937.

Komaki, G. M., Sheikh, S., & Malakooti, B. (2019). Flow shop scheduling problems with assembly operations: A review and new trends. *International Journal of Production Research, 57*(10), 2926–2955.

Li, J., Lin, P., Wu, X., Song, D., Yang, B., & Zhou, L. (2024). Scheduling optimization of ship plane block flow line considering dual resource constraints. *Scientific Reports, 14*(1), 30765.

Liu, X., Chen, X., Zhao, Y., Liu, X., Wu, C. C., & Lin, W. C. (2025). A multi-objective flexible flow shop scheduling problem with an improved NSGA-II algorithm. *Journal of Industrial and Management Optimization, 21*(5), 4041–4062.

Liu, Y., Shen, W., Zhang, C., & Sun, X. (2023). Agent-based simulation and optimization of hybrid flow shop considering multi-skilled workers and fatigue factors. *Robotics and Computer-Integrated Manufacturing, 80*, 102478.

Mansouri, M., Bahmani, Y., & Smadi, H. (2023). Optimization of the Flow-Shop Scheduling Problem under Time Constraints with PSO Algorithm. *Engineering Proceedings, 56*(1), 220.

Miralles, C., Garcia-Sabater, J. P., Andres, C., & Cardos, M. (2007). Advantages of assembly lines in sheltered work centers for the disabled: A case study. *International Journal of Production Research, 45*(3), 509–522.

Moreira, M. C., Costa, A. M., & Miralles, C. (2012). Genetic algorithms for the assembly line worker assignment and balancing problem. *Expert Systems with Applications, 39*(4), 4478–4489.

Nawaz, M., Enscore, E. E., & Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega, 11*(1), 91–95.

Nicosia, G., Pacifici, A., Pferschy, U., Russo, A. R., & Salvatore, C. (2024). Flow shop scheduling with inter-stage flexibility and blocking constraints. arXiv preprint arXiv:2411.18381.

Pan, Q.-K., & Ruiz, R. (2014). An effective iterated greedy algorithm for the mixed no-idle permutation flowshop scheduling problem. *Omega, 44*, 41–50.

Park, I., Lee, K., & Pinedo, M. (2025). A flow shop scheduling problem with machine-dependent speeds: An ensemble approach with worst-case analysis. *European Journal of Operational Research, 330*(2), 373-380.

Rad, S. F., Ruiz, R., & Boroojerdian, N. (2009). New high performing heuristics for minimizing makespan in permutation flowshops. *Omega, 37*(2), 331-345.

Rossi, F. L., & Nagano, M. S. (2021). Heuristics for the mixed no-idle flowshop with sequence-dependent setup times. *Journal of the Operational Research Society, 72*(2), 417–443.

Ruiz, R., & Maroto, C. (2005). A comprehensive review and evaluation of permutation flowshop heuristics. *European Journal of Operational Research, 165*(2), 479–494.

Ruiz, R., & Stützle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *European Journal of Operational Research, 177*(3), 2033–2049.

Ruiz, R., & Stützle, T. (2008). An iterated greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives. *European Journal of Operational Research, 187*(3), 1143–1159.

Samarghandi, H. (2015). Studying the effect of server side-constraints on the makespan of the no-wait flow-shop problem with sequence-dependent set-up times. *International Journal of Production Research, 53*(9), 2652–2673.

Singla, S., Kaur, H., Gupta, D., Modibbo, U. M., & Kaur, J. (2024). No idle flow shop scheduling models for optimization of machine rental costs with processing and separated setup times. *Frontiers in Applied Mathematics and Statistics, 10,* 1355237.

Su, Z., Deng, C., Chiong, R., Jiang, S. L., & Zhang, K. (2025). Hybrid flow shop scheduling with continuous processing and resource threshold constraints: A case of steel plant. *Expert Systems with Applications, 284*, 127247.

Taillard, E. (1993). Benchmarks for basic scheduling problems. *European Journal of Operational Research, 64*(2), 278–285.

Tasgetiren, M. F., Pan, Q. K., Liang, Y. C., & Suganthan, P. N. (2013). Iterated greedy algorithms for the no-idle flowshop scheduling problem. *Computers & Operations Research, 40*(7), 1679–1691.

Tosun, Ö., Marichelvam, M. K., & Tosun, N. (2020). A literature review on hybrid flow shop scheduling. *International Journal of Advanced Operations Management, 12*(2), 156–194.