Journal of Project Management 7 (2022) 167–176

Contents lists available at GrowingScience

Journal of Project Management

homepage: www.GrowingScience.com

A size-reduction algorithm for the order scheduling problem with total tardiness minimization

Stephanie Alencar Braga-Santos[a], Giovanni Cordeiro Barroso[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: October 10, 2021
Received in revised format: December 20, 2021
Accepted: January 18, 2022
Available online:
January 18, 2022

Keywords:
Production Sequencing
Combinatorial Optimization
Matheuristics
Mixed-Integer Linear Programming

A B S T R A C T

We investigated a variant of the customer order scheduling problem taking into consideration due dates to minimize the total tardiness. Since the problem under study is NP-hard, we propose an efficient size reduction algorithm (SR). We perform an extensive computational experience and compare our proposition with JPO-20 matheuristic, the best existing algorithm for the problem under study. We use the Relative Deviation Index (RDI) and the Success Rate (SRa) as the statistical indicators for the performance measure. We must emphasize that SR presented the lowest average RDI (around 15.5 %), whereas the JPO-20 presented an average RDI approximately three times higher (around 52.5 %). Furthermore, the proposed SR presented a higher average SRa (around 66.9%), whereas the JPO-20 presented a lower average success (around 25.7%). Our proposal used a lower computational effort, resulting in a reduction for the computation times of approximately 22%. The obtained results point to the superiority of the proposed SR in comparison with the JPO-20.

1. Introduction

Currently, the increasing of the exigency level of the customers, as well as the rigorous competition among the firms, has resulted in changes in the production paradigms. The strong demand for customized goods continuously has led to the production of orders in different places or production lines, which will be assembled in each facility. Over recent years, the researchers of the production scheduling area have been paid greater attention to the assembly scheduling problems. Framinan et al. (2019) presented a new unified notation for this class of problems, surveying the current contributions and highlighting promising research topics. The customer order scheduling environment appears in several real-world problems, such as the paper and pharmaceutical industries, among others (Leung et al., 2005). In this paper, we address the customer order scheduling problem. Let n be the setof customer orders and m the set of dedicated parallel machines, all the components of the orders must be produced one time in each of the available machines in a givenposition. Each order presents an associated processing time as well as a due date. The objective function is the total tardiness minimization. Since the customer order scheduling is NP-hard with total tardiness minimization for $m \geq 2$ (Wagneur & Sriskandarajah, 1993), heuristic algorithms are required for finding high-level solutions within admissible computational times. This paper aims at presenting a size-reduction algorithm (SR) for the customer order scheduling problem with total tardiness minimization. We extended the traditional size-reduction approach based on processing times to an approach based on due dates. We develop an efficient scheme for fixing as zero some decision variables using the problem due dates anda dispatch rule, such as the well-known earliest due date algorithm. Based on extensive computational experimentation performed with benchmark test instances, our proposition outperformed the JPO-20 matheuristic proposed (Framinan & Perez-Gonzalez, 2018),the best algorithm found in the revised literature.

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

doi: 10.5267/j.jpm.2022.1.001

The remainder of the paper is structured as follows. In Section 2, we present some related approaches. In Section 3, we describe the problem under study. In Section4, we present the proposed size-reduction algorithm. In Section 5 we present the computational experience, as well as the discussion of the results. Finally, in Section6 we present the main conclusions as well as the suggested research avenues.

2. Literature Review

Here, we present the literature review using the notation provided by Framinan et al. (2019). Julien and Magazine (1990) studied a flexible manufacturing environment where customer requirements for each of the possible product types are known in advance. Wagneur and Sriskandarajah (1993) introduced the customer order scheduling environment. Furthermore, these authors proved that this problem is NP-hard for the total tardiness minimization objective. Sung and Yoon (1998) addressed a $DPm \rightarrow 0 \mid\mid \Sigma w_j C_j$ problem in which each order presents two types of components that are processed by two independent machines specialized in a given type of component. They proposed two constructive heuristics that presented high-quality results, in comparison with a proposed lower bound. Ahmadi et al. (2005) introduced the coordinated customer order scheduling problem for the weighted completion time minimization. They proposed a Lagrangian heuristic as well as three constructive heuristics. The first one reached the best results in the analyzed set of test instances. Yang and Posner (2005) considered a production environment in which the jobs are processed in batches, and the objective function is the minimization of the total completion time in the batches. Two heuristics are proposed, presenting near-optimal solutions for the evaluated instances. Leung et al.(2005) presented two heuristics for the customer order scheduling with weighted completion time objective which outperform all heuristics previously reported in the literature. Lin and Kononov (2007)approached the problems $DPm \rightarrow 0 \mid\mid \Sigma U_j$ and $DPm \rightarrow 0 \mid\mid \Sigma w_j U_j$. These authors proved the NP-hardness of the $DPm \rightarrow 0 \mid\mid \Sigma U_j$ variant, and they developed a heuristic for the problems under study based on the tardiness weighting. Shi et al. (2017) presented a quadratic mathematical formulation for the $DPm \rightarrow 0 \mid\mid \Sigma C$, which can be converted into an equivalent mixed-integer linear programming formulation. They proposed a nested partition algorithm that presented high-quality solutions. Xu et al. (2015) addressed a variant of the customer order scheduling where the orders are subdivided into sub-lots. These authors proposed a mixed-integer linear programming formulation, a lower bound, two heuristics, and a matheuristic. The last one outperforms the two constructive heuristics, although with a higher computational cost. Xu et al. (2016) introduceda multiple-machine order scheduling problem with a learning effect to minimize thetotal tardiness. Some dominance relations are presented as well as a lower bound. As solution procedures, these authors present a simulated annealing (SA) meta-heuristic,a particle swarm optimization (PSO) meta-heuristic, and a branch-and-bound algorithm. The PSO outperforms the other approaches, however with a higher computational effort.

Lin et al. (2017) addressed a two-agent multi-facility order scheduling with ready times ($DPm \rightarrow 0 \mid\mid \varepsilon(\Sigma C^A, C^B)$). They derived several dominance properties and a lower bound on the optimal solution. As the solution procedures, a PSO and an opposite-based particle swarm optimization (O-PSO) are presented. Framinan and Perez-Gonzalez (2017) addressed the $DPm \rightarrow 0 \mid\mid \Sigma C$ variant. They developed a new constructive heuristic as well as greedy search algorithm for the problem under study. The first one incorporates a look-ahead procedure for the evaluation of the contribution to the objective function of the candidate orders as well an estimation of the contribution of the non-scheduled orders (named as FP algorithm). The second one is a greedy constructive algorithm (GSA) with some improvement procedures (perturbations and local search). These authors concluded that such approaches outperformed the existing algorithms. Riahi et al. (2019) criticized the FP algorithm because the placement of an unscheduled customer order only at the end of the scheduled partial sequence is a greedy procedure. Faced with this limitation, they proposed a new constructive heuristic considering 8 different initial priority lists. Furthermore, they developed a meta-heuristic based on perturbative and constructive procedures. The computational experiments show that the proposed approaches clearly outperformed the existing algorithms. Lee (2013) presented four constructive heuristics for the $DPm \rightarrow 0 \mid\mid \Sigma T$: total processing time earliest due date (TPT-EDD), maximum processing time earliest due date (MCT-EDD), earliest due date maximum processing time (EDD-MCT), and order modified due date (OMDD). It can be observed that OMDD outperformed all the other algorithms. Framinan and Perez-Gonzalez (2018) proposed a constructive heuristic based on the look-ahead mechanism presented by Framinan and Perez-Gonzalez (2017). Furthermore, they proposed two matheuristics called JPF and JPO for the above-mentioned problem. For the JPO, there is an oscillation parameter δ for the fixation of decision variables in the MILP. Computational highlighted $\delta = 20$ as the best parameter value. Therefore, the JPO-20 algorithm is the best-so-far algorithm for the problem under study. In our view, the definition of the oscillation of the decision variables of the JPO algorithm does not explicitly consider the characteristics of a given instance. For example, if a given algorithm explores the information of the problem due dates, the fixation of decision variables could be more efficient. Moreover, if the oscillation of the JPO algorithms occurs in the first positions of the sequence, some decision variables cannot be fixed in the oscillation window. Thus, our proposal takes into account these issues for a better reduction of the search space.

Prata et al. (2021b) introduced the customer order scheduling with sequence-dependent setup times to minimize the makespan **(**$DPm \rightarrow 0 \mid ST_{SD} \mid C_{max}$**).** As solution procedures, two mixed-integer linear programming models and two matheuristics are proposed. Prata et al. (2022) studied the **(**$DPm \rightarrow 0 \mid ST_{SD} \mid \Sigma C$**)**. A mathematical formulation is developed, as well as an innovative hybrid discrete differential evolution algorithm. Antonioli et al. (2022) addressed the **(**$DPm \rightarrow 0 \mid ST_{SD} \mid \Sigma T$**)**. The properties of the global optimal solutions are studied. Besides, several constructive heuristics and matheuristics are developed.

3. Problem description

Consider the following example with the processing times presented in Table 1. In addition, the due dates for each order are d = {4, 5, 6}. A feasible solution for this instance is the sequence Π ={ 3, 2, 1} , with a completion time vector C = {9, 6, 2} . Thus, the first order presents a tardiness of 5-time units, the second order presents a tardiness of 1 time unit and the third order does not present tardiness. Consequently, the solution illustrated in Fig. 1 presentsa total tardiness of 6-time units.

Table 1
Processing times for order scheduling example

	O_1	O_2	O_2
M_1	3	4	2
M_2	4	1	2

Fig. 1. Gantt chart for the presented solution.

Hence, we present the basic notation for the comprehension of the problem under study. Let I = { 1, 2, ..., m} be a set of machines, J = { 1, 2, ..., n} a set of positions, and K = {1, 2, ..., k} a set of orders. We define p_{ik} is the processing time of order k in machine i, and d_k the due date of order k. We then define the following decision variables: x_{kj} a binary decision variable in which 1 indicates whether order k is producedin position j, 0 otherwise. In addition, T_j is the tardiness of order in position j. The mixed-integer programming model for the problem under study proposed by Framinan and Perez-Gonzalez (2018) is presented as follows.

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

subject to

$$\sum_{k=1}^{n} x_{kj} = 1, \forall j \tag{2}$$

$$\sum_{j=1}^{n} x_{kj} = 1, \forall k \tag{3}$$

$$T_j \geqslant \sum_{k=1}^{n} \sum_{r=1}^{j} p_{ik}\, x_{kr} - d_k x_{kj}, \forall i, j \tag{4}$$

$$T_j \geqslant 0, \forall j \tag{5}$$

$$x_{kj} \in \{0,1\}, \forall k, j \tag{6}$$

The objective function (1) is the total tardiness minimization. Set of constraints (2) ensures that an order is scheduled only in a position k. Set of constraints (3)enforces that a position receives only a job j. Set of constraints (4) calculates the tardiness for each order. Finally, constraint sets (5), and (6) determine domain of the decision variables.

4. Proposed solution approach

In an integer linear programming model with binary decision variables in which permutation constraints appear, the number of decision variables with a value equal to one usually is much smaller than the number of decision variables with a zero value in the optimal solution. In view of the parameters of the integer linear model, the possibility of some decision variables appear in high-quality solutions can be small. Thus, aiming to reduce the size of a given optimization problem and speed up its solution, a percentage of these decision variables can be fixed as zero before the beginning of the analysis. The size reduction algorithm (SR) is introduced by Fanjul-Peyro and Ruiz (2011, 2017) has been applied in other production sequencing optimization problems, presenting competitive results (Fanjul-Peyro et al., 2017, Prata et al., 2021a). We can observe that there is no guarantee that the SR provided the global optimal solution; however, it frequently is a useful matheuristic. Lee (2013) shows that the global optimal solution for the $DPm \rightarrow 0 \,||\, \Sigma T$ presents the same permutation of orders for all machines. Since we present positional decision variables x_{kj} representing a permutation, we have $\frac{k}{jk}$ decisions variables equal to 1 in the feasible solutions. Thus, the greater part of the decision variables is equal to 0 in the feasible solutions. In the problem under study, we are looking to the total tardiness minimization. Therefore, we can use the information related to the due dates for setting several decision variables as 0 in a size-reduction approach. We can infer that orders with the largest due dates hardly will be allocated in the first positions of the sequence inhigh-quality solutions.

Aiming to determine which decision variables can be set as zero, we can use a constructive heuristic that considers the problem due dates. Depending on the solution returned by a constructive heuristic, the decision variable associated with a given position can be fixed since high-quality solutions hardly allocate an order in a position that implies high tardiness. In Fig. 2 we present an example of the proposed size-reduction algorithm. We consider well-known earliest due date (EDD) heuristic, in which the orders are sorted according to a non-decrescent sequence Π ={7, 10, 6, 9, 1, 4, 8, 2, 5, 3}. With the basis on this initial solution, we use a parameter, called α, for which we set as zero all the decision variables associated with a difference between the due date values greater than the α d_k. In this example, as we adopt α =50%. For the order o_7 allocated in the first position, the orders o_4, o_8, o_2, o_5, and o_3 present a percentage difference greater than 50%. Thus, the associated decision variables to these orders are set as 0. In Figure 2, the rectangles illustrate the range of the positions that are not set as zero. The different colors of the rectangles emphasize that some positions are in the extremities of the sequence. The circles highlight the orders considered in each position. Fig. 3 illustrates the proposed SR algorithm. The algorithm receives as input parameters α, which determinates the percentage of decision variables to be maintained free for the solver, and t_{limit}, the time limit adopted for the solver. Firstly, the proposed approach uses an initial solution based on the OMMD heuristic (Lee, 2013). Given a sequence Π, $\overline{x}$ presents the binary decision variable with the corresponding positions to the permutation Π. For all the positions j (j = 1, ..., n), we calculate w_{min} and w_{max}, which are the lower and upper bounds for the window with free positions.The positions located out of this interval are set as zero. After that, the model defined by Eqs. (1-6), adding the constraints (7):

$$x_{kj} = 0, \forall \overline{x} = 0 \tag{7}$$

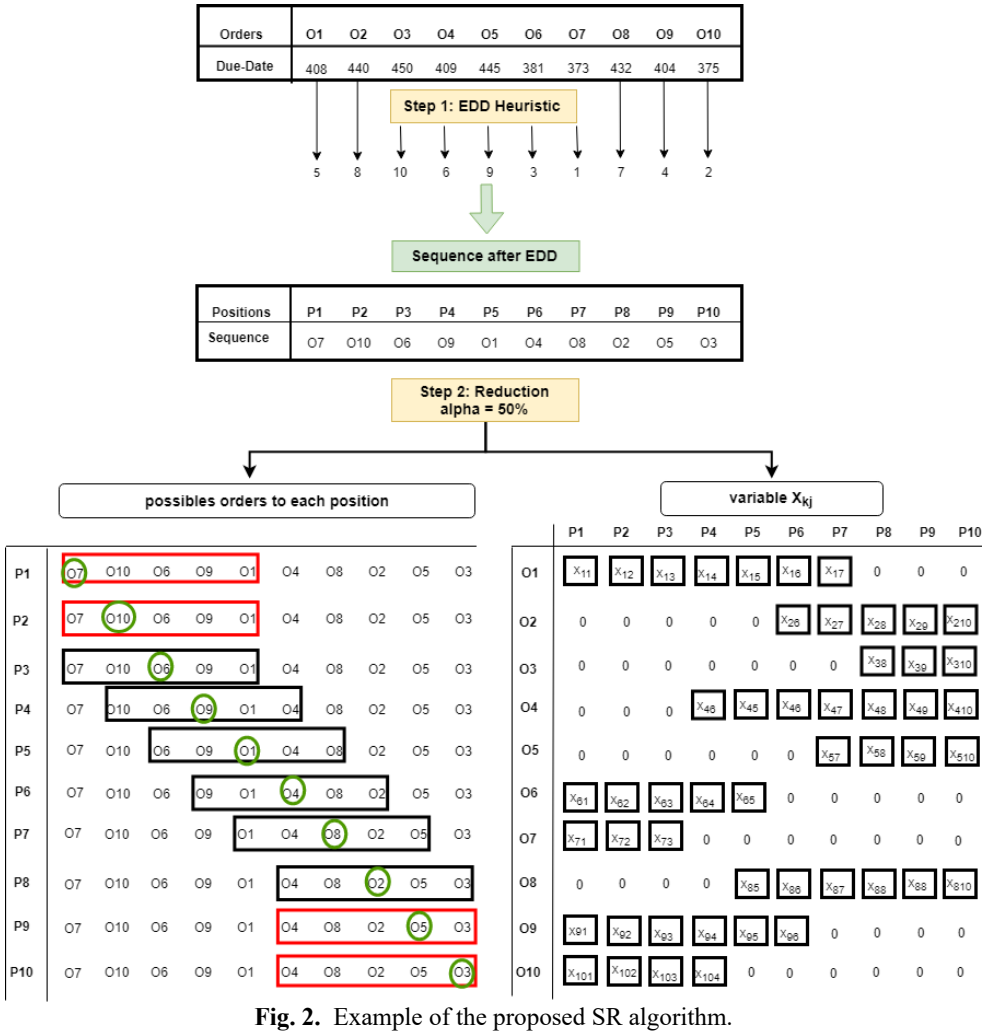

Fig. 2. Example of the proposed SR algorithm.

Algorithm SR α, t_{limit}
Let $\bar{x}$ contains the positional binaries variables corresponding to OMDD heuristic solution $\Pi\Pi_{best} := \Pi$
$x_{best} := \bar{x}$
$\alpha := \alpha \quad n$
$\alpha_h := round(\alpha/2)$
for $j := 1$ *to* n ***do***
$w_{min} := max(1, j \quad \alpha_h)$
if $w_{min} == 1$ **then**
$w_{max} := min(n, 1 + \alpha)$
end
else
$w_{max} := min(n, j + \alpha_h)$
end
if $w_{min} == n$ then
en
$w_{min} := n - \alpha_h$
d
end
for $k := 1$ *to* $= w_{min}$ **do**
$\bar{x}_{kj} := 0$
end
for $k := w_{max}$ *to* n **do**
$\bar{x}_{kj} := 0$
end
end

Solve the minimization of Eq. (1) subject to constraint sets (2), (3), (4), (5), and (6), as well as constraint (7), during t_{limit} seconds Store in Π the solution corresponding to solution x
if $x < x_{best}$ **then**
$x_{best} := x$
$\Pi_{best} := \Pi$
end
return x_{best}, Π_{best}

Fig. 3. Pseudocode of the proposed SR algorithm.

5. Computational experience

5.1 Test instances, statistics used in the computational experiments and methods under comparison

We evaluate the BIG test instances proposed by Framinan and Perez-Gonzalez (2018). For the BIG data set we have n ∈{100, 150, 200, 300} and $m \in \{5, 10\}$. These instances have two key parameters: the range of due dates (RDD) and the tardiness factor (TF). We use these parameters in the discussion of the results. We evaluate the BIG test instances proposed by Framinan and Perez-Gonzalez (2018). For the BIG data set we have $n \in \{100, 150, 200, 300\}$ and $m \in \{5, 10\}$. These instances have two key parameters: the range of due dates (RDD) and the tardiness factor (TF). We use these parameters in the discussion of the results. As the indicators for evaluation measure, we use the Relative Deviation Index(RDI) and the success rate. RDI is the usual indicator of quality for problems involving due dates (Fernandez-Viagas & Framinan, 2015). In this indicator, the total tardiness returned for a given method is compared with the best and the worst results obtained for all the methods under comparison. Mathematically, the RDI fora method $s \in H$ when applied to instance t is defined as follows:

$$RDI_{st} = \begin{cases} 0, & \text{if } \min_{h \in H} T_{ht} = \max_{h \in H} T_{ht}, \\ \dfrac{T_{st} - \min_{h \in H} T_{ht}}{\max_{h \in H} T_{ht} - \min_{h \in H} T_{ht}} \cdot 100, & \text{otherwise.} \end{cases} \tag{8}$$

Firstly, the proposed approach uses an initial solution based on the OMMD heuristic (Lee, 2013), where H = {MILP, JPO-20, SR} and T_{st} is the tardiness value obtained by method$s$ in instance t. In our case $min_{h \in H} T_{ht}$ is the best solution found among the methods in H. The Success Rate (SRa) is calculated as the number of times that a given rule results inthe best solution (with or without a draw) divided by the number of test instances in instance class. We consider the following methods in our computational experiments: mixed- integer programing problem (MILP), proposed by Framinan and Perez-Gonzalez (2018); JPO-20 algorithm, proposed by Framinan and Perez-Gonzalez (2018); and SR algorithm (our proposal). We implemented all the matheuristics using Julia

language (https://julialang.org/) within Atom IDE (https://atom.io/). For the pure MILP model as well as the matheuristics the commercial solver is the IBM ILOG CPLEX (https://www.ibm.com/products/ilog-cplex-optimization-studio) version 12.8. We perform the computational experience on a PC with Intel Core i5-3470 CPU 3.20GHz and 32GB memory. We use 600 seconds as a time limit for the MILP, JPO-20, and SR methods, as presented by Framinan and Perez-Gonzalez (2018). Since the MILP method is not efficient for large-sized instances, we use the OMDD heuristic (Lee, 2013) as a warm start for the JPO-20 and SR matheuristic. After several preliminary computational experiments, we determine the values of the parameter α following the test problem size. The size-reduction parameter is varying with the values of m and n, as illustrated in Table 2. For example, taking into account the test instances with $m = 5$ and $n = 100$, we adopt $\alpha = 0.8$, meaning that 80% of the decision variable values are not fixed. Concerning the calibration process of the parameter α, we perform the adjustment empirically. After several tests, we could observe that small-sized instances require a smaller reduction, and the large-sized instances require a greater reduction.

Table 2
Description of the parameters used in the proposed SR algorithm.

m	*n*	α
5	100	0.8
10	100	0.7
5	150	0.6
10	150	0.5
5	200	0.4
10	200	0.3
5	300	0.1
10	300	0.08

5.2 Results and discussion

Table 3 illustrates the results for the methods under comparison grouped by problem size. We can observe that SR presents the lowest average RDI (around 15.5%), whereas the JPO-20 presents an average RDI approximately three times higher (around 52.5 %). For the test instances with the lower values of m and n, MILP reaches the best results. However, for the large-sized instances, the performance of MILP drastically reduces. For the test instances with 10 machines and 300 orders, CPLEX is not able to find a feasible integer solution within the specified time limit. In contrast, JPO-20 and SR return feasible integer solutions for all the considered test instances. Considering the success rate indicator, we can observe that the SR algorithm returns a success rate value approximately 3 times higher than the JPO-20algorithm and two times higher than the MILP method. Table 4 presents the results for the different values of TF and RDD. Since CPLEX fails in the last set of test instances, the average values for the RDI indicator in Table 3 and Table 4 are different. Once again, there is evidence of the superiority of the SR algorithm in comparison with all the other evaluated methods.

Table 3
RDI and Success rate values for m and n

		MILP				**JPO-20**				**SR**			
m	*n*	Mean	St. Dev.	SRa	time (s)	Mean	St. Dev.	SRa	time (s)	Mean	St. Dev.	SRa	time (s)
5	100	9.92	26.31	77.78	462.39	48.96	45.57	32.22	607.54	28.61	41.44	53.33	444.82
	150	21.55	38.70	57.78	558.12	67.16	46.28	23.33	610.88	13.33	25.36	55.56	488.89
	200	55.52	44.90	21.67	592.70	51.30	43.29	22.22	615.19	7.27	25.83	83.33	507.03
	300	62.40	38.89	17.22	614.21	56.96	46.81	22.78	668.29	5.07	18.35	79.44	488.29
10	100	13.30	31.00	61.11	532.56	66.90	46.37	25.00	609.54	10.83	24.85	59.44	488.56
	150	76.78	39.55	2.78	601.31	23.89	32.44	41.11	621.54	20.62	40.18	71.11	533.61
	200	70.41	45.39	33.33	605.97	33.49	29.07	13.33	636.26	28.47	45.19	66.11	521.17
	300	-	-	-	-	71.25	45.40	28.33	676.65	10.00	30.09	88.33	503.63
TOTAL		44.27	37.82	38.81	566.75	52.49	41.90	26.04	630.74	15.53	31.41	69.58	497.00

Table 4
RDI and Success rate values for TF and RDD

		MILP	JPO-20	SR			MILP	JPO-20	SR			MILP	JPO-20
TF	RDD	Mean	St. Dev.	SRa	t (s)	Mean	St. Dev.	SRa	t (s)	Mean	St. Dev.	SRa	t (s)
0.2	0.2	51.72	46.96	26.43	549.12	47.69	45.14	18.57	629.22	2.19	13.72	95.00	532.50
	0.5	47.87	46.41	41.43	603.55	75.45	32.80	0.00	629.53	20.64	31.12	58.57	600.32
	0.8	18.25	31.03	55.71	603.37	80.05	30.90	5.00	628.68	35.64	47.17	39.29	600.30
0.5	0.2	66.82	46.69	31.43	498.52	7.54	22.97	68.57	639.55	1.35	9.92	96.43	251.00
	0.5	57.69	44.59	25.00	603.92	70.62	32.79	0.00	627.20	10.55	24.84	75.00	600.34
	0.8	14.68	28.10	48.57	604.46	67.29	40.33	14.29	626.17	43.82	47.16	37.14	600.32
0.8	0.2	52.86	50.10	47.14	429.29	0.00	0.00	100.00	640.34	0.00	0.05	99.29	87.56
	0.5	56.57	47.92	26.43	603.45	54.83	39.64	7.14	628.35	15.22	31.92	66.43	600.29
	0.8	16.96	30.18	47.14	605.08	61.61	44.69	17.86	627.61	44.84	46.46	35.00	600.39
Total		42.60	41.33	38.81	566.75	51.68	32.14	25.71	630.74	19.36	28.04	66.91	497.00

Fig. 4 illustrates the boxplots for average RDI values depending on the numbers of orders. In this figure, for each value of *n* we consider the test instances of 5 and 10 machines. We can observe that the MILP method returns the smaller RDI values for the test instances with 100 orders. For the test instances with 150, 200, and 300 orders, the SR algorithm returns lower RDI values than the MILP method as well asthe JPO-20 algorithm.

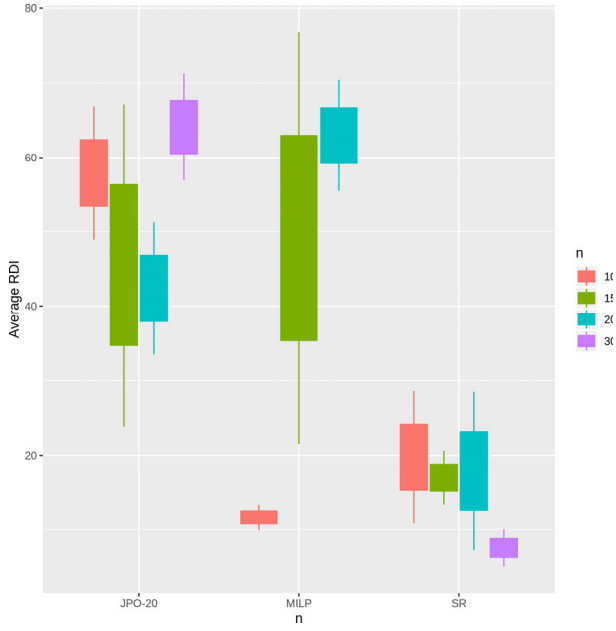

Fig. 4. Boxplots for average RDI values depending on *n* values

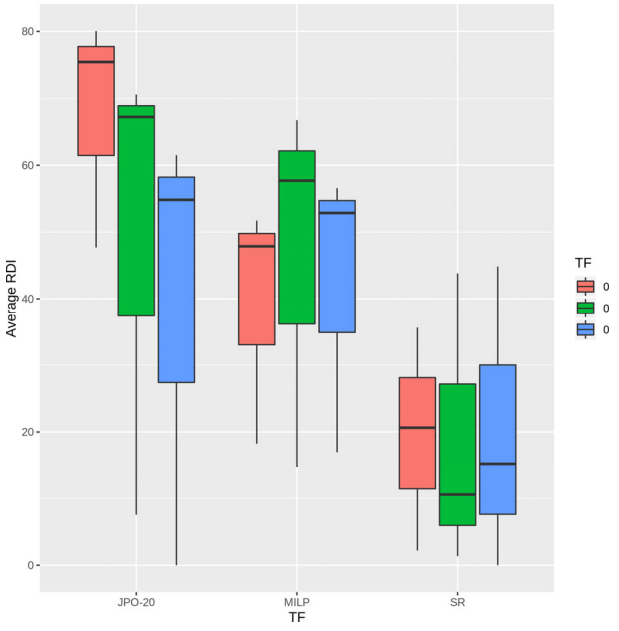

Fig. 5 Boxplots for average RDI values depending on TF

Fig. 5 illustrates the boxplots for the average RDI values depending on the TF. We can observe that the SR algorithm returns lower RDI values than the MILP method and the JPO-20 algorithm for the TF values equals to 0.2 and 0.5. However, for the test problems with TF equal to 0.8, there is no statistically difference between the SR and JPO-20 algorithms, whereas the MILP method returns worse RDI values than both algorithms. Fig. 6 illustrates the boxplots for the average RDI values depending on the RDD. We can observe that JPO-20 presents better average RDI values than the MILP method for the test instances with RDD equals to 0.2. However, for the test instances with RDD values equal to 0.5 and 0.8, the MILP method returns lower RDI values than RDI values than the SR algorithm for the test instances with RDD values equal to 0.8. Nevertheless, for the test instances with RDD values equal to 0.2 and 0.5, theSR algorithm returns lower RDI values than MILP method.

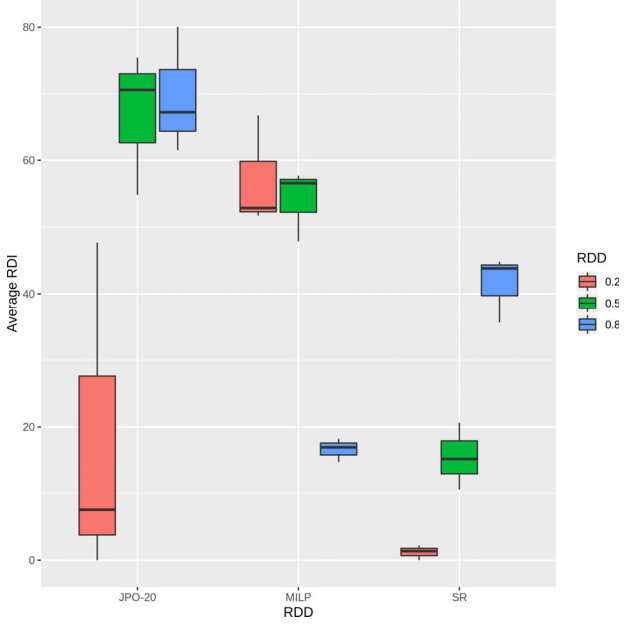

Fig. 6. Boxplots for average RDI values depending on RDD.

To validate the results, an ANOVA experiment is applied to verify the observed differences in the results of the local search algorithms are statistically significant. Since the *F* value is greater than the critical value, as illustrated in Tables 5, 7, and 9, a statistically significant difference between the methods under comparison is found. In these three tables, Df means the degrees of freedom, Sum

Sq means the sum of squares, and Mean Sq means the mean of squares. In Fig. 7, the mean plots with HSD Tukey intervals ($\alpha = 0.05$) of all evaluated methods are presented. Furthermore, Tables 6, 8, 10 illustrate the Tukey HSD results. Table 5 presents the ANOVA of average RDI values depending on TF and RDD, where F is a statistic that determines if the means of two or more populations are significantly different. We can observe that the F is greater than the critical value (in this case, $f = 3.4028$). Thus, it is possible to analyze which algorithms present a difference statistically significant using the Tukey test. According to Table 6, the only value greater than $\alpha = 0.05$ is found for the pair SR-JPO. We can emphasize that SR outperforms the JPO-20 algorithm. Furthermore, we cannot state that there is a statistical significative difference between all the other methods for a 95% confidence level.

Table 5
ANOVA of average RDI values depending on TF and RDD

Df	Sum Sq	Mean Sq	F value	Pr(>F)
2	5000	2500	4771	18
24	12575	524		

Table 6
Tukey HSD 95% confidence of average RDI values for TF and RDD.

	Mean Difference	Lower bound	Upper bound
MILP-JPO20	-9.074944	-36.02176	17.871873
SR-JPO20	-32.313200	-59.26002	-5.366383
SR-MILP	-23.238256	-50.18507	3.708561

One can observe that there are statistically significant differences between the average RDI values among the SR algorithm and the JPO-20 algorithm, as illustrated in Tables 6 and 8. Therefore, the SR algorithm outperforms the JPO-20 algorithm taking into consideration the evaluated test instances. In addition, we can emphasize that the SR algorithm also outperforms the MILP method for RDD values of 0.5 and 0.8, as illustrated in Table 10. Table 7 presents the ANOVA of average RDI values depending on TF. The critical f values for TF equals to 0.2, 0.5, and 0.8 are the same ($f = 5.1433$). Thus, taking into consideration the Tukey HSD values illustrated in Table 8, the single value greater than $\alpha = 0.05$ is for the pair SR-JPO with TF=0.2. Therefore, there is a statistically significant difference between these methods in this case. For all the other situations, we cannot conclude if there is a statistically significant difference for a confidence level of 95%.

Table 7
ANOVA of average RDI values depending on TF.

TF	Df	Sum Sq	Mean Sq	F value	Pr(>F)
0.2	2	3528	1764.1	5737	0.0405
	6	1845	307.5		
0.5	2	1673	836.4	0.99	425
	6	5070	844.9		
0.8	2	853	426.4	598	0.58
	6	4280	713.3		

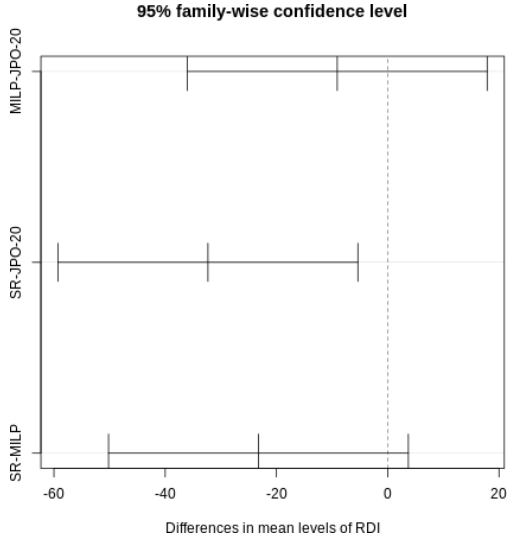

Fig. 7. Tukey HSD intervals at the 95% confidence level for the analyzed methods

Table 8
Tukey HSD 95% confidence of average RDI values for TF

TF		Mean Difference	Lower bound	Upper bound	Sig
0.2	JPO0.2-MILP0.2	28.45253	-15.48030	72.385369	0.1961116
	0.2 SR0.2-MILP0.2	-19.78817	-63.72100	24.144669	0.4067639
	SR0.2-JPO0.2	-48.24070	-92.17354	-4.307864	0.0346983
0.5	JPO0.5-MILP0.5	2.088833	-70.73288	74.91055	0.9957417
	0.5 SR0.5-MILP0.5	-27.819933	-100.64165	45.00178	0.5098670
	SR0.5-JPO0.5	-29.908767	-102.73048	42.91295	0.4649038
0.8	JPO0.8-MILP0.8	-3.316533	-70.22380	63.59073	0.9873552
	0.8 SR0.8-MILP0.8	-22.106667	-89.01393	44.80060	0.5957792
	SR0.8-JPO0.8	-18.790133	-85.69740	48.11713	0.6817528

Table 9 illustrates the ANOVA for average RDI values depending on RDD. The critical values for the RDD levels of 0.2, 0.5, and 0.8 are the same (in this case, $f = 5.1433$). Considering the Tukey test, as presented in Table 10, we find a differ $RDD = 0.2$. For $RDD = 0.5$, the SR outperforms the MILP method and the JPO-20 algorithm. For $RDD = 0.8$, SR outperforms the MILP method and the JPO-20 algorithm.

Table 7
ANOVA of average RDI values depending on RDD

RDD	Df	Sum Sq	Mean Sq	F value	Pr(>F)
0.2	2	4926	2463.2	10.13	0.0119
	6	1459	243.1		
0.5	2	4307	2153.4	37.83	0.000397
	6	342	56.9		
0.8	2	4223	2111.3	53.75	0.000148
	6	236	39.3		

Furthermore, the MILP method outperforms the JPO-20 algorithm. For all the other cases, we cannot state if there is a difference statistically significant between the methodsunder comparison for a confidence level of 95%. Concerning the computation times, the methods under study present a distinct behavior, as illustrated in Table 3. The MILP method, the JPO-20 algorithm, and the SR algorithm present average computational times of 566.8s, 630.7s, and 497.0, respectively. Although we adopt a time limit of 600s, in some cases the CPLEX presents an imprecision in controlling this time because of pre-solve function. Becauseof this imprecision, JPO-20 presents an average computational time greater than thespecified time limit. We can observe that the SR algorithm uses a lower computational effort than all the other methods under comparison. In comparison with the JPO-20 algorithm, theproposed SR algorithm returns an average RDI approximately three times smaller, with a resultant reduction of the computational times of approximately 22%. The computational experience carried out shows that our proposal outperforms the JPO-20 algorithm and can provide high-quality results within admissible CPU times.

6. Conclusions

In this paper, we investigate the customer order scheduling problem, and the objective function is to minimize the total tardiness. We develop a size-reduction matheuristic that led to excellent results within an admissible computational effort. The results of the proposed approach are presented taking into consideration the literature benchmark instances presented by Framinan and Perez-Gonzalez (2018). We used the relative deviation index statistic and Success rate as the performance measures. Considering the above mentioned literature benchmark instances, the proposed size-reduction algorithm outperforms the JPO-20 matheuristic proposed by Framinan and Perez-Gonzalez (2018). The proposed algorithm finds better solutions than the JPO-20 algorithm, using lower computational times. As extensions of this work, we suggest the consideration of explicit setup times in the customer order scheduling. Meta-heuristics could be proposed for the resolution of the problem under study. In addition, future studies could also investigate the behavior of the proposed approaches considering other objective functions, such as total completion time minimization or a just-in time environment.

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

Ahmadi, R., Bagchi, U., & Roemer, T. A. (2005). Coordinated scheduling of customer orders for quick response. *Naval Research Logistics (NRL)*, *52*(6), 493-512.

Antonioli, M., Rodrigues, C., & Prata, B. (2022). Minimizing total tardiness for the order scheduling problem with sequence-dependent setup times using hybrid matheuristics. *International Journal of Industrial Engineering Computations*, *13*(2), 223-236.

Fanjul-Peyro, L., & Ruiz, R. (2011). Size-reduction heuristics for the unrelated parallel machines scheduling problem. *Computers & Operations Research*, *38*(1), 301-309.

Fanjul-Peyro, L., Perea, F., & Ruiz, R. (2017). Models and matheuristics for the unrelated parallel machine scheduling problem with additional resources. *European Journal of Operational Research*, *260*(2), 482-493.

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

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

Julien, F. M., & Magazine, M. J. (1990). Scheduling customer orders: An alternative production scheduling approach. *Journal of Manufacturing and Operations Management*, *3*(3), 177-199.

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. (2005). Order scheduling in an environment with dedicated resources in parallel. *Journal of Scheduling*, *8*(5), 355-386.

Lin, B. M., & Kononov, A. V. (2007). Customer order scheduling to minimize the number of late jobs. *European Journal of Operational Research*, *183*(2), 944-948.

Lin, W. C., Yin, Y., Cheng, S. R., Cheng, T. E., Wu, C. H., & Wu, C. C. (2017). Particle swarm optimization and opposite-based particle swarm optimization for two-agent multi-facility customer order scheduling with ready times. *Applied Soft Computing*, *52*, 877-884.

Prata, B. D. A., de Abreu, L. R., & Lima, J. Y. F. (2021). Heuristic methods for the single-machine scheduling problem with periodical resource constraints. *Top*, *29*(2), 524-546.

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

de Athayde Prata, B., Rodrigues, C. D., & Framinan, J. M. (2022). A differential evolution algorithm for the customer order scheduling problem with sequence-dependent setup times. *Expert Systems with Applications*, *189*, 116097.

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

Shi, Z., Wang, L., Liu, P., & Shi, L. (2015). Minimizing completion time for order scheduling: Formulation and heuristic algorithm. *IEEE Transactions on Automation Science and Engineering*, *14*(4), 1558-1569.

Sung, C. S., & Yoon, S. H. (1998). Minimizing total weighted completion time at a pre-assembly stage composed of two feeding machines. *International Journal of Production Economics*, *54*(3), 247-255.

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

Wang, G., & Cheng, T. E. (2007). Customer order scheduling to minimize total weighted completion time. *Omega*, *35*(5), 623-626.

Xu, J., Wu, C. C., Yin, Y., Zhao, C., Chiou, Y. T., & Lin, W. C. (2016). An order scheduling problem with position-based learning effect. *Computers & Operations Research*, *74*, 175-186.

Xu, X., Ma, Y., Zhou, Z., & Zhao, Y. (2013). Customer order scheduling on unrelated parallel machines to minimize total completion time. *IEEE Transactions on Automation Science and Engineering*, *12*(1), 244-257.

Yang, J., & Posner, M. E. (2005). Scheduling parallel machines for the customer order problem. *Journal of Scheduling*, *8*(1), 49-74.