Impact of learning effect modelling in flowshop scheduling with makespan minimisation based on the Nawaz-Enscore-Ham algorithm

Yenny Alexandra Paredes-Astudillo, Valérie Botta-Genoulaz, Jairo Montoya-Torres

To cite this article:
Yenny Alexandra Paredes-Astudillo, Valérie Botta-Genoulaz & Jairo R. Montoya-Torres (2023) Impact of learning effect modelling in flowshop scheduling with makespan minimisation based on the Nawaz-Enscore-Ham algorithm, *International Journal of Production Research*, DOI: 10.1080/00207543.2023.2204967

Impact of learning effect modeling in flowshop scheduling with makespan minimization based on the Nawaz-Enscore-Ham algorithm

Yenny Alexandra Paredes-Astudillo[a,b*], Valérie Botta-Genoulaz[a],

Jairo R. Montoya-Torres[b]

[a] *Univ Lyon, INSA Lyon, Université Claude Bernard Lyon 1, Univ Lyon 2, DISP UR4570,69621 Villeurbanne, France (e-mail: yenny.paredes-astudillo@insa-lyon.fr, valerie.botta@insa-lyon.fr)*

[b] *School of Engineering, Universidad de La Sabana, km 7 autopista norte de Bogotá D.C., Chía, Colombia (e-mail: yennypaas@unisabana.edu.co , jairo.montoya@unisabana.edu.co)*

Notes on contributors

Yenny Alexandra Paredes-Astudillo is a PhD student and Logistics and Supply Chain Management as part of a cotutelle agreement between INSA-Lyon, the University of Lyon, France and Univerisdad de la Sabana, Colombia. She received her master's degree in industrial engineering from the Pontificia Universidad Javeriana, Colombia, in 2018. Her research interests are related to the integration of human factors engineering in scheduling and managing production.

Valérie Botta-Genoulaz is currently full Professor in the Industrial Engineering Department at INSA-Lyon, Université de Lyon, France. She does her research at the DISP research lab (Decision and Information Systems for Production systems), which she created and directed from 2011 to 2020. Her research interests deal with operation planning, supply chain management, as well as enterprise information system alignment, and their impacts on sustainable performance. She is involved in many national and international research networks, conference program committees and journal editorial board. She published about 130 papers in journals, conferences or book chapters, and co-chaired 10 books or international journal special issues and has an important expertise activity at national and international levels.

Jairo R. Montoya-Torres is Full Professor within the School of Engineering at Universidad de La Sabana, Colombia. He also acts as director of the PhD program in Engineering and PhD program in Logistics and Supply Chain Management. He holds a Ph.D. degree from the Ecole des Mines de Saint-Etienne, Saint- Etienne, France, and an HDR (post-doctoral diploma) from the National Institute of Applied Sciences (INSA) Lyon and Université Claude Bernard Lyon 1, France. He has been invited professor or researcher at different universities in France, Spain, USA and the UK. His research interests include supply chain management and design, sustainability in logistics and SCM, and operations scheduling using optimization, simulation, and hybrid techniques.

ORCID

Yenny Alexandra Paredes-Astudillo: https://orcid.org/0000-0001-9510-4619

Valérie Botta-Genoulaz: https://orcid.org/0000-0003-2565-6690

Jairo R. Montoya-Torres: https://orcid.org/0000-0002-6251-3667

Impact of learning effect modeling in flowshop scheduling with makespan minimization based on the Nawaz-Enscore-Ham algorithm.

Inspired by real-life applications, mainly in hand-intensive manufacturing, the incorporation of learning effects into scheduling problems has garnered attention in recent years. This paper deals with the flowshop scheduling problem with a learning effect, when minimizing the makespan. Four approaches to model the learning effect, well-known in the literature, are considered. Mathematical models are providing for each case. A solver allows us to find the optimal solution in small problem instances, while a Simulated Annealing algorithm is proposed to deal with large problem instances. In the latter, the initial solution is obtained using the well-known Nawaz-Enscore-Ham algorithm, and two local search operators are evaluated. Computational experiments are carried out using benchmark datasets from the literature. The Simulated Annealing algorithm shows a better result for learning approaches with fast learning effects as compared to slow learning effects. Finally, for industrial decision makers, some insights about how the learning effect model might affect the makespan minimization flowshop scheduling problem are presented.

Keywords: scheduling, flowshop, learning effect, simulated annealing, metaheuristic

1 Introduction

Human workers are still an essential resource in manufacturing systems and assembly lines, particularly in developing countries, where production systems are largely manual and factories are often perceived as a source of employment (Baudin 2002). As industrial automation technologies have limited flexibility (Kadir, Broberg and Da Conceicao 2019), complex tasks do still require certain skills typical of human beings (e.g., precision, intelligence, analysis and logic) (Sánchez-Herrera, Montoya-Torres and Solano-Charris 2019). People are inherently more flexible than machines (Daniels, Mazzola and Shi 2004; Hashemi-Petrood et al. 2020) and have been involved in production systems implicitly or explicitly since the appearance of the latter (Dessouky, Moray and Kijowski 1995).

Some examples of hand-intensive systems in industry are the luxury industry, artisan production, manual palletizing and un-palletizing (Calzavara et al. 2019), manual feeding of materials to assembly, and order picking (Vijayakumar et al. 2022; Katiraee et al. 2022; Calzavara et al. 2019). The last two are perhaps the ones that have received the most attention recently. Manual assembly lines have been designed to produce a variety of product variants (Bortolini et al. 2016), and order preparation systems meet a complex, highly customized global demand that requires the processing of many orders in short time windows (Vanheusden et al. 2022).

Yet theories such as scheduling, dating back to the 1950's, have incorporated assumptions and simplifications with regard to humans, such as the claim that workers are not a major resource, or that their performance is deterministic (Lodree, Geiger and Jiang 2009). As a result, there is a dichotomy between ergonomics and operations management. This is witnessed in the fact that publications on workers' well-being are seldom published in engineering, management, or business journals (Neumann and Dul

2010). In recent years, some authors have nevertheless recognized the importance of involving human workers in production systems. They have highlighted the opportunities for research in the production, operations management, and operational research fields to integrate human behavior and ergonomics (Boudreau et al. 2003; Hashemi-Petrood et al. 2020; Lodree, Geiger and Jiang 2009; Sánchez-Herrera, Montoya-Torres and Solano-Charris 2019).

Of all the human characteristics that have an impact on the productivity of industrial environments, the learning effect is one of the most studied. This effect was induced scientifically by Wright (1936), but it was not until 1998 that Dondeti and Mohanty (1998) spoke about the role of learning and fatigue in single-machine problems, when the job processing time depends on the content of both the present and the previously processed jobs. Biskup (1999) formalized the effect of learning on one machine. Since then, a significant number of papers have proposed different ways of modeling the learning effect on scheduling problems (Paredes-Astudillo, Montoya-Torres and Botta-Genoulaz 2022a; Pei et al. 2022). Over the last 20 years, some learning approaches have been applied in practical scenarios involving humans, summarized by Pei et al. (2022) in their review paper. Examples include automotive assembly lines, processing of memory chips and automotive components, catalytic processes in the chemical industry, and order picking, among others.

The most commonly used approaches are the position-based learning effect (Biskup 1999; Cheng and Wang 2000; Dolgui, Gordon and Strusevich 2012), and the sum-of-processing-time-based learning effects (Kuo and Yang 2006a, 2006b, 2006c; Koulamas and Kyparisis 2007), as well as its variations, including truncated parameters (Wang et al. 2013; Wu, Yin and Cheng 2011).

In recent years, researchers have focused on this problem mainly in the single machine environment, because it is possible to conceptualize the problem and extrapolate alternatives for other types of configurations. However, flowshop configurations and their variations are frequently encountered in complex manufacturing systems, due to the necessity to perform sequential operations, as in the case of textiles, footwear and in the automotive industry (Chen et al. 2017; Fernandez-Viagas 2022; Rudek, 2011). While the existing articles do analyze the complexity of problems and propose solution methods, they do not compare the efficiency of solution methods based on the learning effect approach.

The main contributions of this paper, which deals with the flowshop scheduling problem (FSSP) with learning effect, are:

- To provide FFSP mathematical models that address the learning effect with the four approaches referred to in the literature.
- To solve small-problem instances with a mathematical programming solver.
- To solve large-size instances, a simulated annealing (SA) algorithm testing is proposed, using the NEH algorithm to get the initial solution, and two local search operators.
- To discuss the effectiveness of the SA algorithm and its performance regarding the way the learning effect is modeled through a sensitivity analysis.

It is true that NEH and SA have previously been used to solve the FSS with learning effects. Those early studies chose a learning model and developed a solution algorithm. The current study differs from them, however, since both NEH and SA algorithms are adapted to analyze the impact of the different learning models and their parameters on the efficiency of such algorithms. Experimental results are expected to be useful for application in real-world situations, particularly in hand-intensive manufacturing systems where the learning process is a determining factor in productivity rates.

The remainder of this paper is organized as follows. Section 2 reviews the related literature. Section 3 presents the description of the problem and the mathematical models of the learning effect. The proposed solution approach for large instances is presented in Section 4, while Section 5 is devoted to the computational experiments and the analysis of results. Finally, conclusions and future research opportunities are outlined in Section 6.

2 Literature review

Wright (1936) introduced the concept of the learning effect in manufacturing systems – an effect stemming from practical experience (Arditi, Tokdemir and Suh 2001), particularly in monotonous activities. In recent years, the inclusion of the learning effect when dealing with operations management problems has become more widespread and has produced several learning models published in the literature. The baseline of scheduling with learning effects was established by Biskup (1999), who modeled a position-based learning effect into a single-machine scheduling problem. This case defines the actual processing time p_{jr} of the job j located in position r of the schedule to be computed as $p_{jr} = \bar{p}_j r^{\alpha}$ where $\bar{p}_j$ is the baseline processing time of job j (e.g., without learning) and $\alpha < 0$ is a constant learning index. From this approach, several modifications have been introduced, to adapt them to other system configurations and integrate additional parameters. This is the case of truncated position-based learning effects, where learning is not considered to be infinite, and the actual processing time depends on the job position and a control parameter. Wu et al. (2014) modeled this learning effect in a single machine scheduling problem as $p_{jr} = p_j * max\{r^{\alpha}, \beta\}$, where β is a control parameter $(0 < \beta < 1)$.

Other authors have also considered learning effect approaches based on the sum-of-processing-time. Kuo and Yang (2006a) introduced a new model with the premise that the performance will improve if the workers practice for longer. The actual processing time P_{jr} in a single machine system of a job j scheduled in position r is defined as $p_{jr} = (1 + \sum_{k=1}^{r-1} p_k)^{\alpha} p_j$, where $\sum_{k=1}^{r-1} p_k$ is the cumulative processing time of jobs from position 1 to position $r - 1$, and p_j is the baseline processing time of job j. Based on this premise and considering that learning is limited, Wu et al. (2012b) proposed, for a single-machine scheduling problem, a truncated sum-of-processing-time model, which is computed as $p_{jr} = max\{\left(1 + \sum_{k=1}^{r-1} p_{jk}\right)^{\alpha}, \beta\} p_j$. Some other models of learning effects are available in detail in the reviews of Azzouz, Ennigrou, and Ben Said (2018), Glock et al. (2019), Paredes-Astudillo, Montoya-Torres and Botta-Genoulaz (2022a) and Pei et al. (2022).

The flowshop scheduling problem without learning and with makespan minimization is known to be NP-hard for the case of more than two machines (Wang and Xia 2005). When dealing with learning in processing times, the problem becomes NP-hard even for the case of two machines (Pinedo 2018). This means that optimal solutions cannot be obtained for large-sized datasets in reasonable computational time. Because of this computational intractability, the literature has witnessed a variety of solution methods to solve flowshop scheduling problems with learning effects. To deal with a two-machine flowshop scheduling problem with truncated learning effects, both Cheng et al. (2013) and Wu et al. (2012a) propose a genetic algorithm (GA) and a branch-and-bound algorithm, while minimizing the makespan and the total completion time by applying some dominance rules. Zou et al. (2020) addressed a two-stage three-machine flowshop scheduling problem with a sum-of-processing-times-based learning effect to minimize the makespan. They proposed a branch-and-bound algorithm incorporating dominance properties, three heuristics based on Johnson's rule, and a GA. Wu et al. (2020) studied a two-stage three-machine flowshop scheduling problem with a truncated sum-of-processing-time-based learning effect, where the makespan is intended to be minimized. They present some dominance rules, and develop a branch-and-bound algorithm and a GA to obtain near-optimal solutions. Wu et al. (2018a) deal with the re-entrant permutation flowshop scheduling with a sum-of-processing-times-based learning effect to minimize the makespan. They propose four heuristics and a SA to approximate solutions.

In addition to genetic algorithms, other metaheuristics such as simulated annealing (SA) have been widely used. Sun, Geng and Liu (2020) consider the flowshop problem of minimizing the total weighted completion time, where the job processing time is computed according to a general position-weighted learning effect. SA and branch and bound algorithms are proposed. Liu (2020) also proposes a SA to solve the two-stage three-machine flowshop, while approaching the learning effect as a truncated function of sum-of-processing time in order to minimize makespan. Azizi, Jabbari, and Kheirkhah (2016) studied the m-machine flowshop scheduling problem, considering sequence-dependent setup times and truncated learning function to minimize the makespan. To do so, they developed both GA and SA. Lai and Wu (2015) used GA, SA, ant colony optimization (ACO) and particle swarm optimization (PSO) to minimize the makespan. They incorporated three kinds of variations of learning effect models: job-dependent, machine-dependent, and job- and machine-dependent learning effects, depending on the position.

Rudek and Rudek (2013) and Wu et al. (2018b) deal with two- and three-machine flowshop problems to minimize makespan and describe the job processing time by learning based on the position. They construct a Nawaz-Encore-Ham (NEH) algorithm, tabu search (TS) with neighborhood search, and SA algorithms that solve the problem.

Some authors have recently addressed this problem through hybrid-metaheuristics. For example, Wu et al. (2018b) and Zou et al. (2020) alternatively include a cloud theory-based simulated annealing algorithm (CSA). Fu et al. (2019) consider a flowshop scheduling problem with learning and deterioration effects and propose an artificial-molecule-based chemical reaction optimization algorithm (ACRO). Vahedi Nouri, Fattahi, P and Ramezanian (2013) study a flowshop scheduling problem with learning effects and maintenance activities. They develop a hybrid meta-heuristic algorithm based on an SA algorithm and a firefly algorithm (HFSA) to solve it. Muştu and Eren (2018) address a flowshop scheduling problem under a position-based learning effect and minimize the makespan. They propose a kangaroo algorithm (KA) and a genetic-kangaroo hybrid algorithm (GAKA) to solve large instances of this problem. Arık (2021) deals with flowshop scheduling problems with position-dependent learning effects and linear deterioration. He proposes a population-based tabu search algorithm (TSPOP) with evolutionary strategies.

In terms of multi-objective problems, Hosseini and Tavakkoli-Moghaddam (2013) deal with a two-machine flowshop scheduling problem with learning effects that minimizes the total idle time and the mean deviation from a common due date. They solve the problems with a multi-objective genetic algorithm (MOGA) and a multi-objective simulated annealing (MOSA) algorithm. Eren and Güner (2008) studied a two-machine flowshop scheduling problem with learning effects based on the position, the objective function of which is the minimization of a weighted sum of total completion time and makespan. This author presents heuristic algorithms and a TS algorithm to solve large-sized problems. Chen, Wu, and Lee. (2006) addressed a bi-criteria two-machine flowshop scheduling problem with learning effects based on the position. They proposed a branch-and-bound algorithm, a heuristic and a SA algorithm to approximate solutions for large instances of the problem. Table 1 summarizes the information previously described.

Table 1. Synthesis of current works

[Table 1 near here]

3 The flowshop scheduling problem under study

We consider the permutation flowshop scheduling problem with a set I of workers, and a set J of independent jobs, which are processed in the same sequence to minimize the makespan (C_{max}). Each worker can process one job at a given time, and preemption of a job is not allowed (that is, the execution of a job cannot be interrupted once its processing has started). All workers are available at the beginning of the scheduling horizon and have a 100% production rate, and scheduling is performed through the permutation sequence. The inclusion of intermediate buffers between workers is not considered in this approach. The normal (baseline) processing time of the i^{th} operation of the j^{th} job is noted as $\bar{p}_{ij}$. As an illustration, the production system under study might refer to a chocolate truffle production line or a sequential zone-picking line, where the work is entirely manual.

The standard flowshop scheduling problem with makespan minimization can be modeled as a mixed-integer linear programming (MILP) model. The definition and notations are shown below:

Sets

I: workers

J: jobs

R: positions

Parameters

$\bar{p}_{ij}$: normal (baseline) processing time of job j executed by worker i

M: $r$epresents a very big number or Big M

Decision Variables:

$x_{jr}: \begin{cases} 1\text{: if the job } j \text{ is processed in position } r \text{ of the schedule} \\ 0\text{: otherwise} \end{cases}$

c_{ij}: completition time of job j on worker i

C_{max}: makespan value

Objective function

$$Minimize\ Z = C_{max} \tag{1}$$

Subject to:

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

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

$$c_{1j} \geq \bar{p}_{1j} \qquad j = 1, \dots, J \tag{4}$$

$$c_{ij} - \bar{p}_{ij} \geq c_{(i-1)j} \qquad i = 2, \dots, I;\ j = 1, \dots, J \tag{5}$$

$$c_{ij} - \bar{p}_{ij} + M\left(1 - x_{j(r+1)}\right) \geq c_{ih} - M(1 - x_{hr}) \qquad i = 1, \dots, I;\ j = 1, \dots, J;\ h = 1, \dots, J;\ r = 1, \dots, R-1 \qquad (6)$$

$$C_{max} \geq c_{ij} \qquad i = 1, \dots, I;\ j = 1, \dots, J \qquad (7)$$

$$x_{jr} \in \{0,1\} \qquad j = 1, \dots, J;\ r = 1, \dots, R \qquad (8)$$

The objective function (1) corresponds to the minimization of the completion time of the last job of the sequence, i.e. the makespan. Constraints (2) and (3) guarantee that every job is assigned to one position in the sequence, and each position has only one job. Constraints (4) are related to the completion time of jobs for the first worker (i.e. first operation). This ensures the non-negativity constraint, which is usually formalized with constraints (9) and (10). Constraints (5) and (6) calculate the completion time of jobs for the remaining workers. Constraints (7) define the makespan. Constraints (10) define the values of binary decision variables.

$$C_{max} \geq 0 \qquad (9)$$

$$c_{ij} \geq 0 \qquad i = 1, \dots, I;\ j = 1, \dots, J \qquad (10)$$

The previous mathematical model can be modified to take into account the different approaches for modeling the learning effect, as proposed by several authors such as Biskup (2008), and Azzouz, Ennigrou, and Ben Said (2018). The models have been selected because they are the basis for other approaches (Paredes-Astudillo, Montoya-Torres and Botta-Genoulaz 2022b).

- *Case 1*: with position-based learning $p_{ijr} = \bar{p}_{ij} r^{\alpha}$
- *Case 2:* with truncated position-based learning $p_{ijr} = \bar{p}_{ij} \max\{r^{\alpha}, \beta\}$
- *Case 3*: with sum-of-processing-time-based learning $p_{ijr} = \left(1 + \theta \sum_{k=1}^{r-1} p_{ijk}\right)^{\alpha} \bar{p}_{ij}$
- *Case 4*: with truncated sum-of-processing-time-based learning $P_{ijr} = max\left\{\left(1 + \theta \sum_{k=1}^{r-1} p_{ijk}\right)^{\alpha}, \beta\right\} \bar{p}_{ij}$

Where α is the learning index ($\alpha < 0$), β is a control parameter with $0 < \beta < 1$, and θ is a conversion factor (e.g., 1/60 to convert hours to minutes). In case 1, the job processing time is based on the position, so taking the baseline flowshop model, a new decision variable is added:

p_{ij}: actual processing time of job j by worker i

To calculate the job processing time, constraints (11) and (12) are needed:

$$p_{ij} = \sum_{r \in R} \bar{p}_{ij} x_{jr}\, r^{\alpha} \qquad i = 1, \dots, I;\ j = 1, \dots, J \qquad (11)$$

$$p_{ij} \geq 0 \qquad i = 1, \dots, I;\ j = 1, \dots, J \qquad (12)$$

We replace Constraints (4), (5) and (6) by (13), (14) and (15) respectively.

$$c_{1j} \geq p_{1j} \qquad j = 1, \dots, J \qquad (13)$$

$$c_{ij} - p_{ij} \geq c_{(i-1)j} \qquad i = 2, \dots, I;\ j = 1, \dots, J \qquad (14)$$

$$c_{ij} - p_{ij} + M\left(1 - x_{j(r+1)}\right) \geq c_{ih} - M(1 - x_{hr}) \qquad i = 1, \dots, I;\ j = 1, \dots, J;\ h = 1, \dots, J;\ r = 1, \dots, R-1 \qquad (15)$$

From *case 1* and changing constraints (11) for (16), we would get *case 2:*

$$p_{ij} = \sum_{r \in R} \bar{p}_{ij} x_{jr} \max\{r^{\alpha}, \beta\} \qquad i = 1, \dots, I;\ j = 1, \dots, J \qquad (16)$$

In *case 3*, the decision variables c_{ij} and p_{ij} are replaced by c_{ir} and p_{ir}, where c_{ir} is the competition time of the job scheduled in the position r $\in R$ for the worker $i \in I$, and p_{ir} is the actual processing time of the job scheduled in the position r $\in R$ for the worker $i \in I$

Furthermore, equations (17) and (18) are included and replace equation (11) from *case 1*:

$$p_{ir} = \left(1 + \theta \sum_{q \in R}^{q<r} p_{iq}\right)^{\alpha} \left(\sum_{j \in J} \bar{p}_{ij} x_{jr}\right) \qquad i = 1, \dots, I;\ r = 1, \dots, R-1 \qquad (17)$$

$$p_{i1} = \left(\sum_{j \in J} \bar{p}_{ij} x_{j1}\right) \qquad i = 1, \dots, I \qquad (18)$$

Constraints (19), (20) and (21) control the completion times of the jobs at the machines, and ensure the non-negativity constraint, which is usually formalized with Constraints (22).

$$c_{1r} \geq p_{1r} \qquad r = 1, \dots, R \qquad (19)$$

$$c_{ir} - p_{ir} \geq c_{(i-1)r} \qquad i = 2, \dots, I;\ r = 1, \dots, R \qquad (20)$$

$$c_{ir} - p_{ir} + M\left(1 - x_{j(r+1)}\right) \geq c_{ir} - M(1 - x_{hr}) \qquad i = 1, \dots, I;\ j = 1, \dots, J;\ h = 1, \dots, J;\ r = 1, \dots, R-1 \qquad (21)$$

$$c_{ir} \geq 0 \qquad i = 1, \dots, I;\ r = 1, \dots, R \qquad (22)$$

For c*ase 4*, the set *E*, which states *{1:* Learning *2:* Truncate*}* was taken into consideration, as were the two variables which are defined:

u_{ire}: actual processing time of the job scheduled in position $r \in R$ for the worker $i \in I$ in the state $e \in E$

ξ_{ir}: maximun processing time of the job scheduled in position $r \in R$ for the worker $i \in I$

Equations (23), (24), (25), (26) and (27) are used to calculate the job processing time instead of equations (17) and (18). Constraint (28) and (29) are the non-negative constraint:

$$p_{i1} = \left(\sum_{j \in J} x_{j1}\right) \qquad i = 1, \dots, I \qquad (23)$$

$$u_{ir1} = \left(1 + \theta \sum_{q \in R}^{q<r} p_{iq}\right)^{\alpha} \left(\sum_{j \in J} \bar{p}_{ij} x_{jr}\right) \qquad i = 1, \dots, I;\ r = 2, \dots, R \qquad (24)$$

$$u_{ir2} = \sum_{j \in J} \bar{p}_{ij} x_{jr}\, \beta \qquad i = 1, \dots, I;\ r = 2, \dots, R \qquad (25)$$

$$\xi_{ir} \geq u_{ire} \qquad i = 1, \dots, I;\ r = 1, \dots, R;\ e = 1, \dots, E \qquad (26)$$

$$p_{ir} = \xi_{ir} \qquad i = 1, \dots, I;\ r = 2, \dots, R \qquad (27)$$

$$u_{ire} \geq 0 \qquad i = 1, \dots, I;\ r = 1, \dots, R;\ e = 1, \dots, E \qquad (28)$$

$$\xi_{ir} \geq 0 \qquad i = 1, \dots, I;\ r = 1, \dots, R \qquad (29)$$

We thus obtain two MILP models (*case 1* and *case 2*) and two mixed-integer non-linear programming (MINLP) models (*case 3* and *case 4*).

4 Simulated annealing approach

As pointed out above, the flowshop scheduling problem is known in the literature to be NP-hard with learning effects and makespan minimization, even for the case of two resources (workers in this case) (Wang and Xia 2005). This paper therefore proposes a SA algorithm to solve the problem. The NEH algorithm was proposed as a start point because of its efficiency in minimizing the makespan in the FSSP (Turner and Booth 1987; Ruiz and Maroto 2005). Likewise, it corresponds to one of the heuristics commonly referred to in the FSSP approach with a learning effect (Mosheiov and Pruwer 2021; Rudek and Rudek 2013; Wang and Wang 2014; Wu et al. 2018a; Wu et al. 2018b).

Simulated annealing, first proposed by Kirkpatrick, Gelatt and Vecchi (1983), is a popular metaheuristic widely used to solve different variants of the flowshop scheduling problem (with or without learning effects), as discussed in Section 2. SA is a local search metaheuristic capable of escaping from a local optimum due to the hill-climbing moves (Henderson, Jacobson and Johnson 2003). The analysis sensitivity of SA parameters such as initial temperature (T_0), cooling velocity (λ), and final temperature (T_f) are presented in Section 5.4. The solution representation used in this paper is shown in Figure 1 (permutation of jobs).

Figure 1 Caption: Solution representation

Figure 1 Alt Text: Example of permutation sequence encoding in a problem with 5 jobs.

[Figure 1 near here]

4.1 Initial solution: the Nawaz-Enscore-Ham algorithm (NEH)

The initial solution is obtained by a greedy heuristic proposed by the Nawaz-Enscore-Ham algorithm *(NEH)* (Nawaz, Enscore and Ham 1983), consisting of the following steps:

(1) Calculate the Total Processing Time *(TPT)* on all machines for each job *j.* This initial *TPT* per job is calculated with the normal (baseline) processing time and does not account for the learning effect.
(2) Sort all the jobs in decreasing order of *TPT* in a list.
(3) Select the two jobs with the highest *TPT* and remove them from the list. Two possible sequences are obtained with these jobs.
(4) Calculate the actual processing time of each job *j* according to the equation of the respective case (*Case 1, 2, 3* or *4).*
(5) Compute the C_{max} for each sequence and select the sequence with the minimal C_{max}.
(6) If the list is not empty, select the next job from the list and calculate all possible inserts within the sequence. Return to steps 4 and 5. Keep the sequence with the lowest makespan. This will be the initial solution (*S*).

4.2 Neighborhood generation phase

After obtaining the original solution S, the neighborhood generation phase is implemented, which is composed of a diversification strategy (DS) and a local search operator (LS).

The diversification strategy is based on a randomized insertion, and follows these steps:

(1) Randomly choose a job-*j* and position-*r* as part of the solution *S*.
(2) Insert the *j-th* job on the *r-th* position (Figure 2), to obtain a new solution S_0. Compute the $C_{max}(S_0)$
(3) Replace *S* by S_0.

Figure 2 Caption: Diversification strategy representation

Figure 2 Alt Text: Starting with the current permutation, where a random job is inserted into a new random position to obtain a new permutation.

[Figure 2 near here]

Once solution S is obtained, a first improvement local search (LS) operator is implemented to improve the quality of this solution. In this case, the Adjacent Pairwise Interchange (API) and Non-Adjacent Pairwise Interchange (NAPI) operators were selected as they are commonly used to solve flowshop scheduling problems, and have yielded good results (Della Croce, Narayan and Tadei 1996; Li 2018). The SA algorithm with the API and NAPI operators will be named $\mathrm{SA_{API}}$ and $\mathrm{SA_{NAPI}}$ respectively. K_1 and K_2 are the selected positions within the permutation sequence to be switched.

The API operator swaps jobs from adjacent positions as follows:

(1) If $K_1 = 1$ and $K_2 = K_1 + 1$, then the jobs between position K_1 and K_2 are swapped (Figure 3).
(2) The new solution is denoted S_0.
(3) If $C_{max}(S_0)$ is less than $C_{max}(S)$, then S is replaced by S_0 and the local search algorithm stops.
(4) Otherwise, $K_1 = K_1 + 1$ and $K_2 = K_2 + 1$ and the process is repeated until the $C_{max}(S)$ can be improved or up to $K_1 = R - 1$, or until all the position's permutations have been evaluated.

Figure 3 Caption: API operator

Figure 3 Alt Text: In a current permutation, swap jobs are performed between adjacent positions. For example, job in the first position is swapped with job in second position and vice versa.

[Figure 3 near here]

The NAPI operator swaps jobs from non-adjacent positions as follows.

(1) If $K_1 = 1$ and $K_2 = K_1 + 2$, then the jobs between position K_1 and K_2 are swapped respectively (Figure 4).
(2) The new solution is denoted S_0.
(3) If $C_{max}(S_0)$ is less than $C_{max}(S)$, then S is replaced by S_0 and the local search algorithm stops.
(4) Otherwise, $K_1 = K_1 + 1$ and $K_2 = K_2 + 2$ and the process is repeated until the $C_{max}(S)$ can be improved or up to $K_1 = R - 2$, or until all the position's permutations have been evaluated.

Figure 4 Caption: NAPI operator

Figure 4 Alt Text: In a current permutation, swap jobs are performed between non-adjacent positions. For example, job in the first position is swapped with job in third position and vice versa.

[Figure 4 near here]

4.3 *Acceptance probability*

A new given solution is accepted if its C_{max} value is lower than the C_{max} value of the stored solution ($\Delta f \leq 0$). However, when a worse solution is found ($\Delta f > 0$), a uniform random number is generated. If this random number is less than the probability of acceptance (equation 30), then the solution is acceptable.

$$P_{(accept)} = \begin{cases} 1 & \Delta f \leq 0 \\ e^{\frac{-\Delta f}{T}} & \Delta f > 0 \end{cases} \qquad (30)$$

Where $P_{(accept)}$ is the probability of acceptance, Δf is the change in objective function and T is the current temperature.

4.4 Stopping condition

In this algorithm, the current temperature (T) decreases. Once the final temperature (T_f) is reached (stopping condition), the SA stops.

The flowchart in Figure 5 details the components and operation of the SA algorithm.

Figure 5 Caption: SA flowchart

Figure 5 Alt Text: It corresponds to the flowchart of the proposed algorithm, describing four main elements such as: initial solution, neighborhood generation, the probability of acceptance and stopping condition.

[Figure 5 near here]

5 Computational experiments and results analysis

5.1 Description of data sets

A set of problem instances containing 1440 independent data sets (480 small and 960 large-sized instances) was used to test the performance of the proposed algorithm. The characteristics of each instance are briefly outlined in Table 2. The rationale for the values of alpha (α) and beta (β) is based on the most common values used in the literature (e.g., Amirian and Sahraeian 2015; Rudek and Rudek 2013; Muştu and Eren 2018; Wu et al. 2020). For the small-size instances, integer values of processing times were generated using an integer uniform distribution between 1 and 100. For the case of large-size datasets, the instances proposed by Taillard (1993) were used. In practice, combinations are referred to in the characteristics of instances. Each combination was encoded as Case_i_j_α_β (for example, C2_10_100_-0.515_0.5, corresponds to *Case 2* with 10 workers, 100 jobs, α = -0.515 and β = 0.5). A set of 10 instances for each combination was considered.

Table 2. Summary of set of problem instances

[Table 2 near here]

5.2 Experimental results small-size instances

The mathematical models were coded on Python and solved using Pyomo (Hart, Watson and Woodruff 2011). Glpk and Bonmin solvers were used for solving the linear (*case 1* and *case 2*) and non-linear cases (*case 3* and *case 4*) respectively.

The experiments were carried out for up to 7 jobs because for non-linear models (*case 3* and *case 4*), instances with 2 workers and 8 jobs could not be solved within less than 8 hours. The error percentage of the solution obtained by the simulated annealing algorithm ($\mathrm{SA_{API}}$ and $\mathrm{SA_{NAPI}}$) from the optimal solution is calculated as (Equation 31):

$$OPT - SA_{API} = \frac{S_{OPT} - S_{SA}}{S_{OPT}} \qquad (31)$$

Where S_{OPT} is the optimal solution for each instance and S_{SA} is the makespan value obtained from the metaheuristic ($\mathrm{SA_{API}}$ in this case). Table 3 provides a comparison of the mean percentage and the standard deviation of error and CPU time (of all instances by combination). For small-size instances, both $\mathrm{SA_{API}}$ and $\mathrm{SA_{NAPI}}$ achieved mean error percentages are below 1%. As there is no significant difference between the error percentage of $\mathrm{SA_{API}}$ and that of $\mathrm{SA_{NAPI}}$, it is not possible to conclude at this point with which one the SA triggers a better performance.

As can be expected, and is shown in Table 3, that for the exact method the CPU time increases significantly as the size of the problem increases. The CPU time for metaheuristics is significantly lower than the exact method. In particular, its advantages regarding the CPU time and error percentage are remarkable for nonlinear models (*case 3* and *case 4*).

Table 3. Comparison for small instance problems

[Table 3 near here]

5.3 *Experimental results - large-size instances*

An experimental design was conducted to evaluate the performance of the SA algorithm. Three main parameters required for the SA (T_0, T_f and λ) were included in the experiment. The algorithms were codified in Python 3.8 and run on a 64-Core server with CPU AMD EPYC 7702 and 512 G RAM. Table 4 summarizes the data used. Three replications per instance were run for a total of 155.520 executions.

Table 4. Summary experiment

[Table 4 near here]

We performed a complete factor-blocking design where the local search operator is the factor with two levels and the instance of each combination corresponds to a block.

As we cannot compute the optimal solution, we use the makespan value obtained by the NEH algorithm as the reference. The percentage of improvement of the initial solution obtained by NEH with respect to the simulated annealing algorithm ($\mathrm{SA_{API}}$ and $\mathrm{SA_{NAPI}}$) is calculated as follows (Equation 32):

$$\Delta SA_{API} - NEH = \frac{S_{SA} - S_{NEH}}{S_{NEH}} \qquad (32)$$

where $\mathrm{S_{NEH}}$ is the solution obtained with the heuristic NEH and $\mathrm{S_{SA}}$ is the solution achieved with the simulated annealing algorithm (with $\mathrm{SA_{API}}$ for this case) for each instance. With a *p-value* of $2\text{x}10^{-16}$ and a confidence level of 95%, the percentage of improvement of $\mathrm{SA_{API}}$ with the NEH ($\Delta\,\mathrm{SA_{API}} - \mathrm{NEH}$) is better than the improvement obtained with $\mathrm{SA_{NAPI}}$ ($\Delta\,\mathrm{SA_{NAPI}} - \mathrm{NEH}$). Table 5 shows a comparison of the mean percentage and standard deviation of improvement and CPU time. Because the quality of the initial solution of NEH is good, the API operators work as intensive operators, without making any drastic modification to the solution obtained constructively from NEH. Meanwhile, by making a major change to the solution, NAPI operators seem to be moving away from a promising search zone, which explains the predominance of $\mathrm{SA_{API}}$ over $\mathrm{SA_{NAPI}}$.

Table 5. Comparison for large instances problem.

[Table 5 near here]

Figure 6 shows a representative example with 10 workers and 50 jobs, to show the improvement of the two simulated annealing algorithms ($\mathrm{SA_{API}}$ and $\mathrm{SA_{NAPI}}$) in relation to the NEH, segmented by combination.

Figure 6 Caption: Comparison of $\Delta\, SA_{API} - NEH$ vs. $\Delta\, SA_{NAPI} - NEH$

Figure 6 Alt Text: Example of performance of all combinations of a problem with 10 workers and 50 jobs, regarding $\mathrm{SA_{API}}$ and $\mathrm{SA_{NAPI}}$ improvements with respect to the NEH.

[Figure 6 near here]

In most of the cases, $\mathrm{SA_{API}}$ outperforms $\mathrm{SA_{NAPI}}$. And the example in Figure 6 and Table 4 is evidence that the simulated annealing algorithms proposed (both $\mathrm{SA_{API}}$ and $\mathrm{SA_{NAPI}}$) achieve a better solution than does the NEH solution. This applies to combinations with small α values or strong learning rates (remembering that $\alpha = log_2 LR$, where *LR* is the learning rate, e.g., $LR = 70\%$, $\alpha = -0.515$), for *cases 1* and *3* without truncation parameters. In the same way, we note that for fast learning effects, the convergence of the algorithm is fast with respect to CPU time (Table 5), as described in the findings of Muştu and Eren (2018). For example, going back for combinations C3_10_100_-0.515_0.0 and C3_10_100_-0.152_0.0 solved with $\mathrm{SA_{API}}$, the mean CPU time is respectively 27.36 ± 9.88, 36.59 ± 18.97 seconds.

For learning models with truncation, the improvement by $\mathrm{SA_{API}}$ or $\mathrm{SA_{NAPI}}$ of the solution obtained from NEH turns out to be quite poor when β is larger. This could be explained by the fact that a larger β means that the learning effect stops quickly as it reaches the asymptote in a short planning horizon (units or accumulated time as the case may be). Therefore, the NEH solution might be a good approach because the processing time ceases to vary quickly. When β is smaller, the learning effect remains over a longer horizon, therefore metaheuristics are more likely to improve the initial solution.

5.4 *Sensitivity analysis*

For the evaluation of $\mathrm{SA_{API}}$ parameters, an ANOVA test was applied. Experimental factors refer to T_f, T_0 and λ with three, three, and two levels, respectively. The statistical test results showed that the double interaction T_0T_f and $T_f\lambda$ have an effect on the algorithm performance with a *p-value* of *0.0013* and $3.47x10^{-6}$, respectively. Figure 7 shows plots of double interaction effects for each case. A more favorable mean of the $\Delta SA_{API} - NEH$ is found in the four cases where for the double interaction $T_f\lambda$, the levels are $T_f = 0.00001$ and $\lambda = 0.9$. In both linear and non-linear models, this combination of parameters improves the initial solution obtained through the NEH. It can be assumed that the small values of T_f reinforce the intensification, and that the large values λ, together with the acceptance probability, contribute to diversify the solution.

The maximum mean improvement is close to 3%, which would show that if the NEH produces a good quality initial solution, this may be because this heuristic makes the best choice at every step, particularly regarding to the processing times of jobs that vary from step to step.

Figure 7 Caption: Double interaction plot

Figure 7 Alt Text: This shows the double interaction effects (T_0T_f and $T_f\lambda$) for the performance of the $\mathrm{SA_{API}}$ algorithm, for each case studied. The $T_f\lambda$ is more efficient at levels $T_f = 0.00001$ and $\lambda = 0.9$.

[Figure 7 near here]

6 Conclusions

This article addresses makespan minimization of flowshop scheduling problems with learning effects, modeling this phenomenon from 4 linear and non-linear models reported in the literature. We present the mathematical models and use a mathematical programing solver to solve small-instances problem. A simulated annealing algorithm is also proposed, which obtains the initial solution by means of the NEH algorithm, and which has shown remarkable results in addressing these problems (Rudek and Rudek 2013). In addition, two local search operators were tested.

The computational results support the hypothesis that the flowshop scheduling problem with learning effects is NP-hard even in the case of two resources (Wang and Xia 2005). Therefore, for problems with 2 workers and 8 jobs (*cases 3* and *4*, non-linear models), it was not possible to reach the optimum through the mathematical model within a time limit of 8 hours. Two variants of the simulated annealing algorithm were developed ($\mathrm{SA_{API}}, \mathrm{SA_{NAPI}}$), which get the initial solution from NEH and apply the different local search operators. $\mathrm{SA_{API}}$ shows improved performance in relation to $\mathrm{SA_{NAPI}}$.

Similarly, it was found that both simulated annealing algorithms achieve significant improvements over the NEH for combinations implying a fast learning effect (small values of α). For combinations with slow learning effects (big values of α) the proposed metaheuristic does not improve NEH solutions significantly. This is because the problem quickly resembles a "classical" flowshop scheduling problem with makespan minimization, for which NEH has reported exceptional results.

In *cases 2* and *4* (where a truncation parameter is included), the simulated annealing algorithms offered better improvements of the NEH solution for combinations with low β levels. For bigger β values, the algorithm does not significantly improve the initial solution. This may be explained by the fact that the problem quickly becomes a classic makespan minimization flowshop scheduling problem, for which the NEH finds a reasonably acceptable solution (Turner and Booth 1987; Ruiz and Maroto 2005).

This would mean that industries with less experienced employees and monotonous tasks could use a metaheuristic, as proposed here. On the other hand, if these are experienced workers or the learning effect is weak (highly personalized jobs), and a heuristic such as the NEH achieves good enough results for the makespan minimization flowshop, it could help the production planner to adapt the most suitable algorithm according to the workforce's characteristics.

Concerning the $\mathrm{SA_{API}}$ parameters' sensitivity, the double interaction $T_f\lambda$ ($T_f = 0.00001$ and $\lambda = 0.9$) has a positive effect on metaheuristic performance for the four cases studied here.

As future research opportunities, we can highlight the integration of variable learning rates and truncation parameters per worker. The inclusion of different learning models into the same system can also be considered. It would be useful to investigate the differences of workers in production systems and their impact on the performance production system (Katiraee et al. 2021). The incorporation of buffers or intermediate stations can also be evaluated as these configurations are closer to real-life production systems, such as assembly or zone-picking lines. Addressing the problem based on a multi-objective optimization that includes economic and social objectives would be a promising line of research. The learning effect could be combined with other phenomena such as fatigue and/or recovery (Dode et el. 2016; Givi, Jaber and Neumann 2015; Ostermeier 2020), for example, which implies interdisciplinary work and allows progress in modeling human factors. Finally, application in industrial contexts with real data (e.g., real learning rates) is crucial to address the challenge that comes with the arrival of the paradigm that the European Commission (2022) has called Industry 5.0: designing sustainable, human-centered, human-friendly production systems (Katiraee et al. 2022) with favorable working conditions.

Data Availability Statement

The authors confirm that the data supporting the findings of this study are available within the article.

Acknowledgments

This work was carried out thanks to research scholarships from Universidad de La Sabana (grants number INGPhD-45-2021 and INGPhD-51-2022) and from the Eiffel Excellence Scholarship PhD stream awarded to the first author by the French Ministry of Europe and Foreign Affairs.

Disclosure statement

No potential conflict of interest was reported by the author.

References

Amirian, H., and Sahraeian, R. 2015. "Augmented ε-constraint method in multi-objective flowshop problem with past sequence set-up times and a modified learning effect." *International Journal of Production Research*, 53(19), 5962–5976.

Arditi, D., Tokdemir, O. B., and Suh, K. 2001. "Effect of learning on line-of-balance scheduling." *International journal of project management*, 19(5), 265-277.

Arık, O. A. 2021. "Population-based Tabu search with evolutionary strategies for permutation flow shop scheduling problems under effects of position-dependent learning and linear deterioration." *Soft Computing*, *25*(2), 1501–1518.

Azizi, V., Jabbari, M., and Kheirkhah, A. S. 2016. "M-machine, no-wait flowshop scheduling with sequence dependent setup times and truncated learning function to minimize the makespan." *International Journal of Industrial Engineering Computations*, *7*(2), 309–322.

Azzouz, A., Ennigrou, M., and Ben Said, L. 2018. "Scheduling problems under learning effects: classification and cartography". *International Journal of Production Research*, 56(4), 1642-1661.

Baudin, M. 2002. "Lean assembly. the nuts and bolts of making assembly operations flow". *Productivity Press*.

Biskup, D. 2008. "A state-of-the-art review on scheduling with learning effects." *European Journal of Operational Research*, 188(2), 315-329.

Biskup, D. 1999. "Single-machine scheduling with learning functions". *European Journal of Operational Research.* 115, 173–178.

Bortolini, M., Faccio, M., Gamberi, M., and Pilati, F. 2016. "Including material exposure and part attributes in the manual assembly line balancing problem." *IFAC-PapersOnLine*, 49(12), 926-931.

Boudreau, J., Hopp, W., McClain, J. O., and Joseph Thomas, L. 2003. "On the Interface between Operations and Human Resources Management." *Manufacturing and Service Operations Management*, *5*(3), 179–202.

Calzavara, M., Persona, A., Sgarbossa, F., and Visentin, V. 2019. "A model for rest allowance estimation to improve tasks assignment to operators." *International Journal of Production Research*, 57(3), 948-962.

Chen, P., Wu, C. C., and Lee, W. C. 2006. "A bi-criteria two-machine flowshop scheduling problem with a learning effect." *Journal of the Operational Research Society*, *57*(9), 1113–1125.

Chen, X., Chau, V., Xie, P., Sterna, M., and Błażewicz, J. 2017. "Complexity of late work minimization in flow shop systems and a particle swarm optimization algorithm for learning effect." *Computers and Industrial Engineering*, *111*, 176–182.

Cheng, T. E., and Wang, G. 2000. "Single Machine Scheduling with Learning Effect Considerations." *Annals of Operations Research*, *98*(1–4), 273–290.

Cheng, T.C.E., Wu, C.-C., Chen, J.-C., Wu, W.-H., and Cheng, S.-R. 2013. "Two-machine flowshop scheduling with a truncated learning function to minimize the makespan." *International Journal of Production Economics*, *141*(1), 79–86.

Daniels, R. L., Mazzola, J. B., and Shi, D. 2004. "Flow shop scheduling with partial resource flexibility." *Management Science*, 50(5), 658–669.

Della Croce, F., Narayan, V., and Tadei, R. 1996. "The two-machine total completion time flow shop problem." *European Journal of Operational Research*, 90(2), 227-237.

Dessouky, M. I., Moray, N., and Kijowski, B. 1995. "Taxonomy of scheduling systems as a basis for the study of strategic behavior." *Human Factors*, 37(3), 443–472.

Dode, P., Greig, M., Zolfaghari, S., and Neumann, W. P. 2016. "Integrating human factors into discrete event simulation: a proactive approach to simultaneously design for system performance and employees' well-being." *International Journal of Production Research*, 54(10), 3105-3117.

Dolgui, A., Gordon, V., and Strusevich, V. 2012. "Single machine scheduling with precedence constraints and positionally dependent processing times." *Computers and Operations Research*, 39(6), 1218-1224

Dondeti, V. R., and Mohanty, B. B. 1998. "Impact of learning and fatigue factors on single machine scheduling with penalties for tardy jobs." *European Journal of Operational Research*, 105(3), 509-524.

Eren, T., and Güner, E. 2008. "A bicriteria flowshop scheduling with a learning effect." *Applied Mathematical Modelling*, *32*(9), 1719–1733.

European Commission. 2022. Industry 5.0 "What this approach is focused on, how it will be achieved and how it is already being implemented." Accessed February 9, 2023. https://research-and-innovation.ec.europa.eu/research-area/industrial-research-and-innovation/industry-50_en#what-is-industry-50

Fernandez-Viagas, V. 2022. "A speed-up procedure for the hybrid flow shop scheduling problem." *Expert Systems with Applications*, 187, 115903.

Fu, Y., Zhou, M., Guo, X., and Qi, L. 2019. "Artificial-Molecule-Based Chemical Reaction Optimization for Flow Shop Scheduling Problem with Deteriorating and Learning Effects." *IEEE Access*, *7*, 53429–53440.

Glock, C. H., Grosse, E. H., Jaber, M. Y., and Smunt, T. L. 2019. "Applications of

learning curves in production and operations management: A systematic literature review." *Computers and Industrial Engineering*, *131*(October 2018), 422–441.
Givi, Z. S., Jaber, M. Y., and Neumann, W. P. 2015. "Production planning in DRC systems considering worker performance." *Computers and Industrial Engineering*, 87, 317-327.
Hart, W. E., Watson, J. P., and Woodruff, D. L. 2011. "Pyomo: Modeling and solving mathematical programs in Python." *Mathematical Programming Computation*, 3(3), 219–260.
Hashemi-Petroodi, S. E., Dolgui, A., Kovalev, S., Kovalyov, M. Y. and Thevenin, S. 2020." Workforce reconfiguration strategies in manufacturing systems: a state of the art." *International Journal of Production Research*. 0, 1–25.
Henderson, D., Jacobson, S. H., and Johnson, A. W. 2003. "The theory and practice of simulated annealing." *In Handbook of metaheuristics* (pp. 287-319). Springer, Boston, MA.
Hosseini, N., and Tavakkoli-Moghaddam, R. 2013. "Two meta-heuristics for solving a new two-machine flowshop scheduling problem with the learning effect and dynamic arrivals." *International Journal of Advanced Manufacturing Technology*, *65*(5–8), 771–786.
Kadir, B. A., Broberg, O., and Da Conceicao, C. S. 2019. "Current research and future perspectives on human factors and ergonomics in Industry 4.0". *Computers and Industrial Engineering*, 137, 106004.
Katiraee, N., Calzavara, M., Finco, S., Battini, D., and Battaïa, O. 2021. "Consideration of workers' differences in production systems modelling and design: State of the art and directions for future research." *International Journal of Production Research*, 59(11), 3237-3268.
Katiraee, N., Calzavara, M., Finco, S., Battaïa, O., and Battini, D. 2022. "Assembly line balancing and worker assignment considering workers' expertise and perceived physical effort." *International Journal of Production Research*, 1-21
Kirkpatrick, A. S., Gelatt, C. D., and Vecchi, M. P. 1983. "Optimization by Simulated Annealing." *Science*, *220*(4598), 671–680.
Koulamas, C., and Kyparisis, G. J. 2007. "Single-machine and two-machine flowshop scheduling with general learning functions." *European Journal of Operational Research*, *178*(2), 402–407.
Kuo, W. H., and Yang, D. L. 2006a. "Minimizing the makespan in a single machine scheduling problem with a time-based learning effect." *Information Processing Letters*, *97*(2), 64–67.
Kuo, W. H., and Yang, D. L. 2006b. "Minimizing the total completion time in a single-machine scheduling problem with a time-dependent learning effect." *European Journal of Operational Research*, *174*(2), 1184–1190.
Kuo, W. H., and Yang, D. L. 2006c. "Single-machine group scheduling with a time-dependent learning effect." *Computers and Operations Research*, *33*(8), 2099–2112.
Lai, P.-J., and Wu, H.-C. 2015. "Using heuristic algorithms to solve the scheduling problems with job-dependent and machine-dependent learning effects." *Journal of Intelligent Manufacturing*, *26*(4), 691–701.
Lin, W. C. 2018. "Minimizing the makespan for a two-stage three-machine assembly flow shop problem with the sum-of-processing-time based learning effect." *Discrete Dynamics in Nature and Society*, 2018.
Liu, S.-C. 2020. "A Two-Stage Three-Machine Flow Shop Assembly Problem Mixed with a Controllable Number and Sum-of-Processing Times-Based Learning Effect

by Simulated Annealing Algorithms." *Discrete Dynamics in Nature and Society*, *2020*.

Lodree, E. J., Geiger, C. D., and Jiang, X. 2009. "Taxonomy for integrating scheduling theory and human factors: Review and research opportunities." *International Journal of Industrial Ergonomics*, *39*(1), 39–51.

Mosheiov, G., and Pruwer, S. 2021. "On the minmax common-due-date problem: extensions to position-dependent processing times, job rejection, learning effect, uniform machines and flowshops." *Engineering Optimization*, 53(3), 408-424.

Muştu, S., and Eren, T. 2018. "Maximum completion time under a learning effect in the permutation flowshop scheduling problem." *International Journal of Industrial Engineering: Theory Applications and Practice*, *25*(2), 156–174.

Nawaz, M., Enscore, E.E., and Ham, I., 1983. "A heuristic algorithm for the m- machine, n-job flow-shop sequencing problem." *OMEGA: The International Journal of Management Science* 11, 91–95.

Neumann, W. P., and Dul, J. 2010. "Human factors: Spanning the gap between OM and HRM." *International Journal of Operations and Production Management*, *30*(9), 923–950.

Ostermeier, F. F. 2020. "The impact of human consideration, schedule types and product mix on scheduling objectives for unpaced mixed-model assembly lines." *International Journal of Production Research*, 58(14), 4386-4405.

Paredes-Astudillo, Y.A., Montoya-Torres, J.R., and Botta-Genoulaz, V. 2022a. "Taxonomy of Scheduling Problems with Learning and Deterioration Effects." *Algorithms.* 15, 439.

Paredes-Astudillo, Y. A., Botta-Genoulaz, V., and Montoya-Torres, J. R. 2022b. "Comparing linear and non-linear modelling approaches of learning effects in 2-stage flow-shop scheduling problems." *IFAC-PapersOnLine*, 55(10), 842-847.

Pei, J., Zhou, Y., Yan, P., and Pardalos, P. M. 2022. "A concise guide to scheduling with learning and deteriorating effects." *International Journal of Production Research*, 1-22.

Pinedo, M. L. 2018. "Scheduling: Theory, Algorithms, and Systems." *In Operations research proceedings.* Springer (Vol. 1, Issue 1).

Rudek, A., and Rudek, R. 2013. "Makespan minimization flowshop with position dependent job processing times - Computational complexity and solution algorithms." *Computers and Operations Research*, *40*(8), 2071–2082.

Rudek, R. 2011. "Computational complexity and solution algorithms for flowshop scheduling problems with the learning effect." *Computers and Industrial Engineering*, *61*(1), 20–31.

Ruiz, R., and Maroto, C. 2005. "A comprehensive review and evaluation of permutation flowshop heuristics." *European journal of operational research*, 165(2), 479-494.

Sánchez-Herrera, S., Montoya-Torres, J. R., and Solano-Charris, E. L. 2019. "Flow shop scheduling problem with position-dependent processing times." *Computers and Operations Research*, 111, 325-345.

Sun, X., Geng, X.-N., and Liu, F. 2020. "Flow shop scheduling with general position weighted learning effects to minimise total weighted completion time." *Journal of the Operational Research Society*.

Taillard, E. 1993. "Benchmarks for basic scheduling problems." *European journal of operational research*, 64(2), 278-285.

Turner, S., and Booth, D. 1987. "Comparison of heuristics for flow shop sequencing." *Omega*, 15(1), 75-78.

Vahedi Nouri, B., Fattahi, P., and Ramezanian, R. 2013. "Hybrid firefly-simulated annealing algorithm for the flow shop problem with learning effects and flexible maintenance activities." *International Journal of Production Research*, *51*(12), 3501–3515.

Vanheusden, S., van Gils, T., Ramaekers, K., Cornelissens, T., and Caris, A. 2022. "Practical factors in order picking planning: state-of-the-art classification and review." *International Journal of Production Research*, 1-25.

Vijayakumar, V., Sgarbossa, F., Neumann, W. P., and Sobhani, A. 2022. "Framework for incorporating human factors into production and logistics systems." *International Journal of Production Research*, 60(2), 402-419.

Wang, J. B., and Xia, Z. Q. 2005. "Flow-shop scheduling with a learning effect." *Journal of the Operational Research Society*, 56(11), 1325-1330.

Wang, X. Y., Zhou, Z., Zhang, X., Ji, P., and Wang, J. B. 2013. "Several flow shop scheduling problems with truncated position-based learning effect." *Computers and Operations Research*, *40*(12), 2906–2929.

Wang, J. B., and Wang, J. J. 2014. "Flowshop scheduling with a general exponential learning effect." *Computers and Operations Research*, 43, 292-308.

Wright, T. P. 1936. "Factors affecting the cost of airplanes." *Journal of the Aeronautical Sciences*, 3(4), 122-128.

Wu, C. C., Liu, S. C., Cheng, T. C. E., Cheng, Y., Liu, S. Y., and Lin, W. C. 2018a. "Re-entrant flowshop scheduling with learning considerations to minimize the makespan" *Iranian Journal of Science and Technology, Transactions A: Science*, 42(2), 727-744.

Wu, C. C., Wang, D. J., Cheng, S. R., Chung, I. H., and Lin, W. C. 2018b. "A two-stage three-machine assembly scheduling problem with a position-based learning effect." *International Journal of Production Research*, 56(9), 3064-3079.

Wu, C.-C., Wu, W.-H., Hsu, P.-H., and Lai, K. 2012a. "A two-machine flowshop scheduling problem with a truncated sum of processing-times-based learning function." *Applied Mathematical Modelling*, *36*(10), 5001–5014.

Wu, C.-C., Yin, Y., and Cheng, S.-R. 2011. Some single-machine scheduling problems with a truncation learning effect. *Computers and Industrial Engineering*, *60*(4), 790–795

Wu, C.-C., Yin, Y., Wu, W.-H., and Cheng, S.-R. 2012b. "Some polynomial solvable single-machine scheduling problems with a truncation sum-of-processing-times based learning effect." *European Journal of Industrial Engineering*, *6*(4), 441–453.

Wu, C.-C., Zhang, X., Azzouz, A., Shen, W.-L., Cheng, S.-R., Hsu, P.-H., and Lin, W.-C. 2020. "Metaheuristics for two-stage flow-shop assembly problem with a truncation learning function." *Engineering Optimization*.

Wu, W.-H., Yin, Y., Cheng, S.-R., Hsu, P.-H., and Wu, C.-C. 2014. "Genetic algorithm for a two-agent scheduling problem with truncated learning consideration." *Asia-Pacific Journal of Operational Research*, *31*(6).

Zou, Y., Wang, D., Lin, W.-C., Chen, J.-Y., Yu, P.-W., Wu, W.-H., Chao, Y.-P., and Wu, C.-C. 2020. "Two-stage three-machine assembly scheduling problem with sum-of-processing-times-based learning effect." *Soft Computing*, *24*(7), 5445–5462.

Table 4. Synthesis of current works

Author and Year	**Metaheuristics**	**Objective function (minimization)**	**Learning effect model**
Arık (2021)	TSPOP	makespan	P-LE
Azizi, Jabbari, and Kheirkhah (2016)	GA, SA	makespan	Other
Chen, Wu, and Lee. (2006)	SA	weighted sum of the total completion time & maximum tardiness	P-LE
Eren and Güner (2008)	TS	weighted sum of total completion time & makespan	P-LE
Fu et al. (2019)	ACRO	makespan	P-LE
Lai and Wu (2015)	GA, SA, ACO, PSO	makespan	P-LE
Liu (2020)	SA	makespan	TSM-LE
Muştu and Eren (2018)	KA, GAKA	makespan	P-LE
Sun, Geng and Liu (2020)	SA	total weighted completion time	P-LE
Hosseini and Tavakkoli-Moghaddam (2013)	MOGA, MOSA	total idle time & mean deviation from a common due date	P-LE
Vahedi Nouri, Fattahi, P and Ramezanian (2013)	HFSA	makespan	P-LE
Wu et al. (2018a)	SA	makespan	SM-LE
Wu et al. (2018b)	SA, CSA	makespan	P-LE
Wu et al. (2020)	GA	makespan	TSM-LE
Wu et al. (2012a)	GA	maximum lateness & total weighted completion time	SM-LE
Zou et al. (2020)	GA, CSA	makespan	SM-LE

Position-based learning effect (P-LE); Truncated position-based learning effect (TP-LE), Sum-of-processing-time based learning (SM-LE); Truncated sum-of-processing-time based learning (TSM-LE)

Table 5. Summary of set of problem instances

<table>
<tr><th colspan="2"></th><th>Case 1 and Case 2</th><th>Case 3 and Case 4</th></tr>
<tr><td rowspan="4">Small-size instances</td><td>Number of workers (I)</td><td colspan="2">2</td></tr>
<tr><td>Number of jobs (J)</td><td colspan="2">5, 7</td></tr>
<tr><td>Learning index - alpha (α)</td><td colspan="2">– 0.152, –0.322, – 0.515</td></tr>
<tr><td>Truncation parameter - beta (β)</td><td>-</td><td>0.25, 0.5, 0.75</td></tr>
<tr><td rowspan="4">Large-size instances</td><td>Number of workers (I)</td><td colspan="2">5, 10</td></tr>
<tr><td>Number of jobs (J)</td><td colspan="2">50, 100</td></tr>
<tr><td>Learning index - alpha (α)</td><td colspan="2">– 0.152, –0.322, – 0.515</td></tr>
<tr><td>Truncation parameter - beta (β)</td><td>-</td><td>0.25, 0.5, 0.75</td></tr>
</table>

Table 6. Comparison for small instance problems

Case	***I***	***J***	***α***	***β***	$OPT - SA_{API}$			$OPT - SA_{NAPI}$			**CPU time** SA_{API} **(sec)**			**CPU time** SA_{NAPI} **(sec)**			**CPU time OPT (sec)**		
					Mean		**Std**	**Mean**		**Std**	**Mean**		**Std**	**Mean**		**Std**	**Mean**		**Std**
C1	2	5	-0.152	-	0.02%	±	0.06%	0.02%	±	0.07%	0.02	±	0.02	0.05	±	0.04	0.16	±	0.02
			-0.322	-	0.15%	±	0.37%	0.21%	±	0.41%	0.03	±	0.02	0.04	±	0.04	0.15	±	0.01
			-0.515	-	0.27%	±	0.62%	0.51%	±	1.00%	0.03	±	0.02	0.05	±	0.07	0.17	±	0.01
		7	-0.152	-	0.03%	±	0.12%	0.04%	±	0.09%	0.05	±	0.05	0.11	±	0.09	2.32	±	0.05
			-0.322	-	0.07%	±	0.28%	0.21%	±	0.49%	0.06	±	0.05	0.10	±	0.09	2.26	±	0.09
			-0.515	-	0.11%	±	0.28%	0.37%	±	0.90%	0.06	±	0.10	0.10	±	0.08	2.10	±	0.11
C2	2	5	-0.152	0.25	0.02%	±	0.06%	0.03%	±	0.08%	0.03	±	0.02	0.05	±	0.04	0.15	±	0.01
			-0.152	0.5	0.02%	±	0.06%	0.02%	±	0.08%	0.03	±	0.02	0.05	±	0.04	0.15	±	0.00
			-0.152	0.75	0.02%	±	0.06%	0.02%	±	0.07%	0.03	±	0.02	0.05	±	0.04	0.15	±	0.01
			-0.322	0.25	0.13%	±	0.34%	0.22%	±	0.42%	0.02	±	0.02	0.05	±	0.04	0.15	±	0.01
			-0.322	0.5	0.12%	±	0.34%	0.21%	±	0.41%	0.02	±	0.02	0.04	±	0.04	0.14	±	0.01
			-0.322	0.75	0.00%	±	0.00%	0.01%	±	0.13%	0.02	±	0.02	0.03	±	0.03	0.15	±	0.01
			-0.515	0.25	0.31%	±	0.69%	0.46%	±	0.94%	0.03	±	0.02	0.04	±	0.04	0.14	±	0.01
			-0.515	0.5	0.72%	±	2.12%	0.47%	±	1.58%	0.02	±	0.02	0.04	±	0.03	0.15	±	0.01
			-0.515	0.75	0.01%	±	0.12%	0.01%	±	0.12%	0.03	±	0.02	0.03	±	0.02	0.15	±	0.01
		7	-0.152	0.25	0.03%	±	0.09%	0.04%	±	0.09%	0.05	±	0.04	0.13	±	0.16	2.32	±	0.06
			-0.152	0.5	0.02%	±	0.08%	0.04%	±	0.10%	0.05	±	0.04	0.13	±	0.22	2.37	±	0.07
			-0.152	0.75	0.02%	±	0.09%	0.04%	±	0.12%	0.05	±	0.04	0.11	±	0.09	2.31	±	0.06
			-0.322	0.25	0.07%	±	0.31%	0.22%	±	0.48%	0.05	±	0.04	0.10	±	0.09	2.25	±	0.09
			-0.322	0.5	0.07%	±	0.27%	0.19%	±	0.40%	0.05	±	0.04	0.10	±	0.09	2.25	±	0.06
			-0.322	0.75	0.02%	±	0.03%	0.02%	±	0.03%	0.03	±	0.03	0.04	±	0.03	2.29	±	0.06
			-0.515	0.25	0.08%	±	0.19%	0.28%	±	0.67%	0.05	±	0.04	0.10	±	0.09	2.10	±	0.10
			-0.515	0.5	0.11%	±	0.44%	0.31%	±	1.11%	0.04	±	0.03	0.06	±	0.05	2.17	±	0.07
			-0.515	0.75	0.00%	±	0.04%	0.01%	±	0.09%	0.03	±	0.04	0.03	±	0.02	2.33	±	0.07
C3	2	5	-0.152	-	0.00%	±	0.00%	0.00%	±	0.01%	0.03	±	0.03	0.07	±	0.05	18.48	±	3.53
			-0.322	-	0.06%	±	0.22%	0.06%	±	0.20%	0.04	±	0.03	0.07	±	0.06	19.12	±	3.05
			-0.515	-	0.01%	±	0.03%	0.06%	±	0.41%	0.04	±	0.03	0.07	±	0.06	19.33	±	2.88
		7	-0.152	-	0.00%	±	0.00%	0.00%	±	0.01%	0.08	±	0.06	0.18	±	0.15	1580.88	±	22.66
			-0.322	-	0.00%	±	0.00%	0.00%	±	0.02%	0.08	±	0.06	0.17	±	0.14	1591.90	±	48.29
			-0.515	-	0.00%	±	0.01%	0.07%	±	0.21%	0.08	±	0.07	0.17	±	0.14	1586.26	±	98.26
C4	2	5	-0.152	0.25	0.01%	±	0.01%	0.01%	±	0.01%	0.07	±	0.06	0.12	±	0.10	20.60	±	3.48
			-0.152	0.5	0.01%	±	0.01%	0.01%	±	0.01%	0.07	±	0.06	0.12	±	0.10	22.32	±	2.74
			-0.152	0.75	0.01%	±	0.01%	0.01%	±	0.01%	0.07	±	0.06	0.12	±	0.09	23.76	±	3.77
			-0.322	0.25	0.11%	±	0.22%	0.09%	±	0.19%	0.08	±	0.06	0.12	±	0.10	24.16	±	4.15
			-0.322	0.5	0.09%	±	0.21%	0.10%	±	0.21%	0.08	±	0.06	0.12	±	0.10	25.23	±	7.50
			-0.322	0.75	0.00%	±	0.00%	0.01%	±	0.05%	0.06	±	0.05	0.10	±	0.09	25.53	±	5.07
			-0.515	0.25	0.31%	±	0.70%	0.34%	±	0.73%	0.07	±	0.06	0.12	±	0.10	22.87	±	3.06
			-0.515	0.5	0.31%	±	0.70%	0.33%	±	0.71%	0.07	±	0.06	0.12	±	0.10	23.78	±	2.67
			-0.515	0.75	0.00%	±	0.00%	0.00%	±	0.05%	0.06	±	0.05	0.09	±	0.07	27.36	±	8.29
		7	-0.152	0.25	0.02%	±	0.04%	0.02%	±	0.04%	0.16	±	0.14	0.36	±	0.30	2570.01	±	919.55
			-0.152	0.5	0.02%	±	0.04%	0.02%	±	0.04%	0.16	±	0.15	0.36	±	0.30	2172.26	±	209.46
			-0.152	0.75	0.02%	±	0.04%	0.02%	±	0.04%	0.16	±	0.13	0.36	±	0.29	2457.82	±	724.04
			-0.322	0.25	0.04%	±	0.08%	0.04%	±	0.08%	0.16	±	0.13	0.35	±	0.29	2494.90	±	725.52
			-0.322	0.5	0.04%	±	0.08%	0.04%	±	0.08%	0.16	±	0.13	0.35	±	0.29	2207.21	±	241.34
			-0.322	0.75	0.00%	±	0.02%	0.00%	±	0.03%	0.11	±	0.10	0.19	±	0.18	2292.46	±	440.64
			-0.515	0.25	0.05%	±	0.09%	0.10%	±	0.24%	0.17	±	0.14	0.34	±	0.28	2385.95	±	436.48
			-0.515	0.5	0.01%	±	0.03%	0.03%	±	0.05%	0.14	±	0.12	0.33	±	0.27	2496.02	±	994.31
			-0.515	0.75	0.00%	±	0.02%	0.01%	±	0.04%	0.10	±	0.09	0.13	±	0.13	2069.05	±	222.70

Table 4. Summary experiment

	Parameter	**Levels**	**Values**
Factors	Local search operator	2	API, NAPI
	Initial temperature – (T_0)	3	0.5, 0.3, 0.1
	Final temperature – (T_f)	3	0.001, 0.0001, 0.00001
	Cooling velocity – (λ)	2	0.9, 0.5
Total of treatments			(2*3*3*2) = 36
Total of combinations			144
Total instances per combination			(144*10) =1440
Total instances per combination x treatments			(1440 * 36) = 51840
Size of the experiment			(51840 * 3) = 155520

Table 5. Comparison for large instances problem

Case	I	J	α	β	$\Delta SA_{API} - NEH$			$\Delta SA_{NAPI} - NEH$			CPU time SA_{API} (sec)			CPU time SAN_{NAPI} (sec)		
					Mean		Std	Mean		Std	Mean		Std	Mean		Std
C1	10	100	-0.152	-	0.55%	±	0.39%	0.43%	±	0.38%	29.82	±	19.61	255.58	±	247.18
			-0.322	-	1.25%	±	0.74%	1.06%	±	0.68%	27.57	±	18.60	199.49	±	204.12
			-0.515	-	1.92%	±	1.01%	1.48%	±	0.92%	20.95	±	10.12	136.46	±	138.62
		50	-0.152	-	1.06%	±	0.68%	0.82%	±	0.57%	12.53	±	13.25	53.82	±	50.18
			-0.322	-	2.14%	±	1.40%	1.68%	±	1.15%	9.20	±	9.53	44.19	±	44.06
			-0.515	-	3.43%	±	2.10%	2.80%	±	1.84%	6.19	±	5.87	30.83	±	31.74
	5	100	-0.152	-	0.47%	±	0.30%	0.41%	±	0.28%	13.77	±	7.58	116.41	±	119.11
			-0.322	-	1.41%	±	0.78%	1.14%	±	0.70%	12.76	±	6.24	101.79	±	101.48
			-0.515	-	2.98%	±	1.35%	2.28%	±	1.19%	11.15	±	4.60	74.33	±	74.41
		50	-0.152	-	0.73%	±	0.41%	0.62%	±	0.36%	4.29	±	4.22	24.77	±	24.69
			-0.322	-	1.56%	±	0.99%	1.22%	±	0.83%	4.23	±	4.21	23.37	±	23.13
			-0.515	-	3.19%	±	1.43%	2.49%	±	1.26%	3.26	±	2.78	18.47	±	19.29
C2	10	100	-0.152	0.25	0.50%	±	0.35%	0.35%	±	0.32%	29.71	±	12.54	142.74	±	87.91
			-0.152	0.5	0.51%	±	0.35%	0.36%	±	0.30%	29.34	±	12.21	140.76	±	89.08
			-0.152	0.75	0.37%	±	0.37%	0.15%	±	0.21%	54.32	±	36.32	72.82	±	54.85
			-0.322	0.25	1.06%	±	0.60%	0.79%	±	0.55%	27.02	±	12.36	107.23	±	81.50
			-0.322	0.5	0.68%	±	0.48%	0.38%	±	0.37%	46.69	±	33.39	59.46	±	44.12
			-0.322	0.75	0.54%	±	0.39%	0.23%	±	0.28%	61.12	±	40.76	86.40	±	68.92
			-0.515	0.25	1.12%	±	0.77%	0.47%	±	0.69%	38.10	±	30.91	39.40	±	23.06
			-0.515	0.5	0.63%	±	0.57%	0.22%	±	0.36%	58.46	±	42.07	61.50	±	45.99
			-0.515	0.75	0.47%	±	0.35%	0.19%	±	0.22%	65.50	±	44.62	83.53	±	64.26
		50	-0.152	0.25	0.99%	±	0.63%	0.76%	±	0.53%	8.45	±	5.36	36.34	±	24.32
			-0.152	0.5	1.00%	±	0.63%	0.72%	±	0.51%	8.41	±	5.27	36.34	±	24.05
			-0.152	0.75	0.86%	±	0.65%	0.46%	±	0.51%	12.50	±	7.51	26.69	±	21.39
			-0.322	0.25	2.12%	±	1.43%	1.61%	±	1.11%	8.05	±	5.66	33.53	±	25.48
			-0.322	0.5	1.90%	±	0.90%	1.33%	±	0.83%	9.97	±	7.46	21.05	±	17.89
			-0.322	0.75	1.29%	±	0.66%	0.67%	±	0.54%	14.77	±	10.32	28.16	±	21.86
			-0.515	0.25	2.72%	±	1.81%	1.80%	±	1.81%	7.75	±	7.31	9.62	±	6.87
			-0.515	0.5	2.08%	±	1.04%	1.48%	±	1.01%	11.26	±	9.05	19.04	±	15.84
			-0.515	0.75	1.44%	±	1.07%	0.82%	±	0.88%	16.22	±	11.31	27.24	±	21.50
	5	100	-0.152	0.25	0.44%	±	0.27%	0.36%	±	0.25%	14.75	±	5.80	82.86	±	59.67
			-0.152	0.5	0.45%	±	0.28%	0.37%	±	0.25%	14.68	±	5.77	82.37	±	59.31
			-0.152	0.75	0.17%	±	0.26%	0.05%	±	0.12%	42.32	±	29.65	23.93	±	12.26
			-0.322	0.25	1.27%	±	0.71%	0.91%	±	0.62%	13.52	±	5.43	64.83	±	50.93
			-0.322	0.5	0.41%	±	0.28%	0.20%	±	0.23%	25.80	±	20.57	24.07	±	12.48
			-0.322	0.75	0.24%	±	0.29%	0.08%	±	0.14%	38.04	±	30.00	23.97	±	12.51
			-0.515	0.25	1.27%	±	0.78%	0.80%	±	0.83%	18.10	±	13.82	23.11	±	11.57
			-0.515	0.5	0.58%	±	0.51%	0.22%	±	0.34%	33.80	±	28.03	23.75	±	11.98
			-0.515	0.75	0.24%	±	0.25%	0.05%	±	0.09%	42.84	±	32.28	24.05	±	12.53
		50	-0.152	0.25	0.70%	±	0.39%	0.58%	±	0.34%	4.43	±	2.92	19.86	±	15.04
			-0.152	0.5	0.69%	±	0.38%	0.57%	±	0.32%	4.43	±	2.93	19.99	±	14.99
			-0.152	0.75	0.29%	±	0.29%	0.12%	±	0.19%	10.08	±	7.19	5.94	±	4.06
			-0.322	0.25	1.51%	±	0.99%	1.12%	±	0.76%	4.51	±	3.07	19.55	±	14.86
			-0.322	0.5	0.68%	±	0.45%	0.16%	±	0.21%	9.39	±	7.35	5.33	±	3.34
			-0.322	0.75	0.47%	±	0.41%	0.18%	±	0.28%	10.85	±	8.33	5.54	±	3.85
			-0.515	0.25	2.37%	±	1.22%	1.27%	±	1.05%	5.78	±	5.99	5.18	±	3.33
			-0.515	0.5	0.57%	±	0.47%	0.32%	±	0.41%	9.03	±	7.63	5.25	±	3.32
			-0.515	0.75	0.56%	±	0.44%	0.28%	±	0.36%	10.29	±	8.30	5.62	±	3.81
C3	10	100	-0.152	-	0.29%	±	0.27%	0.23%	±	0.25%	36.59	±	18.97	224.76	±	149.59
			-0.322	-	0.56%	±	0.56%	0.35%	±	0.41%	32.89	±	16.50	214.97	±	153.84
			-0.515	-	1.11%	±	1.00%	1.07%	±	0.96%	27.36	±	9.88	177.34	±	143.78
		50	-0.152	-	0.73%	±	0.58%	0.57%	±	0.51%	16.07	±	17.74	76.51	±	65.82
			-0.322	-	1.03%	±	0.83%	0.81%	±	0.67%	10.72	±	10.63	63.96	±	56.64
			-0.515	-	1.37%	±	0.85%	1.21%	±	0.88%	10.29	±	10.56	56.34	±	54.35
	5	100	-0.152	-	0.11%	±	0.18%	0.09%	±	0.14%	19.98	±	8.64	127.00	±	112.24
			-0.322	-	0.25%	±	0.40%	0.12%	±	0.21%	18.64	±	6.79	139.82	±	117.37
			-0.515	-	0.42%	±	0.61%	0.37%	±	0.60%	18.79	±	7.52	136.64	±	117.23
		50	-0.152	-	0.25%	±	0.28%	0.15%	±	0.19%	5.28	±	4.90	36.44	±	37.26
			-0.322	-	0.47%	±	0.57%	0.35%	±	0.50%	4.96	±	4.21	37.63	±	37.84
			-0.515	-	0.62%	±	0.79%	0.49%	±	0.63%	5.01	±	4.33	34.13	±	35.02
C4	10	100	-0.152	0.25	0.27%	±	0.25%	0.23%	±	0.24%	48.44	±	17.42	270.22	±	162.62
			-0.152	0.5	0.29%	±	0.26%	0.22%	±	0.23%	48.33	±	17.21	271.38	±	164.23
			-0.152	0.75	0.34%	±	0.36%	0.14%	±	0.22%	82.48	±	43.82	155.47	±	115.06
			-0.322	0.25	0.55%	±	0.53%	0.35%	±	0.42%	47.10	±	18.13	256.40	±	170.64
			-0.322	0.5	0.43%	±	0.51%	0.25%	±	0.33%	71.07	±	45.40	134.68	±	105.81
			-0.322	0.75	0.39%	±	0.36%	0.17%	±	0.22%	98.19	±	58.10	154.63	±	116.75

Case	I	J	α	β	$\Delta SA_{API} - NEH$			$\Delta SA_{NAPI} - NEH$			CPU time SA_{API} (sec)			CPU time SAN_{NAPI} (sec)		
					Mean		**Std**	**Mean**		**Std**	**Mean**		**Std**	**Mean**		**Std**
			-0.515	0.25	0.77%	±	0.91%	0.47%	±	0.69%	42.80	±	18.59	119.17	±	82.63
			-0.515	0.5	0.52%	±	0.60%	0.31%	±	0.54%	81.85	±	53.20	119.77	±	86.46
			-0.515	0.75	0.49%	±	0.44%	0.21%	±	0.28%	99.01	±	62.21	160.01	±	122.57
			-0.152	0.25	0.66%	±	0.54%	0.50%	±	0.44%	13.43	±	7.61	66.44	±	39.02
			-0.152	0.5	0.68%	±	0.53%	0.51%	±	0.45%	13.12	±	7.49	66.32	±	39.17
			-0.152	0.75	1.01%	±	0.78%	0.59%	±	0.58%	17.19	±	10.74	46.96	±	36.02
			-0.322	0.25	1.01%	±	0.80%	0.77%	±	0.62%	12.15	±	7.82	60.74	±	41.52
		50	-0.322	0.5	1.10%	±	0.82%	0.77%	±	0.68%	15.29	±	11.20	37.65	±	29.96
			-0.322	0.75	0.88%	±	0.70%	0.47%	±	0.53%	23.87	±	14.81	48.79	±	36.04
			-0.515	0.25	1.61%	±	0.98%	1.30%	±	0.94%	12.35	±	8.61	54.03	±	42.13
			-0.515	0.5	1.54%	±	0.83%	1.10%	±	0.79%	19.84	±	13.63	39.12	±	30.65
			-0.515	0.75	1.13%	±	0.97%	0.57%	±	0.67%	23.51	±	15.39	44.51	±	34.54
	5		-0.152	0.25	0.12%	±	0.18%	0.09%	±	0.15%	33.57	±	11.68	168.48	±	127.62
			-0.152	0.5	0.11%	±	0.17%	0.09%	±	0.13%	33.73	±	11.99	168.45	±	128.98
			-0.152	0.75	0.05%	±	0.06%	0.03%	±	0.04%	79.31	±	53.00	65.13	±	36.41
			-0.322	0.25	0.24%	±	0.38%	0.12%	±	0.21%	32.40	±	10.84	186.74	±	132.94
		100	-0.322	0.5	0.11%	±	0.20%	0.06%	±	0.13%	49.40	±	37.95	63.46	±	34.01
			-0.322	0.75	0.10%	±	0.10%	0.04%	±	0.07%	86.04	±	67.31	56.53	±	27.82
			-0.515	0.25	0.37%	±	0.50%	0.23%	±	0.40%	31.09	±	10.78	95.11	±	63.07
			-0.515	0.5	0.22%	±	0.32%	0.10%	±	0.17%	58.28	±	44.58	58.42	±	28.61
			-0.515	0.75	0.15%	±	0.17%	0.05%	±	0.11%	88.55	±	65.61	57.29	±	27.50
			-0.152	0.25	0.26%	±	0.31%	0.14%	±	0.18%	7.91	±	5.13	43.49	±	33.79
			-0.152	0.5	0.27%	±	0.30%	0.15%	±	0.19%	8.05	±	5.30	43.91	±	33.96
			-0.152	0.75	0.22%	±	0.27%	0.09%	±	0.17%	17.21	±	12.99	15.30	±	10.83
			-0.322	0.25	0.52%	±	0.64%	0.34%	±	0.50%	7.80	±	5.16	44.45	±	33.57
		50	-0.322	0.5	0.28%	±	0.36%	0.16%	±	0.27%	11.27	±	9.60	17.70	±	12.88
			-0.322	0.75	0.33%	±	0.26%	0.17%	±	0.23%	22.24	±	17.42	14.34	±	10.16
			-0.515	0.25	0.58%	±	0.80%	0.32%	±	0.42%	8.34	±	5.53	42.69	±	32.98
			-0.515	0.5	0.47%	±	0.49%	0.19%	±	0.30%	19.11	±	14.71	13.42	±	8.97
			-0.515	0.75	0.32%	±	0.34%	0.17%	±	0.29%	22.08	±	17.68	13.06	±	8.93

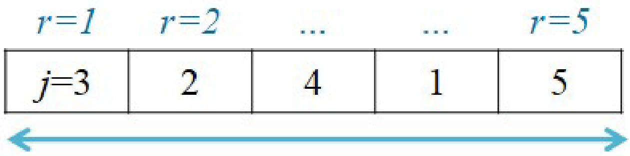


Figure 1

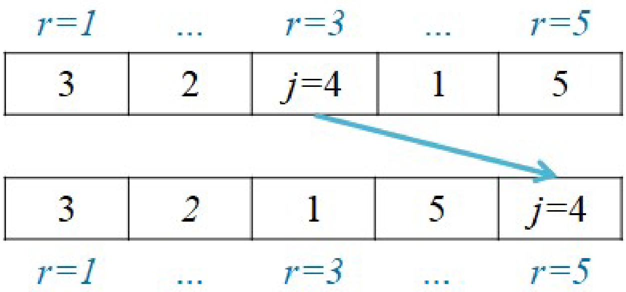


Figure 2

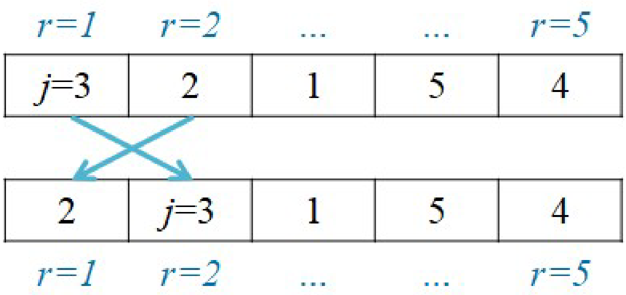


Figure 3

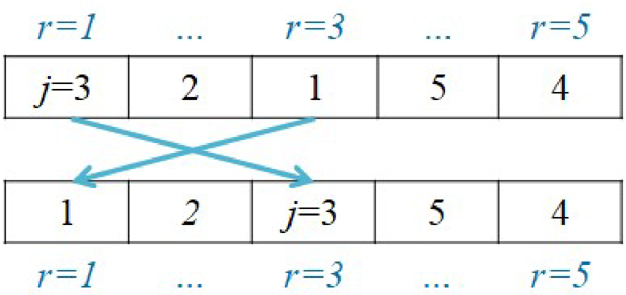
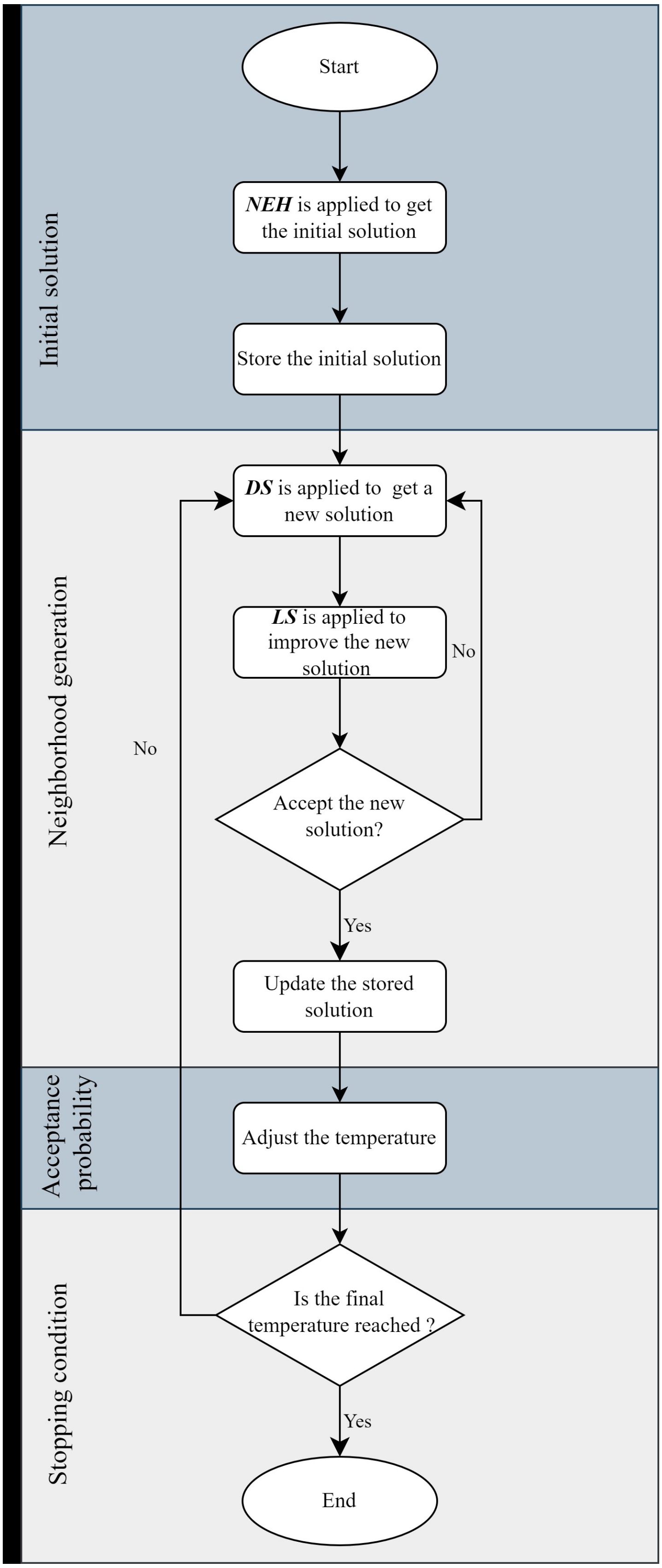


Figure 4



Initial solution
Start
NEH is applied to get the initial solution
Store the initial solution
Neighborhood generation
DS is applied to get a new solution
LS is applied to improve the new solution
No
No
Accept the new solution?
Yes
Update the stored solution
Acceptance probability
Adjust the temperature
Stopping condition
Is the final temperature reached ?
Yes
End

Figure 5

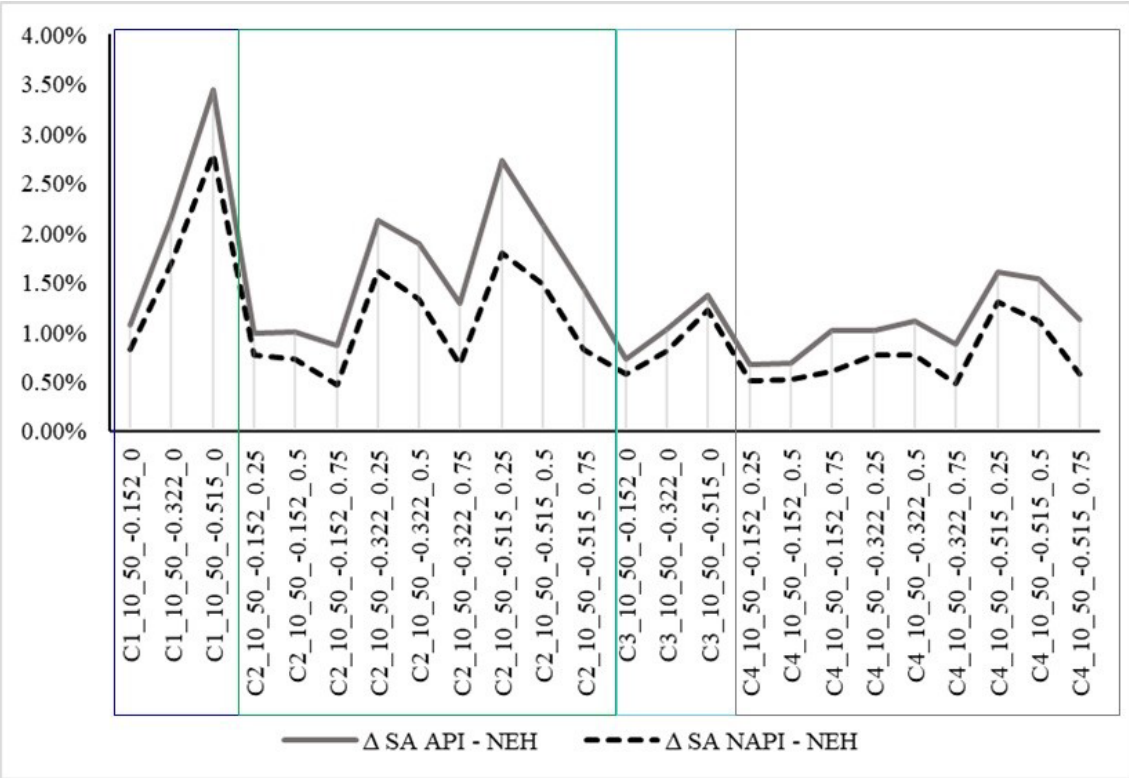


Figure 6

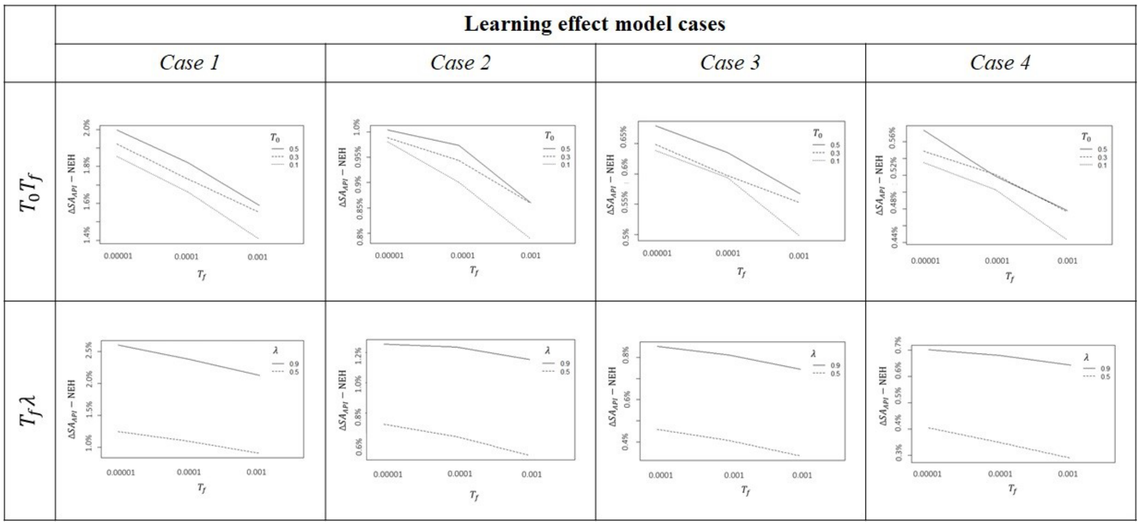


Figure 7