Computers & Industrial Engineering 175 (2023) 108899

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

An exact decomposition method for unrelated parallel machine scheduling with order acceptance and setup times[☆]

Shijin Wang [a,*], Ruochen Wu [a], Feng Chu [b,c], Jianbo Yu [d]

[a] School of Economics and Management, Tongji University, Shanghai 200092, China
[b] Laboratoire IBISC, Univ-Évry, Université Paris-Saclay, Évry 91025, France
[c] School of Economics and Management, Fuzhou University, Fuzhou 350116, China
[d] School of Mechanical Engineering, Tongji University, Shanghai 710049, China

ARTICLE INFO

Keywords:
Scheduling
Unrelated parallel machine
Order acceptance
Sequence and machine-dependent setup times
Logic-based Benders decomposition

ABSTRACT

In this paper, an unrelated parallel machine scheduling problem is studied, where order acceptance, sequence and machine-dependent setup times and the maximum available times of machines are additionally considered. The objective is to maximize the profit, which is the difference between the total revenues of accepted jobs and the cost associated with makespan. A mixed integer programming (MIP) model is formulated. To tackle this problem efficiently, an exact decomposition method, which is a two-layer logic-based Benders decomposition (LBBD) based (denoted by TL-LBBD) method, is developed, where an inner LBBD is embedded into an outer LBBD. Specifically, in the outer LBBD method, the master problem is used to determine the acceptance of jobs, whereas the subproblem examines the schedule given the accepted jobs from the master problem. The subproblem could be further decomposed into an assignment master problem and a sequencing subproblem by the inner LBBD method as well. Extensive computational experiments are conducted and the results show that the developed TL-LBBD method produce better quality solutions in significantly less computation time than solving the MIP model directly and the classic LBBD method. Moreover, the maximum scales of the problem instances that could be solved to optimality by the developed TL-LBBD method within 30 min are also evaluated.

1. Introduction

Faced with a large number of received orders, the first and prime decision of a make-to-order (MTO) production or service system is whether to accept or reject a received order (Naderi & Roshanaei, 2020; Tarhan & Oğuz, 2021). Such a decision becomes even crucial during the COVID-19 pandemic. The mandatory requirements, such as the restricted number of people working in poorly ventilated rooms, social distancing and travel restrictions, have direct impacts on the manufacturing process and the supply chain of raw materials, leading to the unstable production capacity. Driven by the aim to maximize the profit, a MTO manufacturing or service company tends to select the orders with high revenues while the fluctuating production or service capacity is not exceeded. Another essential decision of the MTO system is to schedule the accepted orders properly according to the customers' demands. For example, if the objective is to produce all products as soon as possible, a scheduling problem with the aim of minimizing the makespan or the completion time of the last job should be dealt with (Pinedo, 2016). Therefore, the order acceptance and scheduling (OAS), which consists of deciding whether a received order is going to be accepted for processing and determining the associated schedule with certain criteria, is needed to achieve great profitability and low risk of penalty. The OAS problem has been studied widely in the literature (Slotnick, 2011; Shabtay et al., 2013; Emami et al., 2015; Wang et al., 2015; Ou et al., 2015; Ou et al., 2016; Emami et al., 2017; Jiang et al., 2017; Ou & Zhong, 2017a, 2017b; Wang, Yin, & Cheng, 2018; Wu et al., 2018; Wang & Ye, 2019; Bruni et al., 2020; Kong et al., 2020; Liu & Lu, 2020; Naderi & Roshanaei, 2020).

One typical example of the OAS problem arises in the semiconductor industry which is worth billions dollars annually and is a highly complex global business (Wang, Yang, & Yu, 2018). In recent years, manufacturers of the industry are stimulated to accept limited orders and to make full use of the their production capacity efficiently because of the global semiconductor shortage and the soaring demands in China from car production and consumer electronics like personal computers,

☆ Acknowledgment: This work was supported by the National Natural Science Foundation of China (NSFC) under Grants 71971155. The work was also supported by the Fundamental Research Funds for the Central Universities, China.

* Corresponding author.
E-mail address: shijinwang@tongji.edu.cn (S. Wang).

https://doi.org/10.1016/j.cie.2022.108899
Received 23 March 2022; Received in revised form 12 November 2022; Accepted 7 December 2022
Available online 13 December 2022

smartphones and home appliances (https://www.chinadaily.com.cn). Aside from the decisions about order acceptance, the manufacturers also faces the decisions about scheduling including job allocation and sequencing. In real-world applications, there are usually several manufacturing machines working in parallel and the rates of processing are not the same since the condition of each machine may differ. Consequently, the job processing of each order on different machines could be different and it takes certain setup time for the specific machine to switch from one job to another because of the compatibility issues. Note that the machines in the semiconductor industry are usually very expensive and should be maintained regularly, indicating these machines have maximum available times. Therefore, it is very essential to efficiently schedule the accepted orders on these machines.

Another representative application exists in the operating room scheduling (Roshanaei et al., 2017). As pointed out by Denton et al. (2010), operating rooms are not only a hospital's greatest revenue source but also the largest cost center. It is very crucial to make full use of operating rooms and arrange patients to undergo their surgeries, especially those dying or severely wounded patients. The surgery time of a patient in an operating room depends on the corresponding medical teams and the health status of the patients. Furthermore, there is additional preparation time and cleaning time between two consecutive surgeries. However, the people in the medical teams may get exhausted and not suitable for more surgeries, which leads to the maximum available time.

In this paper, we study the unrelated parallel machine scheduling problem with order acceptance, sequence and machine-dependent setup times and given maximum available times of machines. In addition, we assume that there is only one job for each order. The OAS incorporates these common characteristics to simulate an environment which is close to the real-world settings. Unrelated parallel machine environment is considered since it has unique processing times for the jobs on each machine and it is a generalization of a wide variety of real-world applications, as shown in the semiconductor manufacturing and operation room scheduling mentioned above. The objective function is to maximize the profit, which is the difference between the total revenues of all accepted jobs and the cost associated with makespan. Such a cost is usually related to the overall system operating cost as the entire system could be stopped only when all accepted jobs are completed. Using the well-known three-field notation (Graham et al., 1979), this problem can be denoted as $R_m|oa, sdst, mat|\sum z_j u_j - C_{max}$, where R_m represents unrelated parallel machines, oa denotes the order (job) acceptance, $sdst$ describes the sequence and machine-dependent setup times, mat denotes the maximum available time of machines and $\sum z_j u_j - C_{max}$ represents the profit-oriented objective function which is to maximize the difference between the total revenues of accepted jobs ($\sum z_j u_j$) and the cost of makespan (C_{max}).

To deal with the problem efficiently, a two-layer logic-based Benders decomposition (LBBD) based (denoted by TL-LBBD) exact decomposition method is developed, where an inner LBBD is embedded into an outer LBBD. The core idea of the TL-LBBD method is to separate the decisions about order acceptance, job assignment and sequencing instead of considering intertwined decisions together. The structure of the developed TL-LBBD exact decomposition method is presented in Fig. 1.

The main contributions of this paper are summarized as follows:

(1) An exact decomposition approach, the TL-LBBD method based on the decision-partitioning structure, is proposed to solve the problem. In the outer LBBD, the decisions about order acceptance are determined by the master problem. Then the derived assignments of accepted jobs from the master problem are utilized as the initial solution to the subproblem, in which the minimum makespan of the accepted jobs is verified. The subproblem could be further decomposed by the inner LBBD due to the natural "first assign and sequence later" separable structure of the subproblem. Valid cuts and bounds for the inner and outer LBBD methods are developed to accelerate the search process.

(2) The efficacy of the proposed TL-LBBD method is demonstrated by conducting extensive computational experiments on two different data sets. The computational results show that our developed TL-LBBD method yields better performance in terms of both computation time and solution quality than solving the MIP model in CPLEX directly and the classic LBBD method which considers decisions about the order acceptance and job assignment jointly in the master problem. The maximum scales of the problem instances that can be solved to optimality by the developed TL-LBBD method within the predefined time limit are also evaluated.

The rest of this paper is organized as follows. The related literature are reviewed in Section 2. The mathematical formulation for the problem is presented in Section 3. The details of the developed TL-LBBD method for the problem is described in Section 4. Computational experiments are conducted in Section 5. Finally, conclusions and future research directions are provided in Section 6.

2. Literature review

In this section, we review the relevant papers in the research streamline of parallel machine scheduling in three highly-related topics: scheduling with order acceptance, scheduling with sequence-dependent setup times, and scheduling with limited resources.

2.1. Parallel machine scheduling with order acceptance

The OAS problem and its variants have received considerable attention in the past two decades and there is a plethora of papers on the OAS problems in different make-to-order systems. Interested readers can refer to the early comprehensive survey by Slotnick (2011) and the references therein for further details. The scheduling problem with rejection, which considers the total costs of rejecting a subset of orders rather than the revenues earned by accepting a subset of orders, is equivalent to the OAS problem. Readers can refer to the excellent survey in Shabtay et al. (2013) of the scheduling problem with rejection. Here we focus on the recent studies under the parallel machine environment. The recent studies with their characteristics, including the types of the problem (i.e., deterministic, robust or distributionally robust), the acceptance or rejection version, the deterministic objectives, parallel machine environments, setup times, release date and due date, limited resources and the methods, are summarized in Table 1.

As shown in Table 1, Liu and Lu (2020), Ou et al. (2016, 2015) and Wang, Yin, and Cheng (2018) studied the deterministic scheduling problems with order rejection in the identical parallel machine environment. In terms of objective functions, the prevalent one was the minimization of the makespan of accepted jobs plus the total penalty costs of rejected jobs. Another typical objective was the maximization of the net profit as the difference between the sum of revenues and total weighted tardiness penalty costs (Emami et al., 2017, 2015; Naderi & Roshanaei, 2020; Wang et al., 2015; Wang & Ye, 2019; Wu et al., 2018). Most of the existing researches considered the deterministic OAS problems with identical parallel machines. Nevertheless, Emami et al. (2017, 2015) studied the robust optimization version in the uniform parallel machine environment, and Bruni et al. (2020) studied the distributionally robust optimization problem with job selection. To tackle the problems, numerous heuristics, exact and approximation methods were developed, as shown in Table 1.

This paper considers the objective of maximizing the profit which is the difference between the sum of revenues and the cost associated with the makespan of accepted jobs in the unrelated parallel machine environment, while the prevalent one in the literature (as shown in Table 1) is the identical parallel machine environment. Furthermore, sequence and machine-dependent setup times and the maximum available times of machines are additionally considered in this study.

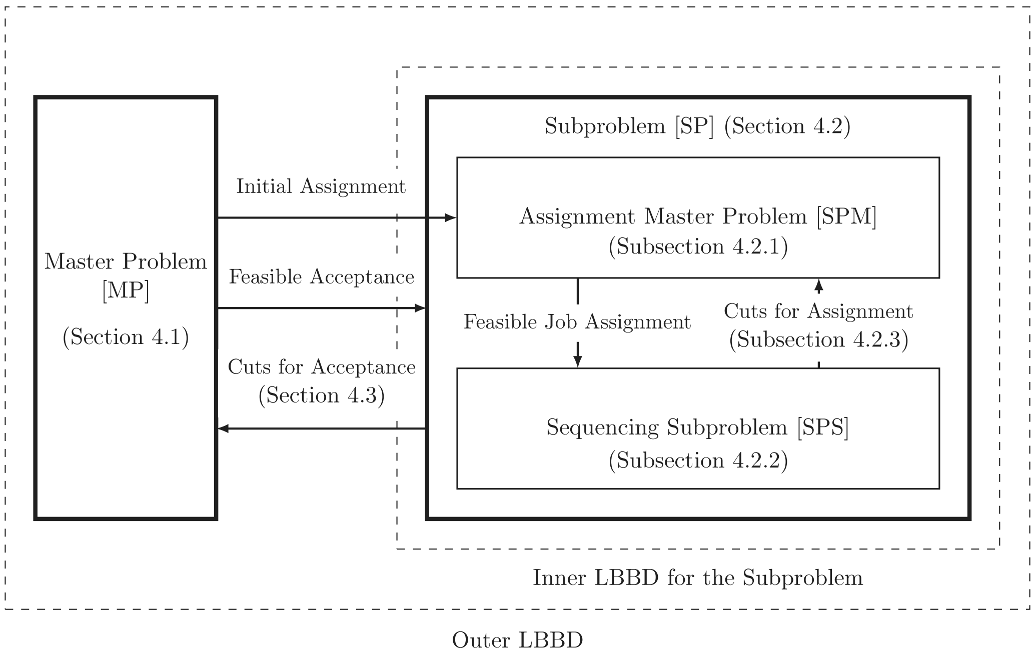
Master Problem [MP] (Section 4.1)
Initial Assignment
Feasible Acceptance
Cuts for Acceptance (Section 4.3)
Subproblem [SP] (Section 4.2)
Assignment Master Problem [SPM] (Subsection 4.2.1)
Feasible Job Assignment
Cuts for Assignment (Subsection 4.2.3)
Sequencing Subproblem [SPS] (Subsection 4.2.2)
Inner LBBD for the Subproblem
Outer LBBD

Fig. 1. The structure of the TL-LBBD method for $R_m|oa, sdst, mat|\sum z_j u_j - C_{max}$.

2.2. Parallel machine scheduling with sequence-dependent setup times

Setup times (costs) are ubiquitous in real-world industrial and service applications. Allahverdi (2015) has surveyed extensive literature including setup times. Here we mainly focus on the relevant and up-to-date studies on parallel machine scheduling problems with sequence-dependent setup times. In recent OAS-related literature with the parallel machine environment (shown in Table 1), sequence-dependent setup times were considered in Bruni et al. (2020), Emami et al. (2017, 2015) and Wu et al. (2018).

As for the unrelated parallel machine scheduling problem with sequence-dependent setup times, there exists rich literature on this topic and its variants (e.g., Bektur & Saraç, 2019; Ekici et al., 2019; Gedik et al., 2016). The sequence-dependent setup times in the unrelated parallel machine environment are usually machine-dependent as well. For example, Tran et al. (2016) proposed two decomposition methods, i.e., LBBD method and branch and check method, for the problem with the objective of makespan minimization. Fanjul-Peyro et al. (2019) considered the same problem in Tran et al. (2016) and proposed new MIPs, which adapted subtour elimination constraints and valid inequalities, and a mathematical programming based algorithm. Rauchecker and Schryen (2019a) studied the problem with the objective of minimizing the weighted sum of completion times in the disaster response settings. A branch-and-price algorithm was developed for scheduling collaborative rescue units to emerging incidents. Rauchecker and Schryen (2019b) focused on the problem with machine eligibility restrictions and the objective was to minimize the total weighted completion times. Yepes-Borrero et al. (2021) considered a bi-objective optimization problem with additional resources required during the setups. A heuristic method based on iterated greedy approaches was designed to find the Pareto front of the problem. Bitar et al. (2021) analyzed the complexity of the problem with auxiliary resources and different criteria. Carvalho and Nascimento (2022) proposed matheuristics for the integrated lot sizing and parallel machine scheduling problem with non-triangular setup costs and times, setup carry-over and capacity limitation.

There are also some researches on identical and uniform parallel machines scheduling problems with sequence-dependent setup times, including Kim and Kim (2020) and Kim and Lee (2021).

2.3. Parallel machine scheduling with limited resources

Limited resources are often encountered in the scheduling area due to various realistic production or service limitations and requirements. Here we exclusively review the most relevant literature on the parallel machine scheduling problems with limited resources, especially the problems with limited available time of machines (makespan). Ji et al. (2013) studied the uniform parallel machine scheduling problem where the objective was to minimize resource consumption given a level of the makespan. A tight lower bound and a particle swarm optimization algorithm were developed. Huo and Zhao (2015) derived an optimal polynomial time algorithm for the preemptive bi-criteria scheduling on parallel machines with the objective of minimizing the total completion time, machine unavailable intervals and a constant for the maximum makespan. Huo and Zhao (2018) studied the two-machine bi-criteria scheduling problems with arbitrary machine availability constraint. Either the makespan or the total completion time was minimized when the other criterion was the minimum. Optimal polynomial time algorithms were proposed for both problems. Kaabi and Harrath (2019) investigated the uniform parallel machine scheduling problem with one unavailability period for each machine and a given deadline for all jobs. They proposed optimal polynomial time algorithms and heuristics for the problems with different complexities. For the OAS problems and equivalent problems in parallel machine environments, other limited resources like the number of rejected jobs and the number of jobs in a batch were also considered (shown in Table 1).

To the best of our knowledge, the unrelated parallel machine scheduling problem integrated with order acceptance, sequence and machine-dependent setup times, and the limited resources like the maximum available times of machines, has not been studied yet in the research.

3. Problem statement and formulation

3.1. Problem statement

There is a set of jobs $\mathcal{N} = \{1, \dots, n\}$ and each job could be accepted to be scheduled on a set of machines $\mathcal{M} = \{1, \dots, m\}$. The objective is to maximize the profit, which is the difference between the total revenues

Table 1
Recent literature of OAS problems and equivalent problems in parallel machine environment.

Authors (year)	Type	A/R	Deterministic objective	PM	ST	RD/DD	Limited resources	Method
Emami et al. (2015)	RO	A	$max\{\sum_{j\in\mathcal{A}} u_j - \alpha_j T_j\}$	UN	SMD	DD	–	LR
Wang et al. (2015)	D	A	$max\{\sum_{j\in\mathcal{A}} u_j - \alpha_j T_j\}$	ID	–	DD	–	H, DP
Ou et al. (2015)	D	R	$min\{C_{max} + \sum_{j\in\mathcal{R}} w_j\}$	ID	–	–	–	H
Ou et al. (2016)	D	R	$min\{C_{max} + \sum_{j\in\mathcal{R}} w_j\}$	ID	–	–	machines for j	APXM
Emami et al. (2017)	RO	A	$max\{\sum_{j\in\mathcal{A}} u_j - \alpha_j T_j\}$	UN	SMD	DD	–	BD
Jiang et al. (2017)	D	A	$min\{\lambda C_{max} + (1-\lambda)(\sum_{j\in\mathcal{R}} w_j + TC)\}$	ID	–	–	number of j in a batch	APXM
Ou and Zhong (2017a)	D	A	$min\{C_{max} + \sum_{j\in\mathcal{R}} w_j\}$	ID	–	–	number (time) for $\|\mathcal{R}\|$ ($\mathcal{R}$)	APXM
Ou and Zhong (2017b)	D	A	$min\{C_{max} + \sum_{j\in\mathcal{R}} w_j\}$	ID	–	–	number for $\|\mathcal{R}\|$	H
Wang, Yin, and Cheng (2018)	D	R	$min\{\sum_{j\in\mathcal{A}} C_j\}$	ID	–	RD, DD	$\sum_{j\in\mathcal{R}} w_j \le U$, $\triangle_{max} \le Q$	DP, B&P
Wu et al. (2018)	D	A	$max\{\sum_{j\in\mathcal{A}} u_j - \alpha_j T_j\}$	ID	SD	DD	–	H
Wang and Ye (2019)	D	A	$max\{\sum_{j\in\mathcal{A}} u_j - \alpha_j T_j\}$	UR	–	DD	–	B&B
Bruni et al. (2020)	DRO	A	$min\{\sum_{j\in\mathcal{A}} C_j\}$	ID	SD	–	minimum profit level	H
Kong et al. (2020)	D	A	$max\{\sum_{i\in\mathcal{M}} \beta_i\}$	ID	–	RD	job deterioration, energy budget	H
Liu and Lu (2020)	D	R	$min\{C_{max} + \sum_{j\in\mathcal{R}} w_j\}$	ID	–	RD	–	APXM
Naderi and Roshanaei (2020)	D	A	$max\{\sum_{j\in\mathcal{A}} u_j - \alpha_j T_j\}$	ID	–	DD	–	BR&C, LBBD
This paper	**D**	**A**	$max\{\sum_{j\in\mathcal{A}} u_j - C_{max}\}$	**UR**	**SMD**	–	**machine available time**	**LBBD**

Notations:
j: the index of jobs; i: the index of machines; $\mathcal{A}$: the set of accepted jobs; $\mathcal{R}$: the set of rejected jobs; $\mathcal{M}$: the set of machines; C_j: the completion time of accepted job j; $\triangle_{max}$: the maximum completion time deviation among accepted jobs; C_{max}: the makespan of accepted jobs; w_j: the rejection penalty cost for rejected job j; u_j: the revenue for accepted job j; α_j: the unit tardiness penalty cost for accepted job j; T_j: the tardiness of accepted job j; β_i: the net revenue for machine i; λ: the constant parameter; TC: total costs for delivering orders; U: the given limit of total rejection penalty; Q: the given limit of the maximum completion time deviation.
Abbreviations:
D: Deterministic; RO: Robust optimization; DRO: Distributionally robust optimization; A: Acceptance; R: Rejection; PM: Parallel-machine environment; ID: Identical parallel machines; UN: Uniform parallel machines; UR: Unrelated parallel machines; ST: Setup times; SMD: Sequence and machine-dependent setup times; SD: Sequence-dependent setup times; RD: Release date; DD: Due date; H: Heuristics; LR: Lagrangian relaxation; DP: Dynamic programming; APXM: Approximation method; BD: Benders decomposition; B&P: Branch and price; B&B: Branch and bound; BR&C: Branch, relax and check; LBBD: Logic based Benders decomposition.

of accepted jobs and the cost associated with makespan, by taking into accounting various constraints including the maximum available time of each machine, sequence and machine-dependent setup times, and disjunctive constraints of jobs on the same machine. Note that the performance measure, makespan, in the objective could be seen as a linear cost function to the makespan and the unit cost is set to 1 for convenience. Each job could be accepted for processing with the revenue of u_j, $j \in \mathcal{N}$. We assume that there is no revenue or penalty if a job is not accepted. Each job j also has its processing time p_{ij} corresponding to the time required to process on machine i, $i \in \mathcal{M}$. The machines in the system are unrelated, and there are sequence and machine-dependent setup times, s_{ijk}: the time that needed between the completion of job j and the start of job k if job k is processed directly after job j on the same machine $i \in \mathcal{M}$. Each machine i has its maximum available time t_i. Therefore, the total processing times of jobs assigned on the machine i plus the total sequence-dependent setup times cannot exceed t_i. The goal of the problem is to determine whether a job should be accepted for processing or not, how to assign the accepted jobs to machines, and then how to sequence these jobs properly on each machine to maximize the profit.

One illustrative example of a feasible solution with 10 jobs and 3 machines is presented in Fig. 2. There are 10 considered jobs $\mathcal{N} = \{1, \dots, 10\}$ and 3 machines, M1, M2 and M3, with maximum available times $t_1 = 100$ and $t_2 = t_3 = 150$, respectively. Details of other parameters are listed in the appendix. The accepted jobs are $\{1, 2, 4, 5, 6, 7, 9\}$ and they are assigned to machine $[2, 2, 1, 3, 3, 1, 3]$. In the solution, on machine M1, job 7 is processed first and there is a sequence and machine-dependent setup time s_{107} with 14 time units before its processing. The earliest starting time for the processing of job 4 is 62 because job 7 completes at time 49 and s_{174} is 13 time units $(49 + 13 = 62)$. Then job 4 is completed with the processing time of 38 time units and the completion time $(62 + 38 = 100)$ reaches the maximum available time of machine M1 $(t_1 = 100)$. Machines M1 and M3 have already reached their maximum available times and the processing times plus the setup times of these jobs on machine M2 are larger than 8 time units, causing the corresponding machine exceeds its maximum available time if any of these jobs are processed after job 2.

3.2. Mathematical formulation of the problem

In this subsection, a MIP model is formulated for the above problem. The required indices, sets, parameters and decision variables are first defined as below.

Indices:
i: index of a machine, $i \in \{1, \dots, m\}$;
j, k: index of a job, $j, k \in \{1, \dots, n\}$.
Sets:
$\mathcal{M}$: set of machines, i.e., $\mathcal{M} = \{1, \dots, m\}$;
$\mathcal{N}$: set of jobs, i.e., $\mathcal{N} = \{1, \dots, n\}$;

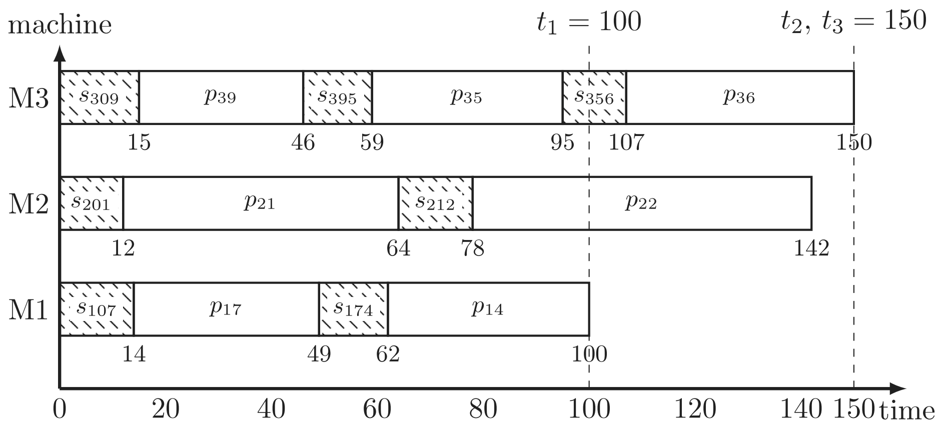

Fig. 2. An illustrative example of a feasible solution with 10 jobs and 3 machines.

$\mathcal{N}_0$: set of jobs with a dummy job 0, i.e., $\mathcal{N}_0 = \mathcal{N} \cup \{0\} = \{0, 1, \ldots, n\}$.

Parameters:

p_{ij}: the processing time of job j on machine i;

s_{ijk}: the sequence and machine-dependent setup time between the completion of job j and the start of job k if job k is processed immediately after job j on machine i. Each job j also has a setup time s_{i0j} if it is processed first on machine i. Moreover, it is assumed that the setup times satisfy the triangular inequality: $s_{ijk} \le s_{ijl} + s_{ilk}$, $j, k, l \in \mathcal{N}, j \ne k \ne l$. We also assume that $s_{ij0} = 0$, i.e., there is no setup time if job j is the last job on machine i;

t_i: the maximum available time of machine i;

u_j: the revenue of accepted job j;

L: a large enough positive integer value.

Decision variables:

z_j: 1 if job j is accepted, 0 otherwise;

x_{ij}: 1 if accepted job j is processed on machine i, $i \in \mathcal{M}$, $j \in \mathcal{N}$, 0 otherwise;

y_{ijk}: 1 if job k is processed directly after job j on machine i, $i \in \mathcal{M}$, $j, k \in \mathcal{N}_0, j \ne k$, 0 otherwise;

C_j: the completion time of job $j \in \mathcal{N}$;

C_{max}: the makespan or maximum completion time of the schedule.

Then the unrelated parallel machine scheduling problem with order acceptance and setup times (denoted by “RM-OAST” afterwards) can be formulated as follows.

$$\text{maximize} \sum_{j\in\mathcal{N}} u_j z_j - C_{max} \tag{1a}$$

subject to

$$\sum_{i\in\mathcal{M}} x_{ij} = z_j \qquad \forall j \in \mathcal{N}; \tag{1b}$$

$$x_{i0} = 1 \qquad \forall i \in \mathcal{M}; \tag{1c}$$

$$\sum_{j\in\mathcal{N}} y_{i0j} \le 1 \qquad \forall i \in \mathcal{M}; \tag{1d}$$

$$x_{ik} = \sum_{j\in\mathcal{N}_0, j\ne k} y_{ijk} \qquad \forall i \in \mathcal{M}, k \in \mathcal{N}_0; \tag{1e}$$

$$x_{ij} = \sum_{k\in\mathcal{N}_0, k\ne j} y_{ijk} \qquad \forall i \in \mathcal{M}, j \in \mathcal{N}_0; \tag{1f}$$

$$\sum_{j\in\mathcal{N}} x_{ij} p_{ij} + \sum_{j\in\mathcal{N}_0} \sum_{k\in\mathcal{N}_0, k\ne j} y_{ijk} s_{ijk} \le t_i \qquad \forall i \in \mathcal{M}; \tag{1g}$$

$$C_k - C_j + L(1 - y_{ijk}) \ge s_{ijk} + p_{ik} \qquad \forall j \in \mathcal{N}_0, k \in \mathcal{N}, j \ne k, i \in \mathcal{M}; \tag{1h}$$

$$C_j \le L z_j \qquad \forall j \in \mathcal{N}; \tag{1i}$$

$$C_0 = 0; \tag{1j}$$

$$C_{max} \ge C_j \qquad \forall j \in \mathcal{N}; \tag{1k}$$

$$C_j \le t_i + (1 - x_{ij})L \qquad \forall j \in \mathcal{N}, i \in \mathcal{M}; \tag{1l}$$

$$C_j \le \sum_{j\in\mathcal{N}} x_{ij} p_{ij} + \sum_{j\in\mathcal{N}_0} \sum_{k\in\mathcal{N}_0, k\ne j} y_{ijk} s_{ijk} + (1 - y_{ij0})L \qquad \forall j \in \mathcal{N}, i \in \mathcal{M}; \tag{1m}$$

$$z_j, x_{ij} \in \{0, 1\} \qquad \forall\ i \in \mathcal{M}, j \in \mathcal{N}; \tag{1n}$$

$$y_{ijk} \in \{0, 1\} \qquad \forall\ i \in \mathcal{M}, j, k \in \mathcal{N}_0, j \ne k; \tag{1o}$$

$$C_j, C_{max} \ge 0 \qquad \forall j \in \mathcal{N}; \tag{1p}$$

The objective function (1a) is to maximize the profit, which is the difference between the total revenues of accepted jobs and the cost of makespan. Constraints (1b) define the relationships between z_j and x_{ij}, and they impose that only a job is accepted can it be processed on a machine. Constraints (1c) impose that there is a dummy job on each machine. Constraints (1d) restrict that there is at most one first job on each machine. Constraints (1e) (resp., (1f)) ensure that each accepted job has a single predecessor (successor) on exactly one of the machines. Constraints (1g) enforce the maximum processing time for each machine. Constraints (1h) to (1m) define C_j and C_{max}. Specifically, constraints (1h) are the disjunctive constraints, which define the completion time of each job such that if job j precedes job k on machine i, the earliest completion time of job k must be greater than the completion time of job j plus the sequence-dependent setup time s_{ijk} and its processing on machine i, p_{ik}. Constraints (1i) restrict that $C_j = 0$ if $z_j = 0$. Constraint (1j) sets the completion time of the dummy job 0 to zero, enforcing the start of a schedule on each machine. Constraints (1k) define the makespan value. Constraints (1l) restrict that the completion time of accepted jobs must not exceed the maximum available time of the assigned machine. Constraints (1m) ensure that the completion time of the last job on each machine is bounded. Constraints (1n)–(1p) define the range of decision variables.

Remark 1. If $z_j = 1$, for all $j \in \mathcal{N}$, and $t_i = +\infty$, for all $i \in \mathcal{M}$, the RM-OAST problem reduces to $R_m|sdst|C_{max}$, which is exactly the NP-hard problem studied by Fanjul-Peyro et al. (2019) and Tran et al. (2016).

Therefore, the RM-OAST problem generalizes $R_m|sdst|C_{max}$ and is NP-hard, posing new challenges for the development of efficient approaches due to the intertwined decisions of acceptance, assignment and sequence of jobs. Hence, in this paper, we develop a novel two-layer LBBD-based exact decomposition method.

4. A two layer LBBD-based exact decomposition method

In this section, we provide the detailed two-layer LBBD-based exact decomposition method for the RM-OAST problem. The LBBD method is first proposed by Hooker (1994, 2007), Hooker and Ottosson (2003). Recently, Li, Côté, et al. (2022), Li, Li, et al. (2022), Wang et al. (2022) and Zhang et al. (2022) also use the LBBD method to deal with parallel machine scheduling related problems. Although the developed TL-LBBD method inherits the variable partitioning nature from the LBBD method, the major difference is that the TL-LBBD method separates the decisions about order acceptance, job assignment and sequencing in two layers and uses the subproblem of each layer to generate decision-specific cuts to the corresponding master problem. In other words, the developed TL-LBBD method explicitly utilizes the decision-partitioning structure and broadens the implementation of the LBBD method.

In the following, we explicitly define the master problem, the subproblem including the assignment master problem and the sequencing subproblem, the inferred cuts and bounds for the master problem and for the assignment master problem, and the complete procedure of the devised TL-LBBD method.

4.1. Master problem [MP]

The master problem (denoted by "[MP]") is a relaxation of the MIP model with (1a)–(1p). In this relaxation, the acceptance of jobs is determined and the accepted jobs are assigned to machines. Instead of requiring a single sequence of jobs on each machine, multiple disjoint sequences are allowed, as did in Tran et al. (2016). The relaxation essentially removes the decision variables C_j and C_{max}, and constraints (1h) to (1m) from the MIP model with (1a)–(1p). Furthermore, instead of removing y-variable from the model completely, y-variable is relaxed to be any real-valued number between 0 and 1, by which setup times could be partially considered in the master problem rather than be completely ignored. The master problem is formulated as follows:

$$[\mathbf{MP}] \quad \text{maximize} \sum_{j\in\mathcal{N}} u_j z_j - w \tag{2a}$$

subject to

constraints (1b)–(1g)

$$O_i = \sum_{j\in\mathcal{N}} x_{ij}p_{ij} + \sum_{j\in\mathcal{N}_0}\sum_{k\in\mathcal{N}_0,k\neq j} y_{ijk}s_{ijk} \qquad \forall i\in\mathcal{M}; \tag{2b}$$

$$w \geq O_i \qquad \forall i\in\mathcal{M}; \tag{2c}$$

$$\mathit{cuts} \tag{2d}$$

$$z_j, x_{ij} \in \{0,1\} \qquad \forall\, i\in\mathcal{M}, j\in\mathcal{N}; \tag{2e}$$

$$0 \leq y_{ijk} \leq 1 \qquad \forall\, i\in\mathcal{M}, j,k\in\mathcal{N}_0, j\neq k; \tag{2f}$$

$$O_i, w \geq 0 \qquad \forall\, i\in\mathcal{M}; \tag{2g}$$

where

O_i: is the maximum completion time on machine $i\in\mathcal{M}$;

w: is used to represent the C_{max} value;

cuts: represent the cuts that will be added to the master problem based on the feasibility of the subproblem. The cuts are defined in Section 4.3. Note that the set of cuts is empty in the first iteration of the master problem.

4.2. Subproblem [SP]

After solving [MP] in iteration h, an optimal solution with $\{\tilde{z}_j^{h*}, \tilde{x}_{ij}^{h*}, \tilde{y}_{ijk}^{h*}, \tilde{w}^{h*}\}$ can be found, providing the decision of acceptance or rejection of jobs and the assignment of each accepted job to one of m machines. Let $\tilde{\mathcal{J}}^h = \{j\in\mathcal{N} : \tilde{z}_j^{h*} = 1\}$ denote the set of accepted jobs in iteration h. Given $\tilde{\mathcal{J}}^h$, the subproblem (denoted by "[SP]") tends to obtain the optimal makespan of these accepted jobs on the unrelated parallel machines while the maximum available times for machines are not violated. If the subproblem is not able to yield a global feasible solution with given $\tilde{\mathcal{J}}^h$, the cuts which eliminate the current infeasible solution of accepted jobs are added to the master problem [MP] in the next iteration. Furthermore, $\tilde{x}_{ij}^{h*}$ could be utilized as an initial solution for the subproblem. However, since the sequence from the master problem [MP] may not be feasible due to the presence of multiple cycles on a single machine, the $\tilde{y}_{ijk}^{h*}$ values are ignored in the subproblem, as did in Tran et al. (2016).

As the subproblem could be transformed to the problem $R_m|sdst|C_{max}$ with the validation of the maximum available times for machines, which shows a suitable structure for decomposition methods as reported in Fanjul-Peyro et al. (2019) and Tran et al. (2016), we are able to further decompose [SP] into an assignment master problem (denoted by "[SPM]") and a sequencing subproblem (denoted by "[SPS]") using the inner LBBD.

4.2.1. Assignment master problem [SPM]

The assignment master problem [SPM] is formulated as follows:

$$[\mathbf{SPM}] \quad \text{minimize } w \tag{3a}$$

subject to

$$\sum_{i\in\mathcal{M}} x_{ij} = 1 \qquad \forall j\in\tilde{\mathcal{J}}; \tag{3b}$$

$$x_{i0} = 1 \qquad \forall i\in\mathcal{M}; \tag{3c}$$

$$\sum_{j\in\tilde{\mathcal{J}}} y_{i0j} \leq 1 \qquad \forall i\in\mathcal{M}; \tag{3d}$$

$$x_{ik} = \sum_{j\in\tilde{\mathcal{J}}_0, j\neq k} y_{ijk} \qquad \forall i\in\mathcal{M}, k\in\tilde{\mathcal{J}}_0; \tag{3e}$$

$$x_{ij} = \sum_{k\in\tilde{\mathcal{J}}_0, k\neq j} y_{ijk} \qquad \forall i\in\mathcal{M}, j\in\tilde{\mathcal{J}}_0; \tag{3f}$$

$$\sum_{j\in\tilde{\mathcal{J}}} x_{ij}p_{ij} + \sum_{j\in\tilde{\mathcal{J}}_0}\sum_{k\in\tilde{\mathcal{J}}_0,k\neq j} y_{ijk}s_{ijk} \leq t_i \qquad \forall i\in\mathcal{M}; \tag{3g}$$

$$O_i = \sum_{j\in\tilde{\mathcal{J}}} x_{ij}p_{ij} + \sum_{j\in\tilde{\mathcal{J}}_0}\sum_{k\in\tilde{\mathcal{J}}_0,k\neq j} y_{ijk}s_{ijk} \qquad \forall i\in\mathcal{M}; \tag{3h}$$

$$w \geq O_i \qquad \forall i\in\mathcal{M}; \tag{3i}$$

$$\mathit{cuts} \tag{3j}$$

$$x_{ij} \in \{0,1\} \qquad \forall\, i\in\mathcal{M}, j\in\tilde{\mathcal{J}}; \tag{3k}$$

$$0 \leq y_{ijk} \leq 1 \qquad \forall\, i\in\mathcal{M}, j,k\in\tilde{\mathcal{J}}_0, j\neq k; \tag{3l}$$

$$O_i, w \geq 0 \qquad \forall\, i\in\mathcal{M}; \tag{3m}$$

where

$\tilde{\mathcal{J}}$: is the set of accepted jobs determined by [MP] and the index h of $\tilde{\mathcal{J}}^h$ is dropped for simplicity of exposition;

$\tilde{\mathcal{J}}_0$: is the set of accepted jobs $\tilde{\mathcal{J}}$ and a dummy job 0;

cuts: represents the cuts that are generated by solving [SPS] and added to [SPM]. Note that the set of cuts is empty in the first iteration. These cuts are discussed in Section 4.2.3.

Note that constraints (3b) state that all accepted jobs should be assigned to machines. Constraints (3c) to (3g) correspond to constraints (1c) to (1g) respectively, except that $\mathcal{N}$ is replaced by $\tilde{\mathcal{J}}$.

4.2.2. Sequencing subproblem [SPS]

Let $\{\hat{x}'^{l}_{ij}, \hat{y}'^{l}_{ijk}, \hat{O}'^{l}_{i}, \hat{w}'^{l}\}$ denote the optimal solution of [SPM] in iteration l. Given the obtained assignments $\hat{x}'^{l}_{ij}$, we could fix the assignments and build a single sequence on each machine by formulating a mathematical formulation [SPS] to minimize the sum of the machine completion times as did in Fanjul-Peyro et al. (2019). In addition, let the decision variable α_j ($j \in \mathcal{N}$) be a lower limit on the number of jobs processed before job j on the machine where it is processed. The associated subtour elimination constraints (constraints (4f) to (4h)) are implemented as did in Fanjul-Peyro et al. (2019). Specifically, constraints (4f) ensure that if job k is the successor of job j on any machine ($\sum_{i\in\mathcal{M}} y_{ijk} = 1$), then $\alpha_j \le \alpha_k - 1$. Otherwise, an upper bound $|\tilde{\mathcal{J}}| - 1$ for the variable α_j is set. Constraints (4g) and (4h) together ensure that if job j is the first job on any machine ($\sum_{i\in\mathcal{M}} y_{i0j} = 1$), then $\alpha_j = 0$. The complete sequencing subproblem [SPS] is proposed as follows:

$$[\textbf{SPS}] \quad \text{minimize} \sum_{i\in\mathcal{M}} O_i \tag{4a}$$

subject to

$$\sum_{j\in\tilde{\mathcal{J}}} y_{i0j} \le 1 \qquad \forall i \in \mathcal{M}; \tag{4b}$$

$$\hat{x}'_{ik} = \sum_{j\in\tilde{\mathcal{J}}_0, j\ne k} y_{ijk} \qquad \forall i \in \mathcal{M}, k \in \tilde{\mathcal{J}}_0; \tag{4c}$$

$$\hat{x}'_{ij} = \sum_{k\in\tilde{\mathcal{J}}_0, k\ne j} y_{ijk} \qquad \forall i \in \mathcal{M}, j \in \tilde{\mathcal{J}}_0; \tag{4d}$$

$$O_i = \sum_{j\in\tilde{\mathcal{J}}} \hat{x}'_{ij} p_{ij} + \sum_{j\in\tilde{\mathcal{J}}_0} \sum_{k\in\tilde{\mathcal{J}}_0, k\ne j} y_{ijk} s_{ijk} \qquad \forall i \in \mathcal{M}; \tag{4e}$$

$$\alpha_j - \alpha_k + |\tilde{\mathcal{J}}| \sum_{i\in\mathcal{M}} y_{ijk} \le |\tilde{\mathcal{J}}| - 1 \qquad \forall j, k \in \tilde{\mathcal{J}}, j \ne k; \tag{4f}$$

$$\alpha_j + \sum_{i\in\mathcal{M}} y_{i0j} \ge 1 \qquad \forall j \in \tilde{\mathcal{J}}; \tag{4g}$$

$$\alpha_j + (|\tilde{\mathcal{J}}| - 1) \sum_{i\in\mathcal{M}} y_{i0j} \le |\tilde{\mathcal{J}}| - 1 \qquad \forall j \in \tilde{\mathcal{J}}; \tag{4h}$$

$$y_{ijk} \in \{0,1\} \qquad \forall\ i \in \mathcal{M}, j, k \in \tilde{\mathcal{J}}_0, j \ne k; \tag{4i}$$

$$O_i \ge 0 \qquad \forall\ i \in \mathcal{M}; \tag{4j}$$

$$0 \le \alpha_j \le |\tilde{\mathcal{J}}| - 1 \qquad \forall j \in \tilde{\mathcal{J}}; \tag{4k}$$

Note that in the above model [SPS], the iteration index l in $\hat{x}'^{l}_{ij}$ is also omitted for the simplicity of exposition. Recall that the objective of the subproblem is minimizing the makespan, while the sequencing subproblem does not yield the makespan directly. Therefore, let $\hat{w}^{l*} = max_{i\in\mathcal{M}}\{\hat{O}^{l*}_i\}$ denote the makespan obtained in iteration l. As the [SPS] ignores the limit of maximum available times for machines, we also need to check whether the maximum completion time on machine i, $\hat{O}^{l*}_i$, in iteration l exceeds the limit or not.

4.2.3. Cuts and bounds for the assignment master problem [SPM]

Now we introduce several cuts for the assignment master problem [SPM]. The cut about the makespan in Tran et al. (2016) can be included into [SPM]. For the sake of completeness, we restate this cut below.

Given the set of jobs assigned to machine i in iteration l of the [SPM], $\hat{\mathcal{J}}^l_i = \{j \in \tilde{\mathcal{J}}_0 : \hat{x}'^{l}_{ij} = 1\}$, let $\max Pre^l_j = \max_{k\in\hat{\mathcal{J}}^l_i, k\ne j}(s_{ikj})$ be the maximum setup time if job j directly succeeds another job that is accepted and assigned to the same machine i in iteration l. Let $\theta^l_{ij} = p_{ij} + \max Pre^l_j$. The cut on the makespan is as follows:

$$w \ge \hat{O}^{l*}_i - \sum_{j\in\hat{\mathcal{J}}^l_i} (1 - x_{ij})\theta^l_{ij} \tag{5}$$

To expedite the process, we introduce several cuts based on the feasibility of the solution of the [SPS]. If the obtained solution of the [SPS] is not feasible, i.e., $\hat{O}^{l*}_i > t_i$ for any machine i, we can yield the following "no-good" cuts (6):

$$\sum_{j\in\hat{\mathcal{J}}^l_i} x_{ij} \le |\hat{\mathcal{J}}^l_i| - 1 \tag{6}$$

The combinatorial cuts (7) are also included if the obtained solution of the [SPS] is feasible, i.e., $\hat{O}^{l*}_i \le t_i$ for all machines. Let $S^l_1 = \{i \in \mathcal{M}, j \in \tilde{\mathcal{J}} | \hat{x}'^{l}_{ij} = 1\}$ and $S^l_0 = \{i \in \mathcal{M}, j \in \tilde{\mathcal{J}} | \hat{x}'^{l}_{ij} = 0\}$ based on the results of $\hat{x}'^{l}_{ij}$ from [SPM] in iteration l. Then at next iteration $l+1$, the generated combinatorial cuts are as follows:

$$\sum_{i,j\in S^I_1} (1 - x_{ij}) + \sum_{i,j\in S^I_0} x_{ij} \ge 1 \quad \forall I = 1, \dots, l \tag{7}$$

As the assignment master problem [SPM] is a relaxation of the original subproblem [SP], the lower bound of the objective value (makespan) in iteration l could be tightened by the $\hat{w}'^{l}$ derived by the [SPM] in the following way:

$$w \ge \underset{I=1,\dots,l-1}{max} \{\hat{w}'^{I}\} \tag{8}$$

Similarly, the upper bound in iteration l could also be enhanced by selecting the minimum makespan among all feasible solutions, which could be represented by:

$$w \le \underset{I=1,\dots,l-1}{min} \{\hat{w}^{I*}\} \tag{9}$$

4.2.4. The complete procedure of the inner LBBD for subproblem [SP]

The complete procedure of the inner LBBD for the subproblem [SP] is summarized in Algorithm 1. Note that the procedure terminates when the [SPM] is infeasible or $\hat{w}^{l*} = \hat{w}'^{l}$. However, [SPM] may not yield a global feasible solution at all because too many jobs could be accepted by the [MP] and the maximum available times of machines are never fully satisfied. In this case, cuts for [MP] are derived, as shown in next subsection.

4.3. Cuts and bounds for the master problem [MP]

Here we present the cuts and bounds for the master problem [MP] since the information from the [SP] could be yielded by the aforementioned Algorithm 1.

Once we obtain the variables $\tilde{z}^{h*}_j$ from the master problem, the [SP] is used to check if there exists a feasible solution with minimum makespan for the given acceptance of jobs. If the [SP] is not feasible, and with $\tilde{\mathcal{J}}^h = \{j \in \mathcal{N} : \tilde{z}^{h*}_j = 1\}$, we can derive the following "no-good" cuts which cut off infeasible set of accepted jobs:

$$\sum_{j\in\tilde{\mathcal{J}}^h} z_j \le |\tilde{\mathcal{J}}^h| - 1 \tag{10}$$

The feasible solutions in the previous iterations should be cut off as well, such that at current iteration h, there must be the different scheme

Algorithm 1 The inner LBBD for [SP]

```
1: Input the problem information including $p_{ij}$, $s_{ijk}$ and $t_i$, and the results from [MP] including $\tilde{x}_{ij}^{h*}$ and $\tilde{\mathcal{J}}^h$.
2: Set the iteration index $l = 1$ and the current best makespan $\hat{w}_{best} = +\infty$.
3: Introduce $\tilde{x}_{ij}^{h*}$ as the initial solution for $x_{ij}$ in [SPM].
4: Solve [SPM] to optimality.
5: while [SPM] is feasible do
6:   Let $\{\hat{x}_{ij}^{\prime l}, \hat{y}_{ijk}^{\prime l}, \hat{O}_i^{\prime l}, \hat{w}^{\prime l}\}$ be a candidate optimal solution of [SPM].
7:   Solve the subproblem [SPS] to optimality with the job assignments given by $\hat{x}_{ij}^{\prime l}$ and let $\hat{O}_i^{l*}$ be the solution of the maximum completion time on machine $i$.
8:   if $\hat{O}_i^{l*} \leq t_i$ for all machines $i \in \mathcal{M}$ then
9:     Obtain the feasible makespan from the [SPS], $\hat{w}^{l*} = \max_{i\in\mathcal{M}}\{\hat{O}_i^{l*}\}$.
10:    if $\hat{w}^{l*} < \hat{w}_{best}$ then
11:      $\hat{w}_{best} = \hat{w}^{l*}$.
12:    end if
13:    if $\hat{w}^{l*} = \hat{w}^{\prime l}$ then
14:      Current solution to the [SPM] is the optimal solution. Go to Step 26.
15:    else
16:      Update the upper bound with cuts (9).
17:      Generate the combinatorial cuts (7).
18:    end if
19:  else
20:    Generate “no-good" cuts (6).
21:  end if
22:  Generate the logic-based Benders cuts (5) and update the lower bound with cuts (8).
23:  Add all cuts including cuts (5) to (9) to the [SPM].
24:  Let $l = l + 1$. Go to step 4.
25: end while
26: if $\hat{w}_{best} \neq +\infty$ then
27:   Output that the [SP] is feasible and the corresponding $\hat{w}_{best}$.
28: else
29:   Output that the [SP] is infeasible.
30: end if
31: STOP.
```

Table 2
Summary of parameter settings.

Parameter	Values
n	$\{10, 20, 30, 40\}$
m	$\{2, 3, 5\}$
p_{ij}	[30, 80]
t_i	$\{50 \times max\{2, \frac{n}{m} - 3\}, \dots, 50 \times \frac{n}{m}\}$
u_j	[100, 200]
s_{ijk}	[10, 20]

of job acceptance. Then at next iteration $h + 1$, we have the following combinatorial cuts (11):

$$\sum_{j\in\tilde{\mathcal{J}}^I} (1 - z_j) + \sum_{j\in\mathcal{N}\setminus\tilde{\mathcal{J}}^I} z_j \geq 1 \quad \forall I = 1, \dots, h \tag{11}$$

Note that the [SP] may obtain the minimum makespan, $\hat{w}_{best}^{h*}$, with given job acceptance $\tilde{z}_j^{h*}$ in iteration h. Therefore, we could derive a lower bound for the objective of the maximization problem in iteration $h + 1$, which is as follows:

$$\sum_{j\in\mathcal{N}} u_j z_j - w \geq \max_{h'=1,\dots,h} \Big(\sum_{j\in\mathcal{N}} u_j \tilde{z}_j^{h'*} - \hat{w}_{best}^{h'*}\Big) \tag{12}$$

4.4. The complete procedure of the TL-LBBD method for RM-OAST

The complete procedure of the TL-LBBD exact method for the problem RM-OAST is summarized in Algorithm 2. Recall that the structure of the method is shown in Fig. 1. Note that the stopping criterion is that either the [MP] is infeasible or the makespan derived from the [MP] is equal to the makespan found in the [SP].

Algorithm 2 The complete TL-LBBD method for RM-OAST

```
1: Input the problem information including $p_{ij}$, $s_{ijk}$, $t_i$ and $u_j$.
2: Set the iteration index $h = 1$ and the current best objective value $obj = -\infty$.
3: Solve [MP] to optimality.
4: while [MP] is feasible do
5:   Let $\{\tilde{z}_j^{h*}, \tilde{x}_{ij}^{h*}, \tilde{y}_{ijk}^{h*}, \tilde{w}^{h*}\}$ be a candidate optimal solution.
6:   Solve the [SP] with Algorithm 1.
7:   if the [SP] is feasible then
8:     Obtain the makespan from the [SP], $\hat{w}_{best}^{h*}$.
9:     Generate the combinatorial cuts (11).
10:    if $\sum_{j\in\mathcal{N}} u_j \tilde{z}_j^{h*} - \hat{w}_{best}^{h*} > obj$ then
11:      $obj = \sum_{j\in\mathcal{N}} u_j \tilde{z}_j^{h*} - \hat{w}_{best}^{h*}$
12:    end if
13:    if $\tilde{w}^{h*} = \hat{w}_{best}^{h*}$ then
14:      Current solution to the [MP] is the optimal solution, go to Step 25.
15:    else
16:      Update the lower bound with cuts (12).
17:    end if
18:  else
19:    Generate “no-good" cuts (10).
20:  end if
21:  Add cuts including (10), (11) and (12) to the [MP].
22:  Let $h = h + 1$. Go to step 3.
23: end while
24: Output the current best objective value $obj$.
25: STOP.
```

5. Computational experiments

In this section, extensive computational experiments on two different data sets are conducted to: (1) compare the performance of the developed TL-LBBD method against those of the MIP model and the classic LBBD method; and (2) to see what scales of the RM-OAST problem instances could be solved to optimality by the TL-LBBD method within a predefined time limit. The methods are coded in C++ programming language and the CPLEX 12.8 is used as the MIP solver. Note that all cuts and bounds are added iteratively, then the CPLEX solver with default settings is called in each iteration of the LBBD methods instead of implementing the callback functions. All computational experiments are run on a laptop running MacOS with 2.3 GHz Intel Core i9 CPU and 16 GB RAM memory. The time limit for each instance is set as 1800 s. Data sets are available on the link: https://github.com/Kennylovelatte/order-acceptance.

5.1. Classic LBBD method

First we introduce the classic LBBD method considered in our computational experiments. As did in Naderi and Roshanaei (2020), the binary decisions of order acceptance and assignment are solved in the master problem and the sequencing decisions are optimized in the subproblems. Therefore, in the classic LBBD method, we utilize the [MP] in Section 4.1 as the master problem and the [SPS] in Section 4.2.2 as the subproblem. In other words, the assignments generated in the [MP] is directly passed to the [SPS]. The stopping criteria of the classic LBBD method is similar to that of the inner LBBD for subproblem [SP] in

Table 3
Results for instances on data set I with $n = 10$.

(n,m)	No.	MIP				$TL-LBBD$				$CL-LBBD$			
		BFS_M	BFB_M	t_M (s)	Gap(%)	BFS_{TL}	$Impr_{TL}$(%)	t_{TL} (s)	If^*_{TL}	BFS_{CL}	$Impr_{CL}$(%)	t_{CL} (s)	If^*_{CL}
(10,2)	1	**886**	886	1	0	**886**	0	1	Y	**886**	0	1	Y
	2	**451**	451	1	0	**451**	0	1	Y	**451**	0	1	Y
	3	**1015**	1015	1	0	**1015**	0	1	Y	**1015**	0	1	Y
	4	**705**	705	1	0	**705**	0	1	Y	**705**	0	1	Y
	5	**692**	692	1	0	**692**	0	1	Y	**692**	0	1	Y
	6	**704**	704	2	0	**704**	0	1	Y	**704**	0	1	Y
	7	**609**	609	1	0	**609**	0	1	Y	**609**	0	1	Y
	8	**823**	823	1	0	**823**	0	1	Y	**823**	0	1	Y
	9	**816**	816	1	0	**816**	0	1	Y	**816**	0	1	Y
	10	**770**	770	1	0	**770**	0	1	Y	**770**	0	1	Y
Average		747.10	747.10	1.10	0.00	747.10	0.00	1.00	(10/10)	747.10	0.00	1.00	(10/10)
(10,3)	1	**1116**	1116	1	0	**1116**	0	1	Y	**1116**	0	1	Y
	2	**984**	984	1	0	**984**	0	1	Y	**984**	0	1	Y
	3	**850**	850	1	0	**850**	0	1	Y	**850**	0	1	Y
	4	**597**	597	1	0	**597**	0	1	Y	**597**	0	1	Y
	5	**657**	657	1	0	**657**	0	1	Y	**657**	0	1	Y
	6	**876**	876	1	0	**876**	0	1	Y	**876**	0	1	Y
	7	**716**	716	1	0	**716**	0	1	Y	**716**	0	1	Y
	8	**819**	819	1	0	**819**	0	1	Y	**819**	0	1	Y
	9	**960**	960	1	0	**960**	0	1	Y	**960**	0	1	Y
	10	**641**	641	1	0	**641**	0	1	Y	**641**	0	1	Y
Average		821.60	821.60	1.00	0.00	821.60	0.00	1.00	(10/10)	821.60	0.00	1.00	(10/10)
(10,5)	1	**1371**	1371	1	0	**1371**	0	1	Y	**1371**	0	1	Y
	2	**1120**	1120	1	0	**1120**	0	1	Y	**1120**	0	1	Y
	3	**970**	970	1	0	**970**	0	1	Y	**970**	0	1	Y
	4	**1047**	1047	1	0	**1047**	0	1	Y	**1047**	0	1	Y
	5	**1229**	1229	1	0	**1229**	0	1	Y	**1229**	0	1	Y
	6	**1163**	1163	1	0	**1163**	0	1	Y	**1163**	0	1	Y
	7	**1097**	1097	1	0	**1097**	0	1	Y	**1097**	0	1	Y
	8	**911**	911	1	0	**911**	0	1	Y	**911**	0	1	Y
	9	**1255**	1255	1	0	**1255**	0	1	Y	**1255**	0	1	Y
	10	**1048**	1048	1	0	**1048**	0	1	Y	**1048**	0	1	Y
Average		1121.10	1121.10	1.00	0.00	1121.10	0.00	1.00	(10/10)	1121.10	0.00	1.00	(10/10)
Overall avg.		896.60	896.60	1.03	0.00	896.60	0.00	1.00	(30/30)	896.60	0.00	1.00	(30/30)

Section 4.2, i.e., whether the master problem is infeasible or the value of the makespan derived by the master problem is equal to the value of the makespan generated by the subproblem.

Several cuts and bounds for the master problem are considered in the classic LBBD method. For convenience, we use the aforementioned symbols to distinguish the variables of the master problem and the subproblem. In each iteration, we add the following combinatorial cuts (13), LBBD cuts (14) and no-good cuts (15) :

$$\sum_{i\in\mathcal{M},j\in\mathcal{N},\tilde{x}_{ij}^{h*}=1}(1-x_{ij})+\sum_{i\in\mathcal{M},j\in\mathcal{N},\tilde{x}_{ij}^{h*}=0}x_{ij}\geq 1 \tag{13}$$

$$w\geq \hat{O}_i^{h*}-\sum_{j\in\mathcal{N},\tilde{x}_{ij}^{h*}=1}(1-x_{ij})\theta_{ij}^{h} \tag{14}$$

$$\sum_{j\in\mathcal{N},\tilde{x}_{ij}^{h*}=1}x_{ij}\leq|\hat{\mathcal{J}}_i^{h}|-1 \tag{15}$$

where the no-good cuts (15) are added when there is any machine with $\hat{O}_i^{h*} > t_i$, i.e., the maximum available time of a machine is violated. These cuts are similar to the cuts mentioned in Section 4.2.3 and could be adapted for the classic LBBD method. Also, the lower and upper bounds, (16) and (17), are added into the master problem of the classic LBBD method iteratively.

$$\sum_{j\in\mathcal{N}}u_jz_j-w\geq\max_{h'=1,\ldots,h}\left(\sum_{j\in\mathcal{N}}u_j\tilde{z}_j^{h'*}-\hat{w}^{h'*}\right) \tag{16}$$

$$\sum_{j\in\mathcal{N}}u_jz_j-w\leq\min_{h'=1,\ldots,h}\left(\sum_{j\in\mathcal{N}}u_j\tilde{z}_j^{h'*}-\tilde{w}^{h'*}\right) \tag{17}$$

where the lower bound (16) could be obtained when the subproblem finds a global feasible solution and the upper bound (17) could be generated by solving the master problem of the classic LBBD method, respectively.

5.2. Computational experiments with instances on data set I

5.2.1. Parameter settings for instances on data set I

The values of each parameter are summarized in Table 2. The number of received jobs n increases from 10 to 40 with increments of 10, i.e., $n \in \{10, 20, 30, 40\}$. The number of machines m is set to be $\{2, 3, 5\}$. For each combination of m and n, 10 random instances are generated. Therefore, the computational experiment includes $4\times3\times10 = 120$ instances in total. The processing times of each job on machines and the revenues for accepted jobs are generated from the integer uniform distributions between $[30, 80]$ and $[100, 200]$, respectively. The values of the maximum available times of machines are at least 100 and randomly selected between $50 \times max\{2, \frac{n}{m} - 3\}$ and $50 \times \frac{n}{m}$ with increments of 50.

As for the sequence and machine-dependent setup times, in order to fit the triangular inequality for the setup times, we apply the same generation process as did in Tran et al. (2016). First, two different sets of coordinates are created on a Cartesian plane for the jobs on each machine. The coordinates along the horizontal and vertical axis are randomly selected between $[0, 100]$. Next, the sequence-dependent setup times on each machine could be generated depending on the Manhattan distances between the coordinates of two jobs. Note that the second set of coordinates are essential to get the asymmetric setup times. Let (χ_x^j, χ_y^j) and (ψ_x^j, ψ_y^j) be the first and second set of coordinates for job j, respectively. The integer sequence-dependent setup times from job j to k on any machine would be $round(10 + 0.05(|\chi_x^j - \chi_x^k| + |\chi_y^j - \chi_y^k|))$ where the first set of coordinates is utilized and $round(v)$ is the default function in C++ that returns the nearest integral value of v. Then the integer setup times from job k to j would be $round(10 + 0.05(|\psi_x^k - \psi_x^j| + |\psi_y^k - \psi_y^j|))$ where the second set of coordinates is

Table 4
Results for instances on data set I with $n = 20$.

(n,m)	No.	MIP				$TL-LBBD$				$CL-LBBD$			
		BFS_M	BFB_M	t_M (s)	$Gap(\%)$	BFS_{TL}	$Impr_{TL}(\%)$	t_{TL} (s)	If^*_{TL}	BFS_{CL}	$Impr_{CL}(\%)$	t_{CL} (s)	If^*_{CL}
(20,2)	1	**1985**	2326.21	1800	17.19	**1985**	0	1	Y	**1985**	0	1	Y
	2	**1900**	2281.59	1800	20.08	**1900**	0	1	Y	**1900**	0	1	Y
	3	**2073**	2380.05	1800	14.81	**2073**	0	1	Y	**2073**	0	1	Y
	4	**1762**	1999.33	1800	13.47	**1762**	0	1	Y	**1762**	0	1	Y
	5	**1816**	2068.57	1800	13.91	**1816**	0	1	Y	**1816**	0	1	Y
	6	**1824**	2183.52	1800	19.71	**1824**	0	1	Y	**1824**	0	1	Y
	7	**1981**	2209.63	1800	11.54	**1981**	0	1	Y	**1981**	0	1	Y
	8	**1904**	2229.4	1800	17.09	**1904**	0	1	Y	**1904**	0	1	Y
	9	**1876**	2185.4	1800	16.49	**1876**	0	1	Y	**1876**	0	1	Y
	10	**1998**	2302.82	1800	15.26	**1998**	0	1	Y	**1998**	0	1	Y
Average		1911.90	2216.65	1800.00	15.94	1911.90	0.00	1.00	(10/10)	1911.90	0.00	1.00	(10/10)
(20,3)	1	**1804**	1804	956	0	**1804**	0	15	Y	**1804**	0	4	Y
	2	**1727**	1772.47	1800	2.63	**1727**	0	6	Y	**1727**	0	14	Y
	3	**1542**	1542	1075	0	**1542**	0	1	Y	**1542**	0	1	Y
	4	**1821**	1943.91	1800	6.75	**1821**	0	1	Y	**1821**	0	1	Y
	5	**1569**	1569	14	0	**1569**	0	1	Y	**1569**	0	3	Y
	6	**1470**	1470	63	0	**1470**	0	1	Y	**1470**	0	1	Y
	7	**2237**	2361.13	1800	5.55	**2237**	0	1	Y	**2237**	0	1	Y
	8	**1467**	1467	9	0	**1467**	0	1	Y	**1467**	0	1	Y
	9	**2070**	2070	563	0	**2070**	0	1	Y	**2070**	0	1	Y
	10	**1655**	1740.43	1800	5.16	**1655**	0	1	Y	**1655**	0	1	Y
Average		1736.20	1773.99	988.00	2.18	1736.20	0.00	2.90	(10/10)	1736.20	0.00	2.80	(10/10)
(20,5)	1	**1611**	1611	1	0	**1611**	0	1	Y	**1611**	0	1	Y
	2	**1979**	1979	20	0	**1979**	0	1	Y	**1979**	0	1	Y
	3	**2250**	2250	59	0	**2250**	0	1	Y	**2250**	0	2	Y
	4	**1882**	1882	8	0	**1882**	0	1	Y	**1882**	0	1	Y
	5	**1930**	1930	2	0	**1930**	0	1	Y	**1930**	0	1	Y
	6	**2133**	2133	97	0	**2133**	0	2	Y	**2133**	0	1	Y
	7	**2022**	2022	66	0	**2022**	0	5	Y	**2022**	0	9	Y
	8	**2262**	2326.97	1800	2.87	**2262**	0	13	Y	**2262**	0	11	Y
	9	**1793**	1793	6	0	**1793**	0	1	Y	**1793**	0	1	Y
	10	**1562**	1562	1	0	**1562**	0	1	Y	**1562**	0	1	Y
Average		1942.40	1948.90	206.00	0.33	1942.40	0.00	2.70	(10/10)	1942.40	0.00	2.90	(10/10)
Overall avg.		1863.50	1979.85	998.00	6.24	1863.50	0.00	2.20	(30/30)	1863.50	0.00	2.23	(30/30)

utilized likewise. Hence, the values of s_{ijk} are determined in the range of $[10, 20]$.

Four common performance indicators for the MIP model (denoted by "MIP") are considered, including the objective value of the best found feasible solution (denoted by "BFS_M"), the best upper bound found by the CPLEX solver (denoted by "BFB_M"), the computation time (denoted by "t_M (s)") and the optimality gap (denoted by "$Gap(\%)$"). The optimality gap for the MIP model ($Gap(\%)$) is calculated as follows: $Gap(\%) = 100\% \times \frac{BFB_M - BFS_M}{BFS_M}$. When the optimality gap reaches zero ($Gap = 0\%$), it means that the global optimal solution of the instance has been found by solving the MIP model directly.

The performance indicators for the developed TL-LBBD method (denoted by "$TL - LBBD$") and the classic LBBD method (denoted by "$CL - LBBD$") are identical, consisting of the objective values of the best found feasible solutions obtained by both LBBD methods which are denoted by "BFS_{TL}" and "BFS_{CL}", the improvement of BFS_{TL} (resp. BFS_{CL}) with respect to BFS_M which is denoted by "$Impr_{TL}(\%)$" (resp. $Impr_{CL}(\%)$), the computation time of two LBBD methods which are denoted by "t_{TL} (s)"and "t_{CL} (s) ", and whether the LBBD methods obtain the optimal solution or not (denoted by "If^*_{TL}" and "If^*_{CL}"). The improvement $Impr_{TL}(\%)$ for the TL-LBBD method (resp. the classic LBBD method) is calculated as follows: $Impr_{TL}(\%) = 100\% \times \frac{BFS_{TL} - BFS_M}{BFS_M}$ (resp. $Impr_{CL}(\%) = 100\% \times \frac{BFS_{CL} - BFS_M}{BFS_M}$). If the TL-LBBD method (resp. the classic LBBD method) terminates within 1800 s, the corresponding BFS_{TL} (resp. BFS_{CL}) is the global optimal solution and it is denoted by "Y" under the column "If^*_{TL}" (resp. "If^*_{CL}"). Otherwise, the global optimality of the solution is not certain (denoted by "N" under the corresponding column). Note that either the TL-LBBD method or the classic LBBD method may not yield a global feasible solution within the time limit. The symbols "-" and "$N.F.$" under the columns "BFS_{TL}", "$Impr_{TL}(\%)$" and "If^*_{TL}" (resp. "BFS_{CL}", "$Impr_{CL}(\%)$" and "If^*_{CL}") all represent that no feasible solution has been found by the TL-LBBD method (resp. the classic LBBD method) in 1800 s and the relative gap is not available. In addition, the best found objective value for each instance is highlighted in bold.

5.2.2. Computational results for instances on data set I

The computational results with comparisons are presented in Tables 3–6. Each table corresponds to the instances with a different size of received jobs n, where the first column of each table refers the combination of number of received jobs n and number of machines m, and the second column "No". identifies the indicators of instances. The results for solving the MIP model directly, the developed TL-LBBD method and the classic LBBD method are separated into three sets of columns under "MIP", "$TL - LBBD$" and "$CL - LBBD$", where the corresponding performance indicators are presented, respectively. For each set of (n, m) instances, the average results are summarized in the "Average" row. In the rows of "Average" under the columns "If^*_{TL}" and "If^*_{CL}", the number of instances solved to optimality is summarized.

With the results in Tables 3–6, some key observations could be summarized as follows:

(1) The overall results show that the developed TL-LBBD method indeed allow significant reduction in terms of computation time, compared to solving the MIP model in CPLEX directly and the classic LBBD method. Specifically, the solving process of the MIP model reaches the time limit in the majority of instances, especially for those with $n \geq 20$. The classic LBBD method is able to reach the optimality in much less computation time, achieving 79.43 and 328.93 s on overall average for instances with $n = 30$

Table 5
Results for instances on data set I with $n = 30$.

(n, m)	No.	MIP				$TL-LBBD$				$CL-LBBD$			
		BFS_M	BFB_M	t_M (s)	Gap(%)	BFS_{TL}	$Impr_{TL}$(%)	t_{TL} (s)	If^*_{TL}	BFS_{CL}	$Impr_{TL}$(%)	t_{CL} (s)	If^*_{CL}
(30,2)	1	**2721**	3382.1	1800	24.30	**2721**	0	1	Y	**2721**	0	1	Y
	2	**2854**	3447.75	1800	20.80	**2854**	0	1	Y	**2854**	0	1	Y
	3	**3095**	3706.98	1800	19.77	**3095**	0	1	Y	**3095**	0	1	Y
	4	3151	3789.1	1800	20.25	**3160**	0.29	1	Y	**3160**	0.29	1	Y
	5	2954	3512.64	1800	18.91	**2955**	0.03	1	Y	**2955**	0.03	1	Y
	6	**3343**	4020.19	1800	20.26	**3343**	0	1	Y	**3343**	0	1	Y
	7	**3216**	3821.07	1800	18.81	**3216**	0	1	Y	**3216**	0	1	Y
	8	**3564**	4195.43	1800	17.72	**3564**	0	1	Y	**3564**	0	1	Y
	9	**3295**	3955.51	1800	20.05	**3295**	0	1	Y	**3295**	0	1	Y
	10	3063	3659.91	1800	19.49	**3065**	0.07	1	Y	**3065**	0.07	1	Y
Average		3125.60	3749.07	1800.00	19.95	3126.80	0.04	1.00	(10/10)	3126.80	0.04	1.00	(10/10)
(30,3)	1	**3352**	3730.75	1800	11.30	**3352**	0	3	Y	**3352**	0	4	Y
	2	3048	3431.53	1800	12.58	**3053**	0.16	38	Y	**3053**	0.16	106	Y
	3	3409	3884.55	1800	13.95	**3456**	1.38	2	Y	**3456**	1.38	2	Y
	4	**3176**	3595.72	1800	13.22	**3176**	0	5	Y	**3176**	0	4	Y
	5	3461	3973.87	1800	14.82	**3539**	2.25	3	Y	**3539**	2.25	5	Y
	6	**3501**	3896.45	1800	11.30	**3501**	0	3	Y	**3501**	0	1	Y
	7	3249	3637.67	1800	11.96	**3252**	0.09	7	Y	**3252**	0.09	5	Y
	8	**3324**	3748.37	1800	12.77	**3324**	0	1	Y	**3324**	0	1	Y
	9	**3761**	4203.48	1800	11.76	**3761**	0	1	Y	**3761**	0	1	Y
	10	**2918**	3281.81	1800	12.47	**2918**	0	26	Y	2918	0	18	Y
Average		3319.90	3738.42	1800.00	12.61	3333.20	0.40	8.90	(10/10)	3333.20	0.40	14.70	(10/10)
(30,5)	1	**3335**	3509.11	1800	5.22	**3335**	0	25	Y	**3335**	0	670	Y
	2	**3085**	3220.14	1800	4.38	**3085**	0	2	Y	**3085**	0	3	Y
	3	**3535**	3743.02	1800	5.88	**3535**	0	21	Y	**3535**	0	43	Y
	4	**3128**	3331.01	1800	6.49	**3128**	0	69	Y	**3128**	0	111	Y
	5	**2884**	2933.53	1800	1.72	**2884**	0	1	Y	**2884**	0	2	Y
	6	**3088**	3299.58	1800	6.85	**3088**	0	10	Y	**3088**	0	167	Y
	7	**2663**	2732.69	1800	2.62	**2663**	0	2	Y	**2663**	0	1	Y
	8	**3321**	3360.64	1800	1.19	**3321**	0	1	Y	**3321**	0	1	Y
	9	3498	3690.39	1800	5.50	**3505**	0.20	27	Y	**3505**	0.20	1183	Y
	10	4148	4491.06	1800	8.27	**4208**	1.45	41	Y	**4208**	1.45	45	Y
Average		3268.50	3431.12	1800.00	4.98	3275.20	0.20	19.90	(10/10)	3275.20	0.20	222.60	(10/10)
Overall avg.		3238.00	3639.54	1800.00	12.40	3245.07	0.22	9.93	(30/30)	3245.07	0.22	79.43	(30/30)

and $n = 40$. The developed TL-LBBD method converges to global optimum in 9.93 s and 194.07 s on overall average for instances with $n = 30$ and $n = 40$, respectively.

(2) The developed TL-LBBD method also evidently outperforms solving the MIP model directly and the classic LBBD method in terms of the quality of the solutions found within the time limit. As for solving the MIP model directly, some instances with $n = 20$ could not be solved to optimality within 1800 s. The average relative gaps between BFS_M and BFB_M are both larger than 10% for instances with $n = 30$ and $n = 40$. As for the classic LBBD method, all optimal solutions have been found for instances with $n \leq 30$. However, it could not converge to the optimal solutions in two instances with $(n, m) = (40, 5)$ and cannot find any feasible solution within 1800 s for instance No. 8 with $(n, m) = (40, 5)$. In contrast, the developed TL-LBBD method can obtain the optimal solution in 337 s for instance No. 8 with $(n, m) = (40, 5)$.

(3) For both the TL-LBBD method and the classic LBBD method, the total number of solved instances is 118 and 117 out of 120, respectively, leaving several instances with $n = 40, m = 5$ unsolved or without proven global optimality. One possible reason is that the master problem of the general LBBD method usually requires a huge effort to be solved to optimality, which is also reported in Tran et al. (2016). Since the RM-OAST is NP-hard, a global feasible solution may not be yielded when the time limit is reached.

5.3. Computational experiments with instances on data set II

In this section, the question of "what scales of the RM-OAST problem instances that can be solved to optimality by the developed TL-LBBD method within 1800 seconds" is addressed, by evaluating the TL-LBBD method with instances with different scales. As shown in Table 6, it is reasonable to say that $(n, m) = (40, 5)$ is the maximum level of the RM-OAST problem instances that could be solved to optimality by the developed TL-LBBD method within 1800 s. Consequently, only the instances with fewer machines m and more received jobs n are needed to be tested.

Next, computational experiments with instances on data set II are conducted considering the developed TL-LBBD method, the classic LBBD method and solving the MIP model directly. Parameters of the problem instances and performance indicators of these methods remain the same as in Section 5.2.1. For instances on data set II, $(n, m) = \{(80, 2), (80, 3), (100, 2), (100, 3), (200, 2)\}$ are considered. The computational results are shown in Table 7 and some observations are summarized as follows:

(1) The computation time of solving the MIP model directly always reach the predefined limit of 1800 s. As for the LBBD methods, the TL-LBBD method still outperforms the classic LBBD method in terms of computation time, achieving 289.92 s versus 433.94 s on overall average results.

(2) As for the solution quality of solving the MIP model directly, none of the instances are solved to optimality and the overall average gap is 19.34%. Furthermore, the corresponding BFS_M is worse than the best found objective values found by the LBBD methods (BFS_{TL} and BFS_{CL}) in all instances. The classic LBBD method could not find any feasible solution in 2 out of 10 instances with $(n, m) = (100, 3)$ whereas the developed TL-LBBD method at least finds feasible solutions in 1800 s. Note that in 10 instances with $(n, m) = (80, 3)$, the classic LBBD method cannot yield the proven optimal solutions in 2 instances whereas all instances are solved to optimality with the TL-LBBD method in 118.40 s on average.

Table 6
Results for instances on data set I with $n = 40$.

(n,m)	No.	MIP				$TL-LBBD$				$CL-LBBD$			
		BFS_M	BFB_M	t_M (s)	$Gap(\%)$	BFS_{TL}	$Impr_{TL}(\%)$	t_{TL} (s)	If^*_{TL}	BFS_{CL}	$Impr_{CL}(\%)$	t_{CL} (s)	If^*_{CL}
(40,2)	1	**4178**	4947.58	1800	18.42	**4178**	0	1	Y	**4178**	0	1	Y
	2	4233	5181.19	1800	22.40	**4247**	0.33	2	Y	**4247**	0.33	1	Y
	3	4006	4888.53	1800	22.03	**4015**	0.22	1	Y	**4015**	0.22	1	Y
	4	4586	5528.13	1800	20.54	**4587**	0.02	1	Y	**4587**	0.02	1	Y
	5	4126	4957.35	1800	20.15	**4128**	0.05	1	Y	**4128**	0.05	1	Y
	6	4231	5186.86	1800	22.59	**4263**	0.76	1	Y	**4263**	0.76	1	Y
	7	**4107**	4975.95	1800	21.16	**4107**	0	1	Y	**4107**	0	2	Y
	8	4353	5235.87	1800	20.28	**4354**	0.02	1	Y	**4354**	0.02	2	Y
	9	**4375**	5303.97	1800	21.23	**4375**	0	2	Y	**4375**	0	1	Y
	10	**4232**	5158.79	1800	21.90	**4232**	0	2	Y	**4232**	0	2	Y
Average		4242.70	5136.42	1800.00	21.06	4248.60	0.14	1.30	(10/10)	4248.60	0.14	1.30	(10/10)
(40,3)	1	4593	5210.56	1800	13.45	**4613**	0.44	14	Y	**4613**	0.44	10	Y
	2	**4372**	4946.85	1800	13.15	**4372**	0	4	Y	**4372**	0	10	Y
	3	4111	4645.46	1800	13.00	**4122**	0.27	21	Y	**4122**	0.27	57	Y
	4	4115	4616.64	1800	12.19	**4126**	0.27	16	Y	**4126**	0.27	13	Y
	5	4234	4869.76	1800	15.02	**4250**	0.38	13	Y	**4250**	0.38	39	Y
	6	**5193**	5790.81	1800	11.51	**5193**	0	5	Y	**5193**	0	8	Y
	7	**4210**	4735.73	1800	12.49	**4210**	0	5	Y	**4210**	0	9	Y
	8	**4595**	5122.08	1800	11.47	**4595**	0	2	Y	**4595**	0	1	Y
	9	**4412**	4985.55	1800	13.00	**4412**	0	1	Y	**4412**	0	1	Y
	10	**4510**	5022.17	1800	11.36	**4510**	0	1	Y	**4510**	0	1	Y
Average		4434.50	4994.56	1800.00	12.63	4440.30	0.13	8.20	(10/10)	4440.30	0.13	14.90	(10/10)
(40,5)	1	4746	5121.86	1800	7.92	**4750**	0.08	86	Y	**4750**	0.08	896	Y
	2	4688	5081.68	1800	8.40	**4689**	0.02	1800	N	**4689**	0.02	1800	N
	3	5092	5560.4	1800	9.20	**5172**	1.57	178	Y	**5172**	1.57	517	Y
	4	4750	5122.49	1800	7.84	**4752**	0.04	1800	N	**4752**	0.04	1800	N
	5	5042	5446.82	1800	8.03	**5092**	0.99	17	Y	**5092**	0.99	10	Y
	6	4916	5346.04	1800	8.75	**4946**	0.61	1086	Y	**4946**	0.61	827	Y
	7	4633	5083.37	1800	9.72	**4642**	0.19	204	Y	**4642**	0.19	320	Y
	8	**4318**	4737.92	1800	9.72	**4318**	0	337	Y	–	–	1800	N.F.
	9	**4150**	4455.32	1800	7.36	**4150**	0	113	Y	**4150**	0	1641	Y
	10	4883	5283.68	1800	8.21	**4890**	0.14	106	Y	**4890**	0.14	95	Y
Average		4721.80	5123.96	1800.00	8.52	4805.67	1.78	572.70	(8/10)	4787.00	1.38	970.60	(7/10)
Overall avg.		4466.33	5084.98	1800.00	13.85	4487.59	0.48	194.07	(28/30)	4481.79	0.35	328.93	(27/30)

Similar results could also be observed in instances with $(n,m) = (200,2)$. Overall, 46 out of 50 instances are solved to optimality by the developed TL-LBBD method within 1800 s, whereas only 42 out of 50 instances are solved by the classic LBBD method.

(3) Since the developed TL-LBBD method fails to find the proven optimal solutions within the time limit in some instances, it is reasonable to say that the maximum levels of the RM-OAST problem instances that could be solved to optimality by the developed TL-LBBD method within 1800 s are $(n,m) = (100,3)$ and $(n,m) = (200,2)$, respectively.

5.4. Analysis of number of cuts in LBBD methods

In this subsection, we report the information of the cuts generated in the developed TL-LBBD method and the classic LBBD method respectively. The average results of generated number of cuts in both LBBD methods are summarized in Table 8. The number of total cuts generated by the developed TL-LBBD methods is much more than that of the classic LBBD method on both data sets. We believe that such a decision-partitioning structure in the developed TL-LBBD method is able to alleviate the computational challenge of the original problem by adding more decision-specific cuts proactively. Moreover, the LBBD cuts, including cuts (5) in the TL-LBBD method and cuts (14) in the classic LBBD method, are the majority cuts over the others.

6. Conclusion

In this paper, we investigated an unrelated parallel machine scheduling problem with order acceptance, machine- and sequence dependent setup times and the maximum available times of machines. A MIP model was formulated for the problem with the objective of maximizing the profit, which is the difference between the total revenues of accepted jobs and the cost of makespan. To solve this problem efficiently, we developed a two layer LBBD-based exact decomposition method, in which the natural separable structure of the problem was exploited and the LBBD method was deployed twice consequently.

The computational experiments on two different data sets demonstrated that the proposed TL-LBBD exact method is significantly effective, achieving a great reduction of computation time. Furthermore, the TL-LBBD method evidently outperformed the classic LBBD method considered in the computational experiments in terms of both the speed and solution quality. Moreover, the maximum scales of the RM-OAST problem that can be solved to optimality by the TL-LBBD method within 1800 s were further discussed. The computational results showed that the problem instances with up to $(n,m) = (40,5)$, $(n,m) = (100,3)$ and $(n,m) = (200,2)$ could be solved to optimality within 30 min by the developed TL-LBBD method.

The developed TL-LBBD method has broadened the implementation of the LBBD method on parallel machine scheduling problems and OAS problems. Also, it may be applied to real-world applications of the RM-OAST problem in the semiconductor manufacturing companies and healthcare industries, helping the decision makers plan their schedules in an efficient way. Furthermore, the decision-partitioning scheme of the developed TL-LBBD method can be extended to other integrated scheduling problems with multiple intertwined decisions.

For future work, firstly, stronger bounds and valid inequalities could be further explored for the master problem of the TL-LBBD method. Secondly, different profit-oriented objectives, like maximizing the profit which is the difference between the total revenues and the total tardiness costs of the accepted jobs, could be considered. Thirdly,

Table 7
Results for instances on data set II.

(n, m)	No.	MIP				$TL-LBBD$				$CL-LBBD$			
		BFS_M	BFB_M	t_M (s)	Gap(%)	BFS_{TL}	$Impr_{TL}$(%)	t_{TL} (s)	If^*_{TL}	BFS_{CL}	$Impr_{CL}$(%)	t_{CL} (s)	If^*_{CL}
(80,2)	1	8634	10468.5	1800	21.25	**8635**	0.01	52	Y	**8635**	0.01	229	Y
	2	8482	10266.0	1800	21.03	**8489**	0.08	30	Y	**8489**	0.08	33	Y
	3	8698	10634.3	1800	22.26	**8706**	0.09	11	Y	**8706**	0.09	9	Y
	4	8680	10610.6	1800	22.24	**8687**	0.08	6	Y	**8687**	0.08	4	Y
	5	8984	10931.7	1800	21.68	**8998**	0.16	3	Y	**8998**	0.16	6	Y
	6	8656	10542.9	1800	21.80	**8657**	0.01	2	Y	**8657**	0.01	1	Y
	7	8762	10661.5	1800	21.68	**8779**	0.19	10	Y	**8779**	0.19	19	Y
	8	8886	10819.9	1800	21.76	**8890**	0.05	15	Y	**8890**	0.05	22	Y
	9	8804	10640.8	1800	20.86	**8857**	0.60	9	Y	**8857**	0.60	29	Y
	10	8465	10348.2	1800	22.25	**8466**	0.01	5	Y	**8466**	0.01	3	Y
Average		8705.10	10592.44	1800.00	21.68	8716.40	0.13	14.30	(10/10)	8716.40	0.13	35.50	(10/10)
(80,3)	1	9740	10891.3	1800	11.82	**9745**	0.05	115	Y	**9745**	0.05	328	Y
	2	9649	10926.2	1800	13.24	**9678**	0.30	11	Y	**9678**	0.30	8	Y
	3	9217	10480.2	1800	13.71	**9239**	0.24	412	Y	**9239**	0.24	1800	N
	4	9797	10957.7	1800	11.85	**9826**	0.30	40	Y	**9826**	0.30	45	Y
	5	9764	11033.8	1800	13.00	**9779**	0.15	35	Y	**9779**	0.15	51	Y
	6	9871	11130.9	1800	12.76	**9903**	0.32	91	Y	**9903**	0.32	424	Y
	7	9303	10584.5	1800	13.78	**9376**	0.78	197	Y	**9376**	0.78	1800	N
	8	9266	10464.2	1800	12.93	**9331**	0.70	60	Y	**9331**	0.70	125	Y
	9	10082	11330.5	1800	12.38	**10087**	0.05	127	Y	**10087**	0.05	217	Y
	10	8891	10077.4	1800	13.34	**8925**	0.38	96	Y	**8925**	0.38	246	Y
Average		9558.00	10787.67	1800.00	12.87	9588.90	0.32	118.40	(10/10)	9588.90	0.32	504.40	(8/10)
(100,2)	1	11432	13882.9	1800	21.44	**11440**	0.07	5	Y	**11440**	0.07	2	Y
	2	11094	13512.8	1800	21.80	**11127**	0.30	34	Y	**11127**	0.30	81	Y
	3	11371	13739.7	1800	20.83	**11404**	0.29	12	Y	**11404**	0.29	5	Y
	4	10964	13273.1	1800	21.06	**10987**	0.21	16	Y	**10987**	0.21	35	Y
	5	10923	13207.7	1800	20.92	**10929**	0.05	6	Y	**10929**	0.05	2	Y
	6	10002	12330.9	1800	23.28	**10054**	0.52	14	Y	**10054**	0.52	8	Y
	7	10942	13319.6	1800	21.73	**10986**	0.40	7	Y	**10986**	0.40	8	Y
	8	10964	13420.8	1800	22.41	**10985**	0.19	12	Y	**10985**	0.19	15	Y
	9	10911	13393.2	1800	22.75	**10967**	0.51	23	Y	**10967**	0.51	36	Y
	10	10714	13113.2	1800	22.39	**10736**	0.21	4	Y	**10736**	0.21	2	Y
Average		10931.70	13319.39	1800	21.84	10961.50	0.27	13.30	(10/10)	10961.50	0.27	19.40	(10/10)
(100,3)	1	11983	13583.5	1800	13.36	**12028**	0.38	1800	N	–	–	1800	N.F.
	2	11936	13450.6	1800	12.69	**11953**	0.14	150	Y	**11953**	0.14	210	Y
	3	12042	13629.6	1800	13.18	**12086**	0.37	1048	Y	**12086**	0.37	1800	N
	4	12353	13961.8	1800	13.02	**12378**	0.20	410	Y	**12378**	0.20	1626	Y
	5	12143	13768.6	1800	13.39	**12213**	0.58	764	Y	**12213**	0.58	1382	Y
	6	12512	14134.9	1800	12.97	**12600**	0.70	1800	N	–	–	1800	N.F.
	7	12626	14243.1	1800	12.81	**12656**	0.24	91	Y	**12656**	0.24	111	Y
	8	12440	14030.6	1800	12.79	**12443**	0.02	25	Y	**12443**	0.02	23	Y
	9	12071	13663.2	1800	13.19	**12128**	0.47	25	Y	**12128**	0.47	10	Y
	10	11833	13453.8	1800	13.70	**11869**	0.30	177	Y	**11869**	0.30	122	Y
Average		12193.90	13791.97	1800.00	13.11	12235.40	0.34	629.00	(8/10)	12215.75	0.18	888.40	(7/10)
(200,2)	1	20100	27573	1800	37.18	**22753**	13.20	18	Y	**22753**	13.20	43	Y
	2	22352	27299.6	1800	22.13	**22382**	0.13	1759	Y	**22382**	0.13	1800	N
	3	22299	27212.5	1800	22.03	**22351**	0.23	1800	N	**22351**	0.23	1800	N
	4	22084	26995.5	1800	22.24	**22139**	0.25	83	Y	**22139**	0.25	92	Y
	5	22238	27116.1	1800	21.94	**22308**	0.31	1800	N	**22308**	0.31	1800	N
	6	23027	27994.9	1800	21.57	**23093**	0.29	96	Y	**23093**	0.29	94	Y
	7	22018	26925.7	1800	22.29	**22060**	0.19	930	Y	**22060**	0.19	1360	Y
	8	22508	27475.5	1800	22.07	**22568**	0.27	80	Y	**22568**	0.27	71	Y
	9	22007	26896.6	1800	22.22	**22033**	0.12	33	Y	**22033**	0.12	40	Y
	10	22196	27040.7	1800	21.83	**22212**	0.07	147	Y	**22212**	0.07	120	Y
Average		22082.90	27253.01	1800.00	23.41	22389.90	1.39	674.60	(8/10)	22389.90	1.39	722.00	(7/10)
Overall avg.		12694.32	15148.90	1800.00	19.34	12778.42	0.66	289.92	(46/50)	12285.86	−3.22	433.94	(42/50)

Table 8
Average results of generated number of cuts in LBBD methods.

$TL-LBBD$									
	Total number of cuts	(5) (%)	(6) (%)	(7) (%)	(8) (%)	(9) (%)	(10) (%)	(11) (%)	(12) (%)
Set I	1621	46.02	5.98	7.46	7.59	7.46	8.64	13.20	3.64
Set II	2575	48.39	11.61	12.43	6.29	12.43	2.64	4.78	1.44
$CL-LBBD$									
	Total number of cuts	(13) (%)	(14) (%)	(15) (%)	(16) (%)	(17) (%)			
Set I	565	16.99	53.45	8.67	7.96	12.92			
Set II	1257	22.75	49.32	18.22	3.34	6.36			

Table 9
Revenues of jobs in the illustrative example.

Job j	1	2	3	4	5	6	7	8	9	10
u_j	156	172	111	165	140	182	129	132	186	147

Table 10
Processing times of jobs on each machine in the illustrative example.

p_{ij}	Job j									
Machine i	1	2	3	4	5	6	7	8	9	10
M1	45	65	58	38	75	57	35	54	63	63
M2	52	64	52	45	80	42	53	38	62	80
M3	67	53	72	77	36	43	56	51	31	44

Table 11
Setup times of jobs on machine M1 in the illustrative example.

s_{1jk}	Job k										
Job j	0	1	2	3	4	5	6	7	8	9	10
0	0	15	14	16	17	15	15	14	13	16	13
1	0	0	16	15	12	13	10	13	14	14	13
2	0	16	0	15	14	14	17	14	14	15	14
3	0	13	13	0	16	11	15	16	12	11	12
4	0	13	14	12	0	15	13	10	14	15	14
5	0	13	14	11	12	0	13	15	12	11	12
6	0	15	12	12	13	12	0	13	15	14	13
7	0	14	12	12	13	12	10	0	15	15	13
8	0	16	11	13	14	14	12	12	0	13	14
9	0	15	11	12	13	12	11	11	13	0	13
10	0	12	15	11	15	11	14	15	11	12	0

Table 12
Setup times of jobs on machine M2 in the illustrative example.

s_{2jk}	Job k										
Job j	0	1	2	3	4	5	6	7	8	9	10
0	0	12	15	14	13	14	12	17	15	15	11
1	0	0	14	14	13	13	13	16	14	14	17
2	0	14	0	14	13	16	14	12	10	13	13
3	0	12	13	0	17	15	12	15	14	17	14
4	0	11	15	12	0	14	16	15	13	13	13
5	0	15	11	14	16	0	13	18	17	16	14
6	0	11	14	12	12	14	0	15	14	16	12
7	0	16	12	14	15	13	15	0	12	12	17
8	0	14	14	12	13	15	14	12	0	13	16
9	0	14	12	11	13	12	13	12	12	0	15
10	0	12	16	15	16	19	16	11	13	13	0

since the uncertainty is quite common in the real-world applications, it would be interesting to explore the problem with uncertainty in the processing times and the sequence and machine-dependent setup times. Fourthly, other exact approaches which take the advantages of the TL-LBBD method or the similar decision-partitioning scheme could be further discussed.

CRediT authorship contribution statement

Shijin Wang: Conceptualization, Methodology, Software, Writing, Funding acquisition. **Ruochen Wu:** Methodology, Software, Writing. **Feng Chu:** Visualization, Investigation. **Jianbo Yu:** Visualization, Investigation.

Data availability

We have shared the link to the data/code in paper.

Appendix

The parameters of the illustrative example in Section 3.1 are shown in Tables 9–13, where “0” represents the dummy job.

Table 13
Setup times of jobs on machine M3 in the illustrative example.

s_{3jk}	Job k										
Job j	0	1	2	3	4	5	6	7	8	9	10
0	0	14	16	14	17	18	14	13	14	15	14
1	0	0	16	13	15	15	15	14	14	14	15
2	0	12	0	11	13	14	13	13	15	15	14
3	0	16	12	0	12	14	12	12	15	15	16
4	0	14	13	13	0	11	11	10	13	13	14
5	0	15	13	13	11	0	12	12	13	13	15
6	0	15	13	10	13	14	0	10	14	13	15
7	0	14	13	12	14	15	11	0	13	13	14
8	0	17	13	12	15	15	12	13	0	10	17
9	0	12	15	15	13	13	15	14	17	0	12
10	0	10	16	13	11	11	11	11	14	14	0

References

Allahverdi, A. (2015). The third comprehensive survey on scheduling problems with setup times/costs. *European Journal of Operational Research*, *246*(2), 345–378.

Bektur, G., & Saraç, T. (2019). A mathematical model and heuristic algorithms for an unrelated parallel machine scheduling problem with sequence-dependent setup times, machine eligibility restrictions and a common server. *Computers & Operations Research*, *103*, 46–63.

Bitar, A., Dauzère-Pérès, S., & Yugma, C. (2021). Unrelated parallel machine scheduling with new criteria: Complexity and models. *Computers & Operations Research*, *132*, Article 105291.

Bruni, M., Khodaparasti, S., & Demeulemeester, E. (2020). The distributionally robust machine scheduling problem with job selection and sequence-dependent setup times. *Computers & Operations Research*, *123*, Article 105017.

Carvalho, D. M., & Nascimento, M. C. (2022). Hybrid matheuristics to solve the integrated lot sizing and scheduling problem on parallel machines with sequence-dependent and non-triangular setup. *European Journal of Operational Research*, *296*(1), 158–173.

Denton, B. T., Miller, A. J., Balasubramanian, H. J., & Huschka, T. R. (2010). Optimal allocation of surgery blocks to operating rooms under uncertainty. *Operations Research*, *58*, 802–816.

Ekici, A., Elyasi, M., Özener, O. Ö., & Sarıkaya, M. B. (2019). An application of unrelated parallel machine scheduling with sequence-dependent setups at Vestel Electronics. *Computers & Operations Research*, *111*, 130–140.

Emami, S., Moslehi, G., & Sabbagh, M. (2017). A benders decomposition approach for order acceptance and scheduling problem: A robust optimization approach. *Computational & Applied Mathematics*, *36*(4), 1471–1515.

Emami, S., Sabbagh, M., & Moslehi, G. (2015). A Lagrangian relaxation algorithm for order acceptance and scheduling problem: A globalised robust optimisation approach. *International Journal of Computer Integrated Manufacturing*, *29*(5), 535–560.

Fanjul-Peyro, L., Ruiz, R., & Perea, F. (2019). Reformulations and an exact algorithm for unrelated parallel machine scheduling problems with setup times. *Computers & Operations Research*, *101*, 173–182.

Gedik, R., Rainwater, C., Nachtmann, H., & Pohl, E. A. (2016). Analysis of a parallel machine scheduling problem with sequence dependent setup times and job availability intervals. *European Journal of Operational Research*, *251*(2), 640–650.

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

Hooker, J. N. (1994). Logic-based methods for optimization. In *Principles and Practice of Constraint Programming* (pp. 336–349). Berlin, Heidelberg: Springer.

Hooker, J. N. (2007). Planning and scheduling by logic-based benders decomposition. *Operations Research*, *55*(3), 588–602.

Hooker, J. N., & Ottosson, G. (2003). Logic-based benders decomposition. *Mathematical Programming*, *96*(1), 33–60.

Huo, Y., & Zhao, H. (2015). Total completion time minimization on multiple machines subject to machine availability and makespan constraints. *European Journal of Operational Research*, *243*(2), 547–554.

Huo, Y., & Zhao, H. (2018). Two machine scheduling subject to arbitrary machine availability constraint. *Omega*, *76*, 128–136.

Ji, M., Wang, J.-Y., & Lee, W.-C. (2013). Minimizing resource consumption on uniform parallel machines with a bound on makespan. *Computers & Operations Research*, *40*(12), 2970–2974.

Jiang, D., Tan, J., & Li, B. (2017). Order acceptance and scheduling with batch delivery. *Computers & Industrial Engineering*, *107*, 100–104.

Kaabi, J., & Harrath, Y. (2019). Scheduling on uniform parallel machines with periodic unavailability constraints. *International Journal of Production Research*, *57*(1), 216–227.

Kim, J., & Kim, H.-J. (2020). Parallel machine scheduling with multiple processing alternatives and sequence-dependent setup times. *International Journal of Production Research*, *59*(18), 5438–5453.

Kim, H.-J., & Lee, J.-H. (2021). Scheduling uniform parallel dedicated machines with job splitting, sequence-dependent setup times, and multiple servers. *Computers & Operations Research*, *126*, Article 105115.

Kong, M., Pei, J., Liu, X., Lai, P.-C., & Pardalos, P. M. (2020). Green manufacturing: Order acceptance and scheduling subject to the budgets of energy consumption and machine launch. *Journal of Cleaner Production*, *248*, Article 119300.

Li, Y., Côté, J.-F., Callegari-Coelho, L., & Wu, P. (2022). Novel formulations and logic-based benders decomposition for the integrated parallel machine scheduling and location problem. *INFORMS Journal on Computing*, *34*(2), 1048–1069.

Li, Y., Li, Y., Cheng, J., & Wu, P. (2022). Order assignment and scheduling for personal protective equipment production during the outbreak of epidemics. *IEEE Transactions on Automation Science and Engineering*, *19*(2), 692–708.

Liu, P., & Lu, X. (2020). New approximation algorithms for machine scheduling with rejection on single and parallel machine. *Journal of Combinatorial Optimization*, *40*(4), 929–952.

Naderi, B., & Roshanaei, V. (2020). Branch-Relax-and-Check: A tractable decomposition method for order acceptance and identical parallel machine scheduling. *European Journal of Operational Research*, *286*(3), 811–827.

Ou, J., & Zhong, X. (2017a). Bicriteria order acceptance and scheduling with consideration of fill rate. *European Journal of Operational Research*, *262*(3), 904–907.

Ou, J., & Zhong, X. (2017b). Order acceptance and scheduling with consideration of service level. *Annals of Operations Research*, *248*, 429–447.

Ou, J., Zhong, X., & Qi, X. (2016). Scheduling parallel machines with inclusive processing set restrictions and job rejection. *Naval Research Logistics*, *63*(8), 667–681.

Ou, J., Zhong, X., & Wang, G. (2015). An improved heuristic for parallel machine scheduling with rejection. *European Journal of Operational Research*, *241*(3), 653–661.

Pinedo, M. L. (2016). *Scheduling: Theory, algorithms, and systems*. Springer International Publishing.

Rauchecker, G., & Schryen, G. (2019a). An exact branch-and-price algorithm for scheduling rescue units during disaster response. *European Journal of Operational Research*, *272*(1), 352–363.

Rauchecker, G., & Schryen, G. (2019b). Using high performance computing for unrelated parallel machine scheduling with sequence-dependent setup times: Development and computational evaluation of a parallel branch-and-price algorithm. *Computers & Operations Research*, *104*, 338–357.

Roshanaei, V., Luong, C., Aleman, D. M., & Urbach, D. (2017). Propagating logic-based Benders' decomposition approaches for distributed operating room scheduling. *European Journal of Operational Research*, *257*(2), 439–455.

Shabtay, D., Gaspar, N., & Kaspi, M. (2013). A survey on offline scheduling with rejection. *Journal of Scheduling*, *16*(1), 3–28.

Slotnick, S. A. (2011). Order acceptance and scheduling: A taxonomy and review. *European Journal of Operational Research*, *212*(1), 1–11.

Tarhan, İ., & Oğuz, C. (2021). Generalized order acceptance and scheduling problem with batch delivery: Models and metaheuristics. *Computers & Operations Research*, *134*, Article 105414.

Tran, T. T., Araujo, A., & Beck, J. C. (2016). Decomposition methods for the parallel machine scheduling problem with setups. *INFORMS Journal on Computing*, *28*(1), 83–95.

Wang, X., Huang, G., Hu, X., & Cheng, T. C. E. (2015). Order acceptance and scheduling on two identical parallel machines. *Journal of the Operational Research Society*, *66*(10), 1755–1767.

Wang, S., Wu, R., Chu, F., & Yu, J. (2022). Unrelated parallel machine scheduling problem with special controllable processing times and setups. *Computers & Operations Research*, *148*, Article 105990.

Wang, P.-S., Yang, T., & Yu, L.-C. (2018). Lean-pull strategy for order scheduling problem in a multi-site semiconductor crystal ingot-pulling manufacturing company. *Computers & Industrial Engineering*, *125*, 545–562.

Wang, S., & Ye, B. (2019). Exact methods for order acceptance and scheduling on unrelated parallel machines. *Computers & Operations Research*, *104*, 159–173.

Wang, D., Yin, Y., & Cheng, T. (2018). Parallel-machine rescheduling with job unavailability and rejection. *Omega*, *81*, 246–260.

Wu, G.-H., Cheng, C.-Y., Yang, H.-I., & Chena, C.-T. (2018). An improved water flow-like algorithm for order acceptance and scheduling with identical parallel machines. *Applied Soft Computing*, *71*, 1072–1084.

Yepes-Borrero, J. C., Perea, F., Ruiz, R., & Villa, F. (2021). Bi-objective parallel machine scheduling with additional resources during setups. *European Journal of Operational Research*, *292*(2), 443–455.

Zhang, C., Li, Y., Cao, J., Yang, Z., & Coelho, L. C. (2022). Exact and matheuristic methods for the parallel machine scheduling and location problem with delivery time and due date. *Computers & Operations Research*, *147*, Article 105936.