POLITECNICO DI TORINO
Repository ISTITUZIONALE

Comparing mathematical formulations for customer order scheduling with dedicated machines

Original
Comparing mathematical formulations for customer order scheduling with dedicated machines / Alfieri, A., Grosso, A., Pastore, E., Salassa, F.. - In: ENGINEERING OPTIMIZATION. - ISSN 0305-215X. - ELETTRONICO. - (2026). [10.1080/0305215X.2026.2691603]

Availability:
This version is available at: 11583/3013599 since: 2026-09-09T11:51:11Z

Publisher:
Taylor and Francis

Published
DOI:10.1080/0305215X.2026.2691603

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

25 October 2026

Engineering Optimization

ISSN: 0305-215X (Print) 1029-0273 (Online) Journal homepage: www.tandfonline.com/journals/geno20

Comparing mathematical formulations for customer order scheduling with dedicated machines

Arianna Alfieri, Andrea Grosso, Erica Pastore & Fabio Salassa

To cite this article: Arianna Alfieri, Andrea Grosso, Erica Pastore & Fabio Salassa (27 Jul 2026): Comparing mathematical formulations for customer order scheduling with dedicated machines, Engineering Optimization, DOI: 10.1080/0305215X.2026.2691603

To link to this article: https://doi.org/10.1080/0305215X.2026.2691603

© 2026 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.

Published online: 27 Jul 2026.

Submit your article to this journal

View related articles

View Crossmark data

Full Terms & Conditions of access and use can be found at
https://www.tandfonline.com/action/journalInformation?journalCode=geno20

ENGINEERING OPTIMIZATION
https://doi.org/10.1080/0305215X.2026.2691603

Taylor & Francis by informa

OPEN ACCESS Check for updates

Comparing mathematical formulations for customer order scheduling with dedicated machines

Arianna Alfieri [a], Andrea Grosso [b], Erica Pastore [a] and Fabio Salassa [a]

[a]Department of Management and Production Engineering, Politecnico di Torino, Torino, Italy; [b]Department of Mathematics 'G. Peano', Università di Torino, Torino, Italy

ABSTRACT
The Customer Order Scheduling (COS) problem arises when performance measures are defined at the order level, and each order consists of multiple jobs. This article studies the COS problem with dedicated machines, aiming to minimize the total completion time of orders. Four mathematical formulations are examined and compared: three Mixed-Integer Linear Programming (MILP) models based on positional, precedence and time-indexed variables, and a Constraint Programming (CP) formulation, which is introduced for the first time in the literature. Computational experiments on benchmark instances show that the models provide good balances between solution quality and implementation complexity.

ARTICLE HISTORY
Received 30 October 2025
Accepted 12 June 2026

KEYWORDS
Customer order scheduling; Dedicated machines; Constraint programming; MILP formulation

1. Introduction

Scheduling problems arise each time resources must be allocated to the entities (*e.g.* items, jobs, customers, depending on the system) flowing in the system, *i.e.* when time is the scarce resource. In manufacturing systems, they are usually related to the sequencing and timing of jobs (the entities that flow through the system) on machines (the system resources). In other words, for each machine, the sequence of jobs to be processed and when the processing of each of them has to start need to be defined.

Many types of scheduling problem exist, depending on the system layout, processing/service constraints, performance measures, *etc.* All such characteristics influence the difficulty of the problems, and most real/realistic scheduling problems are NP-hard. Mentioning all the constraints used in the scheduling problems that have been addressed in the literature is not an easy task owing to their variety, which is strictly problem-dependent. Instead, it is possible to discuss the possible layouts and performance measures (or objective functions).

Considering layouts, scheduling problems generally refer to five types of architecture: single machine (S. K. Gupta and Kyparisis 1987), parallel machines (T. C. E. Cheng and Sin 1990), flow shop (Komaki, Sheikh, and Malakooti 2019; Rossit, Tohmé, and Frutos 2018), job shop (Xiong *et al.* 2022) and open shop (Pastore *et al.* 2022). Single-machine problems are the easiest; the others, depending on the specific layout characteristics (types of resources and technological and/or management constraints, objective functions) can achieve high levels of complexity, and most of them are already known to be NP-hard. The variety of objective functions also contributes to the problem complexity. The most used ones are related to the completion time: makespan (Framinan, J. N. D. Gupta, and Leisten 2004; Hejazi and Saghafian 2005), total completion time (Janiak and Krysiak 2012; Mason

CONTACT erica.pastore@polito.it

© 2026 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.
This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. The terms on which this article has been published allow the posting of the Accepted Manuscript in a repository by the author(s) or with their consent.

and J.-S. Chen 2010), total tardiness (Koulamas 1994), maximum tardiness (Rostami, Kheirandish, and Ansari 2015), *etc.*

Usually, scheduling problems assume that the units to be scheduled (*i.e.* the units flowing in the system) are *jobs*, which can be single items (a job is an item) or batches of items (a job is composed of several items); the parameters are usually given at the job level (*e.g.* processing time, release time), and the performance measures are computed at the job level.

Instead, in the Customer Order Scheduling (COS) problem (Yang 2005), the units flowing in the system are *orders*, and each order is composed of several jobs processed on different machines. In this context, some parameters are order-dependent (*e.g.* release time, due date), while others are job-dependent (*e.g.* processing time). The performance measures are order-dependent, as all the jobs belonging to the order must be completed before the order can leave the system (*e.g.* to be delivered to the customer).

COS problems are very important from a practical point of view, especially in companies dealing with multiple products. Examples can be found in the process industry or in the production stages of assembly facilities. In the process industry, each product (*e.g.* aluminium sheets, paper items) can be produced in various sizes from the same raw material (*e.g.* aluminium coils, paper sheet). Each customer can require, in the same order, different quantities of more than one product type (*e.g.* different amounts of various aluminium sheets, different quantities of various paper products). In assembly facilities, the customer order may be quantities of a given finished product. However, each product must be assembled from different components in given quantities. In these cases, customers usually require more than one type of job in the same order (*e.g.* different paper items, aluminium sheets or assembled products in the previously cited examples), and the different jobs are generally processed on different machines (because they need different processing technologies or machine settings that require too long to be changed continuously). However, customers require a unique delivery for the whole order (as soon as all the jobs are completed), and cannot accept many deliveries of single jobs.

The COS problem has two components: the allocation of the jobs of each order to the different machines and the scheduling of the allocated jobs on each machine. When each job has a dedicated machine, the problem has only the scheduling component, as allocation is no longer needed.

In this article, the COS problem with dedicated parallel machines is studied, with the aim of minimizing the total order completion time. For such problems, there always exists an optimal permutation solution (Leung, H. Li, and Pinedo 2005); thus this article addresses the permutation version of the problem (*i.e.* all the dedicated machines process the orders in the same sequence). Different formulations are proposed and compared on randomly generated instances. Specifically, three mathematical programming formulations (a Mixed-Integer Linear Programming (MILP) model based on positional variables, a MILP model based on precedence variables, and a time-indexed MILP) and a Constraint Programming (CP) formulation are addressed. The positional MILP formulation is adapted from existing models in the literature (Framinan and Perez-Gonzalez 2018; Hoffmann, Neufeld, and Buscher 2022). Also, precedence-based and time-indexed formulations have been studied previously (respectively, in Mastrolilli *et al.* 2010 and G. Wang and T. C. E. Cheng 2007); however, the models proposed here differ in their structure and adaptation to the problem under consideration. In contrast, to the best of the authors' knowledge, the CP formulation has not previously been introduced for this problem and, therefore, it constitutes a novel contribution. Furthermore, a comprehensive computational comparison of these alternative formulations on different benchmark instances available in the literature (Shi *et al.* 2021) has not been conducted before. The comparative assessment provided in this article therefore represents an additional contribution, offering insights into the relative performance and suitability of each modelling approach.

The remainder of the article is organized as follows. Section 2 discusses the relevant literature on COS problems. The formal description of the problem and the proposed formulations are presented in Section 3. Results are reported in Section 4. Section 5 concludes the article.

2. Literature review

The following review of the literature on the COS problem only focuses on single and parallel machines. Other environments are beyond the scope of this article; thus, the related literature is not analysed.

Table 1 shows a classification summary of the reviewed literature. In the table, the reviewed articles are displayed in rows, and each column is a characteristic of the problem. The columns show: the addressed article (Reference); the considered machines (single or parallel) included in the studied environment (Machine); some system characteristics (Setup, Due dates, Release dates, Other); the considered objective function (makespan $c_{\max}$, total completion time $\sum_j C_j$, maximum lateness $L_{\max}$, total lateness $\sum_j L_j$, total tardiness $\sum_j T_j$, total number of tardy orders $\sum_j U_j$, Other); and the solution method (Exact, Approximate).

2.1. Single machine

As shown in the top part of Table 1, the customer order scheduling problem with single machines (referenced with'1' in the column headed 'Machine' in the table) is addressed in Erel and Ghosh (2007), Çetinkaya, Yeloğlu, and Çatmakaş (2021), Hazır, Günalay, and Erel (2008), G. Liu, Xie, and H. Wang (2025), L. Li *et al.* (2023), C.-C. Wu *et al.* (2024), J. N. D. Gupta *et al.* (2023) and W.-C. Lin *et al.* (2023). In such articles, each order is composed of a set of jobs, and jobs belong to families, all to be produced on the single available machine; every time two jobs of different families are produced one after the other, a setup time is needed (*i.e.* sequence-independent setup times are considered, and denoted as 's.i.' in Table 1; sequence-dependent setup times are denoted as 's.d.'). Erel and Ghosh (2007) proved the problem with the objective function represented by the minimization of the total completion time to be NP-hard. They also proposed a dynamic programming exact approach to solve such a problem. The same problem was studied by Çetinkaya, Yeloğlu, and Çatmakaş (2021) and Hazır, Günalay, and Erel (2008), who only considered approximate approaches. In Çetinkaya, Yeloğlu, and Çatmakaş (2021) addressed specific cases of the single-machine COS: heuristics were used to minimize the total completion time on a single machine when either jobs of the same type or jobs of the same customer order had to be processed together. Hazır, Günalay, and Erel (2008) tested various meta-heuristic algorithms to solve the single-machine COS to minimize the average flow time (which is strictly related to the total completion time as the release dates are zero).

Weighted sums of various objective functions were minimized by G. Liu, Xie, and H. Wang (2025), L. Li *et al.* (2023), C.-C. Wu *et al.* (2024), J. N. D. Gupta *et al.* (2023) and W.-C. Lin *et al.* (2023), who also included the minimization of the holding cost ($\sum_j H_j$ in Table 1), while G. Liu, Xie, and H. Wang (2025) proposed a genetic algorithm to solve the problem; all the others proposed branch-and-bound algorithms based on different lower bounds, and some heuristic methods.

As shown in the middle part of Table 1, some authors addressed the COS in a parallel-machine environment. In this system, parallel machines can be either dedicated (PD), identical (P) or unrelated (R). In the following, first the literature on mathematical model formulations and solution methods for each environment is reviewed, then results on the complexity of some problem versions are discussed. To support the discussion about mathematical model formulations, Table 2 summarizes the mathematical formulations addressed in the literature on the parallel-machine environment. In the table, references are separated for dedicated and non-dedicated parallel machines (in columns), and for the type of variables used in the mathematical formulations (in rows).

2.2. Dedicated parallel machines

The parallel dedicated machine COS (PD-COS) problem has also been referred to in the literature as the *concurrent open shop scheduling problem* (Bukchin, Hanany, and Gerchak 2022; T. C. E. Cheng, Nong, and Ng 2011; Mastrolilli *et al.* 2010), and it has been mainly solved through approximate algorithms.

Table 1. COS classification summary.

		System characteristics				Objective function							Solution method	
Reference	Machine	Setup	Due dates	Release dates	Other	$c_{\max}$	$\sum_j C_j$	$L_{\max}$	$\sum_j L_j$	$\sum_j T_j$	$\sum_j Uj$	Other	Exact	Approximate
Erel and Ghosh (2007)	1	✓(s.i.)					✓						✓	
Çetinkaya, Yeloğlu, and Çatmakaş (2021)	1	✓(s.i.)					✓							✓
Hazır, Günalay, and Erel (2008)	1	✓(s.i.)					✓							✓
G. Liu, Xie, and H. Wang (2025)	1	✓(s.i.)	✓		s-batch, size		✓			✓		$\sum_j H_j$		✓
L. Li *et al.* (2023)	1	✓(s.i.)	✓				✓			✓		$\sum_j H_j$	✓	✓
C.-C. Wu *et al.* (2024)	1	✓(s.i.)				✓	✓					$\sum_j H_j$	✓	✓
J. N. D. Gupta *et al.* (2023)	1	✓(s.i.)				✓						$\sum_j H_j$	✓	✓
W.-C. Lin *et al.* (2023)	1	✓(s.i.)	✓						✓			$\sum_j H_j$	✓	✓
de Athayde Prata, Rodrigues, and Framinan (2021)	PD	✓(s.d.)				✓								✓
Roemer (2006)	PD						✓							
Framinan and Perez-Gonzalez (2017)	PD						✓							✓
Riahi *et al.* (2019)	PD						✓							✓
Hoffmann, Neufeld, and Buscher (2022)	PD						✓							✓
Luo *et al.* (2016)	PD						✓							✓
Shi *et al.* (2021)	PD						✓							✓
Borisovsky, Zakharov, and Zakharova (2025)	PD	✓(s.i.)					✓							✓
de Athayde Prata, Rodrigues, and Framinan (2022)	PD	✓(s.d.)					✓							✓
Shi *et al.* (2015)	PD						✓(w)							✓
G. Wang and T. C. E. Cheng (2007)	PD						✓(w)							✓
Mastrolilli *et al.* (2010)	PD						✓(w)							✓
T. C. E. Cheng, Nong, and Ng (2011)	PD						✓(w)							✓
Bukchin, Hanany, and Gerchak (2022)	PD						✓(w)							✓
Leung, H. Li, and Pinedo (2005)	PD		✓	✓			✓(w)			✓(w)	✓(w)			✓

(*continued*).

Table 1. Continued.

		System characteristics				Objective function							Solution method	
Reference	Machine	Setup	Due dates	Release dates	Other	$c_{\max}$	$\sum_j C_j$	$L_{\max}$	$\sum_j L_j$	$\sum_j T_j$	$\sum_j Uj$	Other	Exact	Approximate
W.-C. Lin *et al.* (2019)	PD		✓	✓							✓(w)		✓	✓
Ng, T. C. E. Cheng, and Yuan (2003)	PD		✓		$d_j = d$						✓(w)			✓
B. M. Lin and Kononov (2007)	PD		✓		$d_j = d$						✓(w)			✓
Framinan and Perez-Gonzalez (2018)	PD									✓				✓
L.-Y. Li *et al.* (2022)	PD		✓	✓								$\max_s \min \sum_j w_j U_j$	✓	✓
Zheng *et al.* (2024)	PD		✓		*rej*							$D_{\max}(A) + E(R)$		✓
Hou *et al.* (2023)	PD		✓		*rej*							$D_{\max}(A) + E(R)$		✓
Hoffmann, Neufeld, and Buscher (2024)	PD		✓									$\sum_j T_j + \sum_j E_j$		✓
Shi, Huang, and Shi (2018)	1, P		✓		batch	✓	✓(w)							✓
Yang (2005)	P		✓			✓	✓(w)	✓						
Su, P.-S. Chen, and S.-Y. Chen (2013)	P		✓		✓									✓
C.-C. Wu *et al.* (2021)	P		✓									$\max_s \min \sum_j T_j$	✓	✓
X. Xu *et al.* (2013)	R						✓							✓
H. Li *et al.* (2021)	R						✓(w)							✓
Zipfel, Neufeld, and Buscher (2024)	R	✓(s.d.)	✓		nesting, size		✓(w)							✓
Zhao *et al.* (2021)	R			✓	res. alloc.		✓(w)					E[cycle time]		✓
Zhao *et al.* (2022)	R			✓	res. alloc.		✓(w)					E[cycle time]		✓
X. Xu *et al.* (2025)	R			✓	res. alloc.							E[cycle time]		✓
This article	PD						✓						✓	

Table 2. Mathematical model formulations available in the literature for parallel machines.

Model	Dedicated	Non-dedicated
Positional	Shi *et al.* (2021), de Athayde Prata, Rodrigues, and Framinan (2021, 2022), Hoffmann, Neufeld, and Buscher (2022, 2024), Shi *et al.* (2015), Framinan and Perez-Gonzalez (2018), Zheng *et al.* (2024) and Hou *et al.* (2023), This article	Shi, Huang, and Shi (2018) and Zipfel, Neufeld, and Buscher (2024)
Precedence	Mastrolilli *et al.* (2010), This article	X. Xu *et al.* (2013) and H. Li *et al.* (2021)
Time-indexed	G. Wang and T. C. E. Cheng (2007), This article	
Constraint programming	This article	
Other	Luo *et al.* (2016) (coflow)	

The problem that minimizes the $\sum_j C_j$ objective function was addressed by Shi *et al.* (2021), Roemer (2006), Framinan and Perez-Gonzalez (2017), Riahi *et al.* (2019), Hoffmann, Neufeld, and Buscher (2022), Luo *et al.* (2016), Borisovsky, Zakharov, and Zakharova (2025) and de Athayde Prata, Rodrigues, and Framinan (2022). Mathematical formulations of this problem were presented by Hoffmann, Neufeld, and Buscher (2022), Shi *et al.* (2021), Luo *et al.* (2016) and de Athayde Prata, Rodrigues, and Framinan (2022). The formulations of Hoffmann, Neufeld, and Buscher (2022), Shi *et al.* (2021) and de Athayde Prata, Rodrigues, and Framinan (2022) employed positional variables, whereas Luo *et al.* (2016) proposed a coflow-based formulation. Shi *et al.* (2021) introduced additional variables to represent job starting times on machines. Hoffmann, Neufeld, and Buscher (2022) adapted the formulation of Framinan and Perez-Gonzalez (2018), originally developed for total tardiness minimization, to total completion time minimization. The first three formulations relied on the permutation property of the problem; in contrast, de Athayde Prata, Rodrigues, and Framinan (2022) did not assume this property, as it incorporates sequence-dependent setup times. Problem complexity was discussed by Roemer (2006), while the others proposed solution algorithms. Specifically, constructive heuristics were implemented by Framinan and Perez-Gonzalez (2017), Riahi *et al.* (2019) and Leung, H. Li, and Pinedo (2005); approximate search algorithms were presented by Framinan and Perez-Gonzalez (2017), Riahi *et al.* (2019), Hoffmann, Neufeld, and Buscher (2022) and Luo *et al.* (2016); genetic algorithms were proposed by Shi *et al.* (2021) and Borisovsky, Zakharov, and Zakharova (2025); and an evolutionary algorithm was developed by de Athayde Prata, Rodrigues, and Framinan (2022) for the special case of sequence-dependent setups. Also, Luo *et al.* (2016) showed that the $PD \,||\, \sum_j C_j$ problem is equivalent to the problem of minimizing the average Coflow Completion Time (CCT) of a set of coflows (which is a problem usually solved for data centre networks).

The more general $PD \,||\, \sum_j w_j C_j$ problem was addressed by Leung, H. Li, and Pinedo (2005), Shi *et al.* (2015), G. Wang and T. C. E. Cheng (2007), Mastrolilli *et al.* (2010), T. C. E. Cheng, Nong, and Ng (2011) and Bukchin, Hanany, and Gerchak (2022). Among these, Mastrolilli *et al.* (2010), G. Wang and T. C. E. Cheng (2007) and Shi *et al.* (2015) proposed mathematical formulations of the problem. A quadratic formulation together with an equivalent mixed-integer linear model were proposed by Shi *et al.* (2015), and both used positional variables. A time-indexed formulation was formalized by G. Wang and T. C. E. Cheng (2007), and a relaxation with interval variables was proposed to find a lower bound of the solution of the original problem. Instead, precedence variables were used in the formulation of Mastrolilli *et al.* (2010) to devise linear approximations.

To solve the $PD \,||\, \sum_j w_j C_j$ problem, Shi *et al.* (2015) proposed a mixed-integer quadratic programming model, several heuristics and a local search. An approximation algorithm based on a linear programming relaxation was proposed by G. Wang and T. C. E. Cheng (2007), a combinatorial one by Mastrolilli *et al.* (2010), and a polynomial time approximation scheme by T. C. E. Cheng, Nong, and Ng (2011). Instead, the $PD \,||\, \sum_j w_j C_j$ problem was modelled as a non-cooperative game for two environments by Bukchin, Hanany, and Gerchak (2022), and coordinating mechanisms were proposed.

The special case of the COS problem with makespan minimization and sequence-dependent setups was discussed by de Athayde Prata, Rodrigues, and Framinan (2021), and solved through two math-heuristics; also, two mathematical formulations were proposed, one extending the formulation of Framinan and Perez-Gonzalez (2018), the other considering the number of possible permutations as a parameter given by the user.

Minimization of the weighted number of tardy jobs, *i.e.* $PD \,||\, \sum_j w_j U_j$, was addressed by Leung, H. Li, and Pinedo (2005), W.-C. Lin *et al.* (2019), Ng, T. C. E. Cheng, and Yuan (2003) and B. M. Lin and Kononov (2007). Two heuristic algorithms were proposed by Leung, H. Li, and Pinedo (2005) for the weighted and unweighted cases of $PD \,||\, \sum_j w_j U_j$, while bee colony meta-heuristics were used by W.-C. Lin *et al.* (2019) to solve the problem with release dates (r_j), *i.e.* $PD|r_j|\sum_j w_j U_j$. An exact branch-and-bound approach was proposed by W.-C. Lin *et al.* (2019) to solve the $PD|r_j|\sum_j w_j U_j$ problem. Instead, total tardiness minimization was addressed by Framinan and Perez-Gonzalez (2018), in which a math-heuristic algorithm based on sliding windows was proposed.

The specific problem variant $PD|d_j = d|\sum_j w_j U_j$ (where $d_j = d$ indicates common due dates) was addressed by Ng, T. C. E. Cheng, and Yuan (2003) and B. M. Lin and Kononov (2007): an LP-rounding approximate algorithm is proposed in Ng, T. C. E. Cheng, and Yuan (2003), while B. M. Lin and Kononov (2007) proved that for $PD2|d_j = d|\sum_j w_j C_j$ no FPTAS exists unless $P = \mathrm{NP}$.

The robust PD-COS problem that maximizes the weighted sum of tardy jobs $PD|r_j|\max_s \min \sum_j w_j U_j$ between two scenarios was addressed both with a branch-and-bound algorithm and some genetic heuristic algorithms by L.-Y. Li *et al.* (2022).

Moreover, some authors have included the possibility of rejecting orders and addressed the minimization of both the maximum delivery time and the sum of rejected order penalties ($PD|rej|D_{\max}(A) + E(R)$); such problems were formulated by MILP models using positional variables, and solved by approximation algorithms by Zheng *et al.* (2024) and Hou *et al.* (2023). Other authors have addressed the problem of minimizing earliness–tardiness (Hoffmann, Neufeld, and Buscher 2024) by proposing, also in this case, MILP formulations based on positional variables.

2.3. Non-dedicated parallel machines

The COS problem has also been studied for non-dedicated parallel machines. Parallel machines were assumed to be identical (P) by Shi, Huang, and Shi (2018), Yang (2005), Su, P.-S. Chen, and S.-Y. Chen (2013), C.-C. Wu *et al.* (2021) and unrelated (R) by X. Xu *et al.* (2013), H. Li *et al.* (2021), Zipfel, Neufeld, and Buscher (2024), Zhao *et al.* (2021, 2022) and X. Xu *et al.* (2025).

The maximum lateness objective function (*i.e.* problem $P \,||\, L_{\max}$) was solved approximately by Su, P.-S. Chen, and S.-Y. Chen (2013) through constructive heuristics, while some properties for such problems were devised by Yang (2005). A special case of the identical parallel machine environment was considered by Shi, Huang, and Shi (2018), who took scheduling and batching decisions together to minimize the makespan and the weighted sum of completion times; a mathematical formulation based on positional variables was proposed, and several heuristic algorithms were developed for the case of weighted total completion time, while for the makespan they proved that the problem can be solved polynomially both with single and parallel batching machines. The stochastic version of the COS problem in which processing times vary in scenarios was addressed by C.-C. Wu *et al.* (2021). This article minimized the maximum, over all scenarios, of the minimum total tardiness and, to this end, a branch-and-bound and several heuristic algorithms were proposed.

Unrelated parallel machines imply having parallel machines each operating at a different speed. In such an environment, heuristic algorithms were proposed by X. Xu *et al.* (2013) to minimize the sum of completion time, and by H. Li *et al.* (2021) to minimize $\sum_j w_j C_j$; both these articles used mathematical formulations based on precedence variables to represent the problem. An application of the COS problem in additive manufacturing was considered by Zipfel, Neufeld, and Buscher (2024), who characterized the system by unrelated machines, jobs with two sizes (height, area) and sequence-dependent setup times; also, in this problem, nesting decisions were considered, besides scheduling,

to minimize the total weighted tardiness of orders. A mathematical formulation based on positional variables wass proposed, and an iterated local search algorithm was implemented. Lastly, a stochastic version of the COS problem was considered by Zhao *et al.* (2021, 2022) and X. Xu *et al.* (2025), who made scheduling decisions together with resource allocation decisions to minimize the long run expected order cycle time in an unrelated parallel machine environment with stochastic release dates.

2.4. Complexity results

The COS problem with two parallel machines and $\sum_j C_j$ or $\sum_j w_j C_j$ objective functions ($P2 \,||\, \sum_j C_j$ or $P2 \,||\, \sum_j w_j C_j$ in Graham's notation—Graham *et al.* 1979) was proved to be NP-complete (Yang 2005). Also, the dedicated case with two machines ($PD2 \,||\, \sum_j C_j$) is strongly NP-hard (Roemer 2006). Instead, the dedicated case with m machines, sequence-dependent setups, and makespan minimization as objective function ($PDm|ST_{sd}|c_{\max}$) is proved to be NP-complete (de Athayde Prata, Rodrigues, and Framinan 2021). When the general case of m parallel dedicated machines is considered, the problem $PDm \,||\, \sum_j C_j$ is NP-hard for $m \geq 2$ (Roemer 2006), while it is strongly NP-hard for $m \geq 3$ (Leung, H. Li, and Pinedo 2005). Also, the problem $PDm \,||\, \sum_j w_j U_j$ (*i.e.* the dedicated case with the weighted sum of tardy orders as objective function) is strongly NP-hard (Ng, T. C. E. Cheng, and Yuan 2003).

Lastly, there always exists a permutation optimal solution for the $P \,||\, \sum_j C_j$ problem (Blocher and Chhajed 1996), and the same holds for the dedicated $PD \,||\, \sum_j C_j$ problem (Leung, H. Li, and Pinedo 2005) and for the unrelated $R \,||\, \sum_j C_j$ case (X. Xu *et al.* 2013). Thus, in such problems, there always exists an optimal solution in which all machines have the same order sequence.

3. Problem description and formulations

3.1. Problem description

The considered system is composed of M machines, N customer orders and M job types. Each customer order i is composed of some of the jobs. Each machine m can process a single job type (*i.e.* they are dedicated machines), hence, jobs are indexed with m as machines. Each job m of order i has a processing time on the dedicated machine m defined as $p_{i,m} \in \mathbb{N}$, *i.e.* order i contains a number of units of the type m such that it requires $p_{i,m}$ time units on machine m. The objective function is the minimization of the total order completion time, *i.e.* $\sum_i C_i(\pi)$, where π is a generic permutation of the order set.

The permutation problem is considered, as Leung, H. Li, and Pinedo (2005) proved that there always exists a permutation optimal solution for the problem. The solution of the addressed problem is then a permutation of N orders, as it indicates the unique sequence in which the orders are processed on each machine.

In the following, several MILP formulations as well as a CP formulation are presented.

3.2. MILP formulations

In the following, several mixed integer programming formulations are proposed.

The first formulation is based on positional variables. In this formulation, index $m = 1, \ldots, M$ refers to machines, index $i = 1, \ldots, N$ refers to the orders, and $k = 1, \ldots, N$ refers to the order position in the schedule.

The decision variables are

- $x_{i,k} \in \{0, 1\}$: a binary variable equal to one if order i is scheduled in position k, zero otherwise;
- $c_k \in \mathbb{R}^+$: the completion time of the order at position k in the sequence.

The positional MILP, namely `Pos`, is as follows:

$$\min \quad \sum_{k=1}^{N} c_k \tag{1}$$

$$\text{s.t.} \quad \sum_{i=1}^{N} x_{i,k} = 1 \quad \forall\, k = 1, \ldots, N \tag{2}$$

$$\sum_{k=1}^{N} x_{i,k} = 1 \quad \forall\, i = 1, \ldots, N \tag{3}$$

$$c_k \geq \sum_{l=1}^{k} \sum_{i=1}^{N} p_{i,m} x_{i,l} \quad \forall\, k = 1, \ldots, N \;\; m = 1, \ldots, M \tag{4}$$

$$x_{i,k} \in \{0, 1\} \quad \forall\, i = 1, \ldots, N; \;\; k = 1, \ldots, N \tag{5}$$

$$c_k \in \mathbb{R}^+ \quad \forall\, k = 1, \ldots, N. \tag{6}$$

Constraints (2) and (3) state that each position is assigned to a single order and each order is allocated to a single position, respectively. Constraints (4) define the completion time the customer orders in each position, *i.e.* the completion time of each customer order. Finally, (5) and (6) define the domain of the decision variables.

The second proposed mathematical formulation is based on precedence variables. In this case, indices i, j and l are used to refer to orders while m refers to machines as in the previous model ($i, j, l = 1, \ldots, N$ and $m = 1, \ldots, M$).

The decision variables are:

- $x_{i,j} \in \{0, 1\}$: a binary variable equal to one if order i precedes order j in the schedule, zero otherwise;
- $c_j \in \mathbb{R}^+$: the completion time of the order j.

The precedence formulation, namely `Prec`, is as follows:

$$\min \quad \sum_{j=1}^{N} c_j \tag{7}$$

$$\text{s.t.} \quad x_{i,j} + x_{j,i} = 1 \quad \forall\, i, j = 1, \ldots, N; \;\; i \neq j \tag{8}$$

$$c_j \geq \sum_{i=1_{i \neq j}}^{N} p_{i,m} x_{i,j} + p_{j,m} \quad \forall\, j = 1, \ldots, N; \;\; m = 1, \ldots, M \tag{9}$$

$$x_{i,j} + x_{j,l} + x_{l,i} \geq 1 \quad \forall\, i, j, l = 1, \ldots, N; \;\; i \neq j \neq l \tag{10}$$

$$x_{i,j} \in \{0, 1\} \quad \forall\, i = 1, \ldots, N; \;\; j = 1, \ldots, N \tag{11}$$

$$c_j \in \mathbb{R}^+ \quad \forall\, j = 1, \ldots, N. \tag{12}$$

Constraints (8) state that either order i precedes order j or order j precedes order i. Equations (9) define the completion time of each order j while constraints (10) are used to avoid cycles. Finally, Equations (11) and (12) define the domain of the decision variables.

The last MILP is a time indexed formulation in which variables referring to the time of completion of each order are used. This kind of variable usually increases the problem symmetry (which might have a negative impact on the computation time).

For this formulation, it is necessary to introduce index $t = 1, \ldots, T$, referring to time instants, while $j = 1, \ldots, N$ and $m = 1, \ldots, M$ refer to customer orders and machines as in the precedence model. Parameter T is the upper bound on the makespan computed as $T = \max_{m=1,\ldots,M} \sum_{j=1}^{N} p_{j,m}$. It is, in practice, the sum of the processing times of all the jobs on the worst (*i.e.* the slowest) machine.

The decision variables are:

- $x_{j,t} \in \{0, 1\}$: a binary variable equal to one if order j completes at time t, zero otherwise;
- $y_{m,t} \in \mathbb{Z}^+$: the load allocated to machine m from time zero to time t (*i.e.* the sum of processing times of all jobs allocated to machine m from time zero to time t);
- $c_j \in \mathbb{R}^+$: the completion time of order j.

The time indexed formulation, namely *TI*, is as follows:

$$\min \quad \sum_{j=1}^{N} c_j \tag{13}$$

$$\text{s.t.} \quad \sum_{t=0}^{T} x_{j,t} = 1 \quad \forall\, j = 1, \ldots, N \tag{14}$$

$$y_{m,t} = y_{m,t-1} + \sum_{j=1}^{N} p_{j,m} x_{j,t} \quad \forall\, m = 1, \ldots, M; \;\; t = 1, \ldots, T \tag{15}$$

$$y_{m,t} \leq t \quad \forall\, m = 1, \ldots, M; \;\; t = 0, \ldots, T \tag{16}$$

$$c_j \geq \sum_{t=0}^{T} t x_{j,t} \quad \forall\, j = 1, \ldots, N; \;\; t = 0, \ldots, T \tag{17}$$

$$x_{j,t} \in \{0, 1\} \quad \forall\, j = 1, \ldots, N; \;\; t = 0, \ldots, T \tag{18}$$

$$y_{m,t} \in \mathbb{Z}^+ \quad \forall\, m = 1, \ldots, M; \;\; t = 0, \ldots, T \tag{19}$$

$$c_j \in \mathbb{R}^+ \quad \forall\, j = 1, \ldots, N. \tag{20}$$

Constraints (14) state that each order has to finish in a single time instant. Constraints (15) and (16), instead, compute the machine load until a given time instant, for each machine and each time, and bound the machine load until a time t not exceeding the total capacity, which is exactly t, respectively. Equations (17) define the completion time of order j and Equations (18)–(20) define the domain of the decision variables.

3.3. Constraint programming formulation

In the following, a constraint programming formulation, namely `CP`, for the presented problem is depicted. Given the current development of CP solvers, problem variables specifically tailored for scheduling problems are used, namely interval variables and sequence variables as well as dedicated global constraint structures.

For this formulation, index $i = 1, \ldots, N$ refers to customer orders, while index $m = 1, \ldots, M$ refers to machines. The decision variables are:

(1) J_{im} for $i \in \{1, \ldots, N\}$, $m \in \{1, \ldots, M\}$: interval variables that define the interval of the processing of the job related to customer order i on machine m, with start and end times;
(2) $S_m := \texttt{sequence}(J_{1m}, \ldots, J_{Nm})$ for $m \in \{1, \ldots, M\}$: sequence variables that defines the processing order of the interval variables J_{im} on machine m;

$$\text{minimize} \sum_{i=1}^{N} \left(\max_{m} C_{im} \right) \tag{21}$$

$$\texttt{NoOverlap}(S_m) \quad m = 1, \ldots, M \tag{22}$$

$$\texttt{SameSequence}(S_1, S_m) \quad m = 2, \ldots, M \tag{23}$$

$$C_{im} = \texttt{EndOf}(J_{im}) \quad i = 1, \ldots, N \;\; m = 1, \ldots, M. \tag{24}$$

Constraints (22) are dedicated to impose the non-overlapping of two intervals on the same machine, ensuring that each machine processes at most one job (related to an order) at a time, while constraints (23) assure the permutation feature of the problem, *i.e.* the same sequence must be forced on each machine. Finally constraints (24) set the completion time of jobs on each machine that is used in (21) to calculate the completion time of each order.

4. Computational results

4.1. Test bed description

The computational tests have been performed on a 2.5 GHz Ryzen™ 5 PC with 32 GB of RAM, and CPLEX® 22.1 and a CP Optimizer have been used as solvers. The proposed models have been tested on three instance sets available in the literature, and already used by Shi *et al.* (2021) (where more details on instance generation can be found). The three sets, namely *Test-1-Test-2-Test-3*, are composed of 360 instances each. In each set, the number of orders varies in $N \in \{50, 100, 200\}$ and the number of machines in $M \in \{2, 5, 10, 20\}$. For each combination of N and M, 30 instances are generated. The three sets differ for the distribution used to generate the number of orders and the order processing times. Specifically,

- *Test-1*: each order has M jobs, each job has to be processed on a different machine, and job processing times are generated from a Uniform distribution $U(1, 99)$;
- *Test-2*: each order has a number of jobs l generated from Uniform distribution $U(1, M)$, each job has to be processed on a different machine, and job processing times are generated from a Uniform distribution $U(1, 99)$;
- *Test-3*: each order consists of M jobs, each job has to be processed on a different machine, and job processing times are generated from a Uniform distribution $U(1, l_{\max})$, where $l_{\max} \in \{20, 50, 100\}$, and, for each $l_{\max}$ value, 10 instances are generated.

For each instance, the three models (`Prec`, `Pos`, `CP`) have been run with a 300 second time limit. No result is reported for the time-indexed formulation as, from the analysis presented in Subsection 4.2, it appears to be inefficient with respect to the other three formulations.

Lower bounds and upper bounds have been collected for each model instance and are presented in the following subsections.

4.2. Lower bound

To asses the quality of each proposed solution method, a common baseline for all formulations is provided. To this end, the CPU time the solver needs to compute the continuous relaxation of each formulation on different size instances has been considered. The `CP` formulation is excluded from the test, as it does not give any lower bound. As columns `Pos`, `Prec` and `TI` of Table 3 show, the CPU time needed to compute LBs by the pure formulations on large size instances may be very large. Thus, a common lower bound is computed by relaxing the MILP formulation based on precedence constraints through the omission of the constraint set (10) (column `Prec` - (10)). This relaxation yields a simplified problem that can be solved efficiently and serves as a unified baseline lower bound

Table 3. Lower bounds.

Size			**Pos**		**Prec**		**Prec** - (10)		**TI**	
Test	*M*	*N*	DEV%	*Time*	DEV%	*Time*	DEV%	Time	DEV%	*Time*
Test-1	2	50	0.41	0.06	**0.00**	0.05	0.03	0.00	2.92	1.01
	2	100	0.24	0.59	**0.00**	2.23	0.02	0.03	1.51	6.43
	5	50	0.52	0.21	**0.00**	0.44	0.11	0.02	2.76	1.93
	5	100	0.26	4.15	**0.00**	19.70	0.11	0.13	1.42	9.21
	10	50	0.39	0.89	**0.01**	0.59	0.16	0.10	2.50	3.86
	10	100	0.23	30.61	**0.00**	19.39	0.12	0.28	1.33	17.54
	20	50	0.32	3.58	**0.02**	0.66	0.18	0.40	2.35	11.07
	20	100	0.15	171.76	**0.02**	23.37	0.16	1.08	1.21	80.72
Test-2	2	50	1.03	0.05	**0.00**	0.06	0.02	0.00	3.88	0.70
	2	100	0.58	0.49	**0.00**	1.46	0.01	0.03	2.04	4.31
	5	50	2.02	0.13	**0.00**	0.18	0.08	0.01	4.65	1.07
	5	100	1.15	2.28	**0.00**	15.04	0.07	0.12	2.59	4.58
	10	50	2.23	0.38	**0.00**	0.50	0.18	0.04	4.48	2.01
	10	100	1.22	11.39	**0.00**	19.85	0.15	0.29	2.53	8.06
	20	50	2.02	1.10	**0.00**	0.61	0.23	0.12	4.07	5.04
	20	100	1.17	51.07	**0.00**	21.39	0.20	0.92	2.37	16.69
Test-3	2	50	0.44	0.06	**0.00**	0.06	0.03	0.00	2.85	0.55
	2	100	0.24	0.59	**0.00**	1.89	0.02	0.03	1.47	3.06
	5	50	0.53	0.22	**0.00**	0.43	0.10	0.02	2.69	1.00
	5	100	0.26	4.37	**0.00**	19.24	0.10	0.13	1.39	4.76
	10	50	0.40	0.93	**0.01**	0.59	0.17	0.10	2.44	2.15
	10	100	0.22	30.97	**0.00**	19.67	0.13	0.29	1.28	9.38
	20	50	0.30	4.24	**0.03**	0.63	0.18	0.43	2.27	5.65
	20	100	0.13	169.58	**0.03**	23.50	0.18	1.06	1.16	45.11

$$\text{DEV\%} = \frac{\text{BEST} - \text{LB}}{\text{BEST}}\%$$

for benchmarking all formulations. Table 3 reports, for each testbed and for each relaxation, the average percentage deviation (DEV%, whose formula is displayed at the bottom of Table 3) with respect to the best lower bound, together with the average CPU time (in seconds) required for its computation. The results indicate that the relaxation of `Prec` omitting constraints (10) (column `Prec` - (10)) is significantly faster than the other three, while incurring only a negligible loss in solution quality. Instead, the `Prec` relaxation provides the tightest bound on average (the best DEV% value of each row is shown in boldface); however, in 18 instances the `Pos` relaxation yields a strictly higher value, confirming that neither dominates the other uniformly. The best lower bound is therefore taken as the maximum of all available relaxations. As the `Prec` and `Pos` both outperform the time-indexed formulation both in terms of solution quality and computation time, such formulation is excluded from the experiments of the remainder of the section.

4.3. Numerical tests

The three proposed models (`Prec`, `Pos`, `CP`) have been tested on the three testbeds available in the literature (*Test-1*, *Test-2*, *Test-3*), and the results are shown in Tables 4, 5 and 6. Each table shows the average Relative Percentage Difference (RPD) values for each testbed, and the percentage of instances in which each model finds the best solution (*% Winning* in the tables). Specifically, for each combination $M{-}N$ and for each model, the tables show the average RPDs and the percentage of winnings calculated on the 30 available instances. The RPD of the ith instance for the α model (where α is `Prec`, `Pos`, `CP`) is calculated as follows:

$$RPD_{\alpha,i} = 100 \times \frac{UB_{\alpha,i} - LB_i}{LB_i}, \tag{25}$$

where $UB_{\alpha,i}$ is the upper bound obtained by model α for instance i within the set time limit and LB_i is the lower bound computed as in Subsection 4.2. The computation of the RPD values on the basis

Table 4. Results for the *Test-1* distribution.

		Average RPD			% Winning		
M	*N*	**Prec** (%)	**Pos** (%)	**CP** (%)	**Prec** (%)	**Pos** (%)	**CP** (%)
2	50	0.30	**0.29**	0.36	57	**63**	13
5	50	1.15	**1.05**	1.43	7	**93**	–
10	50	2.46	**1.89**	2.38	27	**70**	3
20	50	9.29	**2.93**	2.98	10	40	50
2	100	0.22	**0.18**	0.26	–	**100**	–
5	100	17.08	**0.94**	1.29	3	**97**	–
10	100	15.33	3.89	**2.18**	–	**70**	30
20	100	13.43	13.43	**2.58**	–	–	**100**
2	200	26.00	**0.11**	0.64	–	**100**	–
5	200	18.54	18.54	**1.93**	–	–	**100**
10	200	15.00	15.00	**2.70**	–	–	**100**
20	200	12.72	12.72	**2.89**	–	–	**100**

Table 5. Results for the *Test-2* distribution.

		Average RPD			% Winning		
M	*N*	**Prec** (%)	**Pos** (%)	**CP** (%)	**Prec** (%)	**Pos** (%)	**CP** (%)
50	2	**0.29**	**0.29**	0.31	**83**	80	30
50	5	1.32	**1.27**	10.94	20	**77**	3
50	10	2.68	**2.46**	8.76	13	**87**	–
50	20	3.73	**3.41**	8.05	23	**77**	–
100	2	0.17	**0.15**	0.17	7	**87**	13
100	5	23.77	**1.36**	4.73	23	**77**	–
100	10	36.68	**2.80**	16.08	–	**97**	3
100	20	34.27	**7.29**	9.64	–	**67**	33
200	2	35.96	**0.10**	0.30	–	**100**	–
200	5	41.37	**2.67**	4.61	–	**87**	13
200	10	38.20	38.20	**13.73**	13	13	**100**
200	20	35.31	35.31	**23.61**	33	33	**100**

Table 6. Results for the *Test-3* distribution.

		Average RPD			% Winning		
M	*N*	**Prec** (%)	**Pos** (%)	**CP** (%)	**Prec** (%)	**Pos** (%)	**CP** (%)
50	2	**0.30**	**0.30**	0.36	57	**80**	10
50	5	1.08	**0.99**	1.32	17	**83**	–
50	10	2.08	**1.83**	2.25	17	**83**	–
50	20	10.98	2.92	**2.90**	3	47	**53**
100	2	1.14	**0.17**	0.27	–	**100**	–
100	5	17.40	**0.90**	1.20	3	**97**	–
100	10	15.02	**1.62**	2.14	–	**100**	–
100	20	13.19	13.19	**2.65**	–	–	**100**
200	2	24.37	**0.10**	0.69	–	**100**	–
200	5	17.77	16.74	**1.80**	–	–	**100**
200	10	14.93	14.93	**2.72**	–	–	**100**
200	20	12.23	12.23	**2.78**	–	–	**100**

of the lower bounds allows for a clearer estimation of the absolute deviation from the best possible solution value, represented by the lower bound of the optimal solution. Also, Figure 1 shows the 95% confidence intervals of the RPD values, for the three testbeds grouped together. To provide each model with a feasible initial solution and potentially accelerate the search process, the initial solution is simply set to the best permutation schedule of jobs in *Shortest Processing Time* (SPT) order on each machine.

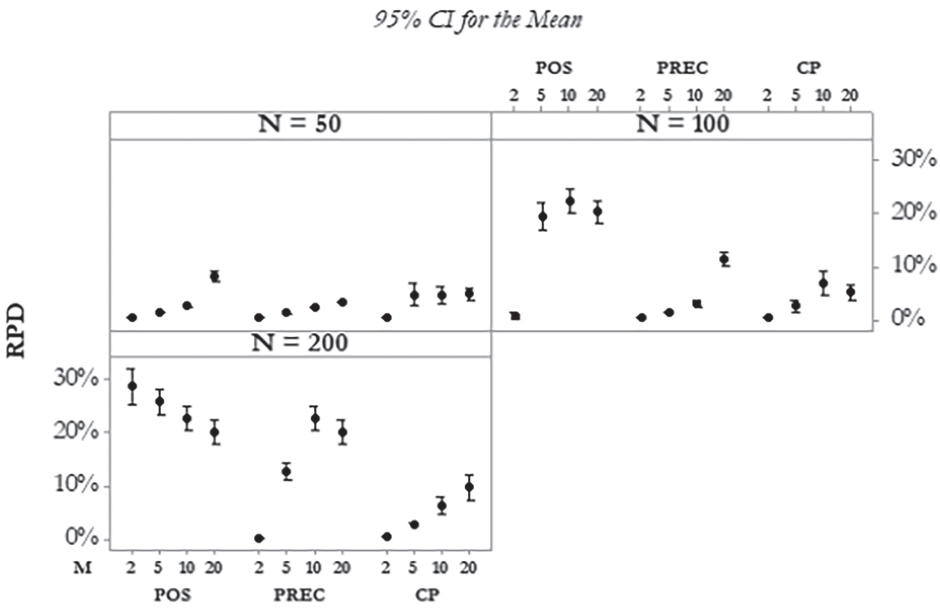

Figure 1. 95% Confidence intervals of RPD values, grouped by *N* (different panels) and *M* (*x*-axis values).

The results reported in the three tables indicate that the models exhibit distinct performance patterns depending on the problem size. For each row of the tables, the algorithm with the lowest average RPD is shown in boldface. For smaller instances (*i.e.* low number of orders N and machines M), the positional model (`Pos`) outperforms the other alternatives. Its low RPD values suggest that the model is not far from finding the optimal solution. In contrast, the `CP` model rarely finds the best solutions, whereas the `Prec` results are generally comparable to those of the `Pos` model. As the problem size increases (*i.e.* with larger values of N and M), the `CP` model consistently outperforms both `Pos` and `Prec`. While the RPD values of `Pos` and `Prec` grow substantially with the instance size, those of the `CP` model increase only marginally. This is particularly evident for *Test-1* and *Test-3* (Tables 4 and 6), where `CP` maintains RPD values below 3%. The performance observed on *Test-2*, instead, has greater variability. In this case, the `Pos` model often performs better than `CP`, with the exception of the largest instances (the last two rows of Table 5), where `CP` once again provides the most competitive results. This behaviour might be related to the specific characteristic of *Test-2* with respect to the other two instance sets, *i.e.* the fact that, in *Test-2*, the number of jobs in each order is not fixed to M but can vary.

All in all, in general the `Pos` and `CP` models are comparable in terms of performance when the instance size is small; when the number of machines and jobs increases, the `CP` model achieves better results. Figure 1 graphically shows such findings: the three panels show the RPD 95% confidence interval for the three models, grouped by M values. The confidence intervals in the figure prove that the difference between the `CP` model and the other two models is statistically significant for large N values ($N = 200$), while there is no significant difference among the proposed models with small N values ($N = 50$). Moreover, Figure 2 shows the performance profile when a time limit of 3600 seconds is used for `Pos` and `CP` models for the three testbeds.

The performance profiles have been built by tracking the behaviour of the two approaches, applying a 1% threshold relative to the value of the best solution obtained by one of the two methods. The threshold has been used since neither method certifies optimality, and then the best solution obtained is itself subject to numerical uncertainty between solvers. In other words, a 0% threshold would measure the favourable case of exact parity, not relative quality. The need for a threshold has

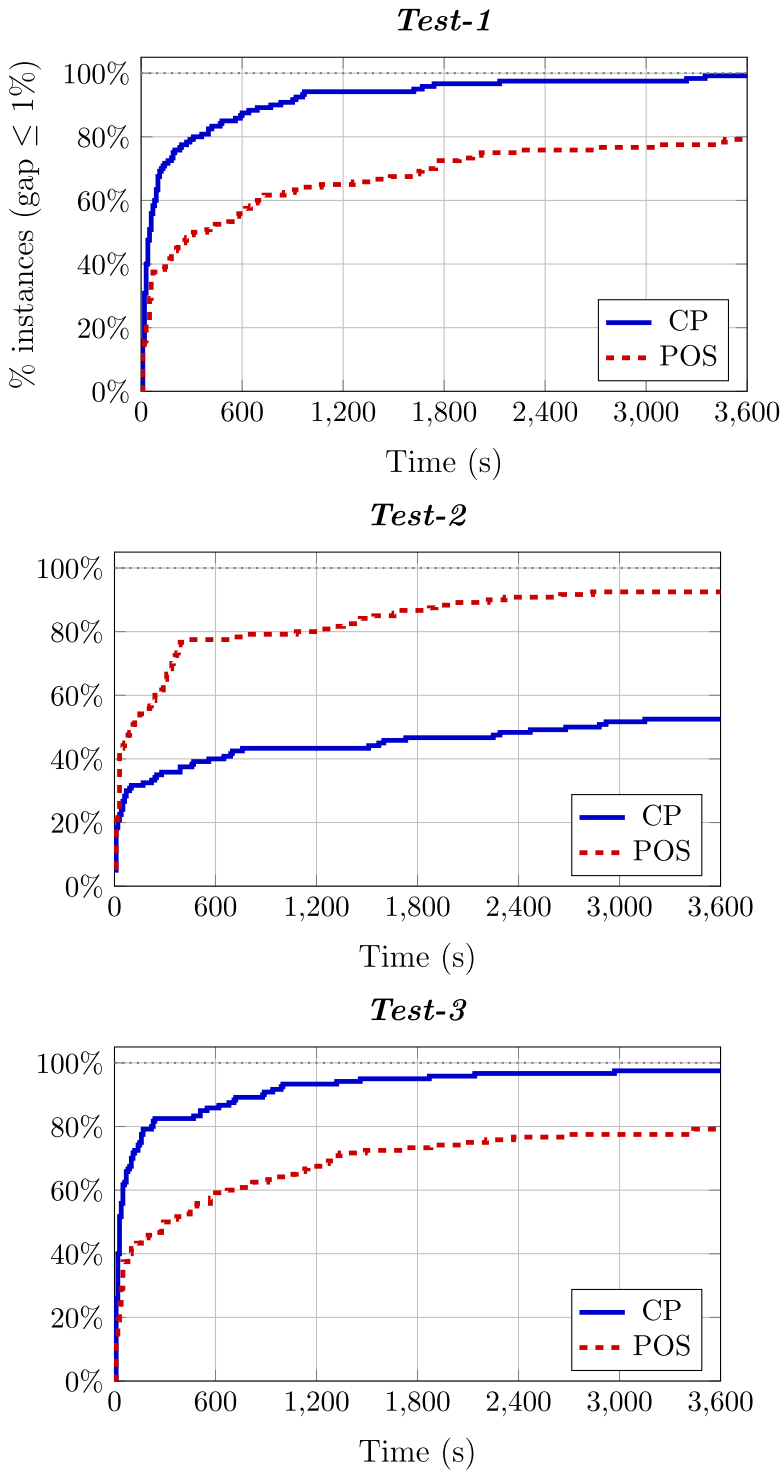

Figure 2. Performance profiles: **CP** versus **Pos**, gap $\leq$ 1% with respect to the best solution known.

been demonstrated by Dolan and Moré (2002) and Moré and Wild (2009), who also suggested the value of 1%, while guidelines for a correct interpretation of the performance profile has been reported by Gould and Scott (2016). The performance profiles show different behaviours of the two approaches depending on the class of instance considered. Specifically, the `CP` model appears to achieve faster convergence in *Test-1* and *Test-3*, while in *Test-2* it is the `Pos` model that achieves faster convergence, and this can be due to the particular structure of the instances.

4.4. Comparison with the state-of-the-art algorithm

As the same testbeds have been used by Shi *et al.* (2021), the results of the proposed models (`Prec`, `Pos`, `CP`) have also been compared with the results of their best algorithm, *i.e.* LDR-II-AS, which is a two-stage heuristic optimization method. In the first stage, it constructs an initial schedule using a dispatching rule generated via feature-enhanced genetic programming; in the second stage, it applies an adaptive local search to intensify and diversify around that initial solution. When comparing the LDR-II-AS algorithm with the methods proposed in this article, it is important to emphasize that the former is heuristic, whereas the latter consists exclusively of exact methods.

The comparison with LDR-II-AS is presented in Table 7. The table reports: (i) the results of `Prec`, `Pos`, and `CP` obtained under a time limit of 300 seconds for all instances described in the previous sections; and (ii) an additional set of results for the two best-performing proposed methods, `Pos` and `CP`, obtained under a time limit of 3600 seconds, for the subset of instances characterized by $M > 50$. The latter configuration ensures a more equitable comparison, as the results for LDR-II-AS reported in Shi *et al.* (2021) are based on a 3600-second time limit. It should be noted, however, that the comparison involves an heuristic method (LDR-II-AS) and exact methods (the three proposed approaches), a distinction that may influence the reported performance. Nonetheless, on a 3600-second time limit base, on the vast majority of instances either the `CP` model or the `Pos` model achieve comparable performances with respect to LDR-II-AS except for the set of largest instances. This assesses the usability of general exact approaches for solving small and medium-to-large problems.

Table 7 reports the comparison results in terms of Percentage Gap (PG), and for each row the best PG value is shown in boldface. The PG for algorithm α (`Prec`, `Pos`, `CP`) of instance i is computed as follows:

$$PG_{\alpha,i} = 100 \times \frac{UB_{\alpha,i} - UB_{\text{LDR-II-AS},i}}{UB_{\text{LDR-II-AS},i}}. \tag{26}$$

For each row, corresponding to a given M–N pair, the average PG across all instances of the three testbeds is presented. For all instances, PG values are always greater than or equal to zero, thus LDR-II-AS always achieves the best solution. However, the proposed `CP` model, run with a time limit

Table 7. Comparison with Shi *et al.* (2021): percentage gap from LDR-II-AS solutions.

		$T = 300$ s			$T = 3600$ s	
M	N	**Prec** (%)	**Pos** (%)	**CP** (%)	**Pos** (%)	**CP** (%)
50	2	**0.00**	**0.00**	0.05		
50	5	0.13	**0.06**	3.47		
50	10	0.56	**0.22**	2.56		
50	20	5.43	**0.63**	2.14		
100	2	0.35	**0.01**	0.08	0.00	0.04
100	5	18.64	**0.41**	1.74	0.14	0.98
100	10	20.94	**1.61**	5.58	0.52	4.75
100	20	18.30	9.49	**3.23**	0.88	2.21
200	2	28.67	**0.03**	0.46	0.01	0.15
200	5	25.38	12.20	**2.36**	0.23	1.55
200	10	21.83	21.83	**5.63**	6.73	5.12
200	20	18.75	18.75	**8.54**	18.80	7.08

one order of magnitude lower than that of LDR-II-AS, has a maximum average gap of only 8.54% (observed for the largest instances). When the time limit is increased to 3600 seconds, the gap further decreases, with the maximum average PG reduced to approximately 7.08% and negligible differences observed for small- and medium-sized instances. Considering both the significantly higher complexity of LDR-II-AS and the relatively modest gap between the two approaches, the results, particularly those obtained under the extended time limit, might suggest that the `CP` formulation can offer the best trade-off between solution quality and implementation complexity. Moreover, since LDR-II-AS is an heuristic method, it cannot certify the optimality of a solution even when the optimum is reached. In contrast, `CP`, being an exact approach, is also capable of proving optimality and terminating once the optimal solution is found. In order to assess `CP` solutions quality, combinatorial lower bounds can be applied, such as those proposed by Mastrolilli *et al.* (2010) and Roemer (2006) for total completion time and X. Xu *et al.* (2013) and H. Li *et al.* (2021) for unrelated parallel machines.

5. Conclusions

The Customer Order Scheduling (COS) problem addresses the allocation of customer orders, each consisting of multiple jobs, to machines in a way that optimizes order-level performance criteria rather than individual job measures.

This article studied the permutation COS problem with dedicated machines, focusing on the minimization of the total order completion time. Three mathematical programming formulations were proposed and compared on benchmark instances: a MILP model with positional variables, a MILP model with precedence variables, and a Constraint Programming (CP) formulation.

The computational experiments showed that the CP formulation provides the most effective and robust performance across the considered models. Furthermore, the CP model was compared with the state-of-the-art approach, a feature-enhanced genetic programming algorithm. Despite being executed under a computational time limit twelve times shorter than that of the genetic programming method, the CP formulation achieved solutions with an average deviation of less than 9%.

These findings highlight that, although the state-of-the-art algorithm incorporates sophisticated design and significantly higher complexity, a relatively simple CP model can deliver competitive results. Overall, the CP formulation represents a compelling trade-off between solution quality and implementation complexity, and it offers a promising direction for practical applications.

Author contributions

CRediT: **Arianna Alfieri:** Conceptualization, Data curation, Writing – original draft, Writing – review & editing; **Andrea Grosso:** Conceptualization, Data curation, Software, Writing – original draft, Writing – review & editing; **Erica Pastore:** Conceptualization, Data curation, Writing – original draft, Writing – review & editing; **Fabio Salassa:** Conceptualization, Data curation, Methodology, Writing – original draft, Writing – review & editing

Disclosure statement

No potential conflict of interests was reported by the authors.

ORCID

Arianna Alfieri https://orcid.org/0000-0002-2700-9430
Andrea Grosso https://orcid.org/0000-0002-9926-2443
Erica Pastore http://orcid.org/0000-0003-0085-2978
Fabio Salassa http://orcid.org/0000-0002-9698-6892

Data availability statement

The data that support the findings of this study are available from the corresponding author, Erica Pastore, upon reasonable request.

References

Blocher, J. D., and D. Chhajed. 1996. "The Customer Order Lead-Time Problem on Parallel Machines." *Naval Research Logistics* 43 (5): 629–654. https://doi.org/10.1002/(ISSN)1520-6750.

Borisovsky, P. A., A. O. Zakharov, and Y. V. Zakharova. 2025. "Evolutionary Algorithms for Customer Order Scheduling." *The Bulletin of Irkutsk State University. Series Mathematics* 53:3–17. https://doi.org/10.26516/1997-7670.2025.53.3.

Bukchin, Y., E. Hanany, and Y. Gerchak. 2022. "Coordination of the Decentralized Concurrent Open-Shop." *IISE Transactions* 54 (12): 1172–1185. https://doi.org/10.1080/24725854.2022.2040760.

Çetinkaya, F. C., P. Yeloğlu, and H. Çatmakaş. 2021. "Customer Order Scheduling with Job-Based Processing on a Single-Machine to Minimize the Total Completion Time." *International Journal of Industrial Engineering Computations* 12 (3): 273–292.

Cheng, T. C. E., Q. Nong, and C. T. Ng. 2011. "Polynomial-Time Approximation Scheme for Concurrent Open Shop Scheduling with a Fixed Number of Machines to Minimize the Total Weighted Completion Time." *Naval Research Logistics (NRL)* 58 (8): 763–770. https://doi.org/10.1002/nav.20484.

Cheng, T. C. E., and C. Sin. 1990. "A State-of-the-Art Review of Parallel-Machine Scheduling Research." *European Journal of Operational Research* 47 (3): 271–292. https://doi.org/10.1016/0377-2217(90)90215-W.

de Athayde Prata, B., C. D. Rodrigues, and J. M. Framinan. 2021. "Customer Order Scheduling Problem to Minimize Makespan with Sequence-Dependent Setup times." *Computers & Industrial Engineering* 151:106962. https://doi.org/10.1016/j.cie.2020.106962.

de Athayde Prata, B., C. D. Rodrigues, and J. M. Framinan. 2022. "A Differential Evolution Algorithm for the Customer Order Scheduling Problem with Sequence-Dependent Setup times." *Expert Systems with Applications* 189:116097. https://doi.org/10.1016/j.eswa.2021.116097.

Dolan, E. D., and J. J. Moré. 2002. "Benchmarking Optimization Software with Performance Profiles." *Mathematical Programming* 91 (2): 201–213. https://doi.org/10.1007/s101070100263.

Erel, E., and J. B. Ghosh. 2007. "Customer Order Scheduling on a Single Machine with Family Setup times: Complexity and Algorithms." *Applied Mathematics and Computation* 185 (1): 11–18. https://doi.org/10.1016/j.amc.2006.06.086.

Framinan, J. M., J. N. D. Gupta, and R. Leisten. 2004. "A Review and Classification of Heuristics for Permutation Flow-Shop Scheduling with Makespan Objective." *Journal of the Operational Research Society* 55 (12): 1243–1255. https://doi.org/10.1057/palgrave.jors.2601784.

Framinan, J. M., and P. Perez-Gonzalez. 2017. "New Approximate Algorithms for the Customer Order Scheduling Problem with Total Completion Time Objective." *Computers & Operations Research* 78:181–192. https://doi.org/10.1016/j.cor.2016.09.010.

Framinan, J. M., and P. Perez-Gonzalez. 2018. "Order Scheduling with Tardiness Objective: Improved Approximate Solutions." *European Journal of Operational Research* 266 (3): 840–850. https://doi.org/10.1016/j.ejor.2017.10.064.

Gould, N., and J. Scott. 2016. "A Note on Performance Profiles for Benchmarking Software." *ACM Transactions on Mathematical Software* 43 (2): 1–5. https://doi.org/10.1145/2950048.

Graham, R. L., E. L. Lawler, J. K. Lenstra, and A. R. Kan. 1979 "Optimization and Approximation in Deterministic Sequencing and Scheduling: A Survey." *Annals of Discrete Mathematics* 5 (3): 287–326. https://doi.org/10.1016/S0167-5060(08)70356-X.

Gupta, J. N. D., C.-C. Wu, W.-C. Lin, X.-G. Zhang, D. Bai, B. M. Lin, and C.-C. Liao. 2023. "Bicriteria Single-Machine Scheduling with Multiple Job Classes and Customer Orders." *Applied Soft Computing* 147:110809. https://doi.org/10.1016/j.asoc.2023.110809.

Gupta, S. K., and J. Kyparisis. 1987. "Single Machine Scheduling Research." *Omega* 15 (3): 207–227. https://doi.org/10.1016/0305-0483(87)90071-5.

Hazır, Ö., Y. Günalay, and E. Erel. 2008. "Customer Order Scheduling Problem: A Comparative Metaheuristics Study." *The International Journal of Advanced Manufacturing Technology* 37 (5–6): 589–598.

Hejazi, S. R., and S. Saghafian. 2005. "Flowshop-Scheduling Problems with Makespan Criterion: A Review." *International Journal of Production Research* 43 (14): 2895–2929. https://doi.org/10.1080/0020754050056417.

Hoffmann, J., J. S. Neufeld, and U. Buscher. 2022. "Iterated Greedy Algorithms for Customer Order Scheduling with Dedicated Machines." *IFAC-PapersOnLine* 55 (10): 1594–1599. https://doi.org/10.1016/j.ifacol.2022.09.618.

Hoffmann, J., J. S. Neufeld, and U. Buscher. 2024. "Minimizing the Earliness–Tardiness for the Customer Order Scheduling Problem in a Dedicated Machine Environment." *Journal of Scheduling* 27 (6): 525–543. https://doi.org/10.1007/s10951-024-00814-z.

Hou, B., H. Zheng, W. Liu, W. Wu, D.-Z. Du, and S. Gao. 2023. "Improved Approximation Algorithm for the Parallel-Machine Customer Order Scheduling with Delivery Time and Submodular Rejection Penalties." *Optimization Letters* 18: 1747–1754.

Janiak, A., and T. Krysiak. 2012. "Scheduling Jobs with Values Dependent on Their Completion Times." *International Journal of Production Economics* 135 (1): 231–241. https://doi.org/10.1016/j.ijpe.2011.07.014.

Komaki, G. M., S. Sheikh, and B. Malakooti. 2019. "Flow Shop Scheduling Problems with Assembly Operations: A Review and New Trends." *International Journal of Production Research* 57 (10): 2926–2955. https://doi.org/10.1080/00207543.2018.1550269.

Koulamas, C. 1994. "The Total Tardiness Problem: Review and Extensions." *Operations Research* 42 (6): 1025–1041. https://doi.org/10.1287/opre.42.6.1025.

Leung, J. Y. T., H. Li, and M. Pinedo. 2005. "Order Scheduling in an Environment with Dedicated Resources in Parallel." *Journal of Scheduling* 8 (5): 355–386. https://doi.org/10.1007/s10951-005-2860-x.

Li, H., Z. Li, Y. Zhao, and X. Xu. 2021. "Scheduling Customer Orders on Unrelated Parallel Machines to Minimise Total Weighted Completion Time." *Journal of the Operational Research Society* 72 (8): 1726–1736. https://doi.org/10.1080/01605682.2020.1718010.

Li, L., W. Lin, D. Bai, X. Zhang, A. Azzouz, S. Cheng, Y. Wu, and C. Wu. 2023. "Composite Heuristics and Water Wave Optimality Algorithms for Tri-Criteria Multiple Job Classes and Customer Order Scheduling on a Single Machine." *International Journal of Industrial Engineering Computations* 14 (2): 265–274. https://doi.org/10.5267/j.ijiec.2023.2.002.

Li, L.-Y., J.-Y. Xu, S.-R. Cheng, X. Zhang, W.-C. Lin, J.-C. Lin, Z.-L. Wu, and C.-C. Wu. 2022. "A Genetic Hyper-Heuristic for an Order Scheduling Problem with Two Scenario-Dependent Parameters in a Parallel-Machine Environment." *Mathematics* 10 (21): 4146–4146+22. https://doi.org/10.3390/math10214146.

Lin, B. M., and A. V. Kononov. 2007. "Customer Order Scheduling to Minimize the Number of Late Jobs." *European Journal of Operational Research* 183 (2): 944–948. https://doi.org/10.1016/j.ejor.2006.10.021.

Lin, W.-C., J. Xu, D. Bai, I.-H. Chung, S.-C. Liu, and C.-C. Wu. 2019. "Artificial Bee Colony Algorithms for the Order Scheduling with Release Dates." *Soft Computing* 23 (18): 8677–8688. https://doi.org/10.1007/s00500-018-3466-5.

Lin, W.-C., X. Zhang, X. Liu, K.-X. Hu, S.-R. Cheng, A. Azzouz, and C.-C. Wu. 2023. "Sequencing Single Machine Multiple-Class Customer Order Jobs Using Heuristics and Improved Simulated Annealing Algorithms." *RAIRO-Operations Research* 57 (3): 1417–1441. https://doi.org/10.1051/ro/2023056.

Liu, G., Y. Xie, and H. Wang. 2025. "Customer Order Scheduling on a Serial-Batch Machine in Precast Bridge Construction." *Computers & Operations Research* 173:106871. https://doi.org/10.1016/j.cor.2024.106871.

Luo, S., H. Yu, Y. Zhao, S. Wang, S. Yu, and L. Li. 2016. "Towards Practical and Near-Optimal Coflow Scheduling for Data Center Networks." *IEEE Transactions on Parallel and Distributed Systems* 27 (11): 3366–3380. https://doi.org/10.1109/TPDS.2016.2525767.

Mason, S. J., and J.-S. Chen. 2010. "Scheduling Multiple Orders per Job in a Single Machine to Minimize Total Completion Time." *European Journal of Operational Research* 207 (1): 70–77. https://doi.org/10.1016/j.ejor.2010.03.034.

Mastrolilli, M., M. Queyranne, A. S. Schulz, O. Svensson, and N. A. Uhan. 2010. "Minimizing the Sum of Weighted Completion Times in a Concurrent Open Shop." *Operations Research Letters* 38 (5): 390–395. https://doi.org/10.1016/j.orl.2010.04.011.

Moré, J. J., and S. M. Wild. 2009. "Benchmarking Derivative-Free Optimization Algorithms." *SIAM Journal on Optimization* 20 (1): 172–191. https://doi.org/10.1137/080724083.

Ng, C. T., T. C. E. Cheng, and J. Yuan. 2003. "Concurrent Open Shop Scheduling to Minimize the Weighted Number of Tardy Jobs." *Journal of Scheduling* 6 (4): 405–412. https://doi.org/10.1023/A:1024284828374.

Pastore, E., A. Alfieri, C. Castiglione, G. Nicosia, and F. Salassa. 2022. "A Matheuristic Approach to the Open Shop Scheduling Problem with Sequence-Dependent Setup Times." *IFAC-PapersOnLine* 55 (10): 2167–2172. https://doi.org/10.1016/j.ifacol.2022.10.029.

Riahi, V., M. H. Newton, M. M. A. Polash, and A. Sattar. 2019. "Tailoring Customer Order Scheduling Search Algorithms." *Computers & Operations Research* 108:155–165. https://doi.org/10.1016/j.cor.2019.04.015.

Roemer, T. A. 2006. "A Note on the Complexity of the Concurrent Open Shop Problem." *Journal of Scheduling* 9 (4): 389–396. https://doi.org/10.1007/s10951-006-7042-y.

Rossit, D. A., F. Tohmé, and M. Frutos. 2018. "The Non-Permutation Flow-Shop Scheduling Problem: A Literature Review." *Omega* 77:143–153. https://doi.org/10.1016/j.omega.2017.05.010.

Rostami, M., O. Kheirandish, and N. Ansari. 2015. "Minimizing Maximum Tardiness and Delivery Costs with Batch Delivery and Job Release Times." *Applied Mathematical Modelling* 39 (16): 4909–4927. https://doi.org/10.1016/j.apm.2015.03.052.

Shi, Z., Z. Huang, and L. Shi. 2018. "Customer Order Scheduling on Batch Processing Machines with Incompatible Job Families." *International Journal of Production Research* 56 (1-2): 795–808. https://doi.org/10.1080/00207543.2017.1401247.

Shi, Z., H. Ma, M. Ren, T. Wu, and J. Y. Andrew. 2021. "A Learning-Based Two-Stage Optimization Method for Customer Order Scheduling." *Computers & Operations Research* 136:105488. https://doi.org/10.1016/j.cor.2021.105488.

Shi, Z., L. Wang, P. Liu, and L. Shi. 2015. "Minimizing Completion Time for Order Scheduling: Formulation and Heuristic Algorithm." *IEEE Transactions on Automation Science and Engineering* 14 (4): 1558–1569. https://doi.org/10.1109/TASE.2015.2456131.

Su, L.-H., P.-S. Chen, and S.-Y. Chen. 2013. "Scheduling on Parallel Machines to Minimise Maximum Lateness for the Customer Order Problem." *International Journal of Systems Science* 44 (5): 926–936. https://doi.org/10.1080/00207721.2011.649366.

Wang, G., and T. C. E. Cheng. 2007. "Customer Order Scheduling to Minimize Total Weighted Completion Time." *Omega* 35 (5): 623–626. https://doi.org/10.1016/j.omega.2005.09.007.

Wu, C.-C., D. Bai, X. Zhang, S.-R. Cheng, J.-C. Lin, Z.-L. Wu, and W.-C. Lin. 2021. "A Robust Customer Order Scheduling Problem Along with Scenario-Dependent Component Processing Times and Due Dates." *Journal of Manufacturing Systems* 58:291–305. https://doi.org/10.1016/j.jmsy.2020.12.013.

Wu, C.-C., X. Zhang, D. Bai, A. Azzouz, W.-H. Wu, X.-R. Chen, and W.-C. Lin. 2024. "Sequencing a Tri-Criteria Multiple Job Classes and Customer Orders Problem on a Single Machine by Using Heuristics and Simulated Annealing Method." *Operational Research* 24 (1): 1–22. https://doi.org/10.1007/s12351-023-00809-2.

Xiong, H., S. Shi, D. Ren, and J. Hu. 2022. "A Survey of Job Shop Scheduling Problem: The Types and Models." *Computers & Operations Research* 142:105731. https://doi.org/10.1016/j.cor.2022.105731.

Xu, X., Y. Ma, Z. Zhou, and Y. Zhao. 2013. "Customer Order Scheduling on Unrelated Parallel Machines to Minimize Total Completion Time." *IEEE Transactions on Automation Science and Engineering* 12 (1): 244–257. https://doi.org/10.1109/TASE.2013.2291899.

Xu, X., Y. Zhao, M. Wu, Z. Zhou, Y. Ma, and Y. Liu. 2025. "Stochastic Customer Order Scheduling to Minimize Long-Run Expected Order Cycle Time." *Annals of Operations Research* 350 (3): 1283–1306. https://doi.org/10.1007/s10479-016-2254-9.

Yang, J. 2005. "The Complexity of Customer Order Scheduling Problems on Parallel Machines." *Computers & Operations Research* 32 (7): 1921–1939. https://doi.org/10.1016/j.cor.2003.12.008.

Zhao, Y., X. Kong, X. Xu, and E. Xu. 2022. "Resource-Controlled Stochastic Customer Order Scheduling via Particle Swarm Optimization with Bound Information." *Industrial Management & Data Systems* 122 (8): 1882–1908. https://doi.org/10.1108/IMDS-02-2022-0105.

Zhao, Y., X. Xu, E. Xu, and B. Niu. 2021. "Stochastic Customer Order Scheduling on Heterogeneous Parallel Machines with Resource Allocation Consideration." *Computers & Industrial Engineering* 160:107539. https://doi.org/10.1016/j.cie.2021.107539.

Zheng, H.-Y., S.-G. Gao, W. Liu, and B. Hou. 2024. "An Approximation Algorithm for the Parallel-Machine Customer Order Scheduling with Delivery Time and Submodular Rejection Penalties." *Journal of the Operations Research Society of China* 12 (2): 495–504. https://doi.org/10.1007/s40305-022-00430-8.

Zipfel, B., J. Neufeld, and U. Buscher. 2024. "An Iterated Local Search for Customer Order Scheduling in Additive Manufacturing." *International Journal of Production Research* 62 (3): 605–625. https://doi.org/10.1080/00207543.2023.2167015.