International Journal of Industrial Engineering Computations 13 (2022) 223–236

Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Minimizing total tardiness for the order scheduling problem with sequence-dependent setup times using hybrid matheuristics

Massimo Pinto Antonioli[a], Carlos Diego Rodrigues[a] and Bruno de Athayde Prata[a*]

[a]*Federal University of Ceará, Fortaleza, Brazil*

C H R O N I C L E

Article history:
Received May 11 2021
Received in Revised Format June 28 2021
Accepted October 22 2021
Available online October, 27 2021

Keywords:
Production Scheduling
Matheuristics
Mixed-Integer Linear Programming

A B S T R A C T

This paper aims at presenting a customer order scheduling environment in which the setup times are explicit and depend on the production sequence. The considered objective function is the total tardiness minimization. Since the variant under study is NP-hard, we propose a mixed-integer linear programming (MILP) model, an adaptation of the Order-Scheduling Modified Due-Date heuristic (OMDD) (referred to as Order-Scheduling Modified Due-Date Setup (OMMD-S)), an adaptation of the Framinan and Perez-Gonzalez heuristic (FP) (hereinafter referred to as Framinan and Perez-Gonzalez Setup (FP-S)), a matheuristic with Same Permutation in All Machines (SPAM), and the hybrid matheuristic SPAM-SJPO based on Job-Position Oscillation (JPO). The algorithms under comparison have been compared on an extensive benchmark of randomly generated test instances, considering two performance measures: Relative Deviation Index (*RDI*) and Success Rate (SR). For the small-size evaluated instances, the SPAM is the most efficient algorithm, presenting the better values of RDI and SR. For the large-size evaluated instances, the hybrid matheuristic SPAM-JPO and MILP model are the most efficient methods.

© 2022 by the authors; licensee Growing Science, Canada

1. Introduction

Consumer demand, combined with global competition among firms, has been introducing new production paradigms in a crescent quest for quality, resulting in the allocation of manufacturing of components of specialized goods to several plants or production environments (Fernandez-Viagas & Framinan, 2015). As a result, in recent years, production scheduling researchers have focused on the assembly scheduling problems, such as the customer order scheduling environment (Framinan et al., 2018). The customer order scheduling problem (COSP) is defined as n orders to be scheduled on m dedicated parallel machines, where each machine can process only one type of operation. The COSP presents several real-world applications, such as paper production, pharmaceutical industry, assembly operations, and in the repair of airplanes and vessels (Magazine and Julien, 1990; Leung et al., 2005b; Roemer, 2006).

Usually, the setup times are embedded in the processing times in the current literature on customer order scheduling (Prata et al., 2021a). Nevertheless, this may not be a realistic assumption in many practical situations since the machines allow for some versatility in handling various items, which typically necessitates some set-up operations. In our proposal, each order presents an associated setup time, which is sequence-dependent, as well as a due date, related to the customer requirements.

Based on recent studies (Framinan et al., 2019, Wu et al., 2021, Prata et al., 2021a, Prata et al., 2021b), we can observe that the variant introduced here is not reported previously. This paper aims at presenting the COSP with sequence-dependent setup times to minimize the total tardiness. We develop a mixed-integer linear programming (MILP) model, as well as explore some problem properties. As a solution procedure, we extend two existing constructive heuristics. Also, we present a matheuristic based on the reduction of the number of decision variables. Finally, we carried out extensive computational experiments with random-generated test instances.

* Corresponding author
E-mail: baprata@ufc.br (B. D. A. Prata)

2022 Growing Science Ltd.
doi: 10.5267/j.ijiec.2021.11.002

The remaining sections of this paper are structured as follows: Section 2 addresses some related approaches, Section 3 describes the problem under study, Section 4 presents the proposed solution approaches, Section 5 summarizes the computational results. Finally, Section 6 addresses the final remarks as well as suggestions for future works.

2. Literature Review

Since the considered problem has not been previously reported, we are looking for related approaches. The presented studies are classified based on the notation provided by Framinan et al. (2018). Wagneur and Sriskandarajar (1993) presented the COSP as an open shop scheduling problem in which the overlap of jobs in the available machines is allowed. Furthermore, the order scheduling to minimize the total tardiness performance measure $DPm \rightarrow 0 \mid\mid \Sigma T$ is classified as NP-hard. A notation for the classification of order scheduling problems is introduced by Leung et al. (2005a). Roemer (2006) addressed a concurrent open shop in which the jobs are scheduled in dedicated parallel machines. Leung et al. (2005b) addressed COSPs with release and due dates. The performance measures f to be minimized are total completion time and total tardiness. Several structural properties and complexity analyses are addressed. In addition, two constructive heuristics, as well as a tabu search metaheuristic, are proposed. Leung et al. (2006) addressed two variants of the COSP two objective functions: the minimization of the maximum lateness and the total number of late orders. A constructive heuristic and an exact method are developed as solution approaches. Lee (2013) addressed an important property for $DPm \rightarrow 0 \mid\mid \Sigma T$: the global optimal solution always can be found with fixed permutations in the available machines. In addition, a MILP model, a Branch-and-Bound (B&B) algorithm, and the Order-Scheduling Modified Due-Date heuristic (OMDD) are presented. Framinan and Perez-Gonzalez (2017) approached the COSP for the minimization of the total completion time. An algorithm using a look-ahead mechanism is developed, which is employed as an initial solution for a Greedy Search Algorithm (GSA) metaheuristic (Karabulut, 2016). Framinan and Perez-Gonzalez (2018) extended this heuristic for total tardiness minimization. Furthermore, the matheuristics JPF and JPO are introduced. The second one is the state-of-the-art algorithm for the $DPm \rightarrow 0 \mid\mid \Sigma T$. Kung et al. (2018) studied a COSP minimizing the total completion time objective with unequal ready times. As theoretical contributions, several problem properties are addressed, as well as two lower bounds. With respect to solution procedures, four simulated-annealing-based and four genetic-algorithm-based metaheuristics are presented, as well as a B&B exact method. Riahi et al. (2019) studied the $DPm \rightarrow 0 \mid\mid \Sigma C$ variant. Several priority rules are developed, as well as a new constructive heuristic that considers all feasible positions in the partial permutation for not scheduled orders. A Perturbative Search Algorithm (PSA) metaheuristic is proposed, outperforming the GSA metaheuristics presented by Framinan and Perez-Gonzalez (2017). Lin et al. (2019) studied a COSP minimizing the weighted number of tardy orders. Some theoretical contributions are addressed, such as dominance rules and a lower bound. Furthermore, a B&B algorithm and four bee-colony-based metaheuristics are presented. Wu et al. (2019) addressed a COSP considering ready times. Several dominance properties and two lower bounds are presented, which are embedded into a B&B algorithm. Five constructive heuristics are adapted to this variant. Besides, an Iterated Greedy (IG) algorithm is developed. Wu et al. (2021) addressed the COSP considering due dates and scenario-dependent processing times. The performance measure is the minimization of the maximum total tardiness across all scenarios. Four lower bounds, as well as four dominance relations, were proposed. Furthermore, some local search and IG population-based algorithms were presented. The IG approaches outperformed the OMDD-based heuristics, although they increased the time required for the problem resolution. Prata et al. (2021a) addressed the COSP with explicit setups that depend on the production sequence to minimize the makespan. Two mixed-integer linear programming models are developed for this problem, along with two matheuristics that reduce the number of decision variables. Their so-called Fixed Variable List Algorithm (FVLA) is shown to outperform all other methods under comparison. Prata et al. (2022) addressed the customer order scheduling with sequence-dependent setup times with the total completion time objective. A MILP model and a hybrid discrete differential evolution algorithm are developed. As innovative characteristics, this metaheuristic presents a parameter-free restart operator and two local search procedures based on heuristic dominance properties. Computational experimentation pointed to the superiority of the discrete differential evolution algorithm in comparison with other methods.

3. Problem description

The $DPm \rightarrow 0 \mid ST_{sd} \mid \Sigma T$ is NP-hard problem since it can be reduced, if all setup times are equal to zero, to the canonical customer order scheduling problem to minimize total completion time, which is known to be an NP-hard problem (Wagneur and Sriskandarajar, 1993). Here we propose a MILP model to find high-quality solutions or even global optimal solutions to the problem under study. Each order k has an associated due date d_k, a processing time p_{ik} in machine i, and a sequence-dependent S_{ikl}. The problem is to find a sequence of orders where the total tardiness is minimized. Next, we define the tardiness for a given order, as well as the total tardiness minimization. The completion time in machine i for a given order processed in postion j of sequence Π, $C_{i,\pi_j}(\Pi)$, can be calculated recursively as in Eq. (1):

$$C_{i,\pi_j}(\Pi) = C_{i,\pi_{j-1}}(\Pi) + p_{i,\pi_j} + S_{i,\pi_{j-1},\pi_j}, \quad i = 1, \dots, m, \quad j = 1, \dots, n \tag{1}$$

with $C_{i,\pi_0}(\Pi) := 0, \forall i$. Therefore, $C_{\pi_j}(\Pi)$ is the completion time of the order scheduled in position j is defined as in Eq. (2):

$$C_{\pi_j}(\Pi) = \max_{1 \le i \le m} \{C_{i,\pi_j}(\Pi)\}. \quad (2)$$

In the same way, the tardiness for a given order scheduled in position j can be defined as in Eq. (3):

$$T_{\pi_j}(\Pi) = \max\{C_{\pi_j}(\Pi) - d_{\pi_j}; 0\}. \quad (3)$$

Finally, the total tardiness is calculated as $T(\Pi) = \sum_{j=1}^{n} T_{\pi_j}(\Pi)$. Likewise, the summation of the completion times $C(\Pi)$ for a given sequence Π can be calculated as $C(\Pi) = \sum_{j=1}^{n} C_{\pi_j}(\Pi)$.

The problem notation is presented as follows.

Indices and sets

Set of machines $I = \{1, \dots, m\}$;

Set of orders $K = \{1, \dots, n\}$;

Set of positions $J = \{1, \dots, n\}$.

Parameters

p_{ik}: machine-dependent processing time of the order k;

S_{ikl}: sequence-dependent setup times.

d_k: due date of the order k;

M: a sufficiently large number.

Decision variables

T_k: tardiness of order k;

C_{ij}: completion time of position j in machine i;

D_{ij}: setup time for the order scheduling in the position j of machine i.

$$x_{ijk} = \begin{cases} 1, & \text{if the order } k \text{ is scheduled in the position } j \text{ of machine } i. \\ 0, & \text{otherwise} \end{cases}$$

Thereby, resulting MILP model can be described in the following manner:

$$\min \sum_{k=1}^{n} T_k \quad (4)$$

subject to:

$$\sum_{k=1}^{n} x_{ijk} = 1, \quad \forall i \in I, \quad \forall j \in J \quad (5)$$

$$\sum_{j=1}^{n} x_{ijk} = 1, \quad \forall i \in I, \quad \forall k \in K \quad (6)$$

$$S_{ikl}\left(x_{ij-1k} + x_{ijl} - 1\right) \le D_{ij}, \quad \forall i \in I, \quad \forall j > 1, \quad \forall k, l \in K \quad (7)$$

$$\sum_{r=1}^{j} D_{ir} + \sum_{k=1}^{n}\sum_{r=1}^{j} p_{ik} \cdot x_{irk} \leq C_{ij}, \quad \forall i \in I, \quad \forall j \in J \tag{8}$$

$$C_{ij} - d_k + x_{ijk} \cdot M \leq M + T_k \quad \forall i \in I, \forall j \in J, \forall k \in K \tag{9}$$

$$x_{ijk} \in \{0,1\}, \quad T_k \geq 0, \quad C_{ij} \geq 0, \quad D_{ij} \geq 0, \quad \forall i \in I, \quad \forall k \in K, \quad \forall j \in J \tag{10}$$

Eq. (4) is the objective function to be minimized. Constraints (Eq. 5) and (Eq. 6) are permutation restrictions. Constraint set Eq. (7) determines the setup time of the machine i from order k before order l. We consider that all the available machines performed the setup operation previously in the first position. Set of constraints (Eq. 8) computes the completion times of the orders. Constraint set (Eq. (9)) determines the tardiness associated with each customer order. Finally, constraint sets (Eq. (10)) determine the scope of the variables.

We define the big-*M* value as the summation of the processing and setup times, as in Eq. (11).

$$M = \sum_{i=1}^{m}\sum_{k=1}^{n} p_{ik} + \sum_{i=1}^{m}\sum_{k=1}^{n}\sum_{l=1}^{n} S_{ikl} \tag{11}$$

Finally, it should be noticed that for this problem we have to consider the possibility of different permutations for every machine. Contrary to the variant without sequence-dependent setup times, where the optimal solution is obtained applying the same permutation on all the available machines by Lee (2013). Here, we show a counterexample to this fact.

Proposition 1: The optimal solution does not necessarily present the same permutation in all the available machines for the $DPm \rightarrow 0|ST_{SD}|\sum T_j$.

Proof: Considering the following example, as described in Table 1. Also, we consider a due date of 5 units of time (u.t.) for both orders.

Table 1
Illustrative example.

Processing times

	O_1	O_2
M_1	2	1
M_2	3	4

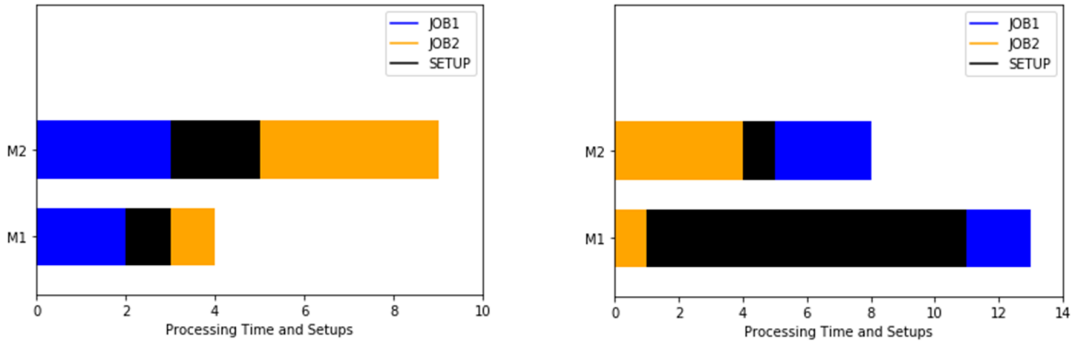
Setup times-machine 1

	O_1	O_2
O_1	-	1
O_2	10	-

Setup times-machine 2

	O_1	O_2
O_1	-	2
O_2	1	-

Using the same permutation in all machines, we obtain the results illustrated in Fig. 1, which present a total tardiness of the 4 u.t. and 8 u.t. time units, respectively. If we use distinct permutations in the machines, we can obtain a total tardiness of 3 u.t. time units, as illustrated in Fig. 2. Thus, solutions with distinct permutations in the machines can lead to smaller total tardiness than solutions with fixed permutations in the machines.



(a) (b)

Fig. 1. Solutions with fixed permutations

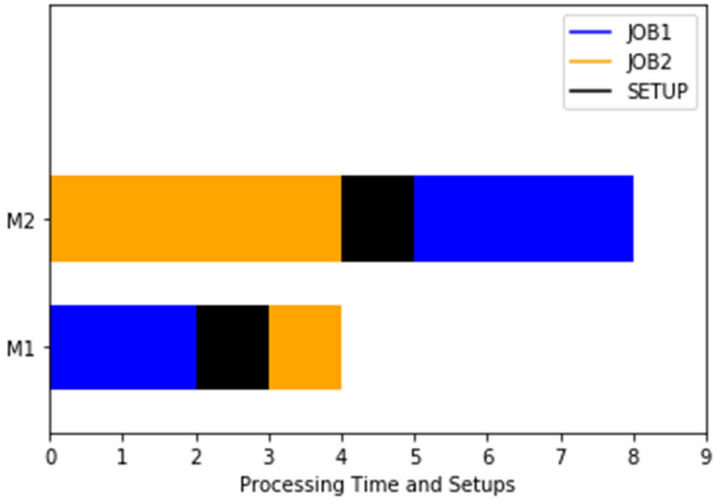


Fig. 2. Solution with distinct permutations.

4. Proposed solution approaches

Since the proposed variant is a NP-hard problem, we presented heuristic algorithms to find high-quality solutions within admissible computational time.

4.1 Constructive heuristics

The OMDD-S heuristic is an extension of the OMDD proposed by Lee (2013), considering sequence-dependent setup times. Our proposal is described in the following steps.

- Step 0: Set $\Omega := (1,2,\dots,n)$, $\Pi := \emptyset$ and $t_i = 0$ $(1 \le i \le m)$.
- Step 1: Choose a given order in Ω using the index α_j:

$$\alpha_j = \max\left\{\max_{1\le i\le m}\{t_i + S_{irj} + p_{ij}\} - \max_{1\le i\le m}\{t_i\}; d_j - \max_{1\le i\le m}\{t_i\}\right\}. \quad (12)$$

- Step 2: Delete a given order from Ω, including it at the end of Π.
- Step 3: Refresh t_i. If $|\Omega| = n - 1$, then $t_i := t_i + p_{ir}$; otherwise, $t_i := t_i + S_{iwr} + p_{ir}$, where w is the last but one elemetn from Π.
- Step 4: While Ω is not empty, return to Step 1.

Also, we present the FP-S heuristic, as an extension of the FP algorithm (Framinan and Perez-Gonzalez, 2018), including the sequence-dependent setup times. The above-mentioned algorithm builds a permutation $\Pi := (\pi_1, \dots, \pi_n)$ from a set of unscheduled orders $\mathcal{U}$. Initially, $\mathcal{U}$ has all the orders since none of the orders was scheduled. In iteration $\mathcal{K}$, each order in $\mathcal{U}$ is chosen to be inserted in the last position of Π. Aiming to choose a given order from the candidate list, considering $\omega_l \in \mathcal{U}$ with a due date d_l, a partial permutation S_l is builded inserting ω_l in the last position of Π, i.e. $\Pi := (\pi_1, \dots, \pi_{k-1}, \omega_l)$.

Therefore, the tardiness indicator η_l for the candidate l is calculated as in Equation (Eq.13):

$$\eta_l = \max\{C(S_l) - d_l; 0\} + TS_l \quad (13)$$

In which the first term is the portion of the total tardiness if l is selected, in which CS_l is the partial completion time of the permutation S_l and d_l is the associated due date. The second term TS_l estimates the parcel for the total tardiness of the remainder orders. Fig. 3 illustrates the FP-S constructive heuristic.

We adopt as a permutation in which all the orders are included before the procedure. For each iteration, we calculate TS_l considering the remnant orders are allocated after the orders previously selected. We consider the sequence returned by the Earliest Due Date (EDD) heuristic. This algorithm is a well-known priority rule to solve a scheduling problem with due dates. We select a given order, append it at the end of the partial permutation, and remove it. As the remnant orders were already selected, they are not calculated again in the next iteration. Our FP-S heuristic is described as follows.

- Step 0: Determine an initial permutation Ω using the EDD dispatch rule.
- Step 1: For each position *j* of the sequence, calculate the indicator for each unscheduled order.
- Step 2: For each $\omega_l \in \Omega$, estimate the completion times of ω_l for each available machine.
- Step 3: Determine the first term of indicator η_l, using Equation (Eq. 13). Next, determine the second term of indicator η_l.
- Step 4: Choose the order presenting the smaller value of η_l. Insert ω_l at the end of permutation Π. Update the list Ω.
- Step 5: Update the completion times for each machine.

4.2 Matheuristics

4.2.1 SPAM

As in the problem under study, each machine can receive a distinct sequence we face a problem with a non-permutational encoding. We can observe that this encoding does not necessarily lead to better solutions because the search space can increase substantially. Fernandez-Viagas et al. (2019) demonstrate that permutational encodings can provide better results than non-permutational encodings in other production scheduling problems. This is because a solution procedure can evaluate a higher number of solutions with a permutational encoding, for a given time limit.

An intuitive procedure to solve the problem under study is to consider a fixed permutation in all machines. In this way, we can replace the decision variable x_{ijk} in the model defined in Equations (*Eq.4*) – (*Eq.10*) by a decision variable x_{kj} equals to 1 if the order k is processed in position j, 0 otherwise. The *Same Permutation in All Machines* (SPAM) matheuristic reduces the number of binary decision variables from mn^2 to n^2. We can observe that the optimal solution returned by the SPAM matheuristic is not necessarily the optimal solution to the original problem.

4.2.2 A hybrid matheuristic

We also propose a hybrid matheuristic based on the JPO algorithm proposed by Framinan and Perez-Gonzalez (2018), denominated *Job-Position Oscillation with the Same Permutation in All Machines* (SPAM-JPO). We consider only solutions with a fixed permutation, restricting that a given order could be replaced δ positions at most in the sequence.

Let a permutation Π with all orders k $(k = 1, \dots, K)$, we can obtain the position $w_k(\Pi)$ of order *k*. Therefore, a set of feasible positions for the order *k* is determined so that the order k can be replaced with a maximal number of δ positions backward or forward. The value δ is a required input of the matheuristic. The set of possible positions for the order *k* is given by k é $\mathcal{P}_k(\Pi, \delta) = \{w_k(\Pi, \delta), w_k(\Pi, \delta) + 1, \dots, \overline{w}_k(\Pi, \delta)\}$, where $w_k(\Pi, \delta) = \max\{0; w_k(\Pi) - \delta\}$ and $\overline{w}_k(\Pi, \delta) = \min\{n; w_k(\Pi) + \delta\}$. Thereby, for each order *k* the decision variables x_{kj} are not significant for $j \in \mathcal{P}_k(\Pi, \delta)$. Similarly, the set of orders that can be placed in a position *j* can be stated as $\mathcal{O}_j(\Pi, \delta) = \{k: j \in \mathcal{P}_k(\Pi, \delta)\}$. The hybrid matheuristic SPAM-JPO is expressed as follows.

$$\min \sum_{j=1}^{n} T_j \tag{14}$$

subject to:

$$\sum_{k \in \mathcal{O}_j(\Pi, \delta)} x_{kj} = 1, \quad \forall j \in J \tag{15}$$

$$\sum_{j \in \mathcal{P}_k(\Pi, \delta)} x_{kj} = 1, \quad \forall k \in K \tag{16}$$

$$S_{ikl}\left(x_{kj-1}+x_{lj}-1\right)\leq D_{ij},\quad \forall i\in I,\quad \forall j>1,\quad \forall k,l\in K \tag{17}$$

$$\sum_{r=\underline{w}_k(\Pi,\delta)}^{\min\{j:\overline{w}_k(\Pi,\delta)\}} D_{ir}+\sum_{k\in\mathcal{O}_j(\Pi,\delta)}\sum_{r=\underline{w}_k(\Pi,\delta)}^{\min\{j:\overline{w}_k(\Pi,\delta)\}} p_{ik}\cdot x_{kr}\leq C_j,\quad \forall i\in I,\quad \forall j\in J \tag{18}$$

$$C_j-\sum_{k\in\mathcal{O}_j(\Pi,\delta)} d_k\cdot x_{kj}\leq T_j,\quad \forall j\in J \tag{19}$$

$$x_{kj}\in\{0,1\},\quad T_j\geq 0,\quad C_j\geq 0,\quad D_{ij}\geq 0,\quad \forall i\in I,\quad \forall k\in K,\quad \forall j\in J,\quad j\in\mathcal{P}_k(\Pi,j) \tag{20}$$

Eq. (19) is the total tardiness minimization. Constraint sets Eq. (15) and Eq. (16) impose that the problem solution is a fixed permutation. Constraint set Eq. (17) calculates the setup time for the order scheduled in the position j of the machine i. Constraint set Eq. (18) calculates the completion time of each order processed in position j. Constraint set Eq. (19) defines the tardiness for of each order processed in position j. Lastly, constraint sets Eq. (20) express the domain of the decision variables.

Our SPAM-JPO matheuristic can be summarized as follows. Taking an initial permutation into account, determine a corresponding solution with the positional decision variables x_{kj}. While a general time limit t_{limit} is not exceeded, solve the MILP model defined by Eqs. (14-20), using a permutation Π, an oscillation parameter δ, and a secondary time limit t_w. Update the solution found as well as the parameter δ. Finally, return the best solution found.

5. Computational results

5.1 Experimental design

Aiming to perform a fair comparison between the evaluated methods, we developed a testbed similar to the test problems proposed by Lee (2013). However, in our test instances, we consider the generation of sequence-dependent setup times. We generate two data sets: the small-sized test instances with $m\in\{3,5,9\}$, $n\in\{8,12,16\}$, and the large-sized test instances with $m\in\{2,5,8\}$ and $n\in\{10,20,30,40\}$. Since the problem under study was not reported yet, we randomly generated the evaluated test instances, based on the related instances previously presented. The testbed is composed of test instances with $m\in\{2,5,8\}$ and $n\in\{10,20,30,40\}$. The processing times follow a uniform distribution $U[1,100]$. The sequence-dependent setup times follow three uniform distributions $\{U[1,25],U[26,75],U[76,125]\}$. The due date d_k follows a uniform distribution, where:

- P is the summation of processing and setup times divided by the number of available machines. Aiming to generate tight due dates, we multiply this expression by a factor μ.

$$P=\left[\frac{\sum_{i=1}^{m}\sum_{k=1}^{n}p_{ik}+\sum_{i=1}^{m}\sum_{k=1}^{n}\sum_{l=1}^{n}S_{ikl}}{m}\right]\cdot\mu,$$

- RDD is the interval for due dates: RDD: $\{0.2,0.5,0.8\}$;
- TF is the tardiness factor for the due dates: TF: $\{0.2,0.5,0.8\}$

For each combination of parameters were generated 10 test instances, totalizing 3240 instances. The methods under comparison are listed below.

- EDD heuristic (a well-know algorithm for problems with due dates);
- OMDD heuristic proposed by Lee (2013);
- FP heuristic proposed by Framinan and Perez-Gonzalez (2018);
- JPO matheuristic by Framinan and Perez-Gonzalez (2018);
- The proposed OMDD-S heuristic;

- The proposed FP-S heuristic;
- The proposed SPAM matheuristic;
- The proposed hybrid matheuristic SPAM-JPO;
- The proposed MILP model.

We can observe that the warm start of the JPO and SPAM-JPO matheuristics was the OMDD-S. Framinan and Perez-Gonzalez (2018) used the FP heuristic as the warm start. We decided to use the OMDD-S as the initial solution, taking into consideration preliminarily computational experiments.

The methods under comparison were evaluated considering four indicators:

- Relative Deviation Index (RDI) indicator as a performance measure, which is a standard indicator for scheduling problems with due-date objectives. Let H be a set of methods, the RDI found for the method $s \in H$ when tested in instance t is calculated as in Eq. (21).

$$RDI_{st} = \begin{cases} 0, & \text{if } \min_{h \in H} T_{ht} = \max_{h \in H} T_{ht}, \\ \dfrac{T_{st} - \min_{h \in H} T_{ht}}{\max_{h \in H} T_{ht} - \min_{h \in H} T_{ht}} \cdot 100, & \text{otherwise.} \end{cases} \quad (21)$$

  where T_{st} is the tardiness value returned by method s in instance t. In our study, $\min_{h \in H} T_{ht}$ the best solution found among the methods under comparison.

- SR (*success rate*). This indicator is determined as the number of instances in which a given solution procedure return the better solution divided by the number of evaluated test instances Eq. (21):

$$SR = \frac{n_{BEST}}{n_{INST}} \times 100 \quad (22)$$

  where n_{BEST} is the number of test problems where the solution procedure returned the best solution and n_{INST} is the number of evaluated test problems.

- Optimality (Opt) evaluates if the solution obtained by the method under comparison in each test instance is the global optimal solution.
- computational time, expressed in seconds.

For the small-sized and large-sized test problems, we generated 10 instances for each combination of parameters, totalizing 2430 and 3240 instances, respectively. The proposed test instances are available here: https://www.researchgate.net/publication/352289324_Evaluated_Instances. The methods under comparison were implemented with Python 3.7. Furthermore, the MILP model and the SPAM matheuristic were solved with IBM ILOG CPLEX version 12.8.0. For all the evaluated methods, we consider a time limit t_{limit} = 600s. All the algorithms were executed in an Intel Core i5-3470 with 3.20 GHz and 8 GB RAM.

5.2 Results and discussion

5.2.1 Small-sized test instances

Tables 2 and 3 illustrate the computational results for different *m* and *n* values, considering the average RDI values, the standard deviations, the percentage of optimal solutions found, and the average computational times. We can observe that in the first set of instances ($m = 3$ and $n = 8$), the MILP model presented better results. However, with the increase of the size of the problems, the matheuristic SPAM returned better results. Based on Table 4, we can emphasize that the MILP model returned the largest number of optimal solutions for the small-sized test instances.

Table 2
Small-sized test instances: RDI values for *m* and *n* (part 1)

m	*n*	EDD			FP			FP-S			OMDD		
		Average	St. Dv.	Time	Average	St. Dv.	Time	Average	St. Dv.	Time	Average	St. Dv.	Time
3	8	95.5924	10.7901	0.0000	72.7618	26.1200	0.0010	45.0979	24.0381	0.0014	69.9282	25.4258	0.0010
	12	97.9772	6.9836	0.0001	75.9761	25.5007	0.0030	48.9008	22.8246	0.0045	72.9033	27.9774	0.0030
	16	92.1670	24.7728	0.0001	73.274	30.1649	0.0066	47.3200	26.5247	0.0103	70.0654	31.4608	0.0069
5	8	93.2917	14.4990	0.0000	68.9482	25.3962	0.0013	42.6921	23.2525	0.002	63.9066	27.7710	0.0013
	12	95.1717	12.1162	0.0001	72.4786	24.0260	0.0038	46.8317	23.7087	0.0062	70.3361	27.2906	0.0039
	16	88.7469	26.0195	0.0001	69.2486	28.9762	0.0087	48.4049	27.7139	0.0144	67.2913	30.6391	0.0090
9	8	91.3742	16.2296	0.0000	68.3929	25.7720	0.0019	42.1489	22.6573	0.0031	60.788	26.8431	0.0019
	12	90.7482	15.9556	0.0001	68.6689	23.9219	0.0056	44.3806	22.2601	0.0098	64.7505	25.2717	0.0058
	16	80.2203	29.7194	0.0001	60.9753	29.9828	0.0127	40.2145	25.8252	0.0226	57.2266	31.6399	0.0131
Total		91.6988	19.4780	0.0001	70.0805	27.0107	0.005	45.1101	24.5031	0.0083	66.3551	28.7118	0.0051

Table 3
Small-sized test instances: RDI values for *m* and *n* (part 2)

m	*n*	OMDD-S			SPAM			MILP		
		Average	St. Dv.	Time	Average	St. Dv.	Time	Average	St. Dv.	Time
3	8	39.6805	22.7597	0.0003	5.9073	8.6510	5.2375	**2.8183**	7.1515	334.8832
	12	39.6883	23.0271	0.0007	**2.6774**	6.7317	438.3857	18.1819	20.9791	601.7716
	16	41.2252	26.7577	0.0011	**2.2289**	6.2924	542.1021	23.2568	25.6060	540.4455
5	8	34.9788	21.1950	0.0005	**2.2768**	4.9861	9.7212	23.7074	28.8696	550.4171
	12	38.3124	23.1388	0.0009	**1.0847**	5.5263	479.4585	46.7985	30.8030	600.1471
	16	36.8398	25.4770	0.0015	**1.3108**	5.0498	555.001	49.9690	33.9772	549.4129
9	8	32.1479	20.2378	0.0006	**0.0859**	0.7310	19.4298	56.5587	30.3852	600.2293
	12	35.4173	21.4594	0.0014	**0.5604**	4.0074	555.9576	68.1536	29.6034	608.7139
	16	29.3475	24.8520	0.0022	**1.5982**	6.1035	565.6225	67.5235	35.8523	582.6817
Total		36.4042	23.5445	0.0010	**1.9700**	5.9310	352.324	39.6631	35.7772	552.078

Table 4
Small-sized test instances: optimality percentage for *m* and *n*

m	*n*	EDD	FP	FP-S	OMDD	OMDD-S	SPAM	MILP
3	8	0.00%	0.00%	1.11%	0.00%	1.11%	17.78%	61.11%
	12	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	1.11%
	16	6.30%	6.30%	6.30%	6.30%	6.30%	10.37%	12.22%
5	8	0.00%	0.37%	0.00%	0.00%	0.00%	4.07%	17.04%
	12	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.74%
	16	5.93%	5.93%	5.93%	5.93%	5.93%	9.26%	11.48%
9	8	0.00%	0.00%	0.00%	0.00%	0.00%	0.37%	1.11%
	12	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	16	5.56%	5.56%	5.56%	5.56%	5.56%	7.04%	7.78%
Total		1.98%	2.02%	2.10%	0.0198	2.10%	5.43%	12.51%

Tables 5 and 6 present the same aggregated results analyzed by the *TF* and *RDD* values. We can observe that the SPAM matheuristic returned the better results, expressed in terms of the average and standard deviation values. Analyzing Table 7, we can also observe that the MILP model obtained the largest percentual of optimality.

Table 5
Small-sized test instances: RDI values for *TF* and *RDD* (part 1)

TF	*RDD*	EDD			FP			FP-S			OMDD		
		Average	St. Dv.	Time	Average	St. Dv.	Time	Average	St. Dv.	Time	Average	St. Dv.	Time
0.2	0.2	97.2370	8.8203	0.0001	78.5159	26.0079	0.0049	55.4418	27.1227	0.0083	77.8198	27.5458	0.0051
	0.5	88.6677	27.4747	0.0001	70.8692	31.8983	0.0050	49.3033	28.6754	0.0082	68.6830	32.7437	0.0051
	0.8	85.1766	31.4354	0.0001	69.8468	33.359	0.0049	49.5453	30.5102	0.0082	65.3195	34.7565	0.0051
0.5	0.2	94.2726	12.7031	0.0001	72.1748	26.3138	0.0050	46.3215	22.3224	0.0082	72.3025	25.9000	0.0051
	0.5	92.1407	15.6173	0.0000	72.5721	25.2546	0.0049	44.8446	21.2566	0.0083	67.2162	28.1337	0.0051
	0.8	89.5994	20.0316	0.0001	70.2081	26.2634	0.0050	42.7151	24.0551	0.0083	65.9920	29.0104	0.0051
0.8	0.2	94.3325	13.0639	0.0001	66.6091	23.0282	0.0050	42.7166	20.5596	0.0082	58.6990	24.3036	0.0051
	0.5	92.2061	16.5154	0.0001	63.613	23.5619	0.0050	37.7572	18.4806	0.0082	59.3584	24.4476	0.0051
	0.8	91.6570	15.9317	0.0001	66.3153	22.9051	0.0050	37.3459	19.1649	0.0083	61.8058	24.6797	0.0051
Total		91.6988	19.4780	0.0001	70.0805	27.0107	0.0050	45.1101	24.5031	0.0083	66.3551	28.7118	0.0051

Table 6
Small-sized test instances: RDI values for *TF* and *RDD* (part 2)

TF	*RDD*	OMDD-S			SPAM			MILP		
		Average	St. Dv.	Time	Average	St. Dv.	Time	Average	St. Dv.	Time
0.2	0.2	51.5566	24.1698	0.0010	**4.3980**	8.6981	373.9367	22.5721	29.3454	541.7492
	0.5	44.0141	27.3227	0.0010	**3.3462**	7.2883	306.9237	27.1313	34.3152	494.8373
	0.8	42.4777	30.1257	0.0010	**4.0201**	9.2227	297.3631	27.2049	34.7209	482.5090
0.5	0.2	37.8550	19.4474	0.0010	**0.9346**	3.6179	375.1919	39.0991	32.4801	566.5620
	0.5	36.4583	20.5938	0.0010	**1.6021**	5.0088	64.7990	44.6720	37.0097	573.6156
	0.8	32.7976	22.2552	0.0010	**1.2377**	4.3834	360.7954	46.0753	37.4934	575.5708
0.8	0.2	29.2587	17.6455	0.0010	**0.6277**	2.8225	359.8273	48.1889	31.7879	582.6589
	0.5	27.2532	17.4737	0.0010	**0.8362**	3.1991	367.9543	52.0129	35.1051	577.6879
	0.8	25.9665	16.3588	0.0010	**0.7279**	3.1273	364.1244	50.0111	35.1252	573.5116
Total		36.4042	23.5445	0.0010	**1.9700**	5.9310	352.324	39.6631	35.7772	552.0780

Fig. 5 describes the aggregated results considering *TF*. We can observe that matheuristic SPAM returned the best average results for the small-sized test instances. The MILP model obtained good results for $TF = 2$: however, for *TF* equals 0.5 and 0.8, it was outperformed by the SPAM and OMDD-S methods, respectively. Figure 6 illustrates the aggregated boxplots concerning the *RDD* indicator. Considering these results, we can highlight that the SPAM matheuristic returned the better average results for all the methods under comparison.

Table 7
Small-sized test instances: % of optimality for *TF* and *RDD*

TF	*RDD*	EDD	FP	FP-S	OMDD	OMDD-S	SPAM	MILP
0.2	0.2	0,00%	0,00%	0,00%	0,00%	0,00%	4,44%	13,70%
	0.5	7,41%	7,41%	7,41%	7,41%	7,41%	12,22%	21,85%
	0.8	10,37%	10,37%	10,37%	10,37%	11,11%	14,81%	27,41%
0.5	0.2	0,00%	0,00%	0,00%	0,00%	0,00%	2,96%	10,37%
	0.5	0,00%	0,00%	0,00%	0,00%	0,00%	2,96%	9,63%
	0.8	0,00%	0,37%	0,37%	0,00%	0,00%	4,07%	8,15%
0.8	0.2	0,00%	0,00%	0,37%	0,00%	0,37%	3,33%	6,30%
	0.5	0,00%	0,00%	0,00%	0,00%	0,00%	1,48%	7,78%
	0.8	0,00%	0,00%	0,37%	0,00%	0,00%	2,59%	7,41%
Total		1,98%	2,02%	2,10%	1,98%	2,10%	5,43%	12,51%

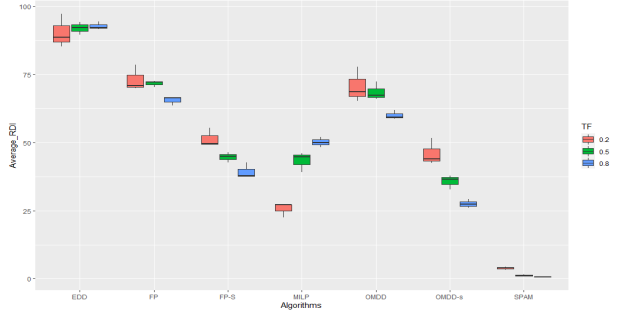


Fig. 5. Small-sized test instances: boxplot for average *RDI* values depending on *TF*

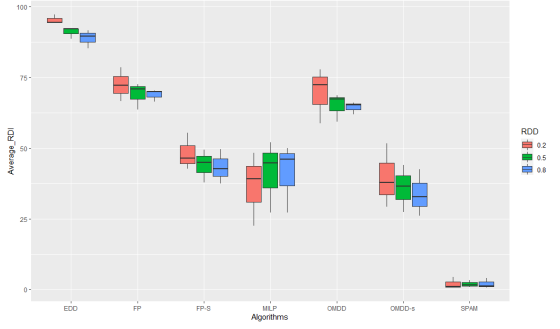


Fig. 6. Large-sized test instances: boxplot for average *RDI* values depending on *RDD*

Table 8 presents the results for the *SR* indicator. We can observe that the SPAM matheuristic returned the largest *SR* value between all the evaluated methods. Based on the performance indicators analyzed in the computational results, we can conclude that SPAM matheuristic is the best method for the small-sized test instances.

Table 8
Small-sized test instances: SR values

EDD	FP	FP-S	OMDD	OMDD-S	SPAM	MILP
1,98%	0,65%	0,93%	0,82%	1,37%	30,75%	8,93%

5.2.2 Large-sized test instances

Table 9 and Table 10 illustrates the computational results considering distinct values for m and n. Taking into consideration the first set of instances ($m = 2$ and $n = 10$), we can observe that the SPAM returns the best solutions among all the

considered methods. However, with the increasing of the values of m and n, MILP model and SPAM-JPO return the best RDI values.

Table 9
Large-sized test instances: RDI values for m and n (part 1)

m	n	EDD		FP		FP-S		OMDD		OMDD-S	
		Average	St. Dv.	Average	St. Dv.	Average	St. Dv.	Average	St. Dv.	Average	St. Dv.
2	10	96.0493	9.3616	74.0344	25.0666	56.2164	27.9392	71.7100	26.1257	38.9925	21.3457
	20	78.0808	39.7039	59.7296	36.7141	50.7682	35.8774	60.5183	37.2953	34.4581	28.589
	30	51.2699	47.9732	38.7929	38.8785	32.3783	34.8791	37.3421	38.5414	14.9562	21.4639
	40	30.6115	41.6665	20.7499	29.2799	16.2446	24.2014	20.6954	29.5093	5.5452	12.3045
5	10	93.7887	12.8440	73.2340	23.6784	58.8424	26.2345	67.0770	26.035	37.4087	21.3143
	20	68.5931	38.5886	51.8989	33.9248	47.7679	33.6858	49.4103	34.2563	26.2327	27.6293
	30	24.8256	30.0024	15.7638	19.9920	14.0551	19.2174	14.7223	19.7352	5.0594	13.3444
	40	12.5403	18.9989	7.7343	11.3168	6.2689	9.0028	7.2335	12.0397	0.8205	3.8954
8	10	91.5210	15.5189	68.1319	24.8727	56.2488	25.4111	64.4638	26.754	34.851	20.2135
	20	47.8643	34.6722	35.4441	30.6042	32.0611	30.927	34.0785	30.4616	16.3654	24.0684
	30	13.8092	17.0370	8.3775	9.3839	7.6681	9.6344	7.2338	9.0156	1.2870	3.4212
	40	7.0822	10.0866	4.0645	5.4368	4.1008	5.6089	3.7166	5.1589	0.0762	0.4868
Total		51.3363	43.6517	38.1630	36.6161	31.8851	32.8421	36.5168	36.2386	18.0044	24.0698

Table 10
Large-sized test instances: RDI values for m and n (part 2)

m	n	SPAM		MILP		SPAM-JPO		JPO	
		Average	St. Dv.	Average	St. Dv.	Average	St. Dv.	Average	St. Dv.
2	10	**2.5691**	5.0185	4.1034	6.0077	4.5762	6.4201	22.7252	18.4103
	20	9.4939	13.0045	**1.4788**	3.8262	10.6221	15.5238	29.8185	25.6702
	30	29.6135	35.5353	**3.3109**	11.3456	6.2066	9.5153	14.9408	21.4362
	40	47.8850	46.3269	7.4285	16.0226	**3.2077**	7.8071	5.5452	12.3045
5	10	44.4169	31.2991	**0.8070**	3.4905	2.6394	5.1434	35.9667	20.1085
	20	56.7089	40.8491	**1.5417**	5.5628	18.0544	24.0035	26.0185	27.2406
	30	63.7955	47.0304	8.1344	18.8562	**3.5100**	10.4422	5.0594	13.3444
	40	63.6526	48.1386	13.7993	23.4163	**0.6903**	3.8231	0.8205	3.8954
8	10	59.8336	29.7481	**0.5792**	2.9924	2.3233	5.0796	34.0763	19.5936
	20	72.9362	42.2317	**2.1991**	6.2835	10.4192	18.2998	16.2987	23.8797
	30	70.3357	45.7281	8.7350	15.6913	**1.0305**	2.7315	1.2870	3.4212
	40	65.4765	47.5565	16.1070	24.7707	**0.0684**	0.4767	0.0762	0.4868
Total		48.8931	44.5045	5.6854	14.6673	**5.2790**	12.3564	16.0528	22.1632

Table 11 describes the optimality percentages for the evaluated methods in the large-sized test instances, considering m and n values. We can observe that the MILP model and the hybrid matheuristic SPAM-JPO can find the largest number of optimal solutions. Analyzing the sets of instances separately, we can highlight that for m=5 and n=10, the MILP model obtained the largest percentage of optimality (64.81%).

Table 11
Large-sized test instances: % of optimality for m and n

m	n	EDD	FP	FP-S	OMDD	OMDD-S	SPAM	MILP	SPAM-JPO	JPO
2	10	0.00%	0.00%	0.00%	0.00%	0.37%	56.30%	43.33%	39.26%	4.07%
	20	20.00%	20.00%	20.00%	20.00%	20.00%	25.93%	25.93%	23.33%	20.37%
	30	40.00%	40.00%	40.00%	40.00%	40.00%	41.48%	41.48%	40.37%	40.00%
	40	45.19%	45.19%	45.19%	45.19%	45.19%	42.22%	45.19%	45.19%	45.19%
5	10	0.00%	0.00%	0.00%	0.00%	0.37%	4.44%	64.81%	49.63%	0.37%
	20	17.78%	17.78%	0.1778	17.78%	17.78%	20.74%	21.11%	17.78%	17.78%
	30	33.70%	33.70%	33.70%	33.70%	33.70%	33.33%	34.07%	33.70%	33.70%
	40	40.37%	40.37%	40.37%	40.37%	40.37%	35.93%	40.37%	40.37%	40.37%
8	10	0.00%	0.00%	0.00%	0.00%	0.37%	0.37%	55.19%	40.74%	0.37%
	20	15.56%	15.56%	15.56%	15.56%	15.56%	19.26%	19.26%	16.30%	15.56%
	30	31.85%	31.85%	31.85%	31.85%	31.85%	28.89%	31.85%	31.85%	31.85%
	40	38.15%	38.15%	38.15%	38.15%	38.15%	33.33%	38.15%	38.15%	38.15%
Total		23.55%	23.55%	23.55%	23.55%	23.64%	28.52%	38.40%	34.72%	23.98%

Based on Table 9 and Table 10, we can observe that FP-S algorithm outperforms the FP heuristics since the first one presents smaller average RDI values for all the sets of instances, except for the problems with $m = 8$ and $n = 40$. However, the difference between the RDI values between FP-S and FP heuristics in these test instances is only 0.0363%. We can also emphisize that OMDD-S algorithm outperforms the OMDD heuristic since it returned best average RDI values for all the test instances. Furthermore, OMDD-S presented the better results considering all the methods under comparison for five groups of test instances.

Table 12 and Table 13 present the *RDI* values (average and standard deviation) considering *TF* and *RDD*. Once again, SPAM and OMDD-S have returned the best values for the RDI indicator. Also, the FP heuristic has presented the worst performance among the methods under comparison.

Table 12
Large-sized test instances: *RDI* values for *TF* and *RDD* (part 1)

TF	*RDD*	EDD		FP		FP-S		OMDD		OMDD-S	
		Average	St. Dv.	Average	St. Dv.	Average	St. Dv.	Average	St. Dv.	Average	St. Dv.
0.2	0.2	49.7108	46.3151	37.9304	39.5661	32.3280	35.8786	39.0234	39.8507	24.8065	28.6791
	0.5	44.1941	45.6207	35.7960	39.8416	33.3222	37.9785	35.1458	39.3492	21.5752	29.1973
	0.8	35.8728	44.9106	27.9817	37.7023	24.3531	34.2296	26.2024	36.6533	14.4111	23.694
0.5	0.2	60.7937	42.2037	46.5010	38.2813	39.9098	35.5864	46.3644	37.7785	26.4463	26.4100
	0.5	52.1433	44.0704	39.5498	37.8962	34.9246	35.7333	37.6886	37.8441	19.9796	26.1782
	0.8	45.3101	44.6130	32.7084	35.6869	26.9918	31.5203	32.0222	36.1065	15.1196	22.7104
0.8	0.2	64.1586	36.8215	44.7169	30.9769	34.4809	25.7323	41.7298	30.6978	14.5048	17.2718
	0.5	57.2549	38.8929	40.3317	31.0763	30.9592	25.4458	36.5535	30.5876	12.9058	16.5442
	0.8	52.5883	41.7649	37.9509	34.1006	29.6960	28.5363	33.9210	32.5531	12.2908	17.1234
Total		51.3363	43.6517	38.1630	36.6161	31.8851	32.8421	36.5168	36.2386	18.0044	24.0698

Table 13
Large-sized test instances: RDI values for *TF* and *RDD* (part 2)

TF	*RDD*	SPAM		MILP		SPAM-JPO		JPO	
		Average	St. Dv.	Average	St. Dv.	Average	St. Dv.	Average	St. Dv.
0.2	0.2	22.0448	35.8186	**2.5654**	7.2692	6.6008	13.2974	20.9657	26.1605
	0.5	25.3158	38.6867	**3.9393**	12.3742	5.7427	14.7685	19.8600	27.4597
	0.8	26.1552	40.3818	5.5456	17.7981	**3.3797**	9.0318	12.6304	21.5056
0.5	0.2	39.4726	41.5005	**1.9476**	7.9712	9.1069	14.8537	23.5571	24.2580
	0.5	45.1672	44.4587	**2.9747**	11.2436	7.7670	18.1712	17.6779	23.9755
	0.8	58.6733	44.3084	5.8597	16.5763	**4.5871**	13.3385	14.0253	21.9911
0.8	0.2	71.9319	36.8566	7.7371	15.4082	**3.6934**	6.2072	12.9980	15.8555
	0.5	75.3769	35.5378	9.9924	16.8006	**3.1289**	5.8179	11.6613	14.7730
	0.8	75.9003	34.8084	10.6064	18.7741	**3.5047**	7.9139	11.0991	15.7275
Total		48.8931	44.5045	5.6854	14.6673	**5.2790**	12.3564	16.0528	22.1632

Taking into consideration the Table 12 and 13, we can observe that FP-S algorithm outperforms the FP heuristic for all the evaluated test instances. In addition, the OMDD-S algorithm outperforms the OMMD heuristics also for all the test instances. Finally, we can highlight that the SPAM matheuristic presented better average *RDI* values than the MILP model for all the available test instances. Table 14 describes the percentage of optimality for the methods under comparison, classified by TF and RDD. We can observe that for TF=0.2 and RDD=0.8 the MILP model obtained the largest percentage of optimal solutions (59.72%). Subsequently, the hybrid matheuristic SPAM-JPO can find a greater number of optimal solutions in comparison with the remainder methods.

Table 14
Large-sized test instances: % of optimality for *TF* and *RDD*

TF	*RDD*	EDD	FP	FP-S	OMDD	OMDD-S	SPAM	MILP	SPAM-JPO	JPO
0.2	0.2	41.67%	41.67%	41.67%	41.67%	41.67%	46.39%	51.11%	50.28%	41.94%
	0.5	44.44%	44.44%	44.44%	44.44%	44.44%	56.39%	56.39%	53.06%	45.00%
	0.8	49.72%	49.72%	49.72%	49.72%	49.72%	57.78%	59.72%	58.06%	50.28%
0.5	0.2	25.00%	25.00%	25.00%	25.00%	25.00%	30.56%	38.33%	34.44%	25.28%
	0.5	26.11%	26.11%	26.11%	26.11%	26.11%	32.22%	42.78%	36.67%	26.67%
	0.8	24.17%	24.17%	24.17%	24.17%	24.44%	20.56%	42.78%	38.33%	25.00%
0.8	0.2	0.00%	0.00%	0.00%	0.00%	0.00%	4.17%	18.06%	13.89%	0.00%
	0.5	0.00%	0.00%	0.00%	0.00%	0.56%	4.44%	17.78%	13.06%	0.83%
	0.8	0.83%	0.83%	0.83%	0.83%	0.83%	4.17%	18.61%	14.72%	0.83%
Total		23.55%	23.55%	23.55%	23.55%	23.64%	28.52%	38.40%	34.72%	23.98%

Fig. 7 illustrates the boxplot for average RDI values taking into consideration the parameter TF. Given the obtained results, we can highlight that the JPO-SPAM matheuristic returned the best results for all the values of *TF*. Besides, the MILP model presented the worse results for TF equals to 0.8. Figure 8 illustrates the average RDI values depending on RDD. Once again, the hybrid matheuristic SPAM-JPO presented better results than all the other methods under comparison. Table 15 presents the SR indicator for each evaluated method. We can observe that the MILP model can find the largest number of best solutions. In fact, for the large-sized instances, the difference between the MILP model and the hybrid matheuristic SPAM-JPO is not statistically significant.

Table 15
Large-sized test instances: SR values

EDD	FP	FP-S	OMDD	OMDD-S	SPAM	MILP	SPAM-JPO	JPO
28.43%	28.43%	28.52%	28.43%	43.95%	32.47%	69.32%	59.81%	44.29%

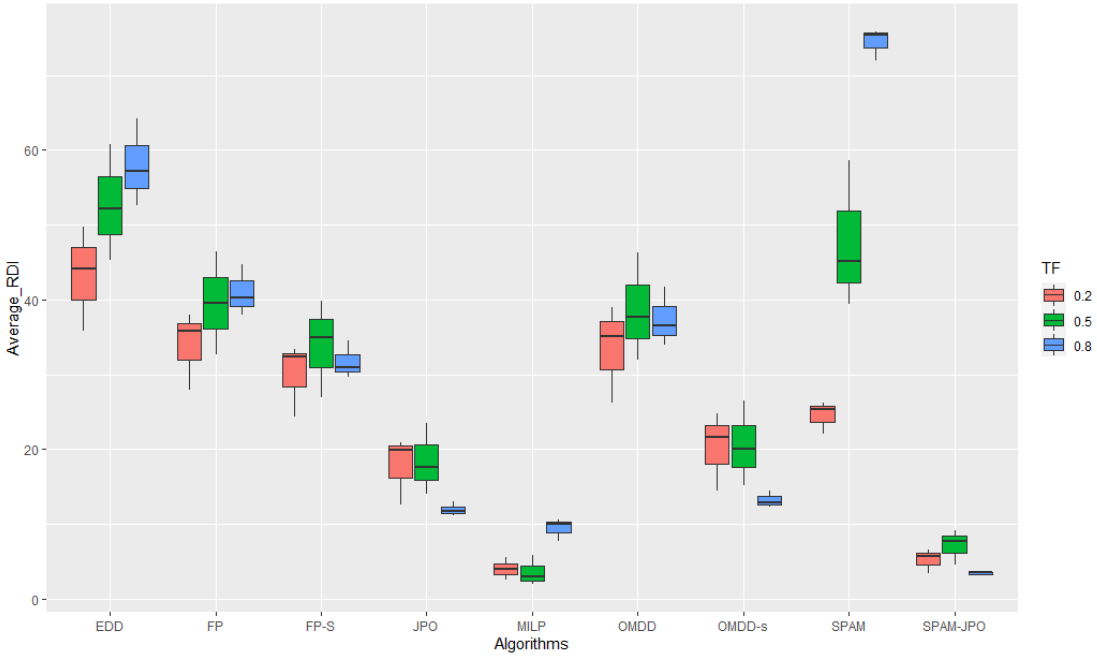


Fig. 7. Large-sized test instances: boxplot for average RDI values depending on TF

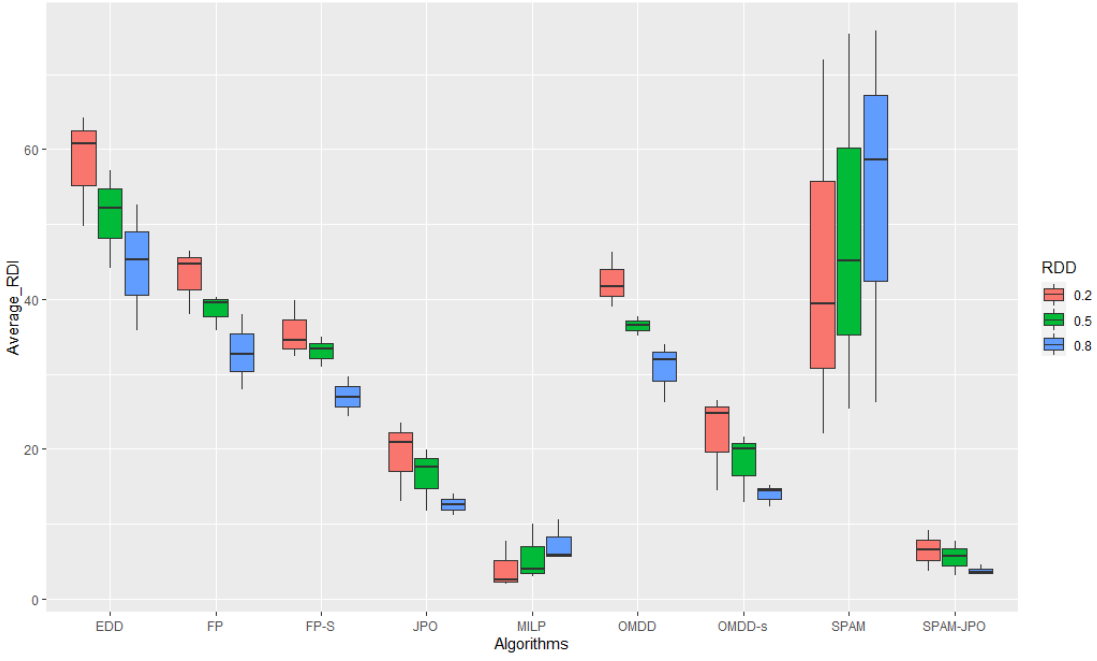


Fig. 8. Large-sized instances: boxplot for average RDI values depending on RDD

6. Conclusions

This paper has introduced a COSP that considers sequence-dependent setup time for the total completion time minimization. An MILP model has been proposed for the proposed variant. Besides, two constructive heuristics, as well as two matheuristics, have also been presented. Extensive computational experiments have been reported with two datasets composed of randomly generated test instances, aiming to evaluate the methods under comparison. As the performance indicators, the study used the RDI, the SR, and the average computational time. We evaluated 5670 instances, divided into 2430 small-sized and 3240 large-sized instances. Based on the computational experience, we can observe that for the small-sized test instances, the SPAM matheuristic and the OMDD-S constructive heuristic returned the best average RDI values. Thus, our proposal presented better results than all the other methods under comparison. Concerning the large-sized test instances, the proposed the hybrid

matheuristic SPAM-JPO outperforms the JPO metaheuristic – which is the state-of-the-art solution method for the COSP to minimize total tardiness – with the same time limit. Therefore, the proposed matheuristic is a promising method to solve the problem under study. As suggestions for future studies, other matheuristics and matheuristics could be developed. The OMDD-S heuristic could be used as a warm start to MILP and SPAM methods, improving the solutions returned by these methods. Another research avenue is the consideration of other performance measures, such as total earliness or just-in-time problems.

Acknowledgments

This study was financed in part by the Coordination for the Improvement of Higher Education Personnel (CAPES), the National Council for Scientific and Technological Development (CNPq), through Grant no. 303595/2018-7.

References

Fernandez-Viagas, V., & Framinan, J.M. (2015). *Neh-based heuristics for the permutation flowshop scheduling problem to minimize total tardiness*. *Computers and Operations Research, 60*, 27– 36.

Fernandez-Viagas, V., Perez-Gonzalez, P., & Framinan, J. M. (2019). *Efficiency of the solution representations for the hybrid flow shop scheduling problem with makespan objective*. *Computers & Operations Research, 109*, 77-88.

Framinan, J.M., & Perez-Gonzalez, P. (2017). *New approximate algorithms for the customer order scheduling problem with total completion time objective. Computers and Operations Research, 78*, 181 – 192.

Framinan, J.M., & Perez-Gonzalez, P. (2018). *Order scheduling with tardiness objective: Improved approximate solutions. European Journal of Operational Research, 266*(3), 840 – 850.

Framinan, J.M., Perez-Gonzalez, P., & Fernandez-Viagas, V. (2019). *Deterministic assembly scheduling problems: A review and classification of concurrent-type scheduling models and solution procedures. European Journal of Operational Research, 273*(2), 401 – 417.

Karabulut, K. (2016). *A hybrid iterated greedy algorithm for total tardiness minimization in permutation flowshops. Computers and Industrial Engineering, 98*, 300 – 307.

Kim, Y.D. (1993). *Heuristics for flowshop scheduling problems minimizing mean tardiness. Journal of the Operational Research Society, 44*(1), 19–28.

Kung, J.Y., Duan, J., Xu, J., Chung, I., Cheng, S.R., Wu, C.C., Lin, W.C., et al. (2018). *Metaheuristics for order scheduling problem with unequal ready times.* Discrete Dynamics in Nature and Society.

Lee, I.S. (2013). *Minimizing total tardiness for the order scheduling problem*. International Journal of Production Economics, 144(1), 128 – 134.

Leung, J.Y.T., Li, H., & Pinedo, M. (2005a). *Order scheduling models: An overview.* Multidisciplinary Scheduling: Theory and Applications, Springer US, Boston, MA., 37–53.

Leung, J.Y.T., Li, H., & Pinedo, M. (2005b). Order scheduling in an environment with dedicated re-sources in parallel. *Journal of Scheduling, 8*(5), 355–386.

Leung, J.Y.T., Li, H., & Pinedo, M. (2006). Scheduling orders for multiple product types with duedate related objectives. *European Journal of Operational Research, 168*(2), 370 – 389.

Lin, W.C., Xu, J., Bai, D., Chung, I.H., Liu, S.C., & Wu, C.C. (2019). Artificial bee colony algorithmsfor the order scheduling with release dates. *Soft Computing, 23*(18), 8677–8688.

Michael, L.P. (2018). *Scheduling: theory, algorithms, and systems.* Springer.

Prata, B.A., Rodrigues, C.D., & Framinan, J.M. (2021a). Customer order scheduling problem to minimize makespan with sequence-dependent setup times. Computers & Industrial Engineering, 151, 106962.

Prata, B.A., Rodrigues, C.D., & Framinan, J.M. (2021b). A differential evolution algorithm for the customer order scheduling problem with sequence-dependent setup times. *Expert Systems With Applications*, 116097.

Riahi, V., Newton, M.H., Polash, M., & Sattar, A. (2019). Tailoring customer order scheduling search algorithms. *Computers & Operations Research, 108*, 155 – 165.

Roemer, T.A. (2006). A note on the complexity of the concurrent open shop problem. *Journal of Scheduling, 9*(4), 389–396.

Wagneur, E., Sriskandarajah, C. (1993). Openshops with jobs overlap. *European Journal of Operational Research, 71*(3), 366 – 378.

Wu, C.C., Yang, T.H., Zhang, X., Kang, C.C., Chung, I.H., Lin, W.C. (2019). Using heuristic and iterative greedy algorithms for the total weighted completion time order scheduling with release times. *Swarm and Evolutionary Computation, 44*, 913 – 926.

Wu, C. C., Bai, D., Zhang, X., Cheng, S. R., Lin, J. C., Wu, Z. L., & Lin, W. C. (2021). A robust customer order scheduling problem along with scenario-dependent component processing times and due dates. *Journal of Manufacturing Systems, 58*, 291-305.

© 2022 by the authors; licensee Growing Science, Canada. This is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC-BY) license (http://creativecommons.org/licenses/by/4.0/).